

Linux

ОТ НОВИЧКА К ПРОФЕССИОНАЛУ



- Системы инициализации SysV, Upstart и Systemd
- Варианты загрузки Linux и управление загрузкой
- Работа с файловой системой и устройствами в Linux
- Файловая система ext4, UUID накопителей, загрузчик GRUB2
- Настройка сети, Интернета и популярных серверов Apache, ProFTPD, Samba, BIND и др.
- Linux как контроллер домена (Samba4)
- Настройка VPN-соединения, выбор VPN-провайдера, настройка VPN-сервера
- Выбор VPS/VDS-провайдера
- Системы виртуализации OpenVZ, Virtuozzo
- Программные системы хранения данных с резервированием

6-е издание



Материалы
на www.bhv.ru

**Наиболее
полное
руководство**

В ПОДЛИННИКЕ®

Денис Колисниченко

Linux

ОТ НОВИЧКА К ПРОФЕССИОНАЛУ

6-е издание

Санкт-Петербург

«БХВ-Петербург»

2018

УДК 004.451
ББК 32.973.26-018.2
К60

Колисниченко Д. Н.

К60 Linux. От новичка к профессионалу. — 6-е изд., перераб. и доп. — СПб.: БХВ-Петербург, 2018. — 672 с.: ил. — (В подлиннике)

ISBN 978-5-9775-3943-2

Даны ответы на все вопросы, возникающие при работе с Linux: от установки и настройки этой ОС до настройки сервера на базе Linux. Материал книги максимально охватывает все сферы применения Linux от запуска Windows-игр под управлением Linux до настройки собственного Web-сервера. Также рассмотрены: вход в систему, работа с файловой системой, использование графического интерфейса, установка программного обеспечения, настройка сети и Интернета, работа в Интернете, средства безопасности, резервное копирование, защита от вирусов и другие вопросы. Материал ориентирован на последние версии дистрибутивов Fedora, openSUSE, Slackware, Ubuntu.

В шестом издании описаны виртуальные частные сети, виртуальные серверы, настройка VPN-соединения и VPN-сервера, выбор VPN-провайдера, системы виртуализации OpenVZ и Virtuozzo, программные системы хранения данных с резервированием.

На сайте издательства находятся дополнительные главы в PDF-файлах и видеоуроки.

Для широкого круга пользователей Linux

УДК 004.451
ББК 32.973.26-018.2

Группа подготовки издания:

Руководитель проекта	<i>Евгений Рыбаков</i>
Зав. редакцией	<i>Екатерина Капалыгина</i>
Компьютерная верстка	<i>Ольги Сергиенко</i>
Дизайн серии	<i>Инны Тачиной</i>
Оформление обложки	<i>Марины Дамбиевой</i>

Подписано в печать 30.03.18.
Формат 70х100^{1/16}. Печать офсетная. Уел. печ. л. 54,18.
Тираж 1500 экз. Заказ № 6273.
"БХВ-Петербург", 191036, Санкт-Петербург, Гончарная ул., 20.
Отпечатано с готового оригинал-макета
ООО «Принт-М», 142300, М.О., г.Чехов, ул. Полиграфистов, д.1

ISBN 978-5-9775-3943-2

©ООО "БХВ", 2018
© Оформление. ООО "БХВ-Петербург", 2018

Оглавление

Предисловие.....	17
Что нового в шестом издании?.....	17
 ЧАСТЬ I ВЫБОР И УСТАНОВКА ДИСТРИБУТИВА.....	19
 Глава 1. Выбор дистрибутива.....	21
1.1. Краткая история Linux.....	24
1.2. Какой дистрибутив лучше?.....	26
1.2.1. Red Hat и Mandrake/Mandriva/Mageia.....	27
1.2.2. Fedora.....	27
1.2.3. CentOS.....	28
1.2.4. ALT Linux.....	29
1.2.5. Debian.....	29
1.2.6. Ubuntu.....	29
1.2.7. Slackware.....	30
1.2.8. openSUSE.....	30
1.3. На каком дистрибутиве основать сервер?.....	31
 Глава 2. Особенности установки.....	32
2.1. Системные требования.....	32
2.2. Первоначальная загрузка.....	34
2.2.1. POST и загрузчики.....	34
2.2.2. Ядро Linux и его параметры.....	34
2.3. Проверка носителей.....	38
2.4. Изменение таблицы разделов.....	39
2.4.1. Разметка диска в Fedora 26.....	40
2.4.2. Разметка диска в Ubuntu 17.04.....	43
2.4.3. Разметка диска в openSUSE.....	45
2.4.4. Шифрование файловой системы.....	47
2.5. Выбор устанавливаемых пакетов программ.....	48
2.6. Выбор графической среды.....	49
2.7. Установка пароля root.....	51
2.8. Создание учетных записей пользователей.....	52
2.9. Порядок установки операционных систем.....	52

2.10. Установка Linux по сети.....	52
2.10.1. Немного о загрузке и установке по сети.....	52
2.10.2. Подготовка загрузочного сервера.....	53
Установка DHCP-сервера.....	53
Настройка TFTP-сервера.....	54
Загрузка установочного образа.....	54
2.10.3. Настройка клиента.....	55
2.11. Проблемы при установке.....	55
2.11.1. Проблема с APIC.....	55
2.11.2. Ошибка: <i>kernelpanic:VFS: Unable to mount rootfs</i>	56
2.11.3. Проблемы с некоторыми LCD-мониторами.....	56
2.11.4. Сообщение <i>Probing EDD</i> и зависание системы.....	56
2.11.5. Установка Linux на HP Mini 2133 (проблема с ACPI).....	57
2.11.6. Проблема с ACPI на Fujitsu Siemens Esprimo Mobile u9200.....	57
2.11.7. Переход в режим паники компьютера с процессором AMD64.....	57
2.11.8. Проблема с механизмом Enhanced Disk Device (EDD).....	58

ЧАСТЬ II. ОСНОВНЫЕ СВЕДЕНИЯ О LINUX.....59

Глава 3. Сразу после установки.....	61
3.1. Вход в систему и завершение работы.....	61
3.2. О графическом интерфейсе Linux.....	64
3.2.1. GNOME и KDE.....	64
3.2.2. Установка альтернативного графического интерфейса.....	67
3.2.3. Основные элементы интерфейса GNOME.....	68
3.2.4. Знакомство с Unity.....	71
3.3. Изменение параметров графического интерфейса.....	73
3.3.1. Отключение блокировки экрана.....	73
3.3.2. Изменение способа переключения языков ввода.....	78
3.3.3. Изменение фона рабочего стола.....	78
3.4. «Аварийные» комбинации клавиш, использование клавиши <SysRq>.....	81
3.5. Практические приемы работы с консолью.....	82
3.5.1. Автодополнение командной строки и псевдонимы команд.....	82
3.5.2. Графические терминалы.....	83
3.5.3. Перенаправление ввода/вывода.....	83

Глава 4. Файловая система Linux.....	85
4.1. Файловые системы, поддерживаемые Linux.....	85
4.1.1. Выбор файловой системы.....	87
4.1.2. Linux и файловые системы Windows.....	88
4.1.3. Сменные носители.....	89
4.2. Особенности файловых систем Linux.....	89
4.2.1. Имена файлов в Linux.....	89
4.2.2. Файлы и устройства.....	90
4.2.3. Корневая файловая система и монтирование.....	90
4.2.4. Стандартные каталоги Linux.....	91
4.3. Внутреннее строение файловой системы.....	92
4.4. Команды для работы с файлами и каталогами.....	95
4.4.1. Работа с файлами.....	95
4.4.2. Работа с каталогами.....	97

4.5. Использование ссылок. Команда <i>ln</i>	99
4.5.1. Жесткие и мягкие ссылки.....	99
4.5.2. Создание ссылок.....	100
4.5.3. Определение ссылок.....	100
4.5.4. Удаление и жесткие ссылки.....	101
4.5.5. Разница между копированием и созданием жесткой ссылки.....	102
4.6. Права доступа и атрибуты файла. Команды <i>chown</i> , <i>chmod</i> и <i>chattr</i>	102
4.6.1. Права доступа к файлам и каталогам.....	102
4.6.2. Смена владельца файла.....	105
4.6.3. Специальные права доступа (SUID и SGID).....	105
4.6.4. Атрибуты файла. Запрет изменения файла.....	105
4.6.5. Команды поиска файлов: <i>find</i> , <i>which</i> и <i>locate</i>	106
4.7. Монтирование файловых систем.....	108
4.7.1. Команды <i>mount</i> и <i>umount</i>	108
4.7.2. Файлы устройств и монтирование.....	109
Жесткие диски.....	109
Приводы оптических дисков.....	111
Флешки и внешние жесткие диски.....	111
4.7.3. Опции монтирования файловых систем.....	111
4.7.4. Монтирование разделов при загрузке.....	113
4.7.5. Подробно о UUID и файле <i>/etc/fstab</i>	114
4.7.6. Монтирование флеш-дисков.....	118
4.8. Настройка журнала файловой системы <i>ext3/ext4</i>	120
4.9. Файловая система <i>ext4</i>	120
4.9.1. Сравнение <i>ext3</i> и <i>ext4</i>	121
4.9.2. Совместимость с <i>ext3</i>	122
4.9.3. Переход на <i>ext4</i>	122
4.10. Использование программы <i>fdisk</i> для разметки диска.....	123
4.11. Таблица разделов GPT.....	126
Глава 5. Командный интерпретатор <i>bash</i>.....	128
5.1. <i>bash</i> : основные сведения.....	128
5.2. Автоматизация задач с помощью <i>bash</i>	129
5.3. Привет, мир!.....	130
5.4. Использование переменных в собственных сценариях.....	130
5.5. Передача параметров сценарию.....	131
5.6. Массивы.....	132
5.7. Циклы.....	133
5.8. Условные операторы.....	134
Глава 6. Пользователи и группы.....	136
6.1. Многопользовательская система.....	136
6.2. Пользователь <i>root</i>	137
6.2.1. Полномочия пользователя <i>root</i>	137
6.2.2. Временное получение полномочий <i>root</i>	138
Команда <i>sudo</i>	138
Команда <i>su</i>	139
Команды <i>gksudo</i> и <i>kdesu</i>	140
Проблемы с <i>sudo</i> в Ubuntu и Kubuntu.....	140
Ввод серии команд <i>sudo</i>	140

6.2.3. Переход к традиционной учетной записи root.....	141
Преимущества и недостатки <i>sudo</i>	141
Традиционная учетная запись root в Ubuntu.....	143
6.3. Создание, удаление и модификация пользователей и групп стандартными средствами.....	143
6.3.1. Отдельные пользователи.....	143
6.3.2. Группы пользователей.....	146
6.4. Управление пользователями и группами с помощью графических конфигураторов.....	146
6.4.1. Конфигураторы в Fedora и Ubuntu.....	146
6.4.2. Графический конфигуратор в openSUSE.....	150
Еще о правах root и командах <i>su</i> и <i>sudo</i> применительно к openSUSE.....	153
Конфигуратор <i>Центр безопасности</i> openSUSE.....	154
6.5. Квотирование.....	157
Глава 7. Пакеты и управление пакетами.....	160
7.1. Способы установки программного обеспечения в Linux.....	160
7.2. Репозитории пакетов.....	162
7.3. Программы для управления пакетами.....	163
7.4. Программа <i>rpm</i> (все Red Hat-совместимые дистрибутивы).....	164
7.5. Программа <i>urpmi</i>	165
7.5.1. Установка пакетов.....	165
7.5.2. Обновление и удаление пакетов.....	166
7.5.3. Поиск пакета. Получение информации о пакете.....	166
7.6. Программа <i>yum</i>	166
7.6.1. Использование <i>yum</i>	166
7.6.2. Управление источниками пакетов.....	169
7.6.3. Установка пакетов через прокси-сервер.....	170
7.6.4. Плагины для <i>yum</i>	171
7.7. Менеджер пакетов <i>dnf</i>	171
7.8. Программы <i>dpkg</i> и <i>apt-get</i> : установка пакетов в Debian/Ubuntu.....	173
7.8.1. Программа <i>dpkg</i>	173
7.8.2. Программа <i>apt-get</i> (<i>apt</i>).....	175
7.8.3. Установка RPM-пакетов в Debian/Ubuntu.....	177
7.8.4. Подключение репозитория Medibuntu.....	177
7.8.5. Графические менеджеры в Debian/Ubuntu.....	177
7.8.6. Волшебная команда <i>update</i>	179
7.9. Установка пакетов в Slackware.....	179
7.9.1. Управление пакетами.....	181
Программа установки пакетов <i>installpkg</i>	182
Программа удаления пакетов <i>removepkg</i>	183
Программа обновления пакетов <i>upgradepkg</i>	184
7.9.2. Нет нужного пакета: вам поможет программа <i>rpm2tgz</i>	184
7.9.3. Программа <i>slackpkg</i> : установка пакетов из Интернета.....	184
7.10. Установка программ в openSUSE.....	186
7.10.1. Менеджер пакетов <i>zypper</i>	186
7.10.2. Графический менеджер пакетов openSUSE.....	188

ЧАСТЬ III. НАСТРОЙКА СЕТИ И ИНТЕРНЕТА.....	191
Глава 8. Настройка локальной сети.....	193
8.1. Локальная сеть с использованием технологии Fast Ethernet.....	193
8.2. Файлы конфигурации сети в Linux.....	196
8.3. Об именах сетевых интерфейсов.....	197
8.4. Настройка сети с помощью конфигуратора nm-connection-editor.....	200
8.5. Конфигуратор netconfig в Slackware.....	203
8.6. Утилиты для диагностики соединения.....	204
8.7. Для фанатов, или настройка сети вручную.....	208
8.7.1. Конфигурационные файлы Fedora/CentOS.....	209
8.7.2. Конфигурационные файлы openSUSE.....	211
8.7.3. Конфигурационные файлы Debian/Ubuntu.....	212
8.7.4. Команда <i>hostnamectl</i>	213
8.7.5. Команда <i>mii-tool</i>	213
8.8. Еще несколько слов о настройке сети.....	214
Глава 9. Настройка соединения Wi-Fi.....	216
9.1. Настройка беспроводного соединения с помощью NetworkManager.....	216
9.2. Что делать, если сети нет в списке?.....	220
9.3. Точка доступа Wi-Fi на смартфоне.....	221
Глава 10. Настройка VPN-соединения.....	223
10.1. Вкратце о выборе VPN-сервера и тарифного плана.....	223
10.2. Настройка VPN-подключения.....	225
Глава 11. Объединение интернет-каналов.....	231
11.1. Цели и средства решения задачи.....	231
11.2. Простой способ со статической маршрутизацией.....	231
11.3. Сложный способ с гибкой настройкой отказоустойчивости.....	234
ЧАСТЬ IV. LINUX ДОМА И В ОФИСЕ.....	239
Глава 12. Поддержка форматов мультимедиа.....	241
12.1. Что такое кодеки и почему их нет в Linux?.....	241
12.2. Настройка дистрибутива Fedora 25-26.....	241
12.3. Установка кодеков в openSUSE.....	242
12.4. Установка кодеков в Ubuntu 16.04-17.04.....	246
12.5. Домашний медиацентр на основе openELEC.....	247
12.5.1. Выбор дистрибутива.....	247
12.5.2. Установка дистрибутива.....	248
12.5.3. Настройка и использование.....	251
12.5.4. Удаленный доступ.....	258
12.5.5. А где же консоль?.....	258
12.5.6. Ложки дегтя.....	258
Глава 13. Графическая подсистема.....	259
13.1. Настройка X.Org в современных дистрибутивах.....	259
13.2. Конфигурационный файл X.Org.....	260

13.3. Синтаксис файла <code>xorg.conf</code>	261
13.4. Установка проприетарных драйверов NVIDIA в Fedora 21-26.....	267
Глава 14. Офисные пакеты.....	272
14.1. Выбор офисного пакета.....	272
14.1.1. LibreOffice.....	272
14.1.2. Calligra Suite.....	274
14.1.3. Kingsoft Office.....	275
14.2. Кроссплатформенная совместимость.....	276
14.3. Вкратце об OpenOffice.org.....	277
Глава 15. Графический редактор GIMP.....	278
15.1. Начало работы.....	278
15.2. Обработка фотографий.....	280
15.2.1. Изменение размера (масштабирование).....	280
15.2.2. Вращение.....	282
15.2.3. Кадрирование (обрезка).....	283
15.2.4. Инструмент Размывание-Резкость.....	285
15.3. Работа в GIMP с помощью скриптов.....	286
15.4. Windows-версия GIMP.....	286
Глава 16. Лазерные диски и программы для их «прожига».....	289
16.1. Что нужно для записи CD и DVD?.....	289
16.2. Отдельно о DVD.....	290
16.2.1. История создания DVD.....	290
16.2.2. Преимущества и недостатки DVD.....	291
16.2.3. Форматы и маркировка DVD-дисков.....	293
16.2.4. Регионы DVD-Video.....	295
16.2.5. Некоторые рекомендации относительно DVD.....	296
16.3. Программа K3b.....	297
16.4. Программа Brasero.....	306
16.5. Запись CD/DVD из консоли.....	309
16.6. Чтение «битых» компакт-дисков.....	309
Глава 17. Популярные программы для работы с Интернетом.....	311
17.1. Браузер Firefox.....	311
17.2. Браузер Chromium.....	314
17.3. Почтовый клиент.....	316
17.4. Skype.....	316
17.5. FTP-клиенты.....	319
17.6. P2P-клиенты.....	321
Глава 18. Виртуальная машина VirtualBox.....	323
18.1. Зачем нужна виртуальная машина?.....	323
18.2. Установка эмулятора VirtualBox.....	324
18.3. Создание новой виртуальной машины.....	325
18.4. Изменение параметров виртуальной машины.....	330
18.4.1. Общие параметры.....	330
18.4.2. Раздел Система.....	330
18.4.3. Виртуальные жесткие диски.....	330

18.4.4. А нужен ли звук?	332
18.4.5. Параметры сети	334
18.4.6. Последовательные порты	335
18.5. Запуск виртуальной машины и установка гостевой операционной системы	336

Глава 19. Эмулятор Wine: запуск Windows-игр в Linux..... 337

19.1. Эмуляторы, эмуляторы	337
19.2. Установка Wine	338
19.3. Настройка Wine и прозрачного запуска Windows-приложений	340
19.4. Использование Wine	342

ЧАСТЬ V. СИСТЕМНЫЕ ТРЮКИ, ИЛИ LINUX ИЗНУТРИ..... 349

Глава 20. Ядро..... 351

20.1. Процесс загрузки ядра	351
20.2. Параметры ядра	357
20.3. Компиляция ядра в дистрибутиве Ubuntu	361
20.3.1. Установка дополнительных пакетов	362
20.3.2. Загрузка исходных текстов ядра	362
20.3.3. Настройка ядра	363
20.3.4. Компиляция ядра	366
20.4. RT-ядро	369
20.5. Особенности компиляции ядра в других дистрибутивах Linux	370

Глава 21. Загрузчики Linux..... 371

21.1. Основные загрузчики	371
21.2. Конфигурационные файлы GRUB и GRUB2	372
21.2.1. Конфигурационный файл GRUB	372
21.2.2. Конфигурационный файл GRUB2	374
21.3. Команды установки загрузчиков	377
21.4. Установка собственного фона загрузчиков GRUB и GRUB2	377
21.5. Постоянные имена устройств	378
21.6. Восстановление загрузчика GRUB/GRUB2	379
21.7. Загрузка с ISO-образов	380
21.8. Установка пароля загрузчика	380
21.8.1. Загрузчик GRUB	381
21.8.2. Загрузчик GRUB2	383

Глава 22. Системы инициализации..... 385

22.1. Начальная загрузка Linux	385
22.2. Система инициализации init	387
22.2.1. Команда <i>init</i>	389
22.2.2. Команда <i>service</i>	389
22.2.3. Редакторы уровней запуска	390
22.2.4. Параллельная загрузка сервисов, или как сделать старый init быстрее	390
22.3. Система инициализации systemd	391
22.3.1. Идеальная система инициализации	391
22.3.2. systemd — основные понятия	392
22.3.3. Основные особенности systemd	393

22.3.4. Сравнение <code>init</code> , <code>upstart</code> и <code>systemd</code>	394
22.3.5. Немного практики.....	396
22.3.6. Команды системного администратора.....	399
22.4. Система инициализации Slackware.....	401
Глава 23. Процессы.....	404
23.1. Аварийное завершение процесса.....	404
23.2. Программа <code>top</code> : кто больше всех расходует процессорное время?.....	406
23.3. Изменение приоритета процесса.....	408
Глава 24. Псевдофайловые системы <code>sysfs</code> и <code>proc</code>.....	409
24.1. Виртуальная файловая система <code>sysfs</code>	409
24.2. Виртуальная файловая система <code>proc</code>	410
24.2.1. Информационные файлы.....	410
24.2.2. Файлы, позволяющие изменять параметры ядра.....	411
24.2.3. Файлы, изменяющие параметры сети.....	412
24.2.4. Файлы, изменяющие параметры виртуальной памяти.....	412
24.2.5. Файлы, позволяющие изменить параметры файловых систем.....	413
24.3. Сохранение произведенных изменений.....	413
Глава 25. Команды Linux, о которых нужно знать каждому линуксоиду.....	414
25.1. Общие команды.....	414
25.1.1. Команда <code>arch</code> — вывод архитектуры компьютера.....	414
25.1.2. Команда <code>clear</code> — очистка экрана.....	414
25.1.3. Команда <code>date</code>	414
25.1.4. Команда <code>echo</code>	415
25.1.5. Команда <code>exit</code> — выход из системы.....	415
25.1.6. Команда <code>man</code> — вывод справки.....	415
25.1.7. Команда <code>passwd</code> — изменение пароля.....	415
25.1.8. Команда <code>startx</code> — запуск графического интерфейса X.Org.....	415
25.1.9. Команда <code>uptime</code> — информация о работе системы.....	416
25.1.10. Команда <code>users</code> — информация о пользователях.....	416
25.1.11. Команды <code>w</code> , <code>who</code> и <code>whoami</code> — информация о пользователях.....	416
25.1.12. Команда <code>xf86config</code> — настройка графической подсистемы.....	417
25.2. Команды для работы с текстом.....	417
25.2.1. Команды <code>diff</code> и <code>cmp</code> — сравнение файлов.....	417
25.2.2. Команды <code>grep</code> и <code>egrep</code> — текстовый фильтр.....	418
25.2.3. Команды <code>more</code> и <code>less</code> — постраничный вывод.....	419
25.2.4. Команды <code>head</code> и <code>tail</code> — вывод начала и хвоста файла.....	419
25.2.5. Команда <code>wc</code> — подсчет слов в файле.....	420
25.2.6. Команды <code>vi</code> , <code>nano</code> , <code>ee</code> , <code>mcedit</code> , <code>pico</code> — текстовые редакторы.....	420
25.3. Команды для работы с Интернетом.....	424
25.3.1. Команда <code>ftp</code> — стандартный FTP-клиент.....	424
25.3.2. Команда <code>lynx</code> — текстовый браузер.....	425
25.3.3. Команда <code>mail</code> — чтение почты и отправка сообщений.....	425
25.4. Команды системного администратора.....	426
25.4.1. Команды <code>free</code> и <code>df</code> — информация о системных ресурсах.....	426
25.4.2. Команда <code>md5sum</code> — вычисление контрольного кода MD5.....	426
25.4.3. Команды <code>ssh</code> и <code>telnet</code> — удаленный вход в систему.....	426

Глава 26. Конфигурационные файлы Linux.....	427
26.1. Каталог /etc.....	427
26.2. Каталог /etc/NetworkManager.....	428
26.3. Каталог /etc/abrt.....	429
26.4. Каталог /etc/alsa.....	429
26.5. Каталоги /etc/audit и /etc/audisp.....	429
26.6. Каталог /etc/avahi — файлы конфигурации демона Avahi.....	429
26.7. Каталог /etc/blkid.....	430
26.8. Файлы конфигурации планировщиков задач.....	430
26.9. Каталог /etc/cups.....	431
26.10. Файл /etc/fonts/fonts.conf.....	433
26.11. Каталог /etc/gdm.....	433
26.12. Файлы конфигурации популярных сетевых служб.....	433
26.13. Каталог /etc/logrotate.d.....	434
26.14. Каталог /etc/mail.....	435
26.15. Каталог /etc/ntp.....	435
26.16. Каталог /etc/openldap.....	435
26.17. Каталог /etc/openvpn.....	436
26.18. Каталоги /etc/pam.d и /etc/security.....	436
26.19. Каталог /etc/ppp.....	436
26.20. Каталог /etc/rc.d.....	436
26.21. Каталог /etc/sane.d.....	436
26.22. Каталог /etc/selinux.....	437
26.23. Каталог /etc/skel.....	437
26.24. Каталог /etc/sysconfig.....	437
26.25. Каталог /etc/X11.....	438
26.26. Конфигурационные файлы yum/dnf.....	438
26.27. Основные конфигурационные файлы сети.....	438
26.28. Остальные конфигурационные файлы каталога /etc.....	438
Глава 27. Протоколирование системы.....	440
27.1. Протоколирование по-новому: journalctl.....	441
27.1.1. Установка времени.....	441
27.1.2. Просмотр и фильтрация логов.....	442
Текущая и предыдущие загрузки.....	442
Фильтр по дате.....	443
Фильтр по сервису.....	444
Фильтр по пути.....	444
Фильтр по процессу или пользователю.....	444
Просмотр сообщений ядра.....	444
Фильтр по уровню ошибки.....	444
27.1.3. Журналы в реальном времени.....	445
27.1.4. Централизованное хранение логов.....	445
27.2. Демоны syslogd и rsyslogd.....	445

ЧАСТЬ VI. LINUX НА СЕРВЕРЕ.....	449
Глава 28. Обеспечение безопасности сервера.....	451
28.1. Защита от «восстановления пароля root».....	451
28.1.1. Параметр ядра <i>single</i>	451
28.1.2. Пароль загрузчика GRUB.....	453
28.1.3. Осторожно: LiveCD.....	453
28.2. Защита от перезагрузки.....	453
28.3. Отключение учетной записи root: нестандартный метод.....	455
28.4. Отключение учетной записи root средствами KDM и GDM.....	457
28.5. Системы управления доступом.....	458
Глава 29. Модули аутентификации PAM.....	459
29.1. Каталог <i>/etc/pam.d</i>	459
29.2. Дополнительные файлы конфигурации.....	460
29.2.1. Содержимое каталога <i>/etc/security</i>	460
29.2.2. Файл <i>access.conf</i> : ограничение доступа к системе.....	461
29.2.3. Файл <i>limits.conf</i> : ограничение на используемые системные ресурсы.....	462
29.2.4. Файл <i>time.conf</i> : регистрация только в рабочее время.....	463
29.3. Список PAM-модулей.....	464
29.4. Борьба с простыми паролями.....	465
Глава 30. Оптимизация системы. Автоматизация выполнения задач.....	467
30.1. Оптимизация подкачки.....	467
30.2. Создание файла подкачки.....	468
30.3. Настройка планировщика ввода/вывода.....	469
30.4. Двухканальный режим памяти.....	470
30.5. Автоматизация выполнения задач.....	470
30.5.1. Планировщик <i>crond</i>	470
30.5.2. Планировщик <i>anacron</i>	472
30.5.3. Разовое выполнение команд — демон <i>atd</i>	473
Глава 31. Маршрутизация. Настройка брандмауэра.....	474
31.1. Таблица маршрутизации ядра. Установка маршрута по умолчанию.....	475
31.2. Изменение таблицы маршрутизации. Команда <i>route</i>	478
31.3. Включение IPv4-переадресации, или превращение компьютера в шлюз.....	481
31.4. Настройка брандмауэра.....	482
31.4.1. Цепочки и правила.....	483
31.4.2. Брандмауэр <i>iptables</i>	485
31.4.3. Шлюз своими руками.....	488
Глава 32. Безопасный удаленный доступ. OpenSSH.....	494
32.1. Протокол SSH.....	494
32.2. Использование SSH-клиента.....	495
32.3. Настройка SSH-сервера.....	495
Глава 33. Web-сервер. Связка Apache + PHP + MySQL.....	500
33.1. Самый популярный Web-сервер.....	500
33.2. Установка Web-сервера и интерпретатора PHP. Выбор версии.....	500
33.3. Тестирование настроек.....	502

33.4. Файл конфигурации Web-сервера.....	505
33.4.1. Базовая настройка.....	505
33.4.2. Самые полезные директивы файла конфигурации.....	506
33.4.3. Директивы <i>Directory</i> , <i>Limit</i> , <i>Location</i> , <i>Files</i>	507
33.5. Управление запуском сервера Apache.....	510
33.6. Оптимизация Apache.....	510
33.7. Пользовательские каталоги.....	512
33.8. Установка сервера баз данных MySQL.....	512
3 3.8.1. Установка сервера.....	512
33.8.2. Изменение пароля root и добавление пользователей.....	513
33.8.3. Запуск и останов сервера.....	514
33.8.4. Программа MySQL Administrator.....	514
33.9. Обеспечение безопасности сайта от вирусов.....	516
33.9.1. Как вирусы попадают на сайт?.....	516
33.9.2. Установка прав доступа.....	517
33.9.3. Антивирус ClamAV.....	518
33.9.4. Сценарий scanner.....	519
Глава 34. FTP-сервер.....	520
34.1. Установка FTP-сервера.....	520
34.2. Конфигурационный файл.....	521
34.3. Настройка FTP-сервера.....	525
34.4. Оптимизация FTP-сервера.....	527
34.5. Программы ftpwho и ftpcount.....	529
Глава 35. DNS-сервер.....	530
35.1. Еще раз о том, что такое DNS.....	530
35.2. Кэширующий сервер DNS.....	531
35.3. Полноценный DNS-сервер.....	536
35.4. Вторичный DNS-сервер.....	541
35.5. Обновление базы данных корневых серверов.....	541
Глава 36. Прокси-сервер: Squid и squidGuard.....	544
36.1. Зачем нужен прокси-сервер в локальной сети?.....	544
36.2. Базовая настройка Squid.....	544
36.3. Практические примеры.....	546
36.3.1. Управление доступом.....	546
36.3.2. Создание «черного» списка адресов.....	547
36.3.3. Отказ от баннеров.....	547
36.4. Управление прокси-сервером squid.....	547
36.5. Настройка клиентов.....	548
36.6. Прозрачный прокси-сервер.....	548
36.7. squidGuard — ваше дополнительное «оружие».....	549
Глава 37. Почтовый сервер.....	553
37.1. Выбор почтового сервера.....	553
37.2. Настройка MTA Exim.....	554
37.3. Настройка аутентификации SMTP.....	556
37.4. Настройка демона SASL.....	557

Глава 38. Сервис Samba.....	558
38.1. Установка Samba.....	558
38.2. Базовая настройка Samba.....	558
38.3. Настройка общих ресурсов.....	560
38.4. Просмотр ресурсов Windows-сети.....	561
38.5. Оптимизация Samba.....	561
38.6. Samba и Active Directory.....	563
38.7. Samba в качестве контроллера домена.....	566
Глава 39. Поддержка КАШ.....	570
39.1. Аппаратные RAID-массивы.....	570
39.2. Программные RAID-массивы.....	573
39.3. Создание программных массивов.....	574
39.4. RAID-массив только для данных.....	575
39.5. Сбой и его имитация.....	576
Глава 40. Программные системы хранения данных.....	577
40.1. Аппаратные хранилища с резервированием.....	577
40.2. Программные хранилища с резервированием.....	579
40.3. Распределенная система хранения данных Ceph.....	581
40.4. Дополнительные материалы.....	582
Глава 41. Средства резервного копирования. Создание ISO-образа диска.....	583
41.1. Необходимость в «живой» резервной копии.....	583
41.2. Средства клонирования Linux.....	584
41.3. Clonezilla.....	585
41.4. Linux Live.....	594
Глава 42. Шифрование файловой системы.....	595
42.1. Шифрование папки.....	595
42.2. Храним пароль на флешке.....	597
ЧАСТЬ VII. ВИРТУАЛЬНЫЕ СЕРВЕРЫ.....	599
Глава 43. А нужен ли физический сервер?.....	601
43.1. Физический или виртуальный?.....	601
43.1.1. Стоимость физического сервера.....	601
43.1.2. Необходимость в аппаратном сервере.....	602
43.1.3. Про VPS, VDS и спекулянтов.....	603
43.1.4. Стоимость VDS.....	605
43.1.5. Физический сервер vs VDS.....	606
43.1.6. Стоимость владения физическим сервером.....	607
43.1.7. Выводы.....	608
43.2. Виртуальный тест-драйв.....	608
43.2.1. Джинно.....	609
О ценах.....	609
Создание сервера.....	610
Тестирование.....	611
Выводы.....	613
43.2.2. Спринтхост.....	615
О ценах.....	615

Создание сервера.....	616
Тестирование.....	617
Выводы.....	619
43.2.3. Макхост.....	620
О ценах.....	620
Создание сервера.....	620
Тестирование.....	621
Выводы.....	623
43.2.4. UltraVDS.....	623
О ценах.....	623
Создание сервера.....	623
Тестирование.....	626
Выводы.....	627
43.2.5. 1cloud.....	628
О ценах.....	628
Тестирование.....	629
Выводы.....	631
43.3. Заключение.....	632
Глава 44. Сервер виртуализации OpenVZ.....	633
44.1. Способы виртуализации.....	633
44.2. Установка OpenVZ.....	635
44.3. Создание и настройка виртуального контейнера.....	637
44.4. Запуск виртуальной машины.....	638
Глава 45. Знакомство с Virtuozzo Linux.....	640
45.1. Что такое Virtuozzo?.....	640
45.2. Как это работает?.....	640
45.3. Системные требования и ограничения.....	641
45.4. Установка Virtuozzo.....	642
45.5. Выбор шаблона.....	645
45.6. Создание и настройка контейнера.....	646
45.7. Управление ресурсами контейнера.....	647
45.8. Управление контейнерами.....	649
45.9. Запуск команд и вход в гостевую операционную систему.....	650
45.10. Настройка сети.....	651
45.11. Делаем работу с Virtuozzo удобнее.....	654
Глава 46. Настройка собственного VPN-сервера.....	655
46.1. Что мы будем настраивать?.....	655
46.2. Установка OpenVPN.....	656
46.3. Настройка центра сертификации.....	656
46.4. Создание сертификата и ключей для сервера.....	657
46.5. Создание сертификата и ключей для клиента.....	658
46.6. Настройка сервера OpenVPN.....	658
46.7. Инфраструктура настройки клиентов.....	660
46.8. Настройка клиентов.....	662
Приложение. Описание электронного архива.....	664
Предметный указатель.....	665

Предисловие

Операционная система Linux уверенно осваивает наши просторы. Но в силу многообразия доступных дистрибутивов Linux, а создать и предложить сообществу свой дистрибутив может каждый «умелец», начинающий¹ пользователь, бывает, теряется при выборе дистрибутива для себя... И это понятно — у каждого дистрибутива свои особенности.

Книга, которую вы держите в руках, поможет вам пройти сложный, но интересный путь от новичка к профессиональному пользователю Linux, а именно — сориентироваться в особенностях различных дистрибутивов, выбрать для себя наиболее подходящий и научиться в нем работать.

Что нового в шестом издании?

Новые версии дистрибутивов выходят постоянно: некоторые — чаще, некоторые — реже. Пользователи Linux к этому привыкли, поэтому простой заменой в книге описаний одних версий дистрибутивов на другие никого не удивить.

Направление этого издания — всевозможные виртуальные технологии. Так, в *главе 10* мы поговорим о выборе VPN-провайдера и настроим VPN-подключение. А в *главе 46* займемся настройкой собственного VPN-сервера для защиты ваших данных, передающихся по незащищенным соединениям, таким как публичные сети Wi-Fi.

В *главе 43* мы рассмотрим выбор VPS/VDS-провайдера — провайдера виртуального сервера, узнаем, на что нужно обратить внимание при выборе такого сервера, поговорим о стоимости его аренды.

В предыдущих изданиях книги рассматривалась технология виртуализации OpenVZ. В этом издании, кроме технологии OpenVZ, которой посвящена *гла-*

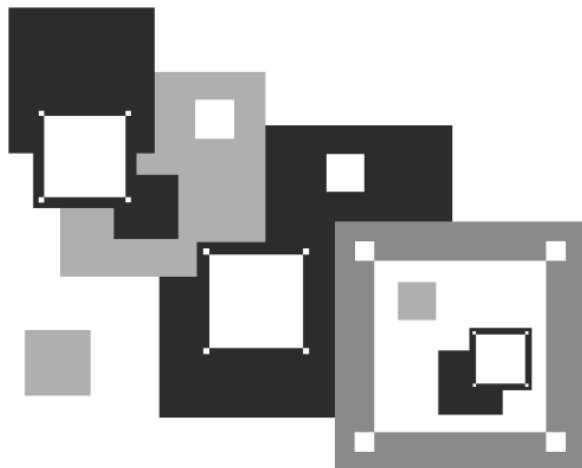
¹ Обращаясь здесь к начинающему пользователю, автор отнюдь не имеет в виду сугубого новичка, впервые подсаживающегося к компьютеру... Напротив, книга ориентирована на вполне уверенного современного пользователя Windows или Mac, по тем или иным причинам заинтересовавшегося работой в Linux.

ва 44, рассматривается еще и технология Virtuozzo — продолжение и дальнейшее развитие OpenVZ (глава 45).

Шестая часть книги — о серверном применении Linux — дополнена новой *главой 40*, рассказывающей о программных системах хранения данных с резервированием. Вы узнаете, какие программные системы хранения существуют, и в чем их преимущества перед аппаратными решениями.

Электронный архив с информацией, расширяющей и дополняющей материал «бумажной» книги, можно скачать с FTP-сервера издательства по ссылке: **<ftp://ftp.bhv.ru/9785977539432.zip>**, а также со страницы книги на сайте **www.bhv.ru**. Подробная информация об архиве приведена в *приложении*.

Вот теперь самое время приступить к чтению книги!

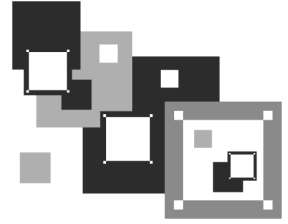


ЧАСТЬ I

Выбор и установка дистрибутива

Первая часть, как следует из ее названия, посвящена выбору и установке дистрибутива. Соответственно, в *главе 1* мы поговорим об исторических корнях Linux и выборе ее дистрибутива, а в *главе 2* — об особенностях установки этой операционной системы на компьютер.

ГЛАВА 1



Выбор дистрибутива

Прежде всего вам нужно решить, какой именно дистрибутив Linux устанавливать. В конце 1990-х годов в этом плане особого выбора пользователям не предоставлялось — скачивать дистрибутив из Интернета было дорого, а в компьютерных магазинах они встречались редко. А если и попадались, то исключительно Red Hat и появившиеся на прилавках чуть позже Black Cat и Mandrake.

Сейчас, наоборот, проблема выбора стоит перед нами в полный рост. Лет десять назад я бы отдал предпочтение отечественному дистрибутиву — например, ALT Linux. Почему? Да потому что в отечественных разработках существенное внимание уделялось локализации — была переведена на русский язык вся документация, включая страницы руководства пользователя (man pages), не говоря уже о качественной русификации графических интерфейсов GNOME и KDE. В настоящее время особой разницы между дистрибутивами по этой части нет — качество локализации зарубежных дистрибутивов не вызывает особых нареканий. Единственный дистрибутив, который до сих пор окончательно не русифицирован, — это Fedora.

Проблем с русским языком при работе в нем у вас не возникнет, но некоторые окна окажутся переведенными на русский язык не полностью, — видимо, это фирменная особенность Fedora.

Может, я предвзято отношусь к Fedora, но в доказательство своих слов приведу несколько скриншотов. Начнем с экрана загрузчика (рис. 1.1) — а вот в других дистрибутивах (в том же Ubuntu) можно выбрать русский язык прямо на этом этапе и уже не гадать, что означает та или иная команда.

Подобную картину вы увидите и при запуске LiveCD (рис. 1.2)— только английский...

Да и после установки кое-где в системных окнах можно заметить, что некоторые надписи так на русский язык и не переведены, — вот, например, как здесь: **OS Type** (рис. 1.3) или в окне описания обновлений пакетов (рис. 1.4).

Подобные небольшие «косяки» вы найдете в любом дистрибутиве, но в Fedora они встречаются чаще.

Так какой же дистрибутив выбрать? Чтобы ответить на этот вопрос, познакомимся с основными этапами развития операционной системы Linux.

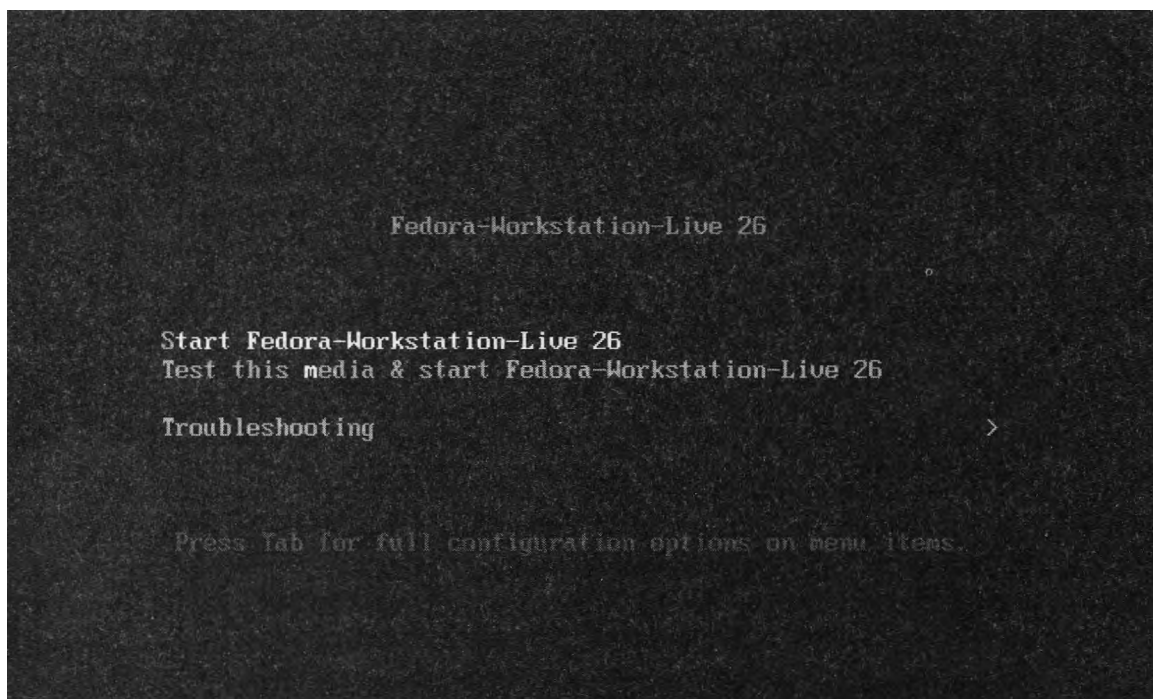


Рис. 1.1. Fedora 26: не русифицирован экран загрузчика установочного диска



Рис. 1.2. Fedora 26: сразу после запуска LiveCD (установочный образ, загруженный с официального сайта)

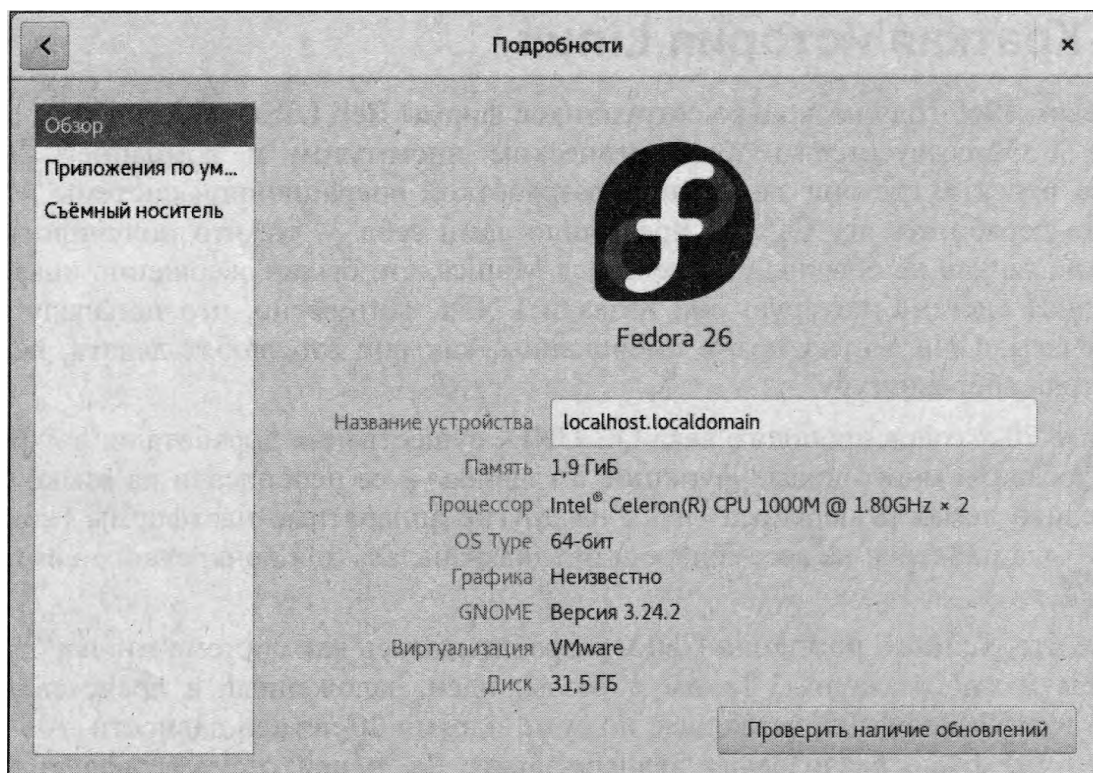


Рис. 1.3. Fedora 26: некоторые надписи не переведены на русский язык



Рис. 1.4. Fedora 26: описание обновлений пакетов

1.1. Краткая история Linux

В далеком 1969 году несколько сотрудников фирмы Bell Labs, вышедшей из совместного с Массачусетским технологическим институтом и компанией General Electric проекта, где они занимались разработкой операционной системы Multics, решили доработать эту ОС, но превзошли сами себя — то, что получилось, уже никак не тянуло на обычный апгрейд для Multics, это была совершенно новая операционная система, которую они называли UNIX. Интересно, что поначалу UNIX называлась «UNICS», но позже американцы, как они это любят делать, немного упростили аббревиатуру.

В начале 70-х годов прошлого века ОС UNIX существенно доработали: в ядро системы добавили много новых функций, а главное — ее переписали на языке C, что обеспечило легкость переноса UNIX на другие аппаратные платформы (исходная UNIX была написана на ассемблере и предназначалась для конкретного компьютера PDP-7).

Важно, что с самого рождения UNIX разрабатывалась как система многопользовательская и многозадачная. Таким образом, идеи, заложенные в представленную в 1995 году Windows 95, оказались, по сути, идеями 20-летней давности — в UNIX все это уже было реализовано давным-давно. Да, в ней отсутствовал красивый «фантик» — графический интерфейс, — но это и не главное в операционной системе.

В начале 1980-х годов появились первые персональные компьютеры фирмы IBM, однако мощности IBM PC никак не хватало для запуска UNIX, поэтому в мире персональных компьютеров десять лет царствовала операционная система DOS компании Microsoft. Но, начиная с 1990-х, ситуация изменилась — мощность «персоналок» уже позволяла запускать UNIX, и к этому времени (а прошло более 20 лет с момента появления первой ее версии) разными фирмами, университетами и отдельными энтузиастами было создано множество UNIX-подобных операционных систем (IRIX, XENIX, HP-UX, BSD, Minix и др.).

Огромное значение в развитии Linux сыграла одна из таких операционных систем — Minix, которая, собственно, полноценной системой и не являлась, а создавалась для демонстрации основных принципов и устройства реальных операционных систем. Да, она не была совершенной, но зато ее исходный код (всего 12 тысяч строк) был опубликован в книге А. Таненбаума «Операционные системы», — именно эту книгу и купил живший тогда в Хельсинки программист Линус Торвальдс (Linus Torvalds).

В 1991 году Линус Торвальдс установил на свой компьютер ОС Minix, но та не оправдала его ожиданий, поэтому он принял решение несколько ее переработать — ведь исходные коды вместе с комментариями были под рукой. Сначала Торвальдс просто переписал программу эмуляции терминала, а затем так углубился в доработку Minix, что вышел фактически на создание собственной операционной системы. В результате 25 августа 1991 года ОС Linux (версия 0.01) и родилась. Конечно, это была не та Linux, что мы имеем сейчас, но уже тогда она оказалась лучше Minix, поскольку в ней запускались командный интерпретатор

bash и компилятор gcc. Сообщение о создании новой операционной системы Торвальдс поместил в группу новостей comp.os.minix, там же всем желающим предлагалось ее протестировать.

С этого и началось интенсивное развитие Linux, а к ее разработке в помощь Торвальдсу подключились энтузиасты со всего мира, — ведь ничто так не сокращает расстояния, как Интернет. С момента появления версии 0.01, которой еще нельзя было пользоваться практически, до создания (вышла в апреле 1994 года) версии 1.0, пригодной для обычных пользователей, а не только лишь для увлеченных программистов, прошло почти три года. Версия обладала поддержкой сети на основе протокола TCP/IP, а также графическим интерфейсом X Window (появившимся в Linux еще в 1992 году одновременно с поддержкой TCP/IP).

Сначала версии Linux распространялись на обыкновенных дискетах. Комплект состоял из двух дискет: одна содержала ядро, а другая — корневую файловую систему и необходимые программы. Установить подобную версию Linux на компьютер мог только специалист. Первые же *дистрибутивы* — комплекты, помимо того же ядра и корневой файловой системы, включающие также программу (как правило, на отдельной дискете) для установки всего этого на компьютер, появились в 1992 году — их начали выпускать отдельные энтузиасты или группы энтузиастов (каждый дистрибутив, естественно, под собственным именем). Впрочем, их дистрибутивы на тот момент отличались друг от друга лишь названием и программой установки, но в дальнейшем различия между дистрибутивами стали более существенными.

Самый первый дистрибутив, созданный в Манчестерском компьютерном центре (Manchester Computing Centre, MCC), вышел в начале 1992 года и назывался MCC Interim Linux. Чуть позже появился дистрибутив TAMU, разработанный в Техасском университете. Настоящий прорыв произвел дистрибутив SLS, выпущенный в октябре 1992 года, поскольку именно он содержал поддержку TCP/IP и систему X Window. Впоследствии этот дистрибутив бурно развивался и постепенно трансформировался в один из самых популярных современных дистрибутивов — Slackware.

Со временем дистрибутивы разрослись до таких размеров, что распространять их на дискетах стало невозможно, — они занимали объем 50-70 Мбайт. Вы можете себе представить дистрибутив на 50 дискетах? А что делать, если, скажем, дискета № 47 окажется бракованной? Впрочем, дистрибутив того времени (как, кстати, и сейчас) можно было бесплатно (если не считать стоимости трафика) скачать из Интернета. Но далеко не все могли себе позволить качать из Интернета такие объемы в режиме online (тогда online-режимом считалась работа со Всемирной паутиной, а offline — с почтой и новостями Usenet), поэтому в начале 1990-х основными носителями для распространения Linux, все же, оставались дискеты. Но как раз к тому времени лазерные компакт-диски и их приводы несколько подешевели, и компания Red Hat стала одной из первых, выпустивших свою разработку на компакт-диске. Новшество прижилось, и, начиная с середины 1990-х, дистрибутивы Linux постепенно почти полностью перекочевали на компакт-диски.

О дистрибутивах можно было бы рассказать еще очень много. Однако важно запомнить следующее:

- основные дистрибутивы — это Red Hat (сейчас существует в виде RHEL — Red Hat Enterprise Linux) и Debian, а все остальные — лишь производные от них. Так, Mandrake и ASPLinux (оба дистрибутива нынче «мертвы») произошли от Red Hat, а ALT Linux взял за основу Mandrake, Ubuntu изначально был основан на Debian. К числу современных RH-подобных дистрибутивов относятся CentOS, Fedora и openSUSE, к числу современных Debian-подобных — Ubuntu, а также его клоны и всевозможные варианты (Kubuntu, Xubuntu, Mint и т. д.);
- номер версии дистрибутива не совпадает с номером ядра — это принципиально разные вещи;
- самыми популярными дистрибутивами на сегодняшний день считаются Ubuntu и Fedora — для настольного применения, а также CentOS и Debian — для серверного.

1.2. Какой дистрибутив лучше?

Дистрибутивов сейчас так много, что порою теряешься — какой из них установить, какой лучше? Здесь мы вкратце рассмотрим сильные и слабые стороны каждого дистрибутива. Каждого, но только из числа представленных в этой книге. Дело в том, что дистрибутивов очень много, и, как уже отмечалось ранее, любой желающий может создать свой дистрибутив. Есть такие дистрибутивы, с которыми я до сих пор не работал, а есть и такие, о которых даже не слышал! Понятно, что все существующие дистрибутивы рассмотреть в одной книге невозможно, да и не нужны вам они все. Могу поспорить, что после прочтения этой книги вы установите от одного до трех дистрибутивов, а потом остановитесь на том единственном, который вам больше всех понравится.

В свое время (1998-1999 годы) я работал с Red Hat, поскольку он был более удобным, чем Slackware. Затем мне удалось раздобыть и установить Mandrake (кажется, это была его седьмая версия), и он оказался еще лучше, чем тот же Red Hat 6, хотя и являлся его клоном. Потом я еще долго пробовал разные дистрибутивы: Debian, Ubuntu, Gentoo, openSUSE.

Возможно, сейчас вам понравится один из дистрибутивов, но со временем вы перейдете на другой. Или же сейчас вам какой-то не понравится, однако с выходом его новой версии он покажется вам лучшим. Так у меня было с openSUSE — первая попавшая ко мне его версия (не помню сейчас ее номер) особо меня не впечатлила, а вот следующая оказалась очень даже приличной. Так что сейчас у меня установлено два дистрибутива: openSUSE и Denix (дистрибутив моей собственной разработки, собранный на базе Debian).

Отдельного внимания заслуживает выбор архитектуры: 32-битная или 64-битная¹. Если в вашем компьютере установлено 4 Гбайт оперативной памяти или больше,

¹ Правильно говорить «32-разрядная», но при загрузке образа обычно указывается «32-bit» или «64-bit» — читателю так будет проще ориентироваться.

вы можете выбрать 64-битную версию, иначе не вся оперативная память окажется доступной¹. Тем не менее, рискну, все же, рекомендовать к выбору 32-битную архитектуру, даже если у вас 64-разрядный процессор. Почему? Во-первых, производительность 32-битной версии на 64-разрядной машине в большинстве случаев не ниже производительности 64-битной версии — т. е. особой разницы вы не почувствуете. Во-вторых, так уж повелось, что 32-разрядные версии операционных систем Linux работают стабильнее.

1.2.1. Red Hat и Mandrake/Mandriva/Mageia

Современной настольной версии Red Hat в природе не существует вследствие того, что разработка Red Hat была в свое время разделена на две ветки: для корпоративных пользователей (Red Hat Enterprise Linux, RHEL) и для домашних пользователей и небольших компаний (Fedora). Так что, обратившись к Red Hat, вам придется остановиться на ее ветке Fedora, поскольку RHEL, ориентированный на современные дата-центры, вряд ли вам подойдет.

Когда-то я был просто в восторге от дистрибутива Mandrake, переименованного потом в Mandriva, но, к сожалению, всему приходит конец, — последний релиз этого дистрибутива вышел 28 августа 2011 года, после чего проект был закрыт. Поэтому дистрибутив Mandriva в этом издании книги мы рассматривать не станем.

Свято место пусто не бывает, и на смену Mandriva пришел его форк (ответвление) — Mageia (<http://www.mageia.org/ru>). В настоящее время выпущена уже 6-я версия этого дистрибутива, в состав которой входят графические окружения KDE Plasma Desktop, GNOME 3 Desktop и LXDE. Примечательно, что дистрибутив Mageia (и это в наше-то время!) распространяется не только на DVD, но и на простых лазерных компакт-дисках (CD), — правда, в этом случае вам будет доступна только графическая среда LXDE, что позволит использовать Mageia на весьма «древних» компьютерах.

Впрочем, подробно дистрибутив Mageia здесь рассмотрен не будет, поскольку за четыре года своего существования он так и не стал популярным. Однако, если вы фанат Mandriva, можете попробовать установить Mageia, в противном случае обратите внимание на другие дистрибутивы: Fedora или Debian — они-то уж точно никуда по прошествии времени не исчезнут.

1.2.2. Fedora

Fedora (fedoraproject.org) — вполне приличный дистрибутив. Да, в нем есть определенные недоработки, но их не больше, чем в других. Здесь мы рассматриваем

¹ 32-битные версии ОС «не видят» полные 4 Гбайт памяти и не могут их до конца использовать. Объяснение этого феномена выходит за рамки книги. Если принципиальна установка именно 32-битной версии на компьютер с объемом оперативной памяти более 4 Гбайт, необходимо задействовать так называемое PAE-ядро.

одну из самых последних на момент написания этих строк (август 2017 года) версий Fedora — 26-ю. Вот основные ее нововведения:

- ядро 4.11.18;
- инструменты разработки GCC 7, Golang 1.8 и Python 3.8;
- обновлена тема Adwaita — пользователи получают обновленный и слегка улучшенный интерфейс;
- рабочий стол GNOME 3.24;
- новый пакетный менеджер DNF, включенный в дистрибутив вместо старого менеджера YUM, — он потребляет значительно меньше памяти и быстрее работает. В общем, он так же хорош, как и apt в Ubuntu;
- виртуализация GNOME Boxes;
- добавлен отдельный образ Fedora Cloud (Облако) с инструментарием Vagrant, который может использоваться для быстрого развертывания виртуальных окружений в системах виртуализации на базе KVM и VirtualBox;
- в серверной версии по умолчанию используется файловая система XFS, хотя в версии Workstation (Рабочая станция) и Cloud (Облако) остались пока на ext4.

Обычно в каждом следующем выпуске Fedora появляются достаточно новые и экспериментальные решения. Но выпуск 26 нас таковыми не порадовал. Тот же выпуск 22 (2015-й год) был более интересен: и новый менеджер пакетов, и виртуализация GNOME, и отдельный образ Fedora Cloud, и другая файловая система в серверной версии. В версии же 26 — ничего сверхъестественного: из улучшений пользователи заметят только новую версию GNOME и новую тему оформления (которая, скорее всего, «затачивалась» под новую версию GNOME, поэтому и была обновлена).

При загрузке ISO-образа дистрибутива Fedora обратите внимание на различные его варианты:

- **Server** (Сервер) — все самое необходимое для построения сервера, при этом нет графического интерфейса (можно установить отдельно, но зачем?), по умолчанию используется файловая система XFS;
- **Workstation** (Рабочая станция) — идеален для офисных/домашних компьютеров. По умолчанию устанавливается графический интерфейс и неплохой набор программ;
- **Atomic** — платформа для вашего стека приложений Linux Docker-Kubernetes (LDK). Впрочем, для установки на обычный компьютер этот вариант не подойдет.

1.2.3. CentOS

Дистрибутив CentOS (<https://www.centos.org>) основан на дистрибутиве Red Hat Enterprise Linux и обладает схожей функциональностью. Основное его отличие в том, что он бесплатный. Так что, если вам нужен бесплатный RHEL, просто установите CentOS. Я был удивлен, но CentOS оказался весьма добротным дистрибути-

вом, — в нем наличествует все, что и должно быть. И если выбирать между Fedora и CentOS, то я бы предпочел последний.

1.2.4. ALT Linux

Еще один хороший, добротный дистрибутив — ALT Linux (www.altlinux.ru), и это не просто клон зарубежной разработки. Да, в свое время ALT Linux был основан на Mandriva, но с тех пор много воды утекло, и теперь этот дистрибутив — собственная разработка компании ALT Linux, в которой нашло применение множество ее собственных решений.

В начальных версиях дистрибутива ALT Linux «хромала» программа установки — создавать разделы для него было удобнее в сторонней программе разметки диска, а не с помощью инсталлятора ALT Linux, сейчас же с этим все в порядке, и установка ALT Linux также удобна, как и любого другого дистрибутива.

1.2.5. Debian

Debian (www.debian.org) — хороший, надежный, стабильный дистрибутив. Практически все его пакеты снабжены собственным конфигуратором `debconf`, что значительно упрощает настройку. Начиная с версии 5.0, дистрибутив содержит принципиально новую программу установки пакетов — Debian Installer, которая отличается существенно большей гибкостью по сравнению со своей предшественницей.

Debian хорош тем, что в его состав входят только уже проверенные временем пакеты, — вы не найдете здесь экспериментальных разработок и самых новых версий ядра. Именно поэтому последние версии моего дистрибутива Denix основаны на Debian — хотелось получить добротный дистрибутив, в котором будут присутствовать все необходимые мне инструменты.

1.2.6. Ubuntu

Ubuntu (www.ubuntu.com) — очень интересный дистрибутив. Любопытно, что его название в переводе с одного из африканских языков означает «человечность, гуманность по отношению к другим». По данным сайта **DistroWatch.com** Ubuntu признан самым популярным в мире дистрибутивом. Готов поспорить с этим, поскольку на территории бывшего СССР Ubuntu не очень распространен, однако в последнее время его популярность и у нас стремительно растет.

Дистрибутив основан на Debian, но отличается тем, что в состав Ubuntu включаются не только проверенные пакеты, но и новые. Разработчикам Ubuntu, кажется, удалось соблюсти баланс между стабильностью системы и новыми функциями.

Дистрибутивов Ubuntu существует целое семейство: Kubuntu, Edubuntu, Lubuntu, Mythbuntu, Xubuntu, Ubuntu Server и Ubuntu GNOME — каждый член семейства «заточен» либо под определенный контингент пользователей, либо под преобладающий набор приложений, либо под конкретную графическую среду, и познакомиться с их характеристиками можно, например, здесь: <http://ubuntu.ru/family>.

«Фишка» этого дистрибутива — частое обновление. Новые версии Ubuntu выходят два раза в год (текущая версия — 17.04). Существует два типа версий Ubuntu: обычные и LTS. Разница между ними в том, что LTS (Long Term Support) — это дистрибутив с увеличенным сроком поддержки: обычные версии дистрибутивов Ubuntu выходят два раза в год, а LTS — только один раз. Однако техническая поддержка и обновление программ для LTS-дистрибутивов доступны на протяжении 5 лет. Это означает, что вы можете установить в 2017 году текущую LTS-версию (16.04 LTS), а следующую вам достаточно будет установить лишь в 2019-м. LTS-дистрибутивы лучше устанавливать на предприятиях, поскольку там не вполне удобно производить обновление дистрибутивов каждые полгода. Впрочем, для предприятий я бы рекомендовал что-либо более стабильное — например, Debian или CentOS, т. к. в настоящее время это два самых стабильных дистрибутива.

В целом, Ubuntu — очень неплохой дистрибутив, а с помощью этой книги вы узнаете, как «довести его до ума».

1.2.7. Slackware

Дистрибутивы Slackware (www.slackware.com) сочетают в себе стабильность, простоту и безопасность. Но для офисного и домашнего применения они не столь удобны из-за весьма посредственной русификации.

Программа установки Slackware также оставляет желать лучшего — это наименее удобная программа установки из всех, которые я видел. Тут, как на машине времени, переносишься лет на десять назад, — давно я вручную не выполнял разметку диска с помощью команды `fdisk` и не выбирал отдельные пакеты с помощью текстовой программы установки. Одним словом, Slackware — не самый лучший выбор для новичка, хотя некоторые фанаты Linux называют Slackware «настоящим Linux» (True Linux). Спорить с ними сложно, но начинающим пользователям лучше выбрать другой дистрибутив.

Нужно отметить, что Slackware — это настоящий старожил. Первая его версия появилась в 1993 году, т. е. 24 года назад. Тем не менее, дистрибутив не умер, а развивается, и на сегодняшний день доступна его четырнадцатая версия (14.2, если быть предельно точным).

Рекомендовать этот дистрибутив начинающим пользователям я не решаюсь также из-за замысловатой системы управления пакетами, усложняющей их установку и обновление (особенно обновление!). Тем не менее, Slackware будет рассмотрен в нашей книге, чтобы после ее прочтения вы смогли работать и с ним.

1.2.8. openSUSE

openSUSE (www.opensuse.org) — превосходный немецкий дистрибутив. Когда я впервые с ним познакомился, то он мне понравился больше, чем Mandriva и Fedora вместе взятые.

Дистрибутив весьма несложен (хотя и не упрощен до того уровня, когда ощущаешь недостаток функционала, — как в случае с Ubuntu), но, в то же время, предоставляет все, что нужно, для полноценной работы, и идеально подойдет для офисного и

домашнего компьютера. При использовании openSUSE создается впечатление добротно сделанного дистрибутива, не требующего «хирургического» вмешательства (как в случае с Fedora и Ubuntu), чтобы «довести систему до ума».

Особого внимания заслуживает технология установки программного обеспечения по одному щелчку. Хотите установить кодеки для просмотра фильма? Или проприетарные драйверы видеокарты? Вам нужно сделать один щелчок мышью и просто подождать, пока все необходимое программное обеспечение будет установлено. При этом вам даже не придется вникать в тонкости системы управления пакетами (тем не менее, мы ее подробно рассмотрим).

В настоящее время существуют два варианта openSUSE: Tumbleweed и Leap. Все самое новое ПО включено в первый, а во второй — лишь все самое стабильное. Для домашнего компьютера я бы выбрал Tumbleweed, а для офиса — лучше Leap. Если вы не склонны к экспериментам, тогда можно и на домашнем ПК установить Leap.

Кстати, недавно я установил этот дистрибутив на сервер. И очень доволен! Никаких нареканий — все работает, как хорошие часы. Чувствуется, что к дистрибутиву «приложила руку» коммерческая компания — Novell.

Одним словом, можете смело устанавливать этот дистрибутив — вы не будете в нем разочарованы.

1.3. На каком дистрибутиве основать сервер?

Очень часто читатели задают именно этот вопрос. И не мудрено, ведь Linux — это не только настольная система, и довольно часто приходится на базе Linux настраивать сервер. Но какой дистрибутив для этого выбрать?

Если вы ожидаете, что я скажу: выбирайте, например, openSUSE или Fedora, то вы ошибаетесь. Выбирайте тот дистрибутив, к которому вы больше привыкли, который освоили лучше всего и в котором ориентируетесь так же хорошо, как в собственном доме, — вам будет комфортнее работать с привычным дистрибутивом, и, следовательно, всевозможных «подводных камней» вы ощутите меньше.

Почему так? Да потому что ядро системы — везде одно и то же (если сравнивать актуальные версии дистрибутивов), а все необходимое для создания сервера программное обеспечение имеется в составе любого дистрибутива. Даже если после установки окажется, что версия, например, Web-сервера не самая новая, никто не запрещает вам скачать самую последнюю его версию с сайта проекта или просто обновить ее, — если дистрибутив, который вы выбрали, выпущен не вчера, наверняка в репозитории уже есть новая версия пакета.

Если же вы желаете установить дистрибутив, который изначально предназначен именно для сервера, то обратите внимание на RHEL, CentOS, Fedora Server 22-25 (начиная с версии 22, ядро в Fedora поддерживает Live Kernel Patching — технологию, которая, возможно, вам и не понадобится, но если возникнет необходимость, лучше, чтобы она была). Можно также с успехом использовать и Debian — пусть это и не сугубо серверный дистрибутив, но зато он один из самых надежных дистрибутивов в мире Linux.

ГЛАВА 2



Особенности установки

Установка Linux совсем не похожа на установку привычной многим операционной системы Windows. И здесь мы рассмотрим особенности установки Linux, с которыми вы просто обязаны разобраться до ее начала. Зная эти особенности, установить Linux сможет даже совсем новичок, — ведь вся установка проходит в графическом режиме, да еще и на русском языке, что существенно облегчает весь процесс.

Забегая вперед (об этом мы еще поговорим позже), хочу сразу предупредить, что Linux нужно устанавливать после Windows, потому что загрузчик Linux без проблем загружает все имеющиеся версии Windows, а вот заставить загрузчик Windows загружать Linux довольно сложно. Поэтому, чтобы не усложнять себе жизнь, сначала установите все нужные вам версии Windows, а затем — все необходимые дистрибутивы Linux.

2.1. Системные требования

В прошлом даже самые современные на то время версии Linux были не очень «прожорливыми» и могли работать на компьютерах с 256-512 Мбайт оперативной памяти. Сейчас же инсталлятор последней версии openSUSE (42.3 Leap), хоть и запустился на одной из моих машин в графическом режиме, но когда я выбрал установку не с локального DVD, а из сетевых репозиториях, то он сообщил мне, что на машине недостаточно оперативной памяти: имелось 768 Мбайт, а инсталлятор потребовал не менее 1000 Мбайт (рис. 2.1).

А ведь установка из сетевых репозиториях необходима, чтобы после ее завершения не производить обновление системы, — из репозитория сразу будут установлены самые новые версии пакетов.

Конечно, все современные компьютеры оснащены как минимум двумя гигабайтами ОЗУ, и сообщение, показанное на рис. 2.1, скорее всего, вы никогда не увидите. Но Linux всегда славилась небольшими требованиями к оперативной памяти, а сейчас же, как видите, и ей уже нужен минимум 1 Гбайт... К слову, 1 Гбайт ОЗУ — это как раз минимальные требования для Windows 10 (и я запускал Windows 10 в виртуальной машине с 1 Гбайт оперативной памяти, хотя работать в Windows 10 при

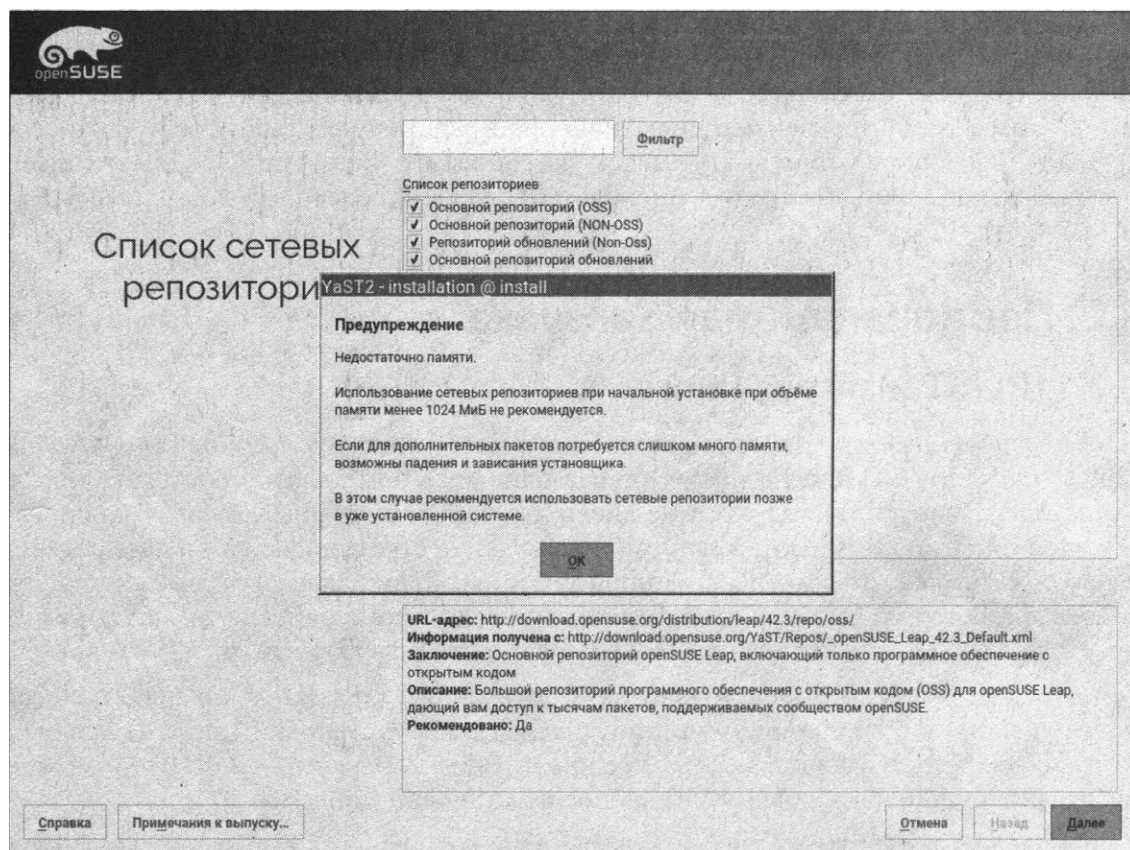


Рис. 2.1. openSUSE 42.3: для установки из сетевых репозиториях необходим 1 Гбайт ОЗУ

таком объеме памяти не слишком комфортно), так что по системным требованиям Linux уже почти сравнялась с Windows.

По части *дискового пространства*— ориентируйтесь минимум на 8-10 Гбайт (это с небольшим запасом — ведь еще нужно оставить место для своих данных), и это вполне приемлемо по нынешним меркам, учитывая, что после установки вы получаете не «голую» систему, а уже практически готовую к работе, — с офисными пакетами и программами мультимедиа. Если же вы настраиваете сервер, то все офисные и мультимедийные программы, понятно, можно не устанавливать, и тогда для самой системы понадобится примерно 2 Гбайт (без графического интерфейса — он на сервере не нужен, но с необходимыми пакетами, содержащими программы-серверы). Впрочем, не нужно забывать, что само слово «сервер» подразумевает достаточное количество дискового пространства, поэтому вам потребуется 2 Гбайт для самой системы и еще сколько-то для данных, которые сервер будет обрабатывать.

Для *корневого раздела*, где содержатся файлы операционной системы и приложения, я бы порекомендовал установить размер минимум 7-8 Гбайт, а для раздела с пользовательскими файлами (*/home*) установите размер, соответствующий предполагаемому объему обрабатываемых данных.

У меня, например, openSUSE 42.3 Leap сразу после установки заняла 5,7 Гбайт (версия с KDE), Ubuntu 17.04 — до 4,0 Гбайт (обновления во время установки не устанавливались), а Fedora 26 — 5 Гбайт. Обратите внимание, что для установленной системы требуется меньше дискового пространства, чем она просит для обеспечения процесса ее установки, — здесь указан размер уже установленных систем, а во время самой установки может понадобиться еще и некоторый дополнительный объем.

2.2. Первоначальная загрузка

2.2.1. POST и загрузчики

После включения питания компьютера запускается *процедура самотестирования* (Power On Self Test, POST), проверяющая основные компоненты системы: видеокарту, оперативную память, жесткие диски и т. д. Затем начинается загрузка операционной системы. Компьютер при этом ищет на жестком диске (и других носителях) *программу-загрузчик* операционной системы. Если такая программа найдена, то ей передается управление, если же такая программа не найдена ни на одном из носителей, выдается сообщение с просьбой вставить загрузочный диск.

В настоящее время актуален только один загрузчик: GRUB2 — он используется по умолчанию в большинстве дистрибутивов и после установки Linux начальным загрузчиком будет именно он (его предшествующую версию — GRUB — можно по желанию установить вручную лишь после установки Linux).

Задача загрузчика — предоставить пользователю возможность выбрать нужную операционную систему (ведь кроме Linux на компьютере может стоять и еще какая-либо операционная система) и передать ей управление. В случае с Linux загрузчик загружает *ядро операционной системы* и передает управление ему. Все последующие действия по загрузке системы: монтирование корневой файловой системы, запуск программы инициализации — выполняет ядро Linux.

2.2.2. Ядро Linux и его параметры

Ядро — это святая святых операционной системы Linux. Ядро управляет всем: файловой системой, процессами, распределением памяти, устройствами и т. п. Когда программе нужно выполнить какую-либо операцию, она обращается к ядру Linux. Например, если программа хочет прочитать данные из файла, то она сначала открывает файл, используя системный вызов **open** (), а затем читает данные из файла с помощью системного вызова **read** (). Для закрытия файла используется системный вызов **close** () .

Конечно, на практике все выглядит сложнее, поскольку Linux — многопользовательская и многозадачная система. Это значит, что с системой могут работать одновременно несколько пользователей, и каждый из пользователей может запустить несколько процессов. Ясно, что программе нужно учитывать «поправку на совместный доступ», т. е. во время работы с файлом одного из пользователей программа должна установить блокировку доступа к этому файлу другим пользователям. Впрочем, в такие нюансы мы сейчас вникать не станем.

Итак, ядро — это программа, самая главная программа в Linux. Как и любой другой программе, ядру Linux можно передать *параметры*, влияющие на его работу. Это можно сделать с помощью любого загрузчика Linux. При установке Linux, особенно если операционная система отказывается устанавливаться с параметрами по умолчанию, полезно передать ядру особые параметры. Например, на некоторых ноутбуках для установки Linux требуется передать ядру параметры **noauto** и **norpmcia**. Первый параметр запрещает автоматическое определение устройств, а второй — проверку PCMCIA-карт.

УСТРАНЕНИЕ ПРОБЛЕМ С ЗАГРУЗКОЙ LINUX

В разд. 2.11 приведено описание ряда проблем с загрузкой Linux и способов их устранения, в том числе и с помощью передачи ядру особых параметров.

Кроме передачи параметров ядру, можно передать параметры и программе установки — например, параметр **vga** при установке ряда дистрибутивов Linux определяет, что эта программа должна работать при разрешении 640x480, а это позволяет произвести установку на самые «древние» компьютеры или такие, видеокарта которых не полностью совместима с Linux (редко, но бывает).

В различных дистрибутивах редактирование параметров ядра, естественно, осуществляется по-разному. Так, в Fedora 26 нужно выбрать необходимый вариант установки (обычно выбирается первый, предлагающий установить или обновить существующую систему) и нажать клавишу <Tab>— в результате мы получим текстовую строку, в которой можно отредактировать параметры ядра (рис. 2.2).

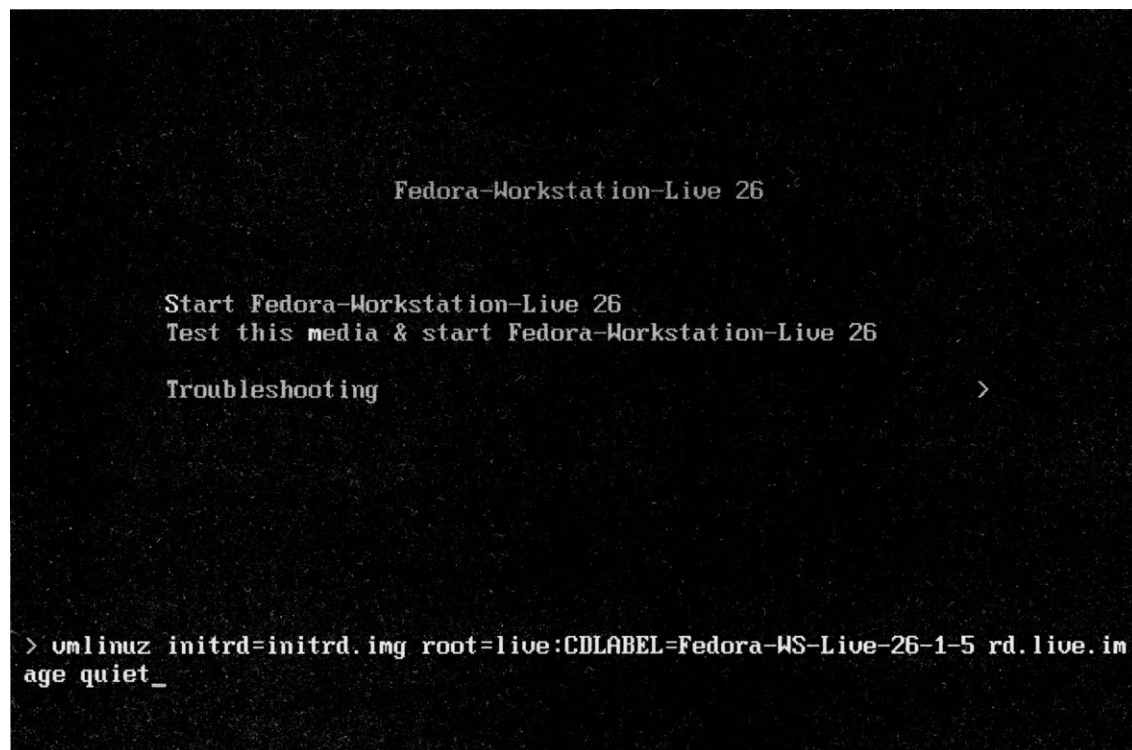


Рис. 2.2. Fedora 26: редактирование параметров ядра

ЗАПУСК ИНСТАЛЛЯЦИИ FEDORA 26

Если вы скачали дистрибутив Fedora 26 Live, то для запуска инсталляции надо сначала выбрать опцию **Start Fedora-Workstation-Live 26** (см. рис. 2.2), а после загрузки выполнить команду Install to Hard Drive.

ПАРАМЕТРЫ ПРОГРАММЫ-УСТАНОВЩИКА

Некоторые дистрибутивы, кроме параметров ядра, позволяют также ввести параметры программы-установщика. По адресу: https://docs.fedoraproject.org/en-US/Fedora/26/html/Installation_Guide/chap-anaconda-boot-options.html вы можете ознакомиться с параметрами программы установки Fedora 26.

При установке openSUSE 42.3 для редактирования параметров ядра следует выбрать необходимый вариант установки (рис. 2.3), нажать клавишу <F5> и добавить параметры загрузки в поле **Варианты загрузки**, находящееся под списком вариантов загрузки (рис. 2.4).

ВЫБОР ЯЗЫКА УСТАНОВКИ

Обратите внимание, что меню загрузки openSUSE на рис. 2.3 и 2.4 представлено на русском языке. Однако сразу после загрузки с DVD меню выводится на английском, и для смены языка установки следует нажать клавишу <F2> и выбрать русский язык из списка. Такая возможность есть не у всех дистрибутивов— например, у Fedora она, к сожалению, отсутствует.

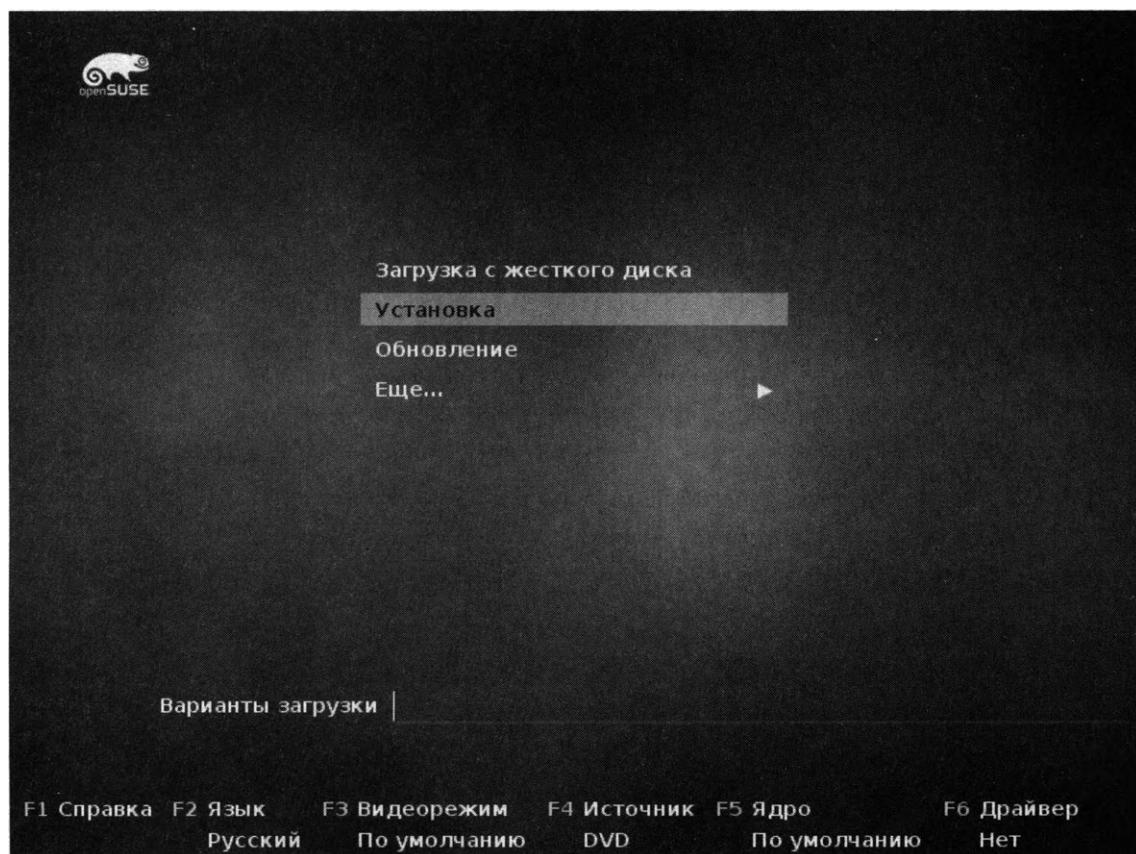


Рис. 2.3. openSUSE 42.3: начальное меню установки

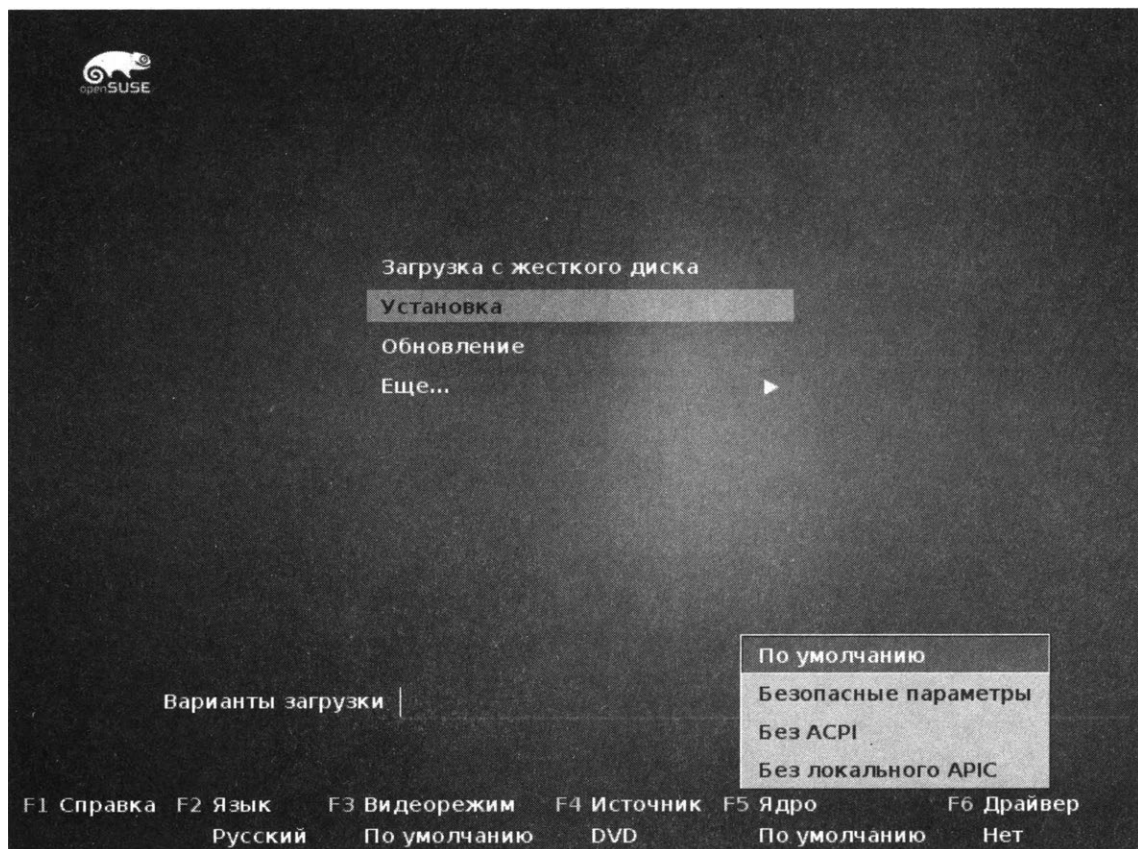


Рис. 2.4. openSUSE 42.3: редактирование параметров ядра при установке

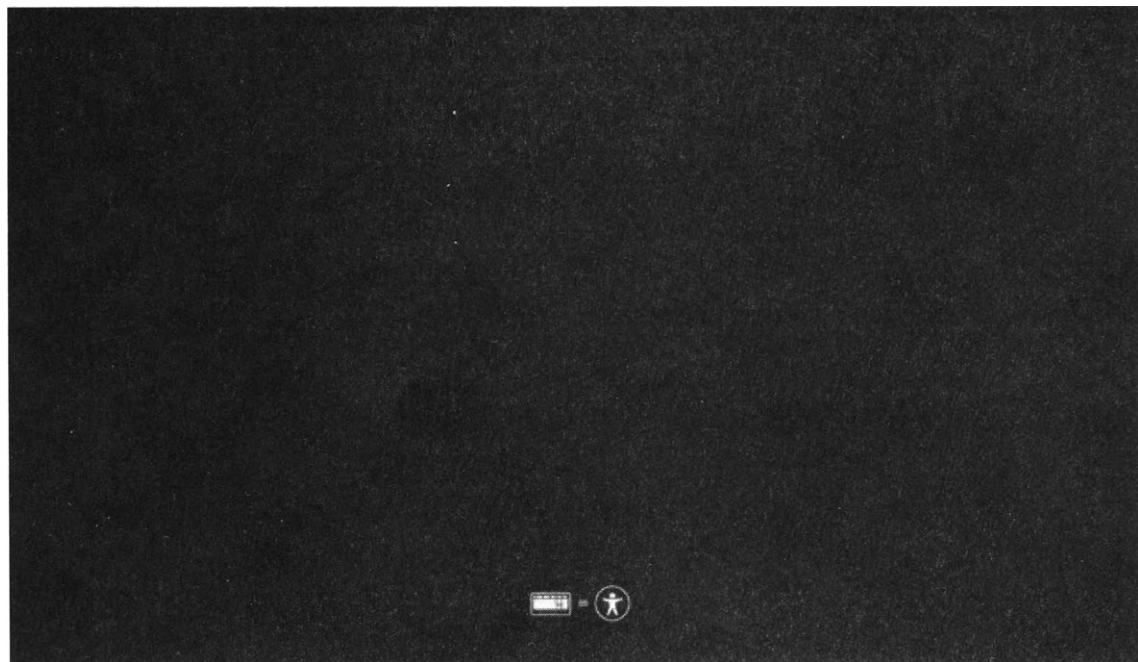


Рис. 2.5. Ubuntu 17.04: заставка при запуске с установочного диска

При установке Ubuntu сначала появится графическая заставка (рис. 2.5) — вам следует нажать здесь любую клавишу, и на экран будет выведено меню выбора языка. После выбора языка появится загрузочное меню на выбранном пользователем языке. Для выбора параметров ядра нужно нажать клавишу <F6> (рис. 2.6).

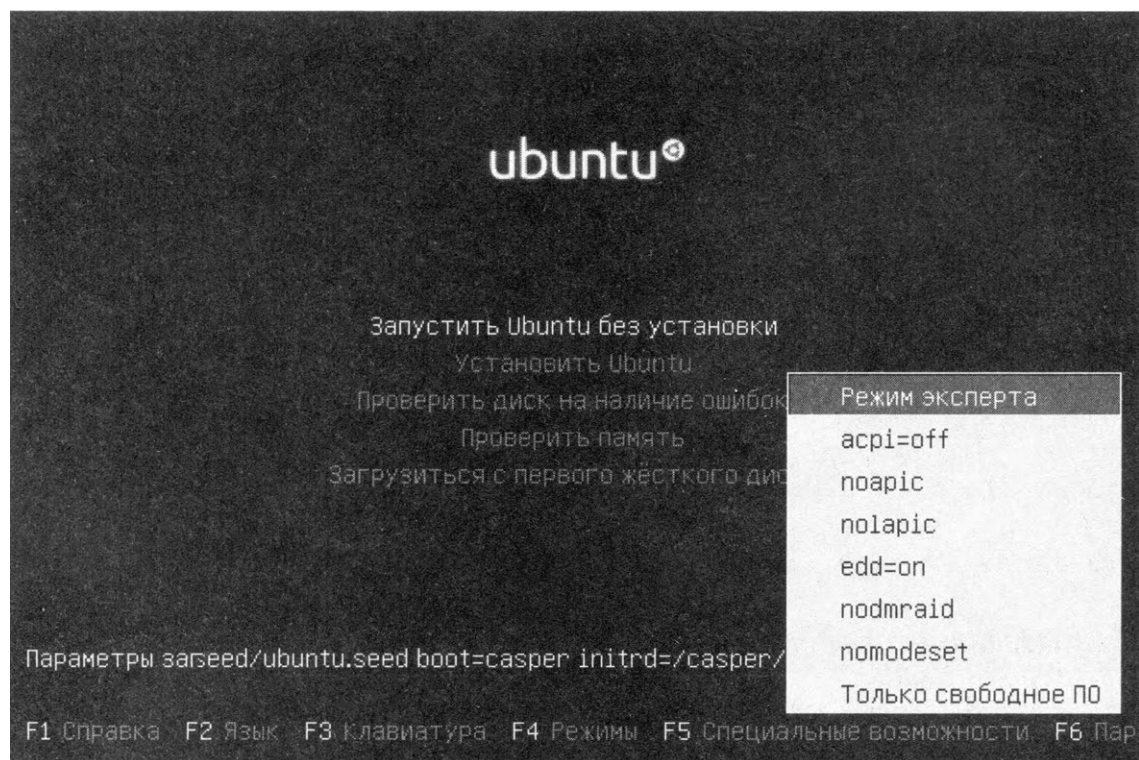


Рис. 2.6. Ubuntu 17.04: выбор параметров ядра при установке

ВИДЕО УСТАНОВКИ UBUNTU

В папке Видео сопровождающего книгу электронного архива (см. приложение) содержится видеофайл, демонстрирующий процесс установки на компьютер дистрибутива Ubuntu 15.10 (процедура инсталляции Ubuntu настолько унифицирована, что переснимать видео под версию 17.04 не потребовалось).

Подробнее о параметрах ядра вы сможете прочитать в *главе 20*.

2.3. Проверка носителей

Некоторые дистрибутивы предлагают перед установкой выполнить проверку установочного DVD. Так что, если поверхность DVD вызывает у вас сомнения, можно его проверить, — зачем тратить время на установку, если на 99-м проценте программа установки сообщит вам, что ей не удастся прочитать какой-то очень важный пакет, и система не может быть установлена? Если же DVD новый (только что купленный или записанный), можно отказаться от проверки носителя — вы сэкономите немного времени.

Так, для проверки носителя Fedora нужно выбрать в загрузочном меню опцию **Test this media & start Fedora-Workstation-Live 26**, а в Ubuntu — опцию **Проверить диск на наличие ошибок**.

2.4. Изменение таблицы разделов

Система Linux не может быть установлена в Windows-разделы: FAT32 или NTFS, и для нее нужно создать на жестком диске компьютера Linux-разделы в файловой системе ext3 или ext4. Понятно, что для этого на жестком диске должно иметься *неразмеченное пространство*. Если его нет, придется или удалить один из Windows-разделов и на его месте создать Linux-раздел, или же уменьшить размер одного из Windows-разделов и на освободившемся месте создать разделы Linux.

Удалять раздел Windows имеет смысл только в том случае, если вся содержащаяся в нем информация вам абсолютно не нужна, поэтому обычно дело до удаления не доходит, — просто размер подходящего Windows-раздела уменьшают на величину имеющегося в нем свободного пространства. Так что, перед началом установки убедитесь, что в каком-либо разделе Windows есть 8-10 Гбайт свободного пространства (вообще, чем больше, тем лучше).

СДЕЛАЙТЕ РЕЗЕРВНУЮ КОПИЮ ВСЕХ ВАЖНЫХ ДАННЫХ

Часто бывает, что возможности программы-установщика по изменению размеров уже существующего раздела ограничены. Поэтому перед установкой Linux на компьютер, где уже установлена другая операционная система, я рекомендую сделать резервную копию всех важных данных и использовать стороннюю программу разметки — например, AOMEI Partition Assistant (<https://www.aomeitech.com/aomei-partition-assistant.html>).

УСТАНОВКА ДИСТРИБУТИВА С ЗАГРУЗЧИКОМ LILO

Если вы устанавливаете очень старый дистрибутив Linux, в котором все еще используется загрузчик LILO, то основной раздел Linux следует расположить ближе к началу диска. Дело в том, что загрузчик LILO может загружать Linux только с тех разделов, которые начинаются до 1024-го цилиндра (т. е. до 1024-го цилиндра должен располагаться первый блок раздела). Это не проблема операционной системы, а требование старого загрузчика Linux. В некоторых случаях проблему удастся обойти, а в некоторых — нет. Лучше лишний раз не тратить время зря и создать Linux-раздел так, чтобы он начинался как можно ближе к «началу» диска. После установки Linux сможет использовать (читать и записывать данные) любые разделы вне зависимости от начального номера цилиндра раздела.

Перед установкой Linux следует также произвести *дефрагментацию* того Windows-раздела, который вы собрались уменьшать, чтобы упростить задачу программе установки по переносу ваших файлов.

В любом дистрибутиве программа установки системы Linux умеет автоматически разбивать жесткий диск — она сама создаст Linux-разделы без вашего участия.

2.4.1. Разметка диска в Fedora 26

В более ранних дистрибутивах Fedora программа разметки диска, на мой взгляд, была более удобной, чем в ее последних версиях, но мы имеем то, что имеем, и должны с этим работать.

Итак, если вы устанавливаете Fedora на новый компьютер, где еще не было установлено никаких других операционных систем, проще всего выбрать вариант **Автоматически** (рис. 2.7).

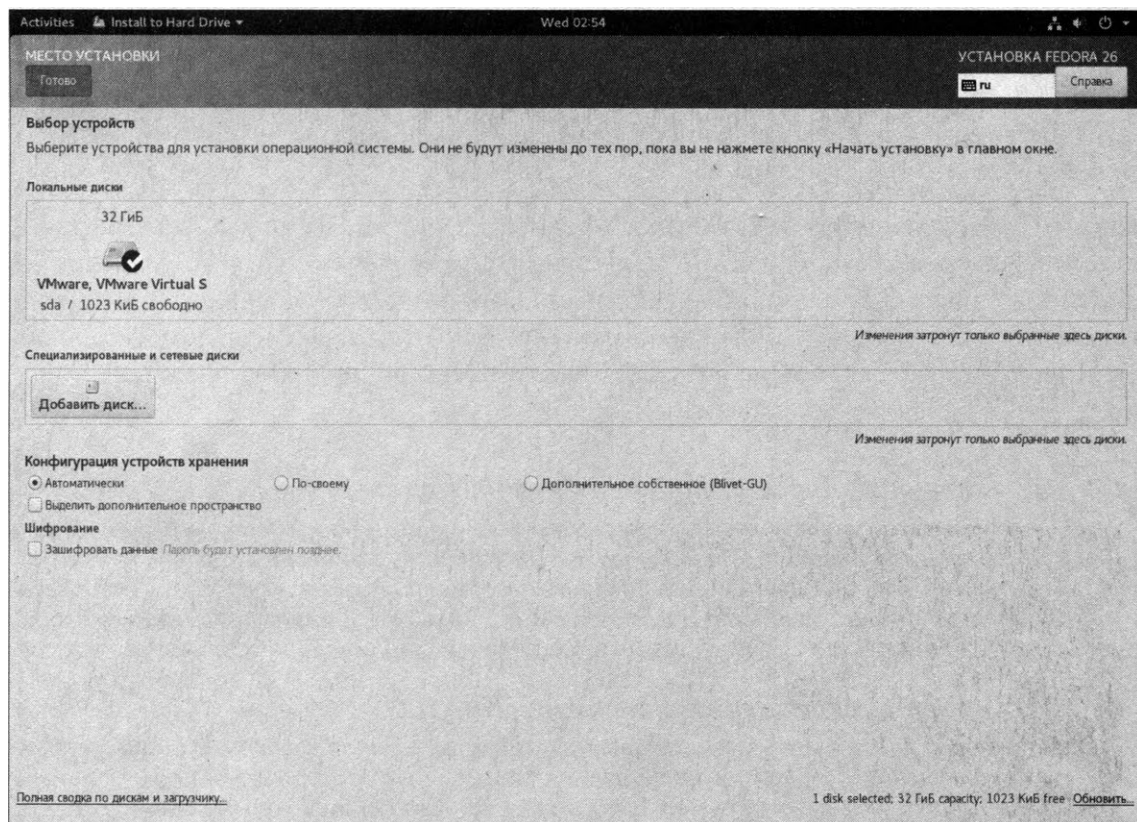


Рис. 2.7. Fedora 26: выбор типа разметки

Однако если на жестком диске уже есть таблица разделов, или вы желаете создать разделы вручную, — выберите вариант **По-своему** и нажмите кнопку **Готово** (она находится в верхнем левом углу окна).

По умолчанию Fedora предлагает использовать LVM (Logical Volume Manager), но на домашней машине можно создать обычные разделы (да и, скорее всего, обычные разделы для Windows у вас уже созданы). Поэтому из списка схем разбиения нужно выбрать **Стандартный раздел** (рис. 2.8).

Затем для создания раздела нажмите кнопку **+** (для удаления раздела, соответственно, служит кнопка **-**), после чего в открывшемся окне выберите *точку монтирования* и введите размер раздела (рис. 2.9). При вводе размера можно добавить

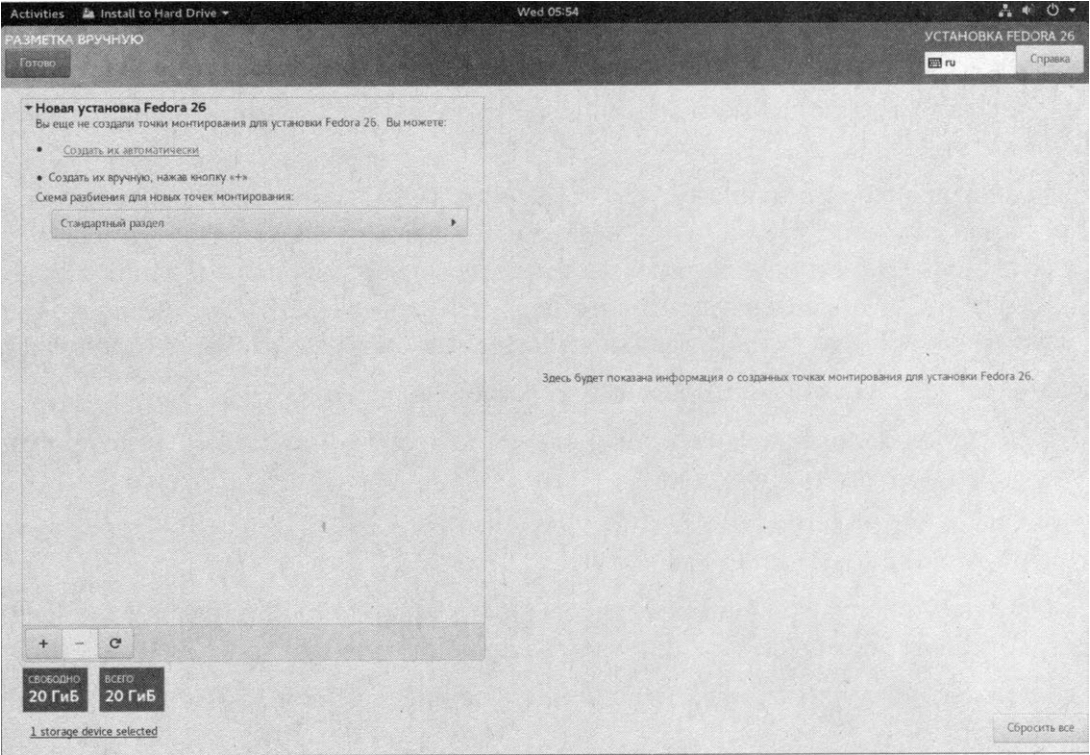


Рис. 2.8. Fedora 26: выберите Стандартный раздел

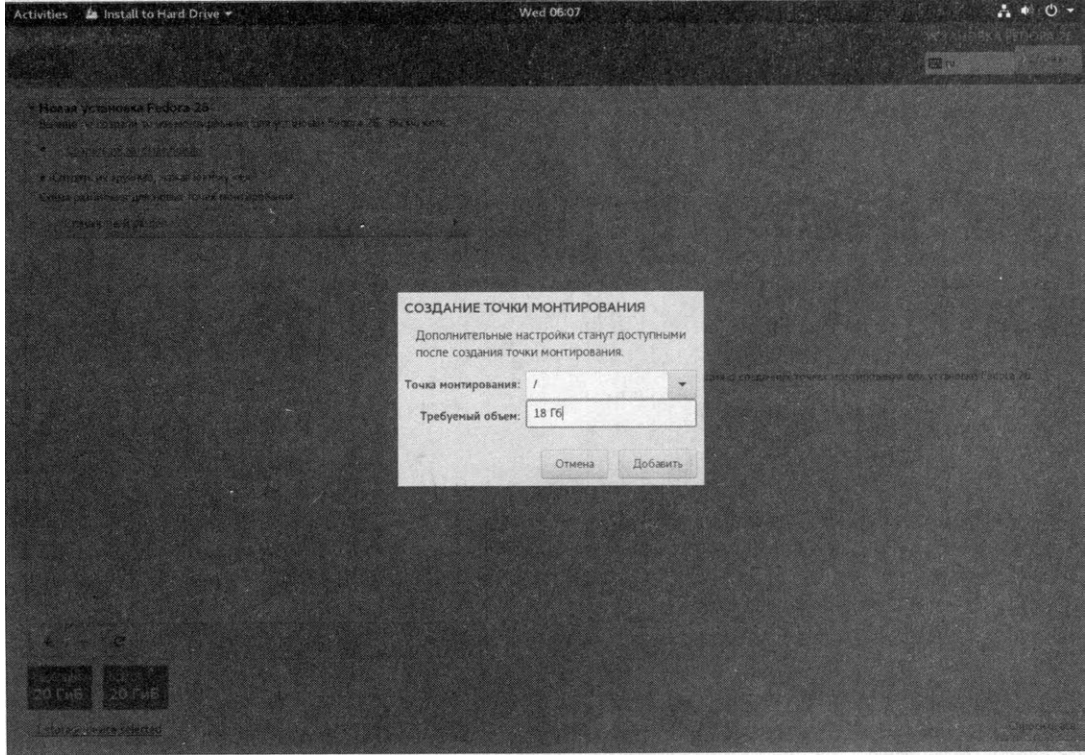


Рис. 2.9. Fedora 26: создание основного раздела

символы **G** или **Гб** (в современных версиях Fedora объем можно вводить на русском языке), чтобы указать, что размер задается в гигабайтах.

Как минимум, вам понадобится создать раздел с точкой монтирования `/` и *раздел подкачки* (swap).

Точка монтирования `/` используется для корневой файловой системы, которая помимо всего прочего (файлов самой системы) содержит также каталоги `/home` (тут хранятся пользовательские данные) и `/var` (а здесь лежат журналы и данные различных серверов: Web-сервера, почтового сервера, сервера БД). Именно поэтому на домашнем компьютере с одним жестким диском возможны две схемы разметки диска:

- один раздел с точкой монтирования `/` и раздел подкачки;
- два раздела: один с точкой монтирования `/`, второй — с точкой монтирования `/home`, а также раздел подкачки.

Для сервера нужно создать отдельные разделы для каждой точки монтирования: `/`, `/home` и `/var`, а также раздел подкачки.

Размер раздела подкачки, учитывая требования к оперативной памяти современных дистрибутивов, должен быть установлен на уровне 2-4 Гбайт (рис. 2.10).

Закончив разметку диска, нажмите кнопку **Готово** для записи изменений на диск (рис. 2.11).

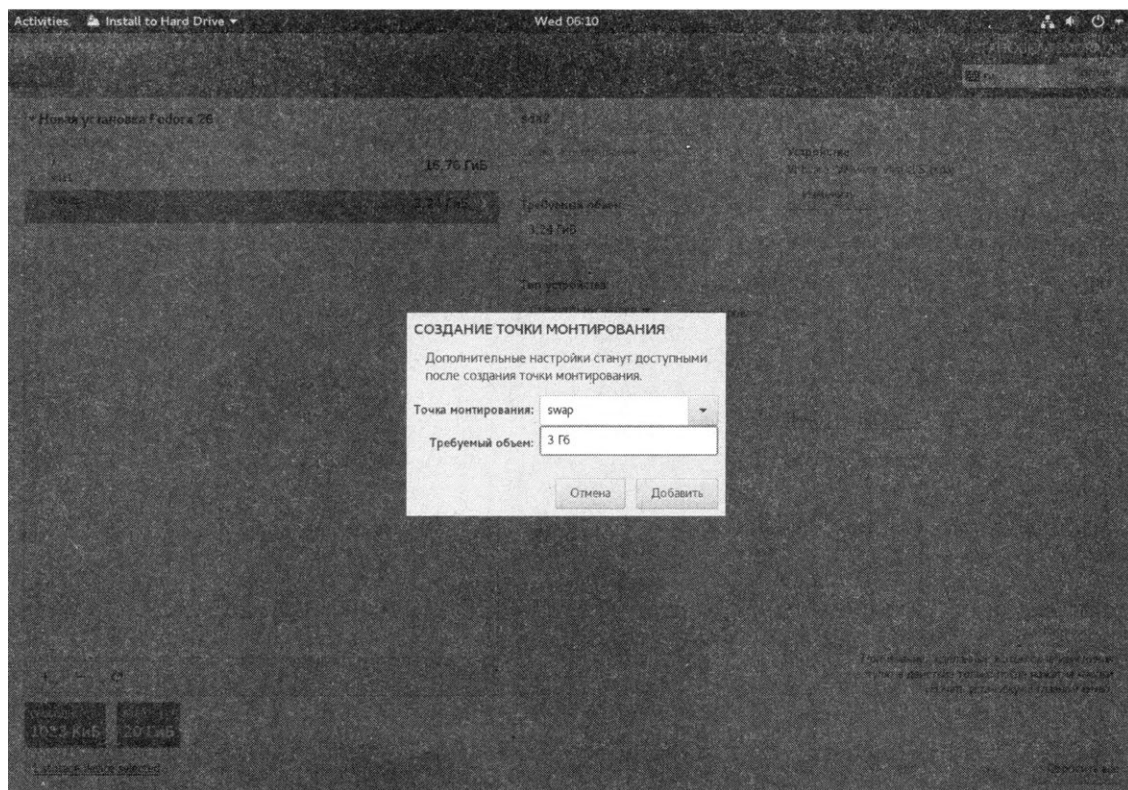


Рис. 2.10. Fedora 26: создание раздела подкачки

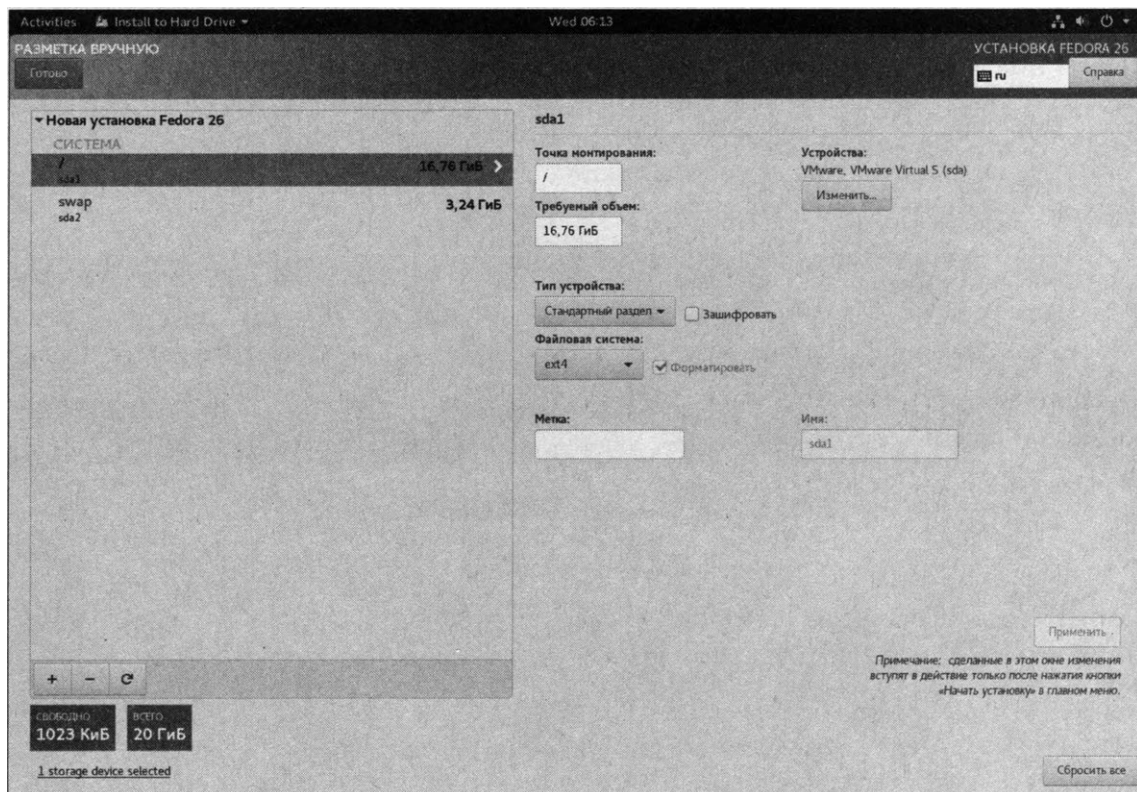


Рис. 2.11. Fedora 26: разметка диска завершена

Основной недостаток программы разметки диска в Fedora — она не умеет уменьшать размер уже существующих разделов. И если диск вашего компьютера уже полностью разбит в Windows на разделы, скажем, C: и D:, то для установки на такой компьютер Linux один из этих разделов придется удалить. Именно поэтому установка Fedora на компьютеры, где уже установлена Windows, и есть всего лишь один раздел, будет весьма проблематичной, — потребуется или задействовать сторонние программы разметки диска (хотя бы для изменения размера существующего раздела Windows), или переустанавливать Windows. При этом сначала надо будет выполнить разметку диска, оставив для Linux нераспределенное пространство, потом установить Windows, а затем уже и Linux. Честно говоря, такая вот система установки никак не способствует росту популярности Linux...

Помню, дистрибутив Mandriva включал отличную программу разметки дисков `diskdrake`, которая могла уменьшить на величину свободного дискового пространства любой выбранный раздел, а на освободившемся пространстве создать Linux-разделы.

2.4.2. Разметка диска в Ubuntu 17.04

Программа разметки диска в Ubuntu 17.04 мне понравилась гораздо больше, чем в Fedora,— она позволяет более гибко управлять созданием разделов: вы можете выбрать тип (первичный, логический) и местоположение нового раздела (рис. 2.12),

а также и устанавливаемую файловую систему (рис. 2.13). В Fedora же по умолчанию устанавливается файловая система ext4 (XFS — для серверной версии), а пользователю предоставляется возможность лишь указать размер раздела и точку монтирования. После создания раздела вы можете сменить тип файловой системы, но для этого нужно будет производить лишние действия: выбирать раздел, менять тип файловой системы, нажимать кнопку **Применить**.

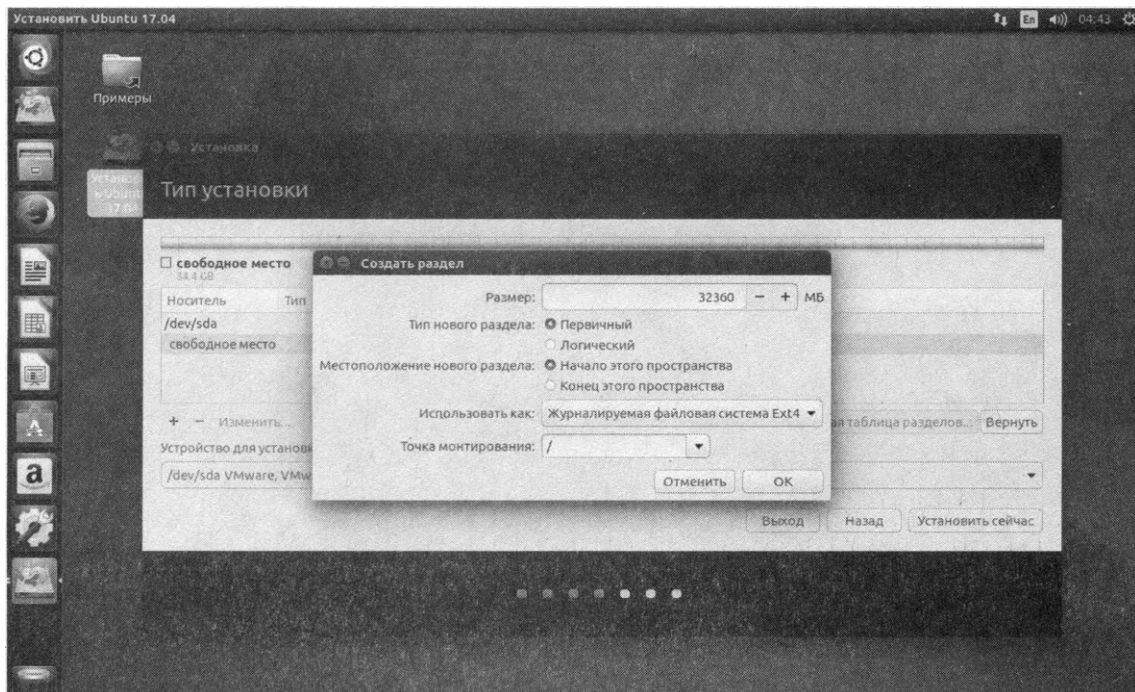


Рис. 2.12. Ubuntu 17.04: создание нового раздела

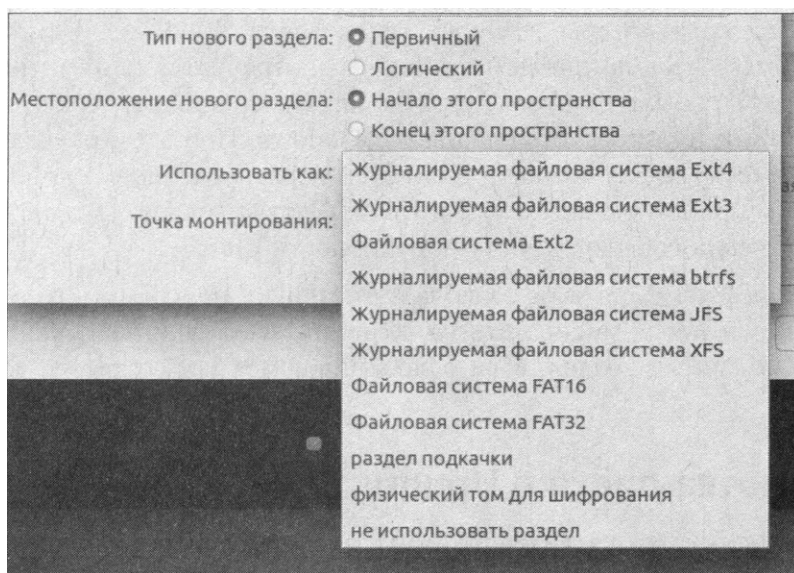


Рис. 2.13. Ubuntu 17.04: выбор файловой системы

К сожалению, программа разметки в Ubuntu не лишена того же недостатка, что и в Fedora, — кнопка **Изменить** позволяет выбрать лишь другую файловую систему и точку монтирования, но не дает возможности изменить размер раздела (рис. 2.14).

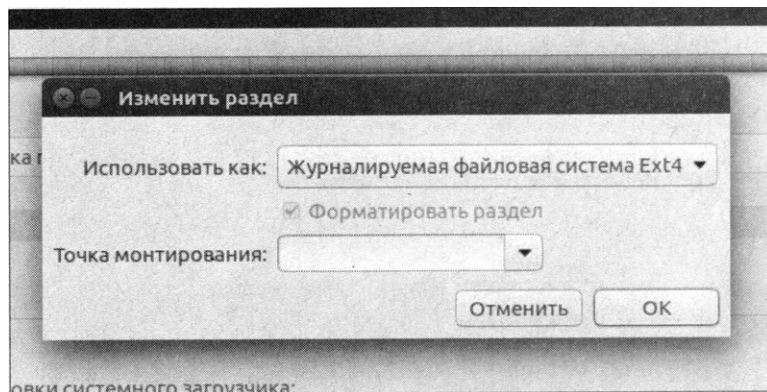


Рис. 2.14. Ubuntu 17.04: изменение раздела

2.4.3. Разметка диска в openSUSE

openSUSE— один из немногих дистрибутивов, программа разметки которого позволяет *изменить размер* существующих разделов. Для этого, когда программа

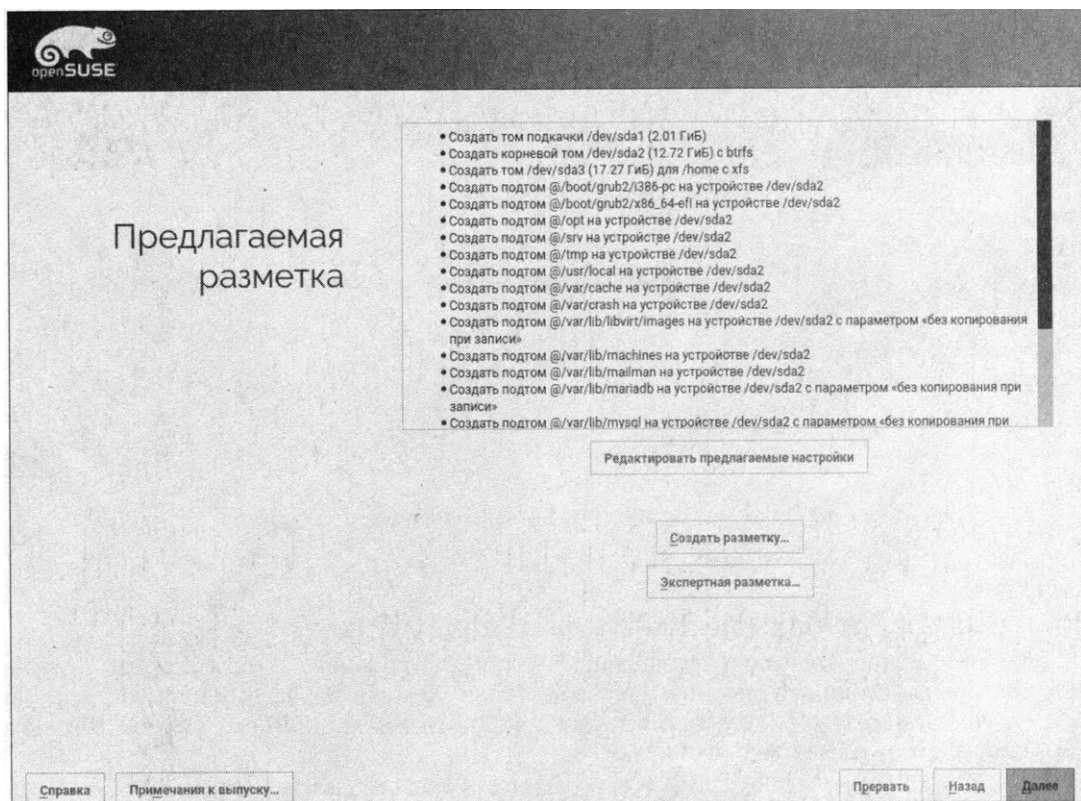


Рис. 2.15. openSUSE 42.3: нажмите кнопку Экспертная разметка

установки предложит вам свою разметку диска, следует нажать кнопку **Экспертная разметка** (рис. 2.15), в открывшемся окне щелкнуть правой кнопкой мыши по нужному разделу и выбрать команду **Изменить размер**, после чего в открывшейся панели можно будет установить новый размер раздела (рис. 2.16).

Надо признать, программа разметки в openSUSE 42.3 весьма сбалансирована и гибка. Как можно видеть, она позволяет выбрать тип файловой системы, отказаться от форматирования, просто создав раздел заданного типа, зашифровать раздел, отказаться от монтирования созданного раздела (рис. 2.17). Конечно, все эти опции доступны при выборе экспертной разметки.

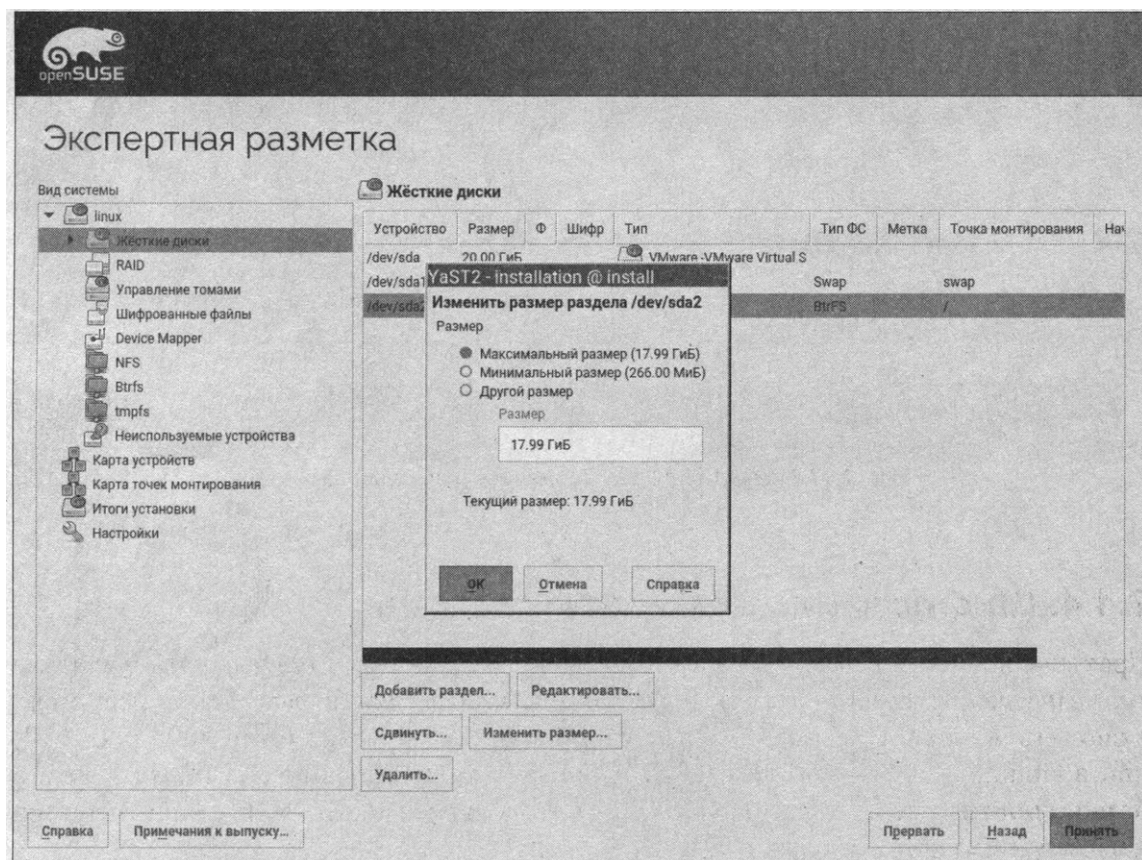


Рис. 2.16. openSUSE 42.3: изменение размера раздела

ПРЕЗЕНТАЦИЯ УСТАНОВКИ ДИСТРИБУТИВА SLACKWARE

В папке *Дополнения* сопровождающего книгу электронного архива (см. приложение) содержится файл презентации *Slackware14.ppt*, демонстрирующий процесс установки на компьютер дистрибутива Slackware 14. Здесь же вы найдете советы по работе с программой разметки диска fdisk.

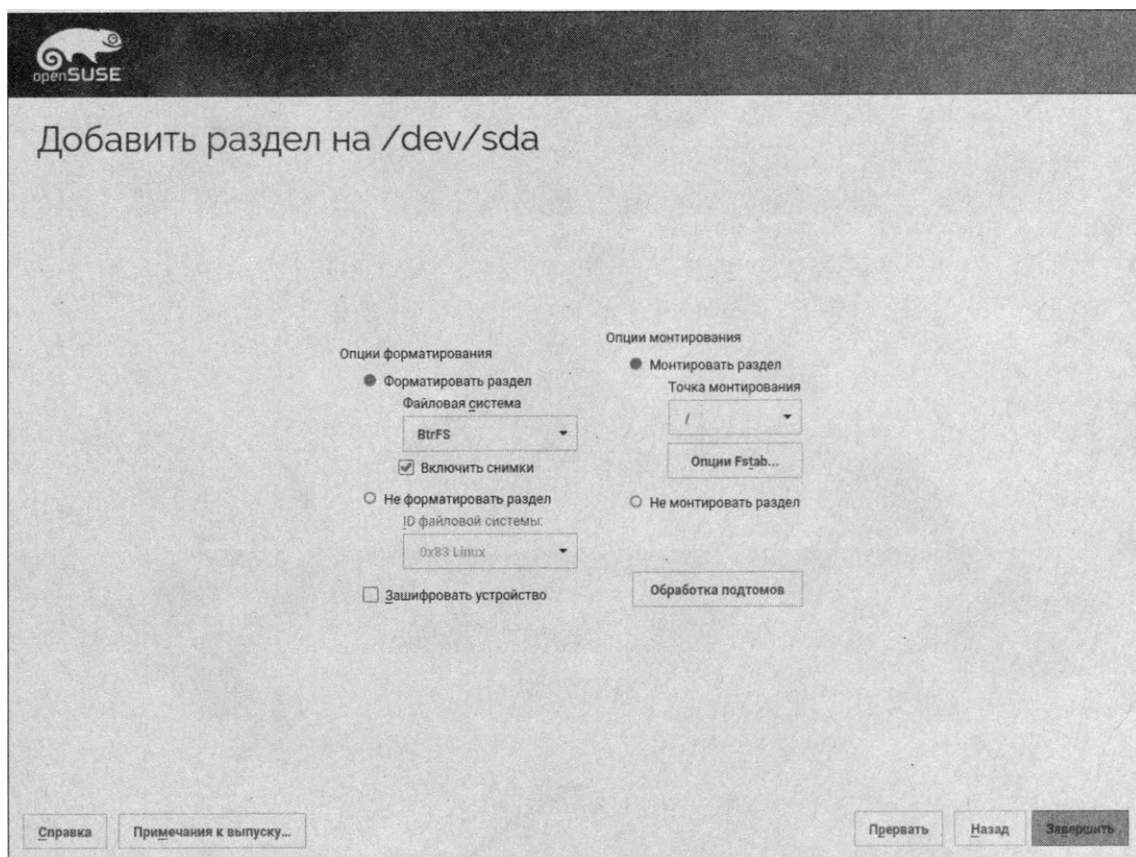


Рис. 2.17. openSUSE 42.3: возможности программы разметки

2.4.4. Шифрование файловой системы

Практически все современные дистрибутивы поддерживают *шифрование файловой системы*, которое вы можете предусмотреть при создании раздела, — например, включить в openSUSE 42.3 параметр **Зашифровать устройство** (см. рис. 2.17) или в Ubuntu 17.04— параметр **Зашифровать новую установку Ubuntu в целях безопасности** (рис. 2.18). Как можно видеть, в зависимости от дистрибутива этот параметр называется по-разному.

Но нужно ли вам это? Если вы агент 007 — бесспорно, шифрование весьма полезная опция. А вот во всех остальных ситуациях при попытке восстановления данных вчухучае сбоя системы опция шифрования создаст вам только дополнительные проблемы.

КРИПТОГРАФИЧЕСКАЯ ФАЙЛОВАЯ СИСТЕМА eCRYPTFS

Шифрование файловой системы с помощью криптографической утилиты eCryptfs описано в *главе 42*.

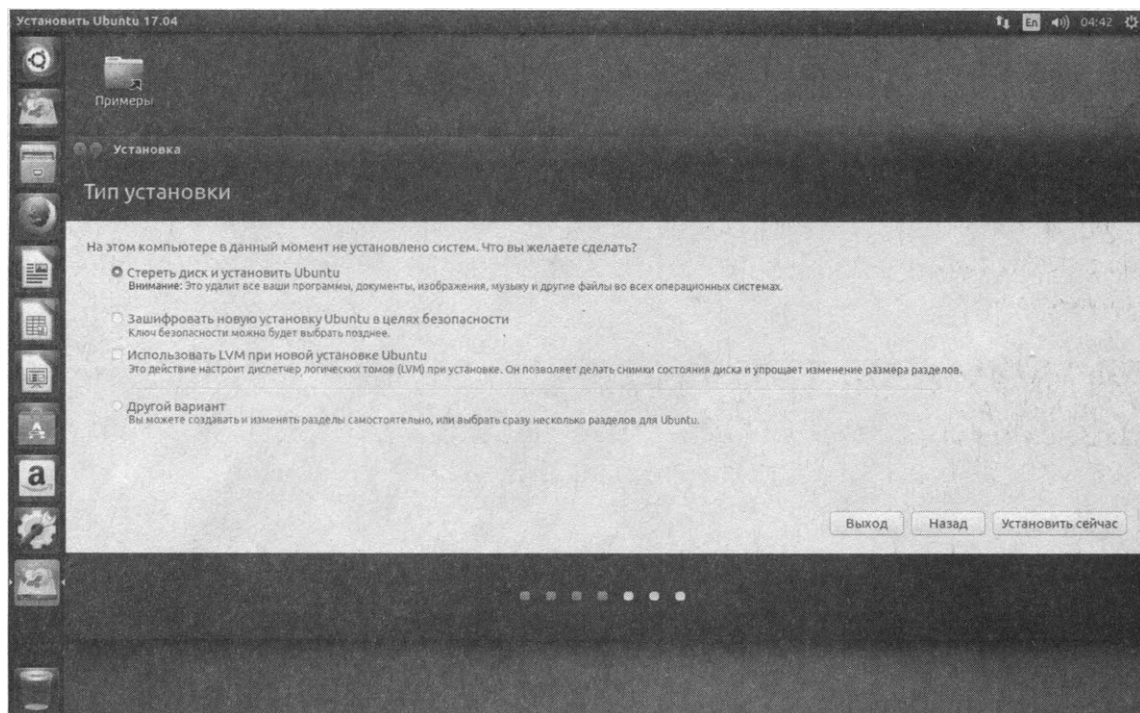


Рис. 2.18. Ubuntu 17.04: включение шифрования

2.5. Выбор устанавливаемых пакетов программ

Честно говоря, старые дистрибутивы мне нравились больше— можно было настроить дистрибутив под себя уже во время его установки. А сейчас разработчики дистрибутивов стараются упростить не только сами дистрибутивы, но и инсталляторы. Однако говорят же, что простота иногда хуже воровства...

Программы установки прошлых лет позволяли выбрать не только группы пакетов программ, но и отдельные пакеты. Было сложнее? Нет! Все выглядело вполне гармонично. Начинающий пользователь просто соглашался с выбором по умолчанию и нажимал кнопку **Далее** (или аналогичную), а опытный — выбирал пакеты (или хотя бы группы пакетов) вручную.

Сейчас же программы установки не только не позволяют выбрать группу пакетов — нельзя выбрать даже графическую среду (хотя в дистрибутиве их может быть несколько)! Да что там графическая среда— пользователь элементарно не знает, сколько места занимает установка по умолчанию. То есть, ставим вслепую: хватит места— хорошо, не хватит— начинай установку сначала. И хотя современные жесткие диски достаточно емкие для установки любого дистрибутива, но бывает, что пользователь ошибется и отведет под раздел с системой, допустим, всего 5 Гбайт, а системе потребуется 8... Однако пока не начнешь установку, об этом не узнаешь. К чему такая простота — не понятно...

Может быть, если устанавливаешь рабочую станцию или домашний компьютер, такое решение себя оправдывает. Но когда планируешь устанавливать сервер, то зачем инсталлятор устанавливает все эти офисные и мультимедиаприложения?

Единственным честным в этом смысле дистрибутивом остался openSUSE — он позволяет выбрать не только графическую среду, но и пакеты. Для выбора пакетов перед самой установкой системы будет выведена сводка (рис. 2.19) — щелкните на разделе **Программное обеспечение**, и у вас появится возможность выбрать устанавливаемые пакеты.

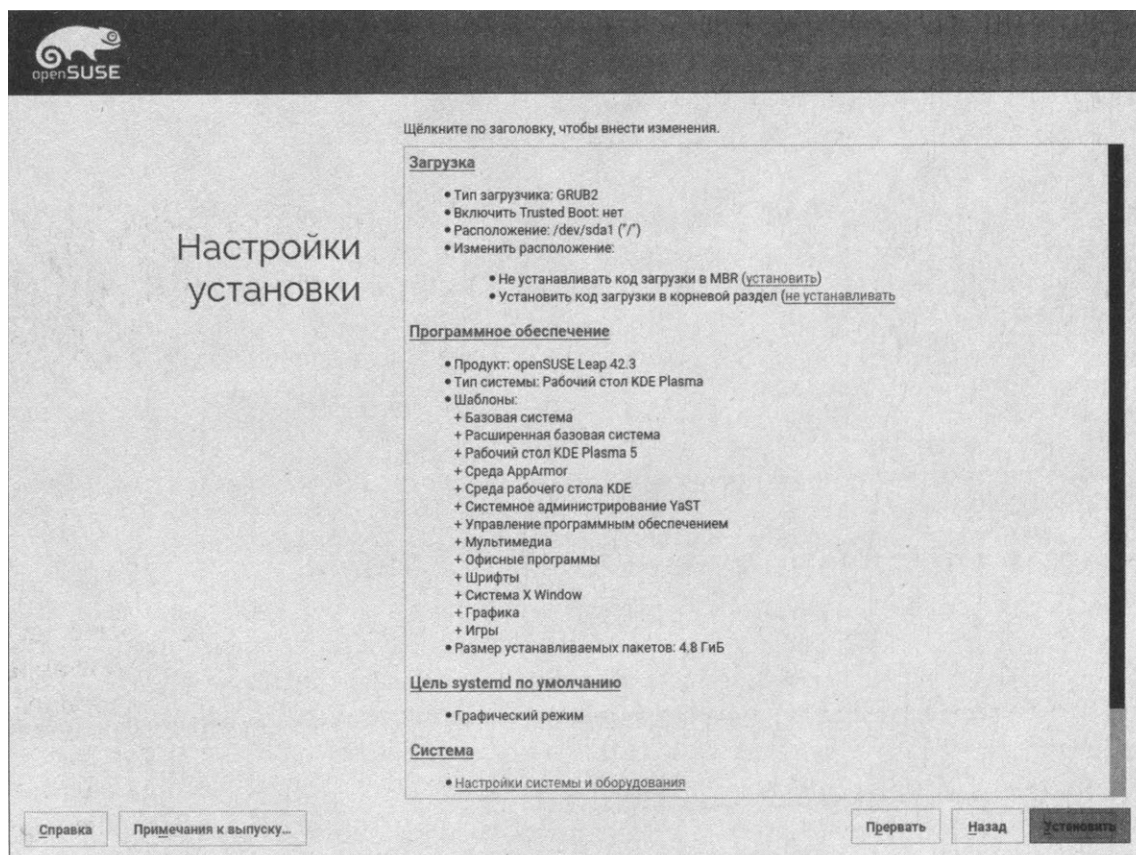


Рис. 2.19. openSUSE 42.3: выбор приложений

2.6. Выбор графической среды

Как было отмечено ранее, современные дистрибутивы редко когда предоставляют выбор *графической среды* (оболочки), поэтому нам придется привыкать к тому, что есть.

В Ubuntu по умолчанию устанавливается оболочка Unity, в Fedora — GNOME. И только openSUSE позволяет пользователю самому выбрать графическую оболочку (рис. 2.20).

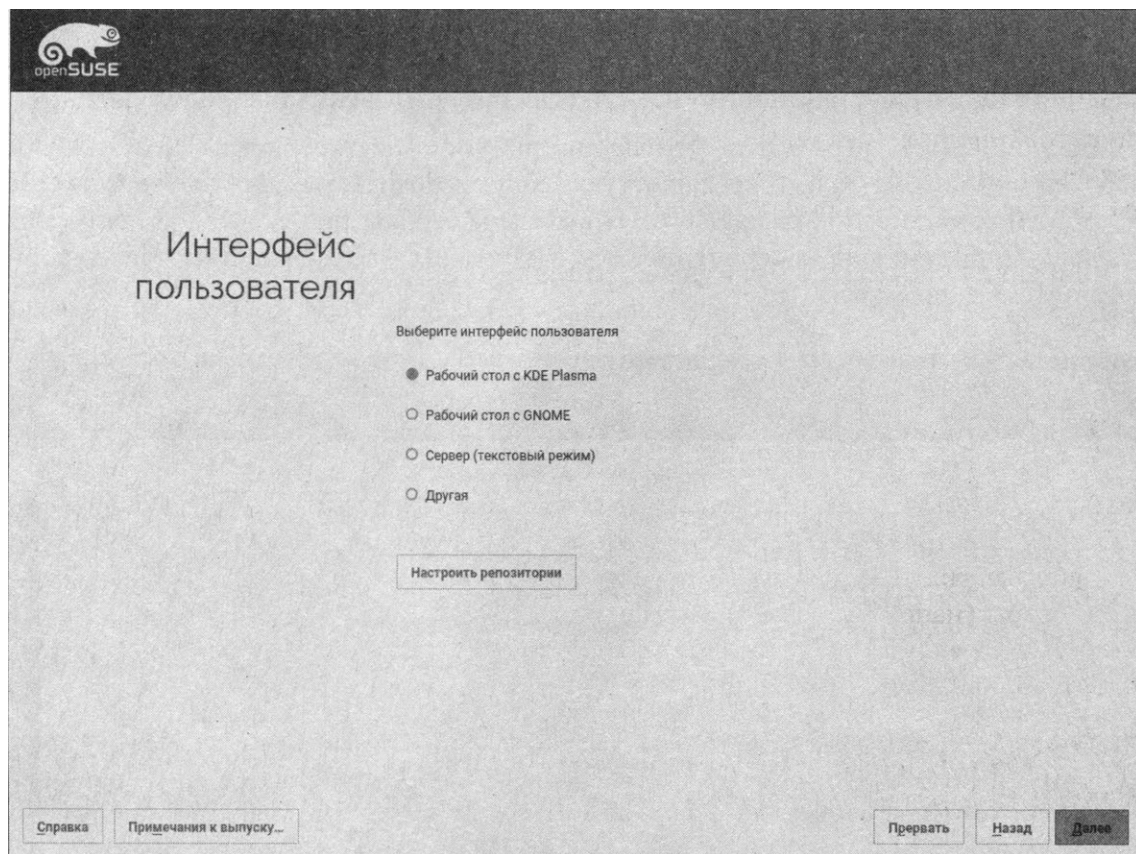


Рис. 2.20. openSUSE 42.3: выбор графической среды при установке системы

Итак, если все-таки возможность выбора имеется, то какую оболочку выбрать? Пользователям, привыкшим к какой-либо оболочке, нужно ее и выбирать, — так будет привычнее. Если же нет никаких предпочтений, выбирайте GNOME — эта графическая среда немного менее требовательна к системным ресурсам. Впрочем, третья версия GNOME тоже отличается «прожорливостью», однако все же не такой, как KDE 5.

Владельцам старых компьютеров можно порекомендовать оболочку LXDE — даже если инсталлятор не позволяет ее выбрать при установке системы, вы можете доустановить ее впоследствии.

Глядя на все это, иногда хочется вернуться в прошлое, когда инсталляторы при установке системы свободно позволяли выбрать графическую среду. Так почему же сейчас по умолчанию устанавливается только какая-то определенная среда, а выбор альтернативной даже не предоставляется? Ответ прост — этим разработчики облегчили себе жизнь. Ведь каждую оболочку нужно настраивать, «кастомизировать» под стилистику дистрибутива, а на все это необходимо время. А когда работы в два раза больше (если предусмотрены две среды вместо одной), то это сделать очень сложно и дорого. Плюс к тому же документацию (те же HOWTO) можно в этом случае написать только для одной среды. Ведь если написать HOWTO, допустим, по настройке кодеков, для GNOME, то для KDE придется описать совсем другую

последовательность действий. Таким образом, разработчики, облегчая себе жизнь, лишили пользователей свободы выбора (или освободили от проблемы выбора?), — впрочем, фанаты той или иной среды и более опытные пользователи и так смогут самостоятельно доустановить любимую графическую среду (см. главу 3) и правильно ее настроить. Да и лучше одна хорошо настроенная графическая среда, чем две недоделанные.

АЛЬТЕРНАТИВНЫЕ ГРАФИЧЕСКИЕ ОБОЛОЧКИ

Если вы любите все нестандартное, предлагаю статью-обзор альтернативных графических оболочек: <http://ubuntunews.ru/articles/20-altemativ-rabochemu-stolu-unity.html>.

2.7. Установка пароля root

Пользователь *root*— это главный пользователь в системе (как Администратор в Windows). Постарайтесь не забыть его пароль! В некоторых дистрибутивах окно для ввода *пароля root* совмещено с окном добавления пользователя, некоторые дистрибутивы (например, Fedora) выводят отдельно окно для задания пароля *root*, а openSUSE предлагает создать обычного пользователя, и при этом его пароль предлагается использовать в качестве пароля *root* (рис. 2.21). Это довольно удобно, но

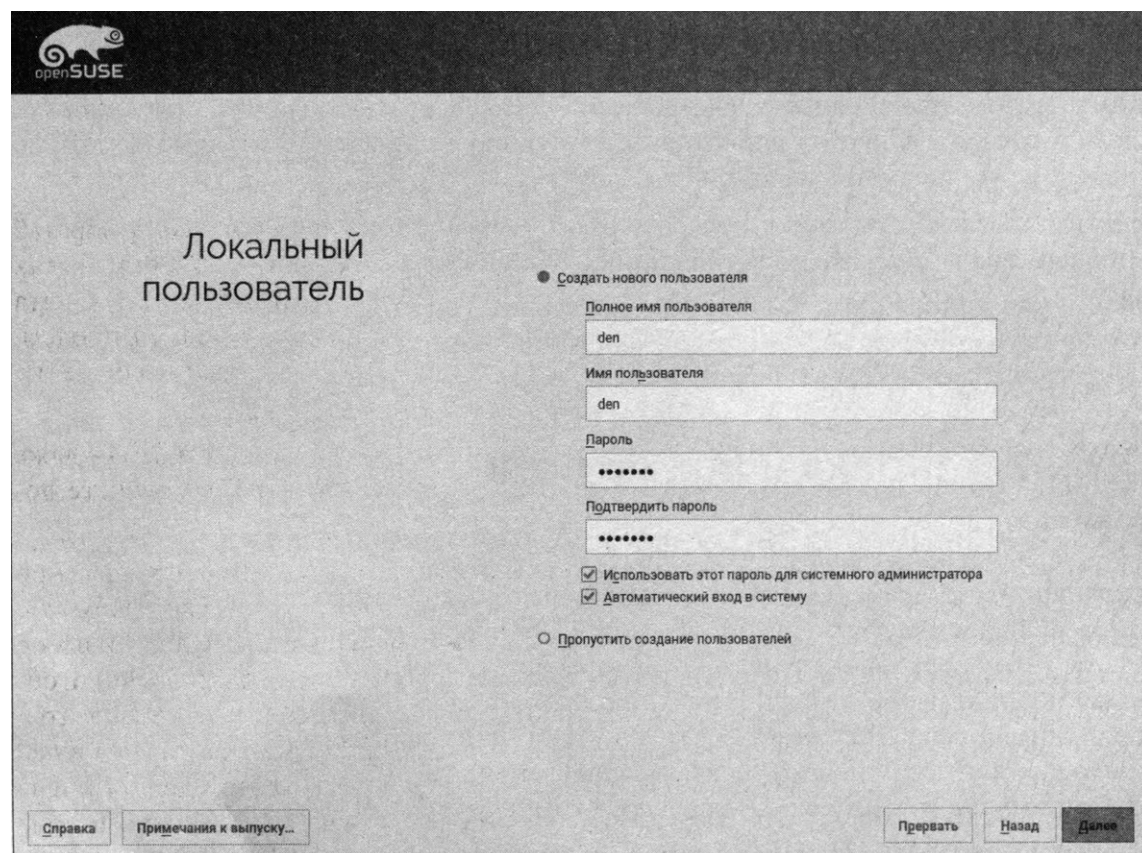


Рис. 2.21. openSUSE 42.3: создание пользователя и задание пароля *root*

с точки зрения безопасности лучше, чтобы пароль root не совпадал с пользовательским паролем. В Ubuntu вообще особая ситуация — учетная запись пользователя root по умолчанию отключена, а создаваемая учетная запись является учетной записью администратора (подробнее об этом мы поговорим в *главе 6*).

ПЕРЕКЛЮЧИТЬСЯ НА АНГЛИЙСКУЮ РАСКЛАДКУ!

При установке Fedora сразу же активируется выбранная в начале установки раскладка клавиатуры (в наших краях — русская), поэтому при вводе имени и пароля пользователя может возникнуть проблема — как переключиться на английскую раскладку, которая необходима для ввода имени пользователя и его пароля? Так вот, используйте для этого комбинацию клавиш `<Alt>+<Shift>`. В иных дистрибутивах могут использоваться другие комбинации, например, `<Ctrl>+<Shift>` или `<Shift>+<Shift>`.

2.8. Создание учетных записей пользователей

При установке системы вам необходимо создать (см. рис. 2.21) хотя бы одну пользовательскую учетную запись — с ее помощью вы будете осуществлять вход в систему. Многие современные дистрибутивы запрещают вход в систему от имени root, поэтому вы будете использовать именно эту созданную при установке учетную запись.

2.9. Порядок установки операционных систем

Как уже отмечалось ранее, сначала нужно устанавливать Windows, а потом Linux. Дело в том, что Windows при установке узурпирует главную загрузочную запись, и после ее установки Linux вы уже не запустите.

При установке Linux такого не происходит — загрузчик Linux настраивается так, чтобы вы могли запускать как Linux, так и Windows.

Если вы планируете установить несколько версий Windows — например, Windows 10 и Windows 7/8, то сначала установите все необходимые вам версии Windows, а уже затем — Linux.

2.10. Установка Linux по сети

2.10.1. Немного о загрузке и установке по сети

Большинство современных компьютеров умеют загружаться по сети — BIOS компьютера находит загрузочный PXE-сервер (Preboot Execution Environment) и загружает с него операционную систему. В этом случае компьютеру для загрузки операционной системы не нужен ни жесткий диск, ни любой другой носитель информации. Обычно такая схема используется на «тонких клиентах» — компьютерах, не имеющих жесткого диска (с целью удешевления), загрузка операционной системы на которых производится с центрального компьютера сети.

В этом разделе мы рассмотрим настройку и использование PXE-сервера, предназначенного для загрузки программы установки Linux.

Установка по сети может понадобиться в двух случаях:

- **при установке Linux на мини-ноутбук, не оснащенный приводом DVD** — покупать USB-привод DVD только для установки Linux не очень-то рационально, правда? Может быть и так, что ваш старенький ноутбук оснащен лишь CD-приводом, тогда как большинство современных дистрибутивов распространяются на DVD;
- **при установке Linux на целый парк компьютеров** — тут все просто: компьютеров много, а установочный диск всего один, поэтому установка по сети позволит значительно сэкономить время. В среднем установка Linux (без настройки) занимает полчаса. 10 компьютеров подряд — это уже более 5 часов работы. А вот при наличии загрузочного сервера, на настройку которого у вас уйдет минут 20, эти 10 компьютеров будут готовы к работе всего за 1 час.

Как видите, PXE-сервер — весьма полезная в хозяйстве вещь. В этой книге, правда, подробно рассматривать создание полноценного PXE-сервера мы не станем.

2.10.2. Подготовка загрузочного сервера

Настройку загрузочного сервера мы рассмотрим на примере Ubuntu. Поскольку установка по сети — довольно специфическая операция, и она нужна далеко не всем пользователям, то я не вижу особой необходимости рассматривать установку PXE-сервера в разных дистрибутивах, — ведь в другом дистрибутиве можно все сделать «по образу и подобию».

Установка DHCP-сервера

Первым делом надо установить DHCP-сервер — в Ubuntu это делается командой:

```
$ sudo apt-get install dhcp-server
```

Затем откройте файл `/etc/dhcp3/dhcpd/dhcpd.conf` и добавьте в него следующие строки:

```
host pxeinstall {  
    hardware ethernet xx:xx:xx:xx:xx:xx:xx;  
    filename "pxelinux.0";  
}
```

ИНСТРУКЦИЯ HARDWARE

Об инструкции `hardware` следует сказать особо. По большому счету — она не нужна. Но если вы запускаете DHCP-сервер в реальной сети, где уже наверняка есть другой DHCP-сервер, а вам надо установить Linux всего на один компьютер, тогда замените символы `xx` в инструкции `hardware` MAC-адресом сетевого адаптера, установленного на компьютере, на который нужно поставить Linux.

Если же вы настраиваете всю сеть компьютеров или же полноценный PXE-сервер, тогда можно инструкцию `hardware` удалить — чтобы ваш сервер могли использовать все компьютеры сети.

С другой стороны, указать MAC-адреса потенциальных клиентов — это хорошая идея с точки зрения безопасности. Но если вы разворачиваете свой PXE-сервер

только для установки операционной системы, нет никакой необходимости тратить время на определение MAC-адресов всех компьютеров сети. Вот, когда надо настроить полноценный PXE-сервер, может и нужно указать адреса «тонких клиентов», чтобы никто другой не смог использовать ваш сервер для загрузки. Тут уже решать вам...

Сохраните файл конфигурации DHCP-сервера и перезапустите сервер:

```
$ sudo /etc/init.d/dhcpd restart
```

Настройка TFTP-сервера

Следующий шаг — настройка TFTP-сервера (Trivial File Transfer Protocol), на котором будет размещен образ операционной системы. В нашем случае — это установочный образ Ubuntu.

Установить TFTP-сервер можно командой:

```
$ sudo apt-get install tftpd-hpa
```

После установки сервера отредактируйте ваш файл **/etc/inetd.conf**. Убедитесь, что в нем есть следующая строка (и что она раскомментирована):

```
tftp dgram udp wait root /usr/sbin/in.tftpd /usr/sbin/in.tftpd -s  
/var/lib/tftpboot
```

Поскольку TFTP-сервер работает не автономно, а через сервер `inetd`, то для запуска TFTP-сервера нужно перезапустить сервер `inetd`:

```
$ sudo /etc/init.d/inetd restart
```

В современных дистрибутивах вместо сервера `inetd` используется суперсервер `xinetd`, поэтому надо отредактировать его конфигурационный файл — **/etc/xinetd.conf**. Добавьте в него следующие строки:

```
service tftp  
{  
    socket_type = dgram  
    protocol = udp  
    wait = yes  
    user = tftp  
    server = /usr/sbin/in.tftpd  
    server_args = -1 /var/lib/tftpboot  
    only_from = client.test.net  
}
```

И перезапустите `xinetd`:

```
$ sudo /etc/init.d/xinetd restart
```

Загрузка установочного образа

Теперь нам нужно загрузить специальный установочный образ, рассчитанный на установку по сети. Подключитесь к Интернету и введите следующие команды:

```
$ mkdir net-install
$ sudo lftp -c "open
http://archive.ubuntu.com/ubuntu/dists/dapper/main/installer-i386/current/
images/; mirror net-install/"
```

Первая команда создаст каталог `net-install`, а вторая — загрузит в нее установочный образ Ubuntu.

Почти все готово, и в каталог `net-install` загружены файлы, необходимые для установки Linux по сети. Но давайте вспомним наш файл `/etc/inetd.conf` (или `xinetd.conf`). Конфигурация TFTP предполагает, что все файлы, доступные по протоколу TFTP, должны быть расположены в каталоге `/var/lib/tftpboot`. Поэтому нам нужно скопировать туда файлы из каталога `net-install`:

```
$ sudo cp -a net-install/* /var/lib/tftpboot
$ sudo cd /var/lib/tftpboot
$ sudo tar xzf netboot.tar.gz
```

Вот и все — ваш PXE-сервер готов к работе.

2.10.3. Настройка клиента

Настраивать клиент, т. е. компьютер, на который вы будете устанавливать Linux, очень просто, — достаточно зайти в его BIOS и установить загрузку по сети. Но загружаться по сети умеют не все компьютеры...

Что делать, если у вас старый компьютер, который не умеет загружаться по сети? Можно попытаться перепрошить BIOS — новая версия наверняка будет поддерживать загрузку по сети. Если перепрошивать BIOS нежелательно, или вы не можете найти подходящую версию BIOS именно для вашего компьютера, тогда вам будет проще изготовить специальную загрузочную дискету, загрузиться с нее, а загрузчик уже сам найдет PXE-сервер и запустит процесс установки.

Создать загрузочную дискету можно с помощью команды `mknbi` (страница руководства по этой команде находится тут: <http://manpages.ubuntu.com/manpages/intrepid/man1/mknbi.html>).

ЗАГРУЗОЧНАЯ ФЛЕШКА

Учитывая, что на большинстве современных ноутбуков уже нет дисководов для дискет, взамен загрузочной дискеты лучше всего изготовить загрузочную флешку, для создания которой используется программа **Система | Администрирование | Startup disk creator**. С другой стороны, все современные машины поддерживают загрузку по сети, так что вам не стоит особо беспокоиться по этому поводу.

2.11. Проблемы при установке

2.11.1. Проблема с APIC

APIC (Advanced Programmable Interrupt Controller) — улучшенный программируемый контроллер прерываний. Поскольку контроллер прерываний «улучшенный», то проблем быть с ним не должно, но на практике это далеко не так. Одним словом,

проблемы с APIC в Linux возникают довольно часто. При загрузке система может зависнуть. Вы можете увидеть сообщение о проблеме с APIC, а можете и не увидеть его. Если сообщение есть, то оно будет выглядеть примерно так:

kernel panic — not syncing: IO-APIC + tinier doesn't work! Boot with apic=debug and send areport. Then try booting with the 'noapic' option

Решить проблему помогает параметр ядра `noapic`, позволяющий SMP-ядру не использовать расширенные возможности контроллера прерываний в многопроцессорных машинах. Обратите внимание — ядро само подсказало, чего ему не хватает!

Подробно о передаче параметров ядру мы поговорим в *главе 20*. А пока, находясь в меню загрузчика GRUB, нажмите для редактирования параметров ядра клавишу `<e>` (или `<F5>` в случае с openSUSE) и просто добавьте в список параметров команду `noapic` — проблема должна исчезнуть. Если этот параметр вам помог, нужно добавить его в файл `/boot/grub/menu.lst` или отключить APIC в BIOS.

2.11.2. Ошибка:

kernel panic:VFS: Unable to mount root fs

Появление такого сообщения означает, что ядро не может подмонтировать корневую файловую систему. Понятно, что дальнейшее продолжение работы невозможно. Наиболее вероятная причина — повреждение установочного диска. Если с поверхностью диска все в порядке (она не поцарапана, отсутствуют следы грязи и/или жира), тогда причина в ошибке при записи DVD. Выход один — раздобыть другой установочный DVD и загрузиться с него.

2.11.3. Проблемы с некоторыми LCD-мониторами

Если ваш LCD-монитор подключен к DVI-разъему видеокарты, и с ним возникают проблемы: не поддерживается максимальное разрешение, низкое качество изображения, самопроизвольное выключение питания монитора — попробуйте передать ядру параметр `nofb`. Если это поможет решить проблему, «пропишите» этот параметр в конфигурационном файле загрузчика (об этом мы также поговорим далее).

Что делать, если параметр `nofb` не помог? Просто подключите монитор к аналоговому разъему видеокарты — все должно заработать нормально.

2.11.4. Сообщение *Probing EDD* и зависание системы

Некоторые дистрибутивы при загрузке могут вывести сообщение **Probing EDD**, и на этом загрузка остановится. Изначально я столкнулся с этой проблемой при установке openSUSE 11.x на ноутбук Toshiba. Но, судя по письмам пользователей, такая проблема проявлялась и в Fedora 10, и в Mandriva 2009 при использовании определенных жестких дисков. Если вы увидели это сообщение, и система зависла, пере-

дайте ядру параметр `edd=off`. Самое интересное — уже в 2017 году при установке Ubuntu на ноутбук ASUS возникла та же ошибка, и пришлось опять применить параметр `edd=off`, иначе система отказывалась загружаться.

2.11.5. Установка Linux на HP Mini 2133 (проблема с ACPI)

При установке Linux на этот нетбук может возникнуть проблема с ACPI (Advanced Configuration and Power Interface) — системой управления питанием и энергосбережением. В этом случае нужно или отключить ACPI (параметр ядра `acpi=off`), или обновить BIOS нетбука до версии F.06. Отключение ACPI на нетбуке можно воспринимать только как временную меру (до обновления BIOS) — это все равно, что не включать кондиционер в жару (при условии наличия самого кондиционера!). Однако, пока вы не обновите BIOS (при отсутствии опыта эту операцию лучше производить в сервисном центре), ACPI выключить можно. А после обновления BIOS ваша система сможет работать нормально.

2.11.6. Проблема с ACPI на Fujitsu Siemens Esprimo Mobile u9200

Проблема с ACPI есть еще у одного ноутбука. На ноутбуке Esprimo Mobile u9200 неправильно работает подсветка. Чтобы все восемь уровней подсветки были доступны, нужно передать ядру параметр `acpi_backlight=vendor`. Понятно, что этот параметр первый раз надо просто передать ядру, чтобы проверить, правильно ли работает подсветка, а затем его следует добавить в конфигурационный файл загрузчика, чтобы не вводить его каждый раз при запуске Linux.

2.11.7. Переход в режим паники компьютера с процессором AMD64

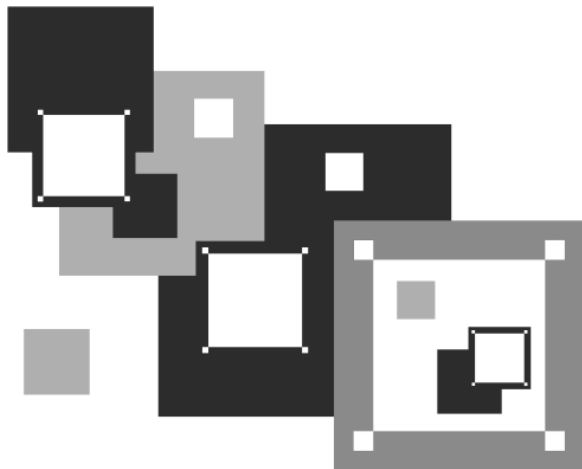
На компьютере с процессором AMD64 ядро переходило в режим паники, и установить Linux было невозможно. При этом на экране красовалось следующее сообщение:

kernel panic — not syncing: IO-APIC + timer doesn't work! Boot with apic=debug and send report. Then try booting with the 'noapic' option

Помог решить проблему параметр ядра `noapic`, позволяющий SMP-ядру не использовать расширенные возможности контроллера прерываний в многопроцессорных машинах. Обратите внимание — ядро снова само подсказало, чего ему не хватает! Впрочем, ради справедливости нужно отметить, что указанная проблема характерна для первых версий ядра линейки 2.6.x. Новые версии ядра с процессорами AMD64 работают нормально.

2.11.8. Проблема с механизмом Enhanced Disk Device (EDD)

Современные версии ядра Linux поддерживают механизм Enhanced Disk Device (EDD) polling, позволяющий собирать информацию о всех дисковых устройствах, с которых возможна загрузка. Собранная информация потом сохраняется в каталоге `/sys`. Иногда с EDD возникает проблема — при загрузке Linux пользователь видит сообщение **Updating EDD...**, и компьютер как бы зависает. В некоторых случаях загрузка секунд через 30-40 продолжается, а в некоторых вообще и не начинается. Суть происходящего в том, что при загрузке система обнаруживает «лишние» загрузочные устройства. В этом случае вам поможет параметр ядра `edd=skipmbr`. Если проблема таким путем не устраняется, попробуйте передать ядру параметр `edd=off`, вообще отключающий механизм EDD.

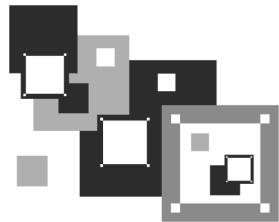


ЧАСТЬ II

Основные сведения о Linux

Вторую часть книги можно рассматривать в качестве своеобразного фундамента знаний любого линуксоида — да будет позволено мне нас, пользователей Linux, так называть. Ведь здесь мы рассмотрим вход в систему, ее базовую настройку, особенности файловой системы Linux, командный интерпретатор `bash`, поговорим о пользователях, группах и правах доступа, а также разберемся с различными системами управления пакетами.

ГЛАВА 3



Сразу после установки...

3.1. Вход в систему и завершение работы

По умолчанию в современных дистрибутивах при входе в систему запускается *графический менеджер регистрации* (рис. 3.1), в окне которого требуется указать имя пользователя и пароль, — после этого загрузится установленная в вашем дистрибутиве по умолчанию графическая среда (обычно это KDE или GNOME).

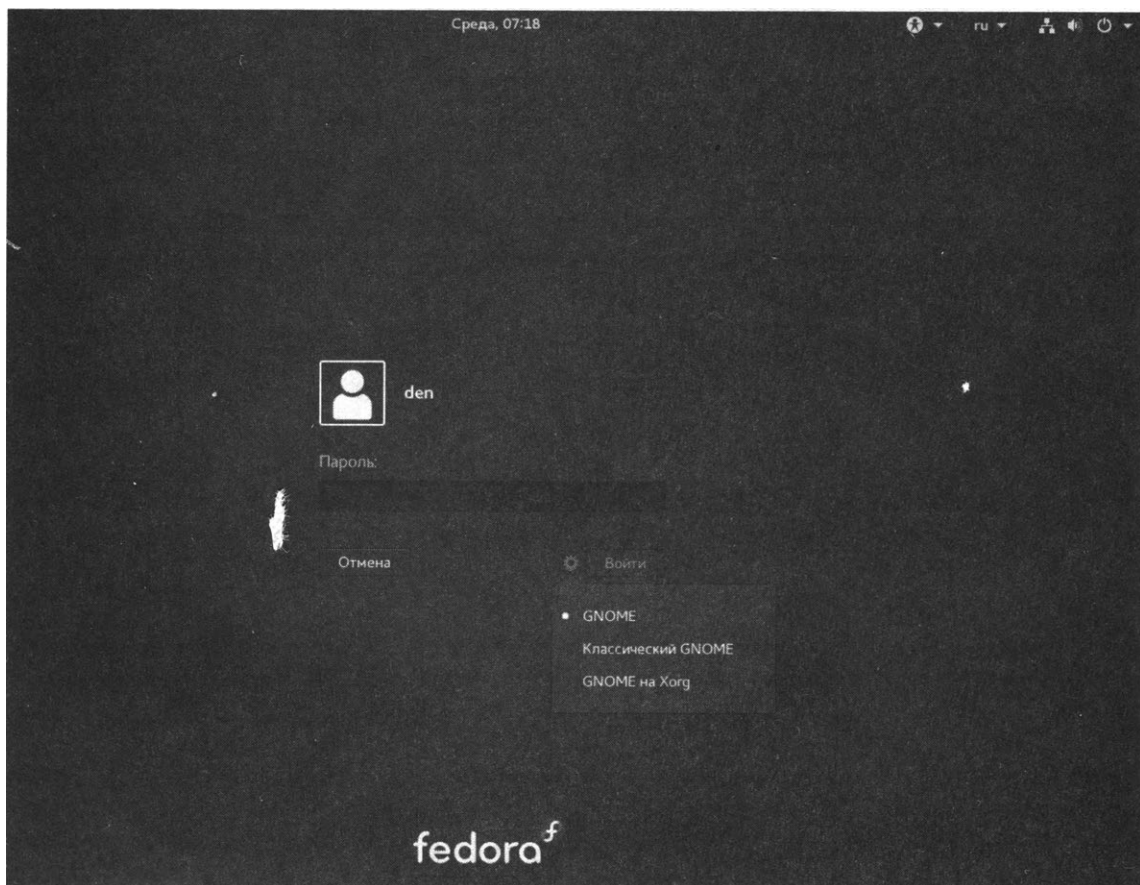


Рис. 3.1. Fedora 26: графический вход в систему

Конечно, может быть загружена и какая-либо другая графическая среда по вашему выбору. Для этого надо нажать соответствующую кнопку выбора типа сеанса, имеющуюся в окне регистрации. В зависимости от дистрибутива она может называться **Тип сеанса** или **Сеанс**, а может быть представлена графической пиктограммой — например, шестеренкой, как в Fedora (см. рис. 3.1), или гаечным ключом, как в openSUSE 13.2. В openSUSE 42.3 (рис. 3.2) в нижнем левом углу экрана вы обнаружите соответствующий список.

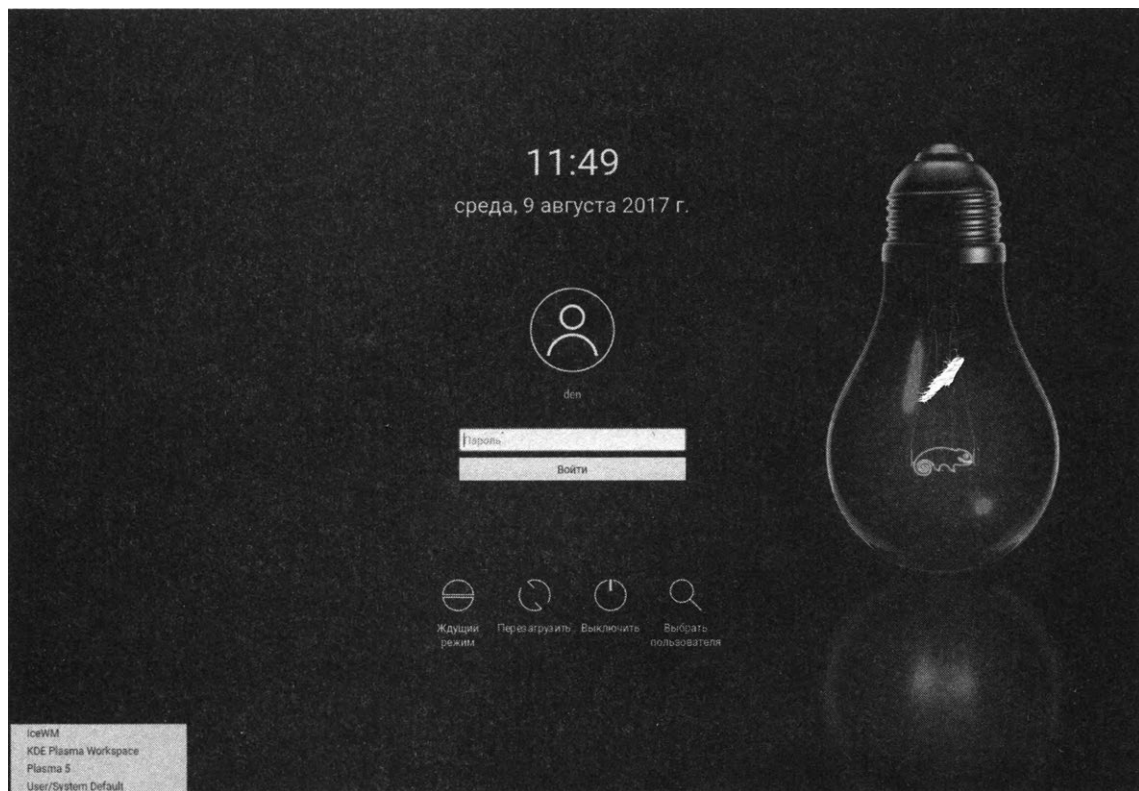


Рис. 3.2. openSUSE 42.3: выбор типа сеанса

ДИСТРИБУТИВ SLACKWARE - ИСКЛЮЧЕНИЕ

Однако из всех правил могут быть исключения. Пример тому дистрибутив Slackware — в нем сначала придется выполнить вход в консоли (рис. 3.3), а потом для запуска графического интерфейса ввести команду `startx`.

Забегая немного вперед, отмечу, что консольный режим (см. *разд. 3.6*), несмотря на свой столь устрашающий вид, оказывается весьма полезен в практической работе с Linux, в чем мы впоследствии не раз убедимся.

Итак, войдя в систему, вы попадаете в *графический режим*. Для того чтобы перейти из графического режима в *консоль*, нажмите клавиатурную комбинацию `<Ctrl>+<Alt>+<Fn>`, где *n* — номер консоли (от 1 до 6). То есть, чтобы перейти на первую консоль, нужно нажать `<Ctrl>+<Alt>+<F1>`, на вторую — `<Ctrl>+<Alt>+<F2>` и т. д. Обратите внимание, что так можно перейти в консоль только

из графического режима. Если вы уже находитесь в консоли, то для переключения между консолями служат комбинации клавиш `<Alt>+<F1>` ... `<Alt>+<F6>`, а также `<Alt>+<F7>` — возвращающая вас в графический режим. Для лучшего запоминания эти комбинации клавиш сведены в табл. 3.1.

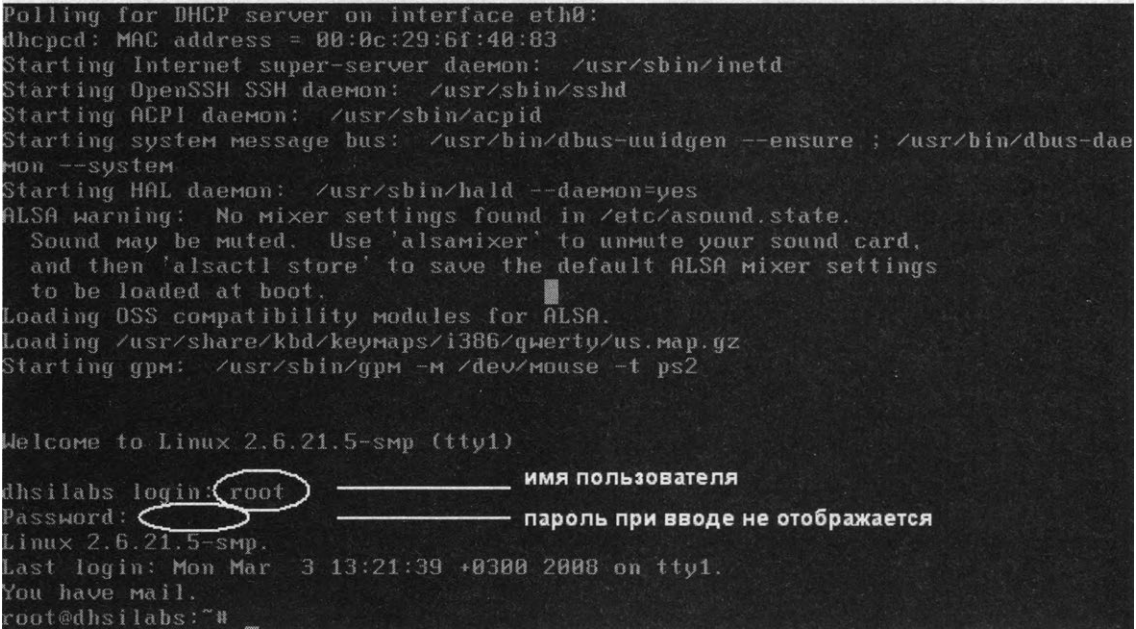


Рис. 3.3. Slackware: регистрация в консоли

Таблица 3.1. Клавиши переключения между консолями и графическим режимом

Комбинация клавиш	Предназначение
<code><Ctrl>+<Alt>+<Fn></code> (n — от 1 до 6)	Переключение из графического режима в консоль с номером n
<code><Alt>+<Fn></code> (n — от 1 до 6)	Переключение между консолями
<code><Alt>+<F7></code>	Переключение из консоли в графический режим

Для выхода из консоли (чтобы ею никто не воспользовался во время вашего отсутствия) предусмотрена команда `logout`, можно выйти из консоли и по команде `exit`. Для перезагрузки компьютера надо отдать команду `reboot`. Существуют также две команды завершения работы: `halt` и `poweroff`:

- команда `halt` завершает работу системы, но не выключает питание. Вы увидите сообщение **System is halted**, свидетельствующее о возможности выключения питания. Эта команда предназначена для старых компьютеров, не поддерживающих расширенное управление питанием;
- команда `poweroff` завершает работу системы и выключает ее питание.

Самая «продвинутая» команда — `shutdown` — позволяет завершить работу или перезагрузить систему в назначенное время. Предположим, вы хотите уйти пораньше, но компьютер нужно выключить ровно в 19.30 (вдруг некоторые пользователи задержались на работе, а вы выключите сервер, — вряд ли это им понравится). Вот тут-то вам и поможет команда `shutdown`:

```
$ shutdown -h 19:30 [сообщение]
```

РЕШЕТКА (#) И ДОЛЛАР (\$)

Здесь и далее решетка (#) означает, что команда должна быть выполнена от имени пользователя `root`. Если перед командой ничего не указано или же указан символ доллара (\$), команду можно выполнить от имени обычного пользователя. Ранее команды завершения работы системы требовали полномочий `root`, что подчеркивало ориентацию Linux на серверы, — только администратору дозволялось завершить работу сервера. Сейчас же команды `shutdown`, `reboot` и `poweroff` могут выполнить обычные пользователи.

Сообщение [сообщение] можно и не указывать — все равно Windows-пользователи его не увидят.

Если нужно завершить работу системы прямо сейчас, вместо времени укажите `now`:

```
$ shutdown -h now
```

Для перезагрузки системы есть опция `-r`:

```
$ shutdown -r now
```

КОМАНДНАЯ СТРОКА И ТЕРМИНАЛ

В этой книге вы часто будете вводить различные команды — например, команды включения/выключения или запуска тех или иных конфигураторов системы. Для выполнения какой-либо команды нужно нажать клавиатурную комбинацию `<Alt>+<F2>`, в открывшейся *командной строке* ввести эту команду и нажать клавишу `<Enter>`. Можно также использовать эмулятор консоли — *терминал*, кнопку запуска которого вы найдете в основном меню графической среды.

3.2. О графическом интерфейсе Linux

3.2.1. GNOME и KDE

Долгого и мучительного экскурса в историю развития графических интерфейсов здесь не будет, отметим только, что основных таких интерфейсов существует два: GNOME и KDE. По умолчанию GNOME используется в Fedora, Ubuntu, Debian, а KDE — в openSUSE.

Основные элементы управления классического интерфейса GNOME: меню, панель задач и область уведомлений (выражаясь терминологией Windows, что будет привычнее большинству пользователей) — находятся в верхней части экрана (рис. 3.4).

Классический интерфейс GNOME за годы его существования несколько «приелся», и в 2010 году компания Canonical Ltd, курирующая операционную систему Ubuntu Linux, в версии Ubuntu 10.10 Notebook Edition представила свою оболочку для

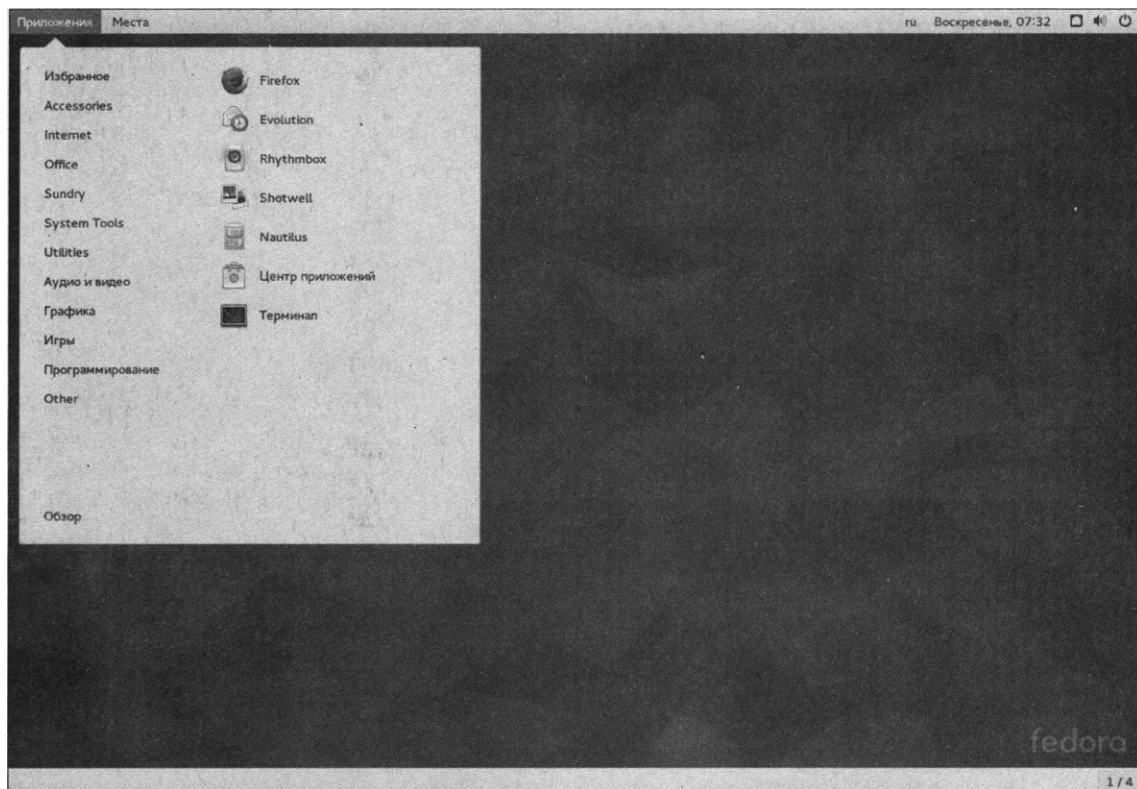


Рис. 3.4. Классический GNOME (какая-то старая версия Fedora)

GNOME — Unity¹, которая, начиная с 11-й версии Ubuntu, принята в этой системе по умолчанию. Как можно видеть, в верхней части экрана Unity находится меню приложения, а всевозможные кнопки быстрого доступа к тем или иным возможностям расположены на ленте слева (рис. 3.5).

Оболочка Unity многим пришлась по вкусу, и, начиная с версии 3.8, GNOME предоставляет теперь на выбор два интерфейса: и классический, и современный — не правда ли, современный интерфейс GNOME в Fedora 26 (рис. 3.6) и Unity в Ubuntu похожи, как две капли воды?

В свое время считалось, что GNOME — интерфейс менее эффектный, но и менее «прожорливый», а KDE — красивее (рис. 3.7), но более требовательный к системным ресурсам. Впрочем, сейчас по части системных требований они если и различаются, то только на бумаге, — субъективно на современных компьютерах пользователь не почувствует разницы в производительности между KDE и GNOME.

Какой интерфейс выбрать? Если вы впервые обратились к Linux, выбирайте используемый по умолчанию. Если же вы уже не новичок, то выбор очевиден — тот, который вам больше нравится.

¹ Ходили слухи, что Canonical откажется от Unity, но вместо этого встречаем Unity 8 — в Ubuntu 17.04. Правда, 8-я версия мало чем визуальнo отличается от всех предыдущих, но факт есть факт.

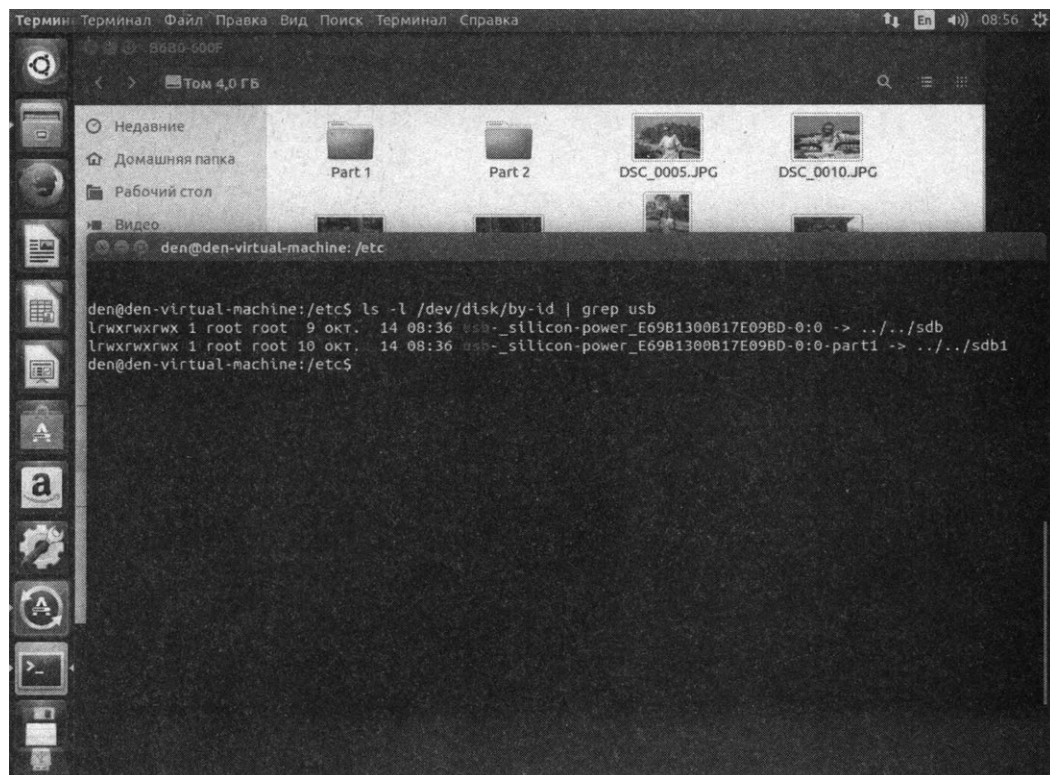


Рис. 3.5. Ubuntu 17.04: оболочка Unity



Рис. 3.6. Fedora 26: современный интерфейс GNOME

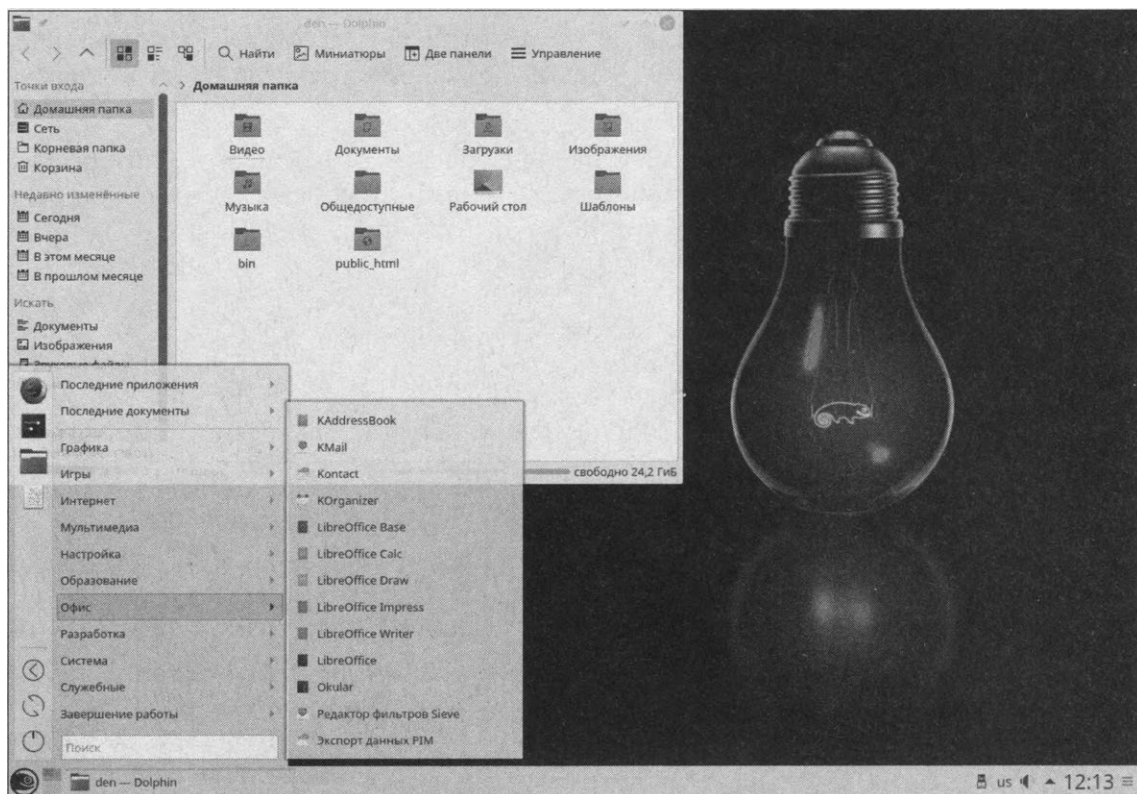


Рис. 3.7. openSUSE 42.3: рабочий стол KDE

3.2.2. Установка альтернативного графического интерфейса

В любом дистрибутиве, независимо от используемого по умолчанию графического интерфейса, можно установить *альтернативный*. Так, в Ubuntu для установки KDE следует выполнить в терминале команду:

```
sudo apt install kubuntu-desktop
```

ВЫПОЛНЕНИЕ КОМАНДЫ В ТЕРМИНАЛЕ

Для выполнения команды в терминале щелкните правой кнопкой мыши на свободной области рабочего стола, выберите команду *Открыть терминал*, введите в поле ввода терминала команду и нажмите клавишу *<Enter>*.

Эта команда установит интерфейс KDE со стандартным набором приложений. Почему пакет называется *kubuntu-desktop*? Потому что существует несколько разных редакций Ubuntu: *Kubuntu*, *Edubuntu*, *Lubuntu*, *Mythbuntu*, *Xubuntu*, *Ubuntu Server* и *Ubuntu GNOME* (см. <http://ubuntu.ru/family>). Стандартный дистрибутив Ubuntu поставляется с *GNOME/Unity*, а *Kubuntu* — с *KDE*.

Чтобы получить KDE с полным набором приложений, установите пакет *kubuntu-full*:

```
sudo apt install kubuntu-full
```

Однако, как показывает практика, если вы хотите работать с KDE, рациональнее установить редакцию Kubuntu сразу — так дисковое пространство будет использоваться более эффективно.

Если же установленная у вас система Ubuntu не содержит графического интерфейса вовсе (например, это редакция Ubuntu Server), и вам потребовалось, чтобы он был, установите пакет `ubuntu-desktop`:

```
sudo apt install ubuntu-desktop
```

В Fedora для установки KDE нужно ввести команду:

```
$ sudo dnf install @kde-desktop
```

Можно также установить в Fedora и следующие графические интерфейсы: Cinnamon, MATE, XFCE, LXDE:

```
sudo dnf install @cinnamon-desktop
sudo dnf install @mate-desktop
sudo dnf install @xfce-desktop
sudo dnf install @lxde-desktop
```

Все эти интерфейсы отличаются ограниченной функциональностью и подходят для очень слабых компьютеров. Более или менее удобным из них считается LXDE.

3.2.3. Основные элементы интерфейса GNOME

Установив openSUSE, в ее интерфейсе вы сможете ориентироваться сразу — расположение элементов и их назначение такое же, как и в Windows. А вот современные версии GNOME заслуживают отдельного разговора.

На рис. 3.8 представлен типичный рабочий стол GNOME 3.24. Чтобы получить такое его состояние, нужно нажать кнопку **Обзор** в верхнем левом углу окна. Слева в виде вертикальной ленты расположена т. н. *панель Dash*, содержащая кнопки быстрого доступа к вашим любимым приложениям, а самая нижняя кнопка на ней открывает полный список установленных приложений (рис. 3.9). Чтобы добавить какое-либо из них в Dash, щелкните по нему правой кнопкой мыши и выберите команду **Добавить в избранное**.

В верхней части экрана заметно *поле поиска Найти* — это универсальный поиск, позволяющий найти как приложения, так и файлы, а над ним — *панель уведомлений*. Именно на ней будут показаны всплывающие уведомления GNOME.

В верхнем правом углу окна вы найдете значок изменения языка ввода, а также значки изменения уровня громкости, завершения работы и др. (рис. 3.10) — кроме управления уровнем громкости и завершением работы системы, отсюда можно также управлять подключением к сетям, включать/выключать определение местоположения (т. е. включать/выключать GPS-модуль, если он у вас есть), вызывать экран настроек (кнопка внизу слева) и блокировать экран (кнопка внизу в центре).

Справа на экране обзора находится *панель рабочих столов* (рис. 3.11). Рабочие столы добавляются туда автоматически по заполнению текущего рабочего стола. При

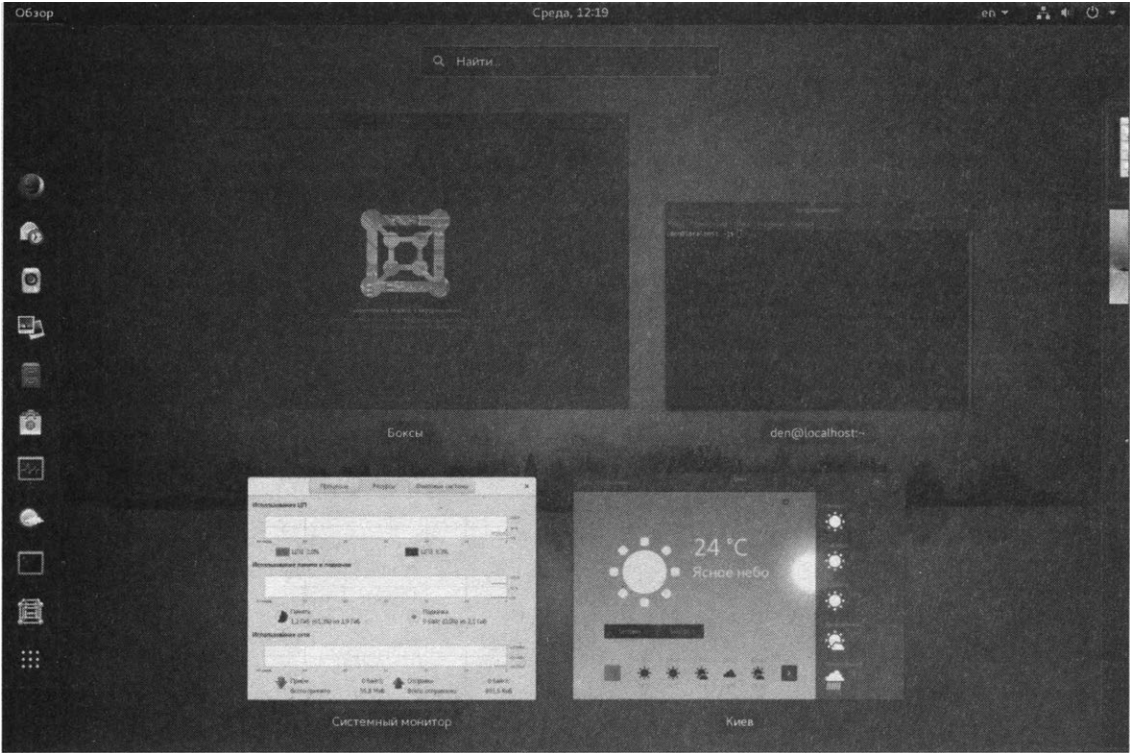


Рис. 3.8. Интерфейс GNOME



Рис. 3.9. Интерфейс GNOME: список установленных приложений

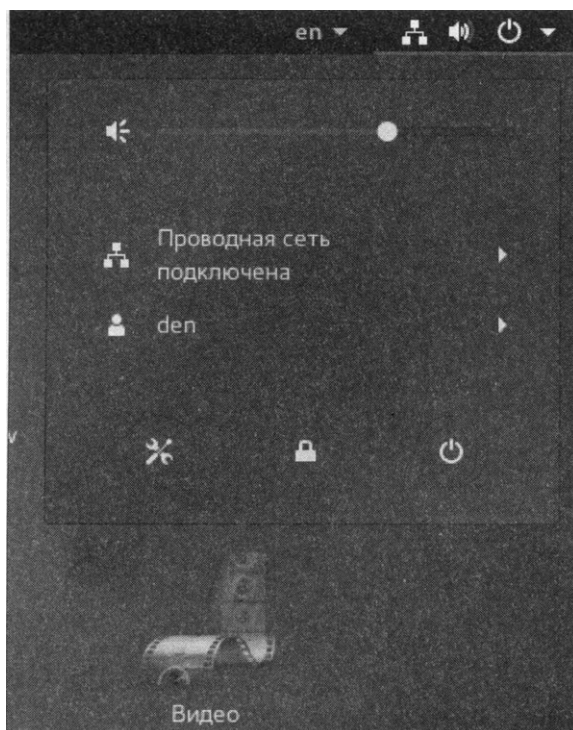


Рис. 3.10. Интерфейс GNOME: изменение громкости, завершение работы и многое другое

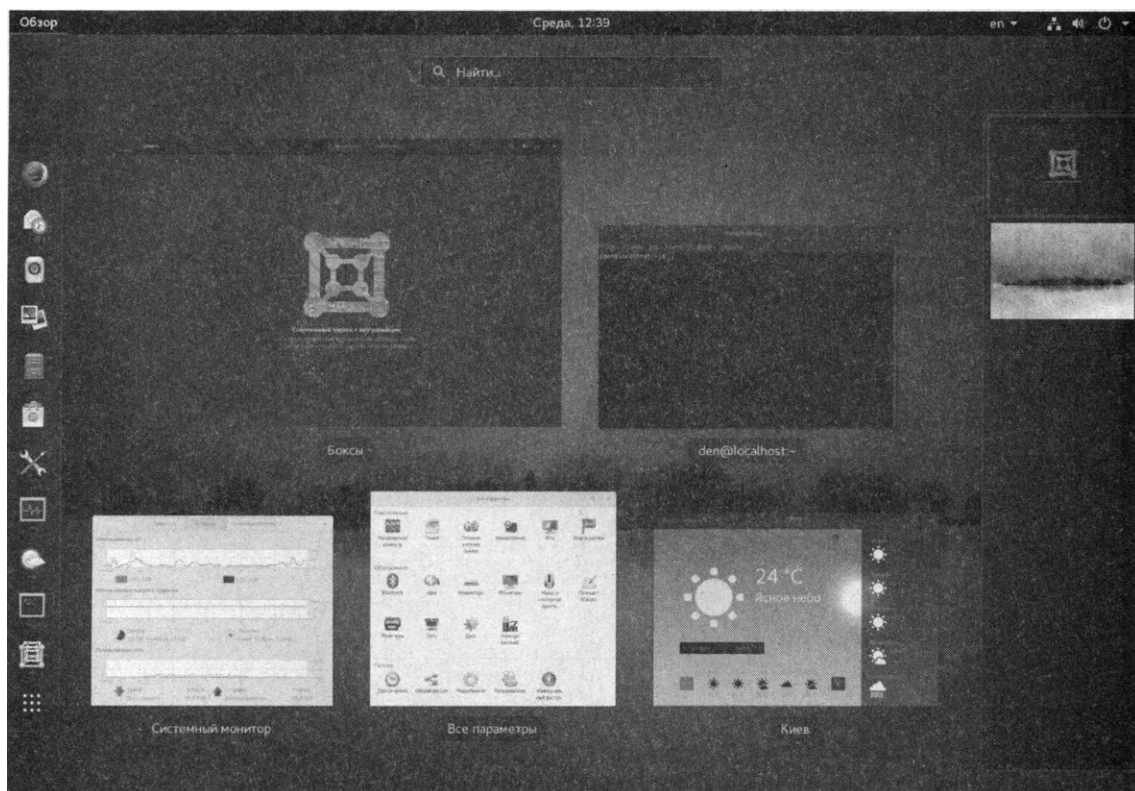


Рис. 3.11. Интерфейс GNOME: панель рабочих столов (справа)

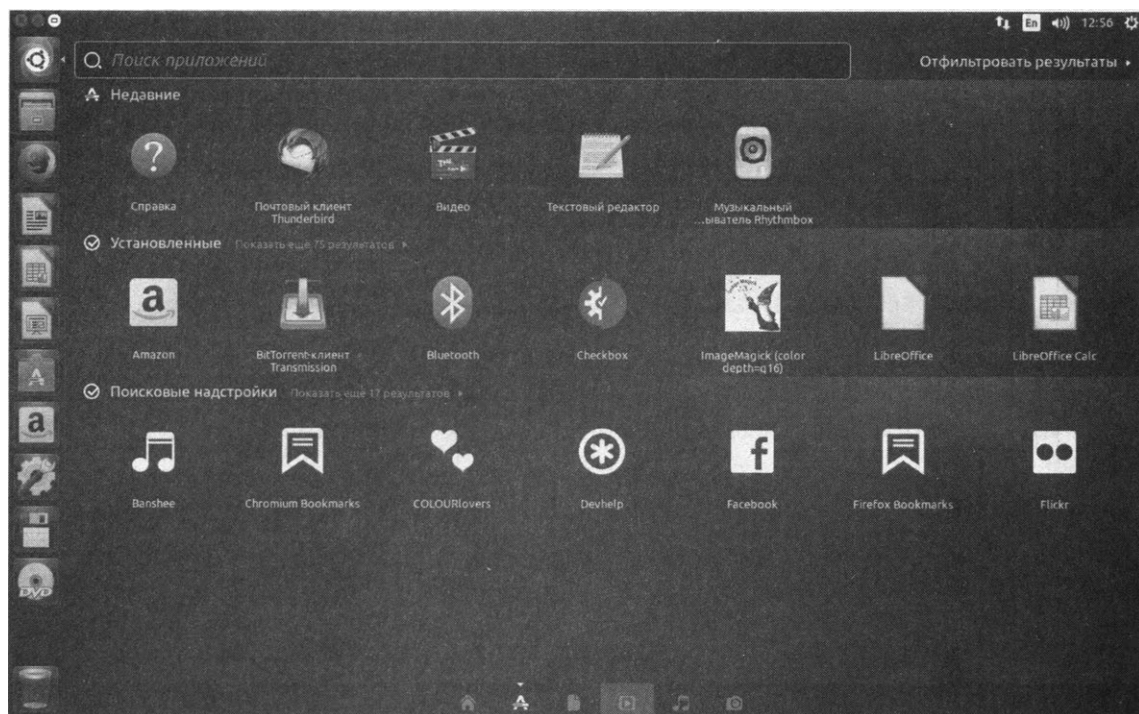


Рис. 3.13. Ubuntu 17.04: главное меню Unity

Чтобы добавить приложение на панель Unity, просто перетащите на нее его значок. А для удаления значка приложения из панели щелкните по нему правой кнопкой мыши и выберите команду **Изъять из панели** (рис. 3.14).

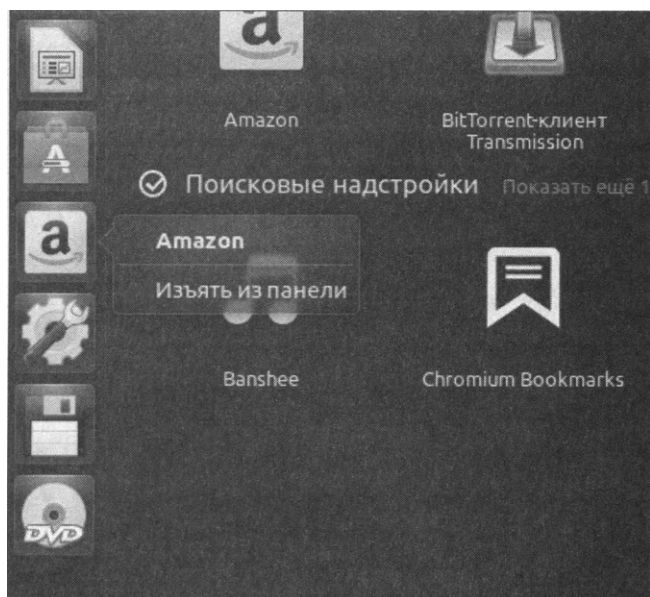


Рис. 3.14. Ubuntu 17.04: изъятие значка из панели Unity

3. 3. Изменение параметров графического интерфейса

Что надо бы изменить в системе сразу после ее установки? Мне представляется разумным отключить блокировку экрана на домашнем компьютере (на предприятии она, может, и не будет лишней), изменить способ переключения языков ввода и, возможно, разрешение экрана (см. главу 13), а также, конечно же, сменить фон рабочего стола. Что ж, приступим.

3.3.1. Отключение блокировки экрана

Блокировка экрана — это самая надоедливая опция графического интерфейса. Стоит отлучиться на минутку, и все — экран заблокирован, и снова приходится вводить пароль. Может быть, в корпоративной среде так и должно быть, но на домашнем компьютере, на мой взгляд, — это совершенно излишняя опция, если, конечно, вы не параноик и не верите в теорию всемирного заговора.

Для отключения блокировки экрана в Fedora нажмите кнопку **Обзор**, затем кнопку **Показать приложения** (самая последняя на панели слева), далее выберите приложение **Параметры** — с изображением гаечного ключа и отвертки (рис. 3.15). В открывшемся окне перейдите в раздел **Конфиденциальность** (рис. 3.16) и выключите параметр **Блокировка экрана** (рис. 3.17).

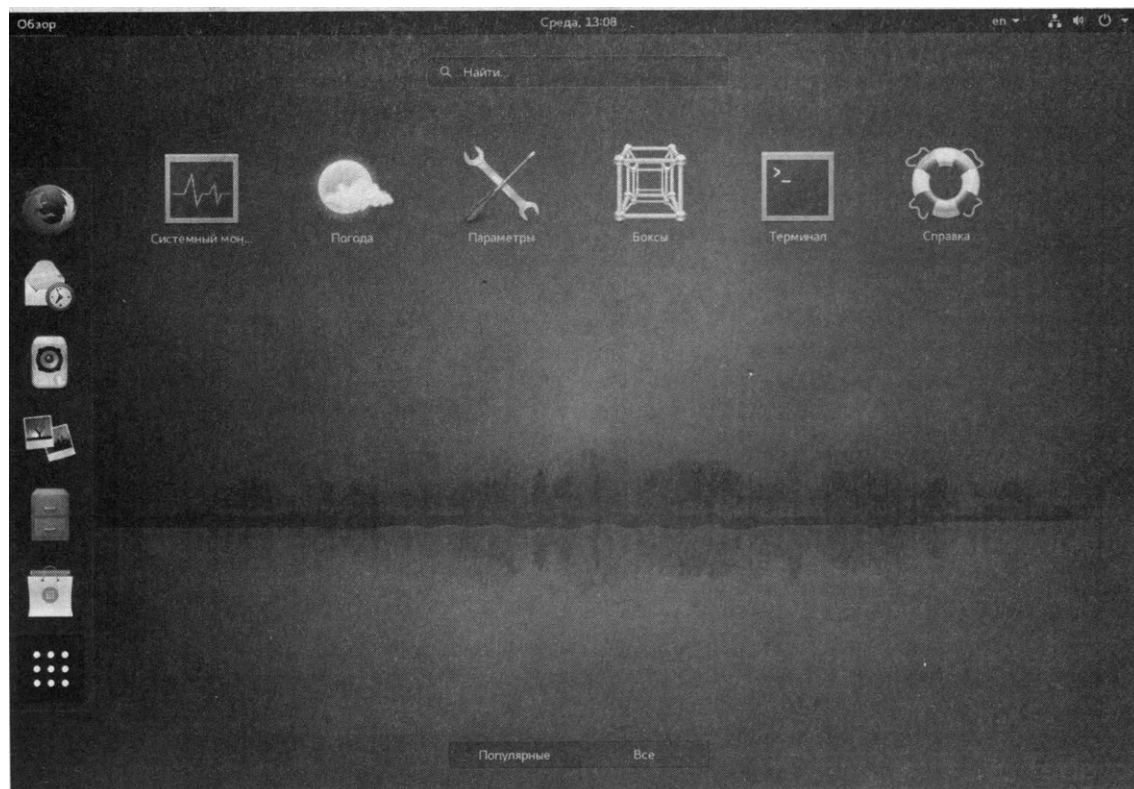


Рис. 3.15. Fedora 26: доступ к параметрам системы



Рис. 3.16. Fedora 26: окно Все параметры

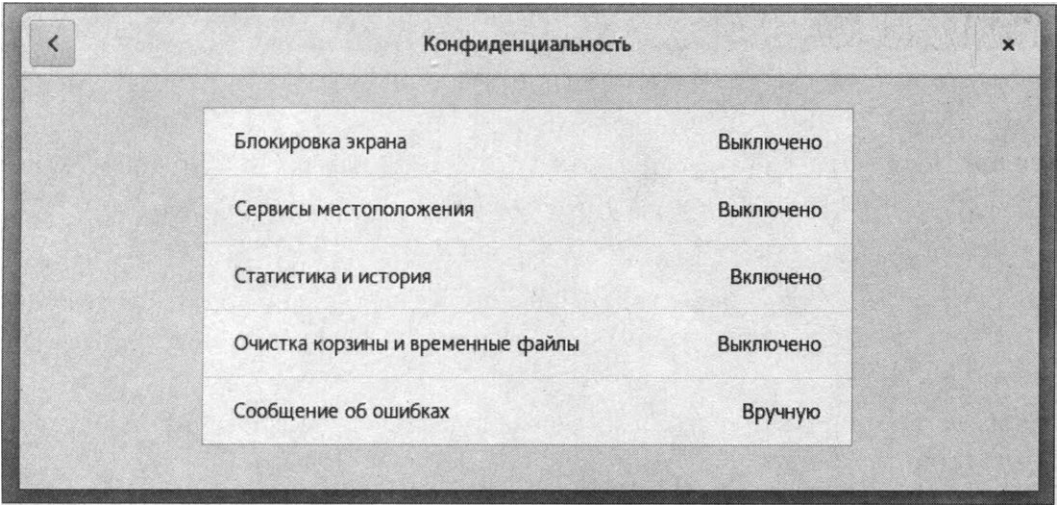
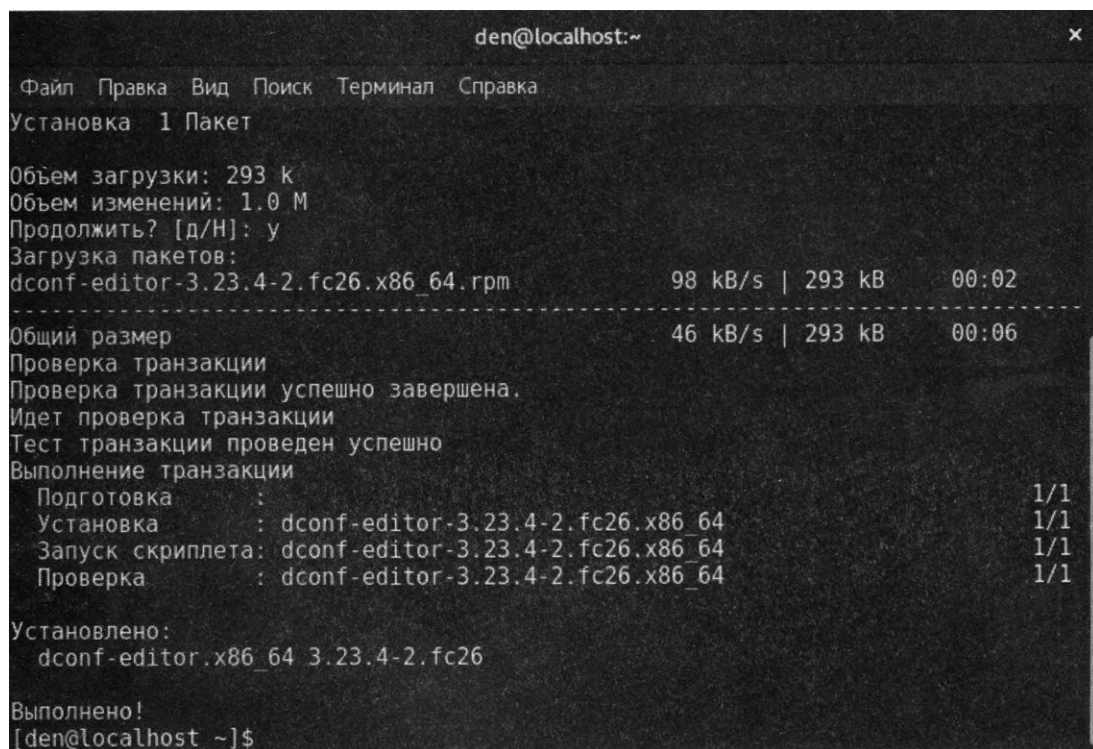


Рис. 3.17. Fedora 26: отключение блокировки экрана

Однако спешу вас разочаровать: да, вам не придется более вводить пароль после каждой блокировки экрана, но, тем не менее, вы по-прежнему будете каждый раз созерцать экран блокировки. Как от него избавиться окончательно?

Здесь нам поможет программа `dconf-editor` — в Linux она служит для редактирования конфигурации GNOME (это в некоторой степени аналог редактора реестра Windows), и, чтобы ее установить (рис. 3.18), выполните в терминале команду:

```
sudo dnf install dconf-editor
```



```
den@localhost:~
Файл Правка Вид Поиск Терминал Справка
Установка 1 Пакет
Объем загрузки: 293 К
Объем изменений: 1.0 М
Продолжить? [д/Н]: у
Загрузка пакетов:
dconf-editor-3.23.4-2.fc26.x86_64.rpm          98 kB/s | 293 kB      00:02
-----
Общий размер                                46 kB/s | 293 kB      00:06
Проверка транзакции
Проверка транзакции успешно завершена.
Идет проверка транзакции
Тест транзакции проведен успешно
Выполнение транзакции
Подготовка :
Установка : dconf-editor-3.23.4-2.fc26.x86_64 1/1
Запуск скриплет: dconf-editor-3.23.4-2.fc26.x86_64 1/1
Проверка : dconf-editor-3.23.4-2.fc26.x86_64 1/1
Установлено:
dconf-editor.x86_64 3.23.4-2.fc26
Выполнено!
[den@localhost ~]$
```

Рис. 3.18. Fedora 26: установка `dconf-editor`

Установив программу `dconf-editor`, запустите ее, нажав комбинацию клавиш `<Alt>+<F2>` и введя в открывшееся поле команду:

```
dconf-editor
```

Ее, в общем-то, можно было бы ввести и в терминале, но мне хочется продемонстрировать вам окно запуска команды, открывающееся по нажатию этой комбинации клавиш (рис. 3.19).

Осталось только найти опцию `disable-lock-screen` и установить для нее значение **True**, как это показано на рис. 3.20.

Чтобы система приняла изменения, нужно перезапустить GNOME, — достаточно просто выйти из системы и снова зайти в нее, перезагружать компьютер полностью не обязательно.

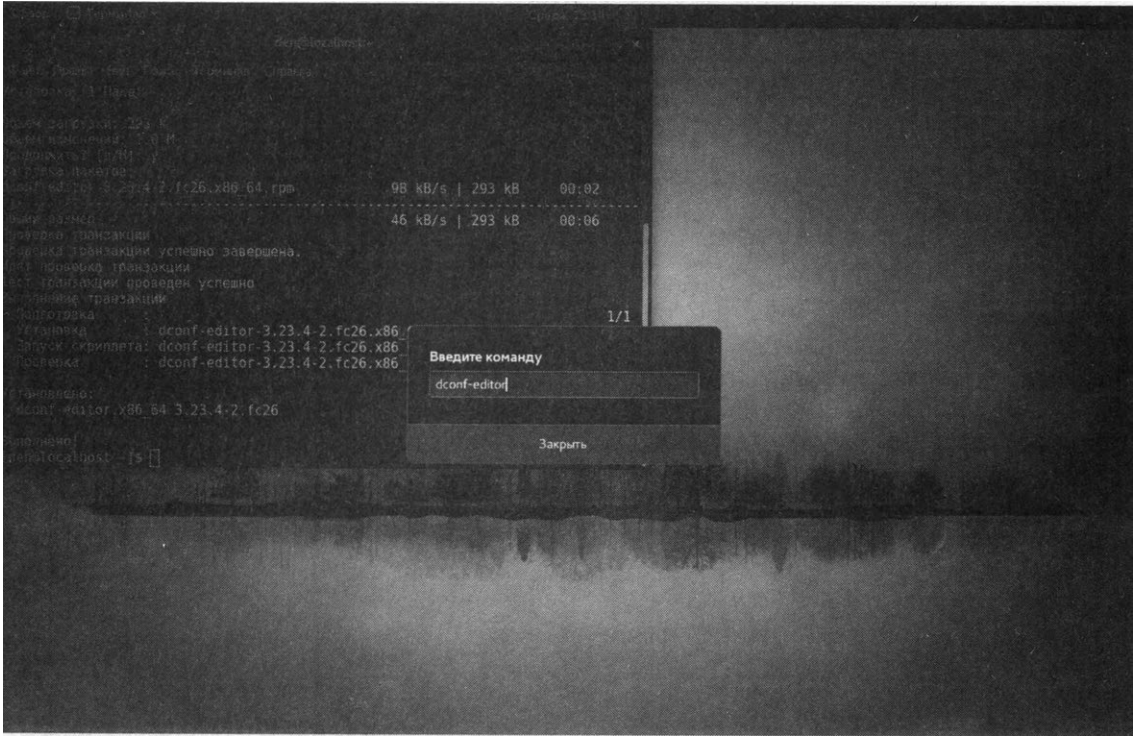


Рис. 3.19. Fedora 26: запуск dconf-editor

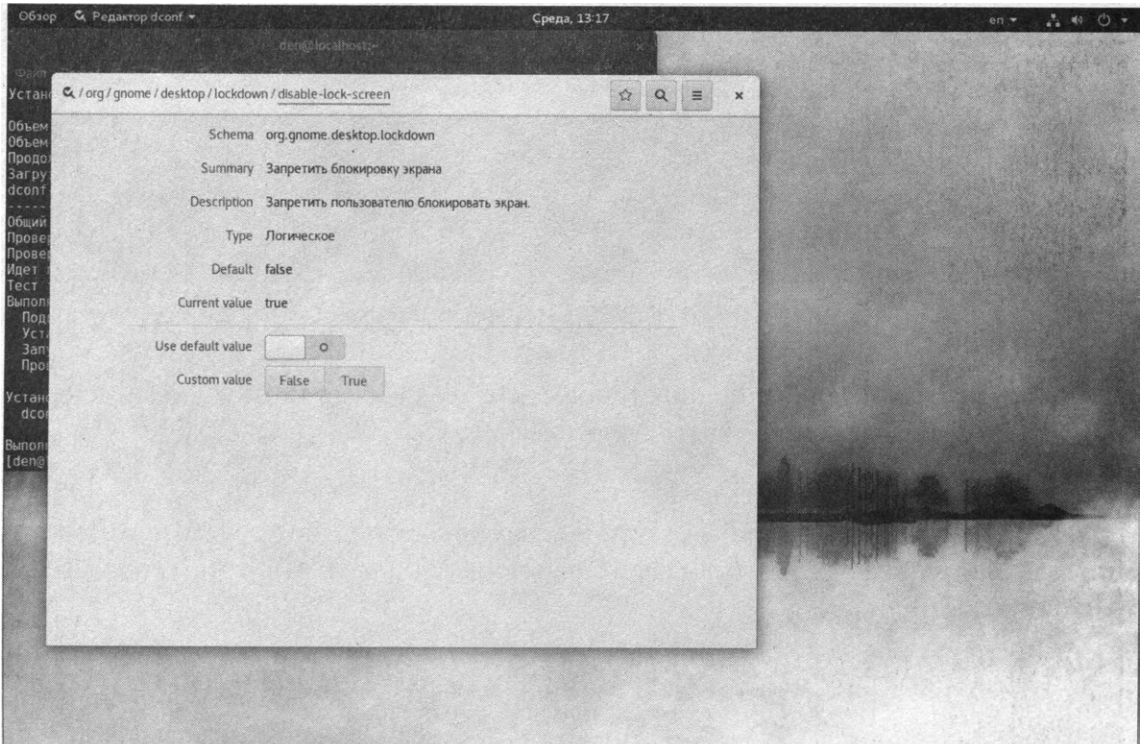



Рис. 3.20. Fedora 26: полное отключение блокировки экрана

В Ubuntu все обстоит гораздо проще, и отключить блокировку экрана можно средствами ее собственного окна настроек, без необходимости редактирования конфигурации GNOME. Для этого нажмите в правом верхнем углу главного окна Ubuntu значок , выберите команду **Параметры системы** (рис. 3.21), в открывшемся окне параметров (рис. 3.22) перейдите в раздел **Яркость и блокировка** и установите там параметры так, как показано на рис. 3.23.

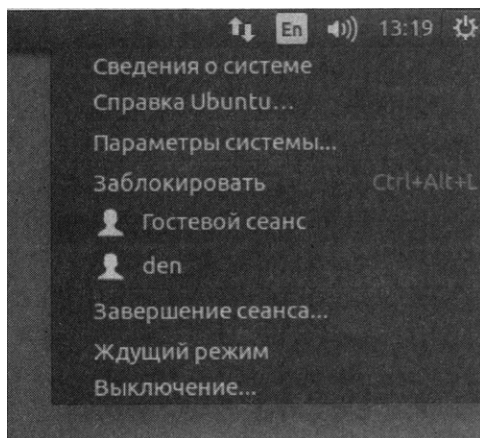


Рис. 3.21. Ubuntu 17.04: открываем окно **Параметры системы**

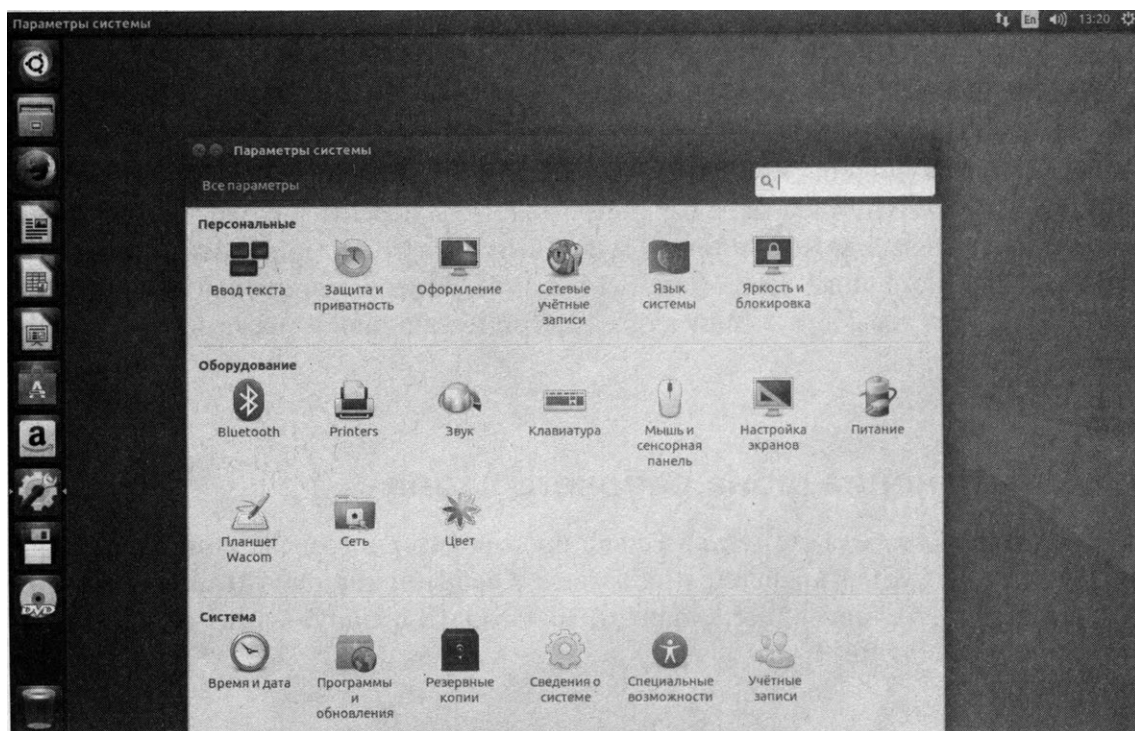


Рис. 3.22. Ubuntu 17.04: окно **Параметры системы**

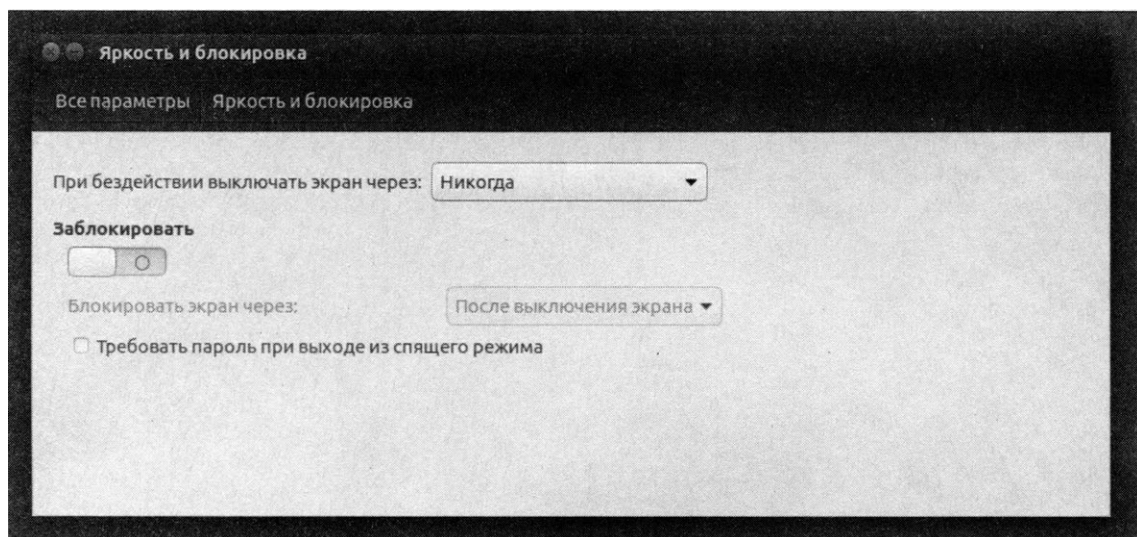


Рис. 3.23. Ubuntu 17.04: отключение блокировки экрана

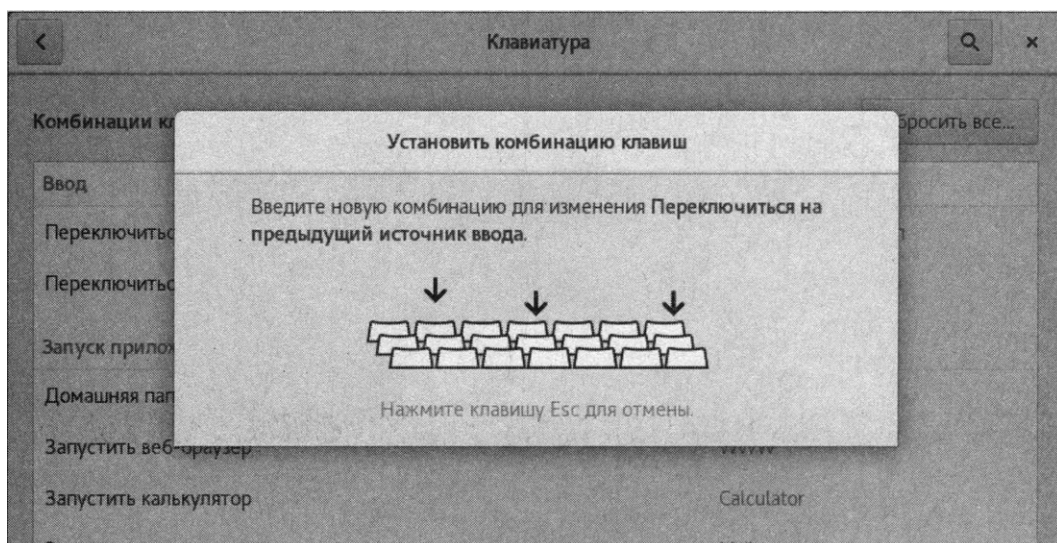
3.3.2. Изменение способа переключения языков ввода

У каждого из нас — своя любимая комбинация клавиш переключения языков ввода: у меня это `<Ctrl>+<Shift>`, но по умолчанию в Fedora предусмотрена `<Alt>+<Shift>`, а в Ubuntu — `<Super>+<Пробел>` (клавиша `<Super>` — это та, что на всех клавиатурах несет изображение логотипа Windows).

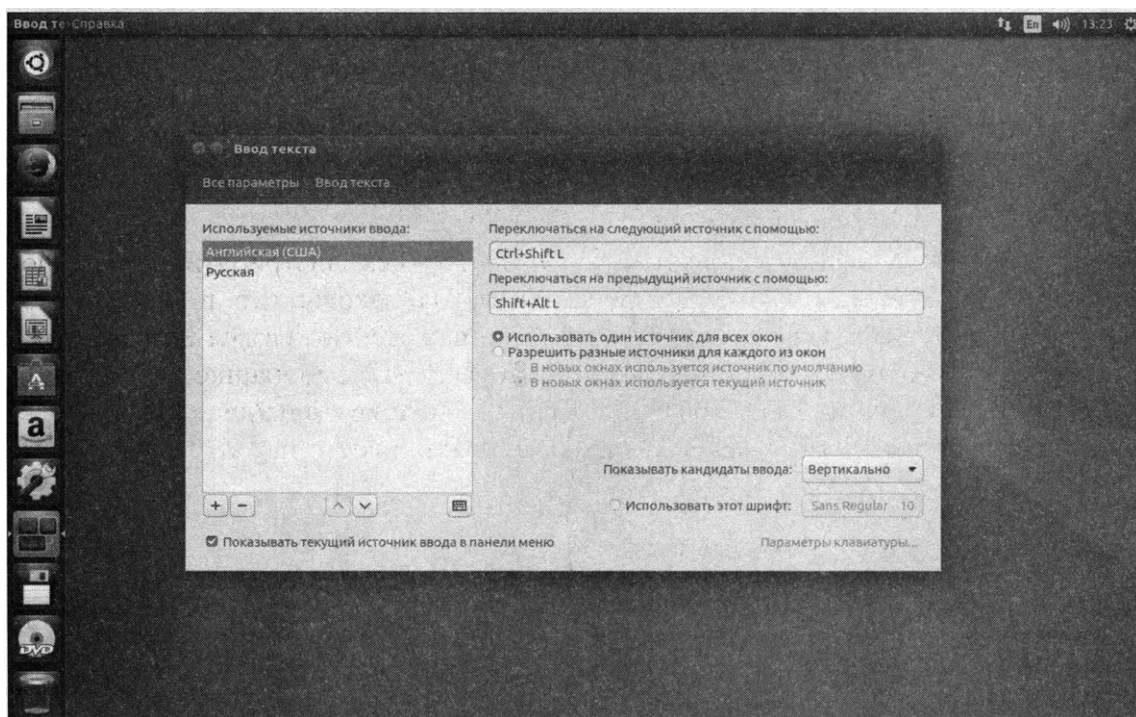
Изменение *способа переключения языков ввода* во всех дистрибутивах, где по умолчанию используется GNOME, производится одинаково: что в Fedora, что в Ubuntu. Для этого нужно в окне параметров системы перейти в раздел **Клавиатура**, затем открыть вкладку **Комбинации клавиш** и выбрать комбинацию клавиш для переключения источника ввода (рис. 3.24, *а* и *б*). Ранее комбинация клавиш устанавливалась путем выбора из списка, сейчас же нужно просто нажать нужную вам комбинацию.

3.3.3. Изменение фона рабочего стола

Изменение *фона рабочего стола* в Fedora производится в разделе **Фон** окна **Параметры** (рис. 3.25, *а*), а в Ubuntu — в разделе **Оформление** окна **Параметры системы** (рис. 3.25, *б*). Как видите, стандартный GNOME и Unity — весьма похожи, но все же не без различий.



а

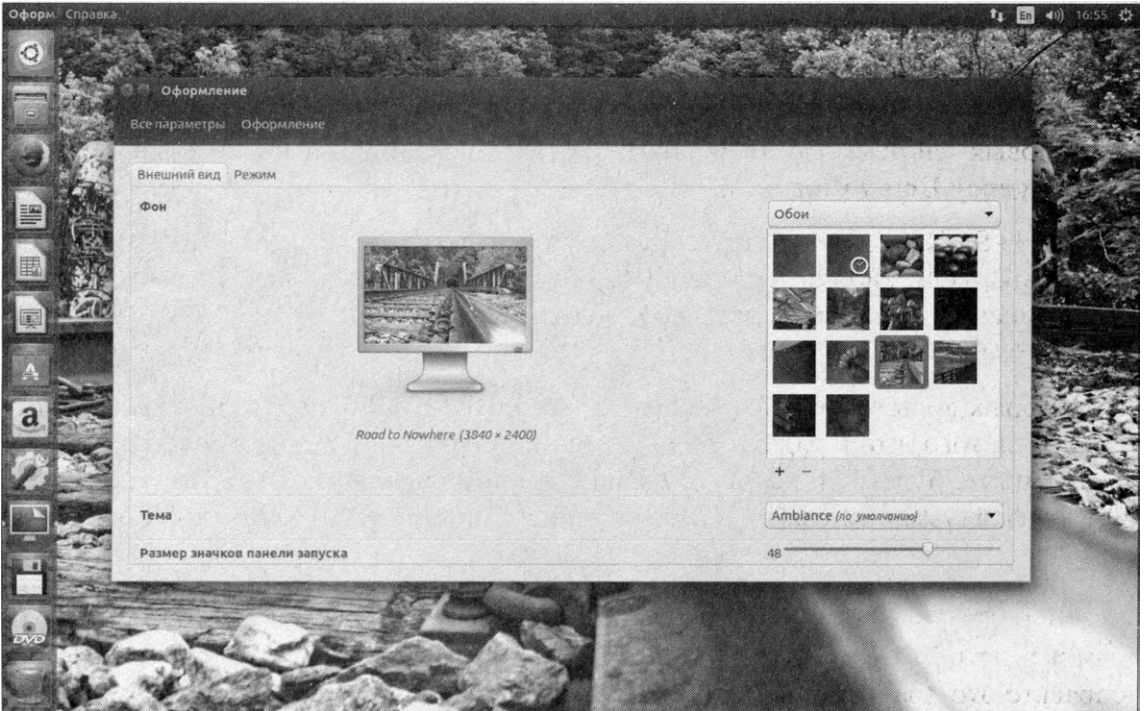


б

Рис. 3.24. Изменение способа переключения языков ввода: в Fedora 26 (а) и в Ubuntu 17.04 (б)



а



б

Рис. 3.25. Изменение фона рабочего стола: в Fedora 26 (а) и в Ubuntu 17.14 (б)

3.4. «Аварийные» комбинации клавиш, использование клавиши <SysRq>

Компьютер завис и, казалось бы, не реагирует на нажатия клавиш? Вполне может быть, но всегда можно попробовать следующие «аварийные» комбинации клавиш:

- <Ctrl>+<Alt>+ — обычно это перезагрузка системы, но реакция на нажатие этой комбинации клавиш задается в файле `/etc/inittab`;
- <Ctrl>+<Alt>+<Backspace> — аварийное завершение графической подсистемы X.Org;
- <Alt>+<SysRq>+<K> — «убивает» все запущенные процессы. Эта комбинация клавиш помогает также для приведения в чувство X.Org, когда нет реакции даже на <Ctrl>+<Alt>+<Backspace>;
- <Alt>+<SysRq>+<E> — посылает всем процессам (кроме `init`) сигнал `SIGTERM`. После этого будут запущены только ядро и `init`. Войдите снова в систему и запустите заново сервисы `/sbin/init 3` или `/sbin/init 5`;
- <Alt>+<SysRq>+<S> — сбрасывает содержимое буферов ввода/вывода на диск. Полезна, если вы боитесь, что в результате нажатия на кнопку `Reset` не будут сохранены важные данные (команда на сохранение давалась, но вы не знаете, была ли произведена физическая запись данных на носитель). Рекомендуется нажать эту комбинацию несколько раз с небольшим перерывом (в 2-3 секунды). Если вы увидели надпись **Emergency Sync**, то все нормально, можно нажимать кнопку `Reset`. А вот если нет, то остается надеяться, что просто невозможен вывод на консоль, а данные все же успели синхронизироваться;
- <Alt>+<SysRq>+<U> — используется для аварийного размонтирования всех файловых систем. По окончании размонтирования вы увидите сообщение: **Emergency Umounting... OK**;
- <Alt>+<SysRq>+ — практически эквивалентно нажатию кнопки `Reset`. Полезно, если кнопки `Reset` нет, или она не нажимается. Перед нажатием этой комбинации клавиш желательно нажать комбинации <Alt>+<SysRq>+<S> и <Alt>+<SysRq>+<U>.

У некоторых пользователей комбинации клавиш с <SysRq> просто не срабатывают. А все из-за того, что в ряде дистрибутивов по умолчанию они отключены. Прежде всего решите, будете ли вы использовать эти комбинации. Если да, тогда отредактируйте файл `/etc/sysctl.conf` — в него нужно добавить строку (если такой строки там нет):

```
kernel.sysrq = 1
```

Если же параметр `kernel.sysrq` в файле `sysctl.conf` присутствует, но его значение 0, исправьте это значение на 1, после чего или перезагрузите систему, или введите команду:

```
sudo sysctl -w «kernel.sysrq=1»
```


Есть еще одна причина, по которой комбинация клавиш с <SysRq> может не работать. Как правило, на ноутбуках в целях экономии места на клавиатуре значения некоторых клавиш перенагружают функциями. Так, например, на моем ноутбуке HP функции клавиш <Delete> и <SysRq> повешены на одну и ту же физическую кнопку на клавиатуре. Обычно она работает как <Delete>, но будучи нажата вместе с клавишей <Fn>, работает как <SysRq>. В таком случае все комбинации клавиш с <SysRq> следует дополнить нажатием клавиши <Fn>. При этом комбинация <Alt>+<SysRq>+<S> будет правильно работать уже в варианте <Alt>+<Fn>+<SysRq>+<S>. Да, понимаю, что это неудобно, но другого выхода нет (если, конечно, не подключить к ноутбуку внешнюю полноразмерную клавиатуру).

3.5. Практические приемы работы с консолью

Работая с этой книгой, вам часто придется вводить различные команды, т. е. взаимодействие с консолью системы будет у вас довольно плотным. Некоторые команды (в основном, для работы с файловой системой) мы рассмотрим в *главе 4*, с рядом полезных команд вы познакомитесь в *главе 25*. Сейчас же поговорим о *практических приемах работы в командной строке*.

3.5.1. Автодополнение командной строки и псевдонимы команд

Работа в консоли заключается во вводе нужной команды — вы вводите команду (например, создания каталога, просмотра файла, вызова редактора и т. д.) и нажимаете клавишу <Enter>. Команда содержит как минимум имя запускаемой программы. Кроме имени программы команда может содержать параметры, которые будут переданы программе, а также символы перенаправления ввода/вывода (об этом чуть позже). Естественно, вам нужно знать имя программы, а также параметры, которые необходимо ей передать.

Если вы помните название программы, а назначение параметров забыли, поможет команда **man**. Man (от англ. manual) — это справочная система Linux. В ней имеется информация о каждой программе, которая установлена в системе. Откуда система знает обо всех программах? Все очень просто — разработчики программ под Linux договорились, что вместе с программой будет поставляться специальный ман-файл, — файл справочной системы. Понятно, если разработчик недобросовестный, он может и не создать файл справочной системы, но это происходит очень редко. И чтобы получить справку по какой-нибудь программе, нужно ввести команду:

man *имя_программы*

Вы никак не можете запомнить, как пишется та или иная команда? Если вы помните хотя бы на какую букву она начинается, воспользуйтесь функцией *автодополнения командной строки* — введите первые буквы команды и нажмите клавишу <Tab>. При первом нажатии система попытается дополнить команду. Иногда дополнить команду невозможно — например, вы ввели букву *a*. Ясное дело, в системе

есть несколько команд, которые начинаются на букву «а», и в такой ситуации система не может дополнить командную строку. Но если вы хотите просмотреть все команды на букву «а», тогда нажмите еще раз клавишу <Tab>.

Вам не с руки вписывать (даже с автодополнением) длинные команды? Тогда можно создать *псевдонимы команд*. Для этого в файл `.bash_profile` добавьте строки вида:

```
alias псевдоним= 'команда'
```

Например:

```
alias cfg-net='system-config-network'
```

Для того чтобы изменения вступили в силу, выйдите из консоли (командой `logout`) и заново зарегистрируйтесь.

Пожалуй, для полноценной работы с консолью вам нужно знать еще одну команду — `clear`. Эта команда очищает консоль (терминал). Очень полезная команда, особенно когда вы хотите все начать с «чистого листа».

3.5.2. Графические терминалы

Понимаю, что практически все дистрибутивы оснащены графическим интерфейсом, который к тому же запускается по умолчанию. Поэтому большинство пользователей не станут жертвовать удобным и привычным интерфейсом ради консоли.

Как уже упоминалось в *разд. 3.1*, вместо переключения в консоль можно использовать *терминал*— эмулятор консоли. Терминал— это графическая программа (см. рис. 3.18), в окне которой вы можете вводить команды и видеть результаты их выполнения.

3.5.3. Перенаправление ввода/вывода

С помощью *перенаправления ввода/вывода* мы можем перенаправить вывод одной программы в файл или на стандартный ввод другой программы. Например, у вас не получается настроить сеть, и вы хотите перенаправить вывод команды `ifconfig` в файл, а затем разместить этот файл на форуме, где вам помогут разобраться с проблемой. А можно командой `ps -ax` перенаправить список всех процессов команде `grep`, которая найдет в списке интересующий вас процесс.

Рассмотрим следующую команду:

```
echo "some text" > file.txt
```

Символ `>` означает, что вывод команды, находящейся слева от этого символа, будет записан в файл, находящийся справа от символа, при этом файл будет перезаписан.

Чуть ранее мы говорили о перенаправлении вывода программы `ifconfig` в файл. Команда будет выглядеть так:

```
ifconfig > ifconfig.txt
```

Если вместо `>` указано `>>`, то исходный файл не будет перезаписан, а вывод команды добавится в конец файла:

```
echo "some text" > file.txt
echo "more text" >> file.txt
cat file.txt
some text
more text
```

Кроме символов `>` и `>>` для перенаправления ввода/вывода часто употребляется вертикальная черта `|`. Предположим, что мы хотим вывести содержимое файла `big_text`:

```
cat big_text
```

Но в файле `big_text` много строк, они быстро проскочат по экрану, и мы ничего не успеем прочитать. Следовательно, целесообразно отправить вывод команды `cat` какой-то программе, которая будет выводить файл на экран постранично, например:

```
cat big_text | more
```

Конечно, этот пример не очень убедительный, потому что для постраничного вывода гораздо удобнее команда `less`:

```
less big_text
```

Вот еще один интересный пример. Допустим, мы хотим удалить файл `file.txt` без запроса — для этого можно указать команду:

```
echo y | rm file.txt
```

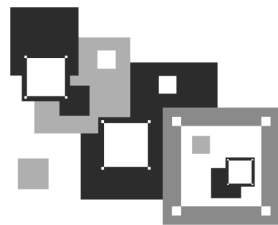
Команда `rm` запросит подтверждение удаления (нужно было бы нажать клавишу `<Y>`), но за нас это сделает команда `echo`.

И еще один пример. Пусть имеется большой файл, и нам нужно найти в нем все строки, содержащие подстроку `555-555`. Чтобы не делать это вручную, можно воспользоваться командой:

```
cat file.txt | grep "555-555"
```

Надеюсь, приведенная в этом разделе информация сделает вашу работу в командной строке максимально комфортной.

ГЛАВА 4



Файловая система Linux

4.1. Файловые системы, поддерживаемые Linux

Linux поддерживает много различных файловых систем. Начинающий пользователь просто теряется, когда видит такое многообразие выбора, — ведь в качестве корневой файловой системы доступны: ext2, ext3, ext4, XFS, ReiserFS, JFS и еще несколько.

ФАЙЛОВАЯ СИСТЕМА EXT

Linux также до сих пор поддерживает файловую систему ext (самую первую файловую систему Linux), однако выбрать ext при установке системы вы не сможете, — поддержка ext добавлена в ядро лишь на тот случай, если вам попадется носитель информации, отформатированный в этой файловой системе.

«Родной» файловой системой современных дистрибутивов Linux является журналируемая файловая система ext4, но при установке Linux вы можете выбрать и предыдущие версии: ext3 или даже ext2.

ЖУРНАЛИРУЕМОСТЬ

Все упомянутые здесь файловые системы (кроме ext2 и, естественно, ext) ведут журналы своей работы, что позволяет восстановить данные в случае сбоя. Осуществляется это следующим образом: перед тем как выполнить операцию, *журналируемая* файловая система записывает ее в особый файл — *журнал*, а после выполнения операции удаляет запись из журнала. Представим, что после занесения операции в журнал произошел сбой (например, отключилось электропитание). Позже, когда сбой будет устранен, файловая система по журналу выполнит все действия, которые в него занесены. Конечно, и это не всегда позволяет уберечься от последствий сбоя — стопроцентной гарантии никто не дает, но все же такая схема работы лучше, чем вообще ничего.

Основное отличие ext3 от ext2 как раз и заключается в ее журналируемости. При этом файловые системы ext2 и ext3 совместимы, т. е. раздел **ext3** могут читать программы, рассчитанные на ext2 (например, Total Commander и Ext2Fsd из-под Windows). Современная версия — ext4 — построена на базе ext3, но отличия столь существенны, что о них мы поговорим отдельно (см. *разд. 4.9*).

Итак, в качестве корневой файловой системы и файловой системы других Linux-разделов могут служить файловые системы ext3 и ext4, а также ReiserFS, XFS, JFS и др. Рассмотрим особенности этих файловых систем, чтобы понять, использовать ли их или же остановить свой выбор на стандартной ext4.

- **Файловая система ReiserFS** (она же **Reiser3**) считается самой экономной, поскольку позволяет хранить несколько файлов в одном блоке (другие файловые системы могут хранить в одном блоке только один файл или одну его часть). Например, если размер блока равен 4 Кбайт, а файл занимает всего 512 байтов (а таких файлов в разных каталогах Linux очень много), то 3,5 Кбайт в этом блоке просто не будут использоваться. А вот ReiserFS позволяет задействовать буквально каждый байт вашего жесткого диска!

Но у этой файловой системы есть два больших недостатка: она неустойчива к сбоям, и ее производительность сильно снижается при фрагментации диска. Поэтому, если вы выбираете ReiserFS, покупайте источник бесперебойного питания и почаще дефрагментируйте жесткий диск.

ИНТЕРНЕТ-МАГАЗИН НА БАЗЕ MAGENTO

Впрочем, именно благодаря ReiserFS, которая превосходно работает с мелкими файлами, удалось обеспечить стабильную работу интернет-магазина на базе платформы Magento. Magento создает множество (особенно при хорошей посещаемости) мелких файлов в каталоге `var/session`. Файлов создается настолько много, что сервер (используя VDS с 5-ю ядрами и 16 Гбайт ОЗУ) начинал работать некорректно: в панель управления Magento войти было нельзя, иногда зависал даже процесс Apache. Сначала проблема решалась периодической очисткой каталога `var/session`. Однако этот каталог приходилось очищать все чаще и чаще — по мере роста посещаемости. Но очистка каталога `var/session` означает, что информация обо всех сессиях будет удалена. Это создавало неудобства как для менеджеров магазина, которых «выбрасывало» из панели управления Magento, так и для самих посетителей — представьте, выбирая покупки, вы добавили в корзину несколько десятков наименований, а вдруг корзина взяла и очистилась! Проблему удалось решить благодаря переносу каталога `var/session` на файловую систему ReiserFS. Сейчас в этом каталоге несколько миллионов файлов, и сервер работает стабильно.

- **Файловая система Reiser4** впервые была представлена в 2004 году. Она поддерживает транзакции, задержку выделения пространства, а также сжатие и шифрование данных. Однако создатель этой файловой системы Ханс Рейзер (Hans Reiser) был осужден в 2008 году за убийство жены, и Reiser4 стала развиваться не столь активно, как хотелось бы. Тем не менее, эта файловая система поддерживается группой энтузиастов во главе с Эдуардом Шишкиным. Впрочем, несмотря на все их старания, в основную ветку ядра файловую систему Reiser4 так и не включили.
- **Файловая система XFS** была разработана компанией Silicon Graphics в 2001 году. Основная ее особенность — высокая производительность (до 7 Гбайт/с). Кроме того, XFS может работать с блоками размером от 512 байтов до 64 Кбайт. Ясно, что если у вас много небольших файлов, то в целях экономии дискового пространства можно установить самый маленький размер блока. А если вы работаете с файлами большого размера (например, с мультимедиа), выбирайте самые большие блоки, — тогда файловая система обеспечит максимальную

производительность (конечно, если «железо» позволяет). Учитывая такие особенности этой файловой системы, ее нет смысла устанавливать на домашнем компьютере, предназначенном для выхода в Интернет и просмотра любительских фотографий, поскольку вы просто не сможете оценить все ее преимущества. А вот если вы реально работаете с файлами очень большого размера, XFS проявит себя с лучшей стороны. Стоит отметить, что эта файловая система используется в Fedora 26 Server, тогда как в Fedora 26 Workstation по умолчанию устанавливается ext4. Другими словами, разработчики Fedora рекомендуют XFS для серверов.

- **Файловая система ZFS (Zettabyte File System)** создана в 2005 году компанией Sun Microsystems для операционной системы Solaris. Отличительные особенности ZFS: отсутствие фрагментации, создание снапшотов диска, которые можно использовать для восстановления данных, организация пулов хранения (storage pools), изменяемый размер блоков, 64-разрядный механизм контрольных сумм.
- **Файловая система Btrfs (B-tree FS или Butter FS)** изначально была представлена компанией Oracle. Многие считают эту файловую систему ответом Oracle на файловую систему ZFS. Файловая система Btrfs настолько хороша, что разработчики openSUSE выбрали ее в качестве основной в openSUSE 13.2 — вместо проверенной годами ext4. Ключевые особенности Btrfs: сжатие данных, оптимизированный для SSD режим работы, контроль за целостностью данных и метаданных, поддержка снапшотов диска и т. д.
- **Файловая система JFS** (разработка IBM) сначала появилась в операционной системе AIX, а потом была модифицирована под Linux. Основные достоинства этой файловой системы — надежность и высокая производительность (выше, чем у XFS). Однако у нее маленький размер блока (от 512 байтов до 4 Кбайт) — следовательно, она хороша на сервере баз данных, но не при работе с данными мультимедиа, поскольку блока в 4 Кбайт для обработки, например, видео в реальном времени будет маловато.
- **Файловая система Tux2** была создана Дэниэлем Филипсом как надстройка над ext2, но не получила публичного распространения.
- **Файловая система Tux3** задумывалась как надстройка над Btrfs. Эта файловая система вместо журналирования предлагает версионное восстановление файлов: для каждого файла создается измененная копия, а не переписывается текущая версия, что позволяет гибко управлять версиями.
- **Файловая система Xiafs** основана на файловой системе MINIX. Это весьма древняя разработка, она создавалась параллельно с ext2 на замену файловой системе ext, и не получила распространения.

4.1.1. Выбор файловой системы

С точки зрения *производительности* рассматриваемых файловых систем напрашиваются следующие рекомендации:

- для рабочей станции и сервера общего назначения оптимальной файловой системой являются ext3/ext4 или ReiserFS (в крайнем случае);

- на сервере баз данных можно использовать JFS — в этом случае (особенно, если база данных огромная) будет наблюдаться определенный прирост производительности;
- файловая система XFS — это удел станции мультимедиа, на обычной рабочей станции или обычном сервере ее использовать не следует.

Но производительность — это не единственный критерий выбора файловой системы, особенно для сервера. Да, производительность учитывать нужно, но, кроме того, нельзя пренебрегать и следующими факторами:

- *надежностью* — все-таки мы выбираем файловую систему для сервера, а не для домашнего компьютера;
- *наличием программ для восстановления файловой системы в случае сбоя* — сбой может произойти даже в случае использования самой надежной файловой системы, поэтому наличие программного комплекса для восстановления файловой системы не будет лишним;
- *максимальным размером файла* — сервер обрабатывает огромные объемы информации, поэтому этот критерий для нас также важен.

Файловые системы ext3/ext4, ReiserFS и XFS одинаково надежны, а вот надежность JFS иногда оставляет желать лучшего. Учитывая это, а также и то, что программы для восстановления файловой системы имеются только в системах ext*, на сервере лучше использовать все-таки ext3/ext4.

Если вы уже интересовались характеристиками файловых систем, то могли в некоторых источниках встретить неверную информацию о максимальном размере файла для файловой системы ext3. Так, иногда сообщается, что максимальный размер файла для ext3 равен 2 Гбайт, что делает ее непригодной для использования на сервере. Это не так. Раньше, во времена ext2 и ядер 2.2 и 2.4, действительно, существовало такое ограничение, но только для ext2. Файловая система ext3 поддерживает файлы размером до 1 Тбайт, а максимальный размер тома (раздела) у нее равен 4 Тбайт, что вполне достаточно даже для сервера. Если же вам нужна поддержка больших объемов данных, рекомендую обратить внимание на другие файловые системы, — например, на ReiserFS (максимальный размер файла 16 Тбайт) или на XFS/JFS (размер файла вообще исчисляется в петабайтах).

4.1.2. Linux и файловые системы Windows

Linux почти безо всяких ограничений поддерживает файловые системы FAT12 (DOS), FAT16 (или просто FAT, как в Windows 95) и FAT32 (Windows 98 и все последующие версии — до появления в них файловой системы NTFS). Вы можете из Linux читать в файловых системах Windows файлы и каталоги, изменять, создавать новые файлы и каталоги, удалять их — в общем все, что можно делать в файловой системе непосредственно в Windows.

Однако файловые системы Windows не поддерживают установку прав доступа, поэтому можно даже не пытаться установить в Linux права доступа к файлу, который находится на Windows-разделе, — у вас ничего не получится.

О файловой системе NTFS — отдельный разговор. По умолчанию (без перекомпиляции ядра) Linux умеет только читать данные, расположенные в NTFS-разделе. Однако даже после перекомпиляции ядра ряд ограничений на запись в NTFS-раздел останется — например, вы не можете создавать в нем новые файлы, разрешено только редактировать уже имеющиеся. Кстати, поддержка NTFS современным ядром до сих пор экспериментальна, т. е. в один не совсем прекрасный момент при попытке записи из-под Linux в раздел NTFS вы можете потерять в нем все свои данные.

Я вас напугал? Существуют решения (и мы рассмотрим их в этой книге далее), позволяющие снять большую часть ограничений на запись в NTFS-разделы. Конечно, все эти решения не идеальные: что-то работает, но ужасно медленно, что-то снимает далеко не все ограничения на запись, но, тем не менее, возможность записывать данные в NTFS-раздел без их потери все же имеется.

4.1.3. Сменные носители

Linux превосходно работает со сменными CD/DVD- и USB-дисками и в большинстве случаев даже выполняет их автоматическое монтирование и размонтирование (хотя эта функция доступна не во всех дистрибутивах). С другой стороны, автоматическое монтирование сменных носителей на сервере — это от лукавого, на домашнем компьютере — да, но не на сервере. О монтировании, в том числе автоматическом, мы поговорим чуть позже в этой главе.

4.2. Особенности файловых систем Linux

4.2.1. Имена файлов в Linux

В Linux, по сравнению с Windows, несколько иные правила построения имен файлов, и вам придется с этим смириться. Начнем с того, что в Linux нет такого понятия, как *расширение имени файла*. В Windows, например, для файла `Document1.doc` именем файла является фрагмент `Document1`, а `doc` — это его расширение. В Linux же `Document1.doc` — это имя файла целиком, никакого деления на имя и расширение нет.

Максимальная длина имени файла — 254 символа. Имя может содержать любые символы (в том числе и кириллицу), кроме `/ \ ? < > * " |`. Тем не менее кириллицу в именах файлов я бы не рекомендовал использовать вовсе. Впрочем, если вы уверены, что не будете эти файлы передавать Windows-пользователям (на флешке, по электронной почте или еще как-то через Интернет) — используйте на здоровье. А при обмене файлами с Windows-пользователями из-за возможных несовпадений кодировок вместо русскоязычного имени файла адресат может увидеть абракадабру... Так что, имена файлов во всех случаях лучше писать латиницей.

Придется вам привыкнуть к тому, что Linux чувствительна к регистру в имени файла: `FILE.txt` и `File.Txt` — это два разных файла.

Разделение элементов пути осуществляется символом `/` (прямой слэш), а не `\` (обратный слэш), как в Windows.

4.2.2. Файлы и устройства

Пользователи Windows привыкли к тому, что файл — это именованная область данных на диске. Отчасти так оно и есть. Отчасти — потому, что приведенное определение файла было верно для DOS (Disk Operating System) и Windows.

В Linux же понятие файла значительно шире. Сейчас Windows-пользователи будут очень удивлены: в Linux есть файлы устройств, позволяющие обращаться с устройством как с обычным файлом. Файлы устройств находятся в каталоге `/dev` (от *devices*). Да, через файл устройства мы можем обратиться к устройству! Если вы работали в DOS, то, наверное, помните, что что-то подобное было и там — существовали зарезервированные имена файлов: PRN (принтер), CON (клавиатура при вводе, дисплей при выводе), LPT*n* (параллельный порт, *n* — номер порта), COM*n* (последовательный порт).

ФАЙЛЫ УСТРОЙСТВ

Кому-то может показаться, что разработчики Linux «украли» идею специальных файлов у Microsoft — ведь Linux появилась в начале 90-х, а DOS — в начале 80-х годов прошлого века. На самом деле это не так. Наоборот, Microsoft позаимствовала идею файлов устройств из операционной системы UNIX, которая была создана еще до появления DOS. Однако сейчас не время говорить об истории развития операционных систем, поэтому лучше вернемся к файлам устройств.

Вот некоторые примеры файлов устройств:

- `/dev/sdx` — файл жесткого диска;
- `/dev/sdxN` — файл устройства раздела на жестком диске, *N* — это номер раздела;
- `/dev/scdN` — файл устройства CD/DVD-привода;
- `/dev/mouse` — файл устройства мыши;
- `/dev/modem` — файл устройства модема (на самом деле является ссылкой на файл устройства `ttySn`);
- `/dev/ttySn` — файл последовательного порта, *n* — номер порта (`ttySO` соответствует COM1, `ttySI` — COM2 и т. д.).

В свою очередь, файлы устройств бывают двух типов: блочные и символьные. Обмен информации с *блочными* устройствами, например с жестким диском, осуществляется блоками информации, а с *символьными* — отдельными символами. Пример символьного устройства — последовательный порт.

4.2.3. Корневая файловая система и монтирование

Наверняка на вашем компьютере установлена система Windows. Запустите Проводник и просмотрите список логических дисков вашего компьютера (рис. 4.1).

Скорее всего, вы увидите значок гибкого диска (имя устройства **A:**), значки разделов жесткого диска (в нашем случае имеется один раздел — **C:**), значок привода CD/DVD (**D:**). Таким способом, с помощью буквенных обозначений **A:**, **C:**, **D:** и т. п., в Windows обозначаются корневые каталоги разделов жесткого диска и сменных носителей.

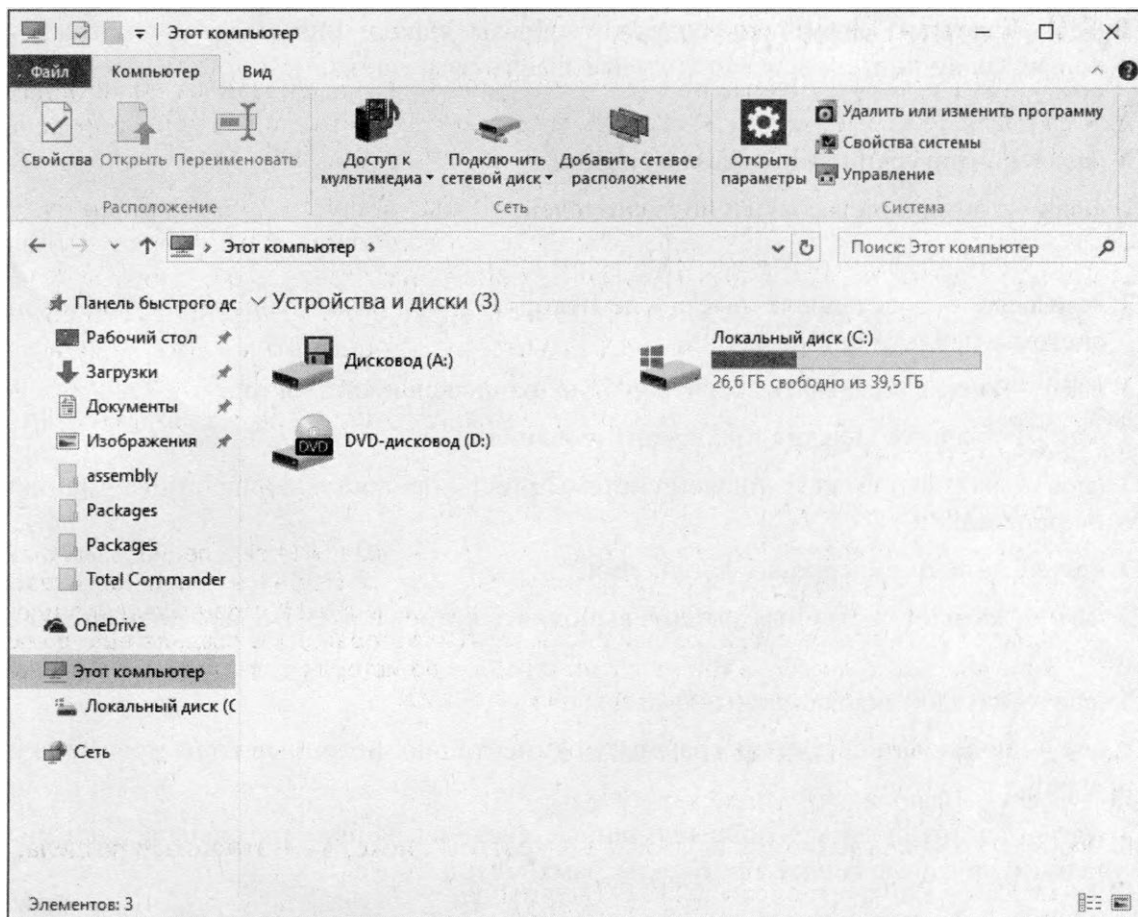


Рис. 4.1. Проводник в Windows 10

В Linux существует понятие *корневой файловой системы*. Допустим, вы установили Linux в раздел с именем `/dev/sda3` — в этом разделе и будет развернута корневая файловая система вашей Linux-системы. Корневой каталог обозначается прямым слэшем — `/`, т. е. для перехода в корневой каталог в терминале (или консоли) нужно ввести команду `cd /`.

Понятно, что на вашем жестком диске есть еще и другие разделы. Чтобы получить к ним доступ, вам нужно *подмонтировать* их к корневой файловой системе. После монтирования вы сможете обратиться к содержимому разделов через точку монтирования — назначенный вами при монтировании специальный каталог, например, `/mnt/cdrom`. Монтированию файловых систем посвящен *разд. 4.7*, поэтому сейчас мы не станем говорить об этом процессе подробно.

4.2.4. Стандартные каталоги Linux

Файловая система любого дистрибутива Linux содержит следующие каталоги:

- `/` — корневой каталог;
- `/bin` — стандартные программы Linux (`cat`, `cp`, `ls`, `login` и т. д.);

- **/boot** — каталог загрузчика, содержит образы ядра и Initrd, может содержать конфигурационные и вспомогательные файлы загрузчика;
- **/dev** — файлы устройств;
- **/etc** — конфигурационные файлы системы;
- **/home** — домашние каталоги пользователей;
- **/lib** — библиотеки и модули;
- **/lost+found** — восстановленные после некорректного размонтирования файловой системы файлы и каталоги;
- **/misc** — может содержать все, что угодно, равно как и каталог **/opt**;
- **/mnt** — обычно содержит точки монтирования;
- **/proc** — каталог псевдофайловой системы **procfs**, предоставляющей информацию о процессах;
- **/root** — каталог суперпользователя **root**;
- **/sbin** — каталог системных утилит, выполнять которые имеет право пользователь **root**;
- **/tmp** — каталог для временных файлов;
- **/usr** — пользовательские программы, документацию, исходные коды программ и ядра;
- **/var** — постоянно изменяющиеся данные системы, например, очереди системы печати, почтовые ящики, протоколы, замки и т. д.

4.3. Внутреннее строение файловой системы

Что такое файловая система? Можно встретить различные определения, и все они будут правильные. Наиболее точным я считаю следующее:

Файловая система — это способ представления информации на носителе данных, а также часть операционной системы, обеспечивающая выполнение операции над файлами.

Из приведенного определения ясно, что файловая система состоит из двух частей, двух уровней: уровня представления данных и набора системных вызовов для работы с этими данными.

Любая операционная система может работать с разными файловыми системами — например, со своей основной файловой системой и с файловой системой компакт-дисков (ISO 9660). Задача операционной системы заключается в предоставлении пользователю стандартного интерфейса, позволяющего ему обращаться к каждой файловой системе, не вникая в ее особенности. Например, в Linux для открытия файла служит системный вызов `open ()` — программа просто вызывает `open ()`, передав ему имя файла, а на какой файловой системе расположен этот файл — дело третье.

Рассмотрим схему архитектуры файловой системы (рис. 4.2): верхние два элемента — это пользовательский уровень, все последующие — уровень ядра.

Приложение может использовать функции `glibc` (библиотека GNU C) или же напрямую системные вызовы ядра — тут уж как будет угодно программисту. Использовать функции `glibc` удобнее, но, вызывая непосредственно системные вызовы, — например, `open()`, `read()`, `write()`, `close()`, — можно немного повысить производительность приложения — ведь вы минуete `glibc`, которая все равно использует те же системные вызовы.



Рис. 4.2. Архитектура файловой системы

VFS — это *виртуальная файловая система*. Именно она позволяет добиться существующего сейчас уровня абстракции. Каждая файловая система имеет свои особенности. Если бы не было VFS, то пришлось бы разрабатывать разные версии системных вызовов для каждого типа поддерживаемой файловой системы, например `open_ext2()` для открытия файла, находящегося на файловой системе `ext2`, или `open_vfat()` — для VFAT. Другими словами, VFS делает системные вызовы независимыми от типа используемой файловой системы.

Драйверы устройств служат для физического доступа к носителям данных. Ведь эти самые носители тоже различны — в компьютере может быть установлено несколько жестких дисков с разными интерфейсами — например, диски PATA и SATA.

Схематически раздел диска с файловой системой `ext4` можно представить так, как показано на рис. 4.3.

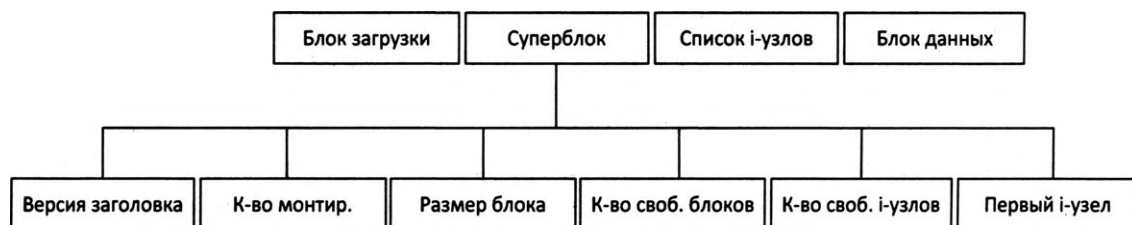


Рис. 4.3. Структура файловой системы `ext4`

Жесткий диск физически разбивается на секторы по 512 байтов каждый. Первый сектор каждого раздела представляет собой *загрузочную область*. В загрузочной области первичного раздела находится *главная загрузочная запись* (Master Boot Record, MBR) — программа, которая запускает операционную систему. На других разделах такой записи нет.

Все последующие (после загрузочной записи) секторы объединены в логические блоки. *Блок* — это наименьшая адресуемая порция данных. Размер блока может быть 1, 2 или 4 Кбайт. Блоки группируются в группы блоков. Нумерация групп начинается с 1.

После загрузочного сектора (блока загрузки) следует *суперблок*, хранящий всю информацию о файловой системе. Размер суперблока — 1 Кбайт (1024 байта). Суперблок дублируется в каждой группе блоков, что позволяет восстановить его в случае повреждения файловой системы.

В суперблоке хранится следующая информация: версия заголовка, количество монтирований (именно по этому «счетчику» система понимает, что пора проверить ту или иную файловую систему), размер блока, количество свободных блоков, количество свободных *i*-узлов, а также ссылка на первый *i*-узел (*i*-узел каталога /).

Каждому файлу соответствует только один *i*-узел, хранящий метаданные файла, — все атрибуты файла, кроме его имени, а именно: режим, информацию о владельце и группе, время доступа, время модификации, время создания, а также список ссылок на файл.

Кроме атрибутов файла в *i*-узле хранится указатель на данные файла. Обычно это массив из 15 адресов блоков, 12 из которых непосредственно ссылаются на номера блоков, хранящие данные файла. Если данные занимают более 12 блоков (напомним, что обычно 1 блок = 1 Кбайт), то используется косвенная адресация. Поэтому следующий адрес (13-й) — это адрес блока, где находится список адресов других блоков, содержащих данные файла.

Не нужно быть гением в математике, чтобы вычислить, сколько блоков можно разместить путем косвенной адресации. Все зависит от размера блока, который может быть 1, 2 или 4 Кбайт. Следовательно, можно адресовать 256, 512 или 1024 блока. А что делать, если файл еще больше? Тогда используется двойная и тройная косвенная адресация: 14-й адрес — это адрес блока, содержащего список последующих адресов блоков данных этого файла, 15-й адрес используется тройной косвенной адресацией и содержит список адресов блоков, которые являются блоками двойной косвенной адресации.

Ранее было сказано, что в *i*-узле хранится вся информация о файле, кроме его имени. Имя файла хранится в каталоге, к которому принадлежит файл. А отсюда следует, что одному *i*-узлу может соответствовать неограниченное количество имен файла (ссылок). При этом ссылки (дополнительные имена) могут находиться как в одном каталоге с исходным файлом, так и в любом другом каталоге файловой системы.

Как мы уже знаем, в Linux есть обычные файлы и есть файлы устройств. В чем между ними разница? Эта разница проявляется на уровне i-узла: i-узел обычного файла указывает на блоки данных, а i-узел файла устройства — на адрес драйвера в ядре Linux.

4.4. Команды для работы с файлами и каталогами

4.4.1. Работа с файлами

Здесь мы рассмотрим основные команды для работы с файлами в Linux (табл. 4.1), а в последующих разделах этой главы — команды для работы с каталогами, ссылками и поговорим о правах доступа к файлам и каталогам.

Таблица 4.1. Основные команды Linux, предназначенные для работы с файлами

Команда	Назначение
<code>touch <файл></code>	Создает пустой файл
<code>cat <файл></code>	Просмотр текстового файла
<code>tac <файла></code>	Вывод содержимого текстового файла в обратном порядке, т. е. сначала выводится последняя строка, потом предпоследняя и т. д.
<code>cp <файл1> <файл2></code>	Копирует файл <файл1> в файл <файл2>. Если <файл2> существует, программа попросит разрешение на его перезапись
<code>mv <файл1> <файл2></code>	Перемещает файл <файл1> в файл <файл2>. Эту же команду можно использовать и для переименования файла
<code>rm <файл></code>	Удаляет файл
<code>locate <файл></code>	Производит быстрый поиск файла
<code>which <программа></code>	Выводит каталог, в котором находится программа, если она вообще установлена. Поиск производится в каталогах, указанных в переменной окружения <code>PATH</code> (это путь поиска программ)
<code>less <файл></code>	Используется для удобного просмотра файла с возможностью скроллинга (постраничной прокрутки)

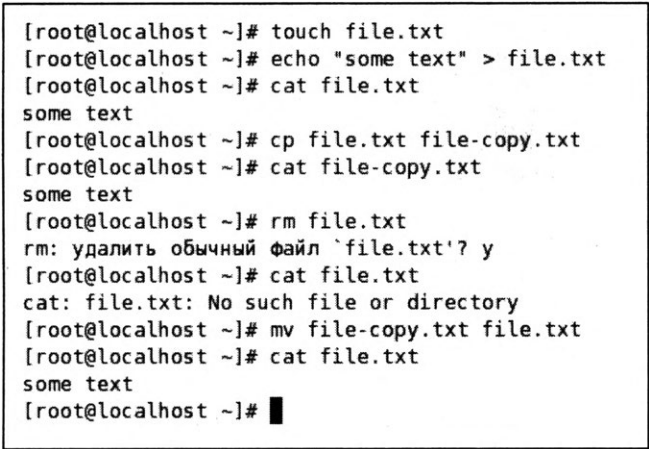
ЕЩЕ РАЗ О КОНСОЛИ...

Все представленные здесь команды предназначены для работы в консоли, т. е. в текстовом режиме. Понятно, что большинство современных дистрибутивов запускаются в графическом режиме, поэтому некоторые пользователи Linux даже не подозревают о том, что существует консоль. Да, таково новое поколение Linux-пользователей, которым проще использовать графический файловый менеджер, чем вводить команды. Но если вы хотите стать квалифицированным пользователем Linux, то просто обязаны знать, как работать в консоли, иначе уподобитесь Windows-пользователям, которые при каждом сбое переустанавливают операционную систему... Если вы пропустили главу 3, в которой рассматривается работа с консолью, настоятельно рекомендую вернуться и прочитать ее!

Рассмотрим небольшую серию команд (протокол выполнения этих команд приведен на рис. 4.4):

```
touch file.txt
echo "some text" > file.txt
cat file.txt
cp file.txt file-copy.txt
cat file-copy.txt
rm file.txt
cat file.txt
mv file-copy.txt file.txt
cat file.txt
```

Первая команда (`touch`) создает в текущем каталоге файл `file.txt`. Вторая команда (`echo`) записывает строку `some text` в этот же файл. Обратите внимание на символ `>` — это символ перенаправления ввода/вывода, о котором мы поговорим чуть позже.



```
[root@localhost ~]# touch file.txt
[root@localhost ~]# echo "some text" > file.txt
[root@localhost ~]# cat file.txt
some text
[root@localhost ~]# cp file.txt file-copy.txt
[root@localhost ~]# cat file-copy.txt
some text
[root@localhost ~]# rm file.txt
rm: удалить обычный файл `file.txt'? y
[root@localhost ~]# cat file.txt
cat: file.txt: No such file or directory
[root@localhost ~]# mv file-copy.txt file.txt
[root@localhost ~]# cat file.txt
some text
[root@localhost ~]# █
```

Рис. 4.4. Операции с файлом

Третья команда (`cat`) выводит содержимое файла — в файле записанная нами строка `some text`. Четвертая команда (`cp`) копирует файл `file.txt` в файл с именем `file-copy.txt`. После этого мы опять используем команду `cat`, чтобы вывести содержимое файла `file-copy.txt` — надо же убедиться, что файл действительно скопировался.

Шестая команда (`rm`) удаляет файл `file.txt`. При удалении система спрашивает, хотите ли вы удалить файл. Если хотите удалить, то нужно нажать клавишу `<Y>`, а если нет, то `<N>`. Точно ли файл удален? Убедимся в этом: введите команду `cat file.txt`. Система нам сообщает, что нет такого файла.

Восьмая команда (`mv`) переименовывает файл `file-copy.txt` в файл `file.txt`. Последняя команда выводит новый файл `file.txt`. Думаю, особых проблем с этими командами у вас не возникло, тем более, что принцип действия этих команд вам должен быть знаком по командам DOS, которые, как квалифицированный пользователь Windows, вы должны знать наизусть.

Вместо имени файла иногда очень удобно указать *маску имени файла*. Например, у нас есть много временных файлов, имена которых заканчиваются фрагментом `tmp`. Для их удаления нужно воспользоваться командой: `rm *tmp`.

Если же требуется удалить все файлы в текущем каталоге, можно просто указать звездочку: `rm *`.

Аналогично можно использовать символ `?`, который, в отличие от звездочки, заменяющей последовательность символов произвольной длины, заменяет всего один символ. Например, нам нужно удалить все файлы, имена которых состоят из трех букв и начинаются на `S`:

```
rm s??
```

Будут удалены файлы `si4`, `sqm`, `sr6` и т. д., но не будут тронуты файлы, имена которых состоят более чем из трех букв и которые не начинаются на `S`.

Маски имен можно также использовать и при работе с каталогами.

4.4.2. Работа с каталогами

Основные команды для работы с каталогами приведены в табл. 4.2.

Таблица 4.2. Основные команды для работы с каталогами

Команда	Описание
<code>mkdir <каталог></code>	Создание каталога
<code>cd <каталог></code>	Изменение каталога
<code>ls <каталог></code>	Вывод содержимого каталога
<code>rmdir <каталог></code>	Удаление пустого каталога
<code>rm -r <каталог></code>	Рекурсивное удаление каталога

При указании имени каталога можно использовать следующие символы:

- `.` — означает текущий каталог. Если вы введете команду `cat ./file`, то она выведет файл `file`, который находится в текущем каталоге;
- `..` — родительский каталог. Например, команда `cd ..` переведет вас на один уровень вверх по дереву файловой системы;
- `~` — домашний каталог пользователя (об этом мы поговорим позже).

Теперь рассмотрим пример работы с каталогами на практике. Выполните следующие команды:

```
mkdir directory
cd directory
touch file1.txt
touch file2.txt
ls
```



```
cd . .  
ls directory  
rm directory  
rmdir directory  
rm -r directory
```

Первая команда (`mkdir`) создает каталог `directory` в текущем каталоге. Вторая команда (`cd`) переводит (изменяет каталог) в только что созданный каталог. Следующие две команды `touch` создают в новом каталоге два файла: `file1.txt` и `file2.txt`.

Команда `ls` без указания каталога выводит содержимое текущего каталога. Команда `cd ..` переводит в родительский каталог. Как уже было отмечено, в Linux родительский каталог обозначается так: `..` (две точки), а текущий так: `.` (одна точка). То есть, находясь в каталоге `directory`, мы можем обращаться к файлам `file1.txt` и `file2.txt` без указания каталога или же так: `./file1.txt` и `./file2.txt`.

Прямой слэш!

Еще раз обратите внимание — в Linux, в отличие от Windows, для разделения элементов пути служит прямой слэш (`/`), а не обратный (`\`).

Кроме обозначений `..` и `.` в Linux часто используется обозначение `~` — это *домашний каталог*. Предположим, что наш домашний каталог `/home/den`. В нем мы создали подкаталог `dir` и поместили в него файл `file1.txt`. Полный путь к файлу можно записать так:

```
/home/den/dir/file1.txt
```

или же так:

```
~/dir/file1.txt
```

Как видите, тильда (`~`) заменяет часть пути. Удобно? Конечно!

Поскольку мы находимся в родительском для каталога `directory` каталоге, чтобы вывести содержимое только что созданного каталога, в команде `ls` нам нужно четко указать имя каталога:

```
ls directory
```

Команда `rm` используется для удаления каталога. Но что мы видим — система отказывается удалять каталог! Пробуем удалить его командой `rmdir`, но и тут отказ. Система сообщает нам, что каталог не пустой, т. е. содержит файлы. Для удаления каталога нужно удалить все файлы. Конечно, делать это не сильно хочется, поэтому проще указать опцию `-r` команды `rm` для рекурсивного удаления каталога. В этом случае сначала будут удалены все подкаталоги (и все файлы в этих подкаталогах), а затем будет удален сам каталог (рис. 4.5).

Команды `cp` и `mv` работают аналогично: для копирования (перемещения/переименования) сначала указывается каталог-источник, а потом каталог-назначение. Для каталогов желательно указывать параметр `-r`, чтобы копирование (перемещение) производилось рекурсивно.

```
[root@localhost ~]# mkdir directory
[root@localhost ~]# cd directory
[root@localhost directory]# touch file.txt
[root@localhost directory]# touch file2.txt
[root@localhost directory]# ls
file2.txt  file.txt
[root@localhost directory]# cd ..
[root@localhost ~]# ls directory
file2.txt  file.txt
[root@localhost ~]# rm directory
rm: невозможно удалить каталог 'directory': Is a directory
[root@localhost ~]# rmdir directory
rmdir: 'directory': Directory not empty
[root@localhost ~]# rm -r directory
rm: спуститься в каталог 'directory'? y
rm: удалить пустой обычный файл 'directory/file.txt'? y
rm: удалить пустой обычный файл 'directory/file2.txt'? y
rm: удалить Каталог 'directory'? y
[root@localhost ~]# █
```

Рис. 4.5. Операции с каталогами

4.5. Использование ссылок. Команда *ln*

4.5.1. Жесткие и мягкие ссылки

Файлы и каталоги физически хранятся на устройстве хранения в виде набора блоков. Информация о файле (владелец, права доступа, размер файла, время последнего обращения, признак каталога и т. д.) хранится в *inode* — индексном дескрипторе.

Номер *inode* также называют *порядковым номером файла*. Этот номер является уникальным в пределах отдельной файловой системы. Запись каталога содержит имя файла (или каталога), а также указатель на дескриптор *inode*, в котором хранится информация об этом файле (каталоге).

Ссылки — это дополнительные записи каталога, позволяющие обращаться к файлам или каталогам по нескольким именам. *Жесткая* ссылка — это запись каталога, указывающая на дескриптор *inode*, а *мягкая* (или *символическая*) ссылка — это запись каталога, указывающая на имя объекта с другим *inode*.

Механизмы хранения дополнительных имен (ссылок) зависят от типа файловой системы и от длины имени.

Жесткие ссылки можно создать только для файлов, для каталогов их создать невозможно. Исключение составляют лишь специальные записи каталогов, указывающие на сам каталог и на ее родительский каталог, — то есть *.* и *..* являются жесткими ссылками. При попытке создать ссылку для каталога вы увидите следующее сообщение об ошибке:

```
ln: 'link': hard link not allowed for directory
ln: 'link': не допускается создавать жесткие ссылки на каталоги
```

Жесткие ссылки можно использовать только в пределах одной файловой системы, поскольку они являются указателями на дескрипторы `inode`, которые, как уже отмечалось, являются уникальными только в пределах отдельной файловой системы.

Файл удаляется только тогда, когда удаляется последняя ссылка на его `inode`, и счетчик ссылок сбрасывается до 0. Об удалении ссылок мы поговорим позже, так как этот вопрос заслуживает отдельного рассмотрения.

Мягкая (или символическая ссылка, `symlink`) указывает на имя другого файла или каталога, а не на его `inode`. В этом и есть различие мягких ссылок от жестких. Мягкие ссылки можно создавать на объекты разных файловых систем, а также на каталоги. Удаление мягкой ссылки не приводит к удалению файла или каталога, на которую она указывает, а удаление целевого объекта не приводит к автоматическому удалению мягких ссылок. Другими словами, если у вас есть файл `file.txt` и вы создали на него символическую ссылку `symlink.txt`, то в случае удаления файла `file.txt` ссылка окажется «битой» — она не будет ни на что указывать.

4.5.2. Создание ссылок

Для создания ссылок служит команда `ln`:

```
ln file.txt link1
ln -s file.txt link2
```

Первая команда создает жесткую ссылку `link1`, ссылающуюся на текстовый файл `file.txt`. Вторая команда создает символическую ссылку `link2`, которая ссылается на этот же текстовый файл `file.txt`.

Модифицируя ссылку (все равно какую: `link1` или `link2`), вы автоматически модифицируете исходный файл `file.txt`.

4.5.3. Определение ссылок

Мы только что научились создавать ссылки. Теперь давайте посмотрим, как различить — где файл, а где ссылка. Представим, что мы создали файл и две ссылки на него:

```
echo "Hello" > file
ln -s file symlink
ln file hardlink
```

Если ввести команду `ls`, то символическая ссылка будет выделена цветом. Каким именно — зависит от вашего дистрибутива. Так, в Ubuntu символические ссылки выделяются цветом ближе к бирюзовому. Если в вашем дистрибутиве символические ссылки не выделяются, попробуйте указать опцию `--color=auto`:

```
ls --color=auto
```

А вот жесткие ссылки никак не выделяются, и визуально нельзя понять: перед нами файл или ссылка — по крайней мере, в Ubuntu. Однако есть дистрибутивы, где жесткие ссылки выделяются каким-либо цветом, — например, темно-синим.

Рассмотрим вывод команды `ls -l`:

```
ls -l
итого 8
-rw-rw-r-- 2 den den6 июл 15 08:46 file
-rw-rw-r-- 2 den den6 июл 15 08:46 hardlink
lrwxrwxrwx 1 den den4 июл 15 08:46 symlink ->file
```

Как можно видеть, проще всего с символическими ссылками — с помощью стрелки (`->`) сразу показывается, на какой файл указывает ссылка.

Также найти все символические ссылки можно с помощью команды `find`:

```
find . -type l
```

Найти с помощью этой команды все символические ссылки на файл "file" можно так:

```
find . -lname "file"
```

С жесткими ссылками все не так просто. Первая колонка вывода команды `ls -l` — это права доступа. Вторая колонка — счетчик жестких ссылок.

Посмотрим на вывод команды `ls -li`, которая показывает индексные дескрипторы файлов:

```
ls -li
2130783 file
2130783 hardlink
2130784 symlink
```

Как можно видеть, здесь есть два одинаковых дескриптора (2130783) и два имени для этого дескриптора.

4.5.4. Удаление и жесткие ссылки

Мы подходим к самому интересному вопросу. Как уже отмечалось, файл не будет удален, пока на него указывает хоть одна жесткая ссылка.

Ранее приводился вывод команды `ls` с опцией `-li`. Также посмотрите на вывод команды `ls -li is -li` — вторая колонка показывает количество жестких ссылок на inode. По сути, в данном контексте нет особой разницы между ссылкой и файлом. Вы можете удалить файл `file`:

```
rm file
ls -li
итого 4
-rw-rw-r-- 1 den den 6 июл 15 08:46 hardlink
lrwxrwxrwx 1 den den 4 июл 15 08:46 symlink -> file
```

И вы увидите, что счетчик ссылок оказался уменьшен на единицу. Но ваши данные останутся в файле `hardlink`, и вы сможете их прочитать. Почему же файл `file` был удален, если на него указывала жесткая ссылка `hardlink`? Да потому, что жесткая ссылка указывает не на имя файла, а на inode. А ведь индексный дескриптор 2130783 остался, и он не будет удален, пока счетчик жестких ссылок больше 0!

Именно поэтому, если вы хотите защитить файл от удаления путем создания на него жесткой ссылки — это неправильный вариант. Для защиты файла от случайного удаления используйте команду `chattr +i` (она также будет рассмотрена в этой главе позже):

```
touch f
sudo chattr +i f
rm f
rm: удалить защищенный от записи пустой обычный файл 'f'? y
rm: невозможно удалить 'f': Операция не позволена
```

Удалить файл можно, только сняв флаг `i`:

```
chattr -i f
```

Кстати, после удаления файла `file`, на который указывала символическая ссылка `symlink`, последняя превращается в «битую» ссылку— в выводе команды `ls` она отмечается красным цветом на черном фоне. Таким образом, символическая ссылка превращается в «битую» по следующим причинам:

- удаление целевого файла;
- переименование целевого файла;
- переименование элементов пути к целевому файлу. Например, ссылка указывала на файл `/home/den/links/file`, а потом каталог `links` был переименован в `test`.

Возвращаемся к жестким ссылкам— найти все жесткие ссылки на файл можно командой `find` с опцией `-samefile`:

```
find . -samefile file
```

4.5.5. Разница между копированием и созданием жесткой ссылки

Учитывая, что жесткая ссылка— это практически то же самое, что и файл, возникает вопрос: когда лучше копировать файл, а когда создавать ссылки?

Все зависит от поставленных задач. Ведь жесткая ссылка ссылается на тот же `inode`, что и файл, следовательно, при изменении файла (или жесткой ссылки) изменяется и содержимое файла. Если же вы создаете копию файла, то ей будет присвоен другой `inode`, и, следовательно, изменение копии никак не отразится на оригинале и наоборот.

4.6. Права доступа и атрибуты файла.

Команды `chown`, `chmod` и `chattr`

4.6.1. Права доступа к файлам и каталогам

Для каждого каталога и файла вы можете задать права доступа. Точнее, права доступа автоматически задаются при создании каталога/файла, а вам при необходимости можно их изменить. Какая может быть необходимость? Например, вам нужно,

чтобы к вашему файлу-отчету смогли получить доступ пользователи — члены вашей группы. Или вы создали обычный текстовый файл, содержащий инструкции командного интерпретатора. Чтобы этот файл стал сценарием, вам нужно установить право на выполнение для этого файла.

Существуют три права доступа: чтение (r), запись (w), выполнение (x). Для каталога право на выполнение означает право на просмотр содержимого каталога.

Вы можете установить разные права доступа для владельца (т. е. для себя), для группы владельца (т. е. для всех пользователей, входящих в одну с владельцем группу) и для прочих пользователей. Пользователь root может получить доступ к любому файлу или каталогу вне зависимости от прав, которые вы установили.

Чтобы просмотреть текущие права доступа, введите команду:

```
ls -l <имя файла/каталога>
```

Например,

```
ls -l video.txt
```

В ответ программа выведет следующую строку:

```
-r--r----- 1den group 300 Apr 1111:11 video.txt
```

В этой строке фрагмент: **-r--r-----** описывает права доступа:

- первый символ — это признак каталога. Сейчас перед нами файл. Если бы перед нами был каталог, то первый символ был бы символом **d** (от directory);
- последующие три символа (**r--**) определяют *права доступа владельца файла или каталога*. Первый символ — это чтение, второй — запись, третий — выполнение. Как можно видеть, владельцу разрешено только чтение этого файла, запись и выполнение запрещены, поскольку в правах доступа режимы **w** и **x** не определены;
- следующие три символа (**r--**) задают *права доступа для членов группы владельца*. Права такие же, как и у владельца: можно читать файл, но нельзя изменять или запускать;
- последние три символа (**---**) задают *права доступа для прочих пользователей*. Прочие пользователи не имеют права ни читать, ни изменять, ни выполнять файл. При попытке получить доступ к файлу они увидят сообщение **Access denied**.

Полный вывод команды ls

Как можно видеть, после символов прав доступа команда `ls` выводит имя владельца файла, имя группы владельца, размер файла, дату и время создания, а также имя файла.

Права доступа задаются командой `chmod`. Существуют два способа указания прав доступа: *символьный* (когда указываются символы, задающие право доступа, — **r**, **w**, **x**) и *абсолютный*.

Так уж заведено, что в мире UNIX чаще пользуются абсолютным методом. Разберемся, в чем он заключается, и рассмотрим следующий набор прав доступа:

`rw-r---`

Этот набор предоставляет владельцу право чтения и модификации файла (`rw-`), запускать файл владелец не может. Члены группы владельца могут только просматривать файл (`r--`), а все остальные пользователи не имеют вообще никакого доступа к файлу.

Возьмем отдельный набор прав, **например**, для владельца: `rw-`.

Чтение разрешено — мысленно записываем 1, запись разрешена — запоминаем еще 1, а вот выполнение запрещено, поэтому запоминаем 0. Получается число 110. Если из двоичной системы перевести число ПО в восьмеричную, получится число 6. Для перевода можно воспользоваться табл. 4.3.

Таблица 4.3. Преобразование чисел из двоичной системы в восьмеричную

Двоичная система	Восьмеричная система	Двоичная система	Восьмеричная система
000	0	100	4
001	1	101	5
010	2	110	6
011	3	111	7

Аналогично произведем разбор прав для членов группы владельца. Получится двоичное 100, т. е. восьмеричное 4. С третьим набором (`---`) все вообще просто — это 000, т. е. 0.

Записываем полученные числа в восьмеричной системе в порядке владелец-группа-остальные. Получится число 640 — это и есть права доступа. Для того чтобы установить эти права доступа, выполните команду:

```
chmod 640 <имя_файла>
```

Наиболее популярные права доступа:

- 644 — владельцу можно читать и изменять файл, остальным пользователям — только читать;
- 666 — читать и изменять файл можно всем пользователям;
- 777 — всем можно читать, изменять и выполнять файл.

Право выполнения для каталога

Напомню, что для каталога право выполнения — это право просмотра оглавления каталога.

Иногда символьный метод оказывается проще. Например, чтобы файл `script` сделать исполнимым, можно отдать команду:

```
chmod +x script
```

Для того чтобы снять право выполнения, указывается параметр `-x`:

```
chmod -x script
```

Подробнее о символьном методе вы сможете прочитать в руководстве по команде `chmod` (выполнив команду `man chmod`).

4.6.2. Смена владельца файла

Если вы хотите «подарить» кому-то файл, т. е. сделать какого-либо пользователя владельцем файла, вам нужно использовать команду `chown`:

```
chown пользователь файл
```

ПОСЛЕДСТВИЯ ИЗМЕНЕНИЯ ВЛАДЕЛЬЦА ФАЙЛА

Возможно, что после изменения владельца файла вы сами не сможете получить к нему доступ, ведь владельцем будете уже не вы.

4.6.3. Специальные права доступа (SUID и SGID)

Мы рассмотрели обычные права доступа к файлам, но в Linux есть еще так называемые *специальные права доступа*: SUID (Set User ID root) и SGID (Set Group ID root).

Эти права доступа позволяют обычным пользователям запускать программы, требующие для своего запуска привилегий пользователя `root`. Например, демон `pppd` требует привилегий `root`, но чтобы каждый раз при установке PPP-соединения (модемное или ADSL-соединение) не входить в систему под именем `root`, достаточно установить специальные права доступа для демона `pppd`. Делается это так:

```
chmod u+s /usr/sbin/pppd
```

Однако не нужно увлекаться такими решениями, поскольку каждая программа, для которой установлен бит SUID, является потенциальной «дырой» в безопасности системы. Для выполнения программ, требующих прав `root`, намного рациональнее использовать программы `sudo` и `su` (описание которых можно получить по командам `man sudo` и `man su`).

4.6.4. Атрибуты файла. Запрет изменения файла

С помощью команды `chattr` можно изменить атрибуты файла. Параметр `+` устанавливает атрибут, а параметр `-` атрибут снимает. Например:

```
# chattr +i /boot/grub/menu.lst
```

Эта команда устанавливает атрибут `i`, запрещающий любое изменение, переименование и удаление файла. Установить этот атрибут, равно как и снять его, имеет право только суперпользователь или процесс с возможностью `CAP_LINUX_IMMUTABLE`. Чтобы изменить файл, нужно очистить атрибут с помощью команды:

```
# chattr -i /boot/grub/menu.lst
```


Если установить атрибут `j`, то все данные, прежде чем быть записанными непосредственно в файл, будут сохранены в журнал файловой системы. Этот атрибут имеет смысл только, если файловая система смонтирована с опциями `data=ordered` или `data=writeback` (см. *разд. 4.8*). Когда файловая система смонтирована с опцией `data=journal`, установка атрибута `j` не имеет смысла, поскольку все данные файла и так уже журналируются.

Рассмотрим еще несколько атрибутов:

- когда для файла установлен атрибут `A` (прописная буква!), тогда не происходит обновление записи `atime` (в ней хранится время доступа к файлу). Это позволяет избежать лишних дисковых операций ввода/вывода, что полезно для медленных компьютеров;
- если для файла установлен атрибут `a`, в файл можно только добавлять данные. Этот атрибут имеет право установить (или очистить) суперпользователь или процесс с возможностью `CAP_LINUX_IMMUTABLE`;
- атрибут `s` заставляет систему упаковывать (сжимать) содержимое файла, что позволяет сэкономить место на диске. При записи в файл информация автоматически сжимается и записывается на диск в уже сжатом виде, при чтении из этого файла возвращаются несжатые данные;
- когда изменяется каталог с установленным атрибутом `D`, изменения сразу же записываются на диск. Это эквивалентно применению опции монтирования `dirsync`;
- если для файла установлен атрибут `d`, для него не будет выполнено резервное копирование программой `dump`;
- при изменении файла с установленным атрибутом `S` его данные синхронно записываются на диск. Это аналогично опции монтирования `sync` к подмножеству файлов;
- когда удаляется файл с установленным атрибутом `S`, система выполняет обнуление его блоков и запись их обратно на диск;
- при удалении файла с атрибутом `i` и его содержимое сохраняется на диске, что позволяет впоследствии легко восстановить этот файл;
- атрибуты `X` и `Z` используются экспериментальными заплатками сжатия для служебных целей.

Установить любой атрибут можно командой `chattr`, а просмотреть — командой `lsattr`. Об остальных атрибутах вы сможете прочитать в справочной системе:

`man chattr`

4.6.5. Команды поиска файлов: *find*, *which* и *locate*

Для поиска файлов в Linux служит команда `find`. Это довольно мощная утилита со сложным синтаксисом, и далеко не всегда она нужна обычному пользователю. Намного проще установить файловый менеджер `mc` и использовать встроенную в него функцию поиска.

Но команду `find` мы все же рассмотрим, по крайней мере ее основы. Синтаксис команды следующий:

```
find список_поиска выражение
```

Мощность команды `find` заключается во множестве самых разных параметров поиска, которые из-за их количества не так-то просто запомнить. К тому же `find` может выполнять операции над найденными файлами. Например, вы можете найти временные файлы и сразу удалить их.

Подробно опции команды `find` мы изучать не станем — это вы можете сделать самостоятельно с помощью команды `man find`. Здесь мы рассмотрим лишь несколько примеров использования этой команды:

- найти файлы с именем `a.out` (точнее, в имени которых содержится строка «a.out»), поиск начать с корневого каталога (/):

```
find / -name a.out
```

- найти файлы по маске `*.txt`:

```
find / -name '*.txt'
```

- найти файлы нулевого размера, поиск начать с текущего каталога (.):

```
find . -size 0c
```

Кстати, для поиска пустых файлов намного проще использовать параметр `-empty`:

```
find . -empty
```

- найти файлы, размер которых от 100 до 150 Мбайт, поиск производить в домашнем каталоге и всех его подкаталогах:

```
find ~ -size +100M -size -150M
```

- найти все временные файлы и удалить их (для каждого найденного файла будет запущена команда `rm`):

```
# find / -name *.tmp -ok rm {} \;
```

Вместо параметра `-ok` можно использовать параметр `-exec`, который также запускает указанную после него команду, но не запрашивает подтверждение выполнения этой команды для каждого файла.

Кроме команды `find` можно использовать команды `which` и `locate`. Первая выводит полный путь к программе или к сценарию, если программа или сценарий находится в списке каталогов, заданном в переменной окружения `PATH`:

```
which sendmail
```

Команда `locate` ищет в базе данных демона `located` файлы, соответствующие заданному образцу. Недостаток этой команды в том, что `located` имеется далеко не во всех дистрибутивах, поэтому команды `locate` у вас может и не быть. Зато если `located` имеется и запущен, поиск файлов будет осуществляться быстрее, чем с помощью `find`.

4.7. Монтирование файловых систем

4.7.1. Команды *mount* и *umount*

Чтобы работать с какой-либо файловой системой, необходимо *примонтировать* ее к корневой файловой системе. Например, вставив в разъем USB флешку, нужно подмонтировать файловую систему флешки к корневой файловой системе, — только так мы сможем получить доступ к файлам и каталогам, которые на этой флешке записаны. Аналогичная ситуация с жесткими, оптическими дисками и другими носителями данных.

Если вы хотите заменить сменный носитель данных (флешку, компакт-диск), вам нужно сначала размонтировать файловую систему, затем извлечь носитель данных, установить новый и заново смонтировать файловую систему. В случае с флешкой о размонтировании должны помнить вы сами, поскольку при этом выполняется синхронизация буферов ввода/вывода и файловой системы, т. е. данные физически записываются на диск, если это еще не было сделано. А компакт-диск система не разрешит вам извлечь, если он не размонтирован. В свою очередь, размонтировать файловую систему можно только тогда, когда ни один процесс ее не использует.

При завершении работы системы (перезагрузке, выключении компьютера) размонтирование всех файловых систем выполняется автоматически.

Команда монтирования (ее нужно выполнять с привилегиями root) выглядит так:

```
# mount [опции] <устройство> <точка монтирования>
```

Здесь точка монтирования — это каталог, через который будет осуществляться доступ к монтируемой файловой системе. Например, если вы подмонтировали компакт-диск к каталогу `/mnt/cdrom`, то получить доступ к файлам и каталогам, записанным на компакт-диске, можно будет через точку монтирования (именно этот каталог: `/mnt/cdrom`). Точкой монтирования может быть любой каталог корневой файловой системы, хоть `/aaa-111`. Главное, чтобы этот каталог существовал на момент монтирования файловой системы.

В некоторых современных дистрибутивах запрещен вход в систему под именем суперпользователя — root. Поэтому для выполнения команд с привилегиями root вам нужно использовать команду `sudo`. Например, чтобы выполнить команду монтирования привода компакт-диска, вам нужно ввести команду:

```
sudo mount /dev/scd0 /mnt/cdrom
```

Перед выполнением команды `mount` команда `sudo` попросит вас ввести пароль root. Если введенный пароль правильный, то будет выполнена команда `mount`.

Для размонтирования файловой системы служит команда `umount`:

```
# umount <устройство или точка монтирования>
```

4.7.2. Файлы устройств и монтирование

В этой главе мы уже говорили о файлах устройств. Здесь мы вернемся к ним снова, но в контексте монтирования файловой системы.

Как уже было отмечено, для Linux нет разницы между устройством и файлом. Все устройства системы представлены в корневой файловой системе как обычные файлы. Например, `/dev/fd0` — это ваш дисковод для гибких дисков (ведь вы все еще помните, что это за устройство?), `/dev/sda` — жесткий диск. Файлы устройств хранятся в каталоге `/dev`.

Жесткие диски

С жесткими дисками сложнее всего, поскольку одно и то же устройство может в разных версиях одного и того же дистрибутива называться по-разному. Например, мой IDE-диск, подключенный как первичный мастер, в Fedora 5 все еще назывался `/dev/hda`, а, начиная с Fedora 8, он называется `/dev/sda`. Раньше накопители, подключающиеся к интерфейсу IDE (PATA), назывались `/dev/hdx`, а SCSI/SATA-накопители — `/dev/sdx` (где в обоих случаях *x* — буква).

После внедрения менеджера устройств `udev`¹ и принятия глобального уникального идентификатора устройств (UUID) все дисковые устройства, вне зависимости от интерфейса подключения (PATA, SATA, SCSI), называются `/dev/sdx`, где *x* — буква. Все современные дистрибутивы поддерживают `udev` и UUID. Так что не удивляйтесь, если вдруг ваш старенький IDE-винчестер будет назван `/dev/sda`. С одной стороны, это вносит некоторую путаницу (см. *разд. 4.7.5*). С другой — все современные компьютеры оснащены именно SATA-дисками (т. к. PATA-диски уже устарели, а SCSI — дорогие), а на современных материнских платах только один контроллер IDE (PATA), потому многие пользователи даже ничего не заметят.

Рассмотрим ситуацию с жесткими дисками чуть подробнее. Пусть у нас есть устройство `/dev/sda`. На жестком диске, понятное дело, может быть несколько разделов. Рассмотрим ситуацию, когда на диске имеются три раздела (логических диска), которые в Windows называются C:, D: и E:. Диск C: обычно является загрузочным (активным), поэтому этот раздел будет записан в самом начале диска. Нумерация разделов жесткого диска в Linux начинается с 1, и в большинстве случаев диску C: будет соответствовать имя `/dev/sda 1` — первый раздел на первом жестком диске.

Резонно предположить, что двум оставшимся разделам (D: и E:) будут присвоены имена `/dev/sda2` и `/dev/sda3`. Это может быть и так, и не так. Как известно, на жестком диске могут существовать или четыре первичных раздела, или три первичных и один расширенный. В расширенном разделе могут разместиться до 11 логических дисков (разделов). Таким образом, раздел может быть первичным (primary partition), расширенным (extended partition) или логическим (logical partition).

¹ `udev` — это менеджер устройств, впервые появившийся в ядре Linux, начиная с версии 2.6. Пришел на смену более громоздкой псевдофайловой системе `devfs`. Управляет всеми манипуляциями с файлами из каталога `/dev`.

Для возможных четырех первичных разделов диска в Linux зарезервированы номера 1, 2, 3, 4. Если разделы D: и E: нашего диска первичные, то, да — им будут присвоены имена `/dev/sda2` и `/dev/sda3`. Но, в большинстве случаев, эти разделы являются логическими и содержатся в расширенном разделе. Логические разделы именуются, начиная с 5, а это означает, что если разделы D: и E: — логические, им будут присвоены имена `/dev/sda5` и `/dev/sda6` соответственно.

РАСШИРЕННЫЙ РАЗДЕЛ WINDOWS

В Windows расширенному разделу не присваивается буква, потому что этот раздел не содержит данных пользователя, а только информацию о логических разделах.

Узнать номер раздела очень просто — достаточно запустить утилиту, работающую с таблицей разделов диска. В Fedora придется использовать стандартный `fdisk` или `cfdisk` (он немного удобнее), а в Debian/Ubuntu — `GParted` (кстати, очень удобное средство разметки диска). В openSUSE нужно выполнить команду Компьютер | Центр управления | YaST, а в открывшемся окне нажать кнопку Средство разметки. Впрочем, в большинстве случаев удобнее всего запустить (от имени root) утилиту `fdisk` — она есть в любом дистрибутиве Linux.

Чтобы узнать номера разделов первого жесткого диска (`/dev/sda`), введите команду:

```
# /sbin/fdisk /dev/sda
```

В ответ на появившееся приглашение `fdisk` нужно ввести `p` и нажать клавишу `<Enter>` — вы увидите таблицу разделов (рис. 4.6). Для выхода из программы введите `q` и нажмите клавишу `<Enter>`.

```
den@localhost:/home/den
Файл Правка Вид Терминал Вкладки Справка
1) программами, запускаемым при загрузке (напр., старые версии LILO)
2) загрузкой и программами разметки из других ОС
   (напр., DOS FDISK, OS/2 FDISK)

Команда (m для справки): p

Диск /dev/sda: 160.0 ГБ, 160041885696 байт
255 heads, 63 sectors/track, 19457 cylinders
Units = цилиндры of 16065 * 512 = 8225280 bytes
Disk identifier: 0xe905e905

Устр-во Загр   Начало      Конец      Блоки   Id Система
/dev/sda1 *    1           543        4361616 b  W95 FAT32
/dev/sda2      544         19457     151926705 f  W95 расшир. (LBA)
/dev/sda5      544         1021       3839503+ 83  Linux
/dev/sda6     1022        1759       5927953+ 83  Linux
/dev/sda7     1760        1825       530113+ 82  Linux свон / Solaris
/dev/sda8     1826        5963     33238453+ b  W95 FAT32
/dev/sda9     5964       10101     33238453+ b  W95 FAT32
/dev/sda10    10102       14268     33471396 b  W95 FAT32
/dev/sda11    14269       16949     21535101 b  W95 FAT32
/dev/sda12    16950       19457     20145478+ b  W95 FAT32

Команда (m для справки):
```

Рис. 4.6. Таблица разделов жесткого диска

На рис. 4.6 изображена таблица разделов первого жесткого диска моего компьютера. Первый раздел (это мой диск C:, где установлена система Windows)— первичный. Сразу после него расположен расширенный раздел (его номер— 2). Следующий за ним — логический раздел (номер 5). Разделы с номерами 3 и 4 пропущены, потому что их нет на моем жестком диске. Это те самые первичные разделы, которые я не создал, — они мне не нужны.

Приводы оптических дисков

Приводы для чтения/записи CD/DVD называются `/dev/scdN`, где *N*— номер устройства. Если у вас только один привод CD/DVD, то его имя будет `/dev/scd0`.

Монтирование привода для чтения оптических дисков осуществляется командой:

```
# mount /dev/scd0 /mnt/cdrom
```

После этого обратиться к файлам, записанным на диске, можно будет через каталог `/mnt/cdrom`. Напомню, что этот каталог должен существовать.

Флешки и внешние жесткие диски

Флешка (флеш-память) и внешние USB-диски определяются системой как обычные жесткие диски. Предположим, что в компьютере установлен всего один жесткий диск, тогда ему соответствует имя устройства `/dev/sda`.

Когда вы подключите флешку или внешний жесткий диск, этому устройству будет присвоено имя `/dev/sdb`. Обычно на флешке или USB-диске всего один раздел, поэтому подмонтировать устройство можно командой:

```
# mount /dev/sdb1 /mnt/usbdisk
```

Далее (см. *разд. 4.7.6*) мы поговорим о монтировании флешек (и устройств, определяемых как флешки: цифровых фотоаппаратов, видеокамер, мобильных телефонов) более подробно. А пока нужно отметить, что в современных дистрибутивах флешки, внешние жесткие диски и диски CD/DVD монтируются автоматически (правда, не к подкаталогу `/mnt` — чаще для этих целей используется каталог `/media`, но все зависит от дистрибутива), и вся информация на этот счет здесь приведена для общего развития или на аварийный случай — когда вы загрузите систему в однопользовательском режиме, и вам придется монтировать носители вручную.

4.7.3. Опции монтирования файловых систем

Теперь, когда мы знаем номер раздела, можно подмонтировать его файловую систему. Делается это так:

```
# mount <раздел> <точка монтирования>
```

Например:

```
# mount /dev/sda5 /mnt/win d
```

У команды `mount` довольно много опций, но на практике наиболее часто используются только некоторые из них: `-t`, `-r`, `-w`, `-a`.

- Параметр `-t` позволяет задать тип файловой системы. Обычно программа сама определяет файловую систему, но иногда это у нее не получается. Тогда мы должны ей помочь. Формат этого параметра следующий:

```
# mount -t <файловая система> <устройство> <точка монтирования>
```

Например,

```
# mount -t iso9660 /dev/sdc /mnt/cdrom
```

Вот опции для указания наиболее популярных монтируемых файловых систем:

- `ext2`, `ext3`, `ext4` — файловая система Linux;
- `iso9660` — указывается при монтировании CD-ROM;
- `vfat` — FAT, FAT32 (поддерживается Windows 9x, ME, XP);
- `ntfs` — NT File System (поддерживается Windows NT, XP, 7, 8, 10). Будет использована стандартная поддержка NTFS, при которой NTFS-раздел доступен только для чтения;
- `ntfs-3g` — будет запущен модуль `ntfs-3g`, входящий в большинство современных дистрибутивов. Этот модуль позволяет производить запись информации на NTFS-разделы.

Модуль NTFS-3G

Если в вашем дистрибутиве нет модуля `ntfs-3g`, т. е. при попытке указания файловой системы NTFS будет выведено сообщение об ошибке, вы можете скачать его с сайта **www.ntfs-3g.org**. На этом сайте доступны как исходные коды, так и уже откомпилированные для разных дистрибутивов пакеты.

Если вы не можете смонтировать NTFS-раздел с помощью опции `ntfs-3g`, то, вероятнее всего, он был неправильно размонтирован (например, работа Windows не была завершена корректно). В этом случае для монтирования раздела нужно использовать опцию `-o force`, например:

```
sudo mount -t ntfs-3g /dev/sdb1 /media/usb -o force
```

- Параметр `-r` монтирует указанную файловую систему в режиме «только чтение».
- Параметр `-w` монтирует файловую систему в режиме «чтение/запись». Этот параметр используется по умолчанию для файловых систем, поддерживающих запись (например, NTFS по умолчанию запись не поддерживает, как и файловые системы CD/DVD-дисков).
- Параметр `-a` служит для монтирования всех файловых систем, указанных в файле `/etc/fstab` (кроме тех, для которых указано `noauto`, — такие файловые системы нужно монтировать вручную). При загрузке системы тогда вызывается команда `mount` с параметром `-a`.

4.7.4. Монтирование разделов при загрузке

Если вы не хотите при каждой загрузке монтировать постоянные файловые системы (например, Windows-разделы), то нужно прописать их в файле `/etc/fstab`. Обратите внимание — в этом файле не следует прописывать файловые системы сменных носителей (дисководов, CD/DVD-приводов, флешки). Следует отметить, что программы установки некоторых дистрибутивов читают таблицу разделов и автоматически заполняют файл `/etc/fstab` — в результате все ваши Windows-разделы оказываются доступны сразу после установки системы. К сожалению, не все дистрибутивы могут похвастаться такой интеллектуальностью, поэтому вам нужно знать формат файла `fstab`:

устройство точка монтирования тип_ФС опции флаг_РК флаг_проверки

Здесь: тип_ФС — это тип файловой системы, а флаг_РК — флаг резервного копирования. Если он установлен (1), то программа `dump` заархивирует эту файловую систему при создании резервной копии. Если не установлен (0), то резервная копия ее создаваться не будет. Флаг_проверки устанавливает, будет ли эта файловая система проверяться на наличие ошибок программой `fsck`. Проверка производится в двух случаях:

- если файловая система размонтирована некорректно;
- если достигнуто максимальное число операций монтирования для этой файловой системы.

Поле опции содержит важные параметры файловой системы. Некоторые из них представлены в табл. 4.4.

Таблица 4.4. Опции монтирования файловой системы в файле `/etc/fstab`

Опция	Описание
auto	Файловая система должна монтироваться автоматически при загрузке. Опция используется по умолчанию, поэтому ее указывать не обязательно
noauto	Файловая система не монтируется при загрузке системы (при выполнении команды <code>mount -a</code>), но ее можно смонтировать вручную с помощью все той же команды <code>mount</code>
defaults	Используется стандартный набор опций, установленных по умолчанию
exec	Разрешает запуск выполняемых файлов для этой файловой системы. Опция используется по умолчанию
noexec	Запрещает запуск выполняемых файлов для этой файловой системы
ro	Монтирование в режиме «только чтение»
rw	Монтирование в режиме «чтение/запись». Используется по умолчанию для файловых систем, поддерживающих запись
user	Эту файловую систему разрешается монтировать/размонтировать обычному пользователю (не <code>root</code>)
nouser	Файловую систему может монтировать только пользователь <code>root</code> . Используется по умолчанию

Таблица 4.4 (окончание)

Опция	Описание
umask	Определяет маску прав доступа при создании файлов. Для не-Linux файловых систем маску нужно установить так: <code>umask=0</code>
utf8	Применяется только на дистрибутивах, которые используют кодировку UTF8 в качестве кодировки локали. В старых дистрибутивах (где используется KOI8-R) для корректного отображения русских имен файлов на Windows-разделах нужно задать параметры <code>iocharset=koi8-u, codepage=866</code>

РЕДАКТИРОВАНИЕ ФАЙЛА /ETC/FSTAB

Редактировать файл `/etc/fstab`, как и любой другой файл из каталога `/etc`, можно в любом текстовом редакторе (например, `gedit`, `kate`), но перед этим нужно получить права `root` (командами `su` или `sudo`).

Рассмотрим небольшой пример:

```
/dev/sdc /mnt/cdrom auto umask=0,user,noauto,ro,exec 0 0
/dev/sdal /mnt/win_c vfat umask=0,utf8 0 0
```

Первая строка— это строка монтирования файловой системы компакт-диска, а вторая — строка монтирования диска C:.

- Начнем с первой строки, `/dev/sdc` — это имя устройства CD-ROM. Точка монтирования— `/mnt/cdrom`. Понятно, что этот каталог должен существовать. Обратите внимание — в качестве файловой системы не указывается жестко `iso9660`, поскольку компакт-диск может быть записан в другой файловой системе, поэтому в качестве типа файловой системы задано `auto`, т. е. автоматическое определение. Теперь идет довольно длинный набор опций. Ясно, что `umask` установлен в ноль, поскольку файловая система компакт-диска не поддерживает права доступа Linux. Параметр `user` говорит о том, что эту файловую систему можно монтировать обычному пользователю. Параметр `noauto` запрещает автоматическое монтирование этой файловой системы, что правильно— ведь на момент монтирования в приводе может и не быть компакт-диска. Опция `ro` разрешает монтирование в режиме «только чтение», а `exec` разрешает запускать исполнимые файлы. Понятно, что компакт-диск не нуждается ни в проверке, ни в создании резервной копии, поэтому два последних флага равны нулю.
- Вторая строка проще. Первые два поля — это устройство и точка монтирования. Третье — тип файловой системы. Файловая система постоянна, поэтому можно явно указать тип файловой системы (`vfat`), а не `auto`. Опция `umask`, как и в предыдущем случае, равна нулю. Указание опции `utf8` позволяет корректно отображать русскоязычные имена файлов и каталогов.

4.7.5. Подробно о UUID и файле /etc/fstab

Пока вы еще не успели забыть формат файла `/etc/fstab`, поговорим о UUID (Universally Unique Identifier), или о *длинных именах* дисков. В некоторых дистрибутивах,

например в Ubuntu, вместо имени носителя (первое поле файла `fstab`) указывается его ID, поэтому файл `fstab` выглядит устрашающе, вот так:

```
# /dev/sda6
UUID=1f049af9-2bdd-43bf-a16c-ff5859a4116a / ext3 defaults 0 1
# /dev/sda1
UUID=45AE-84D9 /media/sda1 vfat defaults,utf8,umask=007 0 0
```

В openSUSE идентификаторы устройств указываются немного иначе:

```
/dev/disk/by-id/scsi-SATA_WDC_WD1600JB-00_WD-WCANM7959048-part5 / ext3
acl,user_xattr 1 1
/dev/disk/by-id/scsi-SATA_WDC_WD1600JB-00_WD-WCANM7959048-part7 swap swap
defaults 0 0
```

Понятно, что использовать короткие имена вроде `/dev/sda1` намного проще, чем идентификаторы в стиле `1f04 9af9-2bdd-43bf-a16c-ff5859a4116a`. Использование имен дисков еще никто не отменял, поэтому вместо идентификатора носителя можете смело указывать его файл устройства — так вам будет значительно проще!

Но все же разбираться в соответствии длинных имен дисков коротким именам устройств следует — ведь система использует именно эти имена, а в файле `/etc/fstab` не всегда указывается, какой идентификатор принадлежит какому короткому имени устройства (или указывается, но не для всех разделов).

Узнать длинные имена устройств можно с помощью простой команды:

```
ls -l /dev/disk/by-uuid/
```

Результат выполнения этой команды приведен на рис. 4.7, а. Можно использовать и следующую команду (рис. 4.7, б):

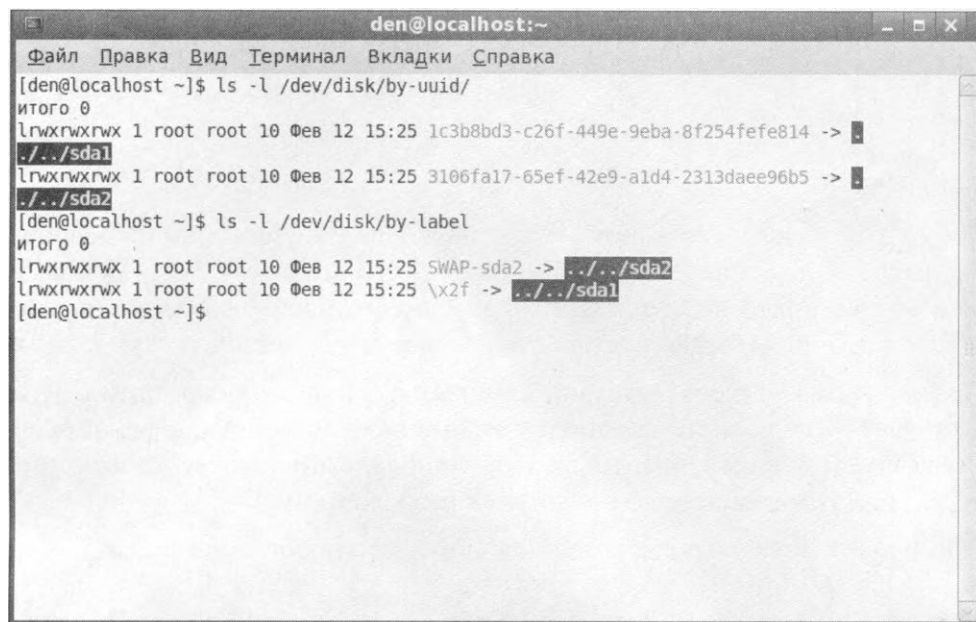
```
sudo blkid
```

Спрашивается, зачем было вводить длинные имена, если короткие имена удобнее, во всяком случае для пользователей? Оказывается, разработчики Linux в первую очередь и заботились как раз о пользователях. Возьмем обычный IDE-диск. Как известно, его можно подключить либо к первичному (primary), либо к вторичному (secondary), если он есть, контроллеру. При этом, согласно положению переключки выбора режима, винчестер может быть либо главным устройством (master), либо подчиненным (slave). Таким образом, в зависимости от контроллера, к которому подключается диск, изменяется его короткое имя: `sda` (primary master), `sdb` (primary slave), `sdc` (secondary' master), `sdd` (secondary slave). То же самое происходит с SATA/SCSI-винчестерами — при изменении параметров подключения изменяется и короткое имя устройства.

При использовании же длинных имен идентификатор дискового устройства остается постоянным вне зависимости от типа подключения устройства к контроллеру. Именно поэтому длинные имена дисков часто также называются *постоянными именами* (persistent name). Получается, что если подключить жесткий диск немного иначе, то разделы, которые назывались, скажем, `/dev/sdaN`, стали бы называться `/dev/sdbN`. Понятно, что загрузить Linux с такого диска не получится, поскольку везде указаны другие имена устройств. Если же используются длинные имена дисков,

система загрузится в любом случае, как бы вы ни подключили жесткий диск. Удобно? Конечно.

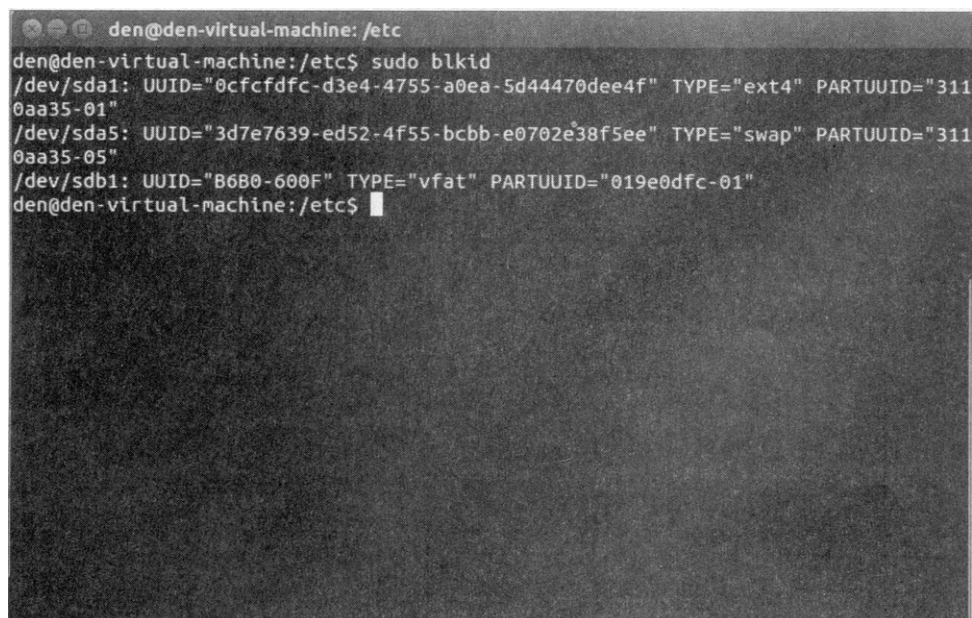
Но это еще не все. Постоянные имена — это только первая причина. Вторая причина заключается в обновлении библиотеки `libata`. В новой версии `libata` все PATA-устройства именуются не как `hdx`, а как `sdx`, что (как отмечалось в этой главе ранее)



```

den@localhost:~
Файл Правка Вид Терминал Вкладки Справка
[den@localhost ~]$ ls -l /dev/disk/by-uuid/
итого 0
lrwxrwxrwx 1 root root 10 Фев 12 15:25 1c3b8bd3-c26f-449e-9eba-8f254fefe814 -> ../../sda1
lrwxrwxrwx 1 root root 10 Фев 12 15:25 3106fa17-65ef-42e9-ald4-2313daee96b5 -> ../../sda2
[den@localhost ~]$ ls -l /dev/disk/by-label
итого 0
lrwxrwxrwx 1 root root 10 Фев 12 15:25 SWAP-sda2 -> ../../sda2
lrwxrwxrwx 1 root root 10 Фев 12 15:25 \x2f -> ../../sda1
[den@localhost ~]$
  
```

а



```

den@den-virtual-machine: /etc
den@den-virtual-machine:/etc$ sudo blkid
/dev/sda1: UUID="0cfcfdcf-d3e4-4755-a0ea-5d44470dee4f" TYPE="ext4" PARTUUID="3110aa35-01"
/dev/sda5: UUID="3d7e7639-ed52-4f55-bcbb-e0702e38f5ee" TYPE="swap" PARTUUID="3110aa35-05"
/dev/sdb1: UUID="B6B0-600F" TYPE="vfat" PARTUUID="019e0dfc-01"
den@den-virtual-machine:/etc$
  
```

б

Рис. 4.7. Соответствие длинных имен дисков коротким: **а** — команда `ls -l /dev/disk/by-uuid/`; **б** — команда `sudo blkid`

вносит некую путаницу. Длинные же имена дисков от этого не изменяются и избавляют пользователя от беспокойства по поводу того, что его старый IDE-диск вдруг превратился в SATA/SCSI-диск.

При использовании UUID однозначно идентифицировать раздел диска можно несколькими способами:

- `UUID=45AE-84D9 /media/sdal vfat defaults,utf8,umask=007, gid=46 0 0` — Здесь с помощью параметра UUID указывается идентификатор диска;
- `/dev/disk/by-id/scsi-SATA_WDC_WD1600JB-00_WD-WCANM7959048-part7 swap swap defaults 0 0` — здесь указывается длинное имя устройства диска;
- `label=/ / ext3 defaults 1 1` — самый компактный третий способ, позволяющий идентифицировать устройства по их метке.

Способы получения длинных имен

Первый способ получения длинного имени в англоязычной литературе называется «by-uuid», т. е. длинное имя составляется по UUID, второй способ называется «by-id», т. е.— по аппаратному идентификатору устройства. Третий способ называется «by-label» — по метке. Просмотреть соответствие длинных имен коротким можно с помощью команд:

```
ls -l /dev/disk/by-uuid
ls -l /dev/disk/by-id
ls -l /dev/disk/by-label
```

Но есть еще и четвертый способ, который называется «by-path» — в этом случае имя генерируется по sysfs. Этот способ является наименее используемым, поэтому вы редко столкнетесь с ним.

Узнать метки разделов можно с помощью команды:

```
ls -lF /dev/disk/by-label
```

Установить метку можно с помощью команд, указанных в табл. 4.5.

Таблица 4.5. Команды для установки меток разделов

Файловая система	Команда
ext2/ext3/ext4	# e2label /dev/XXX <метка>
ReiserFS	# reiserfstune -l <метка> /dev/XXX
JFS	# jfs tune -L <метка> /dev/XXX
XFS	# xfs admin -L <label> /dev/XXX
FAT/FAT32	Только средствами Windows
NTFS	# ntfslabel /dev/XXX <метка>

В файле `/etc/fstab` вы можете использовать длинные имена в любом формате: можно указывать имена устройств в виде: `/dev/disk/by-uuid/*`, `/dev/disk/by-id/*` или

/dev/disk к/by-label/*, можно использовать параметры UUID=идентификатор или LABEL=метка. Используйте тот способ, который вам больше нравится.

4.7.6. Монтирование флеш-дисков

В последнее время стала очень популярна флеш-память. Уже сегодня флеш-память, точнее флеш-диски (они же USB-диски или попросту флешки), построенные с использованием флеш-памяти, практически вытеснили обычные дискеты — они очень компактны и позволяют хранить довольно большие объемы информации. Сегодня никого не удивит небольшим брелоком, вмещающим 8-16 Гбайт.

Принцип использования флеш-диска очень прост — достаточно подключить его к шине USB, и через несколько секунд система его определит. Далее с ним можно будет работать как с обычным диском. Да, флеш-диски не очень шустры, но молниеносной реакции от них никто и не ожидает — во всяком случае, они выглядят настоящими спринтерами на фоне обычных дискет.

Технология флеш-памяти нашла свое применение в различных портативных устройствах: от мобильных телефонов до цифровых фотоаппаратов. Вы можете подключить мобильник к компьютеру и работать с ним как с обычным диском — записывать на него мелодии и картинки. Аналогичная ситуация и с цифровым фотоаппаратом — когда вы фотографируете, то фотографии и видеоролики записываются на его флеш-память. Потом вам нужно подключить фотоаппарат к компьютеру и просто скопировать фотографии. Вы также можете записать фотографии (или другие файлы — не имеет значения) на фотоаппарат, используя его встроенную флеш-память как большую дискету — для переноса своих файлов.

Все современные дистрибутивы умеют автоматически монтировать флеш-диски. После монтирования открывается окно (рис. 4.8) либо с предложением просмотреть содержимое диска или же импортировать фотографии (в зависимости от типа подключенного устройства — обычный это USB-диск или фотоаппарат), либо сразу с содержимым диска (для обычной флешки с файлами).

Понятно, что нам, как настоящим линуксоидам, интересно самостоятельно смонтировать флеш-диск. Оказывается, тут все просто: USB-диск — это обычный накопитель, и его можно увидеть в каталоге /dev/disk/by-id. Напомню, что способ «by-id» подразумевает получение длинного имени по аппаратному идентификатору устройства, а поэтому с помощью каталога /dev/disk/by-id проще всего найти длинное имя USB-диска среди имен других накопителей, — оно будет начинаться с префикса usb. Введите команду:

```
ls -l /dev/disk/by-id | grep usb
```

Результат выполнения этой команды представлен на рис. 4.9.

Судя по выводу этой команды, для монтирования флеш-диска следует выполнить команду:

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

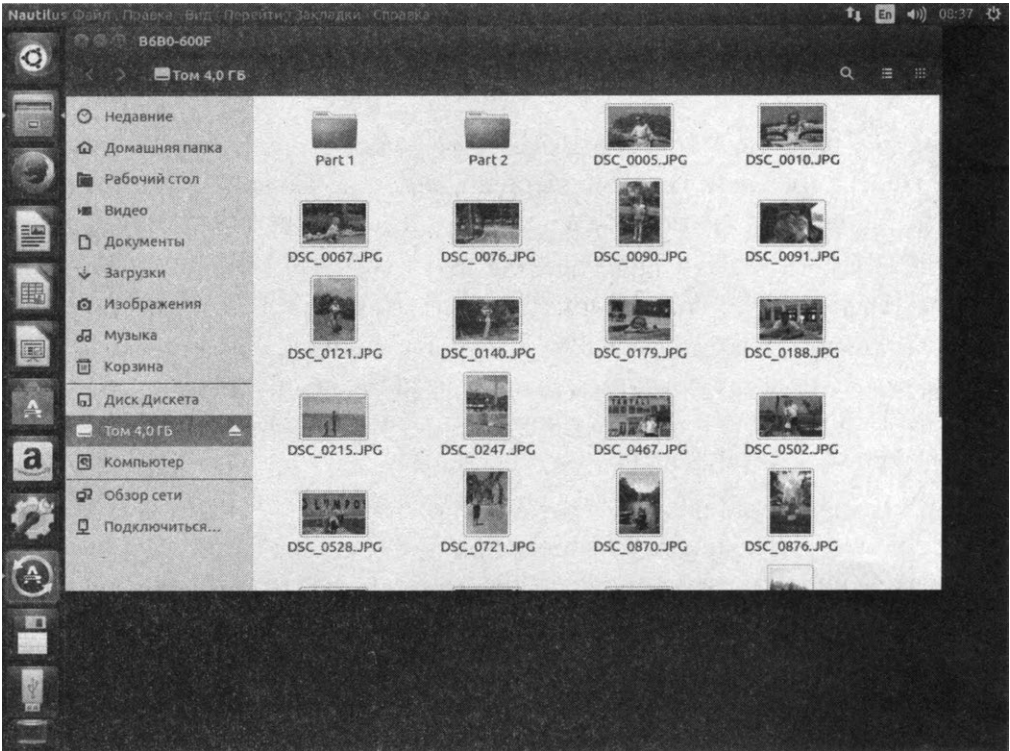


Рис. 4.8. Ubuntu 17.04: содержимое подключенной флешки с файлами

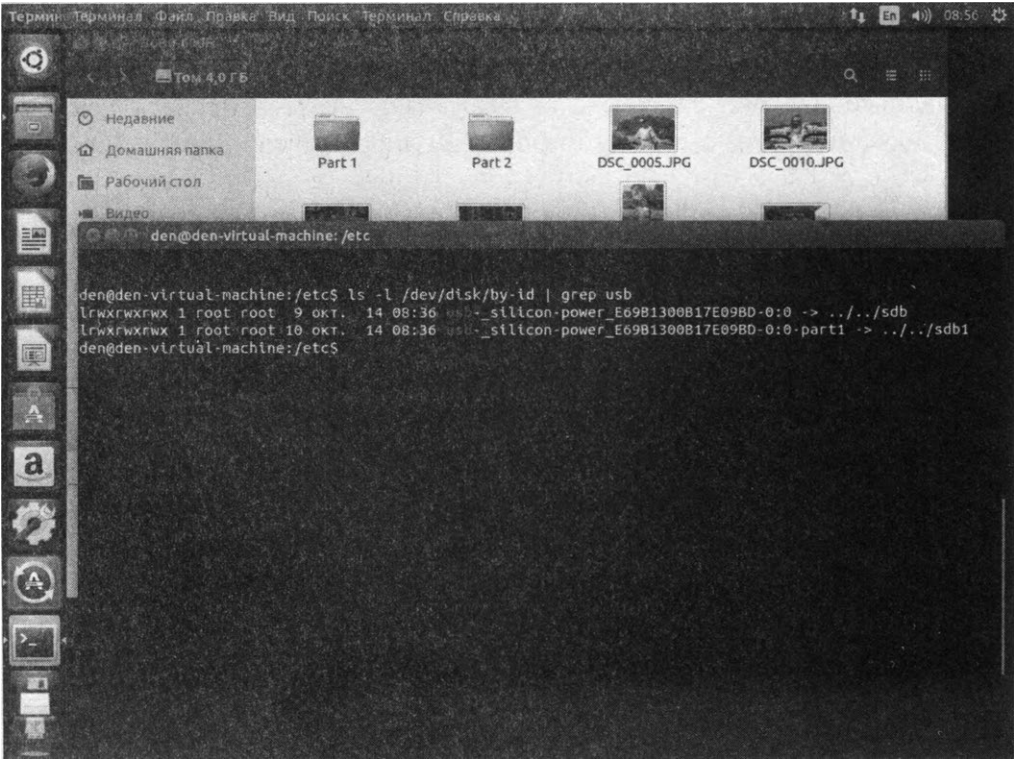


Рис. 4.9. Ubuntu 17.04: USB-диск найден

4.8. Настройка журнала файловой системы ext3/ext4

Журналируемая файловая система имеет три режима работы: `journal`, `ordered` и `writeback`. Первый режим самый медленный, но он позволяет минимизировать потери ваших данных в случае сбоя системы или отключения питания. В режиме `journal` в системный журнал записывается все, что только можно, и это позволяет максимально восстановить файловую систему в случае сбоя.

В последовательном режиме (`ordered`) в журнал заносится информация только об изменении метаданных (служебных данных файловой системы). Этот режим используется по умолчанию и является компромиссным вариантом между производительностью и отказоустойчивостью.

Самым быстрым является режим обратной записи (`writeback`). Но использовать его я вам не рекомендую, поскольку особого толку от него не будет. Проще тогда уже при установке Linux выбрать файловую систему `ext2` вместо `ext3/ext4`.

Если отказоустойчивость для вас на первом месте — выбирайте режим `journal`, во всех остальных случаях лучше выбрать `ordered`. Выбор режима осуществляется редактированием файла `/etc/fstab`. Например,

```
# режим ordered используется по умолчанию,  
# поэтому ничего указывать не нужно  
/dev/sda1 / ext3 defaults 1 0  
# на этом разделе важные данные, используем режим journal  
/dev/sda2 /var ext3 data=journal 1 0  
# здесь ничего важного нет, режим writeback  
/dev/sda2 /opt ext3 data=writeback 0 0
```

После изменения этого файла выполните команду:

```
# mount -a
```

Она заново смонтирует все файловые системы, чтобы изменения вступили в силу.

4.9. Файловая система ext4

Файловая система `ext4` заслуживает отдельного разговора. Все, что было сказано о файловых системах ранее, справедливо и для `ext4`, но у этой файловой системы есть ряд особенностей, о которых мы сейчас и поговорим.

Поддержка `ext4` как стабильной файловой системы появилась в ядре Linux версии 2.6.28. Если сравнивать эту файловую систему с `ext3`, то производительность и надежность в `ext4` существенно увеличена, а максимальный размер раздела доведен до 1024 петабайт (1 эксбибайт). Максимальный размер файла — более 2 Тбайт. Ресурс Phoronix (www.phoronix.com) произвел тестирование файловой системы `ext4` на SSD-накопителе (такие накопители устанавливаются на современные ноутбуки) —

результат, как говорится, налицо: ext4 почти в два раза превзошла файловые системы ext3, XFS, JFS и ReiserFS.

Впрочем, когда я установил Fedora 11 (первая версия Fedora, в которой использовалась ext4 по умолчанию) на рабочую станцию, прироста производительности при работе с файлами мне почувствовать не удалось. Однако производительность — это не основной конек ext4. Но, обо всем по порядку.

4.9.1. Сравнение ext3 и ext4

Описание особенностей файловой системы ext4 и ее преимуществ по сравнению с ext3 сведены в табл. 4.6.

Таблица 4.6. Особенности ext4

Особенность	Комментарий
Увеличенный размер файла и файловой системы	<p>Для ext3 максимальный размер файловой системы составляет 32 Тбайт, а файла — 2 Тбайт, но на практике ограничения были более жесткими. Так, в зависимости от архитектуры, максимальный размер тома составлял до 2 Тбайт, а максимальный размер файла — до 16 Гбайт.</p> <p>В случае с ext4 максимальный размер тома составляет 1 эксбибайт (EiB) — это 2^{60} байт. Максимальный размер файла составляет 16 Тбайт. Такие объемы информации пока не нужны обычным пользователям, однако весьма пригодятся на серверах, работающих с большими дисковыми массивами</p>
Экстенты	<p>Основной недостаток ext3 — ее метод выделения места на диске. Дисковые ресурсы выделялись с помощью битовых карт свободного места, а такой способ не отличается ни скоростью, ни масштабируемостью. Получилось, что ext3 более эффективна для небольших файлов, но совсем не подходит для хранения больших файлов.</p> <p>Для улучшения выделения ресурсов и более эффективной организации данных в ext4 были введены <i>экстенты</i>. Экстент — это способ представления непрерывной последовательности блоков памяти.</p> <p>Для эффективного представления маленьких файлов в экстентах применяется уровневый подход, а для больших файлов используются деревья экстенгов. Например, один индексный дескриптор может ссылаться на четыре экстента, каждый из которых может ссылаться на другие индексные дескрипторы и т. д. Такая структура является мощным механизмом представления больших файлов, а также более защищена и устойчива к сбоям</p>
Отложенное выделение пространства	Файловая система ext4 может отложить выделение дискового пространства до последнего момента, что увеличивает производительность системы
Контрольные суммы журналов	Контрольные суммы журналов повышают надежность файловой системы
Большее количество каталогов	В ext3 могло быть максимум 32 000 каталогов, в ext4 количество каталогов не ограничивается
Дефрагментация «на лету»	Файловая система ext3 не особо склонна к фрагментации, но все же такое неприятное явление имеется. В ext4 производится дефрагментация «на лету», что позволяет повысить производительность системы в целом

Таблица 4.6 (окончание)

Особенность	Комментарий
Наносекундные временные метки	В большинстве файловых систем временные метки (timestamp) устанавливаются с точностью до секунды, в ext4 точность повышена до наносекунды. Также ext4 поддерживает временные метки до 25 апреля 2514 года, в отличие от ext3 (только до 18 января 2038 г.)

4.9.2. Совместимость с ext3

Файловая система ext4 является прямо и обратно совместимой с ext3, однако все же существуют некоторые ограничения. Предположим, что у нас на диске имеется файловая система ext4. Ее можно смонтировать и как ext3, и как ext4 (это и есть прямая совместимость)— и тут ограничений никаких нет. А вот с обратной совместимостью не все так безоблачно — если файловую систему ext4 смонтировать как ext3, то она будет работать без экстентов, что снизит ее производительность.

4.9.3. Переход на ext4

Если вы при установке системы выбрали файловую систему ext3, то перейти на ext4 можно без потери данных и в любой удобный для вас момент. Откройте терминал и введите команду:

```
sudo tune2fs -O extents,uninit_bg,dir_index /dev/имя_устройства
```

На момент ввода этой команды устройство должно быть размонтировано.

ПРЕОБРАЗОВАНИЕ КОРНЕВОЙ ФАЙЛОВОЙ СИСТЕМЫ

Если нужно преобразовать в ext4 корневую файловую систему, то эту команду нужно вводить с LiveCD, поддерживающего ext4.

Теперь проверим файловую систему:

```
sudo fsck -pf /dev/имя_устройства
```

Затем смонтируем файловую систему так:

```
mount -t ext4 /dev/имя_устройства /точка_монтирования
mount -t ext4 /dev/disk/by-uuid/UUID-устройства /точка_монтирования
```

Если раздел автоматически монтируется через /etc/fstab, не забудьте исправить файловую систему на ext4:

```
UUID=UUID-раздела /точка ext4 defaults,errors=remount-ro,relatime 0 1
```

Если вы изменили тип файловой системы корневого раздела, тогда необходимо отредактировать файл /boot/grub/menu.lst и добавить опцию rootfstype=ext4 в список параметров ядра, например:

```
title      Linux
root      (hd0,1)
```

```
kernel /boot/vmlinuz-2.6.28.1 root=UUID=879f797c-944d-4c28-a720-249730705714 ro
quiet splash rootfstype=ext4
initrd /boot/initrd.img-2.6.28.1
quiet
```

СОВЕТ

Рекомендую прочитать статью Тима Джонса «Анатомия ext4»:
<http://www.ibm.com/developerworks/ru/library/l-anatomy-ext4/index.html>.

4.10. Использование программы fdisk для разметки диска

Для разметки диска мы воспользуемся стандартной программой fdisk, которая имеется во всех дистрибутивах Linux.

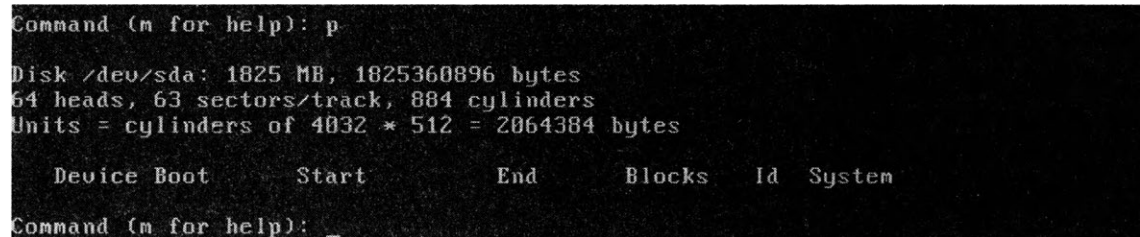
Введите команду (можно использовать короткие имена):

```
# fdisk <имя_устройства>
```

Например, если вы подключили винчестер как вторичный мастер, то команда будет следующей:

```
# fdisk /dev/sda
```

Чтобы убедиться, что диск не размечен, введите команду `p` — программа выведет пустую таблицу разделов (рис. 4.10).

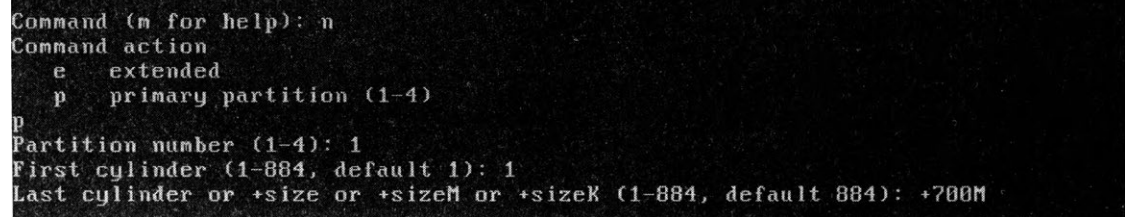


```
Command (m for help): p
Disk /dev/sda: 1825 MB, 1825360896 bytes
64 heads, 63 sectors/track, 884 cylinders
Units = cylinders of 4032 * 512 = 2064384 bytes

   Device Boot      Start         End      Blocks   Id  System
Command (m for help): _
```

Рис. 4.10. Таблица разделов пуста

Самое время создать раздел. Для этого служит команда `n` (рис. 4.11). Кстати, для справки можете ввести команду `h`, которая выведет список доступных команд fdisk (рис. 4.12).



```
Command (m for help): n
Command action
  e   extended
  p   primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (1-884, default 1): 1
Last cylinder or +size or +sizeM or +sizeK (1-884, default 884): +700M
```

Рис. 4.11. Создание нового раздела

```

64 heads, 63 sectors/track, 884 cylinders
Units = cylinders of 4032 * 512 = 2064384 bytes

  Device Boot      Start          End      Blocks   Id  System
Command (m for help): m
'Command action
a  toggle a bootable flag
b  edit bsd disklabel
c  toggle the dos compatibility flag
d  delete a partition
l  list known partition types
m  print this menu
n  add a new partition
o  create a new empty DOS partition table
p  print the partition table
q  quit without saving changes
s  create a new empty Sun disklabel
t  change a partition's system id
u  change display/entry units
v  verify the partition table
w  write table to disk and exit
x  extra functionality (experts only)
Command (m for help): _

```

Рис. 4.12. Список команд программы fdisk

После ввода команды `n` программа попросит вас уточнить, какого типа должен быть раздел (см. рис. 4.11), — можно выбрать первичный или расширенный раздел. В нашем случае больше подойдет первичный, поэтому вводим букву `p`. Затем нужно ввести номер раздела. Поскольку это первый раздел, то вводим `1`. Потом `fdisk` попросит ввести номер первого цилиндра. Это первый раздел, поэтому вводим номер `1`. После ввода первого цилиндра нужно ввести номер последнего цилиндра. Чтобы не высчитывать на калькуляторе номер цилиндра, намного проще ввести размер раздела. Делается это так: `+<размер>М`. После числа должна идти именно буква `м`, иначе размер будет воспринят в байтах, а нам нужны мегабайты, — например, если вы хотите создать раздел размером 10 Гбайт, то введите `+10240М`.

Для создания второго раздела опять введите команду `n`. Программа вновь попросит тип раздела, номер первого цилиндра (это будет номер последнего цилиндра первого раздела плюс 1) и размер раздела. Если вы хотите создать раздел до «конца» диска, то просто введите номер последнего цилиндра.

Теперь посмотрим на таблицу разделов. Для этого опять введите команду `p` (рис. 4.13).

```

Command (m for help): p


Disk /dev/sda: 1825 MB, 1825360896 bytes
64 heads, 63 sectors/track, 884 cylinders
Units = cylinders of 4032 * 512 = 2064384 bytes

  Device Boot      Start          End      Blocks   Id  System
/dev/sda1          1           340       685408+   83   Linux
/dev/sda2          341          884       1096704   83   Linux
Command (m for help): _

```

Рис. 4.13. Создание второго раздела, вывод таблицы разделов

По умолчанию программа `fdisk` создает Linux-разделы. Если вы собираетесь работать только в Linux, можно оставить и так, но ведь не у всех есть Linux, — если вы снимете свой винчестер, чтобы, например, переписать у товарища большие файлы, то вряд ли сможете комфортно с ним работать в его Windows: прочитать данные (например, с помощью `Total Commander`) вам удастся, а что-либо записать — уже нет. Поэтому давайте изменим тип разделов. Для этого служит команда `t`. Введите эту команду. Программа запросит у вас номер раздела и тип файловой системы. С номером раздела все ясно, а вот с кодом файловой системы сложнее. Введите `L`, чтобы просмотреть доступные файловые системы (рис. 4.14).



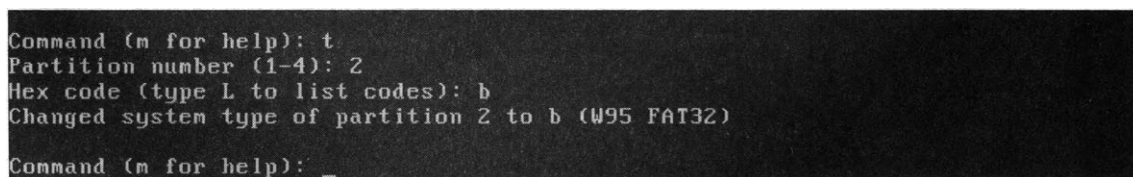
0	Empty	1e	Hidden W95 FAT1	80	Old Minix	be	Solaris boot
1	FAT12	24	NEC DOS	81	Minix / old Lin	bf	Solaris
2	XENIX root	39	Plan 9	82	Linux swap / So	c1	DRDOS/sec (FAT-
3	XENIX usr	3c	PartitionMagic	83	Linux	c4	DRDOS/sec (FAT-
4	FAT16 <32M	40	Unix B0286	84	OS/2 hidden C:	c6	DRDOS/sec (FAT-
5	Extended	41	PPC PReP Boot	85	Linux extended	c7	Syrinx
6	FAT16	42	SPS	86	NTFS volume set	da	Non-FS data
7	HPFS/NTFS	4d	QNX4.x	87	NTFS volume set	db	CP/M / CTOS /
8	AIX	4e	QNX4.x 2nd part	88	Linux plaintext	de	Dell Utility
9	AIX bootable	4f	QNX4.x 3rd part	8e	Linux LVM	df	BootIt
a	OS/2 Boot Manag	50	OnTrack DM	93	Amoeba	e1	DOS access
b	W95 FAT32	51	OnTrack DM6 Aux	94	Amoeba BBT	e3	DOS R/O
c	W95 FAT32 (LBA)	52	CP/M	9f	BSD/OS	e4	SpeedStor
e	W95 FAT16 (LBA)	53	OnTrack DM6 Aux	a0	IBM Thinkpad hi	eb	BeOS fs
f	W95 Ext'd (LBA)	54	OnTrackDM6	a5	FreeBSD	ee	EFI GPT
10	OPUS	55	EZ-Drive	a6	OpenBSD	ef	EFI (FAT-12/16/
11	Hidden FAT12	56	Golden Bow	a7	NeXTSTEP	f0	Linux/PA-RISC b
12	Compaq diagnost	5c	Priam Edisk	a8	Darwin UFS	f1	SpeedStor
14	Hidden FAT16 <3	61	SpeedStor	a9	NetBSD	f4	SpeedStor
16	Hidden FAT16	63	GNU HURD or Sys	ab	Darwin boot	f2	DOS secondary
17	Hidden HPFS/NTF	64	Novell Netware	b7	BSDI fs	fd	Linux raid auto
18	AST SmartSleep	65	Novell Netware	b8	BSDI swap	fe	LANstep
1b	Hidden W95 FAT3	70	DiskSecure Mult	bb	Boot Wizard hid	ff	BBT
1c	Hidden W95 FAT3	75	PC/IX				

Hex code (type L to list codes): _

Рис. 4.14. Коды файловых систем

Код FAT32 — `b`. Введите его, и вы увидите сообщение программы, что тип файловой системы изменен (рис. 4.15).

Еще раз введите команду `p`, чтобы убедиться, что все нормально. Для сохранения таблицы разделов введите `w`, а для выхода без сохранения изменений — `q`.



```

Command (m for help): t
Partition number (1-4): 2
Hex code (type L to list codes): b
Changed system type of partition 2 to b (W95 FAT32)
Command (m for help): _

```

Рис. 4.15. Тип файловой системы изменен

Из графических программ для разметки диска мне нравится только `GParted` (рис. 4.16). Остальные программы не заслуживают внимания — уж лучше использовать текстовые программы `fdisk` или `cfdisk`.

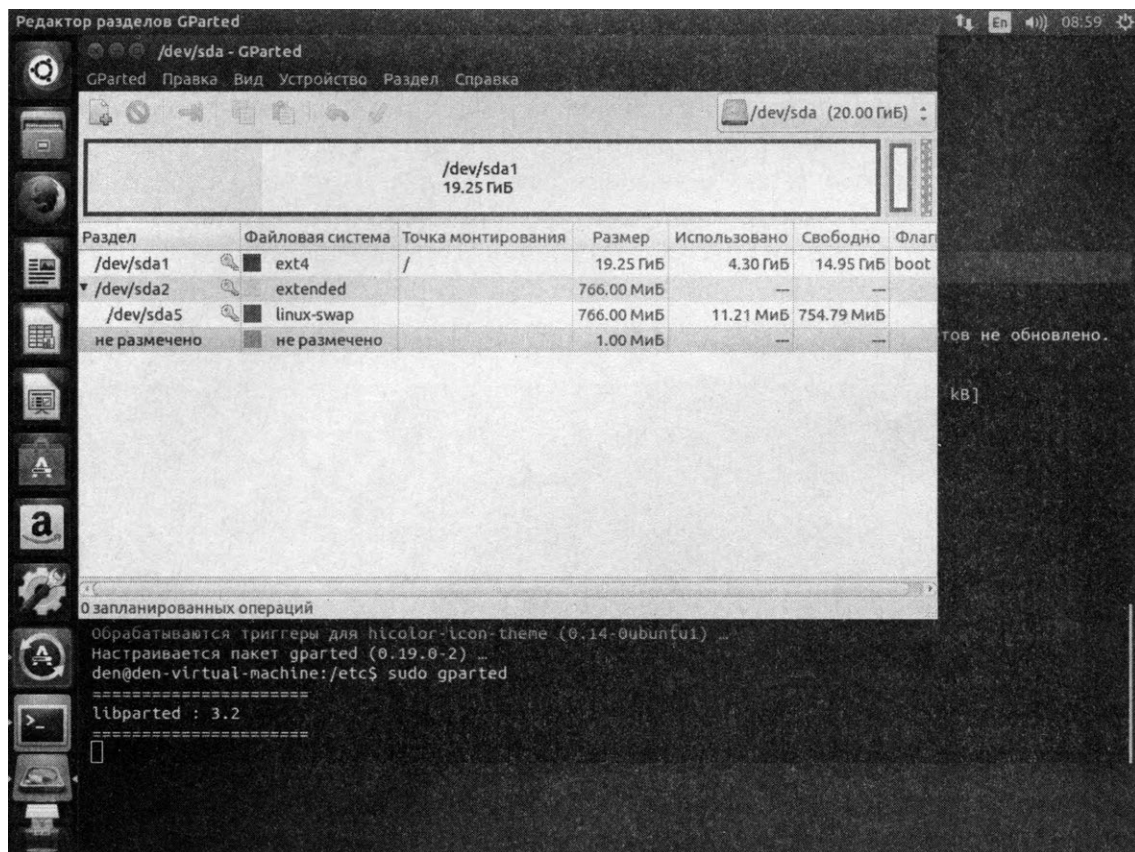


Рис. 4.16. Ubuntu: программа GParted

4.11. Таблица разделов GPT

GUID Partition Table (GPT) — стандартный формат размещения таблиц разделов на физическом жестком диске. GPT является частью EFI (Extensible Firmware Interface, расширяемый микропрограммный интерфейс) — стандарта, который был предложен компанией Intel на смену BIOS. В EFI таблица GPT используется там, где в BIOS используется MBR (Master Boot Record, главная загрузочная запись).

В отличие от MBR, начинающейся с исполняемой двоичной программы-загрузчика, которая должна идентифицировать и загрузить активный раздел, GPT использует для осуществления этих процессов EFI. Но MBR все же присутствует в самом начале диска для обратной совместимости и для защиты, GPT же начинается с оглавления таблицы разделов.

В GPT используется современная система адресации логических блоков (LBA) вместо устаревшей системы CHS (Цилиндр-Головка-Сектор), которая применялась в MBR. Как и MBR, таблица GPT обеспечивает дублирование — оглавление и таблица разделов записаны как в начале, так и в конце диска.

С помощью GPT можно создавать разделы размером до 9,4 зеттабайт ($9,4 \times 10^{21}$ байт), в MBR же максимальный размер диска 2,2 терабайта ($2,2 \times 10^{12}$ байт).

Для работы с разделами GPT нужно использовать утилиты `gdisk` или `gpart`, поскольку при просмотре содержимого диска GPT программой `fdisk` картина будет примерно такой:

```
WARNING: GPT (GUID Partition Table) detected on '/dev/sdb'! The util fdisk doesn't support GPT. Use GNU Parted.
```

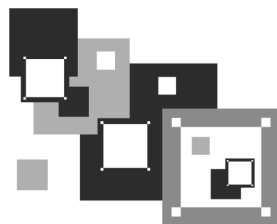
```
Disk /dev/sdb: 1000.2 GB, 1000204886016 bytes
255 heads, 63 sectors/track, 121601 cylinders, total 1953525168 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal) : 512 bytes / 512 bytes
Disk identifier: 0xcd29a27d
```

Device	Boot	Start	End	Blocks	Id	System
/dev/sdb1		1	1953525167	976762583+	ee	GPT

ВОССТАНОВЛЕНИЕ ИНФОРМАЦИИ С GPT

К сожалению, подробное рассмотрение GPT выходит за рамки этой книги, но, понимая важность темы, привожу ссылку на очень полезную статью о восстановлении информации с GPT (да, новые терабайтные жесткие диски тоже «сыплются») программой `gpart`: <http://bu7cher.blogspot.com/2010/10/gpt-gpart.html>.

ГЛАВА 5



Командный интерпретатор bash

5.1. bash: основные сведения

bash — это наиболее часто используемая командная оболочка (командный интерпретатор) Linux. Основное предназначение bash — выполнение команд, введенных пользователем. Пользователь вводит команду, bash ищет программу, соответствующую команде, в каталогах, указанных в переменной окружения PATH. Если такая программа найдена, то bash запускает ее и передает ей введенные пользователем параметры. В противном случае выводится сообщение о невозможности выполнения команды.

Кроме bash в Linux существуют и другие оболочки: sh, csh, ksh, zsh и пр. Все командные оболочки, установленные в системе, прописаны в файле /etc/shells. В листинге 5.1 представлен файл /etc/shells дистрибутива Fedora 26 с установками по умолчанию. В этом дистрибутиве, по сравнению с Fedora 16, набор оболочек существенно сокращен и оставлены только sh и bash (nologin — это не оболочка, а просто программа, отображающая сообщение, что учетная запись недоступна, — в случае обращения к отключенной или системной учетной записи). По сути, в Fedora 26 присутствует одна командная оболочка, потому что sh и bash — это практически одно и то же.

Листинг 5.1. Файл /etc/shells дистрибутива Fedora 26

```
/bin/sh
/bin/bash
/sbin/nologin
/usr/bin/sh
/usr/bin/bash
/usr/sbin/nologin
```

С точки зрения пользователя все прописанные в файле /etc/shells оболочки мало чем друг от друга отличаются, поскольку все они позволяют выполнять введенные пользователем команды. Но оболочки служат не только для выполнения команд, а еще и для автоматизации задач с помощью *сценариев*, и основное их различие за-

ключается в синтаксисе языка описания сценариев. В этой главе мы поговорим о создании *bash*-сценариев, поскольку оболочка *bash* самая популярная.

ПРОГРАММЫ-«ЗАГЛУШКИ»

Иногда в файле `/etc/shells` кроме программы `nologin` можно найти еще и программы `/bin/false` и `/bin/true`, которые тоже не являются оболочками. Это «заглушки», которые можно использовать, если вы хотите отключить ту или иную учетную запись пользователя. Как известно, при входе пользователя в систему запускается установленная для него оболочка. И для каждого пользователя имеется возможность задать свою оболочку. Так вот, если для какого-либо пользователя задать оболочку `/bin/false` (или `/bin/true`), он не сможет войти в систему. Точнее, в систему-то он войдет, но и сразу выйдет из нее, поскольку сессия пользователя длится до завершения работы его оболочки, а обе «заглушки» ничего не делают, кроме того, что просто возвращают значение 0 (для `false`) или 1 (для `true`). В *главе 28* мы рассмотрим, как можно обезопасить сервер с использованием «заглушек».

При запуске оболочка *bash* выполняет сценарий `.bashrc`, обычно расположенный в домашнем каталоге пользователя (этот файл не обязателен и может там отсутствовать). В файле этого сценария можно указать команды, которые нужно выполнить сразу после входа пользователя в систему.

В файле `.bash_history` (он тоже находится в домашнем каталоге) хранится история команд, введенных пользователем, — открыв его, вы можете просмотреть свои команды, которые накануне вводили.

5.2. Автоматизация задач с помощью *bash*

Представим, что нам нужно выполнить резервное копирование всех важных файлов, для чего создать архивы каталогов `/etc`, `/home` и `/usr`. Понятно, что понадобятся три команды вида:

```
tar -cvjf имя_архива.tar.bz2 каталог
```

Затем надо будет записать все эти три файла на DVD с помощью любой программы для прожига дисков.

Ничего страшного, если выполнять эту операцию раз в месяц или хотя бы раз в неделю. Но представьте, что вам нужно делать ее каждый день или даже несколько раз в день? Думаю, такая рутинная работа вам быстро надоест. А ведь можно написать *сценарий*, который сам будет создавать резервные копии и записывать их на DVD! И все, что для этого потребуется, — это вставить чистый DVD перед запуском сценария.

Можно пойти и иным путем. Написать сценарий, который будет делать резервные копии системных каталогов и записывать их на другой раздел жесткого диска. Ведь не секрет, что резервные копии делаются не только на случай сбоя системы, но и для защиты от некорректного изменения данных пользователем. Помню, удалив как-то важную тему своего форума, я попросил хостинг-провайдера сделать откат. И был приятно удивлен, когда мне предоставили на выбор три резервные копии, — осталось лишь выбрать наиболее подходящую. Не думаете же вы, что администра-

торы провайдера только и занимались тем, что три раза в день копировали домашние каталоги пользователей? Поэтому автоматизация — штука полезная, и любому администратору нужно знать, как автоматизировать свою рутинную работу.

5.3. Привет, мир!

Итак, напишем наш первый сценарий, по традиции выводящий всем известную фразу: «Привет, мир!» (листинг 5.2). Вся работа со сценариями выполняется обычно в консоли (или в терминале), но для редактирования сценариев вы можете использовать любимый графический редактор, — например, тот же `mcedit`.

Листинг 5.2. Первый сценарий

```
#!/bin/bash
echo "Привет, мир!"
```

Первая строка нашего сценария — это указание, что он должен быть обработан программой `/bin/bash`. Обратите внимание — если между символами `#` и `!` окажется пробел, то эта директива не сработает, поскольку будет воспринята как обычный комментарий, который, как вы уже догадались, начинается с решетки:

```
# Комментарий
```

Вторая строка — это оператор `echo`, выводящий нашу строку.

Теперь сохраните сценарий под именем `hello` и введите следующую команду (она сделает наш сценарий исполнимым):

```
$ chmod +x hello
```

Для запуска сценария введите команду:

```
./hello
```

и на экране вы увидите строку:

Привет, мир!

Чтобы вводить для запуска сценария просто `hello` (без `./`), сценарий нужно скопировать в каталог `/usr/bin` (точнее, в любой каталог из переменной окружения `PATH`):

```
# cp ./hello /usr/bin
```

5.4. Использование переменных в собственных сценариях

В любом серьезном сценарии вы не обойдетесь без использования *переменных*. Переменные можно объявлять в любом месте сценария, но до места их первого применения. Рекомендуется объявлять переменные в самом начале сценария, чтобы потом не искать, где вы объявили ту или иную переменную.

Для объявления переменной используется следующая конструкция:

```
переменная=значение
```

Пример объявления переменной:

```
ADDRESS=www.dkws.org.ua  
echo $ADDRESS
```

Обратите внимание на следующие моменты:

- при объявлении переменной знак доллара не ставится, но он обязателен при использовании переменной;
- при объявлении переменной не должно быть пробелов до и после знака `=`.

Значение для переменной указывать вручную не обязательно — его можно прочесть с клавиатуры:

```
read ADDRESS
```

или со стандартного вывода программы:

```
ADDRESS=$(hostname)
```

Как можно видеть, чтение значения переменной с клавиатуры осуществляется с помощью инструкции `read` (при этом указывать символ доллара не нужно). Вторая приведенная здесь команда устанавливает в качестве значения переменной `ADDRESS` вывод команды `hostname`.

В Linux часто используются *переменные окружения*. Это специальные переменные, содержащие служебные данные. Вот примеры некоторых часто используемых переменных окружения:

- `HOME` — домашний каталог пользователя, который запустил сценарий;
- `RANDOM` — случайное число в диапазоне от 0 до 32 767;
- `UID` — ID пользователя, который запустил сценарий;
- `PWD` — текущий каталог.

Для установки собственной переменной окружения используется команда `export`:

```
# присваиваем переменной значение  
$ADDRESS=www.dkws.org.ua  
# экспортируем переменную — делаем ее переменной окружения  
# после этого переменная ADDRESS будет доступна в других сценариях  
export $ADDRESS
```

5.5. Передача параметров сценарию

Очень часто сценариям нужно передавать различные параметры — например, для задания режима работы или указания имени файла/каталога. Для передачи параметров используются следующие специальные переменные:

- \$0 — содержит имя сценария;
- \$n — содержит значение параметра (n — номер параметра);
- \$# — позволяет узнать количество параметров, которые были переданы.

Рассмотрим небольшой пример обработки параметров сценария. Я понимаю, что конструкцию `case-esac` мы еще не рассматривали, но общий принцип должен быть понятен (листинг 5.3).

Листинг 5.3. Пример обработки параметров сценария

```
# сценарий должен вызываться так:
# имя_сценария параметр
# анализируем первый параметр
case "$1" in
    start)
        # действия при получении параметра start
        echo "Запускаем сетевой сервис"
        /2
    stop)
        # действия при получении параметра stop
        echo "Останавливаем сетевой сервис"
        ;;
    *)
        # действия в остальных случаях
        # выводим подсказку о том, как нужно использовать сценарий,
        # и завершаем работу сценария
        echo "Usage: $0 {start|stop}"
        exit 1
        ;;
esac
```

Полагаю, приведенных здесь комментариев достаточно, поэтому подробно рассматривать работу сценария из листинга 5.3 мы не станем.

5.6. Массивы

Интерпретатор `bash` позволяет использовать *массивы*. Массивы объявляются подобно переменным. Вот пример объявления массива:

```
ARRAY[0]=1
ARRAY[1]=2

echo $ARRAY[0]
```

5.7. Циклы

Как и в любом языке программирования, в *bash* можно использовать *циклы*. Мы рассмотрим циклы *for* и *while*, хотя в *bash* доступны также циклы *until* и *select*, но они применяются довольно редко.

Синтаксис цикла *for* выглядит так:

```
for переменная in список
do
команды
done
```

В цикле при каждой итерации переменной будет присвоен очередной элемент списка, над которым будут выполнены указанные команды. Чтобы было понятнее, рассмотрим небольшой пример:

```
for n in 1 2 3;
do
echo $n;
done
```

Обратите внимание — список значений и список команд должны заканчиваться точкой с запятой.

Как и следовало ожидать, наш сценарий выведет на экран следующее:

```
1
2
3
```

Синтаксис цикла *while* выглядит немного иначе:

```
while условие
do
команды
done
```

Цикл *while* выполняется до тех пор, пока истинно заданное условие. Подробно об условиях мы поговорим в следующем разделе, а сейчас напомним аналог предыдущего цикла, т. е. выведем 1, 2 и 3, но с помощью *while*, а не *for*:

```
n=1
while [ $n -lt 4 ]
do
echo "$n "
n=$(( $n+1 ));
done
```

5.8. Условные операторы

В `bash` доступны два *условных оператора*: `if` и `case`. Синтаксис оператора `if` следующий:

```
if условие_1 then
    команды_1
elif условие_2 then
    команды_2

elif условие_N then
    команды_N
else
    команды_N+1
fi
```

Оператор `if` в `bash` работает аналогично оператору `if` в других языках программирования. Если истинно первое условие, то выполняется первый список команд, иначе — проверяется второе условие и т. д. Количество блоков `elif`, понятно, не ограничено.

Самая ответственная задача — это правильно составить условие. Условия записываются в квадратных скобках. Вот пример записи условий:

```
# переменная N = 10
[ N==10 ]
```

```
# переменная N не равна 10
[ N!=10 ]
```

Операции сравнения указываются не с помощью привычных знаков больше (`>`) или меньше (`<`), а с помощью следующих выражений:

- `-lt` — меньше;
- `-gt` — больше;
- `-le` — меньше или равно;
- `-ge` — больше или равно;
- `-eq` — равно (используется вместо `=`).

Применять эти выражения нужно следующим образом:

```
[ переменная выражение значение | переменная ]
```

Например:

```
# N меньше 10
[ $N -lt 10 ]
# N меньше A
[ $N -lt $A ]
```

В квадратных скобках вы также можете задать выражения для проверки существования файла и каталога:

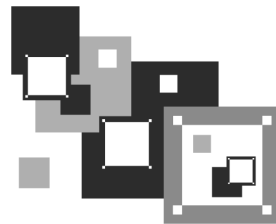
- `-e` файл — условие истинно, если файл существует;
- `-d` каталог — условие истинно, если каталог существует;
- `-x` файл — условие истинно, если файл является исполнимым.

С оператором `case` мы уже немного знакомы, но сейчас рассмотрим его синтаксис подробнее:

```
case переменная in
значение_1) команды_1 ;;
. . .
значение_N) команды_N ;;
*) команды_по_умолчанию; ;
esac
```

Значение указанной переменной по очереди сравнивается с приведенными значениями (`значение_1`, `значение_N`). Если есть совпадение, то будут выполнены команды, соответствующие значению. Если совпадений нет, то будут выполнены команды по умолчанию. Пример использования `case` был приведен в листинге 5.3.

ГЛАВА 6



Пользователи и группы

6.1. Многопользовательская система

Linux, как и ее прародительница UNIX, является многозадачной многопользовательской операционной системой. Это означает, что в один момент с системой могут работать несколько пользователей, и каждый пользователь может запустить несколько приложений. При этом вы можете зайти в систему локально, а кто-то — удаленно, используя один из протоколов удаленного доступа (telnet, ssh) или по FTP. Согласитесь, очень удобно. Предположим, что вы забыли распечатать очень важный документ, а возвращаться домой уже нет времени. Если ваш компьютер должным образом настроен и подключен к Интернету, вы можете получить к нему доступ (даже если компьютер выключен, достаточно позвонить домой и попросить кого-нибудь включить его, а к Интернету компьютер подключится автоматически), зайти в систему по ssh (или подключиться к графическому интерфейсу, если вы предпочитаете работать в графическом режиме) и скопировать нужный вам файл. И если кто-либо в момент вашего подключения уже работает с системой, вы не мешаете друг другу.

Вы можете обвинить меня в рекламе Linux: мол, эта возможность была еще в Windows 98, — если установить соответствующее программное обеспечение вроде Remote Administrator. Должен отметить, что в Windows все иначе. Да, Remote Administrator предоставляет удаленный доступ к рабочему столу, но если за компьютером уже работает пользователь, то вместе вы работать не сможете, — вы будете мешать ему, а он вам. Ведь все, что станете делать вы, будет видеть он, а все, что будет делать он, вы увидите у себя на экране, т. е. рабочий стол получится как бы общий. А если вы предварительно не предупредите пользователя о своем удаленном входе, он даже может подумать, что с системой что-то не то. Помню, со мной так и было: пользователь, работавший за компьютером, закрывал окна, которые я, работая в удаленном режиме, открывал на его компьютере. Пришлось мне самому подойти к рабочему месту того пользователя и попросить его не мешать.

По-настоящему многопользовательский режим возможен в серверных версиях с использованием протокола RDP, но это не всегда удобно. Во-первых, использование RDP не всегда разрешается корпоративными политиками безопасности. А во-вторых, часто для использования RDP нужно перенастраивать брандмауэр (в том

числе и всей организации, а не только брандмауэр конкретного компьютера, к которому будет выполнено RDP-подключение).

В Linux же все так, как и должно быть, — несколько пользователей могут работать с системой и даже не подозревать о существовании друг друга, пока не введут соответствующую команду (`who`), показывающую, кто в это время работает в системе.

6.2. Пользователь `root`

6.2.1. Полномочия пользователя `root`

Пользователь `root` обладает в системе максимальными полномочиями, и она полностью подвластна этому пользователю, — любая его команда будет выполнена безоговорочно. Поэтому работать под именем пользователя `root` нужно с осторожностью. Всегда думайте над тем, что собираетесь сделать, — если вы дадите команду на удаление корневой файловой системы, система выполнит и ее. В отличие от пользователя `root`, пользователю, зарегистрировавшемуся под обычным именем, при попытке выполнить ту или иную команду система сообщит, что у него нет для этого полномочий.

Представим, что кто-то решил пошутить и выложил в Интернете (записал на диск или прислал по электронной почте — способ доставки не важен) вредоносную программу. Если вы ее запустите от имени пользователя `root`, система может быть повреждена. Запуск этой же программы от имени обычного пользователя ничего страшного не произведет — система откажется ее выполнять. Впрочем, все может быть намного проще: вы сами ошибочно введете команду, способную разрушить систему, или отойдете ненадолго от своего компьютера, а появившийся тут как тут некий «доброжелатель», имея полномочия пользователя `root`, сможет уничтожить систему одной командой.

Именно поэтому практически во всех современных дистрибутивах вход под именем пользователя `root` запрещен. В одних дистрибутивах вы не сможете войти как `root` в графическом режиме (но можете войти в консоли, переключившись на первую консоль с помощью комбинации клавиш `<Ctrl>+<Alt>+<F1>`), а в других — вовсе не можете войти в систему как `root`: ни в графическом режиме, ни в консоли (например, в Ubuntu).

А если вы запускаете какую-либо графическую программу, требующую привилегий `root`, то увидите окно с требованием ввести соответствующий пароль (рис. 6.1).

Отсюда можно сделать следующие выводы:

- старайтесь реже работать пользователем `root`;
- всегда думайте, какие программы вы запускаете под именем `root`;
- если программа, полученная из постороннего источника, требует `root`-полномочий, это должно насторожить;
- создайте обычного пользователя (даже если вы сами являетесь единственным пользователем компьютера) и рутинные операции (с документами, использование Интернета и т. д.) производите от имени этого пользователя;

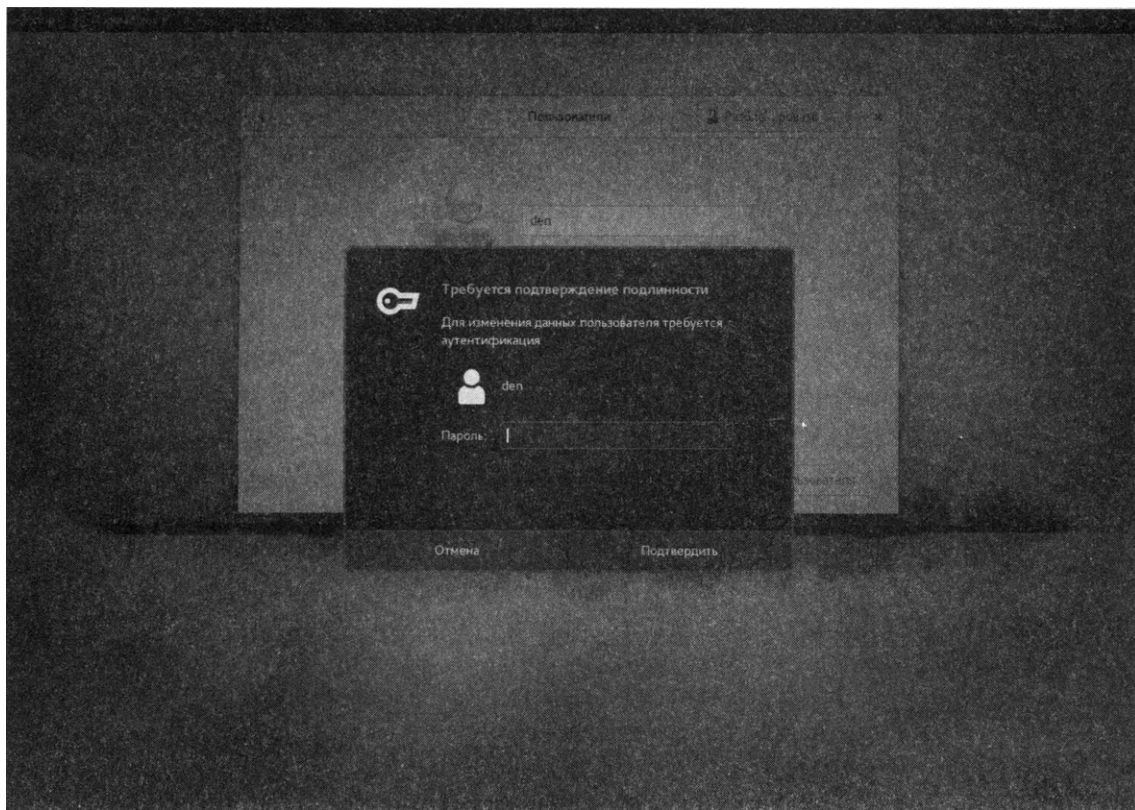


Рис. 6.1. Fedora 26: требование ввести пароль

- если полномочия `root` все же нужны, совсем необязательно заходить в систему под этим пользователем, — достаточно запустить терминал и выполнить команду `sudo` или `su` (см. *разд. 6.2.2*), после чего в этом терминале можно будет выполнять команды с правами `root`. А закрыв терминал, вы потеряете права `root`, что весьма удобно — ведь обычно такие права требуются для одной-двух операций (например, выполнить команду установки программы или создать/удалить пользователя).

6.2.2. Временное получение полномочий `root`

Некоторые операции, например установка программного обеспечения, изменение конфигурационных файлов и т. п., требуют полномочий `root`. Чтобы их временно получить, следует использовать команды `sudo` или `su` (эти команды, скорее всего, вы будете запускать в терминале).

Команда `sudo`

Команда `sudo` позволяет запустить любую команду с привилегиями `root`. Использовать ее нужно так:

```
sudo <команда_которую_нужно_выполнить_с_правами_root>
```

Например, вам необходимо изменить файл `/etc/apt/sources.list`. Для этого служит команда:

```
sudo gedit /etc/apt/sources.list
```

ТЕКСТОВЫЙ РЕДАКТОР GEDIT

Программа `gedit` — это текстовый редактор, ему мы передаем один параметр — имя файла, который нужно открыть.

Если ввести эту же команду, но без `sudo` (просто: `gedit /etc/apt/sources.list`), текстовый редактор тоже запустится и откроет файл, но сохранить изменения в нем вы не сможете, поскольку у вас не хватит полномочий.

Перед выполнением указанной вами команды команда запросит у вас пароль:

```
sudo gedit /etc/apt/sources.list
```

Password:

Вы должны ввести свой *пользовательский пароль* — тот, который применяете для входа в систему, но не пароль пользователя `root` (кстати, мы его и не знаем).

ФАЙЛ /ETC/SUDOERS

Использовать команду `sudo` имеют право не все пользователи, а только те, которые внесены в файл `/etc/sudoers`. Администратор системы (пользователь `root`) может редактировать этот файл с помощью команды `visudo`. Если на компьютере установлен дистрибутив, в котором запрещен вход под учетной записью `root` (следовательно, у вас нет возможности отредактировать файл `sudoers`), то в файле `sudoers` содержатся пользователи, которых вы добавили при установке системы.

Команда su

Команда `su` позволяет получить доступ к консоли с правами `root` любому пользователю (даже если пользователь не внесен в файл `/etc/sudoers`) при условии, что он знает пароль `root`. Понятно, что в большинстве случаев этим пользователем будет сам пользователь `root`, — не станете же вы всем пользователям доверять свой пароль? Поэтому команда `su` предназначена, в первую очередь, для администратора системы, а `sudo` — для остальных пользователей, которым иногда нужны права `root` (чтобы они меньше отвлекали администратора от своей работы).

Использовать команду `su` просто:

```
su
```

После этого нужно будет ввести пароль пользователя `root`, и вы сможете работать в консоли как обычно. Использовать `su` удобнее, чем `sudo`, потому что вам не потребуется вводить `su` перед каждой командой, которая должна быть выполнена с правами `root`.

Чтобы закрыть сессию `su`, нужно или ввести команду `exit`, или просто закрыть окно терминала.

Команды *gksudo* и *kdesu*

Если вы в терминале хотите запустить графическую программу с правами root (например, *gedit*), желательно использовать команду не *sudo*, а *gksudo* (т. е. *gksu* — для GNOME или *kdesu* — для KDE).

Проблемы с *sudo* в Ubuntu и Kubuntu

Программа *sudo* не всегда корректно работает с графическими приложениями, поэтому рано или поздно вы можете получить сообщение **Unable to read ICE authority file**, и после этого вообще станет невозможным запуск графических программ с правами root. Если это все же произошло, поправить ситуацию можно, удалив файл `./{ICE,X}authority` из вашего домашнего каталога:

```
rm ~/.{ICE,X}authority
```

Напомню, что тильда здесь означает домашний каталог текущего пользователя.

Графические приложения с правами root проще запускать, используя главное меню. Но не все приложения есть в главном меню, и не все приложения вызываются с правами root, — например, в главном меню есть команда вызова текстового редактора, но нет команды для вызова текстового редактора с правами root. Поэтому намного проще нажать клавиатурную комбинацию `<Alt>+<F2>` (она работает не только в Ubuntu, но и в других дистрибутивах) и ввести в соответствующее поле (рис. 6.2) команду:

```
gksu <команда>
```



Рис. 6.2. Ubuntu: быстрое выполнение программы

Ввод серии команд *sudo*

Вам надоело каждый раз вводить *sudo* в начале команд? Тогда выполните команду:

```
sudo -i
```

Эта команда запустит оболочку root, т. е. вы сможете вводить любые команды, и они будут выполнены с правами root. Обратите внимание, как изменился вид

приглашения командной строки (рис. 6.3): до этого приглашение имело вид `$` — это означало, что вы работаете от имени обычного пользователя, а после выполнения команды `sudo -i` приглашение изменилось на `#` — это верный признак того, что каждая введенная команда будет выполнена с правами `root`.

Опция `-i` позволяет так же удобно вводить команды, как если бы вы использовали команду `su`.

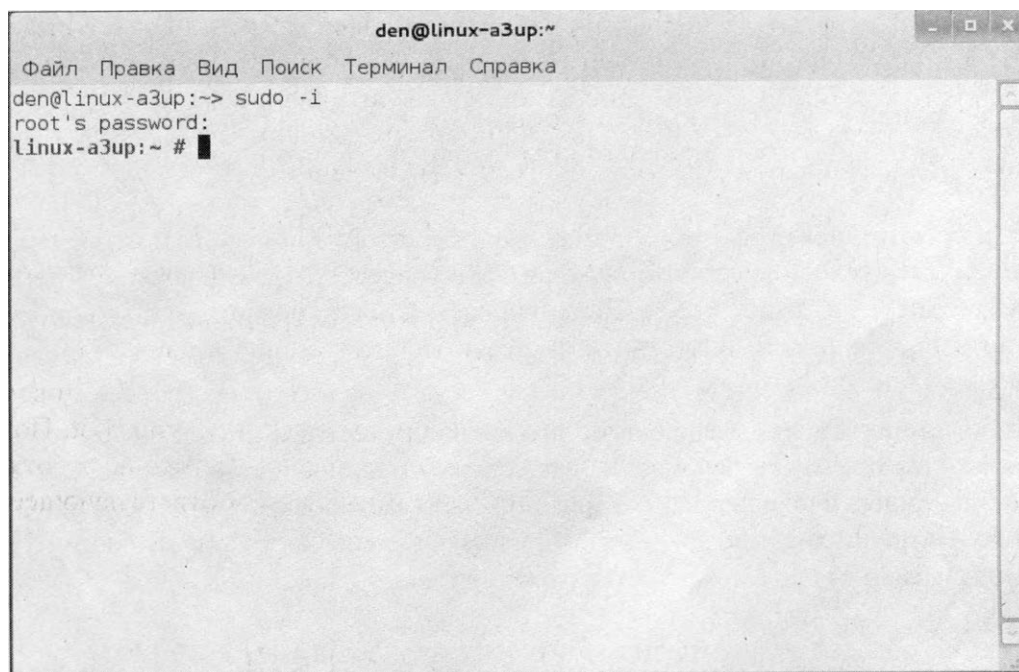


Рис. 6.3. Оболочка `root`

Можно также запустить оболочку `bash` с правами `root` — тогда все команды, введенные в этой оболочке, будут выполнены с правами `root`, например:

```
sudo bash
```

6.2.3. Переход к традиционной учетной записи `root`

Преимущества и недостатки `sudo`

Как уже было отмечено, во многих дистрибутивах учетная запись `root` несколько ограничена в использовании: в одних дистрибутивах она отключена, и для получения необходимых полномочий приходится задействовать команду `sudo`, в других невозможно, например, войти как `root` в графическом режиме.

Тем не менее, возможность перейти к традиционной учетной записи `root`, т. е. заходить в систему под именем `root`, имеется всегда. Чуть позже мы разберемся, как это сделать, но сначала рассмотрим преимущества (и недостатки) использования команды `sudo`.

Преимущества `sudo` заключаются в следующем:

- вам не нужно помнить несколько паролей (т. е. ваш пользовательский пароль и пароль пользователя `root`)— вы помните только свой пароль и вводите его, когда нужно;
- с помощью `sudo` вы можете выполнять практически те же действия, что и под именем `root`, но перед каждым действием у вас будет запрошен пароль, что позволит еще раз подумать о правильности своих действий;
- каждая команда, введенная с помощью `sudo`, записывается в журнал `/var/log/auth.log`, и у вас сохранится история введенных команд с полномочиями `root`, тогда как при работе под именем `root` никакого журнала не ведется. Кроме того, если что-то пойдет не так, вы хотя бы будете знать, что случилось, изучив этот журнал;
- предположим, некто захотел взломать вашу систему. Он не знает, какие учетные записи имеются на вашем компьютере, зато уверен, что учетная запись `root` есть всегда. Знает он также, что завладев паролем к этой учетной записи, можно получить неограниченный доступ к системе. Но не к вашей системе — у вас-то учетная запись `root` отключена!
- вы можете разрешать и запрещать другим пользователям использовать полномочия `root` (позже мы разберемся, как это сделать), не предоставляя пароль `root`, — это практически сводит на нет риск скомпрометировать учетную запись `root` (впрочем, риск есть всегда— ведь при неправильно настроенной системе с помощью команды `sudo` можно легко изменить пароль `root`).

Но у `sudo` есть и недостатки:

- неудобно задавать перенаправление ввода/вывода — например, команда:

```
sudo ls /etc > /root/somefile
```

работать не будет, и вместо нее следует использовать команду:

```
sudo bash -c "ls /etc > /root/somefile"
```

Длинновато, правда?

- имеются также неудобства, связанные с технологией NSS. К счастью, она используется не очень часто, поэтому основной недостаток `sudo` будет связан только с перенаправлением ввода/вывода.

С другой стороны, если вы знаете пароль `root`, то можно просто ввести команду `su` и получить полноценную сессию `root` без необходимости постоянно вводить команду `sudo`, — и удобно, и безопасно. Исходя из этих соображений, в этом издании не будет показано, как обеспечить вход пользователя `root` в графическом режиме, — это очень небезопасно, поскольку с максимальными правами будут запускаться не только вводимые вами команды, но и все остальные, которые запускает система. Зато мы разберемся, как включить учетную запись `root`, поскольку в некоторых дистрибутивах (и в частности, в Ubuntu) она по умолчанию выключена. После чего вы сможете входить как `root` в консоли.

Традиционная учетная запись root в Ubuntu

Вы все-таки хотите использовать обычную учетную запись root? Для этого достаточно задать пароль для пользователя root. Делается это командой:

```
sudo passwd root
```

Сначала программа запросит ваш пользовательский пароль, затем новый пароль root и его подтверждение:

Enter your existing password:

Enter password for root:

Confirm password for root:

После этого вы сможете входить в систему под учетной записью root.

Для отключения учетной записи root используется команда:

```
sudo passwd -l root
```

Помните, что после закрытия учетной записи root у вас могут быть проблемы с входом в систему в режиме восстановления, поскольку пароль root уже установлен (т. е. он не пустой, как по умолчанию), но в то же время учетная запись закрыта. Поэтому, если вы уж включили учетную запись root, будьте внимательны и осторожны. А, вообще, повторюсь — лучше ее не включать, а пользоваться командой `sudo -i`.

6.3. Создание, удаление и модификация пользователей и групп стандартными средствами

6.3.1. Отдельные пользователи

Для добавления нового пользователя выполните следующие команды (от имени root):

```
# adduser <имя пользователя>
# passwd <имя пользователя>
```

Первая команда (`adduser`) добавляет пользователя, а вторая (`passwd`) изменяет его пароль. Ясно, что и в первом, и во втором случае вы должны указать одно и то же имя пользователя.

В некоторых дистрибутивах — например, в Ubuntu и Debian — сценарий `adduser` не только добавляет пользователя, но и позволяет указать дополнительную информацию о пользователе и сразу же задать пароль пользователя (рис. 6.4).

СЦЕНАРИИ ADDUSER И USERADD

В некоторых дистрибутивах (например, в openSUSE) вместо команды `adduser` используется команда `useradd`. Сценарии `adduser` и `useradd` обычно находятся в каталоге `/usr/sbin`.

```

denix@denix-pc: ~
denix@denix-pc:~$ sudo adduser denis
[sudo] password for denix:
Добавляется пользователь `denis` ...
Добавляется новая группа `denis` (1001) ...
Добавляется новый пользователь `denis` (1001) в группу `denis` ...
Создаётся домашний каталог `/home/denis` ...
Копирование файлов из `/etc/skel` ...
Введите новый пароль UNIX:
Повторите ввод нового пароля UNIX:
passwd: пароль успешно обновлён
Изменение информации о пользователе denis
Введите новое значение или нажмите ВВОД для выбора значения по умолчанию
Полное имя []: Denis Kolisnichenko
Номер комнаты []:
Рабочий телефон []:
Домашний телефон []:
Другое []:
Данная информация корректна? [Д/н] у
denix@denix-pc:~$

```

Рис. 6.4. Ubuntu: добавление нового пользователя

Обратите внимание — если пароль слишком прост для подбора, программа `passwd` выдаст соответствующее предупреждение: `BAD PASSWORD` и сообщит, чем же наш пароль плох (например, в основе пароля лежит словарное слово, что делает пароль легким для подбора).

Для модифицирования учетной записи пользователя можно использовать команду `usermod`. О ней вы прочитаете в руководстве `man`, вызвав его командой:

```
man usermod
```

Особого смысла рассматривать эту команду я не вижу, ведь обычно нужно менять только пароль пользователя, а это можно сделать с помощью команды `passwd`. А если вам требуется изменить саму учетную запись (например, указать другой домашний каталог), то это гораздо удобнее сделать с помощью графического конфигуратора (об этом позже) или обычного текстового редактора.

КОМАНДА `PASSWD`

Команду `passwd` может использовать не только администратор, но и сам пользователь для изменения собственного пароля.

Для удаления пользователя служит команда `userdel`:

```
# userdel <имя пользователя>
```

Давайте разберемся, что же происходит при создании новой учетной записи пользователя.

Во-первых, создается запись в файле `/etc/passwd`. Формат записи следующий:

```
имя_пользователя: пароль : UID: GID: полное_имя: домашний_каталог: оболочка
```

Рассмотрим фрагмент этого файла (две строки):

```
root:x:0:0:root:/root:/bin/bash
den:x:500:500:Denis:/home/den:/bin/bash
```

- первое поле— это логин пользователя, который он вводит для регистрации в системе. Пароль в современных системах в этом файле не указывается, и второе поле осталось просто для совместимости со старыми системами. Пароли хранятся в файле `/etc/shadow`, о котором мы поговорим чуть позже;
- третье и четвертое поле: `UID` (User ID) и `GID` (Group ID)— идентификаторы пользователя и группы соответственно. Идентификатор пользователя `root` всегда равен 0, как и идентификатор группы `root`. Список групп вы найдете в файле `/etc/groups`;
- пятое поле — это настоящее имя пользователя. Оно может быть не заполнено, а может содержать фамилию, имя и отчество пользователя, — все зависит от педантичности администратора системы, т. е. от вас. Если вы работаете за компьютером в гордом одиночестве, то, думаю, свою фамилию вы не забудете. А вот если ваш компьютер — сервер сети, тогда просто необходимо указать Ф.И.О. каждого пользователя, а то, когда придет время обратиться к пользователю по имени, вы его знать не будете (попробуйте запомнить 500 фамилий и имен!);
- шестое поле содержит имя домашнего каталога. Обычно это каталог `/home/<имя_пользователя>`;
- последнее поле — это имя командного интерпретатора, который будет обрабатывать введенные вами команды, когда вы зарегистрируетесь в консоли.

В целях повышения безопасности пароли в свое время были перенесены из файла `/etc/passwd` в файл `/etc/shadow` (доступен для чтения/записи только пользователю `root`), где они и хранятся в закодированном виде (используются алгоритмы MD5 или Blowfish— в некоторых системах). Узнать, с помощью какого алгоритма зашифрован пароль, очень просто: посмотрите на шифр — если он достаточно короткий и не начинается с символа `$`, то применен алгоритм DES (самый слабый и ненадежный— он, как правило, используется в старых дистрибутивах). Если же шифр начинается с символов `1`, то это MD5, а если в начале шифра имеются символы `$2a$`, то это Blowfish.

Во-вторых, при создании пользователя создается каталог `/home/<имя пользователя>` в который копируется содержимое каталога `/etc/skel`. Каталог `/etc/skel` содержит «джентльменский набор» — файлы конфигурации по умолчанию, которые должны быть в любом пользовательском каталоге. Название каталога `skel` (от *skeleton*) полностью оправдывает себя — он действительно содержит «скелет» домашнего каталога пользователя.

РЕДАКТИРОВАНИЕ ФАЙЛА `/ETC/PASSWD`

Файл `/etc/passwd` можно редактировать с помощью обычного текстового редактора. То есть, вы можете очень легко, не прибегая к помощи ни графического конфигуратора, ни команды `usermod`, изменить параметры учетной записи любого пользователя, — например, задать для него другую оболочку или прописать его настоящую фамилию.

Однако при изменении домашнего каталога пользователя нужно быть осторожным! Если вы это сделали, то, чтобы у пользователя не возникло проблем с правами доступа к новому каталогу, следует выполнить команду:

```
chown -R Пользователь > <каталог>
```

6.3.2. Группы пользователей

Иногда пользователей объединяют в *группы*. Группы позволяют более эффективно управлять правами пользователей. Например, над каким-либо проектом у вас должны совместно работать три разных пользователя — их достаточно объединить в одну группу, и тогда они получают доступ к домашним каталогам друг друга (по умолчанию один пользователь не имеет доступа к домашнему каталогу другого, поскольку они находятся в разных группах).

Создать группу, а также поместить пользователя в группу позволяют графические конфигураторы. Вы можете использовать их — они очень удобные, но если вы хотите стать настоящим линуксоидом, то должны знать, что доступные в системе группы указываются в файле `/etc/group`. Добавить новую группу в систему можно с помощью команды `groupadd`, но, как правило, проще в текстовом редакторе добавить еще одну запись в файл `/etc/group`, а изменить группу пользователя еще легче — для этого достаточно отредактировать файл `/etc/passwd`.

6.4. Управление пользователями и группами с помощью графических конфигураторов

Обычно добавлять/изменять учетные записи пользователей принято в командной строке. Но сейчас мы поговорим о *графических конфигураторах* — они пригодятся любителям графического интерфейса, а также начинающим пользователям, которые еще не уверены в своих силах. Понятно, что в каждом дистрибутиве имеются свои конфигураторы, поэтому мы остановимся лишь на трех наиболее популярных дистрибутивах: Fedora, Ubuntu и openSUSE (с графическими конфигураторами других дистрибутивов после этого, думаю, вы разберетесь и без моих комментариев).

6.4.1. Конфигураторы в Fedora и Ubuntu

Конфигураторы управления учетными записями в Fedora и Ubuntu — как два брата-близнеца. Вы только посмотрите на рис. 6.5 и 6.6: на первом представлен конфигуратор Ubuntu 17.04, на втором — Fedora 26. В конфигураторе Fedora 26 список пользователей сейчас отображается в верхней части окна, а не слева, как в Ubuntu 17.04, но до этой версии конфигураторы выглядели вообще одинаково.

Вот только запускаются конфигураторы по-разному:

- в Fedora нужно открыть окно **Все параметры** и щелкнуть на кнопке **Пользователи**;
- в Ubuntu нужно нажать на панели Unity кнопку **Параметры системы** (она имеет вид гаечного ключа на шестеренке) и запустить конфигуратор **Учетные записи**.

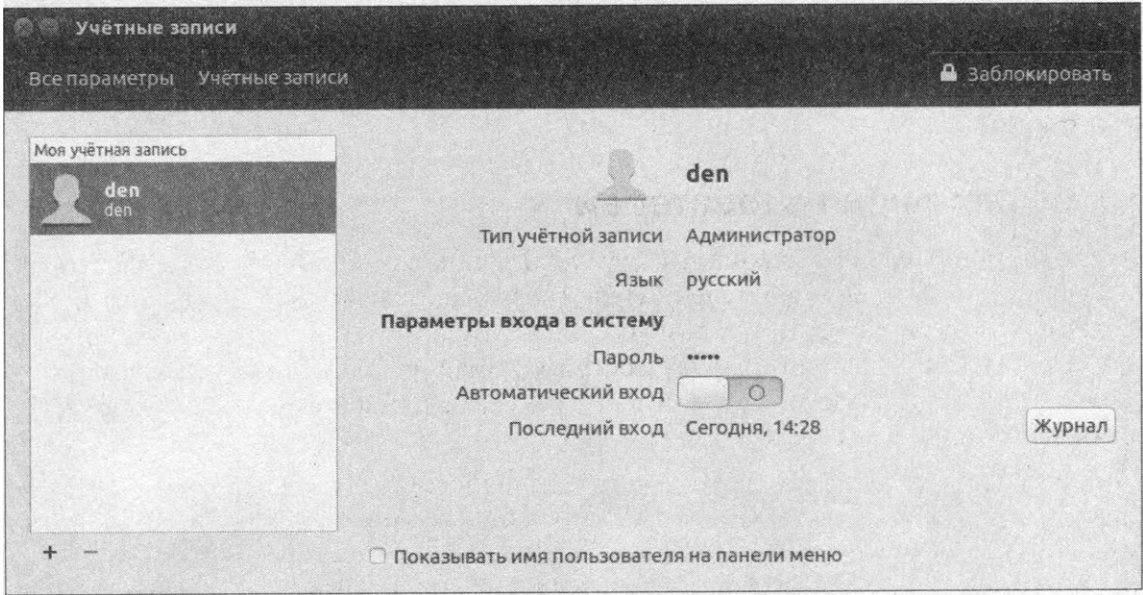


Рис. 6.5. Ubuntu 17.04: конфигуратор управления пользователями

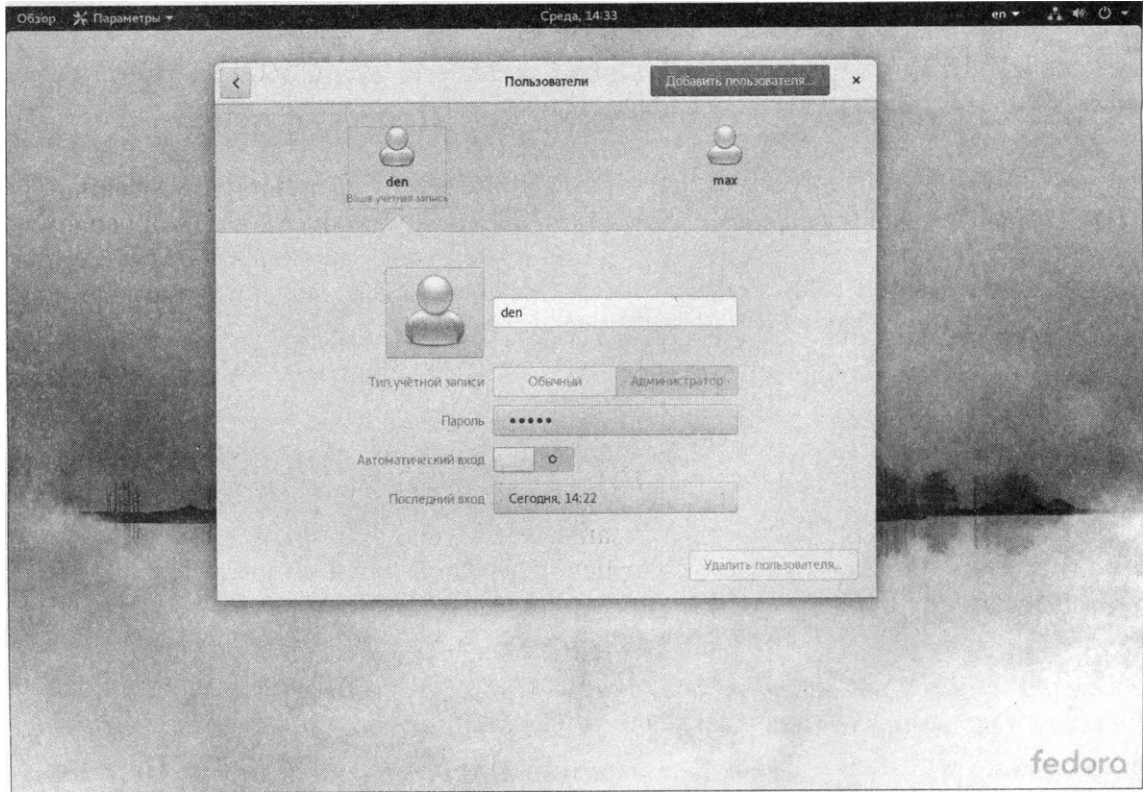


Рис. 6.6. Fedora 26: конфигуратор управления пользователями

Первым делом нужно разблокировать конфигуратор — для этого нажмите кнопку **Разблокировать** в верхнем правом углу окна. Откроется окно (рис. 6.7), в котором надо ввести пароль пользователя (не root, а того, от имени которого запущен конфигуратор).

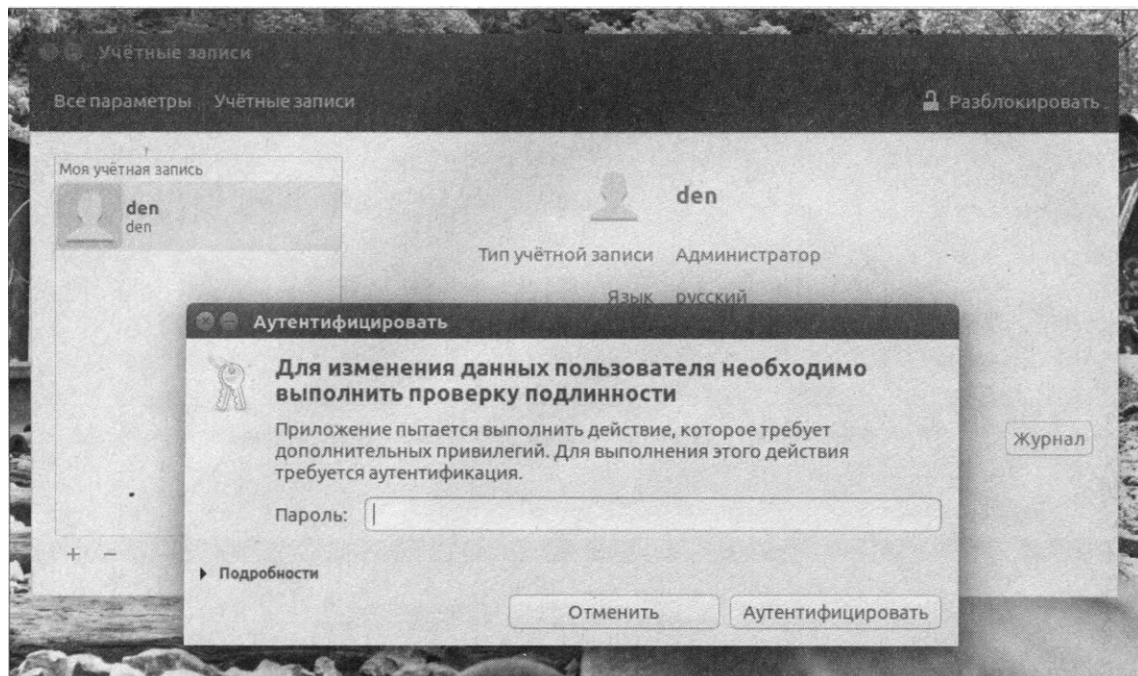


Рис. 6.7. Ubuntu 17.04: разблокирование конфигуратора

ПРИМЕЧАНИЕ

Далее снимки с экрана соответствуют дистрибутиву Ubuntu, но в Fedora все выглядит так же (за исключением самого оформления окна).

Начнем с изменения собственной учетной записи. Чтобы изменить какое-либо поле, надо по нему щелкнуть. Например, для изменения изображения пользователя щелкните по пользователю и выберите подходящее для него изображение (рис. 6.8). Аналогично изменяются имя пользователя (у изображения), тип учетной записи, язык, пароль и тип входа в систему.

Особого внимания заслуживает выбор типа учетной записи. Здесь предлагаются варианты (рис. 6.9):

- **Обычный** — пользователь может работать в системе, но не может администрировать ее (использовать команду `sudo`, устанавливать программы, управлять пользователями и т. д.);
- **Администратор** — пользователь может администрировать систему.

Изменить тип какой-либо учетной записи можно, только если в вашей системе есть несколько пользователей, и один из них администратор. Но если, например, у вас есть только один пользователь, и он же является администратором, то тип его учет-

ной записи изменить нельзя. При создании же новой учетной записи выбрать для нее тип разрешается.

Для добавления пользователя нажмите под списком пользователей кнопку +. Вам потребуется выбрать тип учетной записи, ввести имя пользователя и его полное имя (см. рис. 6.9). После этого учетную запись пользователя нужно отредактировать: установить пароль, картинку и т. д.

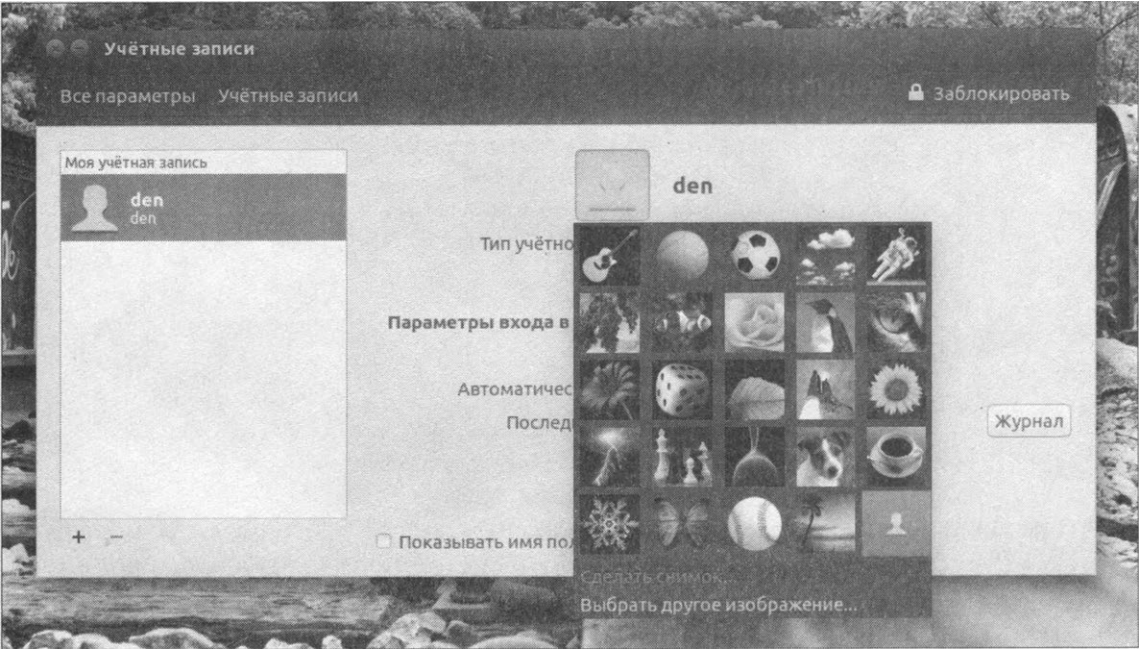


Рис. 6.8. Ubuntu 17.04: выбор картинки пользователя

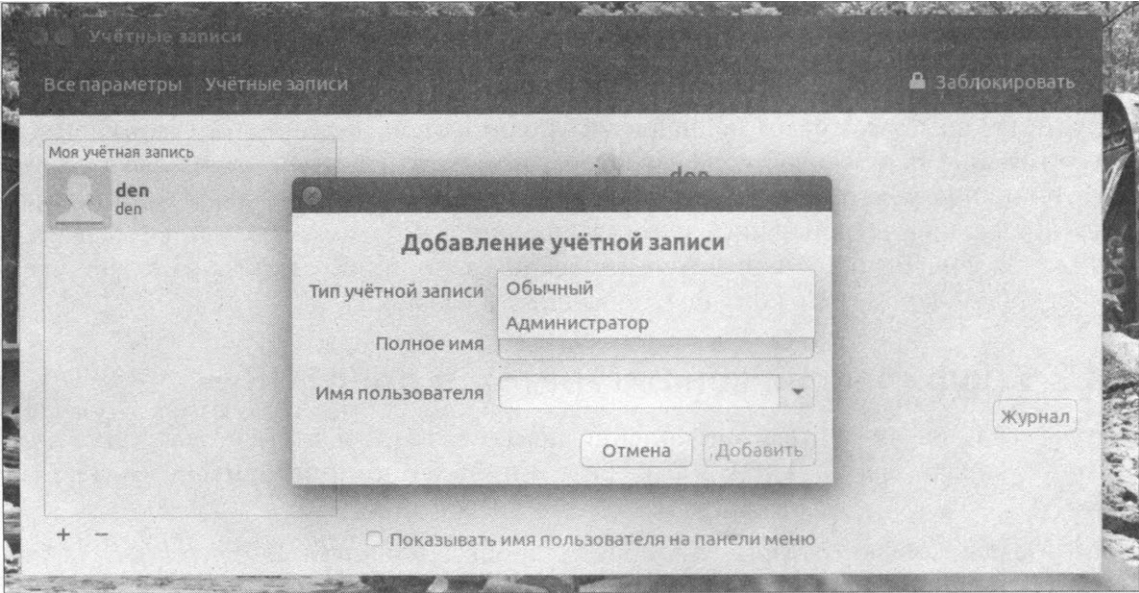


Рис. 6.9. Ubuntu 17.04: выбор типа учетной записи при создании новой учетной записи

Для удаления пользователя служит имеющаяся под списком пользователей кнопка **При этом** конфигуратор спросит, что сделать с файлами пользователя (с его домашним каталогом): удалить или сохранить на диске (рис. 6.10).

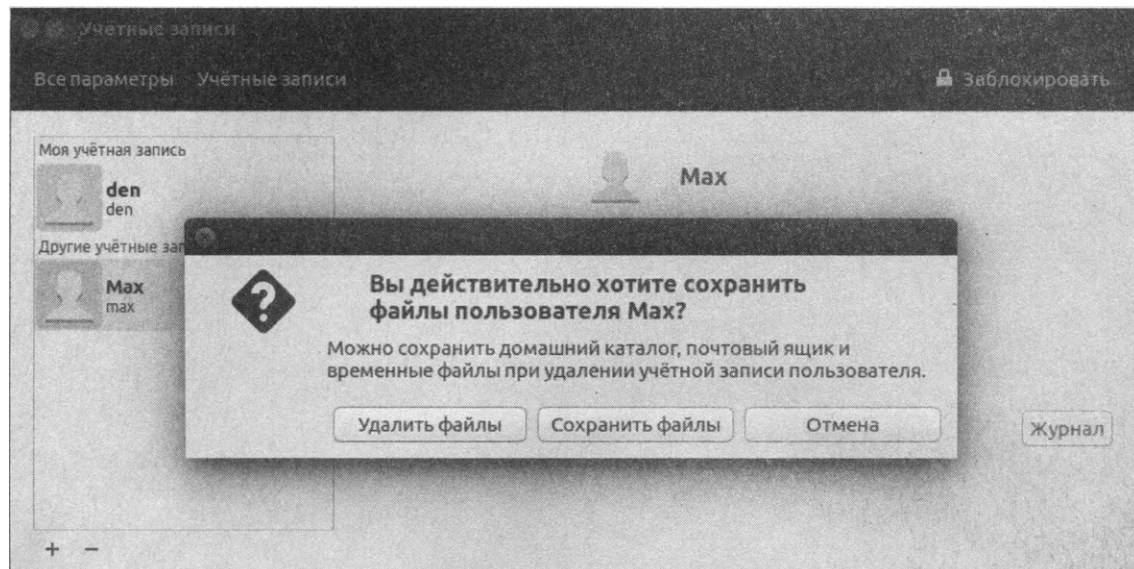


Рис. 6.10. Ubuntu 17.04: удаление пользователя

Если выбрана опция **Автоматический вход** (см. рис. 6.5 и 6.6), то при загрузке системы будет выполнен вход пользователя в систему без запроса его имени и пароля. Включение автоматического входа件лезно или когда вы работаете в гордом одиночестве и вам нечего скрывать, или когда вы настраиваете публичный компьютер (в интернет-кафе, библиотеке). Во втором случае из соображений безопасности не следует включать автоматический вход для пользователя с административными правами.

Сожалею, но современные графические конфигураторы управления пользователями Ubuntu и Fedora меня разочаровали. В предшествующих их версиях можно было выбрать группы, к которым принадлежит пользователь, установить его расширенные права и т. п. А сейчас конфигураторы позволяют выполнить только базовые операции с пользователем: создать и удалить учетную запись — даже возможности редактирования учетной записи и то ограничены. Поэтому, на мой, естественно, взгляд, эти конфигураторы практически бесполезны, и для управления пользователями лучше использовать команды, описанные в *разд. 6.3*.

6.4.2. Графический конфигуратор в openSUSE

Для запуска конфигулятора **Управление пользователями и группами** (рис. 6.11) запустите конфигуратор **YaST** и выберите **Управление пользователями и группами**.

Использовать конфигуратор очень просто: кнопка **Добавить** служит для создания нового пользователя, а кнопки **Редактировать** и **Удалить** — для изменения и удаления, соответственно, уже созданного.

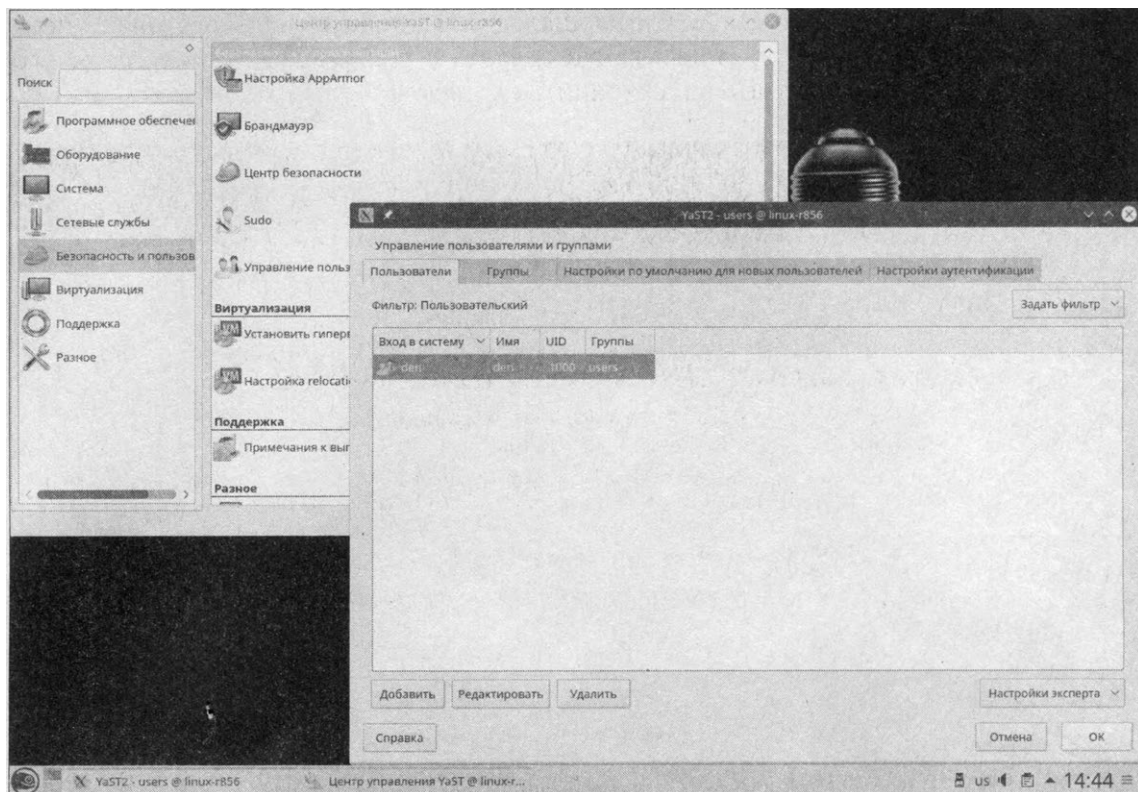


Рис. 6.11. openSUSE: окно Управление пользователями и группами, вкладка Пользователи

При создании пользователя (рис. 6.12) у вас есть возможность (на вкладке **Подробности**) выбрать, к каким группам должен принадлежать этот пользователь. Если пользователю не нужен доступ к Интернету, не следует пометать его принадлежность к группе **dialout**.

Даже если при создании пользователя вы забыли определить группы, к которым должен принадлежать пользователь, то всегда сможете сделать это позже — при изменении его учетной записи (кнопка **Редактировать**).

ЗАПРЕТ ВХОДА В СИСТЕМУ

Если вам нужно временно запретить пользователю вход в систему (но удалять его вы не хотите), выделите этого пользователя, нажмите кнопку **Редактировать** и установите флажок **Отключить вход пользователя в систему**.

Для редактирования групп: создания, удаления, изменения списка членов группы — следует перейти на вкладку **Группы** (рис. 6.13). Нажав здесь кнопку **Редактировать**, вы можете изменить параметры группы (рис. 6.14), — например, добавить в ее состав новых пользователей. А вот чтобы удалить пользователя из группы, вам придется перейти на вкладку **Пользователи**, выбрать нужного пользователя, нажать кнопку **Редактировать**, затем перейти на вкладку **Подробности** и уже там отключить группы, членом которых не должен быть пользователь. Да, неудобно, но другого способа нет.

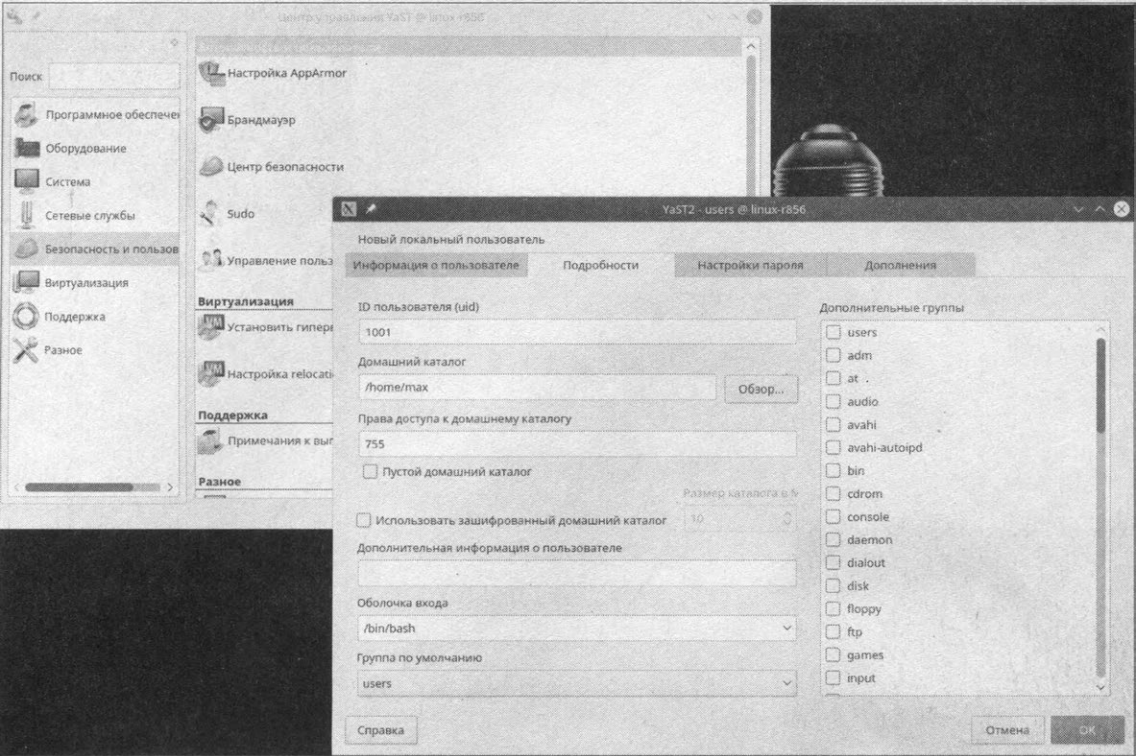


Рис. 6.12. openSUSE: создание нового пользователя

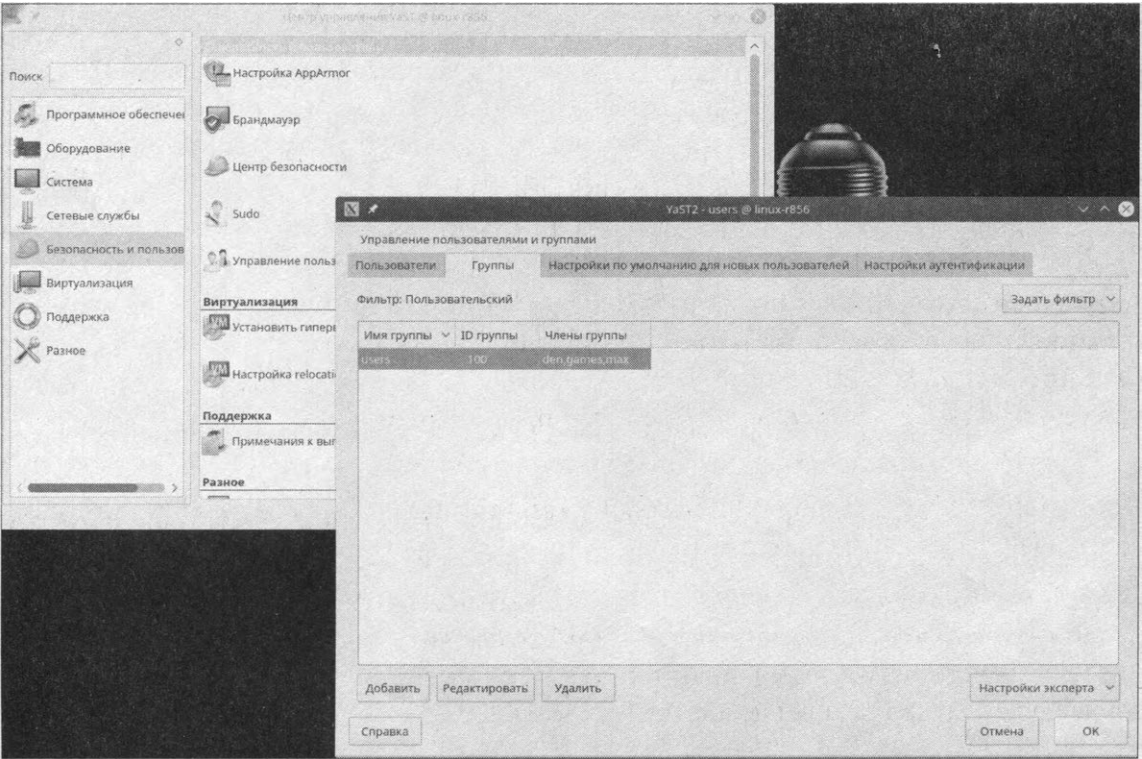


Рис. 6.13. openSUSE: окно Управление пользователями и группами, вкладка Группы

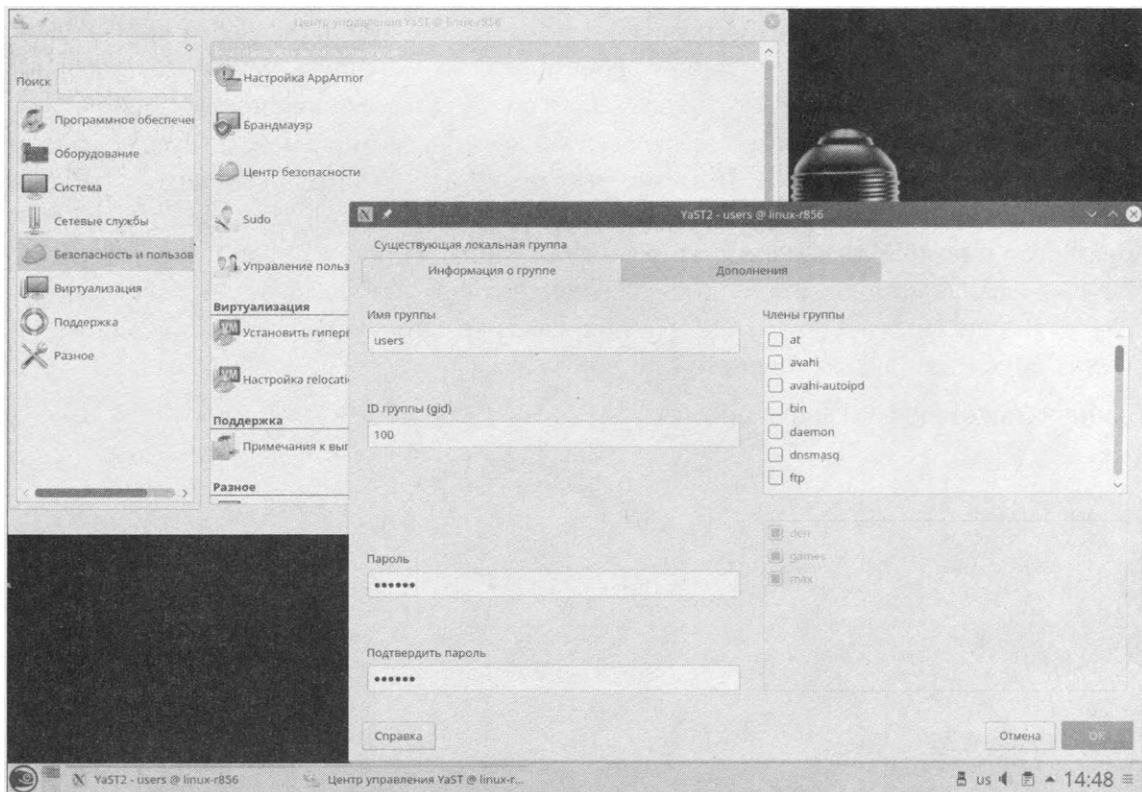


Рис. 6.14. openSUSE: изменение группы

Еще о правах root и командах *su* и *sudo* применительно к openSUSE

Когда вы запускаете какой-нибудь конфигуратор, система просит вас ввести пароль root. Вы его вводите, запускается конфигуратор с правами root, и вы успешно производите настройку системы.

А что делать, если вам нужно отредактировать вручную какой-нибудь файл конфигурации, — например: `/boot/grub/menu.lst`? Если вы его откроете в текстовом редакторе, в том же gedit, то не сможете потом сохранить изменения, поскольку у вас нет прав доступа к каталогу `/boot` (точнее, нет права изменять файлы в этом каталоге). Короче, вам нужны права root.

Чтобы их получить, откройте терминал (**Приложения | Стандартные | Терминал**) среды GNOME и введите команду `su`.

Программа `su` запросит у вас пароль пользователя root. При вводе пароля в терминале он не отображается на экране — просто введите пароль и нажмите клавишу `<Enter>`. Теперь вы можете вводить команды от имени пользователя root. В нашем случае для редактирования файла `/boot/grub/menu.lst` нужно ввести команду:

```
gedit /boot/grub/menu.lst
```


Если вы работаете за компьютером один, можете смело использовать команду `su`. Но бывают ситуации, когда нужно предоставить возможность настройки компьютера другому пользователю, но вы не хотите сообщать ему пароль `root`. В этом случае на помощь приходит команда `sudo`. После ввода команды `sudo` нужно ввести *свой пароль*, а не пароль `root`. Понятно, что право использовать `sudo` имеет не каждый пользователь, а только указанные в файле `/etc/sudoers` (файл редактируется не вручную, а с помощью конфигуратора **YaST | Sudo**). Но по умолчанию в openSUSE для этого файла установлена политика, разрешающая использовать `sudo` всем пользователям системы (рис. 6.15). Да, это неправильно с точки зрения безопасности, но вполне приемлемо для домашнего компьютера.

Выполнять команду `sudo` нужно так:

```
sudo команда_которую_нужно_выполнить_с_правами_root
```

Например,

```
sudo gedit /boot/grub/menu.lst
```

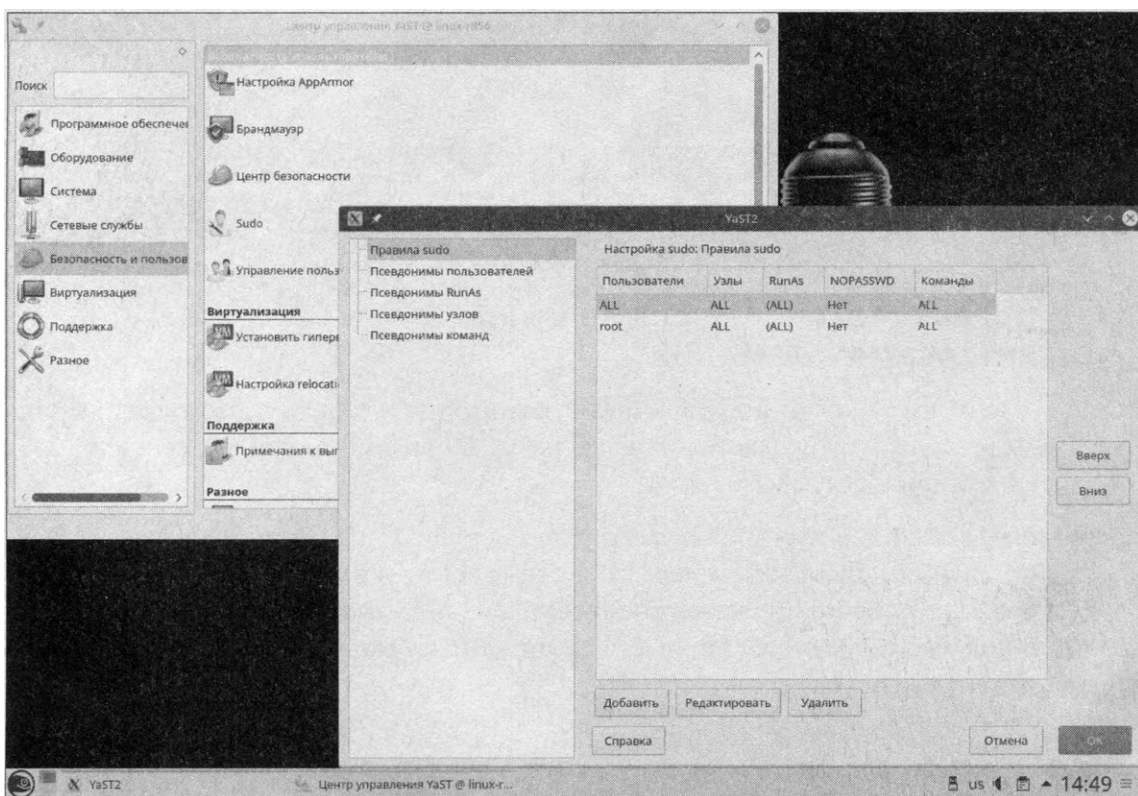


Рис. 6.15. openSUSE: использовать команду `sudo` могут все пользователи этой системы

Конфигуратор *Центр безопасности* openSUSE

В группе **Безопасность и пользователи** конфигуратора YaST имеется конфигуратор **Центр безопасности**. При его запуске открывается окно **Обзор безопасности**

(рис. 6.16), в котором содержится список настроек, определяющих безопасность системы.

- Для максимальной безопасности в разделе **Обзор безопасности** выберите следующие установки:
 - **Использовать безопасные разрешения файлов** — в файлах `/etc/permissions` * содержатся разрешения файлов. Самые жесткие разрешения находятся в файлах `secure` или `paranoid`;
 - **Запускать демон DHCP в chroot** — демон DHCP будет запускаться в chroot-окружении (в так называемой «песочнице»). Даже если его взломают, злоумышленник не сможет добраться до основной файловой системы компьютера;
 - **Удаленный доступ к X-серверу** — не выбирайте эту опцию, если планируете предоставить удаленный доступ к своему компьютеру;
 - **IPv6-переадресация**— IPv6 пока не используется, поэтому переадресация IPv6 не нужна.

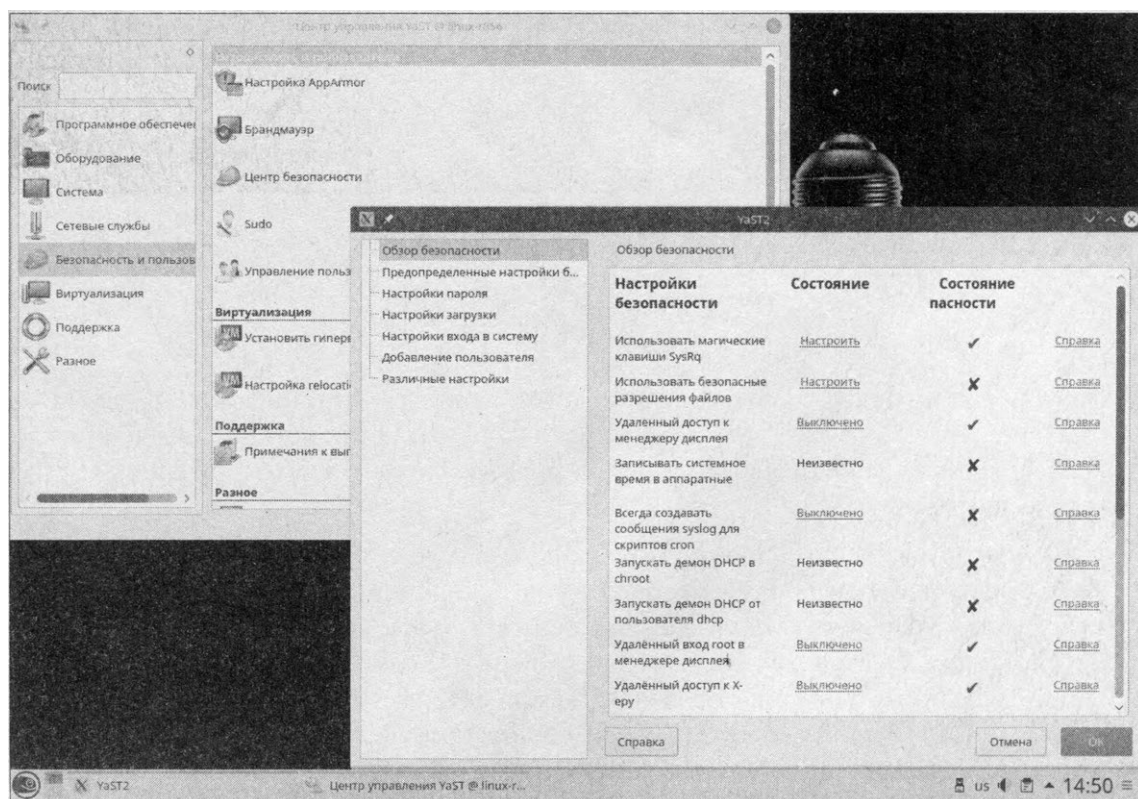


Рис. 6.16. openSUSE: Центр безопасности

- В разделе **Предопределенные настройки** вы можете выбрать параметры безопасности для домашнего компьютера, для рабочей станции и для сервера сети. По умолчанию используются пользовательские настройки, определенные ранее.

- Раздел **Настройки пароля** (рис. 6.17) позволяет изменить параметры паролей — например, выбрать другой метод шифрования (хотя используемый по умолчанию Blowfish является самым безопасным), установить «возраст» пароля.

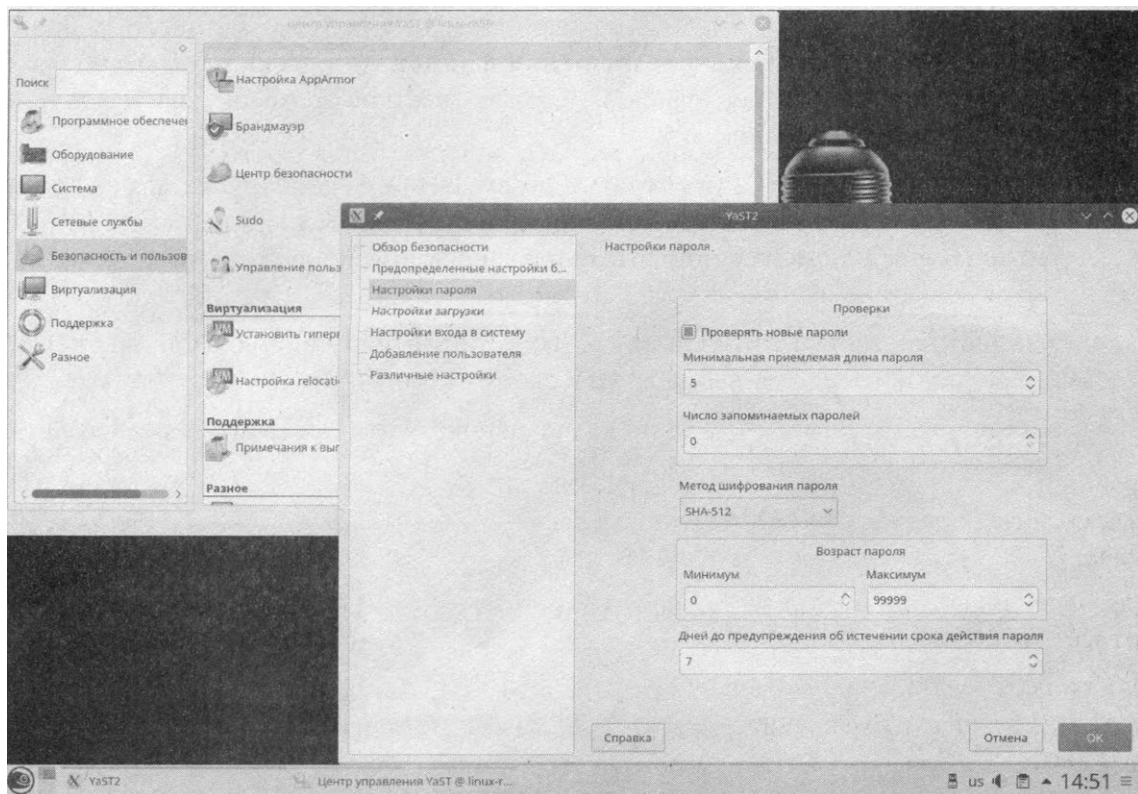


Рис. 6.17. openSUSE: параметры пароля

- В разделе **Настройки загрузки** вы можете установить реакцию на нажатие комбинации клавиш <Ctrl>+<Alt>+. Выключить реакцию на нажатие этой комбинации клавиш целесообразно на сервере, чтобы никто случайно его не перезагрузил.
- Параметры из разделов **Настройки входа в систему** и **Добавление пользователя** вы вряд ли будете изменять, а вот в разделе **Различные настройки** имеется параметр **Разрешить магические клавиши SysRq** — включите его, если ваша система часто зависает, и вам нужно контролировать процесс ее «разгрузки», когда система находится в «полузависшем» состоянии. Описание этих «магических» (они же аварийные) клавиш приведено в *разд. 3.5*.

ПАРАМЕТР «ИСПОЛЬЗОВАТЬ МАГИЧЕСКИЕ КЛАВИШИ SysRq»

Как можно видеть на рис. 6.16, параметр **Использовать магические клавиши SysRq** находится в разделе **Обзор безопасности**. Это тот же самый параметр, что и параметр **Разрешить магические клавиши SysRq** из раздела **Различные настройки**.

Почему один и тот же параметр в разных разделах называется по-разному, мне не понятно. Им виднее...

Как видите, средства управления пользователями в openSUSE намного удобнее, чем в Ubuntu и Fedora. При использовании этих средств практически нет необходимости задействовать консольные утилиты.

6.5. Квотирование

Квотирование — это механизм ограничения дискового пространства пользователей. Linux — система многопользовательская, поэтому без ограничения дискового пространства вам не обойтись. Когда используешь компьютер в гордом одиночестве, то все дисковое пространство доступно вам и только вам. А вот когда пользователей несколько, нужно ограничить доступное пространство, чтобы один из пользователей не «узурпировал» все место на диске. Как именно вы будете ограничивать дисковое пространство, решать только вам — можно поделить дисковое пространство поровну между пользователями, можно одним пользователям отдать больше места, а другим — меньше.

На домашнем компьютере квотирование вряд ли понадобится, а на сервере, как правило, для каталога `/home` отводится отдельный раздел жесткого диска. Поэтому будем считать, что у нас есть отдельный раздел, который монтируется к каталогу `/home`.

Для настройки квот нужно установить пакет `quota`. Более ничего устанавливать не потребуется.

Чтобы пользователи не потеряли свои данные, перезагрузитесь в однопользовательский режим (параметр ядра `single`). Теперь можно приступить к редактированию квот.

Первым делом разрешим устанавливать квоты на разделе, который содержит файлы пользователей. Откройте файл `/etc/fstab`:

```
# nano /etc/fstab
```

Добавьте параметр `usrquota` к списку параметров раздела:

```
/dev/sda5    /home          ext4 defaults,usrquota    0    2
```

Параметр `usrquota` включает поддержку квот для отдельных пользователей. Если вам нужна поддержка квот групп пользователей, тогда добавьте параметр `grpquota`.

Теперь перемонтируем `/home`, поскольку мы только что изменили его параметры:

```
# mount -o remount /home
```

Механизм квотирования требует создания файлов `aquota.user` и `aquota.group`, но поскольку мы будем устанавливать квоты только для пользователей, а не для групп, то и создадим лишь файл `aquota.user`:

```
# touch /home/aquota.user'  
# chmod 600 /home/aquota.user
```

После этого введем команду:

```
# quotacheck -vagum
```

Раз мы создали файл `aquota.user` вручную, то получим сообщение об ошибке, но это только в первый раз, — далее все будет нормально:

quotacheck: WARNING— Quotafile /home/aquota.user was probably truncated. Can't save quota settings...

quotacheck: Scanning /dev/sda5 [/home] quotacheck: Old group file not found.

Usage will not be substracted.

done

quotacheck: Checked 3275 directories and 54301 files

Теперь отредактируем квоты для пользователя `user`:

```
# edquota -u user
```

Будет запущен текстовый редактор по умолчанию, и вы увидите следующий текст:

```
Disk quotas for user user (uid 1001):
```

Filesystem	blocks	soft	hard	inodes	soft	hard
/dev/sda5	16	0	0	5	0	0

УСТАНОВКА РЕДАКТОРА ПО УМОЛЧАНИЮ

По умолчанию используется редактор `vi`, который, мягко говоря, не очень удобен. Для изменения редактора по умолчанию установите переменную окружения `editor`. Например, `EDITOR=nano`.

Разберемся, что это значит:

- `blocks` — место в блоках, используемое пользователем (1 блок = 1 Кбайт);
- `soft` — максимальное дисковое пространство (в блоках по 1 Кбайт), которое может занимать пользователь. Если вы включите период отсрочки (`grace period`), то пользователь получит только лишь сообщение о превышении квоты;
- `hard` — жесткое ограничение, эту квоту пользователь превысить не может, даже если включен период отсрочки. Предположим, вы хотите «отдать» пользователю 500 Мбайт. В качестве жесткой квоты можно установить значение 500 Мбайт (или 500 000 блоков), а в качестве «мягкой» — значение 495 Мбайт (495 000 блоков). Когда пользователь превысит 495 Мбайт, он получит сообщение о превышении квоты, а вот когда будет превышена жесткая квота, то пользователь больше не сможет сохранять файлы в своем домашнем каталоге;
- `inodes` — число используемых пользователем файлов.

Отредактируйте квоты так:

```
Disk quotas for user user (uid 1001):
```

Filesystem	blocks	soft	hard	inodes	soft	hard
/dev/sda5	16	95000	500000	5	0	0

Теперь сохраните файл, выйдите из редактора и введите команду:

```
# edquota -t
```

Сейчас мы установим период отсрочки:

```
Grace period before enforcing soft limits for users:
  Time units may be: days, hours, minutes, or seconds
  Filesystem Block grace period Inode grace period
  /dev/sda8      7days          7days
```

Вы должны вместо `7days` вписать свой период отсрочки, при этом используйте названия единиц изменения времени на английском:

- | | |
|--|--|
| <input type="checkbox"/> <code>seconds</code> — секунды; | <input type="checkbox"/> <code>days</code> — дни; |
| <input type="checkbox"/> <code>minutes</code> — минуты; | <input type="checkbox"/> <code>weeks</code> — недели; |
| <input type="checkbox"/> <code>hours</code> — часы; | <input type="checkbox"/> <code>months</code> — месяцы. |

Например:

- ☐ `24hours` — 24 часа;
- ☐ `2days` — 2 дня;
- ☐ `1 weeks` — 1 неделя.

Включим квотирование для наших файловых систем:

```
# quotaon файловая_система
```

Например:

```
# quotaon /
```

После этого перезагрузим систему:

```
# reboot
```

При загрузке вы увидите сообщение: **Turning on user and group quotas for local filesystems** (Включаем квоты пользователей и групп для локальных файловых систем) — это означает, что механизм квотирования работает правильно.

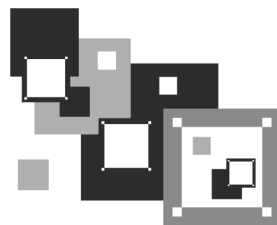
Для просмотра квот служит команда `repquota`, например:

```
# repquota /home
```

Наверняка использовать редактор `vi` вам не очень нравится. Значительно упрощают задание квот так называемые *прототипы*. Например, вы задали ограничение для пользователя `den`, но у вас есть еще несколько пользователей, для которых нужно задать такие же ограничения. Вы можете использовать квоту пользователя `den` в качестве прототипа:

```
# edquota -p den user1
# edquota -p den user2
```

ГЛАВА 7



Пакеты и управление пакетами

7.1. Способы установки программного обеспечения в Linux

Установка программного обеспечения в Linux осуществляется двумя основными способами:

- с помощью пакетов;
- из исходных кодов.

Пакет представляет собой набор файлов, содержащий все необходимое для установки программы. Существуют два основных типа пакетов:

- RPM-пакеты — применяются во всех Red Hat-совместимых дистрибутивах: Red Hat, Fedora, CentOS, ALT Linux и др.;
- DEB-пакеты — применяются в дистрибутиве Debian и в дистрибутивах, на нем основанных: Ubuntu, Kubuntu, Edubuntu, Denix и др.

Название RPM-пакетов связано с предложенной компанией Red Hat технологией, основанной на использовании менеджера пакетов rpm (Red Hat Package Manager). С DEB-пакетами все проще — они так названы, потому что последние три символа имени у файлов пакетов deb (сокращение от Debian).

ПАКЕТЫ SLACKWARE

В Slackware Linux используется собственный формат пакетов, не совместимый ни с RPM, ни с DEB. Об установке пакетов в Slackware мы поговорим отдельно.

Не найдя в своем дистрибутиве нужной вам программы, попробуйте поискать ее пакет на следующих сайтах:

- RPM-пакеты: <http://rpmfind.net> и <http://rpm.pbone.net>;
- DEB-пакеты: <http://www.debian.org/distrib/packages> и <http://packages.ubuntu.com/>.

Если вы не можете найти в Интернете комплектный пакет программы, тогда придется компилировать программу самому, — при условии, что вы нашли архив с ее исходным кодом. Да, в Linux некоторые программы распространяются только

в исходных кодах. Для установки такой программы нужно распаковать архив с ее исходными кодами (желательно, в каталог `/usr/src`), затем перейти в каталог, содержащий файлы распакованного архива исходных кодов, и поочередно выполнить следующие команды:

```
./configure  
make  
make install
```

Сценарий `configure` проверит, содержит ли ваша система необходимые библиотеки или программы, и, если все нормально, создаст файл `Makefile`. Если вы увидели сообщение об ошибке, внимательно прочитайте его и попытайтесь устранить причину ошибки, — например, установите недостающую библиотеку. Ясно, что в случае ошибки вводить последние две команды не нужно.

Вторая команда (`make`) на основании созданного файла `Makefile` компилирует программу. А последняя команда (`make install`) устанавливает программу и дополнительные файлы в дерево файловой системы: программы — обычно в каталог `/usr/bin`, документацию — в `/usr/share/doc`, конфигурационные файлы — в `/etc` и т. д.

ЧИТАЙТЕ ФАЙЛ README!

Для получения подробных инструкций по установке и удалению таких программ лучше всего прочитать файл `README`, который обычно присутствует в архиве.

Пакет установки программы, как правило, состоит из набора файлов — например, исполнимого и конфигурационного, а также файла справки. В зависимости от организации программы установки все эти файлы могут быть:

- заархивированы каждый отдельно — в этом случае мы получаем набор из архивов файлов программы плюс программа установки;
- заархивированы в один общий архив, содержащий комплект файлов программы и программу установки;
- укомплектованы в саму программу установки — самый удобный случай, когда у нас всего один файл, — программа установки.

Кроме файлов собственно программы, в пакете хранится также и служебная информация, описывающая процесс установки программы:

- *пути* — ведь один файл нужно скопировать, например, в каталог `/usr/bin`, а другой — в `/usr/share/doc`;
- *дополнительные действия* — например, создание каталога, установка тех или иных прав доступа к файлам и каталогам программы;
- *зависимости* — программе для работы может потребоваться какая-либо библиотека, без которой она не может запускаться, поскольку использует ее функции. Тогда в пакете указывается, что он *зависит* от другого пакета, содержащего эту библиотеку. При установке менеджер пакетов проверяет зависимости: если установлены не все пакеты, от которых зависит устанавливаемый пакет, установка будет прервана — пока вы не установите все необходимое. Впрочем, име-

ется возможность установки программы и без удовлетворения зависимостей (тогда информация о зависимостях будет просто проигнорирована), но в большинстве случаев установленная таким образом программа работать не станет;

- *конфликты* — та или иная программа может в системе конфликтовать с другой программой. Например, программы `sendmail` и `postfix` являются серверами электронной почты — МТА-агентами (МТА, Mail Transfer Agent). Поскольку в системе может быть только один МТА-агент, установить можно или `sendmail`, или `postfix`, т. е. пакет `sendmail` конфликтует с пакетом `postfix` и наоборот.

Некоторая информация о содержащейся в пакете программе, как правило, содержится в самом имени пакета. Сделано это исключительно для удобства — взглянув на название пакета, можно узнать версию программы и еще кое-какую информацию о ней, например:

```
program-1.5-14.i586.rpm
```

Здесь `program` — название программы, `1.5` — ее версия, `14` — выпуск пакета, `i586` — архитектура процессора, на которую рассчитана программа. Если программа независима от архитектуры, то указывается параметр `noarch`, — обычно так делают для документации, примеров конфигурационных файлов, т. е. для пакетов, содержащих информацию, которая не зависит от архитектуры.

7.2. Репозитории пакетов

Репозиторий — это хранилище пакетов. Репозиторий может быть локальным, например, каталогом на жестком диске или на DVD, или же сетевым — сервером в Интернете или в локальной сети, содержащем RPM- или DEB-пакеты. Для чего создаются репозитории? Для централизованного управления обновлением пакетов. Представьте, что у нас нет репозиториев. Тогда, чтобы узнать, вышла ли новая версия нужной вам программы, вам пришлось бы посещать сайт ее разработчика или, по крайней мере, сайт разработчика дистрибутива Linux. А это не очень удобно. Один раз вы можете забыть проверить наличие обновлений, а потом вам вообще надоест это делать. Проще дожидаться выхода новой версии дистрибутива и обновить все программы за один раз.

Так раньше и было. Вот вышла программа, ее включили в состав дистрибутива, но полностью не протестировали (да и невозможно предварительно протестировать все варианты использования любой новой программы). И в процессе эксплуатации программы выяснилось, что она работает неправильно, но только при некоторых условиях, — например, с определенным форматом файла. Или же на платформе Linux был организован, например, Web-сервер. А через некоторое время оказалось, что в этой версии Web-сервера имеется «дыра», поэтому разработчики вскоре выпустили новую ее версию, эту «дыру» закрывающую. Пользователь же, установивший программу Web-сервера, ничего не подозревая о том, что вышла новая ее версия, находился бы под угрозой взлома минимум полгода

или даже год — до выхода следующей версии дистрибутива. А его сервер могли бы взломать уже на следующий день после обнаружения «дыры».

Но не тут-то было — разработчики Linux, заботясь о нас с вами, создали репозитории, с помощью которых можно быстро и удобно отслеживать обновления тех или иных пакетов. Причем это делает в автоматическом режиме сам менеджер пакетов, а вам остается лишь указать, какие обновления нужно загружать, а какие — нет. Практически все системы управления пакетами современных дистрибутивов поддерживают работу с репозиториями.

7.3. Программы для управления пакетами

Для управления пакетами в разных дистрибутивах используются разные программы. В табл. 7.1 приведены программы управления пакетами, которые можно встретить в современных дистрибутивах.

Таблица 7.1. Программы управления пакетами

Программа	Дистрибутив	Описание
rpm	Red Hat-совместимые дистрибутивы (Fedora, ALT Linux, openSUSE и др.)	Простой менеджер пакетов. Работает в текстовом режиме. Не умеет разрешать зависимости пакетов
urpmi	Mageia	Текстовый менеджер пакетов, поддерживающий источники пакетов и автоматически разрешающий зависимости
dpkg	Дистрибутивы, основанные на Debian (Ubuntu, Kubuntu и др.)	Простой менеджер пакетов. Работает в текстовом режиме. Не умеет разрешать зависимости пакетов
apt	Debian, Ubuntu (и ее клоны), ALT Linux и др.	Мощный менеджер пакетов, работающий в текстовом режиме. Умеет разрешать зависимости пакетов и поддерживает репозитории (источники пакетов)
yum	Устаревшие версии Fedora (до версии 22) и дистрибутивы, основанные на нем	Мощный менеджер пакетов, работающий в текстовом режиме. Умеет разрешать зависимости пакетов и поддерживает репозитории (источники пакетов)
dnf	Современные версии Fedora (начиная с версии 22) и дистрибутивы, основанные на нем	Современный менеджер пакетов, пришел на смену yum. Умеет разрешать зависимости пакетов и поддерживает репозитории (источники пакетов)
gnome-software	Любые со средой GNOME 3	Центр приложений GNOME 3. Может использоваться для установки, удаления и обновления приложений
pkgtool	Slackware	Менеджер пакетов Slackware, заслуживающий отдельного разговора
zypper	openSUSE	Менеджер пакетов SUSE. Работает в текстовом режиме. Умеет разрешать зависимости пакетов

РАЗРЕШЕНИЕ ЗАВИСИМОСТЕЙ

Наверное, вы обратили внимание на фразу в таблице «умеет разрешать зависимости пакетов». Это означает следующее: если при установке пакета будет обнаружено, что для корректной его установки ему нужны дополнительные пакеты, то менеджер пакетов установит их. Если же менеджер пакетов не умеет разрешать зависимости, то он лишь сообщит, что установить пакет невозможно, и выведет список файлов (файлов, а не пакетов!), которые нужны для установки этого пакета. А уж какой файл в каком пакете находится, вам придется догадываться самостоятельно.

7.4. Программа rpm (все Red Hat-совместимые дистрибутивы)

Если вы хотите установить пакет, который *не входит* в состав дистрибутива (например, загруженный из Интернета), вам следует использовать программу rpm.

ГРАФИЧЕСКИЙ МЕНЕДЖЕР ПАКЕТОВ RPM DRAKE

Для установки пакетов, которые *входят* в состав дистрибутива, намного удобнее использовать графический менеджер пакетов rpm-drake.

Программа rpm — полноценный текстовый менеджер пакетов, позволяющий устанавливать, удалять пакеты, просматривать информацию об уже установленных и новых пакетах, обновлять пакеты.

Чтобы установить пакет с помощью rpm, выполните команду:

```
# rpm -ihv <имя_пакета>
```

Удалить пакет так же просто:

```
# rpm -e <имя_пакета>
```

Для обновления пакета служит команда:

```
# rpm -U <имя_пакета>
```

Просмотреть, установлен ли тот или иной пакет, можно с помощью команды:

```
# rpm -qa | grep <имя_пакета>
```

Если вы хотите просмотреть информацию о пакете, то введите команду:

```
# rpm -qi <имя_пакета>
```

Просмотреть список файлов, входящих в состав пакета, можно командой:

```
# rpm -ql <имя_пакета>
```

Наконец, вывести все пакеты можно командой:

```
$ rpm -qa | grep more
```

СБОРКА СОБСТВЕННЫХ ПАКЕТОВ

Программа rpm может также использоваться и для сборки собственных пакетов, но рассмотрение такой процедуры выходит за рамки этой книги. Мою статью о сборке собственных RPM-пакетов вы найдете по адресу:

<http://www.dkws.org.ua/article.php?id=58>.

7.5. Программа urpmi

Программа urpmi представляет собой систему управления пакетами, использующуюся в Mageia (ранее — в Mandriva). Как уже было отмечено в табл. 7.1, urpmi поддерживает зависимости пакетов.

Не нужно расценивать urpmi как замену rpm — система urpmi просто делает управление пакетами проще (хотя желающие могут использовать утилиту rpm, если сочтут ее более удобной).

ЛОКАЛЬНАЯ УСТАНОВКА ПАКЕТОВ

Я, например, предпочитаю для локальной установки пакетов (когда пакет из какого-либо источника уже закачан на мой компьютер) использовать rpm.

7.5.1. Установка пакетов

Для установки пакета служит команда:

```
# urpmi <имя пакета>
```

Так, чтобы установить пакет tc (файловый менеджер Midnight Commander), следует ввести команду:

```
# urpmi tc
```

Программа просматривает список источников пакетов, хранящийся в файле `/etc/urpmi/urpmi.conf`. Если она находит пакет в одном из источников, то устанавливает его вместе со всеми необходимыми для его работы пакетами (при этом urpmi автоматически разрешает зависимости пакетов).

Существуют три вида репозиториев, поддерживаемых urpmi:

- хранилища на съемных носителях (removable) — репозитории на компакт-дисках, DVD, ZIP-носителях, флеш-дисках и т. д.;
- локальные (local) — находятся в каталоге на жестком диске;
- удаленные (distant server) — пакеты находятся на удаленном FTP- или HTTP-сервере.

Просмотреть список источников пакетов можно с помощью команды:

```
# urpmq --list-media
```

Добавить источники пакетов можно с помощью команды:

```
# urpmi.addmedia <источник>
```

Список источников по умолчанию обычно записан в файл `urpmi.conf` и редко требует редактирования — тогда, например, когда вы хотите добавить сторонние репозитории.

7.5.2. Обновление и удаление пакетов

Для удаления пакета нужно ввести команду:

```
# urpme <пакет>
```

Если пакет нужен для работы других пакетов, то программа спросит у вас, хотите ли вы удалить и эти пакеты, иначе придется отказаться от удаления выбранного пакета.

Для обновления всей системы, т. е. получения списка новых версий пакетов, используется команда:

```
# urpmi --auto-select
```

7.5.3. Поиск пакета.

Получение информации о пакете

Найти пакеты, содержащие в названии определенную строку, можно с помощью команды:

```
# urpmq <строка>
```

Команда `urpmf` позволяет получить различную информацию о пакете, например:

- `urpmf <файл>` — выводит пакеты, содержащие указанный файл;
- `urpmf --group <группа>` — выводит пакеты, входящие в указанную группу;
- `urpmf --size <пакет>` — выводит размер указанного пакета;
- `urpmf --summary <пакет>` — **выводит общую информацию о пакете.**

7.6. Программа yum

Программа `yum` (Yellow dog Updater Modified) используется во многих дистрибутивах, в том числе в CentOS и ранних версиях Fedora (в последние версии Fedora включен менеджер пакетов `dnf`, который будет рассмотрен далее).

`Yum` работает аналогично другим подобным программам (`urpmi`, `apt`)— когда вы устанавливаете пакет, `yum` производит поиск пакета в репозиториях, перечисленных в конфигурационном файле, загружает пакет и устанавливает его. В качестве репозитория могут выступать как дистрибутивные диски, так и серверы Интернета.

7.6.1. Использование yum

Общий формат вызова `yum` выглядит так:

```
yum команда [пакет(ы)]
```

Команды `yum` приведены в табл. 7.2.

Таблица 7.2. Использование *yum*

Команда	Описание
<code>yum install пакет</code>	Установить пакет из репозитория (также устанавливаются пакеты, необходимые для работы устанавливаемого пакета, т. е. разрешаются зависимости)
<code>yum remove пакет</code>	Удалить пакет, а также все пакеты, которые зависят от него
<code>yum update</code>	Проверить наличие обновлений всех пакетов. Если обновления есть, то они будут установлены
<code>yum update пакет</code>	Проверить обновления конкретного пакета. Если есть свежая версия, то она будет установлена
<code>yum check-update</code>	Только проверить наличие обновлений (обновления не устанавливаются)
<code>yum check-update пакет</code>	Проверить наличие обновлений конкретного пакета (обновления не устанавливаются)
<code>yum info пакет</code>	Вывести информацию о пакете
<code>yum list</code>	Вывести список всех пакетов: как установленных, так и доступных для установки из репозитория
<code>yum list a*</code>	Вывести список всех пакетов, которые начинаются на букву «а»
<code>yum search строка</code>	Найти все пакеты, в описаниях которых есть указанная строка
<code>yum groupinstall "группа"</code>	Установить все пакеты из указанной группы
<code>yum grouplist</code>	Вывести список групп пакетов

При установке пакетов с помощью *yum* не следует далеко отходить от компьютера — часто нужные пакеты находятся не на локальных источниках, а на серверах в Интернете, поэтому *yum* выведет общий объем пакетов, которые вы хотите установить, и спросит вас, хотите ли вы их установить или нет:

Total download size: 10.5 M

It this ok [Y/N]:

Если вы согласны для установки выбранных пакетов загрузить 10,5 Мбайт файлов, нажмите клавишу <Y>, если передумали — нажмите <N>. Довольно-таки удобно, иначе (с учетом того, что при разрешении зависимостей будут установлены дополнительные пакеты) можно при установке одного небольшого, на первый взгляд, пакета превысить месячную норму по трафику.

Получить информацию о пакете, как было показано в табл. 7.2, можно с помощью команды:

```
yum info пакет
```

При этом на экран выводится следующая информация (рис. 7.1):

- ☐ **Название (Name)** — имя пакета;
- ☐ **Архитектура (Arch)** — архитектура компьютера;

```

den@localhost:~$ yum info mc
Заргружены модули: langpacks, presto, refresh-packagekit
fedora/metalink | 31 kB | 00:00
fedora | 4.2 kB | 00:00
fedora/primary_db | 12 MB | 00:06
fedora/group | 1.9 MB | 00:00
updates/metalink | 22 kB | 00:00
updates | 4.7 kB | 00:00
updates/primary_db | 1.8 MB | 00:00
updates/group | 1.9 MB | 00:00
Установленные пакеты
Название: mc
Архитектура: i686
Период: 1
Версия: 4.8.0
Выпуск: 2.fc16
Объем: 5.3 М
Источник: installed
Из источника: fedora
Аннотация: User-friendly text console file manager and visual shell
Ссылка: http://www.midnight-commander.org/
Лицензия: GPLv3+
Описание: Midnight Commander is a visual shell much like a file manager, only
: with many more features. It is a text mode application, but it also
: includes mouse support. Midnight Commander's best features are its
: ability to FTP, view tar and zip files, and to poke into RPMs for
: specific files.
[den@localhost ~]$

```

Рис. 7.1. Вывод информации о пакете

- **Период (Epoch)** — как бы подверсия пакета, поле **Epoch** используется, когда требуется уменьшить версию или релиз пакета по сравнению с имеющимся в репозитории;
- **Версия (Version)** — версия пакета;
- **Выпуск (Release)** — релиз пакета (можете считать это подверсией пакета);
- **Объем (Size)** — размер занимаемого места на диске;
- **Источник (Repo)** — хранилище пакета или значение **installed**, если пакет уже установлен;
- **Из источника (From repo)** — хранилище, из которого был установлен пакет (только для установленных пакетов);
- **Аннотация (Summary)** — общая информация о пакете;
- **Ссылка (URL)** — Web-страничка разработчика программы;
- **Лицензия (License)** — лицензия, по которой распространяется программа;
- **Описание (Description)** — описание пакета.

Для вывода информации обо всех пакетах можно использовать команду `yum list`, но пакетов слишком много, и такой вывод может оказаться слишком громоздким.

Удобнее задать маску имени пакета— например, `yum list a*`— в этом случае будут выведены все пакеты, начинающиеся на букву «а».

7.6.2. Управление источниками пакетов

Источники пакетов `yum` описываются в файле конфигурации `/etc/yum.conf`. Откройте этот файл (листинг 7.1).

РЕДАКТИРОВАНИЕ ФАЙЛА /ETC/YUM.CONF

Обычно файл `/etc/yum.conf` приходится редактировать редко. Но помните, что делать это можно только от имени пользователя `root`. Если вы привыкли к графическому режиму, тогда в терминале для редактирования этого файла нужно ввести команду:

```
su -c <редактор> /etc/yum.conf
```

В качестве редактора могут выступать программы `gedit` (если у вас GNOME), `kwrite` или `kate` (если у вас KDE). Если открыть этот файл в редакторе без прав `root`, то просмотреть его вы сможете, но сохранить изменения не удастся.

Листинг 7.1. Конфигурационный файл yum.conf

```
[main]
cachedir=/var/cache/yum/$basearch/$releasever
keepcache=0
debuglevel=2
logfile=/var/log/yum.log
exactarch=1
obsoletes=1
gpgcheck=1
plugins=1
installonly_limit=3
# PUT YOUR REPOS HERE OR IN separate files named file.repo
# in /etc/yum.repos.d
```

Ранее репозитории описывались непосредственно в файле `yum.conf` (как и в случае с `urpmi.conf`), но потом было принято решение хранить описания репозиториев в отдельных REPO-файлах в каталоге `/etc/yum.repos.d`. Каждый файл в этом каталоге называется так: `<имя репозитория>.repo`.

В листинге 7.2 приведен пример описания источника пакетов Fedora (версия до 22), взятый из файла `fedora.repo`.

Листинг 7.2. Пример описания источника пакетов

```
[fedora]
name=Fedora $releasever - $basearch
baseurl=http://download.fedora.redhat.com/pub/fedora/linus/releases/$releasever/Everything/$basearch/os/
mirrorlist=http://mirrors.fedoraproject.org/mirrorlist?repo=fedora-$releasever&arch=$basearch
```



```
enabled=1
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-fedora file:///etc/pki/rpm-gpg/RPM-
GPG-KEY
```

В квадратных скобках здесь указывается сокращенное имя репозитория. Параметр `name` задает полное имя источника пакетов. Интернет-адрес (URL) источника пакетов указан параметром `baseuri`, а параметр `mirrorlist` определяет список зеркал — копий репозитория, которые будут использоваться, если URL источника, указанный в `baseuri`, окажется недоступен. Параметр `enabled`, установленный в 1, указывает на то, что этот источник активен, и `yum` использует его при установке пакетов. Следующий параметр: `gpgcheck` — обязывает `yum` проверить подпись источника (если `gpgcheck=1`), а ключ для проверки подписи задан параметром `gpgkey`.

Добавление источника производится путем размещения соответствующего ему REPO-файла в каталоге `/etc/yum.repos.d`. Где этот файл взять? Обычно они представлены в виде RPM-пакетов на Web-серверах репозитория, поэтому надо просто скачать такой RPM-пакет и установить его.

Например, для установки REPO-файла популярного репозитория RPM Fusion нужно выполнить команду:

```
su -c 'rpm -Uvh http://downloadl.rpmfusion.org/free/fedora/
rpmfusion-free-release-stable.noarch.rpm
http://downloadl.rpmfusion.org/nonfree/fedora/
rpmfusion-nonfree-release-stable.noarch.rpm'
```

Что делает эта команда, ясно и без комментариев. Если вы не можете найти соответствующий источнику REPO-файл, его можно написать вручную по формату листинга 7.2. При этом нужно еще знать базовый URL источника пакетов.

Удалять файлы источников пакетов, если сам источник уже не нужен, совсем не обязательно. Достаточно установить параметр `enabled` для источника в 0, и этот источник использоваться не будет.

7.6.3. Установка пакетов через прокси-сервер

По умолчанию `yum` полагает, что наш компьютер напрямую подключен к Интернету (не через прокси-сервер). Если вы подключаетесь к Интернету по локальной сети, т. е. через прокси-сервер, этот факт нужно отразить в файле `yum.conf`, иначе вы не сможете устанавливать пакеты.

Узнайте у администратора сети параметры подключения к прокси-серверу (адрес, порт, имя пользователя и пароль) и пропишите их в файле `yum.conf` таким вот образом:

```
# Адрес прокси и его порт
proxy=http://proxy.company.ru:8080
# Имя пользователя и его пароль
proxy_use rname=dhsilabs
proxy_password=secret
```

7.6.4. Плагины для yum

Для yum доступно множество плагинов. Мы установим два: `fastestmirror` и `presto`. Первый плагин позволяет найти самый быстрый источник пакетов, что существенно сокращает время установки пакетов. А второй — пытается загружать только обновленные части пакетов вместо полной загрузки пакетов при обновлении, что сокращает трафик и уменьшает время обновления.

Для установки этих плагинов введите команды:

```
# yum install yum-plugin-fastestmirror
# yum install yum-presto
```

УСТАНОВКА ПАКЕТОВ С ДИСТРИБУТИВНОГО ДИСКА

Ранее в моих книгах (как правило, изданных до 2010 года) описывалось, как заставить менеджер пакетов Fedora устанавливать пакеты с дистрибутивного диска. В этой книге подобного материала для Fedora не предусмотрено. Во-первых, скорость Интернета выросла, стоимость снизилась, и высокоскоростной Интернет теперь доступен почти каждому. А при установке пакетов из Интернета у вас будут всегда самые новые версии пакетов. Во-вторых, мы только что установили два плагина, уменьшающих время загрузки пакетов и экономящих ваш трафик, — поэтому я не вижу более смысла использовать устаревшие пакеты с установочного DVD. Если же у вас медленное соединение или вы принципиально желаете устанавливать пакеты с установочного диска, а не из интернет-репозитория, тогда посетите следующую страничку: <http://www.dkws.org.ua/phpbb2/viewtopic.php?p=23984>. Там хотя и описывается настройка менеджера пакетов Fedora 9, вам не составит большого труда настроить «по образу и подобию» этого описания и более свежие версии Fedora.

7.7. Менеджер пакетов dnf

Менеджер пакетов `dnf` пришел в дистрибутиве Fedora на смену `yum`. Впервые экспериментальная версия `dnf` появилась в Fedora 18, а, начиная с Fedora 22, менеджер `dnf` стал использоваться по умолчанию, хотя все еще есть возможность установить `yum`.

По сравнению с `yum`, новый менеджер более быстрый, обладает низким потреблением памяти и эффективнее управляет зависимостями.

Основные команды менеджера `dnf` (`install`, `remove`, `upgrade`, `info`, `search` и т. д.) такие же, что и в менеджере пакетов `yum`, поэтому при переходе с `yum` на `dnf` вы не ощутите никакого дискомфорта. Основным конфигурационным файлом менеджера является файл `/etc/dnf/dnf.conf`. Его синтаксис полностью повторяет синтаксис файла `yum.conf`.

Каковы же отличия `dnf` от `yum` с точки зрения пользователя? Во-первых, команды `upgrade` и `update` теперь идентичны. Во-вторых, опция `-skip-broken` теперь не поддерживается, как и команды `resolvedep` и `deplist`, вместо которых следует использовать `dnf provides` и `dnf repoquery --requires`.

Некоторое время в Fedora все еще будет поддерживаться `yum`, хотя при вводе команды `yum` выводится предупреждение о том, что эта команда устарела (рис. 7.2), и вместо нее рекомендуется использовать `dnf`.

```

mc [root@localhost.localdomain]:/etc
Файл  Правка  Вид  Поиск  Терминал  Справка
[root@localhost etc]# yum
Yum command has been deprecated, redirecting to '/usr/bin/dnf '.
See 'man dnf' and 'man yum2dnf' for more information.
To transfer transaction metadata from yum to DNF, run:
'dnf install python-dnf-plugins-extras-migrate && dnf-2 migrate'

Необходимо задать команду
usage: dnf [options] COMMAND

Список основных команд

autoremove
check-update      Проверить доступные обновления для пакетов
clean             Удаление кэшированных данных
distro-sync       Обновить установленные пакеты до новейших доступных ве
рсий
downgrade         откат к предыдущей версии пакета
group             Display, or use, the groups information
help             Отобразить подсказку к использованию
history           Отобразить (или использовать) журнал операций
info             Отобразить информацию о пакете или о коллекции пакетов
install           Установка пакета(ов) в систему
list             Вывести список пакетов или коллекций пакетов
makecache         Создание кэша метаданных

```

Рис. 7.2. Fedora 22-26: команда yum все еще поддерживается, хотя является просто псевдонимом для dnf

```

den@localhost:/home/den
Файл  Правка  Вид  Поиск  Терминал  Справка
sysctl      systemd-hwdb
syslinux    systemd-inhibit
system-config-abrt  systemd-machine-id-setup
systemctl   systemd-notify
systemd-analyze  systemd-nspawn
systemd-ask-password  systemd-path
systemd-cat   systemd-run
systemd-cgls  systemd-stdio-bridge
systemd-cgtop  systemd-sysusers
systemd-delta  systemd-tmpfiles
systemd-detect-virt  systemd-tty-ask-password-agent
systemd-escape  sys-unconfig
systemd-firstboot
[den@localhost ~]$ su
Пароль:
[root@localhost den]# mc
bash: mc: команда не найдена...
Установить пакет 'mc' предоставляющий команду 'mc'? [N/y] y

* Ожидание в очереди...
Следующие пакеты должны быть установлены:
gpm-libs-1.20.7-6.fc22.i686  Dynamic library for for the gpm
Продолжать с изменениями? [N/y]

```

Рис. 7.3. Fedora 26: предложение установить необходимый пакет

Вот, нравится мне в Fedora 22-26, что дистрибутив сам предлагает установить недостающие пакеты. Например, вы вводите команду `mc`, но пакет этой программы в системе не установлен, — в таком случае вы получите предложение установить необходимый пакет (рис. 7.3).

Также в Fedora 22-26 (и в других дистрибутивах на базе GNOME 3) можно задействовать Центр приложений (команда `gnome-software`), изображенный на рис. 7.4. Использовать эту утилиту очень просто, поэтому с ней вы сможете разобраться самостоятельно.

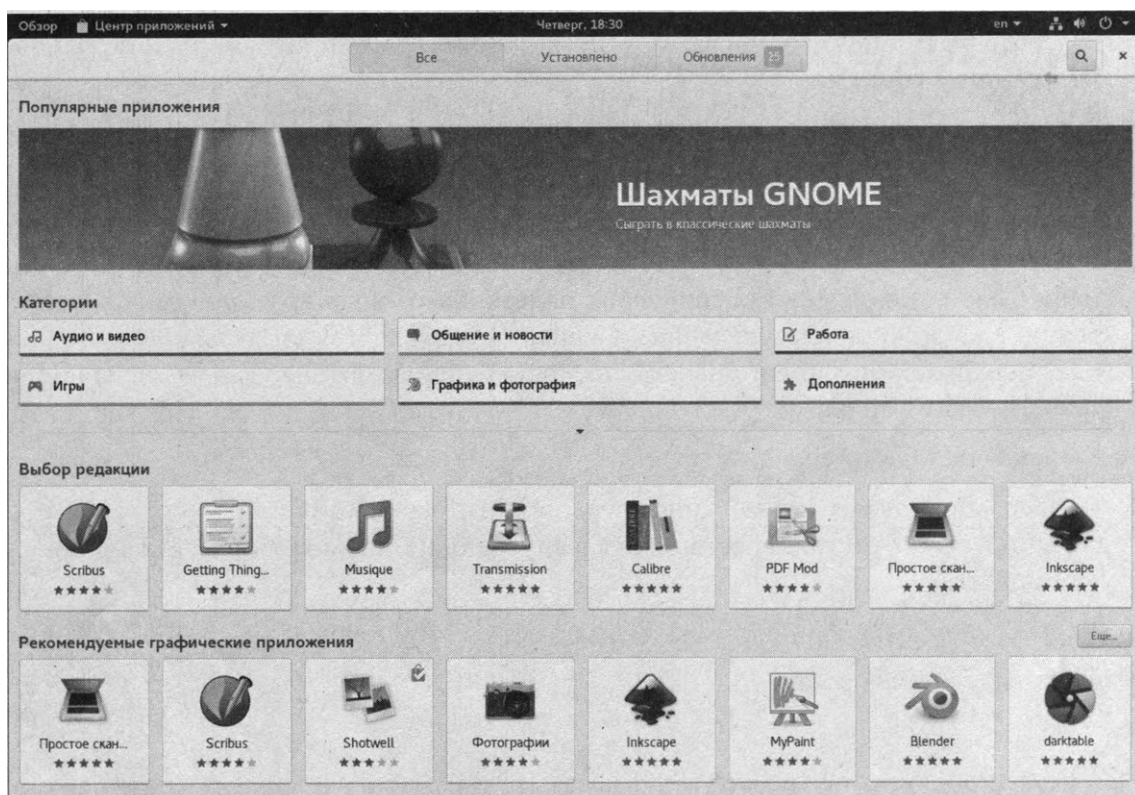


Рис. 7.4. Fedora 26: Центр приложений GNOME 3

7.8. Программы `dpkg` и `apt-get`: установка пакетов в Debian/Ubuntu

7.8.1. Программа `dpkg`

Программа `dpkg` используется для установки, удаления и управления пакетами Debian/Ubuntu и вызывается из командной строки. Формат ее вызова следующий:

```
dpkg [ключи] действие
```

Для запуска `dpkg` нужно обладать полномочиями `root`, получить которые можно с помощью команды `sudo`. Рассмотрим, как правильно работать с программой `dpkg`.

Предположим, у нас есть пакет `package.deb`. Для его установки откройте **Терминал (Приложения | Стандартные | Терминал)** и введите команду:

```
sudo dpkg -i /<путь>/package.deb
```

Как видите, в установке пакета нет ничего сложного. Процесс установки состоит из следующих шагов:

1. Из пакета извлекаются управляющие файлы.
2. Если уже была установлена старая версия этого пакета, тогда из старого пакета запускается сценарий `prepm` — он подготавливает систему к удалению старой версии пакета. Другими словами, если требуется, то обновление пакета выполняется автоматически.
3. Выполняется сценарий `preinst`, если он есть в этом пакете.
4. Из пакета распаковываются остальные файлы. Если был установлен старый пакет, то его файлы не удаляются, а сохраняются в другом месте, чтобы их можно было восстановить, если что-то пойдет не так.
5. Если была установлена старая версия пакета, то из него выполняется сценарий `postrm` (действия после удаления). Сценарий запускается сразу после выполнения сценария `preinst` нового пакета, поскольку старые файлы удаляются во время записи новых файлов.
6. Выполняется настройка пакета:
 - распаковываются новые конфигурационные файлы, а старые сохраняются, если нужно будет их восстановить в случае ошибки во время установки нового пакета;
 - запускается сценарий `postinst`, если он есть в этом пакете.

Удалить пакет тоже просто:

```
sudo dpkg -r <package>
```

При удалении пакета не требуется указывать путь к пакету и «расширение» пакета, т. е. символы `.deb` в конце имени файла.

Однако установка и удаление пакетов — это далеко не все, что можно выполнить с помощью программы `dpkg`. Другие действия программы `dpkg`, которые могут быть интересны каждому пользователю Ubuntu, представлены в табл. 7.3.

Таблица 7.3. Вспомогательные действия программы `dpkg`

Ключ	Описание
<code>-l [образец]</code>	Выводит все установленные пакеты, имена которых соответствуют образцу. Образец задается с помощью масок <code>*</code> и <code>?</code> — например, образец <code>a*</code> соответствует любому имени пакета, начинающемуся на букву «а». Если образец не задан, выводятся все пакеты
<code>-l <имя пакета></code>	Выводит имена файлов из указанного пакета (пакет должен быть установлен)

Таблица 7.3 (окончание)

Ключ	Описание
-р <имя пакета>	Выводит информацию об установленном пакете
-s <имя пакета>	Выводит информацию о статусе пакета
--unpack <имя пакета.deb>	Распаковывает, но не устанавливает пакет (полезно, если устанавливать пакет не требуется, а нужно лишь достать из него один или несколько файлов)

Если вы хотите получить более подробную информацию о программе `dpkg`, введите команду: `man dpkg` — страница руководства будет выведена на русском языке.

7.8.2. Программа apt-get (apt)

Программа `apt-get` используется не только в Debian/Ubuntu, но и в других дистрибутивах, причем даже в Red Hat-совместимых (например, в ALT Linux), но там с ее помощью устанавливаются RPM-пакеты. Вообще, выбор менеджера пакетов зависит от разработчиков дистрибутива. В одной версии дистрибутива может использоваться `apt-get`, в другой — `yum`, а в третьей — какой-то новый и перспективный менеджер пакетов.

КОМАНДА АРТ

В современных версиях дистрибутивов Debian/Ubuntu предлагается вводить команду `apt`, а не `apt-get`. Это один и тот же менеджер пакетов, а какую команду вводить — дело вкуса. Если вы не занимаетесь поддержкой разношерстных серверов на базе Debian, то можете вводить команду `apt` — так проще. Команду `apt-get` рекомендуется использовать из соображений обратной совместимости — несколько устаревшие дистрибутивы не знают команды `apt`.

Итак, предположим, что мы устанавливаем пакет `package.deb`. Но в процессе установки обнаружилось, что он требует пакет `lib.deb`, который в системе не установлен. Что ж, вы находите в Интернете недостающий пакет `lib.deb`, устанавливаете его, а затем заново устанавливаете пакет `package.deb`. Не очень удобно, правда?

Намного проще выполнить команду:

```
sudo apt-get install package
```

Программа `apt-get` просматривает файл `/etc/apt/sources.list` — в этом файле перечислены источники (репозитории) DEB-пакетов, в качестве которых может выступать как компакт-диск, содержащий пакеты, так и сервер в Интернете. Программа находит указанный пакет, читает служебную информацию о нем, затем разрешает зависимости (т. е. устанавливает все другие пакеты, необходимые для работы программ устанавливаемого пакета), а затем устанавливает нужный нам пакет. Все загруженные программой `apt-get` и менеджером Synaptic (о нем — далее) пакеты записываются в каталог `/var/cache/apt/archives`.

Чтобы просмотреть содержимое файла `/etc/apt/sources.list`, можно выполнить следующую команду:

```
sudo gedit /etc/apt/sources.list
```

СТАНДАРТНЫЕ ТЕКСТОВЫЕ РЕДАКТОРЫ

В Ubuntu стандартным текстовым редактором является gedit. В Kubuntu его нет, поэтому для правки файла там следует использовать текстовый редактор Kate. А в Xubuntu в качестве текстового редактора служит mousepad.

В репозиториях Ubuntu программы распределены особым образом. Так, в репозитории main включены основные программы, они распространяются свободно и регулярно поддерживаются (обновляются). В репозитории restricted содержатся программы, распространяемые по несвободным лицензиям, а также имеющие ограниченную поддержку. Репозиторий universe содержит программы с открытыми лицензиями — поддержка программ из этого репозитория не гарантируется, но вполне возможна, все зависит от разработчика программы. В репозитории multiverse содержатся программы, распространяемые несвободно и безо всякой поддержки и гарантий. Репозиторий security содержит исправления пакетов из репозитория main и restricted. Наконец, в репозитории backports содержатся неофициальные пакеты свежих версий программ, собранные из исходных текстов энтузиастами Ubuntu (а не разработчиками программ).

Чтобы настроить менеджер пакетов на российские репозитории (соответственно скорость загрузки пакетов будет выше), замените во всех строках файла `/etc/apt/sources.list` адрес `archive.ubuntu.com` на `ru.archive.ubuntu.com`.

Понятно, что программа `apt-get` может использоваться не только для установки пакетов. Общий формат вызова этой программы следующий:

```
apt-get [опции] команды [пакет]
```

Основные команды `apt-get` представлены в табл. 7.4.

Таблица 7.4. Основные команды `apt-get`

Команда	Описание
<code>update</code>	Синхронизирует файлы описаний пакетов (внутреннюю базу данных о пакетах) с источниками пакетов, которые указаны в файле <code>/etc/apt/sources.list</code>
<code>upgrade</code>	Обновляет указанный пакет. Может использоваться для обновления всех установленных пакетов. При этом установка новых пакетов не производится, а загружаются и устанавливаются только новые версии уже установленных пакетов
<code>dist-upgrade</code>	Обновляет дистрибутив. Для обновления всех пакетов рекомендуется использовать именно эту команду
<code>install</code>	Устанавливает один или несколько пакетов
<code>remove</code>	Удаляет один или несколько пакетов
<code>check</code>	Служит для поиска нарушенных зависимостей
<code>clean</code>	Используется для очистки локального хранилища полученных пакетов (перед установкой пакет загружается в локальное хранилище, а затем устанавливается оттуда. Эта команда может очистить хранилище для экономии дискового пространства)

7.8.3. Установка RPM-пакетов в Debian/Ubuntu

Всегда может случиться так, что нужная вам программа найдется только в виде RPM-файла. Что ж, файл формата RPM можно преобразовать в формат DEB с помощью команды `alien`. Сразу хочу заметить, что установка таких — преобразованных — пакетов не желательна, поскольку нет никакой гарантии, что установленная программа будет работать, но если другого выхода нет, можно попробовать:

```
sudo alien package_file.rpm
```

Если система сообщит вам, что команда `alien` не найдена, тогда нужно подключиться к Интернету и установить ее с помощью команды:

```
sudo apt-get install alien
```

7.8.4. Подключение репозитория Medibuntu

В предыдущих изданиях этой книги было показано, как подключить репозиторий Medibuntu, содержащий кодеки и другие «полезности». В настоящее время этот репозиторий закрыт, а все пакеты, содержащиеся в нем, «перекочевали» в официальный репозиторий Ubuntu. Поэтому в подключении Medibuntu уже нет смысла.

7.8.5. Графические менеджеры в Debian/Ubuntu

Дистрибутивы Debian/Ubuntu содержат удобный графический менеджер пакетов Synaptic (рис. 7.5). Правда, не во всех версиях Debian/Ubuntu он установлен по умолчанию. Попробуйте ввести команду `synaptic` — если Synaptic не установлен, ничто не мешает вам его установить:

```
sudo apt-get install synaptic
```

Кроме Synaptic в Ubuntu доступен еще один графический менеджер пакетов — Muon (рис. 7.6). Получается, что вам здесь доступно как минимум три графических менеджера пакетов: Synaptic, Muon и Центр приложений Ubuntu (рис. 7.7). Почему «как минимум»? Честно говоря, может где-то в недрах репозитория Ubuntu вы найдете и дополнительные менеджеры пакетов. Я не искал, поскольку использую `apt-get` или, в крайнем случае, Synaptic.

Какой из менеджеров пакетов выбрать, если вам не нравится `apt-get`? Для самых начинающих пользователей подойдет Центр приложений Ubuntu. А если вы намерены получать больше информации о пакетах и эффективнее контролировать процесс их установки, тогда или Synaptic, или Muon. Последний — довольно-таки неплохой менеджер, но вот незадача — он является менеджером пакетов для KDE. И если вы не хотите, чтобы при его установке были загружены «тяжеловесные» библиотеки KDE, используйте Synaptic. Я же установил Muon из «академического» интереса — чтобы посмотреть на это чудо и рассказать о нем вам, читателю. Честно говоря, смотреть там не на что (и рис. 7.6 тому подтверждение). Установите Synaptic — он намного удобнее.

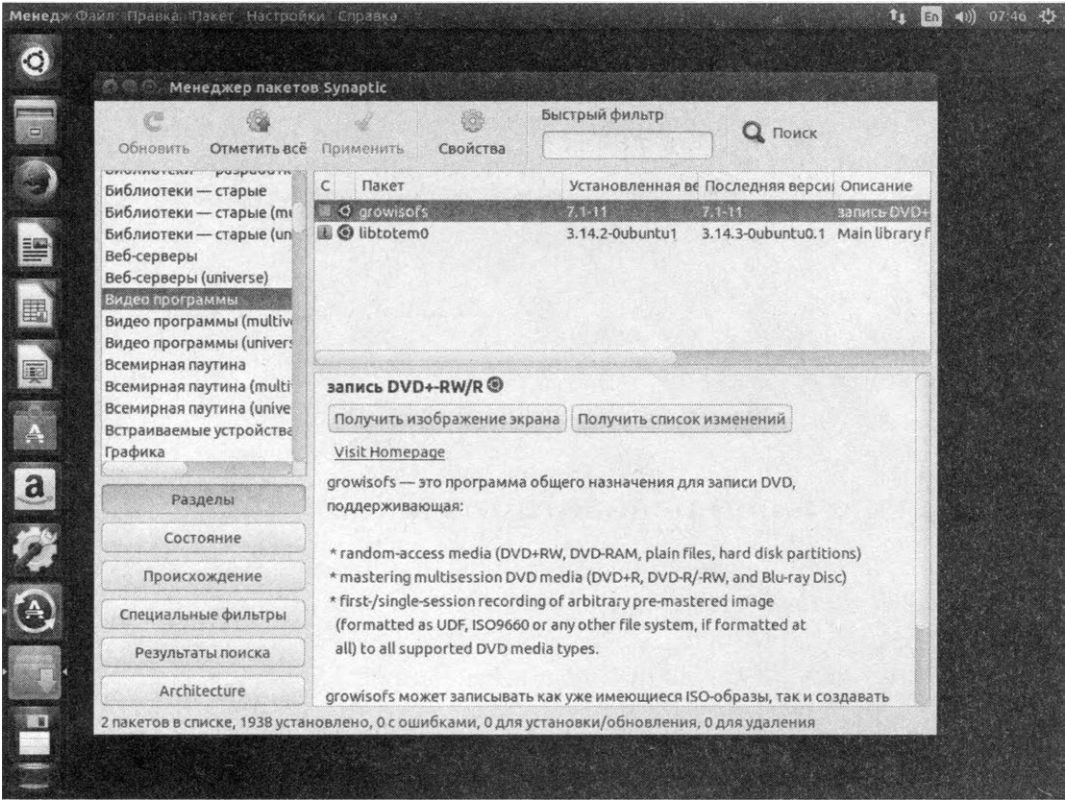


Рис. 7.5. Ubuntu 17.04: менеджер пакетов Synaptic

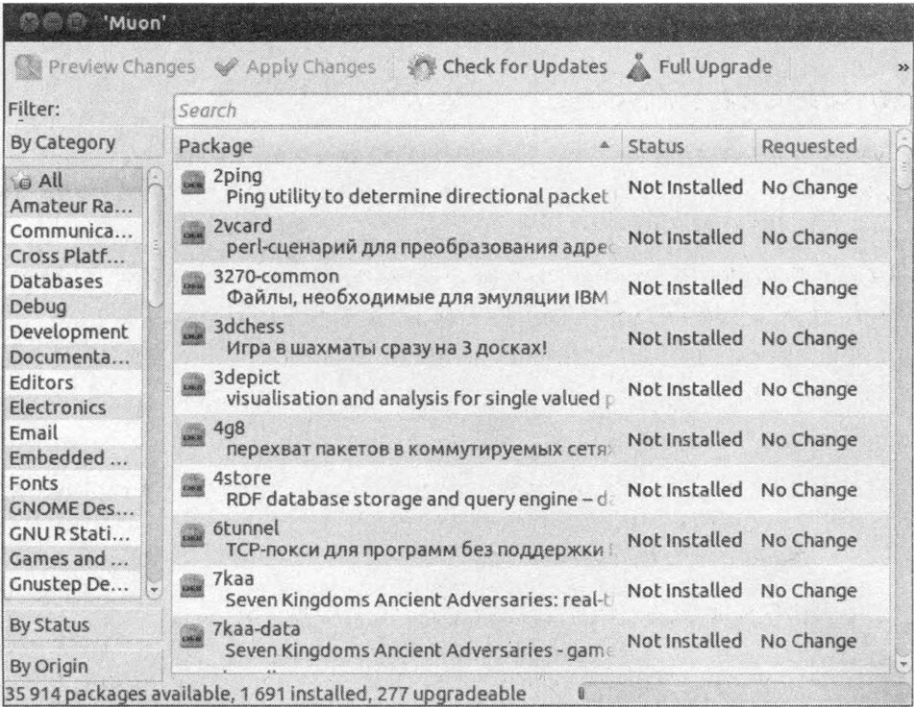


Рис. 7.6. Ubuntu 17.04: менеджер пакетов Muon

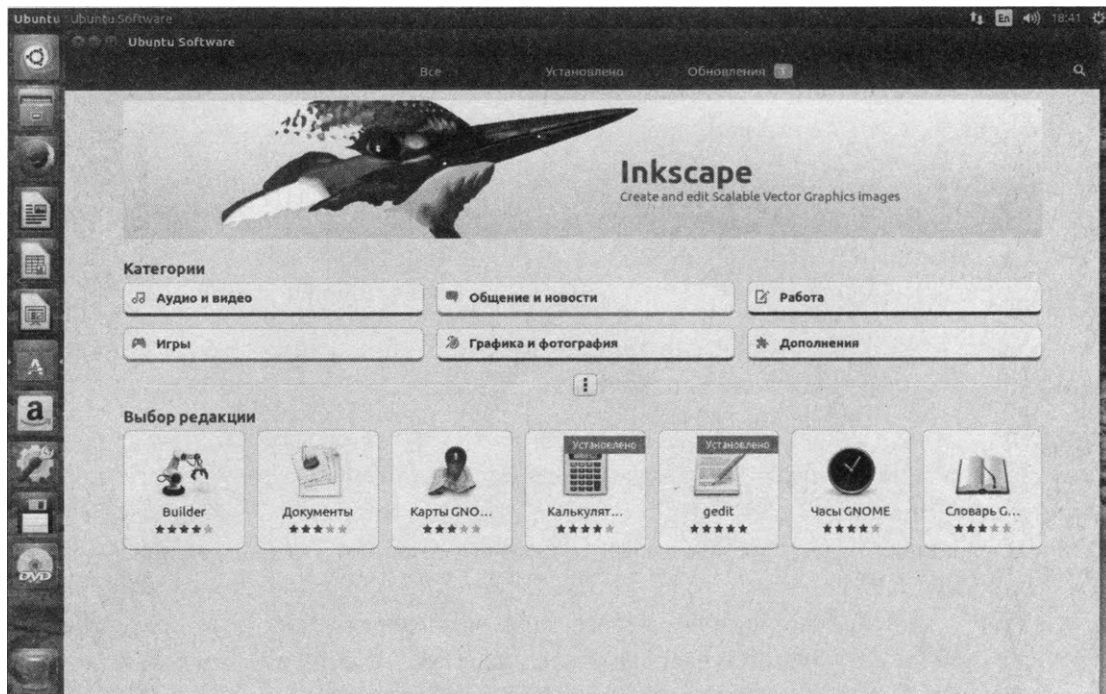


Рис. 7.7. Ubuntu 17.04: Центр приложений

На самом же деле Synaptic, Muon и другие подобные программы — просто оболочки для apt-get, но Synaptic — оболочка наиболее продуманная. Рассматривать Synaptic подробно мы здесь не станем — управляться с ним очень просто, и вы разберетесь с этим без моих комментариев.

7.8.6. Волшебная команда *update*

Ubuntu — уникальный дистрибутив. Еще вчера все прекрасно работало, а сегодня он не загружается. Или еще вчера я устанавливал пакеты, а сегодня они не устанавливаются, и я получаю сообщение:

Е: Невозможно получить некоторые архивы, вероятно надо запустить apt-get update или попытаться повторить запуск с ключом —fix-missing

Следуя этой рекомендации, при любых недоразумениях с установкой пакетов нужно использовать команду:

```
sudo apt-get update
```

И после ее выполнения большая часть ошибок, связанных с установкой пакетов в Ubuntu, будет устранена.

7.9. Установка пакетов в Slackware

Slackware в плане установки пакетов — весьма специфический дистрибутив. Мне частенько приходилось слышать мифы о сложности установки и управления пакетами в Slackware. Но все эти мифы, как оказалось, от незнания. Просто пользовате-

лям, привыкшим к Red Hat-совместимым дистрибутивам, трудно привыкнуть к особенностям Slackware. Возможно, «коренным» пользователям Slackware трудно привыкнуть к обращению с RPM-пакетами... Так утверждать не буду, потому что сам начинал свой путь линуксоида с дистрибутива Red Hat.

Но однажды я не выдержал и установил на свой компьютер Slackware. Цель была одна — разобраться с установкой пакетов. Неужели все так сложно? Как оказалось, ничего сложного нет, если разобраться в особенностях Slackware, не известных пользователям Red Hat.

Прежде чем приступить к рассмотрению системы управления пакетами, приведу ряд мифов, которые мне удалось разрушить:

- *в Slackware нет системы управления пакетами* — очевидно, этот миф сотворили пользователи, которые никогда не устанавливали Slackware, потому что такая система в Slackware есть. Другое дело, что она не поддерживает RPM/DEB-пакеты. Пакеты Slackware выполнены в виде обычных TGZ-архивов. Но и формат пакетов RPM — это тоже слегка модифицированный архивный формат, просто его называли иначе, в Slackware же используются обычные архивы. Хорошо это или плохо, решать вам. Но учитывая, что Slackware появился намного раньше, чем Red Hat с его системой RPM, использование архивов TGZ вполне закономерно;
- *в Slackware нет зависимостей пакетов* — это тоже миф, правда, с долей правды. Зависимости есть, но программы для установки пакетов их не обрабатывают — обработка зависимостей возложена на пользователя. Хорошо это или плохо? С одной стороны, есть вероятность недоустановить какой-то пакет или же удалить пакет, необходимый другим пакетам, что нарушит зависимости пакетов. Можно также установить пакет, который будет конфликтовать с уже установленными пакетами. Одним словом, при установке программного обеспечения в Slackware нужно четко себе представлять, что вы делаете, а то очень легко превратить свою систему в мусорку, для наведения полного порядка в которой поможет только переустановка системы. Если в дистрибутивах, основанных на RPM/DEB, можно положиться на менеджера пакетов, то в Slackware нужно рассчитывать только на себя, поэтому перед установкой пакета поможет прочтение соответствующей пакету документации. С другой стороны, пакеты в Slackware достаточно объемные и содержат практически все необходимое для работы конкретного программного продукта. Например, чтобы установить PHP в той же Mandriva, вам понадобился бы 21 пакет, причем каждый из них каким-то образом зависел от других пакетов группы. А вот для установки PHP в Slackware нужен всего один пакет, который включает все необходимое. Поэтому можно сказать, что разрешение зависимостей в Slackware совсем необязательно;
- *в Slackware отсутствует механизм обновления системы* — комментарии здесь примерно такие же, как и в предыдущем случае. Такой механизм есть, и его достаточно просто использовать, нужно только знать как;

- в *Slackware* неудобно устанавливать программы, не входящие в состав дистрибутива, — вот тут огромная доля правды. Можно даже сказать, что это не миф... С самой установкой ничего сложного нет, есть сложности с поиском необходимых пакетов. Но об этом мы поговорим чуть позже.

7.9.1. Управление пакетами

Для управления пакетами в *Slackware* используются четыре основные программы:

- `pkgtool` — псевдографический (использует текстовые меню) менеджер пакетов, позволяющий устанавливать, удалять и обновлять пакеты (рис. 7.8).

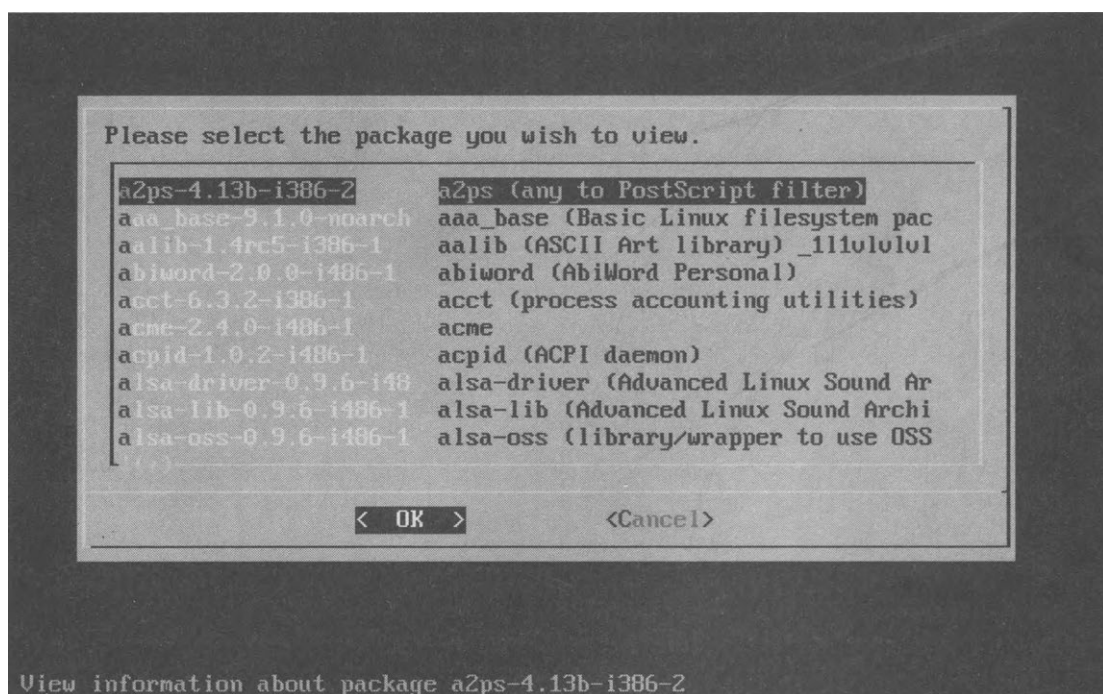


Рис. 7.8. Slackware: программа `pkgtool`

В его работе несложно разобраться, поэтому мы подробно его рассматривать не станем. А любителям графических конфигураторов наверняка понравится графическая версия этой программы — `XPKGTOOL` (рис. 7.9);

- `installpkg` — программа для установки пакетов;
- `removepkg` — программа удаления пакетов;
- `upgradepkg` — программа обновления пакетов.

Обратите внимание на каталог `install` — в нем находится сценарий `doinst.sh`, запускающийся после установки пакета.

Синтаксис команды для установки пакета:

```
# installpkg <опция> <имя пакета>
```

Вы можете задать одну из трех опций программы:

- `-m` — используется для сборки пакета (действие `makepkg`) в текущем каталоге;
- `-warn` — режим предупреждений: установка пакета не производится, однако выводится список планируемых действий. Если вы устанавливаете пакет на критически важной системе или просто не уверены в своих действиях, перед установкой пакета рекомендуется использовать режим предупреждений;
- `-r` — рекурсивно устанавливает все пакеты из текущего каталога и всех его подкаталогов.

Информация об установленных пакетах хранится в файле `/var/log/packages`. При установке пакетов вы можете указывать сразу несколько пакетов, а также использовать маски имен (типа `gnome*`).

Как уже было отмечено, при установке пакетов не проверяются зависимости пакетов, поэтому желательно первую установку производить в режиме `-warn`. Также `installpkg` не сообщит, если вы попытаетесь установить уже установленный пакет. Программа просто перезапишет старые файлы новыми версиями (из устанавливаемого пакета). Вы думаете, что это недостаток? Может и так, зато легко производить обновление пакета,— можно просто запустить программу `installpkg`, хотя для более безопасного обновления рекомендуется использовать программу `upgradepkg`.

Пример вызова программы:

```
# installpkg bash-2.04b-i386-2.tgz
```

Программа удаления пакетов `removerpkg`

Формат вызова программы `removerpkg` такой же, что и в предыдущем случае:

```
# removerpkg <опция> <имя пакета>
```

Опций у `removerpkg` немного больше — четыре:

- `-copy` — копирует пакет в резервный каталог, но не удаляет его (см. опцию `preserve`);
- `-keep` — сохраняет временные файлы, которые программа создает при удалении пакета. Полезно при тестировании созданных вами пакетов (если вы разработчик/сборщик пакета);
- `-preserve` — удаляет пакет, но перед удалением копирует его в резервный каталог. Место на диске с этой опцией не сэкономишь, зато пакеты можно полностью из системы не удалять;
- `-warn` — режим предупреждения: не удаляет пакет, а просто показывает список действий, которые будут выполнены при удалении пакета.

Пример вызова программы:

```
# removepkg bash
```

Программа обновления пакетов `upgradepkg`

Использовать программу обновления пакетов очень просто:

```
# upgradepkg <имя пакета>
```

Программа сначала устанавливает новую версию пакета, а затем удаляет старую, чтобы в системе не остались старые версии файлов.

7.9.2. Нет нужного пакета: вам поможет программа `rpm2tgz`

Иногда просто невозможно найти программу, распространяющуюся в пакете Slackware, — как известно, большинство пакетов распространяется в формате RPM. В этом случае можно воспользоваться программой `rpm2tgz`, преобразующей пакет формата RPM в формат Slackware. При этом следует понимать, что эта программа преобразует лишь формат пакетов, но не занимается разрешением зависимостей и т. п., т. е. нет никакой гарантии, что после такого преобразования установленная программа будет работать.

ПОИСК SLACKWARE-ПАКЕТОВ

Вы думаете, что для вашей программы нет Slackware-пакета? А может, вы не там искали? Попробуйте посетить сайт <http://linuxpackages.net/> — там есть очень много Slackware-пакетов.

7.9.3. Программа `slackpkg`: установка пакетов из Интернета

Наверное, вы заметили, что программа `installpkg` занимается установкой пакетов из локального каталога. А что делать, если пакет находится в Интернете? Понятно, что его нужно скачать и установить программой `installpkg`, но если вы привыкли к программам вроде `yum`, Slackware из-за этого может показаться вам ущербным и малофункциональным дистрибутивом.

На помощь приходит программа `slackpkg`, позволяющая несколько автоматизировать установку пакетов из сетевых источников — в Slackware сетевые источники называются *зеркалами* (от англ. *mirrors*). Программа `slackpkg` может скачать и установить пакет, находящийся на одном из серверов-зеркал. Но эта программа не занимается разрешением зависимостей, а только слегка упрощает установку и обновление пакетов. Не нужно думать, что `slackpkg` — это замена `installpkg`, она всего лишь ее полезное дополнение, позволяющее немного облегчить установку пакетов.

Программа `slackpkg` находится в каталоге `extra`. После установки этой программы нужно подготовить ее к работе. Первым делом откройте ее главный конфигураци-

онный файл `/etc/slackpkg/mirrors` и раскомментируйте географически ближайшее к вам зеркало, т. е. адрес ближайшего FTP-сервера, содержащего Slackware-пакеты:

```
ftp://ftp.nluug.nl/pub/os/Linux/distr/slackware/slackware-12.0/
```

ИСПОЛЬЗОВАНИЕ ЗЕРКАЛ

Помните, что `slackpkg` позволяет использовать только одно зеркало. Если вы раскомментируете несколько зеркал, будет использоваться первое раскомментированное зеркало.

После редактирования файла зеркал нужно подготовить программу для работы с GPG-ключами.

Для этого введите команды:

```
# mkdir ~/.gnupg
# gpg --keyserver pgp.mit.edu --search security@slackware.com
```

При выполнении второй команды на экран будет выведено следующее сообщение:

gpg: searching for "security@slackware.com" from HKP server pgp.mit.edu

Keys 1-2 of 2 for "security@slackware.com"

(1) Slackware Linux Project <security@slackware.com>

1024 bit DSA key 40102233, created 2003-02-25

(2) Slackware Linux Project <security@slackware.com>

1024 bit DSA key 40102233, created 2003-02-25

Enter number(s), N)ext, or Q)uit >

Как видите, вас просят выбрать номер GPG-ключа. Введите номер одного из доступных GPG-ключей (список ключей перед вами, обычно можно ввести `i`).

Теперь вам осталось ввести еще одну команду:

```
gpg --fingerprint security@slackware.com
```

Все, программа `slackpkg` готова к использованию.

Перед установкой пакетов не помешает обновить список пакетов активного зеркала. Для этого служит команда:

```
# slackpkg upgrade
```

Чтобы иметь постоянно свежие сведения о пакетах, рекомендуется регулярно выполнять эту команду.

Для установки пакета введите команду:

```
# slackpkg install <пакет>
```

Для обновления пакета используется команда:

```
# slackpkg upgrade <пакет>
```


7.10. Установка программ в openSUSE

7.10.1. Менеджер пакетов zypper

Менеджер пакетов openSUSE— zypper— работает по уже знакомому нам сценарию: в системе имеется список источников пакетов (каталог `/etc/zypp/repos.d`), который просматривается перед установкой пакета с целью определения хранилища, в котором находится устанавливаемый пакет. Затем менеджер пакетов загружает необходимый пакет (или пакеты) и устанавливает его.

Зайдите в каталог `/etc/zypp/repos.d`. В нем вы обнаружите несколько REPO-файлов, в каждом из которых прописан один репозиторий. В листинге 7.3 представлен репозиторий установочного DVD.

Листинг 7.3. Репозиторий установочного DVD (локальный репозиторий)

```
[openSUSE-Leap-42.3-0]
name=openSUSE-Leap-42.3-0
enabled=0
autorefresh=0
baseurl=cd:///?devices=/dev/disk/by-id/ata-
IDE_CDROM_Drive_10000000000000000001
path=/
type=yast2
keeppackages=0
```

Параметр `baseuri` задает здесь путь к источнику пакетов, а параметр `enabled`, установленный в 0, говорит о том, что репозиторий неактивен (установка программного обеспечения с него не производится). Если включен параметр `keeppackages`, менеджер пакетов не будет удалять пакеты после их установки.

Пример сетевого источника пакетов Main Repository (OSS) приведен в листинге 7.4.

Листинг 7.4. Пример сетевого репозитория

```
[repo-oss]
name=openSUSE-Leap-42.3-0ss
enabled=1
autorefresh=1
baseurl=http://download.opensuse.org/distribution/42.3/repo/oss/
path=/
type=yast2
keeppackages=0
```

Как видите, параметр `baseuri` указывает здесь не на локальное устройство, а на сервер в Интернете. Также обратите внимание на опцию `autorefresh` (автоматическое обновление) — для сетевого репозитория она установлена в 1, поскольку пакеты в репозитории могут меняться (например, там появляются новые версии пакетов).

А для локального источника пакетов автоматическое обновление отключено, потому что пакеты в нем остаются одни и те же.

Если у вас нет соединения с Интернетом или же оно медленное, вам придется использовать только один источник пакетов — локальный установочный DVD. Поэтому откройте терминал, введите команду `su`, а затем `gedit`, — этими командами вы запустите обычный текстовый редактор от имени администратора. Перейдите в каталог `/etc/zypp/repos.d` и откройте все файлы, кроме `openSUSE-Leap-42.3.repo` (он как раз и описывает установочный DVD). Установите для всех сетевых источников пакетов параметр `enabled` в 0. Затем откройте файл DVD-диска и опцию `enabled` установите в 1.

Если установить опцию `keeppackages` в 1, то для этого репозитория менеджер пакетов будет сохранять все загруженные пакеты. Если `keeppackages=0`, то после установки загруженный пакет удаляется.

Основным файлом конфигурации менеджера пакетов является файл `/etc/zypp/zypp.conf`, но в нем нет ничего интересного — обычно все опции там закомментированы, поскольку параметры по умолчанию устраивают всех, и их редко приходится менять.

Файлы репозиториев обычно не приходится подключать вручную — вы скачиваете из Интернета YMP-файл, в котором описаны все необходимые репозитории и пакеты, которые нужно установить (хотя могут быть прописаны только репозитории — без пакетов). Этот файл представлен в формате XML (extended Markup Language). В каждой секции `<repository>` описывается один репозиторий (если репозиториев несколько, то и секций `<repository>` будет несколько). В листинге 7.5 представлена секция `<repository>` YMP-файла для главного сетевого репозитория — Main Repository (OSS).

Листинг 7.5. Секция `<repository>` YMP-файла для главного сетевого репозитория

```
Repository recommended="true">
  <name>Main Repository (OSS)</name>
  <summary>Main OSS Repository</summary>
  <description>The largest and main repository from openSUSE for open source
                                     software</description>
  <url>http://download.opensuse.org/repositories/openSUSE:42.3/standard/</url>
</repository>
```

Каждый пакет, который нужно установить, прописывается в отдельной секции YMP-файла: `<item>` (листинг 7.6).

Листинг 7.6. Секция `<item>` YMP-файла для установки пакета `w32codec-all`

```
<item>
  <name>w32codec-all</name>
  <summary>Win 32 Codecs</summary>
  <description>This packages contains the media player windows codec dlls
                                     for several multimedia formats.</description>
</item>
```

Понятно, что если нужно установить несколько пакетов, то и секций `<item>` будет несколько.

ПОДДЕРЖКА МУЛЬТИМЕДИАФОРМАТОВ

В листингах 7.5 и 7.6 приведены фрагменты файла `codecs-gnome.ymf`, благодаря которому в openSUSE устанавливается поддержка мультимедиаформатов.

ИЗМЕНЕНИЕ YMP-ФАЙЛОВ

Приведенная здесь информация нужна лишь для общего развития — вам никогда не придется изменять YMP-файлы (хотя кто знает, что нас ждет в этой жизни?), а установка таких файлов производится автоматически, практически без вмешательства пользователя.

Теперь перейдем непосредственно к использованию менеджера пакетов `zypper`. Формат его вызова следующий:

```
zypper <команда> [пакеты]
```

Основные команды `zypper` приведены в табл. 7.5.

Таблица 7.5. Основные команды `zypper`

Команда	Описание
<code>sl</code>	Выводит список используемых репозиториев
<code>sa URL имя</code>	Добавляет репозиторий (URL — адрес репозитория, а имя — имя, под которым он будет отображаться). Пример: <code>zypper sa http://сервер/pub/linux/suse/SUSE-Linux-Updates</code>
<code>sd URL имя</code>	Удаляет репозиторий. При удалении вы можете указать URL или имя репозитория
<code>install пакеты</code>	Устанавливает пакеты. Пример: <code>zypper install me</code> Если нужно установить несколько пакетов, то имена пакетов разделяются пробелами
<code>search маска</code>	Ищет пакеты по маске. Маска — это часть имени (или полное имя) пакета. Пример: <code>zypper search me*</code>
<code>list-updates</code>	Отображает доступные обновления
<code>update пакет</code>	Обновляет пакет. Если пакет не задан, обновляет всю систему
<code>info пакет</code>	Выводит информацию о пакете
<code>remove пакет</code>	Удаляет пакет

7.10.2. Графический менеджер пакетов openSUSE

Устанавливать RPM-пакеты в openSUSE можно с помощью трех программ: собственно `zypper`, графической оболочки для `zypper` и программы `rpm`.

Программой `zypper` (см. *разд. 7.10.1*) пользоваться решаться не все — она работает в командной строке. Программу `rpm` (см. *разд. 7.4*) удобно использовать, если есть уже скачанный собственными силами RPM-пакет и его нужно установить — т. е. для локальной установки RPM-пакета. Для установки пакетов из любого репозитория, будь то DVD или сервер Интернета, намного удобнее воспользоваться графической оболочкой программы `zypper`: ввел название пакета, отметил его для установки и установил.

Для запуска графической оболочки `zypper` (графического менеджера пакетов) выполните команду **Компьютер | Система | Установка/удаление программ**. Использовать графическую утилиту всегда просто, поэтому, думаю, вы разберетесь без моих комментариев (рис. 7.10).

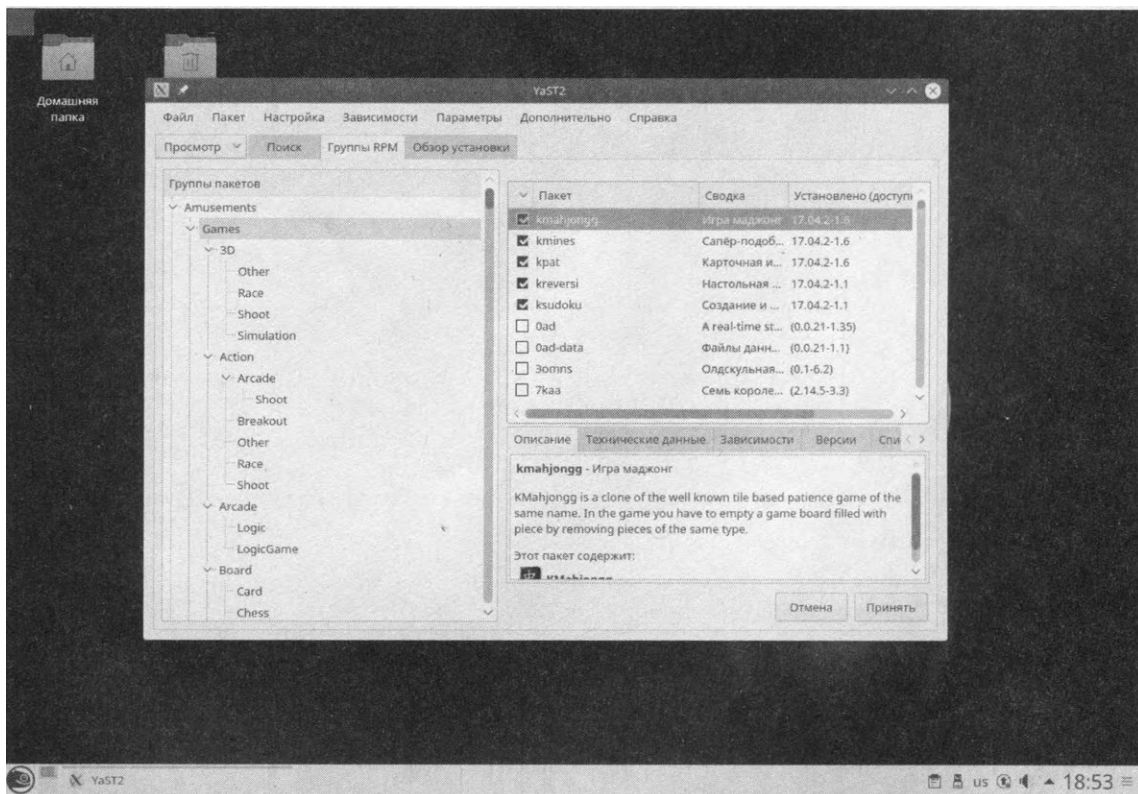
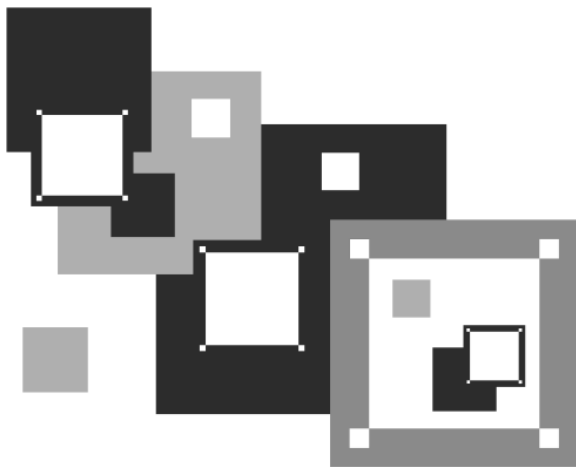


Рис. 7.10. openSUSE: менеджер пакетов

ПРИМЕЧАНИЕ

При просмотре списка пакетов обратите внимание на значки у названий пакетов: пустой квадратик говорит о том, что пакет не установлен, а «галочка» свидетельствует об установке пакета. Зеленая галка говорит о том, что пакет только выбран для установки, но пока еще не установлен. Для установки выбранных пакетов нажмите кнопку **Принять**. Для удаления пакетов снимите галки с ненужных пакетов и также нажмите кнопку **Принять**.

На этом обзор систем управления пакетами можно считать завершенным. Надеюсь, я ничего не забыл!



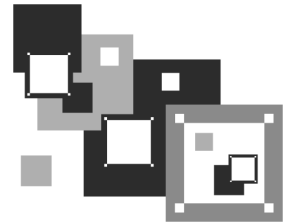
ЧАСТЬ III

Настройка сети и Интернета

Третья часть книги посвящена настройке в Linux локальной сети, интернет-соединения Wi-Fi, настройке VPN-соединения, а также интересной задаче объединения интернет-каналов.

Слышал как-то любопытное изречение (к сожалению, не помню, чье — давно это было): «Linux без сети, как птица без полета». Полностью согласен с ним, поэтому настройку сети в Linux можно считать одним из основных этапов конфигурации системы.

ГЛАВА 8



Настройка локальной сети

8.1. Локальная сеть с использованием технологии Fast Ethernet

Сетевых технологий существует много, но в этой книге мы займемся настройкой локальной сети, построенной по технологии Fast Ethernet. Зато рассмотрим мы ее полностью: от обжатия кабеля до конфигурирования сети.

Основные характеристики стандарта Fast Ethernet:

- скорость передачи данных: ЮОМбит/с;
- метод доступа к среде передачи данных: CSMA/CD;
- среда передачи данных: витая пара UTP 3, 4 или 5-й категории (лучше 5-й), оптоволоконный кабель;
- максимальное количество компьютеров: 1024;
- максимальная длина сети: 200 м (272 м для оптоволокна).

Прежде всего нужно убедиться, что компьютеры, предназначенные для соединения в сеть, оснащены *сетевыми адаптерами*, поддерживающими технологию Fast Ethernet. Как правило, сейчас сетевые адаптеры интегрированы в материнскую плату, и устанавливать их отдельно необходимости нет. Однако все еще встречаются иногда материнские платы и без интегрированных сетевых адаптеров — что ж, в таком случае их придется приобрести (рис. 8.1). Стоят они не столь и дорого — от 550 рублей за штуку, причем поддерживают также и технологию Gigabit Ethernet — модификацию Fast Ethernet, позволяющую передавать данные со скоростью до 1000 Мбит/с. Правда, *коммутаторы* (см. далее) для Gigabit Ethernet стоят пока чуть дороже, чем для Fast Ethernet.

Установка сетевого адаптера проблем не вызывает — просто вставьте его в свободный разъем шины PCI (все адаптеры Fast Ethernet выполнены в виде плат расширения именно для шины PCI). Существуют также сетевые адаптеры, подключаемые к компьютеру по USB (рис. 8.2), — стоят они не сильно дороже PCI-адаптеров, и нет никаких оснований подозревать, что при работе в Linux с ними возникнут какие-либо проблемы.

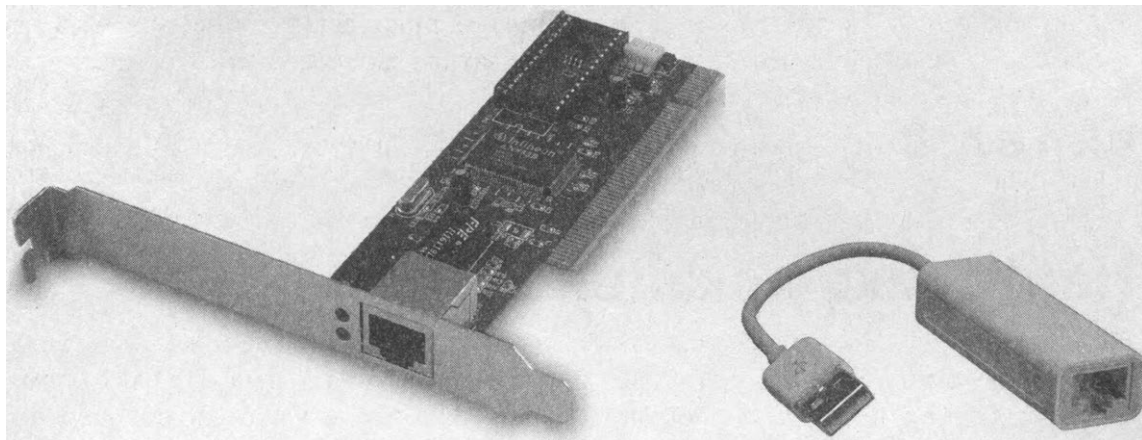


Рис. 8.1. Сетевой адаптер Fast Ethernet

Рис. 8.2. Сетевой адаптер USB

Ясно, что устанавливать сетевой PCI-адаптер полагается при выключенном компьютере — шина PCI пока еще не поддерживает «горячей замены». Собрав и включив компьютер, подключите к сетевому адаптеру *коннектор* (специальный накопечник) сетевого кабеля.

Сетевые кабели различной длины (от 0,5 до 5 м), оснащенные коннекторами (патч-корды), можно приобрести в магазинах компьютерной техники, а можно и сделать самим, нарезав кабель на нужные отрезки и закрепив (обжав) коннекторы на их концах.

ОБЖАТИЕ КАБЕЛЯ

Обжать кабель — значит особым образом закрепить на его концах специальные накопечники-коннекторы (см. далее).

Выполняются эти манипуляции, как правило, администратором сети. Вы сами администрируете свою сеть и не знаете, как это сделать? Нет проблем, сейчас разберемся. Для создания сети Fast Ethernet вам потребуются следующие устройства:

- сетевые адаптеры — о них мы только что поговорили;
- коммутатор (switch) — его можно купить в любом компьютерном магазине. Дизайном и количеством портов коммутаторы могут отличаться друг от друга. На рис. 8.3 изображен 24-портовый коммутатор, более подходящий для корпоративной сети (и внешним видом и возможностью помещения в специальную стойку), нежели для дома. А для домашней сети можно найти и более симпатичное устройство;

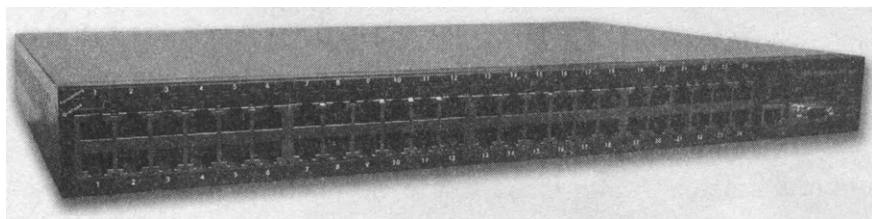


Рис. 8.3. Коммутатор (switch)

- сетевой кабель (витая пара 5-й категории)— приобретайте именно такой тип кабеля и такой длины, чтобы нормально хватило для соединения каждого компьютера сети с коммутатором;
- коннекторы RJ-45 — таких коннекторов вам понадобится в два раза больше, чем компьютеров, поскольку каждый отрезок кабеля нужно обжать с двух концов. Но я рекомендую купить еще несколько лишних штук — если вы будете обжимать кабель впервые, думаю, без неудачных проб не обойдется. Не пожалейте пару копеек, а то придется сбегать в магазин еще раз;
- инструмент (специальные обжимные щипцы) для обжимки коннекторов витой пары — хороший инструмент стоит относительно дорого (примерно как коммутатор), а плохой лучше не покупать. Если не хотите выкладываться, одолжите такие щипцы у кого-нибудь на пару дней.

Теперь приступим к самому процессу обжимки. Внутри кабеля идут четыре витые пары проводов (всего восемь), причем у каждого провода своя цветовая маркировка. Суть процесса обжимки заключается в том, чтобы подключить каждый из проводов к нужному контакту коннектора (табл. 8.1). Для этого сначала надо вставить провода в коннектор таким образом, чтобы каждый провод зашел на всю глубину по направляющим соответствующего контакта (зачищать провода необязательно — за вас это сделает инструмент), затем коннектор со вставленными проводами осторожно (чтобы провода из него не выпали) помещается в специальное гнездо обжимных щипцов, и их рукоятки сильно сжимаются. Используя данные табл. 8.1, вы без проблем сможете обжать кабель.

Таблица 8.1. Обжим витой пары

Контакт	Цвет провода	Контакт	Цвет провода
1	Бело-оранжевый	5	Бело-синий
2	Оранжевый	6	Зеленый
3	Бело-зеленый	7	Бело-коричневый
4	Синий	8	Коричневый

Осталось один конец обжатого отрезка кабеля своим коннектором подключить к коммутатору (концентратору), а второй — к сетевому адаптеру компьютера. Если вы неправильно (или несильно) обожмете кабель, то ваша сеть работать не будет или же будет работать только на скорости 10 Мбит/с.

Проверить, правильно ли вы обжали кабель, очень просто — обратите внимание на коммутатор: возле каждого его порта имеется по два индикатора. Если горят оба — все нормально. Если же горит только один из них, то этот порт работает в режиме 10 Мбит/с. А если вообще не горит ни один из индикаторов, вам нужно переобжать кабель — отрезать плохо обжатые коннекторы и обжать концы кабеля новыми коннекторами заново.

Как видите, в процессе обжима нет ничего сложного.

8.2. Файлы конфигурации сети в Linux

Прежде чем приступить к настройке сети, следует ознакомиться с файлами конфигурации сети, которые имеются в любом дистрибутиве Linux, вне зависимости от его версии (табл. 8.2).

Таблица 8.2. Общие файлы конфигурации сети в Linux

Файл	Описание
/etc/aliases	База данных почтовых псевдонимов. Формат этого файла очень прост: псевдоним пользователь
/etc/aliases.db	Системой на самом деле используется не файл /etc/aliases, а файл /etc/aliases.db, который создается программой newaliases по содержимому файла /etc/aliases. Поэтому после редактирования этого файла не забудьте выполнить от имени root команду newaliases
/etc/hosts.conf	Содержит параметры разрешения доменных имен. Например, директива <code>order hosts,bind</code> означает, что сначала поиск IP-адреса по доменному имени будет произведен в файле /etc/hosts, а затем лишь будет произведено обращение к DNS-серверу, заданному в файле /etc/resolv.conf. Директива <code>multi on</code> означает, что одному доменному имени могут соответствовать несколько IP-адресов
/etc/hosts	В этом файле можно прописать IP-адреса и имена узлов локальной сети, но обычно здесь указывается только IP-адрес узла localhost (127.0.0.1), потому что сейчас даже в небольшой локальной сети устанавливается собственный DNS-сервер
/etc/hosts.allow	Содержит IP-адреса узлов, которым разрешен доступ к сервисам данного узла
/etc/hosts.deny	Содержит IP-адреса узлов, которым запрещен доступ к сервисам данного узла
/etc/hostname	В Debian/Ubuntu содержит имя узла
/etc/iftab	Содержит таблицу интерфейсов, т. е. соответствие имен интерфейсов и их MAC-адресов
/etc/motd	Файл задает сообщение дня (Message of the day). Этот файл используется многими сетевыми сервисами (например, серверами FTP и SSH), которые при регистрации пользователя могут выводить сообщение из этого файла
/etc/network/interfaces	В Debian и Ubuntu используется для ручной настройки сетевых интерфейсов (не с помощью NetworkManager). Вообще-то принято настраивать сетевые интерфейсы с помощью NetworkManager, но некоторые администраторы предпочитают отключать NetworkManager и настраивать сетевые интерфейсы вручную — по старинке
/etc/rc.config	В старых версиях SUSE (не openSUSE) содержит имя компьютера, IP-адрес интерфейса и другую сетевую информацию

Таблица 8.2 (окончание)

Файл	Описание
/etc/resolv.conf	Задаёт IP-адреса серверов DNS. Формат файла прост: nameserver IP-адрес Всего можно указать четыре DNS-сервера. В Ubuntu этот файл автоматически перезаписывается при установке соединения с Интернетом — сюда записываются адреса DNS-серверов, полученных от провайдера, что не совсем хорошо, особенно когда вы настроили собственный DNS-сервер и желаете его использовать. О моей борьбе с перезаписью этого файла можно прочитать статью по адресу: http://www.dkws.org.ua/index.php?page=show&file=a/ubuntu/static-dns-ubuntu9
/etc/route.conf	В старых версиях SUSE этот файл содержит описание статических маршрутов, в том числе и маршрут по умолчанию
/etc/services	База данных сервисов, задающая соответствие символического имени сервиса (например, pop3) и номера порта (110/tcp, tcp — это наименование протокола)
/etc/sysconfig/network	Параметры сетевого интерфейса в Fedora, Red Hat и других дистрибутивах, основанных на Fedora/Red Hat
/etc/sysconfig/static-routes	Статические маршруты в Fedora/CentOS
/etc/sysconfig/network/routes	Статические маршруты в современных версиях openSUSE
/etc/sysconfig/network-scripts/ifcfg-имя	Параметры конкретного сетевого интерфейса. Например, параметры интерфейса eth0 хранятся в файле /etc/sysconfig/network-scripts/ifcfg-eth0 (дистрибутив Fedora)
/etc/sysconfig/network/ifcfg-имя	Параметры конкретного сетевого интерфейса (имя — имя сетевого интерфейса). Дистрибутив openSUSE
/etc/NetworkManager/system-connections/	В дистрибутивах, использующих NetworkManager, в этом каталоге хранятся настройки соединений: в отдельных файлах — по одному для каждого соединения. При этом название файла соответствует названию соединения, введенному при настройке

8.3. Об именах сетевых интерфейсов

Все течет, и все меняется. В мире компьютеров обычно все меняется в лучшую, более простую и понятную сторону. Взять хотя бы настройку сети с помощью конфигураторов. Раньше у каждого дистрибутива был свой конфигуратор сети, и не один — это и понятно: свои файлы конфигурации, сервисы настройки сети и т. д.

Сейчас же, благодаря тому что все (или практически все) современные дистрибутивы перешли на единый сервис настройки сети NetworkManager, конфигуратор сети у всех стал один — nm-connection-editor, и выглядит он примерно одинаково во всех дистрибутивах. Это как универсальное зарядное USB-устройство, которым можно зарядить любой современный смартфон любого производителя.

Удобно? Определенно! Но когда я ввел на своем компьютере команду `ifconfig`, то обнаружил, что моя единственная сетевая карта стала называться теперь **ens33** вме-сто привычного имени `eth0` (рис. 8.4), что не есть хорошо... А в некоторых дистри-бутивах сетевая карта получила еще менее благозвучное название — `enp3s0`. В об-щем, в связи с полным переходом новых дистрибутивов на `udev` (см. главу 4), по-нятная с первого взгляда строгость в назначении имен сетевых адаптеров (когда первая сетевая плата называлась `eth0`, вторая `eth1` и т. д.) исчезла.

```

den@localhost:~
Файл Правка Вид Поиск Терминал Справка
[den@localhost ~]$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
    inet 192.168.52.179 netmask 255.255.255.0  broadcast 192.168.52.255
    inet6 fe80::20c:29ff:fe01:6729 prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:01:67:29  txqueuelen 1000  (Ethernet)
    RX packets 101434  bytes 146964622 (140.1 MiB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 50802  bytes 2775496 (2.6 MiB)
    TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
    device interrupt 19  base 0x2000

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 0  (Local Loopback)
    RX packets 8  bytes 800 (800.0 B)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 8  bytes 800 (800.0 B)
    TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[den@localhost ~]$
  
```

Рис. 8.4. Новые имена сетевых устройств

Впрочем, определенная система в назначении новых имен также имеется, только она не столь очевидная. Сетевые адаптеры, встроенные в материнскую плату (`id_net_name_onboard`), носят теперь название типа `enol`. Если же сетевая карта подключена к PCI Express (`ID_NET_NAME_SLOT`), то она будет нести название типа `ens33`. Имена устройств, содержащие физическое/географическое расположение кон-нектора (`ID_NET_NAME_PATH`), будут выглядеть как `enp2s0`. А самые сложные и «страшные» имена у сетевых адаптеров, определяемых через MAC-адрес,— `enx66e7bles34dd`.

Как работать с такими именами? Первый способ — использовать их и смириться с новой схемой именования. Все равно, сетевой интерфейс настраивается не так часто и, по сути, вы столкнетесь с его настройкой один-два раза (а при использо-вании DHCP, возможно, его не придется настраивать вовсе).

Второй способ — передать ядру параметр `net.ifnames=0` (рис. 8.5) и вернуть старую схему именования сетевых устройств. Как показано на рис. 8.6, после передачи ядру параметра `net.ifnames=0` сетевая карта стала называться по-старому: **eth0**.

```

insmod part_msdos
insmod ext2
set root='hd0,msdos1'
if [ x$feature_platform_search_hint = xy 1; then
    search --no-floppy --fs-uuid --set=root --hint-ieee1275='ieee1275//s\
as/disk@0,msdos1' --hint-bios=hd0,msdos1 --hint-efi=hd0,msdos1 --hint-baremeta\
l=ahci0,msdos1 --hint='hd0,msdos1' f3f92c5a-8515-42c0-82e9-878ec21c2d32
else
    search --no-floppy --fs-uuid --set=root f3f92c5a-8515-42c0-82e9-878e\
c21c2d32
fi
linux16 /boot/vmlinuz-4.0.4-301.fc22.i686 root=/dev/sda1 ro rhgb quiet\
LANG=ru_RU.UTF-8 net.ifnames=0_
initrd16 /boot/initramfs-4.0.4-301.fc22.i686.img

Press Ctrl-x to start, Ctrl-c for a command prompt or Escape to
discard edits and return to the menu. Pressing Tab lists
possible completions.

```

Рис. 8.5. Передача ядру параметра net.ifnames=0

```

den@localhost:~
Файл Правка Вид Поиск Терминал Справка
[den@localhost ~]$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.52.185 netmask 255.255.255.0 broadcast 192.168.52.255
    inet6 fe80::20c:29ff:fe01:6729 prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:01:67:29 txqueuelen 1000 (Ethernet)
    RX packets 68 bytes 14770 (14.4 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 78 bytes 8624 (8.4 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 19 base 0x2000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 0 (Local Loopback)
    RX packets 8 bytes 800 (800.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 8 bytes 800 (800.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

[den@localhost ~]$

```

Рис. 8.6. Fedora: сетевым устройствам вернулись старые имена

Вернуться к старым именам или использовать новые — дело вкуса и личных предпочтений. На работе сети выбор схемы именования имен никак не отразится.

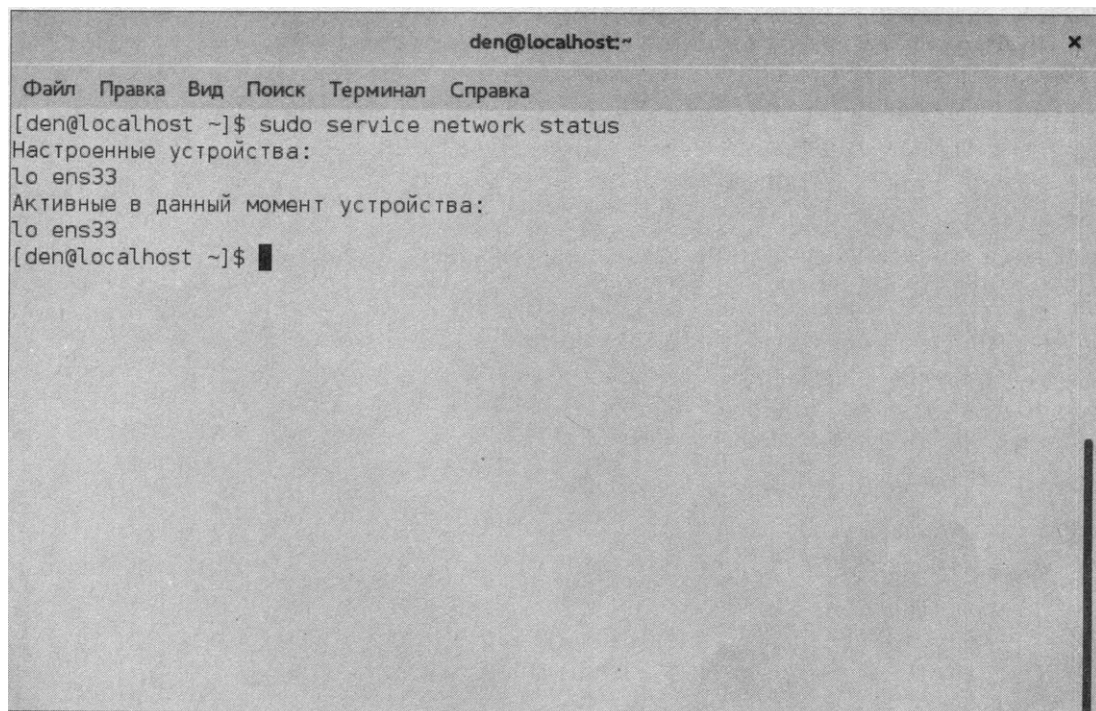
8.4. Настройка сети с помощью конфигуратора nm-connection-editor

Как уже было отмечено ранее, теперь настройка сети осуществляется с помощью единого конфигуратора сети nm-connection-editor, средствами которого можно создать/настроить любой поддерживаемый тип подключения.

Прежде чем приступить к настройке сети с помощью конфигуратора, посмотрим сначала на сервис network. Его можно использовать для управления сетью — например, остановить все сетевые интерфейсы, завершив работу этого сервиса. А перезапустив его, можно перезапустить сеть, что полезно при изменении параметров сетевых адаптеров, — так вам не придется перезапускать компьютер. Управлять сервисом network можно, как и обычным сервисом:

```
sudo service network stop      # останавливает сеть
sudo service network start     # запускает сеть
sudo service network restart   # перезапускает сеть
sudo service network status    # состояние сети
```

Если в выводе команды `ifconfig` или `service network status` (рис. 8.7) отсутствует интерфейс `lo`, значит, сервис network остановлен. Такие случаи — редкость, но все

A screenshot of a terminal window titled "den@localhost:~". The window has a menu bar with "Файл", "Правка", "Вид", "Поиск", "Терминал", and "Справка". The terminal shows the command `[den@localhost ~]$ sudo service network status` and its output: "Настроенные устройства: lo ens33" and "Активные в данный момент устройства: lo ens33". The prompt `[den@localhost ~]$` is followed by a cursor.

```
den@localhost:~
Файл Правка Вид Поиск Терминал Справка
[den@localhost ~]$ sudo service network status
Настроенные устройства:
lo ens33
Активные в данный момент устройства:
lo ens33
[den@localhost ~]$
```

Рис. 8.7. Команда `service network status` выводит список настроенных и активных сетевых интерфейсов

же с ними иногда приходится сталкиваться. Для запуска сервиса нужно ввести команду `service network start`.

Если же в вашей сети работает DHCP-сервер, то настраивать сеть в современных дистрибутивах вовсе не придется, — Linux автоматически распознает ваш адаптер, активирует соответствующие модули ядра и установит сетевые параметры, полученные от DHCP-сервера. Сейчас даже в самых небольших домашних сетях действует DHCP-сервер, запущенный на маршрутизаторе Wi-Fi, предоставляющем доступ к Интернету.

Настраивать сеть придется в двух случаях:

- если у вас небольшая сеть, использующая статические IP-адреса, — ради всего двух-трех компьютеров вы не стали настраивать DHCP-сервер;
- если вы настраиваете сеть «с нуля», и компьютер, на который вы установили Linux, как раз и будет тем DHCP-сервером, который потом станет настраивать остальные узлы сети.

УСТАНОВИТЕ DHCP-СЕРВЕР!

Даже если у вас небольшая домашняя сеть из двух-трех компьютеров, присутствие DHCP-сервера в ней весьма желательно. Во-первых, вам не придется настраивать сеть на клиентских компьютерах, надо будет настроить только сервер. Во-вторых, DHCP-сервер поможет избежать конфликтов IP-адресов при расширении сети — вам не придется вспоминать, какие адреса уже использованы, вы просто подключите новый компьютер к сети, а остальное сделает DHCP-сервер.

Часто DHCP-сервер «крутится» на точке доступа Wi-Fi или на DSL-модеме, совмещающем также и функции коммутатора. Современные сетевые устройства позволяют существенно снизить стоимость монтажа сети, особенно домашней. Так вы можете приобрести точку доступа с четырьмя Ethernet-портами (к которым могут подключаться стационарные компьютеры) и встроенным DSL-модемом. По сути, такое единое устройство обеспечивает все необходимые функции: ноутбуки будут подключаться по Wi-Fi, стационарные компьютеры — к встроенным портам Ethernet, а само подключение к Интернету осуществляется через встроенный DSL-модем.

Вот только на предприятии от подобных устройств толку мало, разве что в самых небольших офисах, поскольку количество Ethernet-портов в них редко превышает четыре, чего явно недостаточно для предприятия. Поэтому там понадобятся дополнительные устройства — как минимум еще один коммутатор для подключения остальных компьютеров.

Вернемся к нашему конфигуратору (запускается он через команду `sudo` или `gksudo`) — кнопка **Добавить** служит для создания нового соединения (в главе 9 будет показано, как создать DSL-соединение), а Ethernet-соединение, как правило, уже создано, поэтому для изменения его параметров нужно нажать кнопку **Изменить** (рис. 8.8), в результате чего откроется соответствующее окно.

Как правило, в большинстве случаев вас будут интересовать в нем вкладка **Параметры IPv4** (рис. 8.9), где можно изменить параметры протокола IP, и вкладка Ethernet (рис. 8.10), позволяющая изменить MAC-адрес устройства.

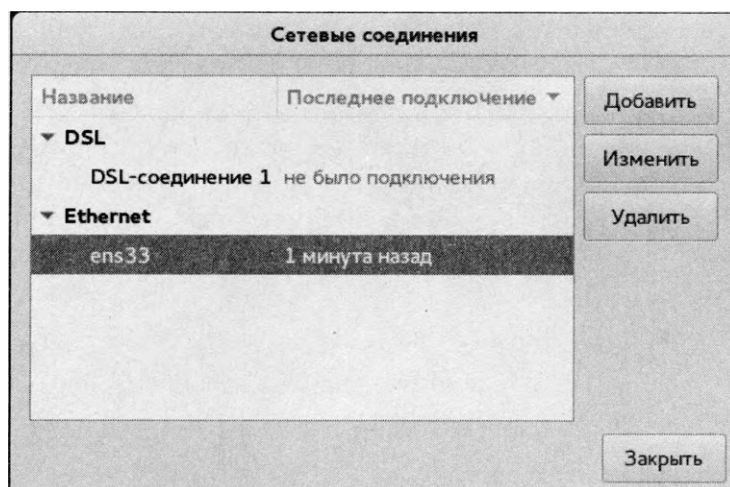
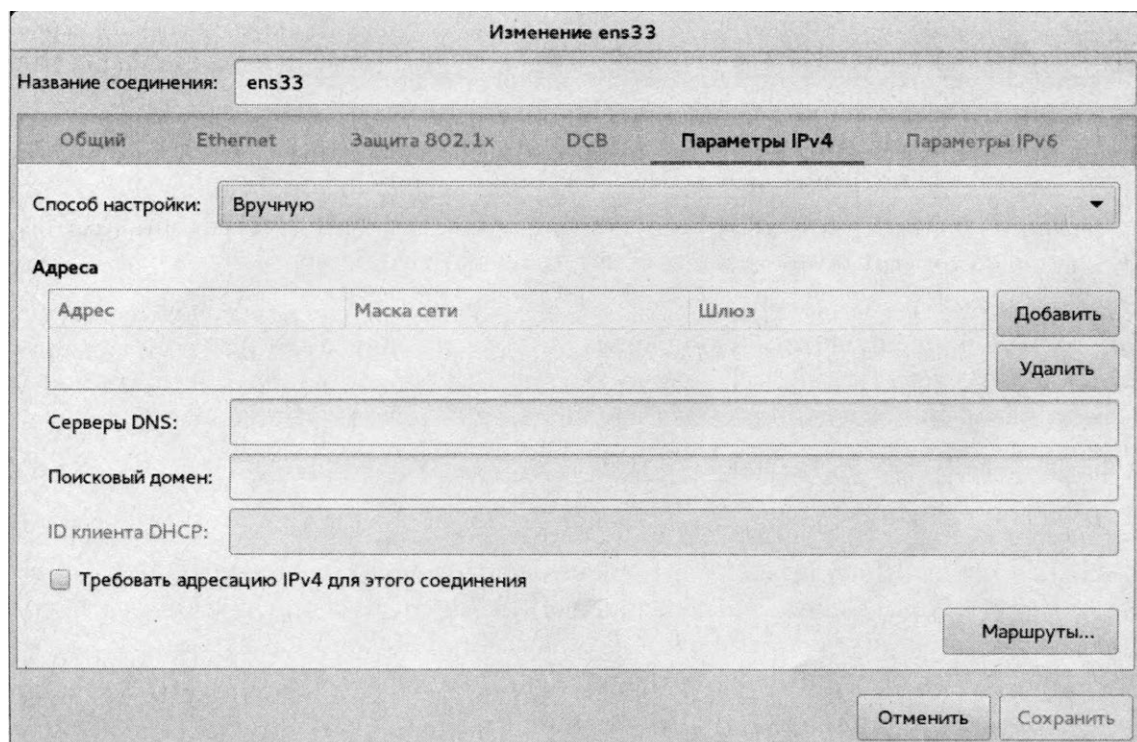


Рис. 8.8. Конфигуратор nm-connection-editor

Обычно сеть настраивается автоматически, поэтому в поле **Способ настройки** по умолчанию выбран автоматический метод настройки. Если же вы хотите указать IP-адрес вручную, нажмите на вкладке **Параметры IPv4** (см. рис. 8.9) кнопку **Добавить** и впишите IP-адрес, маску сети и IP-адрес шлюза (gateway). Всю эту информацию вы можете узнать у администратора сети. Если вы сами ее администратор, то, я надеюсь, вы знаете, что делаете.

Рис. 8.9. Конфигуратор nm-connection-editor, вкладка **Параметры IPv4**

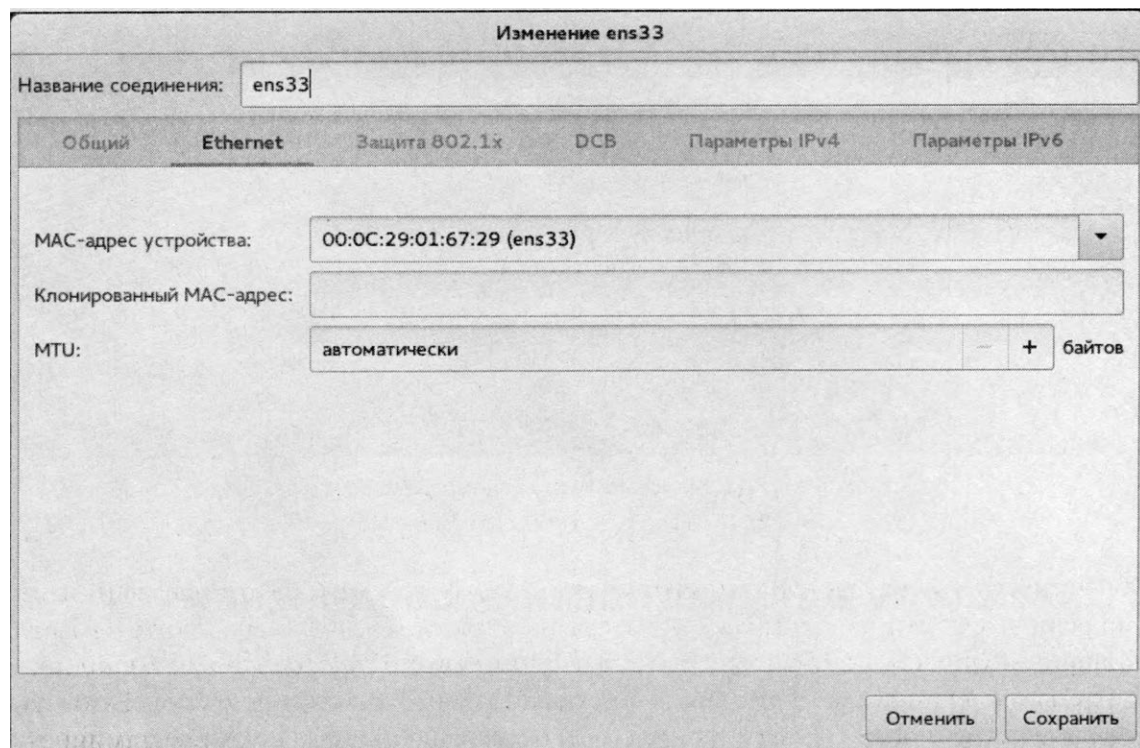


Рис. 8.10. Конфигуратор nm-connection-editor, вкладка Ethernet

В поле **Серверы DNS** можно указать IP-адреса DNS-серверов (через пробел), которые будут использоваться при разрешении доменных имен. Если вы не знаете, что указать, укажите IP-адреса 8.8.8.8 и 8.8.4.4.

Если вы собираетесь использовать DHCP, но хотите указать свои DNS-серверы, то выберите способ **Автоматически (DHCP, только адрес)**.

На вкладке **Параметры IPv6** можно указать параметры, относящиеся к протоколу IPv6, если вы таковой используете.

Linux поддерживает технологию VLAN (Virtual LAN), что позволяет назначить одному сетевому адаптеру несколько IP-адресов, однако на практике такая возможность используется редко. Дополнительную информацию о VLAN можно получить в моих статьях:

- <http://www.xakep.ru/magazine/xa/121/122/l.asp>;
- <http://www.dkws.org.ua/index.php?page=show&file=a/ubuntu/network-interfaces>.

8.5. Конфигуратор netconfig в Slackware

Конфигуратор netconfig в Slackware можно запускать даже в консоли (рис. 8.11). Он поочередно задаст вам ряд вопросов: от имени компьютера до IP-адреса шлюза. По сути, его работа ничем не отличается от работы ранее рассмотренного конфигуратора, просто у него несколько своеобразный интерфейс пользователя.

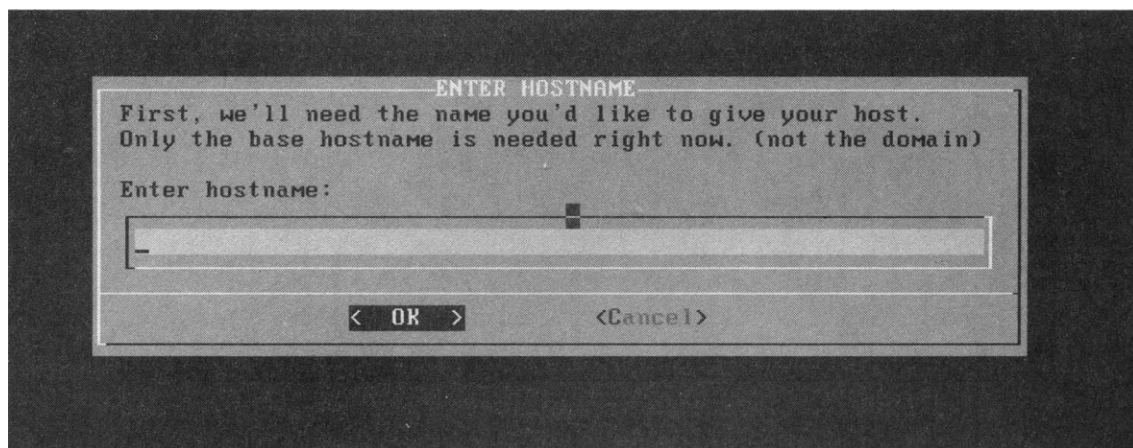


Рис. 8.11. Slackware: конфигуратор netconfig

8.6. Утилиты для диагностики соединения

Причинами отказа сети могут быть физические или программные неполадки. Физические связаны с неработающим сетевым оборудованием или повреждением среды передачи данных. Программные — с неправильной настройкой сетевого интерфейса. Как правило, избавиться от программных проблем помогает конфигуратор сети — вы его еще раз запускаете и правильно настраиваете сетевые интерфейсы. Если сомневаетесь в своих действиях, обратитесь за помощью к более опытному коллеге.

Для диагностики работы сети мы воспользуемся стандартными сетевыми утилитами, которые входят в состав любого дистрибутива Linux. Предположим, что у нас не работает PPPoE/DSL-соединение. Проверить, «поднят» ли сетевой интерфейс, можно с помощью команды `ifconfig`.

На рис. 8.12 показано, что сначала я предпринял попытку установить соединение (ввел команду `sudo pon dsl-provider`), а затем вызвал `ifconfig` — чтобы убедиться, установлено ли соединение. В случае, если соединение не было бы установлено, интерфейс `ppp0` в списке бы отсутствовал.

ИМЕНА ИНТЕРФЕЙСОВ

Интерфейс `eth0` относится к первой сетевой плате (вторая называется `eth1`, третья — `eth2` и т. д.), а интерфейс `lo` — это интерфейс обратной петли, который служит для тестирования программного обеспечения (у вас он всегда будет «поднят»).

Итак, если интерфейс не «поднят», нам нужно просмотреть файл протокола `/var/log/messages` сразу после попытки установки соединения:

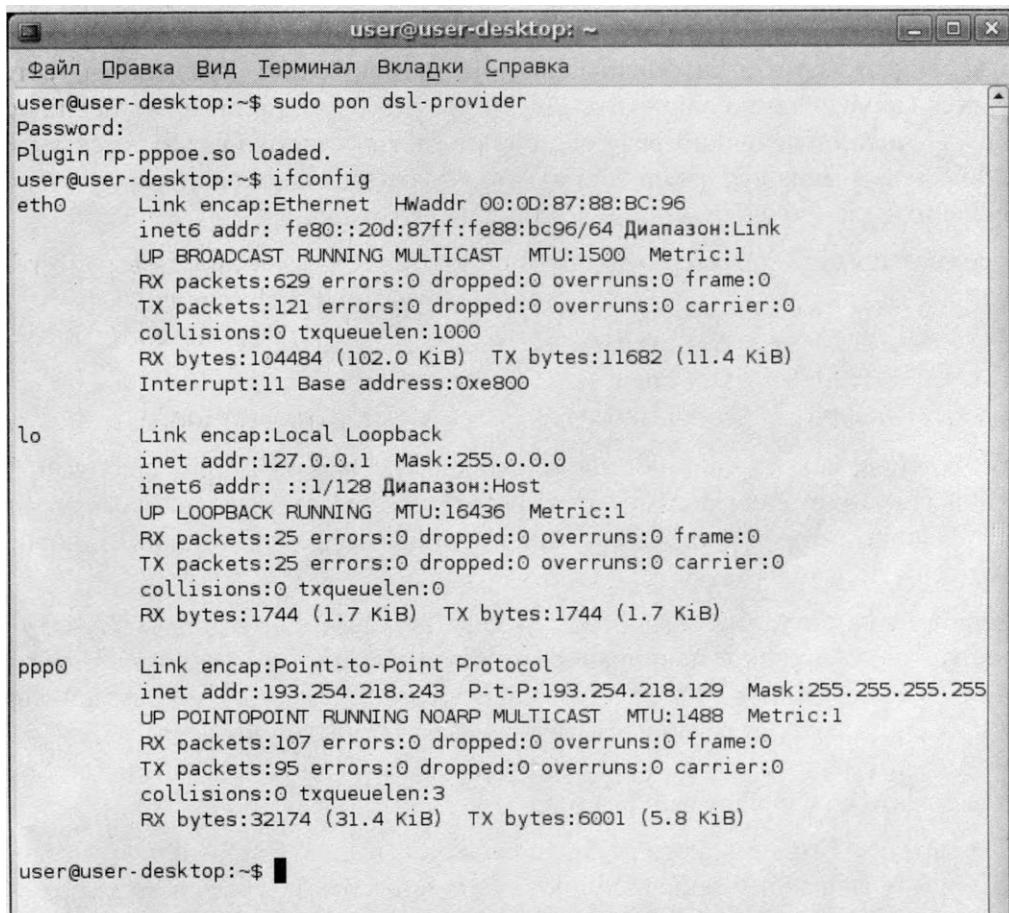
```
tail -п 10 /var/log/messages
```

Эта команда просматривает «хвост» файла протокола (выводит последние 10 сообщений). В случае удачной установки соединения сообщения в файле протокола будут примерно следующими:

```

Feb 6 14:28:33user-desktop pppd[5176]: Plugin rp-pppoe.so loaded.
Feb 6 14:28:33user-desktop kernel: [17179852.932000] CSLIP: code copyright
198 9 Regents of the University of California
Feb 6 14:28:33 user-desktop kernel: [17179852.944000] PPP generic driver
version 2.4.2
Feb 6 14:28:33user-desktop pppd[5183]: pppd 2.4.4 started by root,uid 0
Feb 6 14:28:33user-desktop pppd[5183]: PPP session is 2838
Feb 6 14:28:33user-desktop kernel: [17179852.984000] NET: Registered protocol
family 24
Feb 6 14:28:33user-desktoppppd[5183]: Using interface ppp0
Feb 6 14:28:33user-desktoppppd[5183]: Connect: ppp0 <=> eth0
Feb 6 14:28:33user-desktoppppd[5183]: Remote message: Login ok
Feb 6 14:28:33user-desktoppppd[5183]: PAP authentication succeeded
Feb 6 14:28:33 user-desktop pppd[5183]: peer from calling number 00:15:F2:60:28
:97 authorized
Feb 6 14:28:33user-desktoppppd[5183]: local IP address 193.254.218.243
Feb 6 14:28:33user-desktoppppd[5183]: remote IP address 193.254.218.129
Feb 6 14:28:33user-desktoppppd[5183]: primary DNS address 193.254.218.1
Feb 6 14:28:33user-desktoppppd[5183]: secondary DNS address 193.254.218.27

```



```

user@user-desktop: ~
Файл Правка Вид Терминал Вкладки Справка
user@user-desktop:~$ sudo pon dsl-provider
Password:
Plugin rp-pppoe.so loaded.
user@user-desktop:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0D:87:88:BC:96
          inet6 addr: fe80::20d:87ff:fe88:bc96/64 Диапазон:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:629 errors:0 dropped:0 overruns:0 frame:0
          TX packets:121 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:104484 (102.0 KiB)  TX bytes:11682 (11.4 KiB)
          Interrupt:11 Base address:0xe800

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Диапазон:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:25 errors:0 dropped:0 overruns:0 frame:0
          TX packets:25 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:1744 (1.7 KiB)  TX bytes:1744 (1.7 KiB)

ppp0      Link encap:Point-to-Point Protocol
          inet addr:193.254.218.243  P-t-P:193.254.218.129  Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1488  Metric:1
          RX packets:107 errors:0 dropped:0 overruns:0 frame:0
          TX packets:95 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:3
          RX bytes:32174 (31.4 KiB)  TX bytes:6001 (5.8 KiB)

user@user-desktop:~$ █

```

Рис. 8.12. Вывод команды ifconfig: 3 сетевых интерфейса

Первая строка — сообщение о том, что загружен модуль поддержки PPPoE. Следующие два сообщения информируют нас о поддержке нашим компьютером протоколов CSLIP и PPP. Затем сообщается, что демон `pppd` запущен, от чьего имени он запущен (`root`) и приводится версия самого `pppd`. Далее приводятся имя используемого интерфейса (`ppp0`) и имя вспомогательного интерфейса (помните, что протокол PPPoE подразумевает передачу кадров PPP по Ethernet) — `eth0`. Следующие два сообщения свидетельствуют об удачной регистрации:

```
Feb 6 14:28:33 user-desktop pppd[5183]: Remote message: Login ok
Feb 6 14:28:33 user-desktop pppd[5183]: PAP authentication succeeded
```

Затем система сообщает нам наш IP-адрес, адрес удаленного компьютера, который произвел аутентификацию, а также IP-адреса серверов DNS.

А вот пример неудачной попытки соединения:

```
Feb 6 09:23:48 user-desktop pppd[6667]: PPP session is 2336
Feb 6 09:23:48 user-desktop pppd[6667]: Using interface ppp1
Feb 6 09:23:48 user-desktop pppd[6667]: Connect: ppp1 <--> eth0
Feb 6 09:23:48 user-desktop pppd[6667]: Remote message: Login incorrect
Feb 6 09:23:48 user-desktop pppd[6667]: Connection terminated.
```

Причина неудачи понятна: имя пользователя или пароль неправильные, о чем красноречиво свидетельствует сообщение **Login incorrect**. Чтобы изменить имя пользователя или пароль, можно запустить конфигуратор `pppoeconf`. Но не спешите это делать — если в предыдущий раз соединение было установлено (а настройки соединения вы не изменяли), возможно, нужно обратиться к провайдеру — это явный признак неправильной работы оборудования на его стороне.

Вот еще один пример, характерный для PPPoE:

```
Feb 6 09:23:48 user-desktop pppd[6667]: PPP session is 2336
Feb 6 09:23:48 user-desktop pppd[6667]: Using interface ppp1
Feb 6 09:23:48 user-desktop pppd[6667]: Connect: ppp1 <--> eth0
Feb 6 09:23:48 user-desktop pppd[6667]: Connection terminated.
```

И здесь мы снова видим указание на неправильную работу оборудования провайдера. Иногда в таком случае помогает перезагрузка точки доступа (`access point`) — просто выключите и включите ее снова. Если соединение так и не восстановилось, обращайтесь к провайдеру.

Наиболее простая ситуация, когда сеть вообще не работает — в этом случае очень легко обнаружить причину неисправности. Если работает устройство, значит, повреждена среда передачи данных (сетевой кабель). В случае с модемной линией нужно проверить, нет ли ее обрыва. В случае с витой парой обрыв маловероятен (хотя возможен), поэтому нужно проверить, правильно ли обжат кабель (возможно, придется обжать витую пару заново).

Намного сложнее ситуация, когда сеть то работает, то нет. Например, вы не можете получить доступ к какому-либо узлу, хотя пять минут назад все работало отлично. Если исключить неправильную работу удаленного узла, к которому вы подключаетесь, следует поискать решение в маршруте, по которому пакеты добираются от

вашего компьютера до удаленного узла. Для этого используется команда `ping` (пре-
рвать выполнение команды `ping` можно с помощью нажатия комбинации клавиш `<Ctrl>`
`+<C>`). Сначала «пропингуем» удаленный узел:

```
ping dkws.org.ua
```

```
PING dkws.org.ua (213.186.114.75) 56(84) bytes of data.  
64 bytes from wdt.org.ru(213.186.114.75): icmp_seq=1ttl=58time=30.7ms  
64 bytes from wdt.org.ru(213.186.114.75): icmp_seq=2ttl=58time=24.8ms  
64 bytes from wdt.org.ru(213.186.114.75): icmp_seq=5ttl=58time=12.2ms  
64 bytes from wdt.org.ru (213.186.114.75): icmp_seq=6 ttl=58 time=159 ms  
64 bytes from wdt.org.ru(213.186.114.75): icmp_seq=7ttl=58time=19.3ms  
64 bytes from wdt.org.ru(213.186.114.75): icmp_seq=9ttl=58time=29.0ms
```

Здесь все нормально. Но иногда ответы от удаленного сервера то приходят, то не
приходят. Чтобы узнать, в чем причина (где именно теряются пакеты), нужно вы-
полнить трассировку узла:

```
traceth path dkws.org.ua
```

КОМАНДЫ ТРАССИРОВКИ

В некоторых дистрибутивах вместо команды `traceth path` используется команда
`traceroute`, а в Windows — `tracert`.

Из вывода команды `traceth path` (рис. 8.13) сразу видно, что с прохождением пакетов
до удаленного узла имеются определенные проблемы, т. е. по пути пакеты теря-
ются.

Чтобы выяснить причину этого, вам нужно обратиться к администратору того
маршрутизатора, который не пропускает пакеты дальше, — причина именно в нем.
В нашем случае, как можно видеть, пакеты доходят до маршрутизатора **dc-m7i-l-
ge.interfaces.dc.utel.ua**, а после него движение пакетов прекращается.

Если соединение установлено (о чем свидетельствует наличие «поднятого» интер-
фейса в выводе `ifconfig`), а Web-страницы не открываются, попробуйте пропинго-
вать любой удаленный узел по IP-адресу. Если вы не знаете, какой узел пинговать
(т. е. не помните ни одного IP-адреса), пропингуйте узел 213.186.114.75. Если вы
получите ответ, а Web-страницы, когда вы вводите символьное имя, по-прежнему
не открываются, значит, у вас проблемы с DNS, — сервер провайдера почему-то не
передал вашему компьютеру IP-адреса DNS-серверов. Позвоните провайдеру, вы-
ясните причину этого, а еще лучше, уточните IP-адреса серверов DNS и пропишите
их в файле `/etc/resolv.conf`. Формат этого файла прост:

```
nameservr IP-адрес
```

Например:

```
nameserver 193.254.218.1  
nameserver 193.254.218.27
```

Всего там можно указать до четырех серверов DNS.

```

user@user-desktop: ~
Файл Правка Вид Терминал Вкладки Справка
user@user-desktop:~$ tracepath dkws.org.ua
 1: ip-193-254-218-243.romb.net (193.254.218.243)      0.320ms pmtu 1488
 1: ip-193-254-218-129.romb.net (193.254.218.129)    94.739ms
 2: sat-router.romb.net (193.254.218.2)              16.841ms
 3: border.romb.net (80.91.172.97)                   48.562ms
 4: L9-KTU.rtr.newline.net.ua (80.91.178.81)         109.070ms
 5: utel-gw.ix.net.ua (195.35.65.89)                 asymm 6 54.850ms
 6: dc-m7i-1-ge.interfaces.dc.utel.ua (213.186.112.129) asymm 7 29.092ms
 7: no reply
 8: no reply
 9: no reply
10: no reply
11: no reply
12: no reply
13: no reply
14: no reply
15: no reply
16: no reply
17: no reply
18: no reply
19: no reply
20: no reply
21: no reply
22: no reply
23: no reply
24: no reply
25: no reply
26: no reply
27: no reply
28: no reply
29: no reply
30: no reply
31: no reply
Too many hops: pmtu 1488

```

Рис. 8.13. Проблема с прохождением пакетов

Если же не открывается какая-то конкретная страница, а все остальные работают нормально, тогда понятно, что причина в самом удаленном сервере, а не в ваших настройках.

8.7. Для фанатов, или настройка сети вручную

Иногда мои книги критикуют за то, что при настройке сети я использую только графические конфигураторы. С одной стороны, такие конфигураторы просты и удобны, — ведь в Windows вы пользуетесь панелью управления, а не редактором реестра, хотя можно изменять сетевые настройки и через regedit. С другой стороны, редактирование конфигурационных файлов позволяет глубже познать Linux. Если вам интересно, в какие файлы сохраняются сетевые настройки после нажатия кнопки **ОК** в окне конфигуратора, тогда этот раздел — для вас. И самое время сейчас снова обратиться к данным табл. 8.2 — осознанно используя эти данные, вы быстро вспомните, какой конфигурационный файл нужно редактировать. Далее в этом разделе мы рассмотрим конфигурационные файлы конкретных дистрибутивов.

А если вы не считаете, что на это нужно тратить свое время (ведь за считанные секунды можно все настроить конфигуратором), можете смело приступать к чте-

нию следующей главы. Хотя, я вовсе не исключаю и такого развития ситуации, — вы с интересом прочитаете этот раздел, но все-таки будете использовать конфигураторы, потому что это сильно упрощает процесс.

8.7.1. Конфигурационные файлы Fedora/CentOS

Начнем с файла `/etc/sysconfig/network` — в нем можно задать имя машины, шлюз по умолчанию и включить IP-переадресацию. Пример этого файла приведен в листинге 8.1.

Листинг 8.1. Файл `/etc/sysconfig/network`

```
NETWORKING=yes
NETWORKING_IPV6=no
HOSTNAME=den.dkws.org.ua
# Дополнительно
DHCP_HOSTNAME=den.dkws.org.ua
GATEWAY=192.168.0.1
GATEWAYDEV=eth0
FORWARD_IPV4 =no
```

В большинстве случаев хватает первых трех параметров:

- параметр **NETWORKING** определяет, будет ли включена поддержка сети. Обычно нужно включить такую поддержку сети (`yes`), поскольку даже функции печати в Linux требуют поддержки сети;
- параметр **NETWORKING_IPV6** включает поддержку IPv6. Поскольку этот протокол еще не используется, то следует задать значение `no`;
- параметр **HOSTNAME** задает имя узла.

В ряде ситуаций могут потребоваться и дополнительные параметры:

- параметр **DHCP_HOSTNAME** задает имя узла при использовании DHCP. Если вы не задали значение параметра **DHCP_HOSTNAME**, то DHCP-сервер может назначить узлу другое имя. Если же значение задано, то DHCP не станет изменять имя узла;
- параметр **GATEWAY** задает шлюз по умолчанию. В этом конфигурационном файле указывать шлюз по умолчанию не обязательно, поскольку его можно указать в файле `/etc/sysconfig/network-scripts/ifcfg-eth0` — конфигурационном файле сетевого интерфейса `eth0`;
- параметр **GATEWAYDEV** указывает имя интерфейса для доступа к шлюзу. Часто этот параметр опускается;
- последний параметр, **FORWARD_IPV4**, позволяет превратить ваш компьютер в шлюз.

После редактирования файла `/etc/sysconfig/network` нужно перейти в каталог `/etc/sysconfig/network-scripts/`, где содержатся конфигурационные файлы для каждого сетевого интерфейса. Например, конфигурация интерфейса `eth0` прописана в файле `/etc/sysconfig/network-scripts/ifcfg-eth0`. Конфигурации интерфейсов могут различаться

в зависимости от того, как настраивается интерфейс: автоматически по DHCP или же сетевая информация присваивается статически. Как правило, на рабочих станциях сетевая информация присваивается автоматически — по DHCP. А вот на серверах (в том числе и на DHCP-сервере) сетевая информация указывается статически — вручную.

В листинге 8.2, а приведена конфигурация интерфейса, настраиваемого по DHCP.

Листинг 8.2, а. Конфигурация интерфейса, настраиваемого по DHCP

```
DEVICE=eth0
BOOTPROTO=dhcp
HWADDR=XX: XX: XX: XX: XX: XX
ONBOOT=yes
TYPE=Ethernet
IPV6INIT=no
```

Здесь параметр `DEVICE` задает имя устройства (`eth0`), параметр `BOOTPROTO` — тип конфигурации (по протоколу DHCP). Параметр `HWADDR` позволяет изменить аппаратный MAC-адрес сетевого адаптера (как правило, этот параметр указывается только тогда, когда нужно изменить MAC-адрес, в обычных же условиях он не нужен). Параметр `ONBOOT` определяет, будет ли «поднят» интерфейс при загрузке (`yes` — да, `no` — нет). Последние два параметра необязательны (`TYPE` — задает тип интерфейса, `IPV6INIT` — включает для этого интерфейса протокол IPv6).

Пример статической настройки интерфейса приведен в листинге 8.2, б.

Листинг 8.2, б. Статическая настройка интерфейса

```
DEVICE=eth0
BOOTPROTO=none
HWADDR=XX:XX:XX:XX:XX:XX
ONBOOT=yes #
NETMASK=255.255.255.0 IPADDR=192.168.0.10
GATEWAY=192.168.0.1
#
NETWORK=192.168.0.0
BROADCASTS92.168.0.255 USERCTL=no
```

Первые четыре параметра нам знакомы. Разница лишь в том, что параметр `BOOTPROTO` содержит значение `none` вместо `dhcp`. Параметр `NETMASK` задает сетевую маску, параметр `IPADDR` — IP-адрес узла, `GATEWAY` — шлюз по умолчанию для этого сетевого интерфейса.

Можно также задать и необязательные параметры: `NETWORK` (адрес сети), `BROADCAST` (широковещательный IP-адрес) и `USERCTL`. Если последний параметр включен (`yes`), то интерфейсом могут управлять не-root пользователи. Обычно в этом нет необходимости, поэтому ему присваивается значение `no`.

С остальными файлами вы знакомы из табл. 8.2:

- `/etc/resolv.conf` — конфигурация DNS (здесь указываются DNS-серверы);
- `/etc/hosts` — статическая таблица поиска имен узлов (применяется, если ваша сеть не использует DNS);
- `/etc/sysconfig/static-routes` — этот файл по умолчанию отсутствует (он содержит список статических маршрутов).

8.7.2. Конфигурационные файлы openSUSE

В openSUSE все конфигурационные файлы, относящиеся к настройкам сети, находятся в каталоге `/etc/sysconfig/network`:

- `/etc/sysconfig/network/ifcfg-имя` — содержит параметры сетевого интерфейса (здесь имя — это имя сетевого интерфейса);
- `/etc/sysconfig/network/ifroute-имя` — содержит маршруты для конкретного интерфейса;
- `/etc/sysconfig/network/routes` — список статических маршрутов;
- `/etc/sysconfig/network/config` — различные переменные.

Основные файлы — это файлы `/etc/sysconfig/network/ifcfg-имя`. Рассмотрим пример файла `/etc/sysconfig/network/ifcfg-eth0`, задающего параметры сетевого интерфейса `eth0` (листинг 8.3).

Листинг 8.3. Файл `/etc/sysconfig/network/ifcfg-eth0`

```
BOOTPROTO='dhcp'
IPADDR=' '
MTU=' '
NAME='79c970 [PCnet32 LANCE]'
NETMASK=' '
NETWORKS '
STARTMODE='auto'
USERCONTROL='no'
```

В файле конфигурации сетевого интерфейса может быть множество самых разных параметров. Все возможные параметры с пояснениями и допустимыми значениями представлены в файле `ifcfg.template`. Сейчас же мы рассмотрим только те параметры, которые показаны в листинге 8.3:

- параметр `BOOTPROTO` задает протокол конфигурации интерфейса. Для автоматического назначения IP-адреса по DHCP используется значение `dhcp`. Если нужно назначить адрес вручную, то используется значение `static`. Есть еще два полезных значения:
 - `autoip` — производится поиск свободного IP-адреса, найденный IP-адрес назначается статически;

- `dhcp+autoip` — основной способ — `dhcp`, но если DHCP-сервер отсутствует, то работает вариант `autoip`;
- назначение следующих параметров ясно: это IP-адрес, размер MTU (Maximum Transmission Unit, максимальный блок передачи), описание устройства (ни на что не влияет), сетевая маска, адрес сети;
- параметр `startmode` задает режим запуска интерфейса:
 - `auto` — автоматический запуск при загрузке системы;
 - `manual` — интерфейс будет «подниматься» вручную;
 - `off` — интерфейс не используется.

Есть и другие режимы запуска — о них вы прочитаете в файле `ifcfg.template`;

- последний параметр запрещает управление интерфейсом не-root пользователям.

Следует упомянуть полезную опцию: `DHCLIENT_SET_HOSTNAME`. Она определяет, будет ли DHCP-клиент изменять имя узла, что полезно, если не нужно изменять имя узла каждый раз при получении нового IP-адреса (значение по).

Также можно установить значение по для опции `DHCLIENT_SET_HOSTNAME` в файле `/etc/sysconfig/network/dhcp`. Разница заключается в том, что в первом случае вы изменяете параметр `DHCLIENT_SET_HOSTNAME` локально — только для конкретного интерфейса, а во втором случае глобально — для всех интерфейсов.

А где же хранится имя узла? Привычного файла `/etc/hostname` я не нашел. Пришлось действовать старым проверенным способом: вызвать конфигуратор, установить имя узла, а потом смотреть, какой файл изменился. Меня ждал небольшой сюрприз. Да, файла `/etc/hostname` нет, но зато есть файл `/etc/HOSTNAME` (все буквы прописные) — этого файла я просто не заметил. В нем и хранятся имя узла и имя домена.

8.7.3. Конфигурационные файлы Debian/Ubuntu

Основной конфигурационный файл Debian/Ubuntu — `/etc/network/interfaces`. В нем можно изменить все: от IP-адреса интерфейса до параметров маршрутизации. Файл `/etc/network/interfaces` подробно описан в моей статье <http://dkws.org.ua/index.php?page=show&file=a/ubuntu/network-interfaces>, и нет смысла ее сюда переписывать.

Параметры некоторых соединений (например, DSL-соединений) также могут храниться в каталоге `/etc/NetworkManager/system-connections`.

Кроме файла `/etc/network/interfaces` вам пригодится и файл `/etc/hostname`, содержащий имя узла.

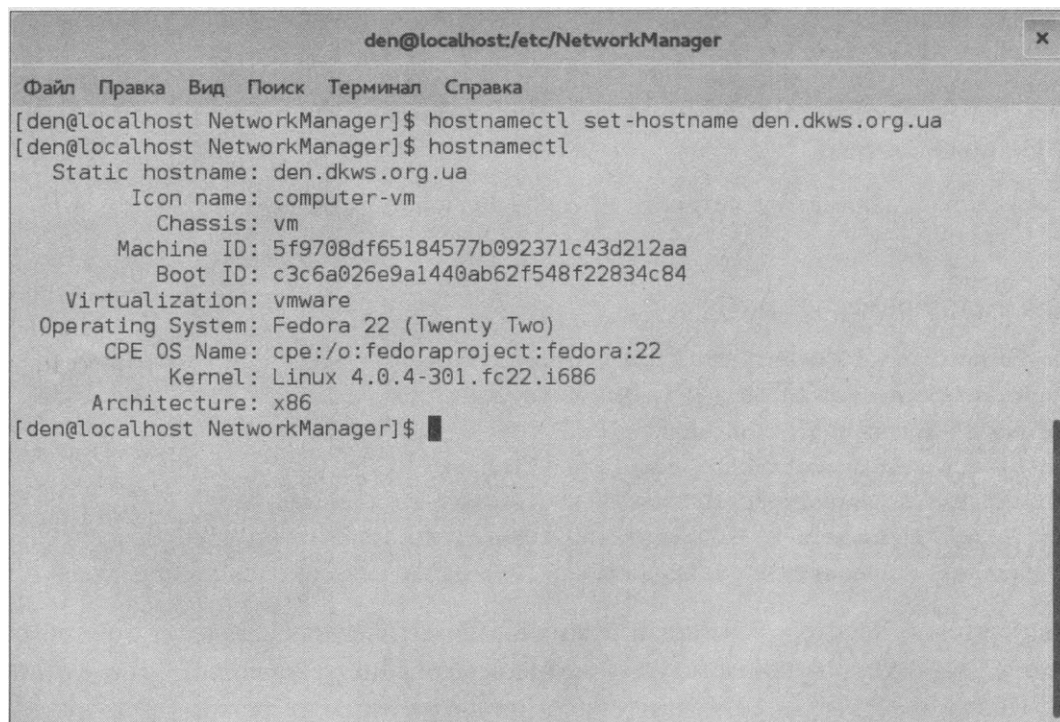
Файл `/etc/resolv.conf`, как и в других дистрибутивах, содержит параметры DNS. Но этот файл перезаписывается системой при перезагрузке. Если у вас рабочая система, то такое поведение — оптимально. А вот на сервере хотелось бы больше контроля. О том, как побороть перезапись этого файла, рассказано в другой моей статье: <http://dkws.org.ua/index.php?page=show&file=a/ubuntu/static-dns-ubuntu9>.

8.7.4. Команда *hostnamectl*

Как изменить имя узла, не редактируя никакие файлы? Для этого в современных дистрибутивах служит команда `hostnamectl`. Вызов этой команды без параметров выводит имя узла и другую информацию о системе (рис. 8.14). Для установки имени узла нужно выполнить команду:

```
hostnamectl set-hostname <имя узла>
```

Удивительно, но в Fedora 22-26 эта команда сработала без прав `root`, что и показано на рис. 8.14.



```
den@localhost:/etc/NetworkManager
Файл Правка Вид Поиск Терминал Справка
[den@localhost NetworkManager]$ hostnamectl set-hostname den.dkws.org.ua
[den@localhost NetworkManager]$ hostnamectl
  Static hostname: den.dkws.org.ua
        Icon name: computer-vm
        Chassis: vm
    Machine ID: 5f9708df65184577b092371c43d212aa
       Boot ID: c3c6a026e9a1440ab62f548f22834c84
  Virtualization: vmware
 Operating System: Fedora 22 (Twenty Two)
    CPE OS Name: cpe:/o:fedoraproject:fedora:22
         Kernel: Linux 4.0.4-301.fc22.i686
   Architecture: x86
[den@localhost NetworkManager]$
```

Рис. 8.14. Fedora: установка параметров узла

Существует также команда `hostname`, но она позволяет лишь изменить имя компьютера до перезагрузки, а после перезагрузки будет восстановлено его старое имя.

8.7.5. Команда *mii-tool*

Современные сетевые адаптеры поддерживают несколько скоростей передачи данных: 10, 100 и 1000 Мбит/с, а также два режима передачи данных: полудуплексный и полнодуплексный.

Помню, настраивал как-то PPPoE-соединение в Windows XP. И оно отказывалось работать на штатной скорости адаптера 100 Мбит/с— происходили постоянные срывы связи через произвольные интервалы времени с момента установки соединения. Пришлось «зажать» сетевой адаптер до скорости 10 Мбит/с— после этого

проблема исчезла. На скорости самого соединения это никак не отразилось, поскольку оно было изначально ограничено провайдером до 5 Мбит/с.

До сих пор для меня загадка, почему все не работало по умолчанию. Возможно, дело в самом сетевом адаптере. А может, даже в коммутаторе. Ведь по умолчанию и сетевая плата, и порт коммутатора находятся в режиме автоматического согласования, когда оба устройства пытаются подобрать совместимые параметры. И как следствие — высокая потеря пакетов. Лучший способ в такой ситуации — зафиксировать скорость и режим работы сетевого адаптера и порта коммутатора.

В Windows изменение скорости и режима работы сетевого адаптера производится в окне изменения его параметров. А в Linux для этого служит команда `mii-tool`. Изменение режима работы порта коммутатора осуществляется через его Web-интерфейс — о том, как это делается, вы сможете прочитать в документации к коммутатору (а дешевые коммутаторы, как правило, вообще не позволяют изменять свои параметры).

Для просмотра параметров сетевого интерфейса выполните команду:

```
# mii-tool -v eth0
```

Вывод будет примерно такой:

```
eth0: negotiated 100baseTx-ED flow-control, link ok
product info: vendor 88:58:43, model 0 rev 0
basic mode: autonegotiation enabled
basic status: autonegotiation complete, link ok
capabilities: 100baseTx-ED 100baseTx-HD 10baseT-ED 10baseT-HD
advertising: 100baseTx-ED 100baseTx-HD 10baseT-ET) 10baseT-HD flow control
link partner: 100baseTx-FD 100baseTx-HD 10baseT-FD 10baseT-HD flow control
```

Сейчас сетевой адаптер работает в режиме автоматического согласования режима (autonegotiation), текущий статус — автосогласование завершено, связь установлена. Поле `capabilities` содержит список поддерживаемых режимов, а поле `link partner` — список режимов, поддерживаемых коммутатором.

Для установки режима используется опция `-force`:

```
# mii-tool -force=режим интерфейс
```

Например:

```
# mii-too -force=10baseT-FD eth0
```

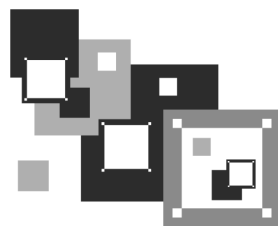
8.8. Еще несколько слов о настройке сети

Напоследок отмечу, что в большинстве случаев вообще сеть настраивать не приходится — ведь DHCP-сервер сейчас не роскошь. Именно из-за этого здесь не рассмотрен конфигуратор сети openSUSE (хотя вы без проблем разберетесь с ним, запустив Центр управления YaST).

Спрашивается тогда, зачем была нужна эта глава, если все настраивается автоматически? Да, обычному пользователю, может, и не обязательно все это знать, а вот администратор обязан разбираться в таких тонкостях. Впрочем, пользователи сейчас немного расслабились, осознав, что Linux — это просто. А расслабляться вредно. И когда требуется присвоить статический IP-адрес (например, при настройке того же DHCP-сервера, который должен иметь статический адрес), они начинают «плавать».

Надеюсь, эта глава полностью заполнила пробел в ваших знаниях по настройке локальной сети в Linux.

ГЛАВА 9



Настройка соединения Wi-Fi

Беспроводные сети Wi-Fi (g/n) стали в последнее время весьма популярны благодаря дешевизне оборудования и стремлению пользователей ко всему мобильному. Соответственно, подключиться к сети Wi-Fi можно даже с помощью мобильного телефона. А DSL-провайдеры все чаще и чаще вместо обычных DSL-модемов устанавливают DSL-маршрутизаторы с функциями Wi-Fi, что очень удобно (описание настройки DSL-соединений вынесено в папку Дополнения сопровождающего книгу электронного архива— см. *приложение*). Такой маршрутизатор монтируется при входе в квартиру и охватывает беспроводным Интернетом всю ее площадь, что позволяет не тянуть кабели внутри квартиры. Даже если у вас не ноутбук, оснащенный адаптером Wi-Fi «из коробки», а стационарный компьютер, можно незадолго до покупки к нему внешний адаптер Wi-Fi и пользоваться всеми преимуществами скоростного беспроводного Интернета.

В этой главе мы не станем подробно рассматривать процесс настройки беспроводной сети (об этом читайте в других моих книгах) — да и не имеет этот процесс прямого отношения к настройке Linux. К тому же, настройки беспроводных маршрутизаторов различаются в зависимости от их производительности и конкретной модели. При этом, если маршрутизатор устанавливал провайдер, а не вы сами, то обычно беспроводная сеть уже настроена, и вам не придется изменять какие-либо настройки маршрутизатора, — достаточно будет только настроить свои компьютеры на подключение к этой беспроводной сети.

9.1. Настройка беспроводного соединения с помощью NetworkManager

Рад вам сообщить, что наконец-то настройка беспроводного соединения в Linux упрощена по максимуму. Вам больше не придется вводить команды, похожие на шаманские заклинания, устанавливать для беспроводных адаптеров эмуляторы Windows-драйверов, бродить по дебрям конфигурационных файлов.

Современные дистрибутивы Linux поддерживают беспроводные адаптеры так же, как и обычные сетевые адаптеры. Поэтому все, что вам нужно сделать, — это оказаться в зоне действия беспроводной сети и ввести пароль, если доступ к сети защищен.

Далее мы рассмотрим процесс настройки беспроводного подключения в программе NetworkManager, которая используется во многих дистрибутивах для управления сетевыми подключениями.

Щелкните на значке NetworkManager, и вы увидите меню, позволяющее управлять беспроводными соединениями. В этом же меню будут перечислены беспроводные сети, в зоне действия которых вы находитесь. В нашем случае в окне менеджера видны две беспроводные сети: **dhsilabs** и **Shtormik** (рис. 9.1).

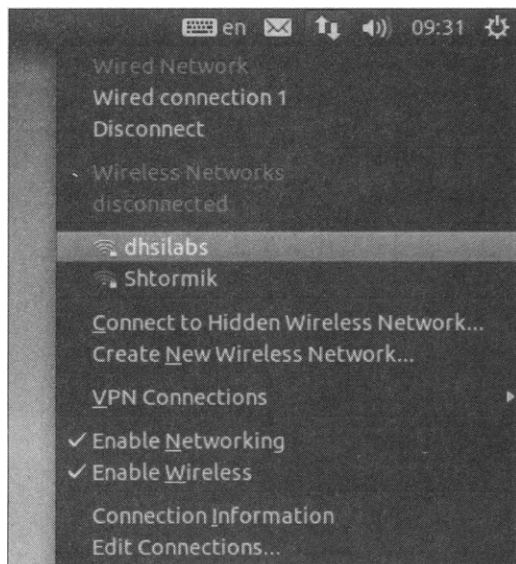


Рис. 9.1. Ubuntu: меню NetworkManager

Возле каждой беспроводной сети отображается индикатор уровня сигнала — чем больше на нем активных полосок, тем ближе вы к беспроводному маршрутизатору. Моя сеть называется **dhsilabs**, ее я и выбрал. Посмотрите внимательно на значок индикатора — если возле него имеется маленький значок замка (смотреть лучше на своем мониторе, на иллюстрации этот замок виден плохо), то сеть закрыта — для доступа к ней нужно ввести пароль (рис. 9.2).

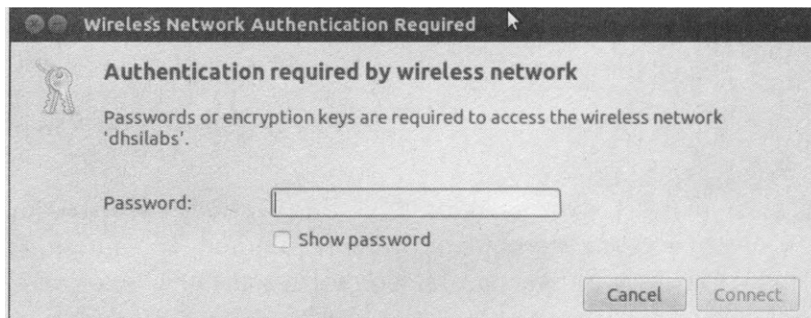


Рис. 9.2. Ubuntu: ввод пароля для доступа к сети

Затем (если введенный пароль правильный!) вы увидите уведомление о подключении к сети (рис. 9.3), а в меню NetworkManager появится команда **Disconnect** (Отключить) для отключения от сети (рис. 9.4).

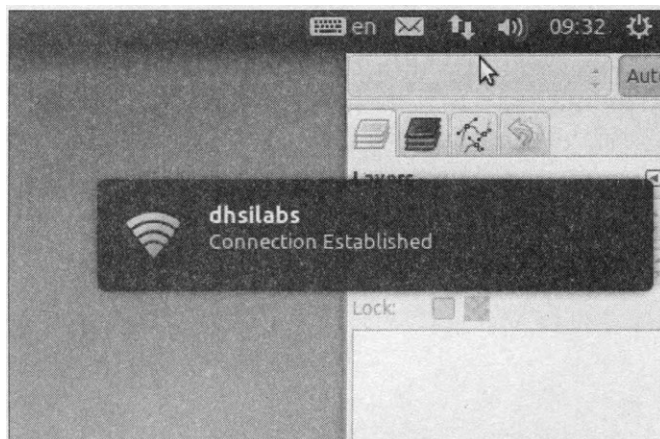


Рис. 9.3. Ubuntu: соединение с беспроводной сетью установлено

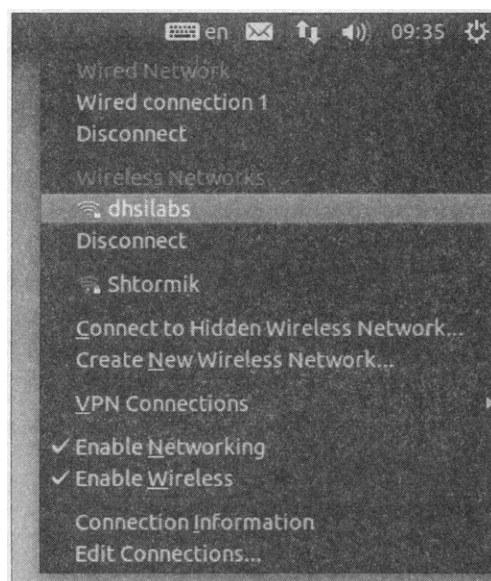


Рис. 9.4. Ubuntu: команда Disconnect

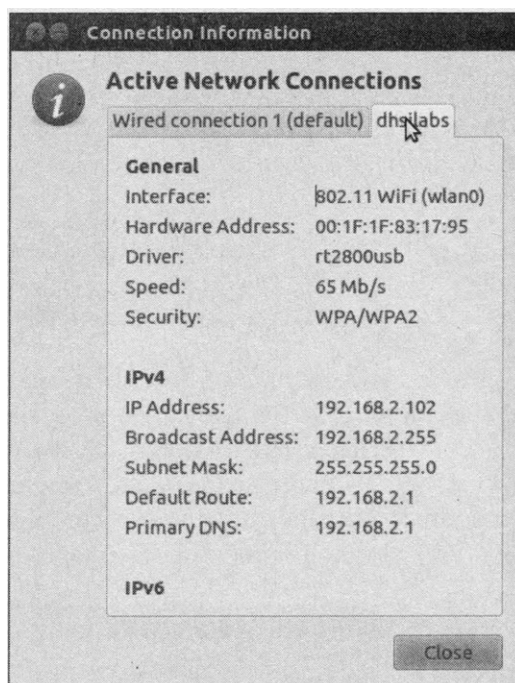


Рис. 9.5. Ubuntu: сведения о соединении

Выбрав из меню NetworkManager команду **Connection Information** (Сведения о соединении), вы сможете просмотреть информацию о вашем беспроводном соединении (рис. 9.5): MAC-адрес беспроводного адаптера, скорость соединения и т. д.

Команда **Edit Connections** (Изменить соединения) позволяет просмотреть и изменить параметры беспроводного соединения— так, на вкладке **Wireless** (Wi-Fi) отображаются беспроводные сети, к которым вы когда-нибудь подключались (рис. 9.6).

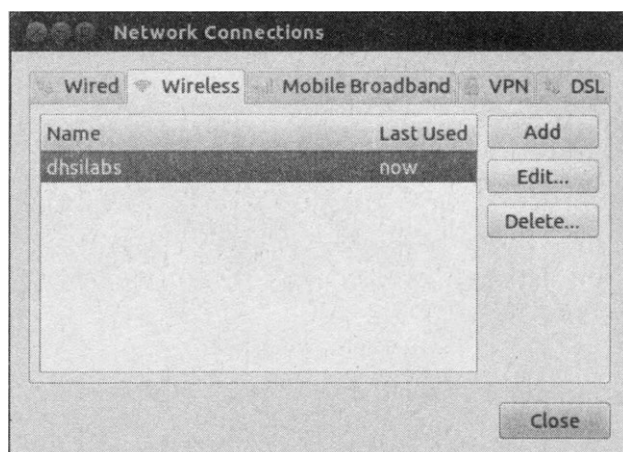


Рис. 9.6. Ubuntu: список беспроводных соединений

Выберите соединение и нажмите кнопку **Edit** (Изменить). Какие параметры нужно изменять? Обычно параметры беспроводного соединения не требуют редактирования, исключение составляет один параметр — пароль доступа к сети, который следует периодически менять, что и можно сделать на вкладке **Wireless Security** (Защита Wi-Fi), показанной на рис. 9.7.

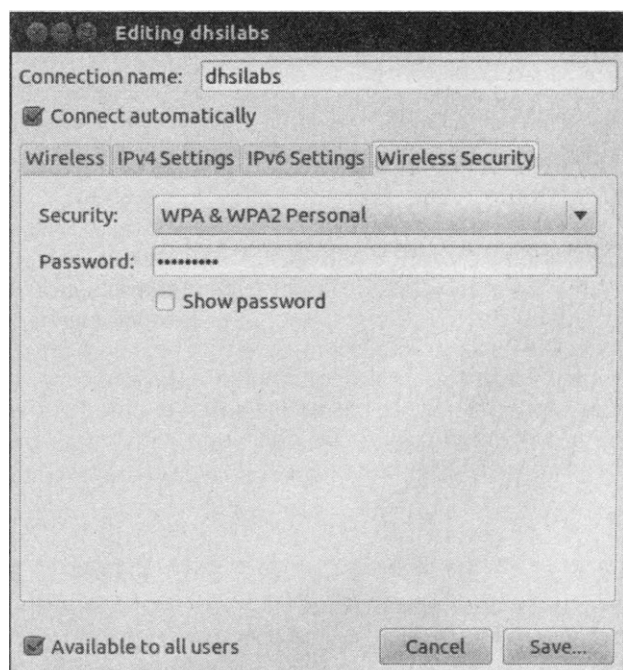


Рис. 9.7. Ubuntu: изменение пароля доступа к беспроводной сети

Параметры всех ваших соединений хранятся в каталоге `/etc/NetworkManager/system-connections`. В нем вы найдете файлы с параметрами конкретного подключения (название файла соответствует названию подключения). В листинге 9.1 приведен файл параметров соединения для моей беспроводной сети.

Листинг 9.1. Файл с параметрами беспроводного подключения

```
[connection]
id=dhsilabs
uuid=c8b546ed-3e12-4960-bc76-82fc3409cf69
type=802-11-wireless

[802-11-wireless]
ssid=dhsilabs
mode=inf ract ructure
mac-address=0:If:If:83:17:95
security=802-11-wireless-security

[802-11-wireless-security]
key-mgmt=wpa-ps k
auth-alg=open
psk=12345678

[ipv4]
method=auto

[ipv6]
method=auto
```

Обратите внимание: пароль доступа к сети задается параметром `psk` (в нашем случае пароль - это строка `12345678`).

ПРИМЕЧАНИЕ

Почему все иллюстрации в этой главе — на английском языке? Мне не сложно выбрать русский язык и сделать иллюстрации русифицированной версии программы, но для некоторых случаев это будет неправильно. Например, Fedora 22 по умолчанию англоязычна, а русский язык можно доустановить только после настройки соединения. То есть, настраивать беспроводное соединение вам придется с помощью англоязычной версии NetworkManager. То же самое касается и Ubuntu, в которой и были сделаны иллюстрации для этой главы. Даже если при запуске выбирается русский язык, окна программы NetworkManager все равно сначала выводятся на английском.

9.2. Что делать, если сети нет в списке?

Вы точно знаете, что находитесь в зоне действия беспроводной сети Wi-Fi, но ее нет в списке NetworkManager. Что делать?

Первым делом нужно еще раз убедиться, что вы действительно находитесь в зоне действия сети. Сделать это достаточно просто. Если вы пытаетесь подключиться к домашней сети, просто убедитесь, что маршрутизатор включен, и вы находитесь недалеко от него, — в большинстве случаев находиться рядом с маршрутизатором не требуется, но счастливым обладателям огромных квартир лучше подойти к нему

поближе, чтобы убедиться, что он включен, и что вы находитесь в зоне его действия.

Затем проверьте, можно ли подключиться к этой сети с других устройств (например, с другого компьютера или мобильного телефона) или в другой операционной системе (например, в Windows). Логика проста — если подключиться удастся, то дело в Linux...

Иногда сети бывают скрытыми, т. е. они функционируют, но широковещание SSID (имени сети) выключено, и поэтому ее не видно в списке. Для подключения к такой сети нужно выбрать команду **Connect to Hidden Wireless Network** (Подключиться к скрытой беспроводной сети) из меню NetworkManager (см. рис. 9.4) и ввести SSID и пароль для доступа к ней.

Самый плохой случай, когда в других ОС подключиться получается, а в Linux — нет. Такие случаи довольно редки для современных дистрибутивов, и это означает, что, скорее всего, в Linux нет драйвера для вашего беспроводного адаптера. Что ж, вам придется поискать в Интернете инструкции по настройке *вашего* беспроводного адаптера Wi-Fi в *вашем* дистрибутиве Linux. Сей процесс в книге не рассматривается, поскольку он, к сожалению, будет различен для каждого беспроводного адаптера.

9.3. Точка доступа Wi-Fi на смартфоне

Ну, и еще один момент. Если вы читали предыдущие издания этой книги, то вам, наверное, интересно, куда подевался материал о настройке в Linux соединений GPRS/EDGE/3G. Дело в том, что такая настройка сейчас неактуальна. Практически у всех пользователей Linux есть смартфон на базе Android (по сути Android — тот же Linux, и так уж получилось, что пользователи iPhone с iOS на борту выбирают себе компьютеры на Mac OS, но никак не на Linux). А в любом современном Android-смартфоне есть возможность включить точку доступа Wi-Fi. В результате ваше мобильное 3G/EDGE-соединение будет «расшариваться» по Wi-Fi, и никакого другого оборудования для этого не потребуется.

Для настройки в смартфоне точки доступа Wi-Fi перейдите в меню Настройки | Подключения (рис. 9.8).

В разделе Точка доступа и модем перейдите в раздел Мобильная точка доступа (рис. 9.9) — появится экран с параметрами доступа: именем точки (SSID) и паролем (рис. 9.10).

Все, что вам остается, — включить эту функцию с помощью соответствующего переключателя. Используя указанные параметры, вы сможете настроить компьютер под управлением Linux на соединение с этой точкой доступа.

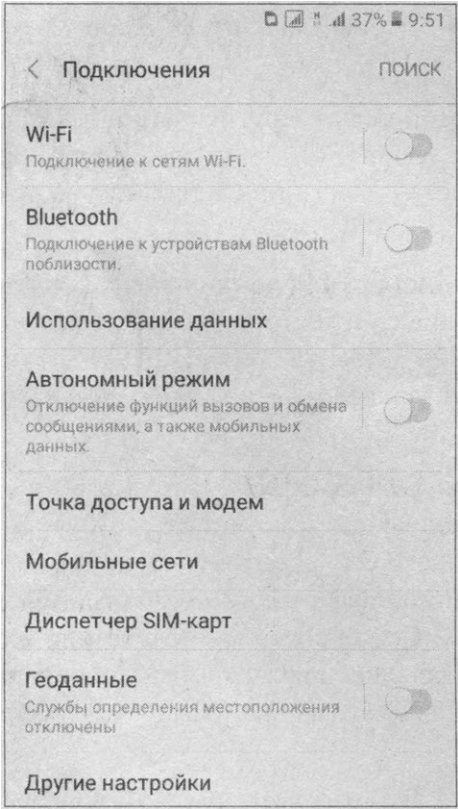


Рис. 9.8. Android 6.1: меню Подключения

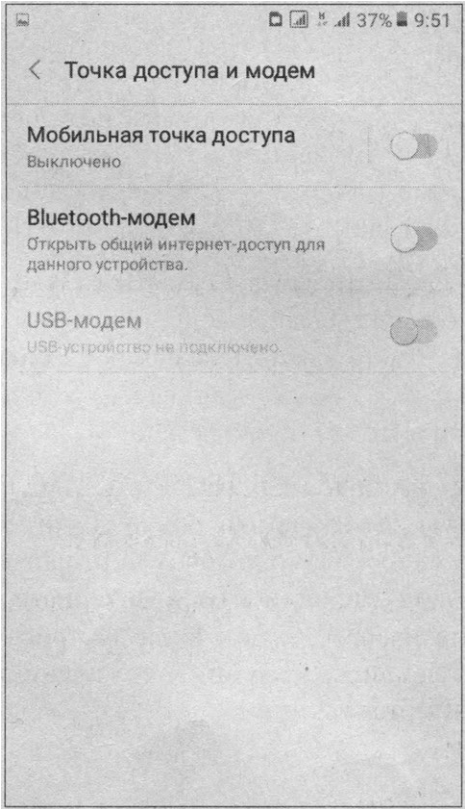


Рис. 9.9. Точка доступа и модем



Рис. 9.10. Параметры мобильной точки доступа

ГЛАВА 10



Настройка VPN-соединения

10.1. Вкратце о выборе VPN-сервера и тарифного плана

В последнее время многие из нас по тем или иным причинам задумываются о защите своего соединения с Интернетом. Если вы часто путешествуете или ездите в командировки, вам иногда приходится подключаться к незащищенной сети через Wi-Fi отеля или кафе, поэтому наличие VPN-доступа для вас будет очень актуальным.

VPN – ВИРТУАЛЬНАЯ ЧАСТНАЯ СЕТЬ

Википедия следующим образом определяет VPN (англ. Virtual Private Network, виртуальная частная сеть) — обобщенное название технологий, позволяющих обеспечить одно или несколько сетевых соединений (логическую сеть) поверх другой сети (например, Интернет).

В Интернете есть множество различных VPN-сервисов. Все они работают одинаково — вы подключаетесь к VPN-серверу, и после этого весь ваш трафик передается по зашифрованному туннелю. Даже если он и будет перехвачен злоумышленником, расшифровать его ему все равно не получится (а если и получится, то лет через пять, когда информация потеряет актуальность).

Какой из VPN-сервисов выбрать? При выборе VPN-сервиса нужно учитывать следующие факторы:

- *скорость работы* — скорость доступа к Интернету через VPN будет ниже, чем напрямую, но насколько — зависит от VPN-сервиса. Как правило, все платные сервисы предоставляют вполне нормальную скорость доступа к Сети. Во всяком случае, ее хватает для полноценного Web-серфинга, просмотра видео онлайн, видеоразговоров в Skype. А вот при работе через бесплатные сервисы скорость доступа может оказаться низкой;
- *ограничения на передачу трафика* — бесплатные (или условно-бесплатные) сервисы часто имеют ограничения на объем передаваемых данных. Например, популярный сервис Security KISS при использовании бесплатного аккаунта разрешает передачу всего 300 Мбайт в день;

- *ограничения на загрузку файлов* — некоторые бесплатные сервисы имеют ограничения на размер загружаемых файлов. Например, используя OperaVPN, вы не сможете загрузить файл размером более 15 Мбайт¹;
- *ограничения на использование портов* — некоторые VPN-сервисы закрывают определенные порты, например, порты отправки почты 25, 465 и т. д. Это означает, что почту можно будет отправлять только с помощью Web-интерфейса, что не позволит воспользоваться для переписки привычной вам почтовой программой. И хорошо, если у вашего почтового ящика есть Web-интерфейс. А вот если его нет, это создаст вам реальные неудобства;
- *стоимость использования* — здесь все зависит от ваших финансовых возможностей.

Если вы хотите воспользоваться услугами полностью бесплатного VPN-сервиса, могу порекомендовать проект FreeOpenVPN (<https://www.freeopenvpn.org/>), на сайте которого публикуются списки публичных (бесплатных) VPN-сервисов. Однако вы должны помнить, что бесплатные сервисы не гарантируют сохранности ваших данных. То есть нет гарантии, что сам бесплатный VPN-сервис не станет перехватывать ваши пароли и другую конфиденциальную информацию. Поэтому я не рекомендовал бы вам использовать такие сервисы. Впрочем, если ваша цель — только скрыть свой IP-адрес, то можно попробовать использовать и эти сервисы.

Далее будет показано, как настроить соединение с крупнейшим VPN-провайдером SecurityKISS (<https://securitykiss.com/>). К преимуществам этого сервиса можно отнести его надежность — защищаемые вами данные точно не будут переданы третьей стороне, а также наличие бесплатного аккаунта. Но есть у него и недостатки:

- *стоимость* — самый доступный тариф OLIVINE обойдется вам в 3 евро в месяц, при этом вам будет установлено ограничение в 20 Гбайт трафика в месяц, то есть примерно 682 Мбайт в день. У бесплатного аккаунта этого сервиса, как уже отмечалось ранее, ограничение составляет 300 Мбайт в день.

Что 300 Мбайт, что 682,— этого мало для полноценной работы в Интернете. С другой стороны, все зависит от того, что вы там делаете. Например, сегодня за 10 часов (с 7:00 по 17:00) я израсходовал примерно 450 Мбайт трафика. При этом я не слушал музыку онлайн, не смотрел видео онлайн, у меня не было видеоразговоров в Skype, но был один голосовой Skype-разговор длительностью 3 минуты. Остальное — это Web-серфинг по интересующим сайтам и работа с почтой. Вполне может быть, что вам при таком виде работы 680 Мбайт будет достаточно.

Впрочем, всегда можно купить тариф MALACHITE (всего за 4 евро в месяц), и вам будет предоставлено 30 Гбайт трафика в месяц, или ровно 1 Гбайт в день. Максимальный тариф EMERALD, где нет ограничений на размер передаваемого трафика, обойдется вам в 10 евро в месяц. Как видите, тарифы у SecurityKISS весьма демократичные, но платить все равно придется;

¹ По состоянию на 15 августа 2017 года это ограничение снято, но временно или постоянно — пока не известно.

- *ограниченное использование почтовых программ*— если вы привыкли читать почту почтовой программой, например, Thunderbird, то вам придется потратить как минимум на тариф MALACHITE (4 евро в месяц), поскольку в тарифах FREE и OLIVINE отправить почту с помощью почтовой программы не получится. Впрочем, если у вашего почтового ящика есть Web-интерфейс, это не составит вам проблемы.

На мой взгляд, оптимальным является тарифный план JADEITE, согласно которому с ежемесячной платой в 6 евро (или за 53,9 евро при оплате за год) вам предоставляется 50 Гбайт в месяц. При этом вам сохраняется возможность использования привычных почтовых программ, а средний расход трафика в день будет порядка 1700 Мбайт — этого, полагаю, хватит с лихвой даже для видеозвонков и просмотра видео онлайн.

VPN-сервис SecurityKISS имеет и мобильное приложение, которое можно установить на смартфон, с тарифом, например, FREE. Сужу по своему опыту — я редко использую более 2 Гбайт мобильного трафика (3G) в месяц, поэтому 300 Мбайт в день мне вполне хватает.

КОРПОРАТИВНЫЙ VPN-СЕРВИС

Конечно, если ваша организация предоставляет сотрудникам собственный VPN-сервис, то проблема выбора для вас решена сама по себе. А в *седьмой части* книги будет показано, как настроить VPN-сервер собственноручно (см. *главу 46*).

10.2. Настройка VPN-подключения

Настройку VPN-подключения мы рассмотрим на примере провайдера SecurityKISS и в дистрибутиве Ubuntu, как одном из самых популярных на настольных компьютерах.

Первым делом нужно удостовериться, что вы используете нужные DNS-серверы. Как бы там ни было, но вам следует установить либо DNS-серверы Google: 8.8.8.8 и 8.8.4.4, либо OpenDNS: 208.67.222.222 и 208.67.220.220.

Для этого установите пакет `resolvconf` и отредактируйте файл `/etc/resolvconf/resolv.conf.d/base` (рис. 10.1), добавив в него эти DNS-серверы (листинг 10.1).

Листинг 10.1. Файл `/etc/resolvconf/resolv.conf.d/base`

```
nameserver 8.8.8.8
nameserver 8.8.4.4
```

Затем настройте систему на использование пакета `resolvconf` для разрешения доменных имен:

```
sudo resolvconf -u
/etc/init.d/network-manager force-reload
```

Теперь все готово к настройке самого VPN-соединения. Зайдите на сайт <https://securitykiss.com/> и зарегистрируйтесь. Перейдите в раздел **Download | Linux**



Рис. 10.1. Ubuntu: установка пакета resolvconf и редактирование его конфигурационного файла

(рис. 10.2) и загрузите конфигурацию для Network Manager (рис. 10.3). Конкретную ссылку я здесь не привожу, поскольку она будет разной для разных пользователей.

Распакуйте загруженный файл. В нем вы найдете файл README.txt с данными шлюзов SecurityKISS. Выберите лучший для себя шлюз. Обычно есть смысл выбирать ближайший (географически) к своему местоположению шлюз, но можно поэкспериментировать с разными шлюзами. Далее мы будем использовать следующий шлюз:

```
US Los Angeles 23.19.26.250 udp 5000
```

Теперь нужно установить плагин Network Manager для OpenVPN:

```
sudo apt install network-manager-openvpn
```

Перезапустите сеть и Network Manager:

```
sudo service network-manager restart
sudo service networking restart
```

Если команда настройки VPN не появится в меню Network Manager, установите еще один пакет:

```
sudo apt-get install network-manager-openvpn-gnome
```

После этого опять перезагрузите сеть и Network Manager.

Если после перезапуска сети она пропала...

Если после перезапуска сети она у вас пропала (сообщение: Network is unreachable), выберите в Network Manager используемое вами соединение (например, Wi-Fi) и подключитесь заново!

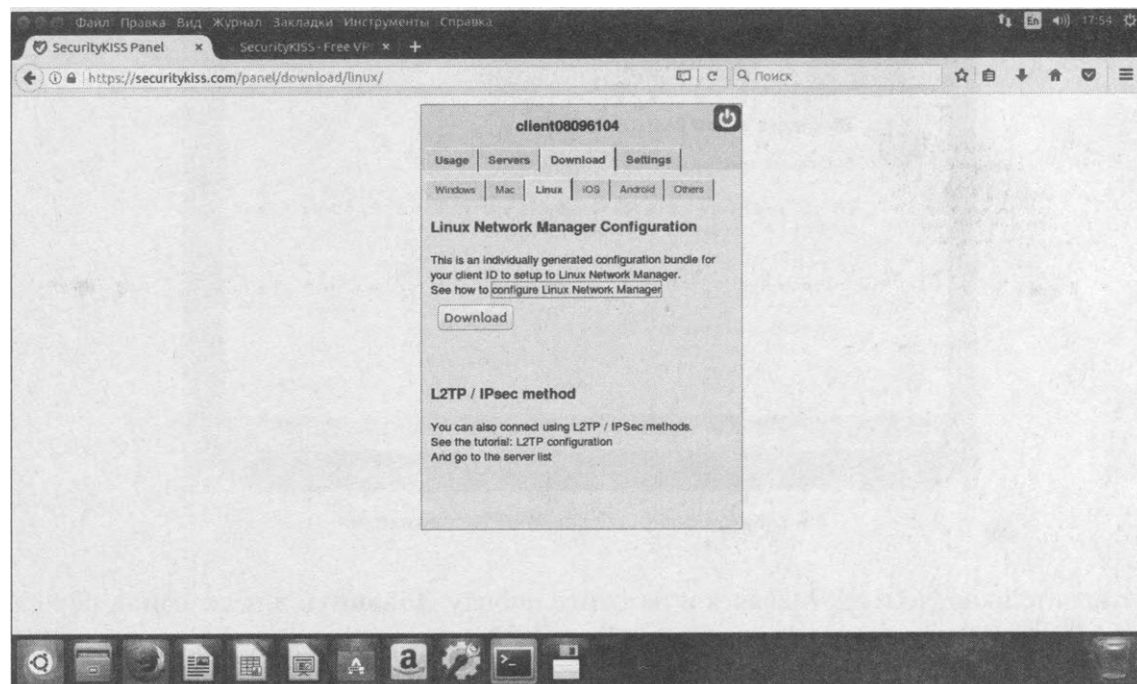


Рис. 10.2. Ubuntu: раздел загрузок конфигурации

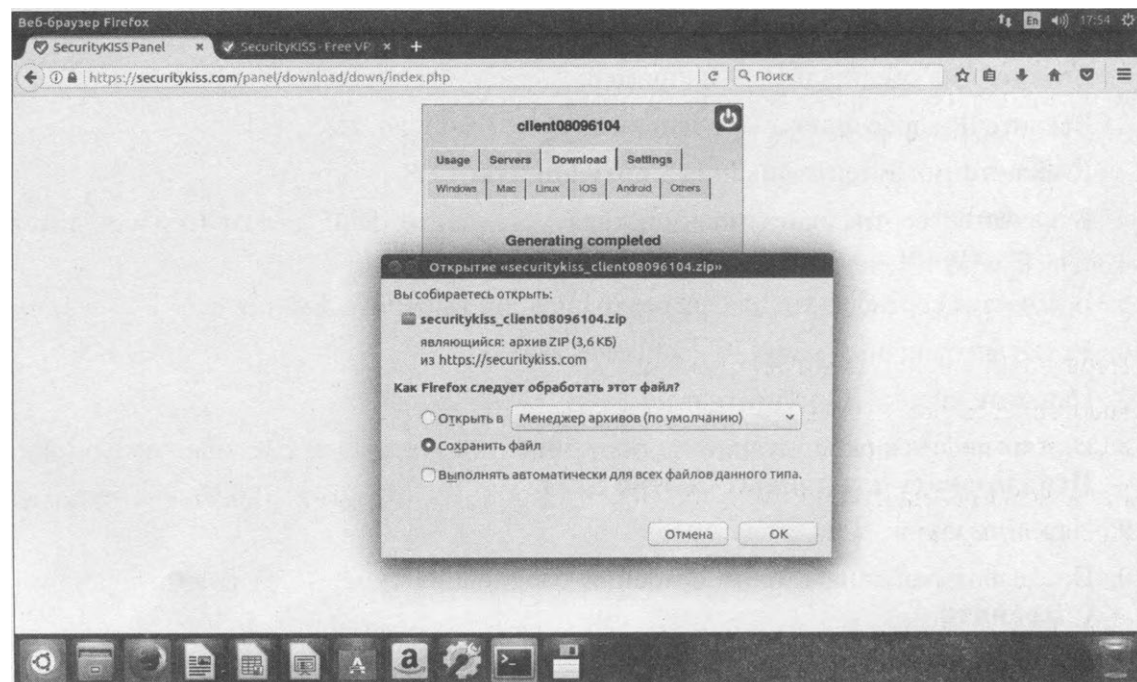


Рис. 10.3. Ubuntu: загрузка файла

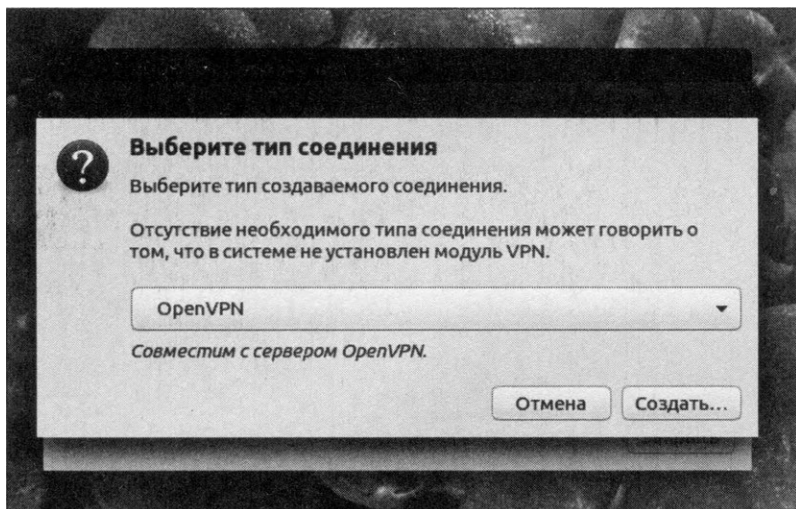


Рис. 10.4. Ubuntu: создание VPN-соединения

Откройте окно Network Manager и нажмите кнопку **Добавить** для создания нового соединения. Выберите тип соединения OpenVPN (рис. 10.4).

Затем перейдите в папку, в которую вы распаковали настройки OpenVPN, и откройте файл с IP-адресами шлюзов. Выбирая IP-адрес, вы должны понимать, что именно этот IP-адрес будет появляться в логах сайтов, которые вы будете посещать.

Выполните следующие действия:

1. Введите имя соединения — например: `securityKISS`.
2. Введите IP-адрес шлюза — в нашем случае: `23.19.26.250`.
3. Выберите тип аутентификации **Сертификаты TLS**.
4. В качестве сертификата пользователя установите файл `cert.crt` (он находится в загруженном архиве).
5. В качестве сертификата центра сертификации выберите файл `ca.crt`.
6. Установите личный ключ — файл `client.key`.
7. Нажмите кнопку **Дополнительно**.
8. В открывшемся окне установите порт 5000 (рис. 10.5), а также включите опцию **Использовать для данных сжатие LZO**.
9. Нажмите кнопку **ОК**.
10. После возвращения в окно создания соединения (рис. 10.6) нажмите кнопку **Сохранить**.
11. В меню Network Manager выберите созданное соединение для подключения (рис. 10.7).
12. Если вы все сделали правильно, вы увидите уведомление об успешном подключении (рис. 10.8). Откройте браузер и перейдите на страничку <https://>

2ip.ru, чтобы узнать свой теперешний IP-адрес. Он должен быть таким, кото
рый вы выбрали из файла README.txt (рис. 10.9).
На этом все... Как видите, в установке VPN-соединения нет ничего сложного.

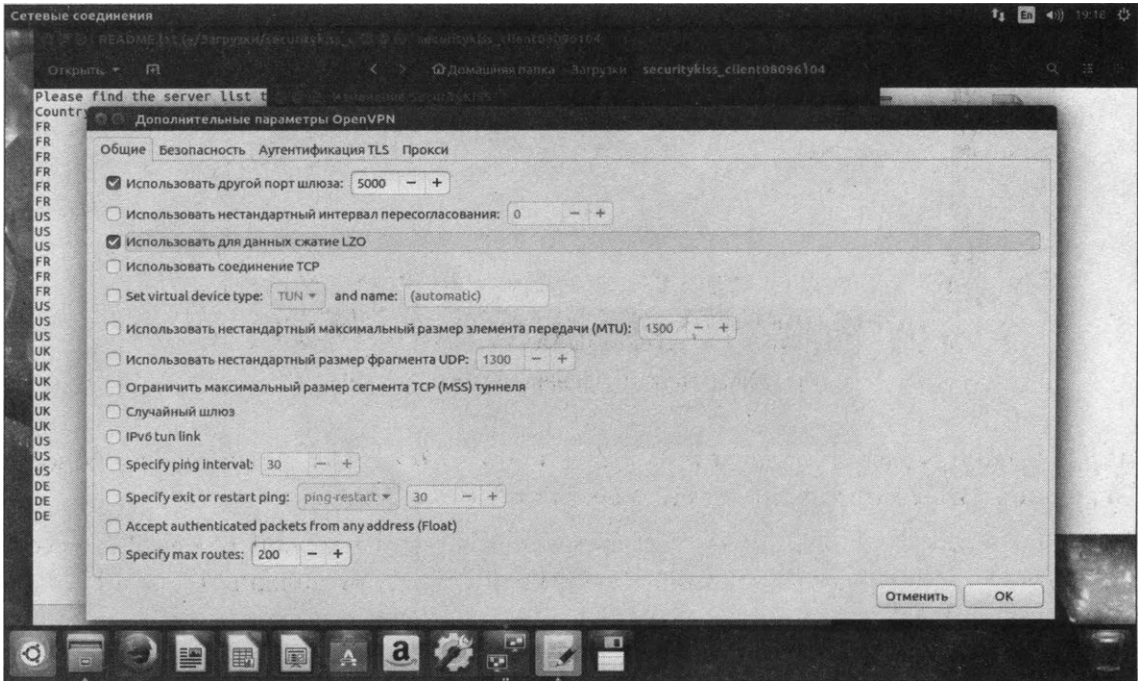


Рис. 10.5. Ubuntu: дополнительные параметры

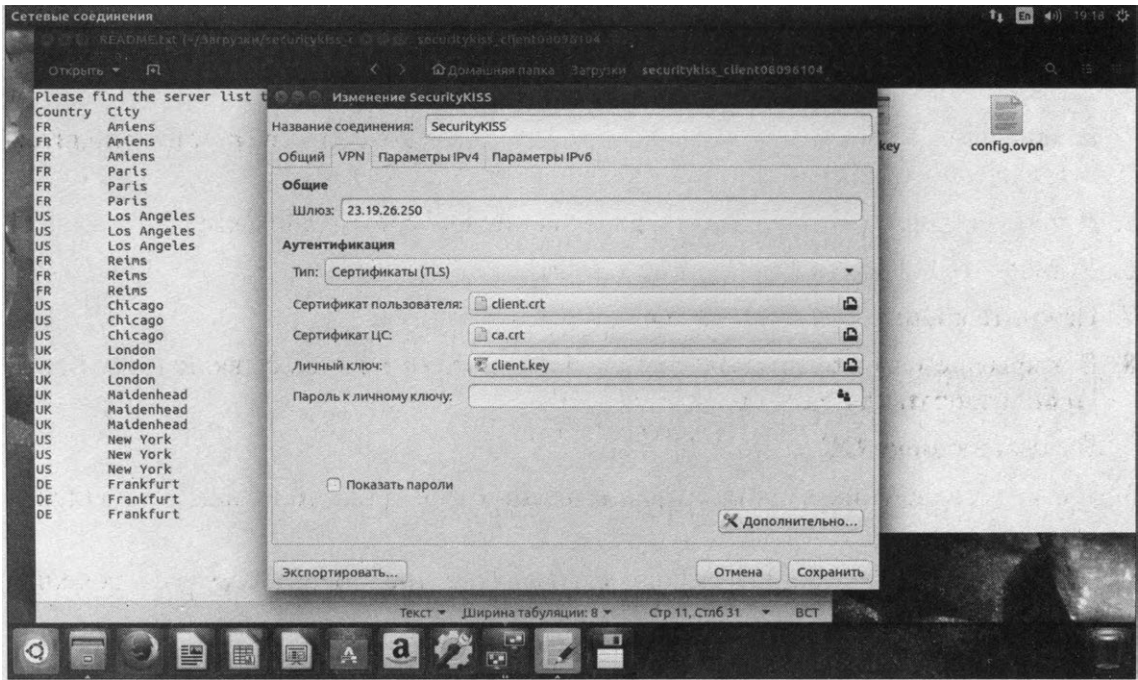


Рис. 10.6. Ubuntu: параметры VPN-соединения

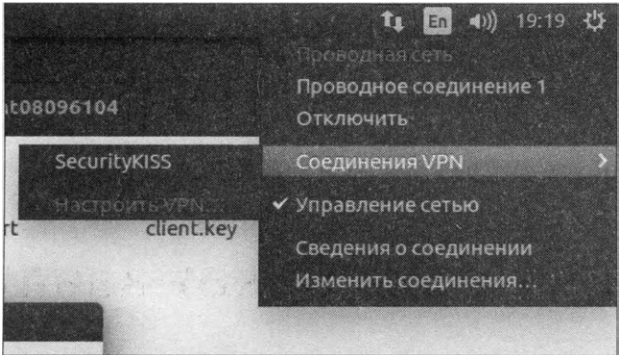


Рис. 10.7. Ubuntu: выбор соединения

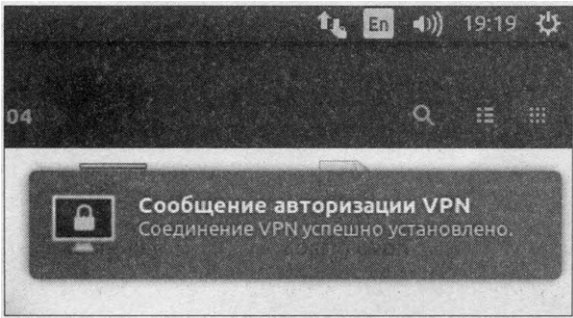


Рис. 10.8. Ubuntu: уведомление об успешном подключении

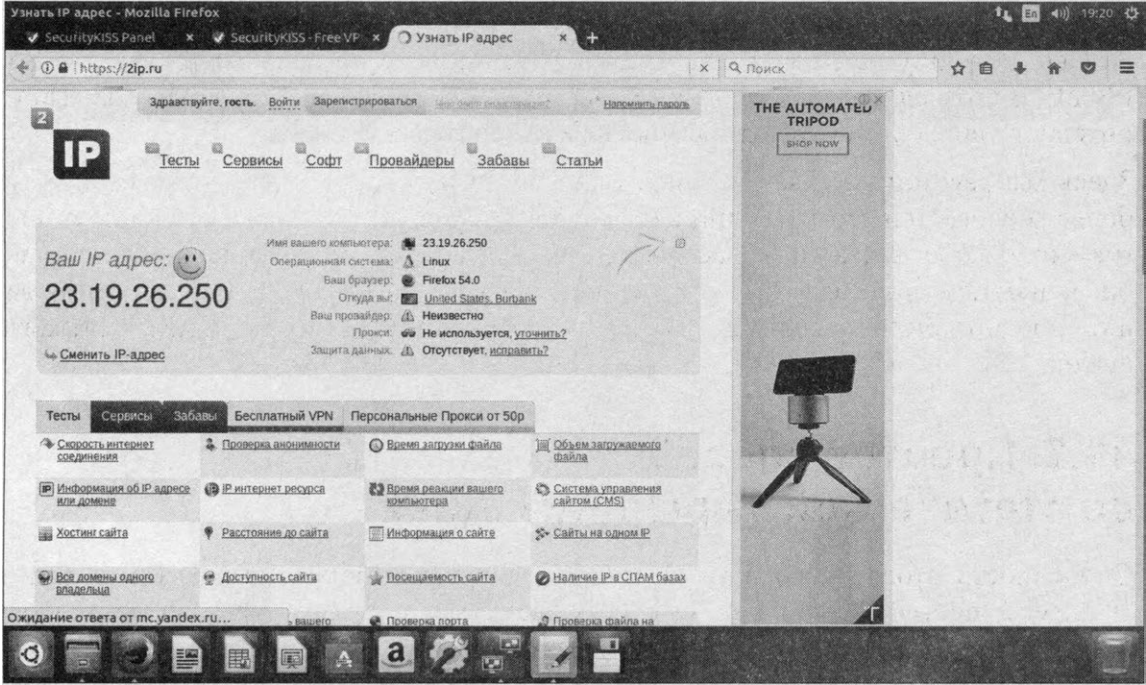
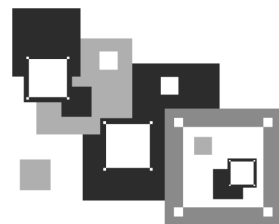


Рис. 10.9. Ubuntu: IP-адрес скрыт

ГЛАВА 11



Объединение интернет-каналов

11.1. Цели и средства решения задачи

Представим, что существуют два или более канала для доступа к Интернету, работающих на разных интерфейсах. Если объединить эти каналы, то можно увеличить скорость доступа к Интернету, а также повысить отказоустойчивость интернет-соединения.

Приведенное в этой главе решение будет полезно не только предприятиям, где особенно важна отказоустойчивость соединения, но и обычным пользователям, которые хотели бы увеличить пропускную способность, используя несколько каналов доступа к Интернету.

Обратите внимание, что интернет-каналы должны находиться на разных интерфейсах, и это важно, — у вас может быть один интерфейс и два интернет-канала, которые вы используете поочередно: доступ к Интернету по локальной сети и через PPPoE. В этом случае вам понадобится еще один сетевой адаптер, чтобы каждый из интернет-каналов работал на собственном интерфейсе.

Здесь мы рассмотрим два решения: одно более простое, второе — посложнее, но более гибкое. При этом ни один из способов не требует установки стороннего программного обеспечения — настройка осуществляется стандартными средствами операционной системы. Выбор решения зависит от поставленной задачи и от ваших предпочтений — сначала ознакомьтесь с обоими способами, а затем сделайте выбор.

11.2. Простой способ со статической маршрутизацией

Особенность этого способа — статическая маршрутизация, позволяющая задать IP-адреса, доступ к которым будет осуществляться только через определенного провайдера.

Прежде всего нужно отредактировать файл `/etc/iproute2/rt_tables`, в котором описываются таблицы для каждого из провайдеров (листинг 11.1).

Листинг 11.1. Файл /etc/iproute2/rt_tables

```
# Не изменяйте эти значения
255 local
254 main
253 default
0 unspec
#
# local
#
#1 inr.ruhep
# Таблицы интернет-провайдеров
1 ISP1
2 ISP2
```

Затем создайте файл /etc/iproute2/ISP1.txt и перечислите в нем IP-адреса, путь к которым должен проходить строго через провайдера ISP1 (по одному адресу в строке).

Далее создайте сценарий /etc/iproute2/balance.sh (листинг 11.2). Ясное дело, IP-адреса и другие переменные в нем нужно соответственно исправить.

Листинг 11.2. Сценарий /etc/iproute2/balance.sh

```
#!/bin/sh
ISP1="/etc/iproute2/ISP1.txt"
# Локальная сеть
local_eth=eth1          # Интерфейс
local_ip=192.168.1.1    # IP-адрес
local_net=192.168.1.0/24# Подсеть

# Сеть локального провайдера
li_net=10.0.0.0/8

# Параметры ISP1
i1_eth=eth0
i1_ip=1.2.3.104
i1_net=1.2.3.0/24
i1_gw=1.2.3.1

# Параметры ISP2
i2_eth=eth2
i2_ip=2.2.2.222
i2_net=2.2.2.0/16
i2_gw=2.2.0.1

# Таблицы маршрутизации iproute2 (нужно указать номера, которые указаны
# в файле /etc/iproute2/rt_tables)
```



```
table1=1
table2=2

# Сбрасываем iptables
iptables -t mangle -F NEW_OUT_CONN
iptables -t mangle -F PREROUTING
iptables -t mangle -F OUTPUT
iptables -t mangle -X NEW_OUT_CONN
ip route flush table $table2
ip rule del table $table2
ip route flush table $table1
ip rule del table $table1
ip route flush cache

# Установка новых правил
iptables -t mangle -N NEW_OUT_CONN
iptables -t mangle -A NEW_OUT_CONN -j CONNMARK --set-mark 1
iptables -t mangle -A NEW_OUT_CONN -m statistic --mode random --probability
0.50 -j RETURN
iptables -t mangle -A NEW_OUT_CONN -j CONNMARK --set-mark 2

# Обработка адресов из файла ISP1.txt
for file in $ISP1; do
if [ -f "$file" ]; then
{ cat "$file" ; echo ; } | while read ip_addr; do
if [ "$ip_addr" != "" ]; then
echo "Static routing for $ip_addr"
iptables -t mangle -A NEW_OUT_CONN -d $ip_addr -j CONNMARK --set-mark 1
fi
done
fi
done

iptables -t mangle -A PREROUTING -d $local_net -j RETURN
iptables -t mangle -A PREROUTING -d $li_net -j RETURN

iptables -t mangle -A PREROUTING -s $local_net -m state --state new,related -j
NEW_OUT_CONN
iptables -t mangle -A PREROUTING -s $local_net -j CONNMARK --restore-mark

iptables -t mangle -A OUTPUT -d $local_net -j RETURN
iptables -t mangle -A OUTPUT -d $li_net -j RETURN

iptables -t mangle -A OUTPUT -s $local_net -m state --state new,related -j
NEW_OUT_CONN
iptables -t mangle -A OUTPUT -s $li_net -j CONNMARK --restore-mark
```



```

ip route add $local_net dev $local_eth scope link table $table1
ip route add      $i2_netdev $i2_ethscope link table $table1
ip route add      $i1_netdev $i1_ethscope link src $i1_ip table $table1
ip route add 127.0.0.0/8 dev lo scope link table $table1
ip route add default via $i1_gw table $table1

ip rule add prio 51 fwmark 1 table $table1
ip rule add from $i1_ip table $table1

ip route add $local_net dev $local_eth scope link table $table2
ip route add      $i1_netdev $i1_ethscope link table $table2
ip route add      $i2_netdev $i2_ethscope link src $i2_ip table $table2
ip route add 127.0.0.0/8 dev lo scope link table $table2
ip route add default via $i2_gw table $table2

ip rule add prio 52 fwmark 2 table $table2
ip rule add from $i2_ip table $table2

ip route flush cache

```

После создания и редактирования сценария назначьте ему право выполнения и запустите. Его также нужно добавить в сценарии загрузки системы, чтобы не запускать его при каждой перезагрузке.

Обратите внимание, что в этом сценарии мы прописываем статические IP-адреса. Если у вас динамические IP-адреса, сценарий требует модификации, — нужно будет вычислить IP-адреса, которые присвоены DHCP каждому интерфейсу (см. пример из листинга 11.3).

11.3. Сложный способ с гибкой настройкой отказоустойчивости

Это решение больше подойдет пользователям, желающим обеспечить отказоустойчивость доступа к Интернету. Способ сложнее представленного в *разд. 11.2* и не предполагает статической маршрутизации, хотя вы можете организовать ее по образцу и подобию предыдущего способа, — нужно будет лишь незначительно модифицировать его код.

Прежде всего, как обычно, правим файл `/etc/iproute2/rtTables`. Он будет таким же, как в предыдущем способе (см. листинг 11.1). Затем в каталоге `/etc/iproute2` нужно создать файл `config`, в котором прописать различные переменные (листинг 11.3).

Листинг 11.3. Файл `/etc/iproute2/config`

```

#!/bin/bash

# Локальный интерфейс ("смотрит" в локальную сеть)
IF0="eth0"

```

```
# Интерфейс к провайдеру ISP1
IF1="eth1"

# Интерфейс к провайдеру ISP2
IF2="ppp0"

# IP-адрес для первого провайдера задаем статично, для второго - используется
# протокол DHCP, поэтому нам нужно вычислить IP при каждом запуске сценария
IP1="1.2.3.xx"
IP2="'ip addr show $IF2 | grep inet | awk '{print $2}''"

# шлюз 1
GW1="1.2.3.1"
# шлюз 2
GW2="2.2.2.1"

# Маска локальной сети
LOCAL_NET="192.168.0.0/24"
# Маска сети провайдера ISP1
ISP1_NET="194.9.xx.xx/xx"
# Маска сети провайдера ISP2
ISP2_NET="195.5.xx.xx/xx"

# Таблицы маршрутизации
TBL1="ISP1"
TBL2="ISP2"

# Относительный "вес" каналов (второй канал более важный)
W1="1"
W2="2"
```

Теперь создадим сценарий `/etc/iproute2/routing`, устанавливающий все необходимые маршруты и правила iptables (листинг 11.4). После редактирования сценария не забываем Сделать его **ИСПОЛНИМЫМ**: `chmod +x routing`.

Листинг 11.4. Сценарий `/etc/iproute2/routing`

```
#!/bin/bash
# Импортируем файл конфигурации
. /etc/iprouter2/config

# Включаем IPv4 Forwarding
echo "1" > /proc/sys/net/ipv4/ip_forward

# Устанавливаем правила маршрутизации и брандмауэра
ip route add $ISP1_NET dev $IF1 src $IP1 table $TBL1 > /dev/null 2>&1
ip route add default via $GW1 table $TBL1 > /dev/null 2>&1
```

```

ip route add $ISP2_NET dev $IF2 src $IP2 table $TBL2 > /dev/null 2>&1
ip route add default via $GW2 table $TBL2 > /dev/null 2>&1

ip route add $ISP1_NET dev $IF1 src $IP1 > /dev/null 2>&1
ip route add $ISP2_NET dev $IF2 src $IP2

ip route add default via $GW1 > /dev/null 2>&1

ip rule add from $IP1 table $TBL1 > /dev/null 2>&1
ip rule add from $GW2 table $TBL2 > /dev/null 2>&1

ip route add $LOCAL_NET dev $IF0 table $TBL1 > /dev/null 2>&1
ip routeadd      $ISP2_NET dev $IF2      table$TBL1>/dev/null 2>&1
ip routeadd      127.0.0.0/8 dev lo      table$TBL1>/dev/null 2>&1
ip route add $LOCAL_NET dev $IF0 table $TBL2 > /dev/null 2>&1
ip routeadd      $ISP1_NET dev $IF1      table$TBL2>/dev/null 2>&1
ip routeadd      127.0.0.0/8 dev lo      table$TBL2>/dev/null 2>&1

iptables -t nat -F POSTROUTING
iptables -t nat -A POSTROUTING -s $LOCAL_NET -o $IF1 -j MASQUERADE
iptables -t nat -A POSTROUTING -s $LOCAL_NET -o $IF2 -j MASQUERADE

```

Осталось самая малость— написать сценарий, который будет проверять работоспособность того или иного канала и в случае необходимости менять шлюз по умолчанию. Сценарий работает просто: он отправляет пять «пингов» подряд, и, если нет ответа, считается, что канал не работает, и он исключается из таблицы маршрутизации. Код сценария приведен в листинге 11.5.

Листинг 11.5. Сценарий `/etc/iproute2/test_connect`

```

#!/bin/bash

# Подключаем конфигурацию
. /etc/iproute2/config

OLDIF1=0
OLDIF2=0

# Настраиваем маршрутизацию
. /etc/iproute2/routing
while true; do
ping -c 5 -s 100 $GW1 -I $IF1 > /dev/null
if [ $? -ne 0 ]; then
echo "Failed ISP1!"
NEWIF1=0
else
NEWIF1=1
fi

```

```

ping -c 5 -s 100 $GW2 -I $IF2 > /dev/null
if [ $? -ne 0 ]; then
echo "Failed ISP2!"
NEWIF2=0
else
NEWIF2=1
fi

if (( ($NEWIF1!=$OLDIF1) || ($NEWIF2!=$OLDIF2) )); then
echo "Changing default routes"

if (( ($NEWIF1==1) && ($NEWIF2==1) )); then
echo "Both ISP"
ip route delete default
ip route add default scope global nexthop via $GW1 dev $IF1 weight $W1 \
nexthop via $GW2 dev $IF2 weight $W2
elif (( ($NEWIF1==1) && ($NEWIF2==0) )); then
echo "ISP1"
ip route delete default
ip route add default via $GW1 dev $IF1
elif (( ($NEWIF1==0) && ($NEWIF2==1) )); then
echo "ISP2"
ip route delete default
ip route add default via $GW2 dev $IF2
fi

else
echo "OK"
fi

OLDIF1=$NEWIF1
OLDIF2=$NEWIF2
sleep 5
done

```

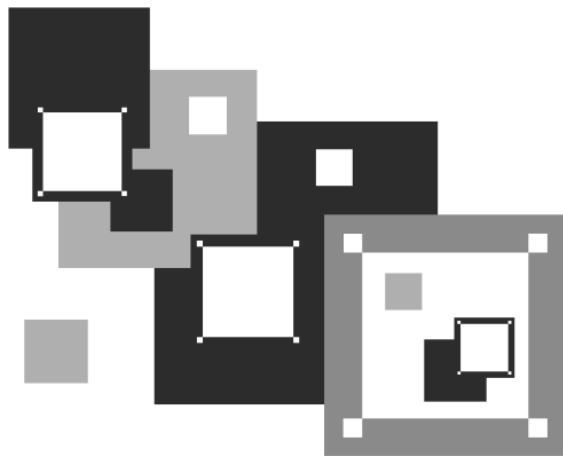
Теперь рассмотрим, как всем этим пользоваться. Запустите сценарий `test_connect` — он сам подключит сценарий `routing`, настраивающий маршрутизацию. Один раз в 5 секунд сценарий `test_connect` станет опрашивать каждый из шлюзов. Если шлюз не ответил ни на один из пяти «пингов», он исключается из таблицы маршрутизации, а шлюзом по умолчанию назначается работоспособный канал:

```

ip route delete default
ip route add default via $GW1 dev $IF1

```

Когда же оба канала работают, то через второй шлюз пойдет в два раза больше трафика, чем через второй. Если нужно, чтобы больше трафика пошло через первый шлюз, просто измените содержимое переменных `W1` и `W2`, — чем выше «вес», тем главнее считается канал.

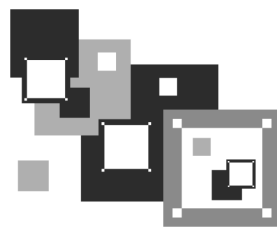


ЧАСТЬ IV

Linux дома и в офисе

Эта часть книги посвящена домашнему и офисному применению Linux. Первым делом мы рассмотрим, как добавить поддержку популярных форматов мультимедиа MP3 и DivX в ваш дистрибутив, поскольку разработчики современных дистрибутивов по лицензионным соображениям исключили поддержку форматов мультимедиа из своих продуктов. Затем построим собственный медиацентр. А уже затем поговорим о настройке графической подсистемы, трехмерном рабочем столе, пакете LibreOffice, программе GIMP и других полезных приложениях.

ГЛАВА 12



Поддержка форматов мультимедиа

12.1. Что такое кодеки и почему их нет в Linux?

Существует очень много мультимедиаформатов для хранения звука и видео: MP3, OGG, WMA, WMV, MP4 и пр. Чтобы ваша система могла воспроизводить каждый конкретный формат, для этого формата нужен *кодек* (codec, от COder/DECoder) — специальная программа, «знающая» как работать с тем или иным форматом. Кодеком можно сравнить с драйвером устройства, только драйвер «обучает» систему, как работать с определенным устройством, а кодек — как воспроизводить тот или иной формат мультимедиа.

Практически из всех дистрибутивов Linux исключена поддержка MP3, DivX, WMV, DVD и других запатентованных форматов. Впрочем, это не означает, что вы не можете смотреть в Linux фильмы или слушать музыку. Поддержка форматов «из коробки» (т. е. сразу после установки дистрибутива) исключена лишь для того, чтобы не нарушать действующие патенты. Конечно, можно включить поддержку этих форматов в состав дистрибутивов, но тогда разработчикам Linux пришлось бы покупать лицензию на распространение каждого кодека. Сами понимаете, лицензия в таких случаях стоит не пару долларов, и чтобы вернуть вложенные средства, Linux пришлось бы сделать платным, что никому не нужно. Поэтому все остается, как было: Linux — бесплатен, но без кодеков.

Вы же, как конечный пользователь, можете совершенно бесплатно загрузить кодеки для воспроизведения всех мультимедиаформатов. При этом не будут нарушены ни действующие патенты, ни чьи-либо авторские права, поскольку вы загружаете кодеки для личного использования, а не для распространения или получения прибыли.

12.2. Настройка дистрибутива Fedora 25-26

В ранних дистрибутивах Fedora по умолчанию устанавливался проигрыватель мультимедиа Totem, сейчас же Fedora комплектуется проигрывателем Videos. Это неплохой проигрыватель, но я бы рекомендовал установить более продвинутый

проигрыватель — VLC. Для этого нужно сначала установить пакет RPMFusion, а затем — собственно VLC:

```
----- Fedora 24 -----
sudo rpm -ivh http://downloadl.rpmsfusion.org/free/fedora/rpmsfusion-free-
release-24.noarch.rpm
----- Fedora 25 -----
sudo rpm -ivh http://downloadl.rpmsfusion.org/free/fedora/rpmsfusion-free-
release-25.noarch.rpm
sudo dnf install vlc
```

Осталось только установить кодеки — просто введите эту длинную команду:

```
sudo dnf install gstreamer-plugins-bad gstreamer-plugins-bad-free-extras
gstreamer-plugins-ugly gstreamer-ffmpeg gstreamer1-libav gstreamer1-
plugins-bad-free-extras gstreamer1-plugins-bad-free-world gstreamer-
plugins-base-tools gstreamer1-plugins-good-extras gstreamer1-plugins-
ugly gstreamer1-plugins-bad-free gstreamer1-plugins-good gstreamer1-
plugins-base gstreamer1
```

На этом все... Если раньше приходилось «изобретать велосипед», то сейчас проблема решается просто путем установки длинного списка пакетов.

12.3. Установка кодеков в openSUSE

Установить кодеки в openSUSE можно двумя способами: автоматически или вручную, и здесь вам будут наглядно продемонстрированы оба способа.

Первый способ заключается в следующем: перейдите по адресу: <http://opensuse-guide.org/codecs.php>, нажмите кнопку **Install Multimedia Codecs** и в открывшемся окне выберите открытие файла в **YaST 1-Click Install** (рис. 12.1).

В окне **YaST2 - Установка в 1 клик** следуйте инструкциям инсталлятора. Кстати, последний покажет, что он собирается сделать: добавить репозитории Packman и libdvdcss, а также установить девять пакетов (рис. 12.2).

После нажатия кнопки **Далее** начнется мучительная установка пакетов. Отходить от компьютера нельзя, поскольку вам придется то пароль root ввести, то отвечать на разные бессмысленные вопросы инсталлятора (рис. 12.3).

В процессе установки нужно будет разрешить один из конфликтов пакетов (рис. 12.4)— выберите опцию **1** и нажмите кнопку **ОК - попробовать снова**. Останется подождать, пока будут установлены все пакеты (рис. 12.5), и, получив сообщение о том, что установка прошла успешно, нажать кнопку **Завершить** (рис. 12.6).

Лично меня этот процесс настолько утомил, что напрочь убил желание полностью его иллюстрировать. Да и вы бы спасибо мне не сказали — ведь я сделал целых 19 скриншотов установки в openSUSE одних только кодеков! Так что, внимательно читайте все, что предлагает вам инсталлятор, благо, это выводится на русском языке.

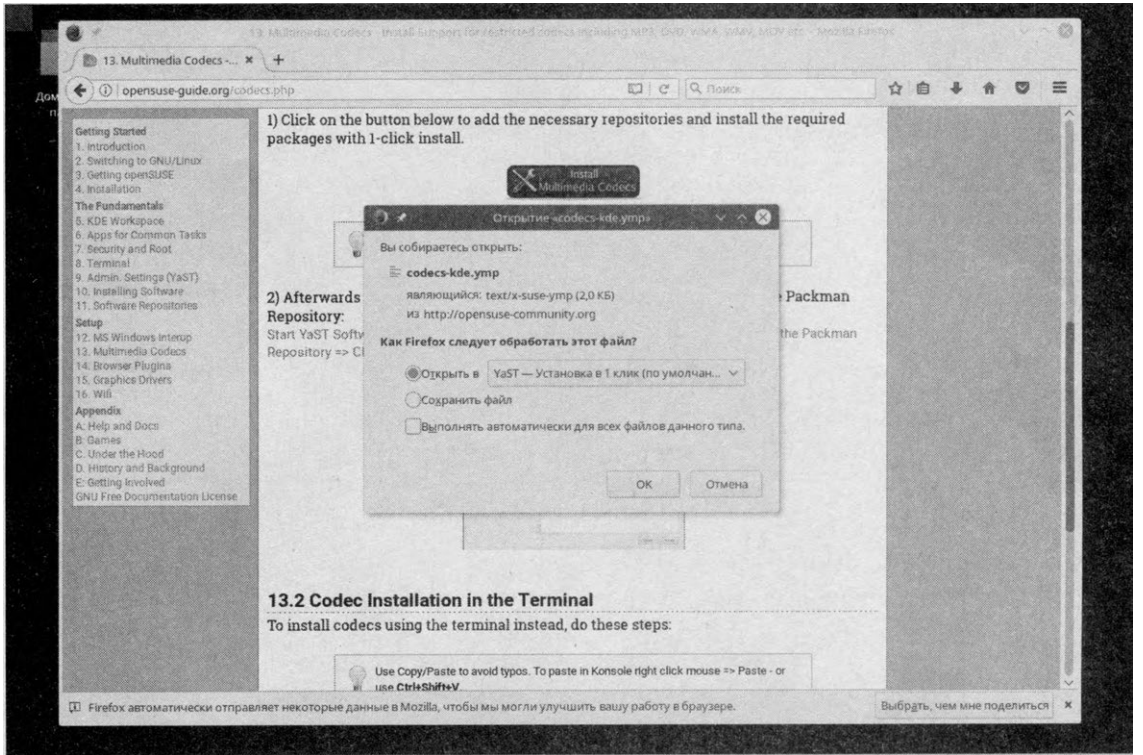


Рис. 12.1. openSUSE: открываем загружаемый файл в YaST-Установка в 1 клик

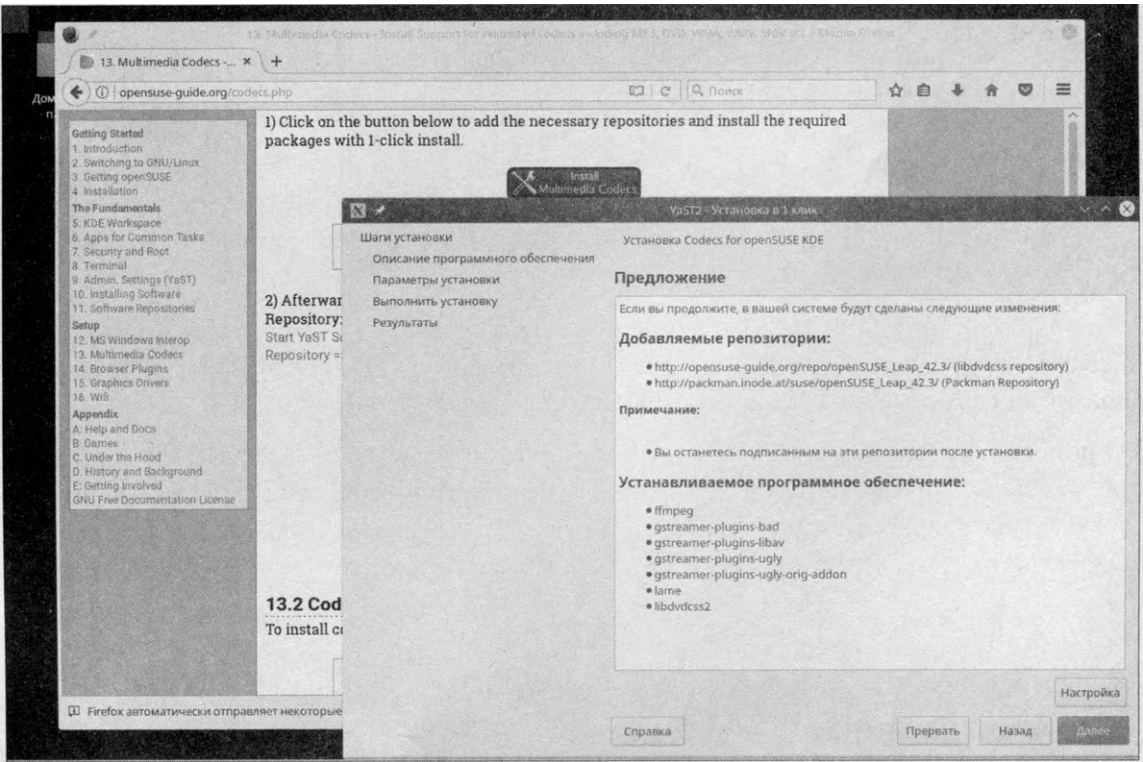


Рис. 12.2. openSUSE: добавление репозиториев и установка пакетов

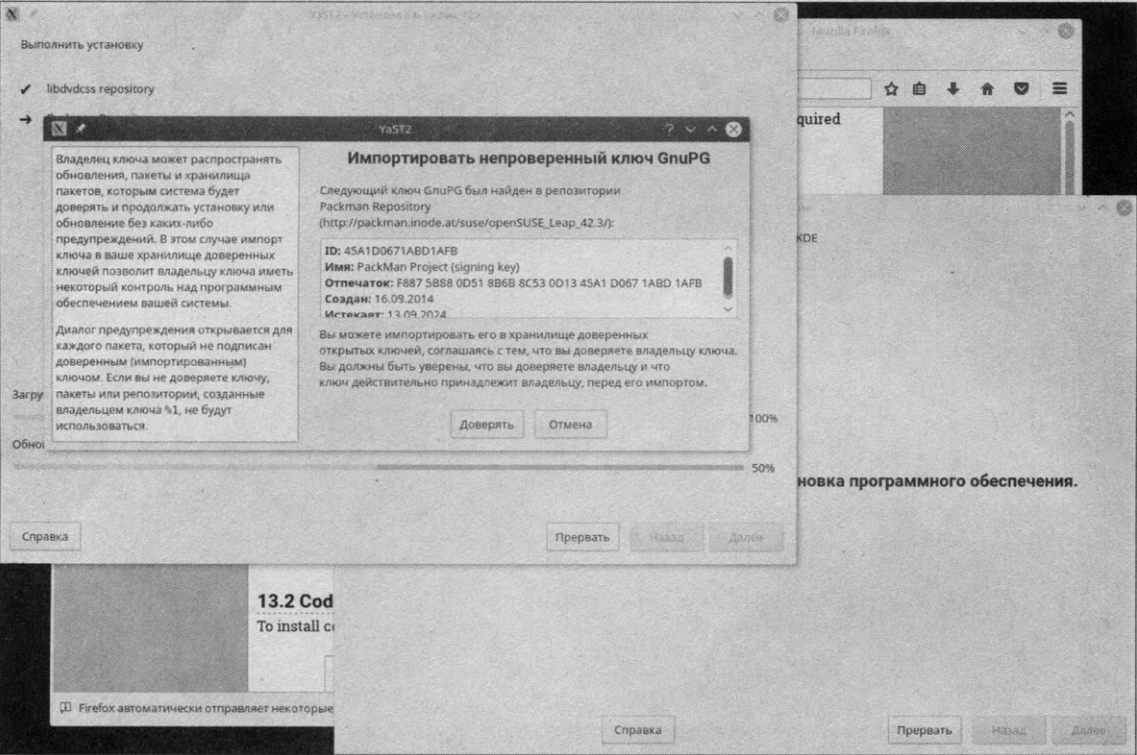


Рис. 12.3. openSUSE: импортировать непроверенный ключ?

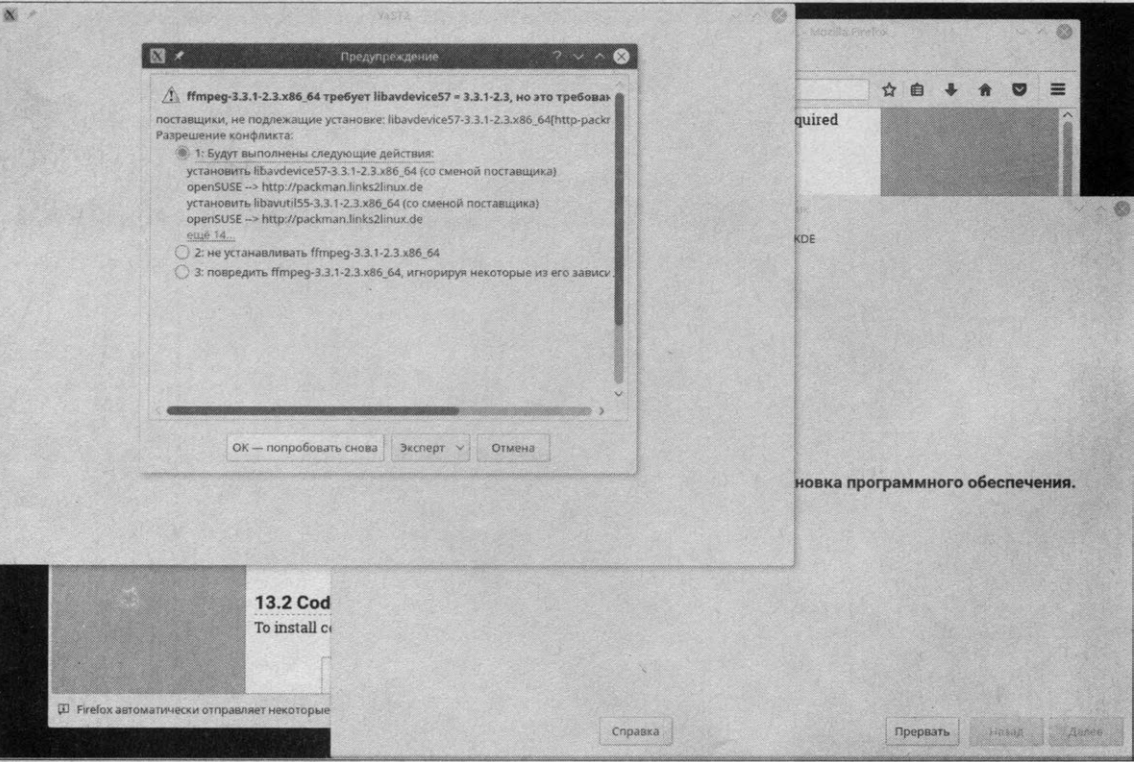


Рис. 12.4. openSUSE: разрешение конфликта пакетов

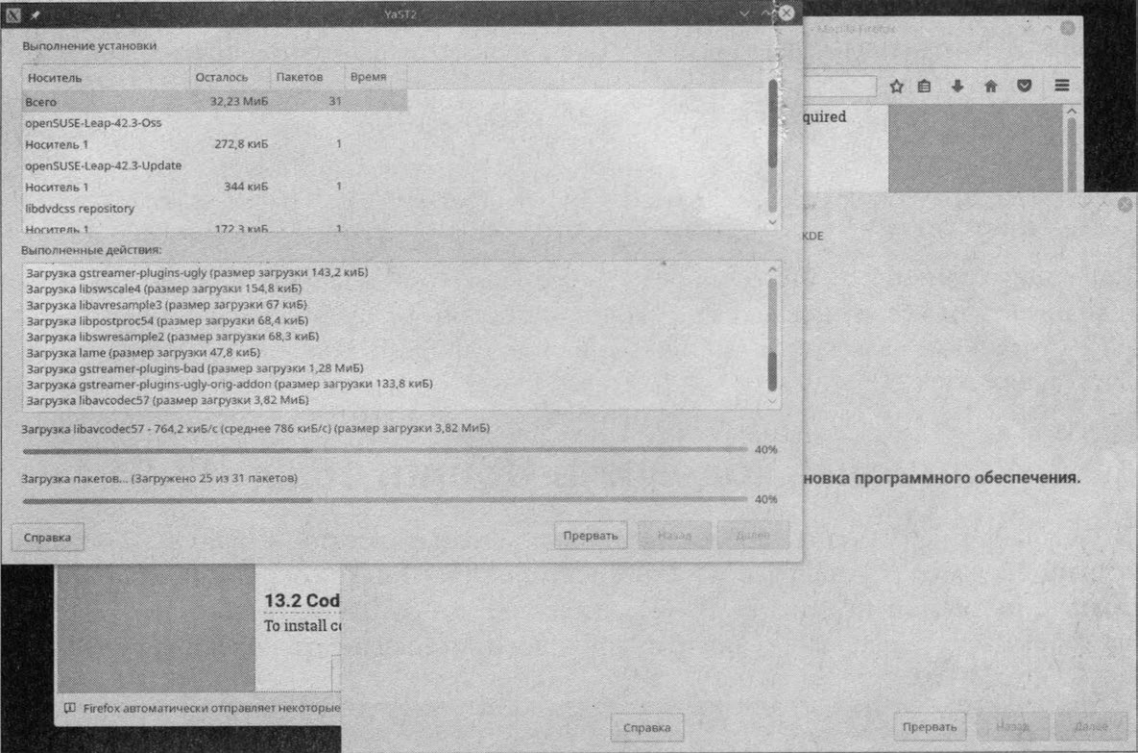


Рис. 12.5. openSUSE: установка пакетов

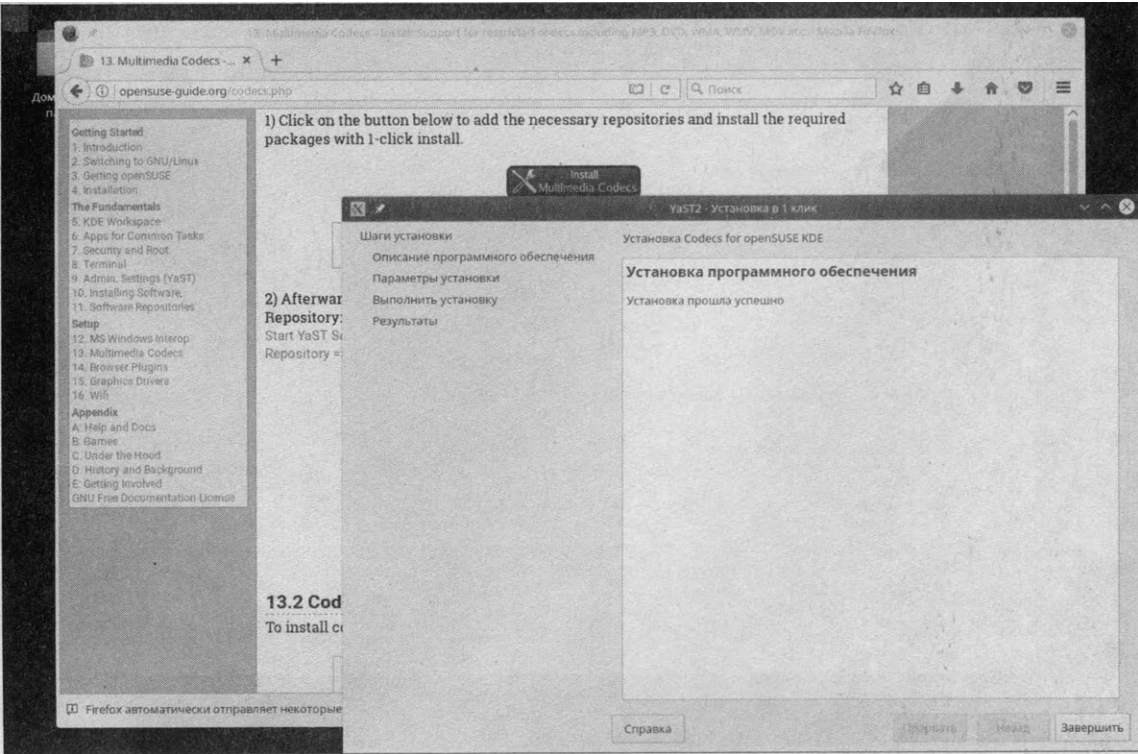


Рис. 12.6. openSUSE: установка завершена

А теперь посмотрим, как установить кодеки вручную, — все здесь сводится всего лишь к трем командам: первые две устанавливают репозитории, третья — необходимые пакеты:

```
zypper addrepo -f http://packman.inode.at/suse/openSUSE_Leap_42.3/
packman
zypper addrepo -f http://opensuse-guide.Org/repo/openSUSE_Leap_42.3/dvd
zypper install ffmpeg lame gstreamer-plugins-bad gstreamer-plugins-ugly
gstreamer-plugins-ugly-orig-addon gstreamer-plugins-libav libdvdcss2
```

Как видите, командная строка иногда — более простой и гибкий инструмент. А вот с количеством всевозможных запросов в инсталляторе openSUSE явно перемудрили — установка называется «за один клик», а этих самых «кликов» пришлось сделать множество.

12.4. Установка кодеков в Ubuntu 16.04-17.04

В Ubuntu также есть два способа установки: автоматический и ручной. Автоматический сводится к установке пакета **Расширения Ubuntu, ограниченные патентами или законами** (рис. 12.7). К сожалению, этого пакета в версии 17.04 я не нашел (может, плохо искал), поэтому пришлось устанавливать кодеки вручную.

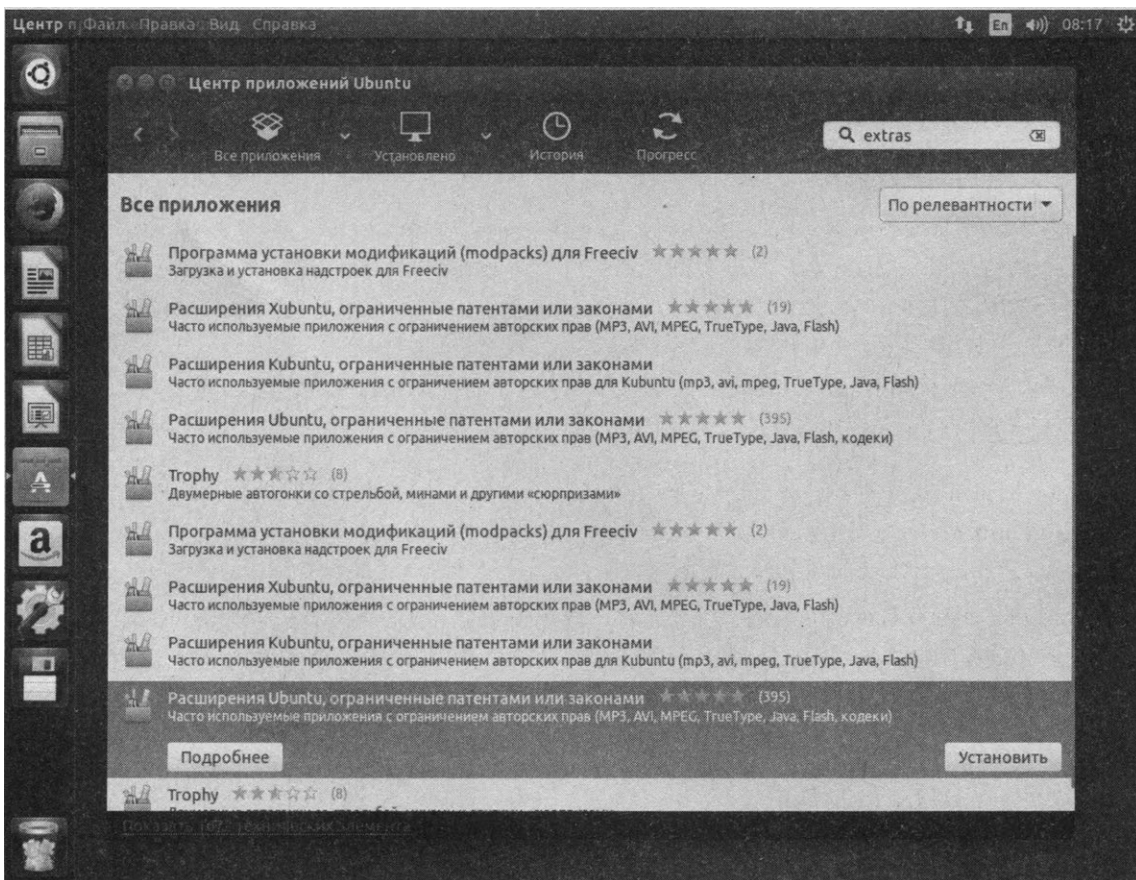


Рис. 12.7. Ubuntu 16.04: автоматическая установка кодеков

А в ручном для установки кодеков нужно выполнить следующие команды:

```
$ sudo apt-get install ubuntu-restricted-extras  
$ sudo apt-get install ffmpeg gxine libdvdread4 icedax tagtool libdvd-pkg  
easytag id3tool lame libxine2-ffmpeg nautilus-script-audio-convert libmad0  
mpg321 libavcodec-extra gstreamer1.0-libav
```

12.5. Домашний медиацентр на основе openELEC

12.5.1. Выбор дистрибутива

На самом деле, мои мучения с мультимедиа не ограничились установкой кодеков в различных дистрибутивах. Захотелось создать медиацентр, который заменил бы обычный DVD-проигрыватель. Ведь, если разобраться, в DVD-проигрывателе нет ничего интересного — примитивное устройство с точки зрения программной части. А если подключить к телевизору компьютер, то открываются огромные возможности: можно и видео онлайн посмотреть (тот же YouTube), и фильмы из Интернета (чтобы не бегать с болванкой или флешкой от компьютера к DVD-проигрывателю).

Но какой дистрибутив выбрать для медиацентра? С технической точки зрения можно выбрать любой, который умеет воспроизводить аудио и видео, но, согласитесь, это не столь интересно — интерфейс будет обычный, компьютерный. А хочется чего-то в стиле интерфейса того же DVD-проигрывателя, но, в то же время, с возможностями обычного компьютера.

Я нашел такой дистрибутив — openELEC. И вся оставшаяся часть главы посвящена этому дистрибутиву — вы узнаете, как его установить, как настроить, как установить в нем программы и как их использовать. Благо, все это настолько просто, что даже не заслуживает отдельной главы.

Так что же представляет собой openELEC? Это легкий дистрибутив Linux, инсталляционные файлы которого «вешат» чуть больше 120 Мбайт. Для сравнения: та же Ubuntu после установки всего необходимого программного обеспечения заняла 4,81 Гбайт, а openSUSE (куда из дополнительного программного обеспечения были добавлены лишь файловый менеджер mc и кодеки) — 5,6 Ебайт. Создавать же какой-то собственный дистрибутив только для видеопросмотров было мне не с руки.

Лично для меня большой интерес заключался в предоставляемой openELEC возможности просмотра фильмов онлайн (благо, скорость доступа к Интернету позволяет), для чего большой жесткий диск не нужен, поэтому и появилась идея сэкономить на нем и поставить медиацентр на флешку. Так вот, на openELEC можно с легкостью реализовать медиацентр и установить его на 8-гигабайтную флешку, обойдясь вовсе без жесткого диска, или, по крайней мере, сэкономить за счет такого медиацентра 4-6 Гбайт на жестком диске для пары-тройки фильмов.

В итоге мой медиацентр состоит из компьютера без жесткого диска с приводом DVD (планируется установка Blu-ray) и подключением к Интернету. Для более

требовательного пользователя никто не мешает установить жесткий диск (и инсталлировать на него дистрибутив), а также добавить еще и ТВ-тюнер. Процесс установки openELEC от этого не изменится.

Чем еще хорош openELEC? — его не нужно (ну, практически не нужно) настраивать: вы не заботитесь ни о видеокарте, ни о звуковой плате, ни о кодеках — все это работает «из коробки». А вам надо только выбрать язык и, возможно, изменить параметры сети. К тому же, все это и управляется по сети — вы можете удаленно управлять своим медиацентром, загружать удаленно на него фильмы и т. д.

12.5.2. Установка дистрибутива

Итак, приступим. Если вы решили пойти моим путем и установить дистрибутив на флешку, вам понадобятся две флешки: на первую вы запишете инсталлятор, а на вторую — установите дистрибутив. Обе флешки должны быть отформатированы в FAT.

Первым делом нужно загрузить инсталлятор дистрибутива с официального сайта <http://openelec.tv/>. На этом сайте вы найдете несколько сборок openELEC, в том числе и для процессоров Intel и Apple TV. Если у вас самый обычный компьютер, можете загрузить сборку **Generic Build**.

Загруженный архив OpenELEC-Generic.i386-1.0.2.tar.bz2 распакуйте в любой каталог и перейдите в полученный каталог OpenELEC-Generic.i386-1.0.2. Если вы работаете в Linux, введите команду:

```
./create_installstick
```

В Windows следует запустить на выполнение файл `createinstallstick.bat` с правами администратора (рис. 12.8).

По запросу (рис. 12.9) введите букву накопителя флешки (а для Linux-версии — имя устройства флешки), на которую нужно установить инсталлятор дистрибутива.

Запись инсталлятора занимает около 20 секунд. Если процесс затянется, можно завершить его, переформатировать флешку и запустить файл `createinstallstick.bat` заново.

После завершения записи инсталлятора на флешку, о чем вы увидите соответствующее сообщение (рис. 12.10), перезагрузите компьютер, выбрав в его BIOS Setup загрузку с флешки. Не забудьте также предварительно вставить флешку, на которую собираетесь установить openELEC!

После перезагрузки компьютера в режиме загрузки с флешки загрузится инсталлятор, и вы увидите его меню (рис. 12.11). Честно говоря, не знаю, зачем оно нужно, если в нем работает только первый пункт — быстрая установка. Поэтому просто нажмите клавишу <Enter> для продолжения.

На следующем шаге вам будет предложено выбрать носитель, на который должна быть установлена openELEC, — заранее подготовленную флешку. Будьте осторожны и не установите ненароком дистрибутив на жесткий диск! Это не openSUSE или Ubuntu, которые используют для создания Linux-раздела свободное пространство

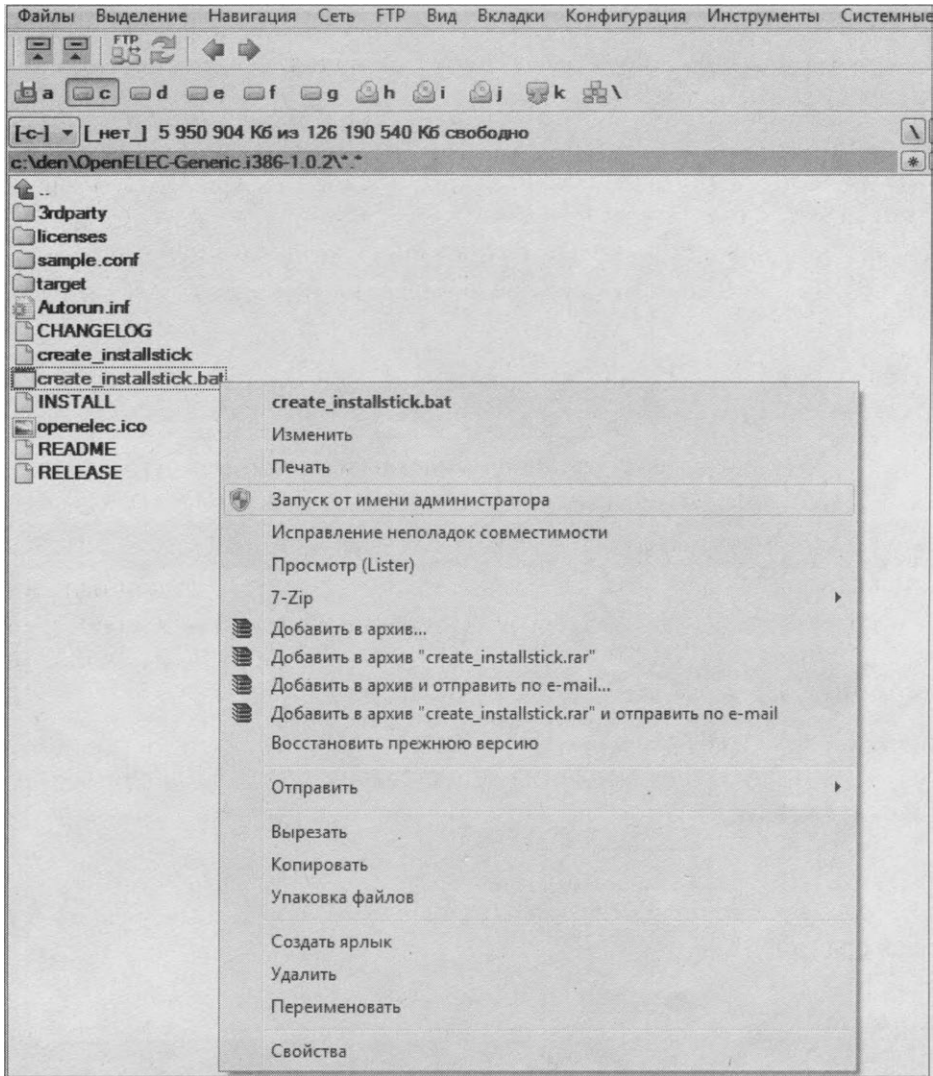


Рис. 12.8. Windows: запуск файла create_installstick.bat

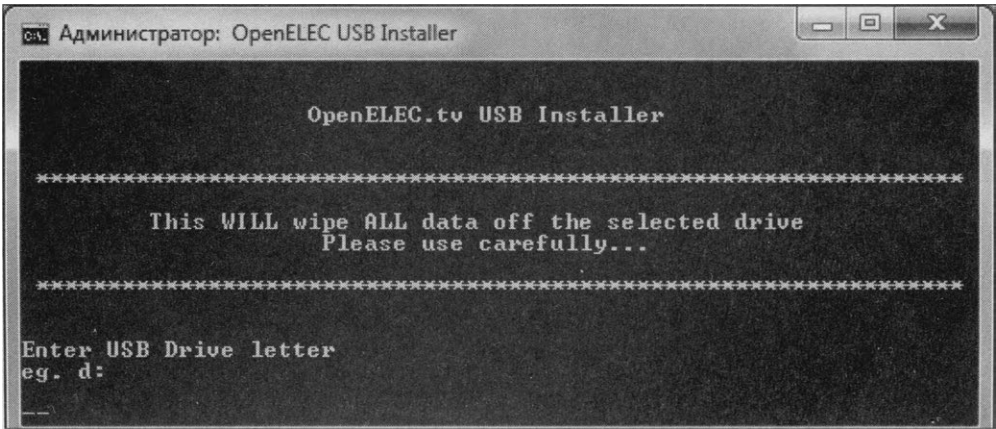


Рис. 12.9. Windows: введите букву накопителя

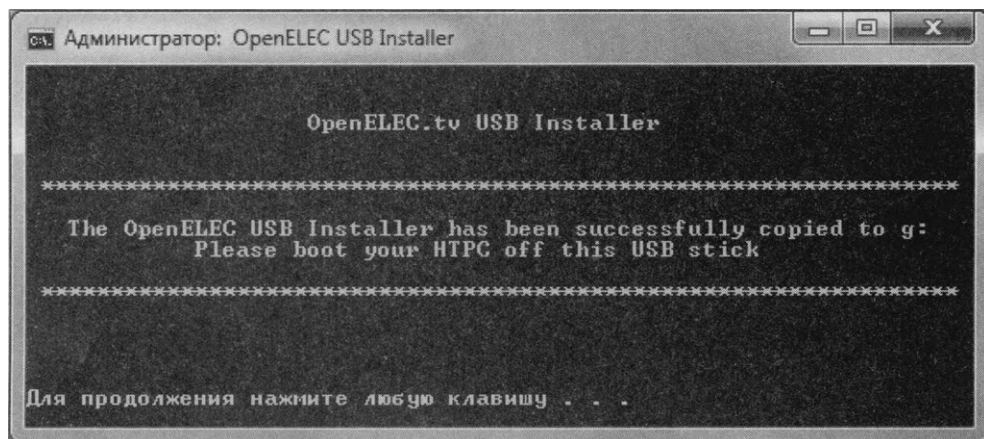


Рис. 12.10. Windows: запись инсталлятора завершена

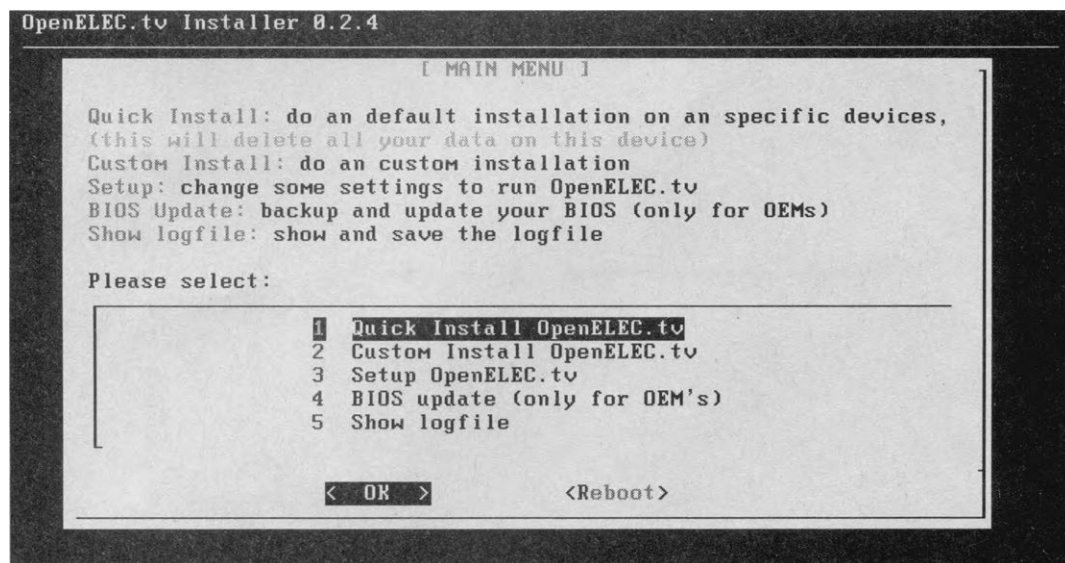


Рис. 12.11. openELEC: меню инсталлятора

диска. Инсталлятор openELEC примитивен до ужаса — он удаляет с выбранного носителя все, что там имеется, и создает структуру разделов, необходимую для openELEC.

После выбора носителя вам продемонстрируют процесс установки (рис. 12.12), правда, недолгий, — установка дистрибутива занимает около минуты, может, даже меньше.

Теперь загружаемся со второй флешки. Поначалу загрузка меня не порадовала: сначала я увидел приглашение загрузчика, потом часть сообщений ядра, после чего пришлось любоваться классикой — черным квадратом Малевича (ну, почти квадратом, — монитор-то у меня 4:3). Я уже собирался было нажать кнопку Reset, как открылся интерфейс XBMC, ради которого я и устанавливал этот дистрибутив (рис. 12.13) — правда, впечатляет? Во всяком случае, для домашнего кинотеатра он гораздо лучше подходит, чем уже приевшиеся GNOME и KDE.

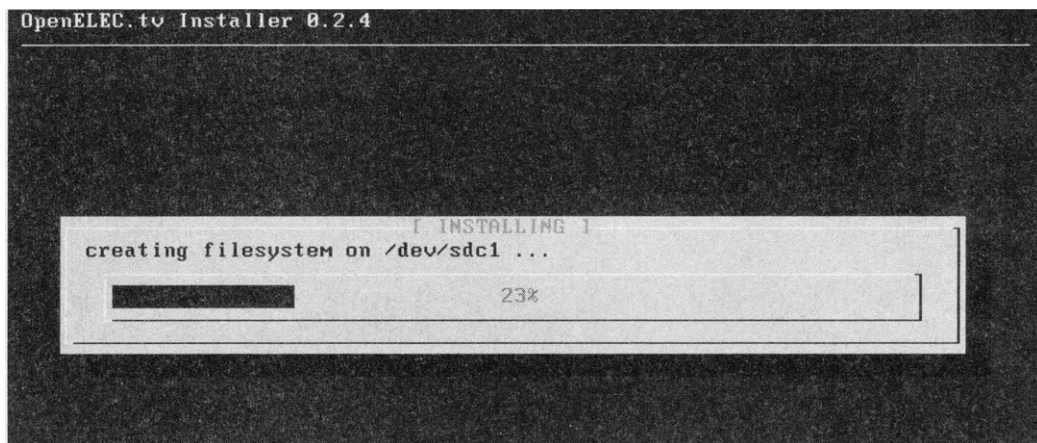


Рис. 12.12. openELEC: установка дистрибутива



Рис. 12.13. openELEC: интерфейс XBMC

12.5.3. Настройка и использование

Поскольку домашним кинотеатром должны пользоваться близкие, то первым делом следует русифицировать интерфейс, — идем в меню **SYSTEM | Settings | Appearance | International** и изменяем параметр **Language**. Рекомендую выбрать **Russian** — думаю, вы уже догадались. Впрочем, никто не запрещает выбрать и японский (рис. 12.14)— тогда пользоваться кинотеатром станет совсем просто. Шутка. После выбора языка интерфейс станет еще приятнее — своя рубашка ближе к телу (рис. 12.15).

Теперь о самом главном— о доступе к Интернету. Есть две новости: хорошая и плохая. Начну с хорошей — поддержка сети есть. А теперь плохая — поддержка

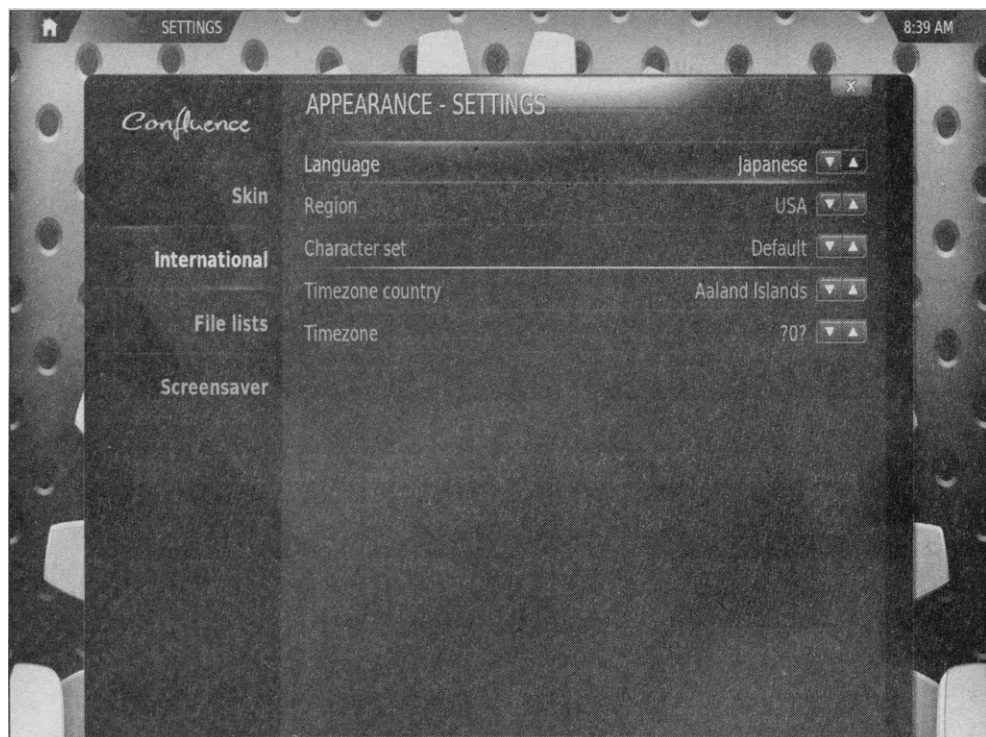


Рис. 12.14. openELEC: изменение языка

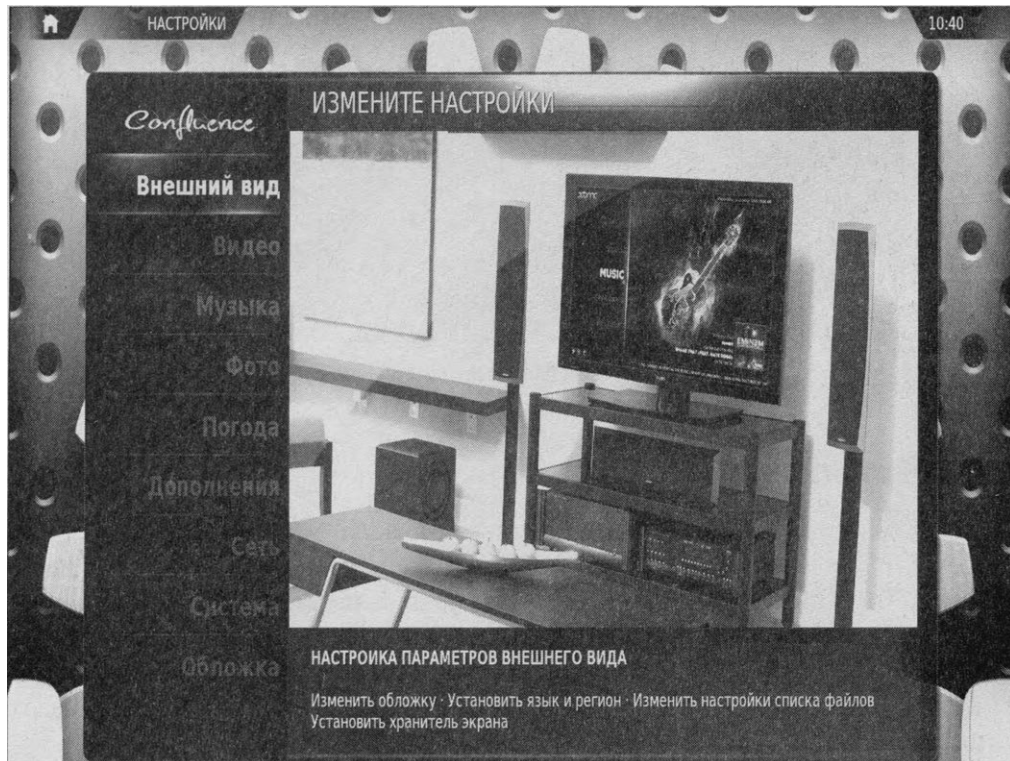


Рис. 12.15. openELEC: выбран русский язык

сети, насколько я понял, только Ethernet. Никакой поддержки ни Wi-Fi, ни PPPoE. Пришлось подключать медицентр к маршрутизатору Wi-Fi с помощью Ethernet-кабеля. Честно говоря, сейчас, когда даже в мобильном телефоне есть поддержка Wi-Fi, длинный Ethernet-кабель через всю квартиру смотрится немного дико. А тянуть его пришлось именно так, поскольку маршрутизатор у меня установлен в одной комнате, а телевизор — в другой.

Настройки сети (**Система | Сеть | Доступ в интернет**) весьма скудны — вы можете установить только параметры HTTP-прокси (рис. 12.16). Ну, с одной стороны, чего же ожидать от дистрибутива для DVD-проигрывателя? Если хочется универсальности, следует устанавливать универсальный дистрибутив и мириться со скучным интерфейсом.

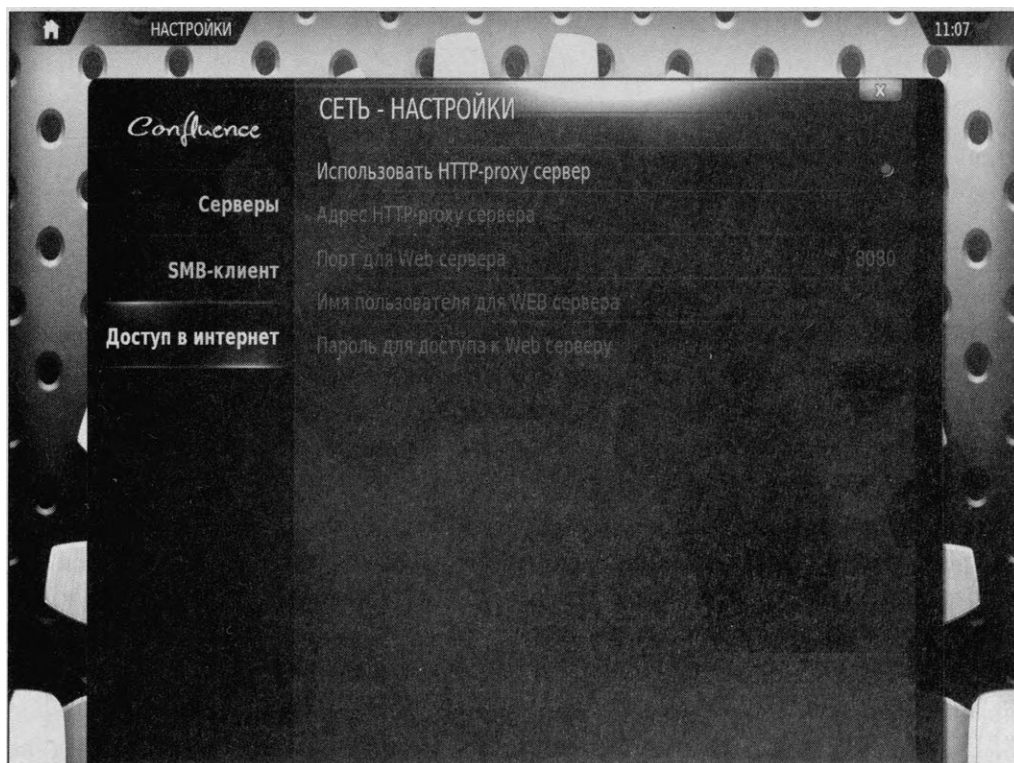


Рис. 12.16. openELEC: параметры сети

С использованием openELEC разберется даже совсем неопытный пользователь. Думаю, вам хватит 10 минут, чтобы освоиться. Впрочем, сделаю небольшой экскурс. Начнем с просмотра видео — перейдите только в соответствующий раздел (рис. 12.17).

Первый источник (**Storage**) — это собственно флешка. Понятно, что пока вы на нее фильмы не записывали, они там сами не появятся. Кстати, поскольку флешка форматируется в файловой системе Linux, то прочитать и записать ее можно теперь только в Linux. А вот команда **Добавить источник** очень полезна — она не только позволяет добавить источник видео (скажем, отдельный диск), но и произвести



Рис. 12.17. openELEC: выбор видеофайлов

поиск видео на YouTube. На рис. 12.18 как раз и отображены результаты поиска на YouTube по ключевому слову *Mountains*.

Что делать дальше, надеюсь, вы догадались — выбираем фильм и наслаждаемся просмотром (рис. 12.19). Особенностью проигрывателя openELEC является также и то, что пока вы явно не остановите просмотр, воспроизведение будет продолжаться в фоновом режиме, — даже если вы начнете бродить по дебрям меню (рис. 12.20), все равно ничего не пропустите!

Теперь о плагинах — программах, расширяющих функционал дистрибутива. Бич openELEC — практическое отсутствие таких программ. Они есть, но их весьма мало. Стандартных программ вроде офисных приложений вы можете здесь и не искать. Зато есть Torrent-клиенты, почтовые клиенты, программы для просмотра ТВ (при наличии ТВ-тюнера) и т. д. Зайдите в раздел **Программы** (рис. 12.21), и вы увидите, что из них установлено по умолчанию. Для установки дополнительных программ нажмите ссылку **Еще**, выберите программу (я выбрал программу *naipo*), прочитайте ее описание и, если она вам подходит, нажмите кнопку **Установить** (рис. 12.22) — все необходимые файлы загрузятся из Интернета и установятся на ваш компьютер.

Для завершения работы openELEC нажмите кнопку питания — она находится в главном меню, в нижнем левом углу рядом с кнопкой плейлиста (см. рис. 12.20), — вы увидите окошко, позволяющее выключить, перезагрузить или отправить в сон ваш компьютер (рис. 12.23).



Рис. 12.18. openELEC: результаты поиска на YouTube



Рис. 12.19. openEL.EC: просмотр видео

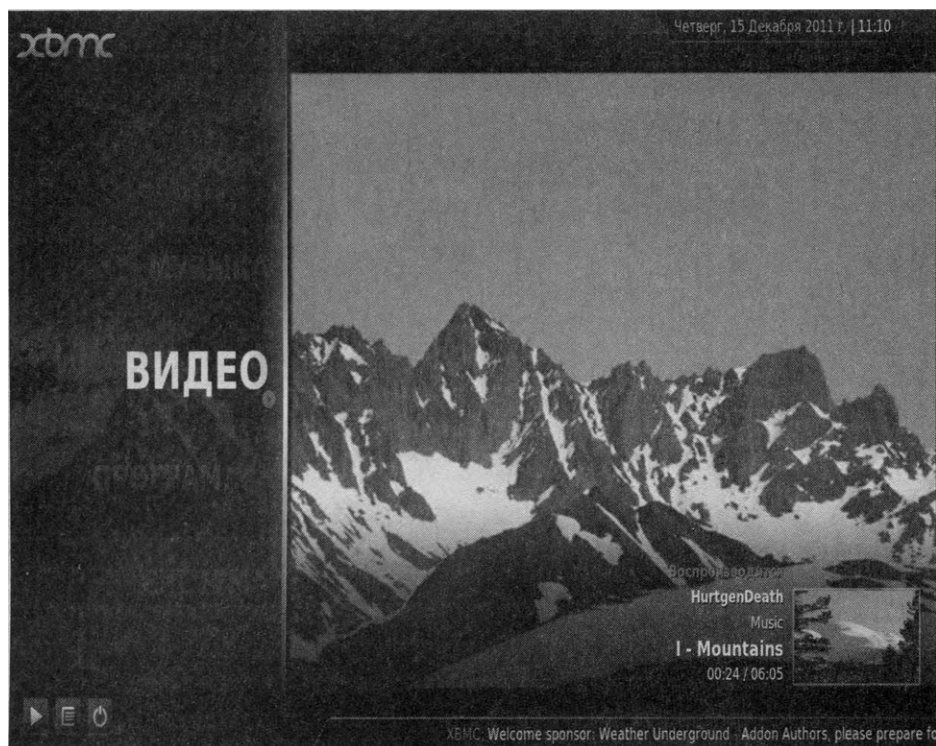


Рис. 12.20. openELEC: воспроизведение в фоновом режиме



Рис. 12.21. openELEC: раздел Программы

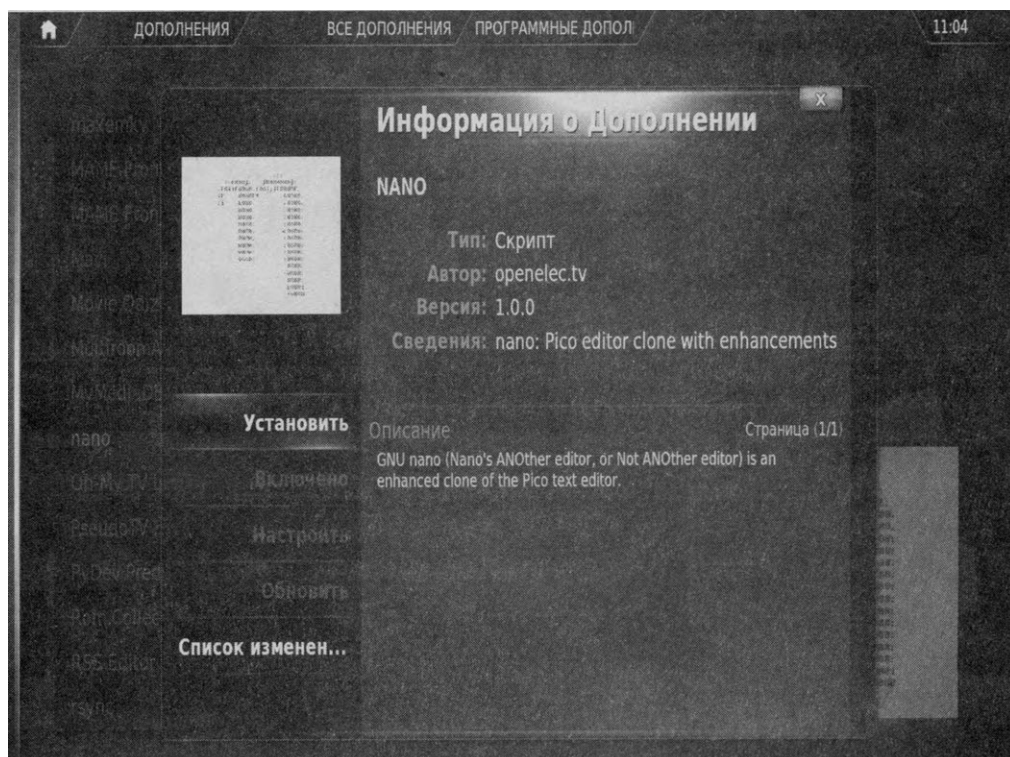


Рис. 12.22. openELEC: установка программы



Рис. 12.23. openELEC: завершение работы

12.5.4. Удаленный доступ

Как уже отмечалось, к нашему домашнему кинотеатру можно получить удаленный доступ. Для этого откройте на другом компьютере браузер и введите адрес: **http://<ip-адрес>: 9981**, где *ip-адрес* — это IP-адрес домашнего кинотеатра. Откроется Web-интерфейс, позволяющий управлять медиacentром.

12.5.5. А где же консоль?

Немного поэкспериментировав с графическим интерфейсом, мне захотелось взглянуть на дистрибутив, так сказать, изнутри, и я попытался найти консоль, но так и не понял, как на нее переключиться. Поиск в Google прояснил, что консоль есть, но удаленная — по ssh.

Если вы работаете в Windows, скачайте любой ssh-клиент (могу порекомендовать программу PuTTY, как одну из наиболее удобных). Для доступа к кинотеатру используются такие параметры:

- IP : IP-адрес вашего медиacentра;
- имя пользователя: root;
- пароль: openelec.

12.5.6. Ложки дегтя...

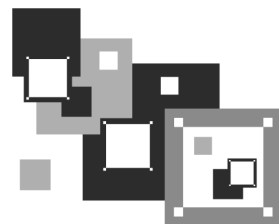
Конечно, мои *планы* предусматривают дооснастить свой медиacentр ТВ-тюнером (читайте в следующем издании продолжение истории) и приводом Blu-ray, но я опасаюсь, что дистрибутив к тому времени подвергнется тем или иным, не всегда полезным, изменениям.

Кроме того, бочку меда может испортить всего одна ложка дегтя, а в случае с openELEC их оказалось подмешано несколько:

- как уже отмечалось, первая загрузка заняла довольно-таки много времени. Последующие загрузки происходили быстрее, но, все равно, не столь быстро, как хотелось, — сам по себе небольшой дистрибутив, а загружается примерно как Fedora 16, может быть, даже медленнее. Я ожидал от него более шустрой работы;
- замечены небольшие подвисания в процессе работы, особенно при открытии каталога с медиафайлами. Даже пара секунд подвисания кинотеатра оставляет весьма неприятный осадок;
- огорчает отсутствие поддержки Wi-Fi. Не знаю, как для кого, а для меня это очень актуально, — уж очень хочется избавиться от лишнего Ethernet-кабеля;
- хоть интерфейс медиacentра и русифицирован, далеко не все программы (плагины) понимают русский. Тот же браузер вообще не знает, что такое русский язык, и не позволяет выбрать соответствующую кодировку.

Тем не менее, первое впечатление от openELEC — весьма хорошее, хоть и несколько подпорчено этим дегтем...

ГЛАВА 13



Графическая подсистема

Когда-то основным камнем преткновения на пути развития Linux было отсутствие у нее удобного графического интерфейса. Базовый графический интерфейс, называемый тогда X Window, существовал уже в 1992 году, но по удобству пользования его нельзя было сравнить с интерфейсом той же Windows 3.11. Помню, даже в 1997 году, когда всю процветала Windows 95, а на подходе была Windows 98, графический интерфейс Linux все еще оставлял желать лучшего. Однако X.Org — современная графическая подсистема Linux — может дать фору интерфейсу любой другой коммерческой операционной системы.

Именно X.Org, через драйверы работающая на низком уровне с видеокартой и монитором, вкупе с графическими оболочками KDE и GNOME создает то многообразие графических возможностей, которыми обладает современный дистрибутив Linux.

13.1. Настройка X.Org в современных дистрибутивах

Могу вас заверить, что в большинстве случаев в современном дистрибутиве вам не придется настраивать X.Org. Вообще, и ни при каких обстоятельствах. Современные дистрибутивы правильно определяют всю конфигурацию графической подсистемы: видеокарту, монитор, несколько мониторов. Максимум, что вам понадобится сделать, — это установить разрешение монитора. Для этого в окне Параметры зайдите в раздел Мониторы, выберите монитор (рис. 13.1) и установите его разрешение (рис. 13.2).

Обратите внимание — несмотря на то, что система определила мой монитор как неизвестный, никаких проблем с ним в дальнейшем не возникло: ни с разрешением (как можно видеть, оптимальное разрешение было выбрано уже по умолчанию — в раздел Мониторы я зашел только для того, чтобы сделать иллюстрацию к книге), ни с отображением информации.

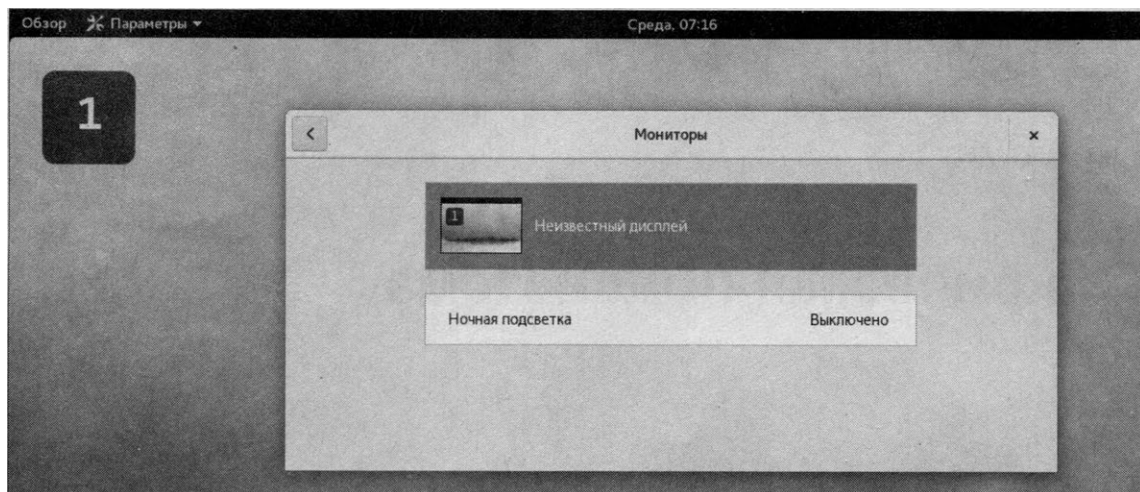


Рис. 13.1. Fedora 26: список мониторов

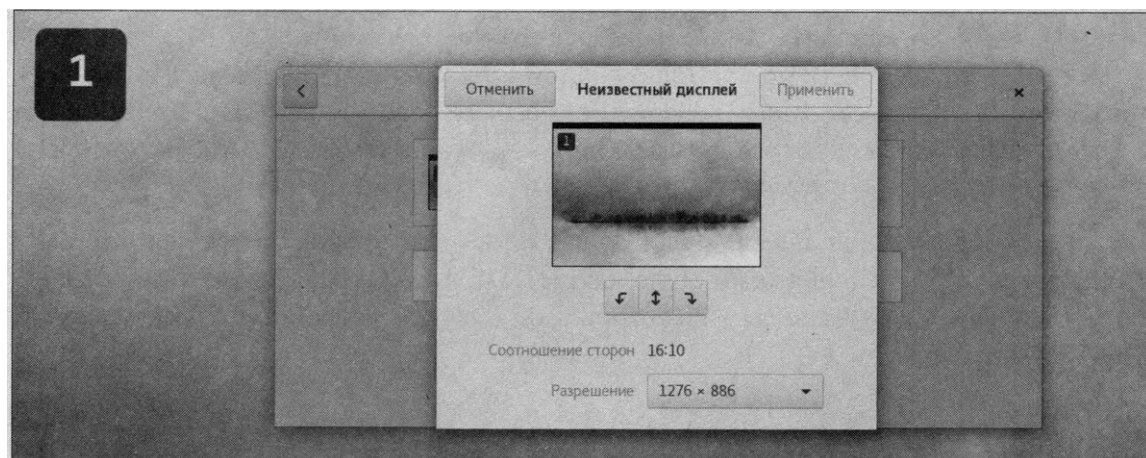


Рис. 13.2. Fedora 26: изменение разрешения монитора

13.2. Конфигурационный файл X.Org

Основным конфигурационным файлом X.Org до версии 7.3 являлся `/etc/X11/xorg.conf` — в нем хранились все настройки графической подсистемы: параметры видеокарты, монитора, клавиатуры, мыши, а также параметры самого X.Org. Однако, начиная с версии 7.3, X.Org может запускаться вообще без файла конфигурации: вы просто вводите команду `startx` или настраиваете систему на пятый уровень запуска, и она запускается безо всяких конфигурационных файлов. Как же настраивать X.Org в такой ситуации? А никак — в большинстве случаев ее, как и отмечалось ранее, настраивать не придется, она и так превосходно работает.

Однако иногда может потребоваться указать для X.Org особые параметры. Что ж, тогда придется создать файл `xorg.conf` вручную. И чтобы не писать его с нуля, можно воспользоваться следующей командой:

```
sudo Xorg -configure
```

Учтите при этом, что команду `xorg -configure` нельзя вводить при запущенной X.Org— надо сначала завершить ее работу, закрыв менеджер дисплеев `gdm`, например, с помощью такой команды:

```
sudo service gdm stop
```

В современных версиях Ubuntu с оболочкой Unity используется менеджер дисплеев `lightdm`, поэтому там нужно завершать работу не `gdm`, а `lightdm`:

```
sudo service lightdm stop
```

Вот теперь можно запускать конфигуратор X.Org. При этом в домашнем каталоге пользователя, запустившего конфигуратор, будет создан файл `xorg.conf.new`, содержащий «скелет» конфигурационного файла X.Org. Отредактируйте его под себя (чуть позже будет показано, что можно в него добавить), а затем переместите или скопируйте этот файл в `/etc/X11/xorg.conf`:

```
sudo mv xorg.conf.new /etc/X11/xorg.conf
```

После этого можно запустить сервис `gdm/lightdm`:

```
sudo service gdm start
```

или

```
sudo service lightdm start
```

В современных дистрибутивах принято хранить конфигурацию не в одном большом файле, а в нескольких маленьких, и для X.Org они, как правило, хранятся в каталоге `/etc/X11/xorg.conf.d`. Названия этих файлов начинаются с двузначной цифры, определяющей порядок чтения конфигурации. Например, названия `00-keyboard`, `10-videocard`, `20-monitor` означают, что сначала будет загружена конфигурация клавиатуры, потом видеокарты, а затем — монитора (конечно, это только в том случае, когда в соответствующих файлах хранится соответствующая конфигурация).

Далее мы рассмотрим синтаксис файла `xorg.conf`, но — еще раз повторяюсь — вряд ли вам пригодится эта информация. Она была актуальной, скажем, лет десять назад, когда настраивать X.Org приходилось если не вручную, то хотя бы редактированием некоторых параметров в файле конфигурации.

13.3. Синтаксис файла `xorg.conf`

Итак, мы уже выяснили, что конфигурационные файлы графической подсистемы хранятся в каталоге `/etc/X11`, а основным ее конфигурационным файлом является `xorg.conf`. Откройте его или создайте, если в вашей системе он отсутствует. Одного взгляда на него достаточно, чтобы понять, что этот файл лучше всего редактировать не вручную, а с помощью конфигуратора. Но мы все же попытаемся в нем разобраться.

Файл состоит из нескольких секций:

- **Files** — параметры файлов, которые используются графической системой, обычно здесь задается путь к шрифтам;

- `serverFlags` — различные флаги сервера;
- `Module` — подключение разных модулей, например, `v4l` (Video For Linux);
- `InputDevice` — с помощью этой секции конфигурируются устройства ввода: клавиатура и мышь;
- `Monitor` — здесь задаются параметры монитора;
- `Modes` — описывается разрешение монитора;
- `Device` — а эта секция содержит параметры видеокарты;
- `Screen` — конфигурационный файл может описывать несколько мониторов и несколько видеокарт, а в секции `Screen` задается, какой именно монитор и какая именно видеокарта будут использоваться в данный момент. Здесь же определяется и текущее разрешение монитора;
- `ServerLayout` — задает, какая секция `Screen` должна использоваться, и описывает устройства ввода: клавиатуру и мышь;
- `Extensions` — служит для указания разных расширений X-сервера.

Вот пример файла конфигурации, настроенного на 17-дюймовый монитор РnР и встроенную видеокарту ATI Radeon Xpress 1250. Если у вас такая же конфигурация, а вы нечаянно изменили этот файл, и больше графическая система не работает, можете использовать листинг 13.1 в качестве образца.

Листинг 13.1. Пример конфигурационного файла `/etc/X11/xorg.conf`

```
# /.../
# Automatically generated by [ISaX] (8.1)
# PLEASE DO NOT EDIT THIS FILE!
#
Section "Files"
    FontPath  "/usr/share/fonts/misc:unsealed"
    FontPath  "/usr/share/fonts/local"
    FontPath  "/usr/share/fonts/75dpi:unsealed"
    FontPath  "/usr/share/fonts/100dpi .-unsealed"
    FontPath  "/usr/share/fonts/Type1"
    FontPath  "/usr/share/fonts/URW"
    FontPath  "/usr/share/fonts/Speedo"
    FontPath  "/usr/share/fonts/PEX"
    FontPath  "/usr/share/fonts/cyrillic"
    FontPath  "/usr/share/fonts/latin2/misc:unsealed"
    FontPath  "/usr/share/fonts/latin2/75dpi:unsealed"
    FontPath  "/usr/share/fonts/latin2/100dpi:unsealed"
    FontPath  "/usr/share/fonts/latin2/Type1"
    FontPath  "/usr/share/fonts/latin7/75dpi:unsealed"
    FontPath  "/usr/share/fonts/baekmuk:unsealed"
    FontPath  "/usr/share/fonts/japanese:unsealed"
    FontPath  "/usr/share/fonts/kwintv"
```

```
FontPath "/usr/share/fonts/truetype"
FontPath "/usr/share/fonts/uni:unsealed"
FontPath "/usr/share/fonts/CID"
FontPath "/usr/share/fonts/ucs/misc:unsealed"
FontPath "/usr/share/fonts/ucs/75dpi:unsealed"
FontPath "/usr/share/fonts/ucs/100dpi:unsealed"
FontPath "/usr/share/fonts/hellas/misc:unsealed"
FontPath f"/usr/share/fonts/hellas/75dpi:unsealed"
FontPath "/usr/share/fonts/hellas/100dpi:unsealed"
FontPath "/usr/share/fonts/hellas/Typel"
FontPath "/usr/share/fonts/misc/sgi:unsealed"
FontPath "/usr/share/fonts/xtest"
FontPath "/opt/kde3/share/fonts"
InputDevices "/dev/gpmdata"
InputDevices "/dev/input/mice"
EndSection

Section "ServerFlags"
    Option "AllowMouseOpenFail" "on"
EndSection

Section "Module"
    Load "dbe"
    Load "typel"
    Load "freetype"
    Load "extmod"
    Load "glx"
EndSection

Section "InputDevice"
    Driver "kbd"
    Identifier "Keyboard[0]"
    Option "Protocol" "Standard"
    Option "XkbLayout" "us,ru"
    Option "XkbModel" "microsoftpro"
    Option "XkbOptions" "grp:ctrl_shift_toggle,grp_led:scroll"
    Option "XkbRules" "xfree86"
    Option "XkbVariant" ",winkeys"
EndSection

Section "InputDevice"
    Driver "mouse"
    Identifier "Mouse[1]"
    Option "Buttons" "5"
    Option "Device" "/dev/input/mice"
    Option "Name" "ImPS/2 Generic Wheel Mouse"
    Option "Protocol" "explorerps/2"
    Option "Vendor" "Sysp"
    Option "ZAxisMapping" "4 5"
EndSection
```

```
Section "Monitor"
    Option      "CalcAlgorithm" "XServerPool"
    HorizSync 30-83
    Identifier "Monitor[0]"
    ModelName "AL1916"
    Option      "DPMS"
    VendorName "ACR"
    VertRefresh      43-75
    UseModes "Modes[0]"
EndSection

Section "Modes"
    Identifier "Modes[0]"
    Modeline "1280x1024" 108 1280 1328 1440 1688 1024 1025 1028 1066 +hsync
                                                +vsync
EndSection

Section "Screen"
    Subsection "Display"
        Depth      16
        Modes       "default"
    EndSubSection
    Device "Device[0]"
    Identifier "Screen[0]"
    Monitor "Monitor[0]"
EndSection

Section "Device"
    BoardName "Framebuffer Graphics"
    Driver "fbdev"
    Identifier "Device[0]"
    VendorName "VESA"
EndSection

Section "ServerLayout"
    Identifier "Layout[all]"
    InputDevice "Keyboard[0]" "CoreKeyboard"
    InputDevice "Mouse[1]" "CorePointer"
    Option      "Clone" "off"
    Option      "Xinerama" "off"
    Screen      "Screen[0]"
EndSection

Section "DRI"
    Group      "video"
    Mode      0660
EndSection

Section "Extensions"
EndSection
```

Рассмотрим секции этого файла подробнее.

- Секция `Files`, как уже было отмечено, содержит каталоги, в которых системе нужно искать шрифты и модули графической подсистемы `X.Org`. Путь к шрифтам задается директивой `FontPath`, а путь к модулям — директивой `ModulePath`.

В листинге 13.1 рассматривается пример файла `xorg.conf` дистрибутива `openSUSE`. Поскольку в этом дистрибутиве нет сервера шрифтов, путь к каждому каталогу со шрифтами задается директивой `FontPath`. В дистрибутивах, где имеется сервер шрифтов, система `X.Org` настраивается на использование сервера шрифтов вот такой директивой:

```
FontPath "unix/:-1"
```

- Перейдем к секции `ServerFlags` (ее наличие не является обязательным) — она может содержать некоторые флаги сервера `X`. Флаги сервера задаются так:

```
Option "название флага" "состояние"
```

Состояние может быть либо `on` — если флаг установлен, либо `off` — если флаг сброшен. Самые полезные флаги `X`-сервера приведены в табл. 13.1 (понятно, что это не все возможные флаги, но остальные редко используются на практике).

Таблица 13.1. Флаги X-сервера

Флаг	Описание
<code>AllowMouseOpenFail</code>	Если флаг установлен (<code>on</code>), то <code>X</code> -сервер продолжит работу даже в случае неработоспособности мыши (когда мышь сломана или не подключена)
<code>AllowNonLocalModInDev</code>	Разрешает удаленным пользователям изменять параметры клавиатуры и мыши <code>X</code> -сервера. По умолчанию флаг выключен (<code>off</code>), и из соображений безопасности его рекомендуется не включать
<code>AIGLX</code>	AIGLX (Accelerated Indirect GLX) нужен для работы трехмерного рабочего стола <code>Compiz Fusion</code> . Поэтому, если вы планируете использовать трехмерный рабочий стол, вам нужно включить этот флаг. В <code>openSUSE</code> этого делать не стоит, поскольку вместо AIGLX в <code>SUSE</code> используется собственная разработка — Xgl
<code>DontZap</code>	Запрещает комбинацию клавиш <code><Ctrl>+<Alt>+<Backspace></code> , служащую для аварийной остановки <code>X</code> -сервера
<code>StandbyTime</code>	Время простоя (в минутах), после которого <code>X</code> -сервер выключит монитор. Монитор должен поддерживать DPMS (Display Power Management Signaling) — систему управления энергопотреблением дисплеев
<code>NoPM</code>	Запрещает управление питанием монитора — монитор будет включен всегда

- Следующая секция — `Modules` — используется для загрузки дополнительных модулей. Секция может отсутствовать, если дополнительные модули не загружаются.

- В секции `inputDevice` описываются устройства ввода: клавиатура и мышь:
 - обратите внимание на опцию `xkbvariant` в секции `inputDevice`, описывающей клавиатуру. Эта опция позволяет указать вариант раскладки клавиатуры. В нашем случае задана Windows-раскладка ("winkeys"), к которой привыкло большинство пользователей. Если в секции `inputDevice` вашего файла нет строки `Option "xkbvariant" ", winkeys"`, добавьте ее — вам будет удобнее;
 - опция `xkbLayout` задает раскладки клавиатуры — сейчас используются две раскладки: английская (us) и русская (ru);
 - опция `xkbOptions` определяет комбинацию клавиш для переключения раскладок — в данном случае задана комбинация клавиш `<Ctrl>+<Shift>`. Если вы привыкли к `<Alt>+<Shift>`, то вместо `ctrl_shift_toggle` укажите `alt_shift_toggle`. Если вы не хотите, чтобы при активации русской раскладки на клавиатуре загорался индикатор Scroll Lock, удалите строку `", grp_led: scroll"` **из опции** `XkbOptions`.

Как видите, можно настроить параметры клавиатуры, не прибегая к помощи конфигураторов.

- Секция `Monitor` задает параметры монитора:
 - `Option` — различные опции монитора. Например, часто используется опция `DPMS`, подтверждающая, что монитор поддерживает DPMS;
 - `HorizSync`, `vertRefresh` — допустимая частота горизонтальной и вертикальной разверток соответственно (в кГц). Обычно значение лучше устанавливать конфигуратором, поскольку без подробного руководства по вашему монитору вам не обойтись;
 - `identifier` — уникальное имя монитора. В файле конфигурации вы можете описать несколько мониторов, а потом в секции `screen` указать идентификатор монитора, используемого в данный момент;
 - `useModes` — задает массив режимов монитора, описываемый секцией `Modes`;
 - `ModeiName`, `vendorName` — наименование модели монитора и название его производителя. Сугубо информационные строки — можете вписать сюда все, что угодно.

- Секция `Modes`, задающая массив режимов для конкретного монитора, тесно связана с секцией `Monitor`. Как вы уже догадались, для каждого монитора должен быть свой массив режимов. Каждый режим описывается так:

```
Modeline "название режима" 1 2 3 4 5 6 7 8 9 флаги
```

где:

- название режима — обычная строка, сугубо информационная — чтобы вы знали, какому разрешению соответствует данный режим (вообще-то эта строка может содержать все, что угодно);
- 1 — частота подачи пикселей на монитор, указывается в мегагерцах;

- 2–5 — значения строчной синхронизации (то же, что и горизонтальная развертка);
- 6–9 — значения кадровой синхронизации (вертикальная развертка);
- флаги — флаги развертки, обычно используются флаги `+hsync` и `+vsync`.

Строку режимов лучше всего изменять с помощью конфигулятора, поскольку в его базе данных есть описание режимов практически для всех мониторов.

- Секция `Screen` предназначена для описания экрана, главным образом — для связки видеокарты и монитора. В этой секции представлены используемые в данный момент видеокарта (ее идентификатор задается директивой `Device`) и монитор (директива `Monitor`).

Также в секции `Screen` может быть подсекция `Display`, в которой указывается параметр `Depth`, задающий глубину цвета, но главная задача этой секции — связка монитора и видеокарты воедино.

- Параметры видеокарты описываются в секции `Device`.
- Задача секции `serverLayout` — связать воедино устройства ввода (клавиатура и мышь), а также секцию `Screen`, которая, в свою очередь, связывает секции `Device` и `Monitor`. Как видите, в `X.Org` все взаимосвязано.
- `DRI` (`Direct Rendering Infrastructure`) — это платформа, предоставляющая прямой доступ к графическому оборудованию самым безопасным и эффективным способом. Параметры `DRI` задаются в секции `DRI`. Как правило, параметры этой секции приходится редактировать при проблемах с аппаратным трехмерным ускорением.
- Секция `Extensions`, описывающая расширения `X.Org`, может быть пуста (в большинстве случаев) или вообще отсутствовать.

СДЕЛАЙТЕ РЕЗЕРВНУЮ КОПИЮ РЕДАКТИРУЕМОГО ФАЙЛА

Помните, что файл конфигурации `xorg.conf` можно редактировать, обладая полномочиями `root`. Перед каждым редактированием файла вручную (не с помощью конфигулятора) делайте его резервную копию, чтобы в случае отказа `X.Org` или ее нестабильной работы вы могли бы восстановить предыдущее состояние.

13.4. Установка проприетарных драйверов NVIDIA в Fedora 21-26

Сразу хочу отметить, что если вы не планируете в `Fedora` использовать трехмерные эффекты или запускать игры, то проприетарные драйверы `NVIDIA` вам не понадобятся, — вполне будет достаточно стандартного драйвера.

ВНИМАНИЕ!

Изложенные здесь советы подходят для карт `GeForce 6/7/8/9/200/300/400/500/600/700/800/900`.

Также предупреждаю— процесс установки проприетарных драйверов не прост и требует определенного времени. Поэтому, как в Windows — скачать инсталлятор и установить — здесь не получится. Запасайтесь временем и терпением.

Чтобы не делать лишней работы, проверим сначала, какая у нас видеокарта:

```
lspci |grep -i VGA
```

Вывод может быть примерно таким:

```
01:00.0 VGA compatible controller: NVIDIA Corporation GF119 [GeForce GT 610]
(rev a1)
```

Найденное наименование видеокарты должно присутствовать в списке, который можно закатать по ссылке: ftp://download.nvidia.com/XFree86/Linux-x86_64/352.63/README/supportedchips.html.

ВНИМАНИЕ!

Ваша видеокарта должна найтись в этом списке до секции 173.14.xx— далее перечислены устаревшие версии видеокарт NVIDIA, которые рассматривать смысла не имеет.

Последнюю версию драйверов NVIDIA для Linux можно получить по адресу: <http://www.nvidia.com/Download/Find.aspx?lang=en-us>.

Какую версию выбрать? Для выбора правильной версии используйте таблицу, представленную на рис. 13.3. Поскольку не все версии драйверов протестированы для работы с той или иной версией дистрибутива, руководствуйтесь этой таблицей, чтобы загрузить драйвер, который точно в вашем дистрибутиве будет работать.

Fedora 26	Fedora 25	Fedora 24	Fedora 23/22/21
384.59 (July 24, 2017)	384.59 (July 24, 2017)	384.59 (July 24, 2017)	384.59 (July 24, 2017)
381.22 (May 9,2017)	381.22 (May 9, 2017)	381.22 (May 9, 2017)	381.22 (May 9, 2017)
375.82 (July 24, 2017)	375.82 (July 24, 2017)	375.82 (July 24, 2017)	375.82 (July 24, 2017)
340.102 (February 14,2017)	340.102 (February 14, 2017)	340.102 (February 14, 2017)	340.102 (February 14, 2017)
304.135 (February 14,2017)	304.135 (February 14, 2017)	304.135 (February 14, 2017)	304.135 (February 14, 2017)

Рис. 13.3. Протестированные версии драйверов NVIDIA

Обычно закиваемый драйвер загружается в папку -/Downloads и носит имя NVIDIA-Linux-XXXX.run. Сразу, чтобы потом не забыть, установите право выполнения этого файла:

```
cd -/Downloads
chmod +x NVIDIA-Linux-*.run
```

Теперь обновим менеджер пакетов — напомним, в Fedora 21 в этом качестве служит yum, а в Fedora 22 и более новых — dnf:

```
sudo dnf update
```

или

```
sudo yum update
```

Потом обновим ядро (в Fedora 21 вместо параметра dnf впишите yum):

```
sudo dnf install kernel-devel kernel-headers gcc dkms acpid
```

Затем нужно отключить nouveau (бесплатный драйвер для карт NVIDIA с открытым исходным кодом):

```
echo "blacklist nouveau" » /etc/modprobe.d/blacklist.conf
```

Отредактируем теперь файл /etc/sysconfig/grub, добавив строку rd.driver.blacklist=nouveau в конец параметра GRUB_CMDLINE_LINUX:

```
GRUB_CMDLINE_LINUX="rd. lvm. lv=fedora/swap rd. lvm. lv=fedora/root rhgb quiet  
rd.driver.blacklist=nouveau"
```

Обновим конфигурацию GRUB2:

```
sudo grub2-mkconfig -o /boot/grub2/grub.cfg
```

Правда, если у вас UEFI, то команда будет другой:

```
sudo grub2-mkconfig -o /boot/efi/EFI/fedora/grub.cfg
```

Удалим с чистой совестью стандартный драйвер:

```
sudo dnf remove xorg-x11-drv-nouveau
```

Сгенерируем другой initramfs:

```
sudo mv /boot/initramfs-$(uname -r).img /boot/initramfs-$(uname -r)-nouveau.img  
sudo dracut /boot/initramfs-$(uname -r).img $(uname -r)
```

Отключим SELinux — для этого откройте файл /etc/sysconfig/selinux и добавьте строку:

```
SELINUX=disabled
```

Переключитесь на то, что раньше называлось *третьим уровнем выполнения* (runlevel, т. е. многопользовательский режим без поддержки графики):

```
sudo systemctl set-default multi-user.target  
reboot
```

После перезагрузки запустите инсталлятор драйвера (xxxx — здесь версия драйвера и его модификация, например, x86_64):

```
cd ~/Downloads  
sudo ./NVIDIA-Linux-XXXX.run
```

Установка драйвера проста — на все вопросы инсталлятора отвечайте **Yes** или **OK** (рис. 13.4), а по окончании процесса вы должны получить соответствующее сообщение (рис. 13.5).

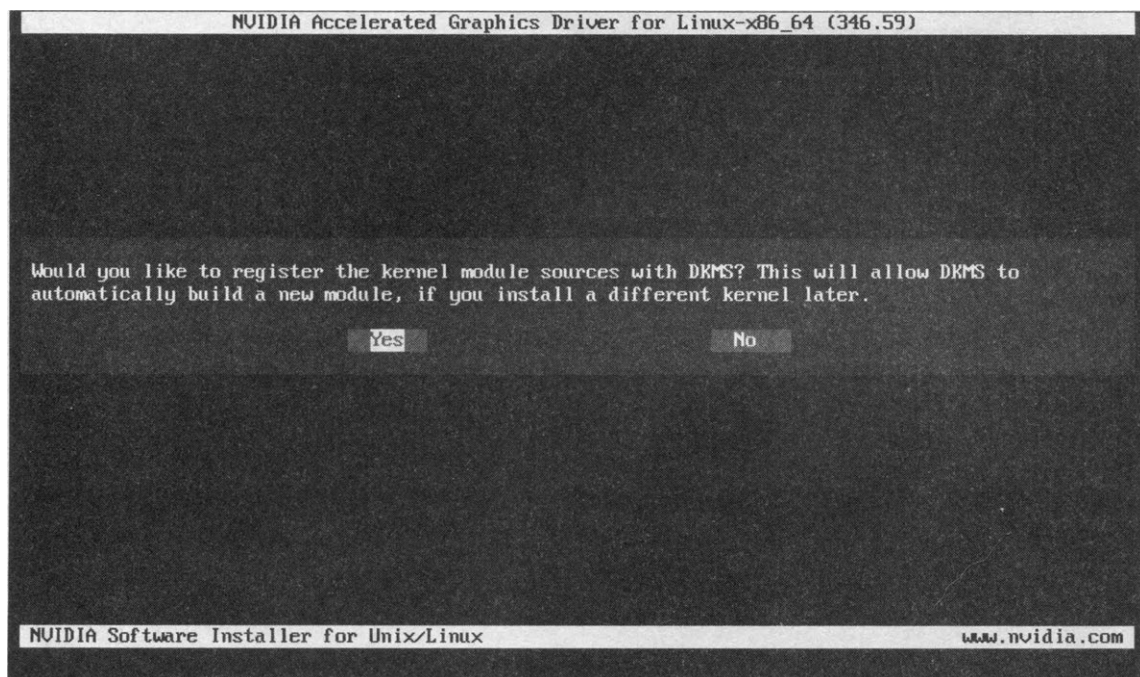


Рис. 13.4. Нажмите Yes

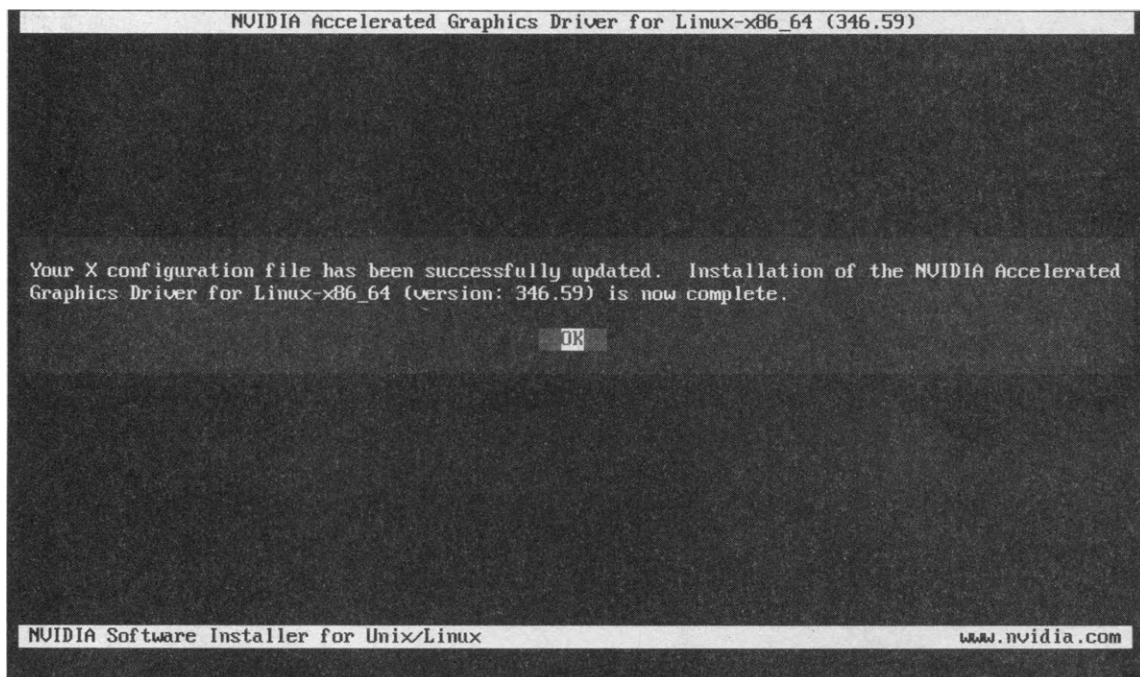


Рис. 13.5. Драйвер установлен

Восстанавливаем запуск графического интерфейса:

```
sudo systemctl set-default graphical.target  
reboot
```

После перезагрузки нужно включить видеоускорение (для этого нужна карта GeForce 8 или выше):

```
sudo dnf install vdpauinfo libva-vdpau-driver libva-utils
```

Собственно, на этом и все — теперь можно пользоваться полноценными проприетарными драйверами для карт NVIDIA. Можно также запустить NVIDIA X Server Settings (вы найдете эту настройку в списке приложений), чтобы просмотреть версию установленного драйвера. На рис. 13.6 показано, что драйверы корректно установлены в Fedora 26.

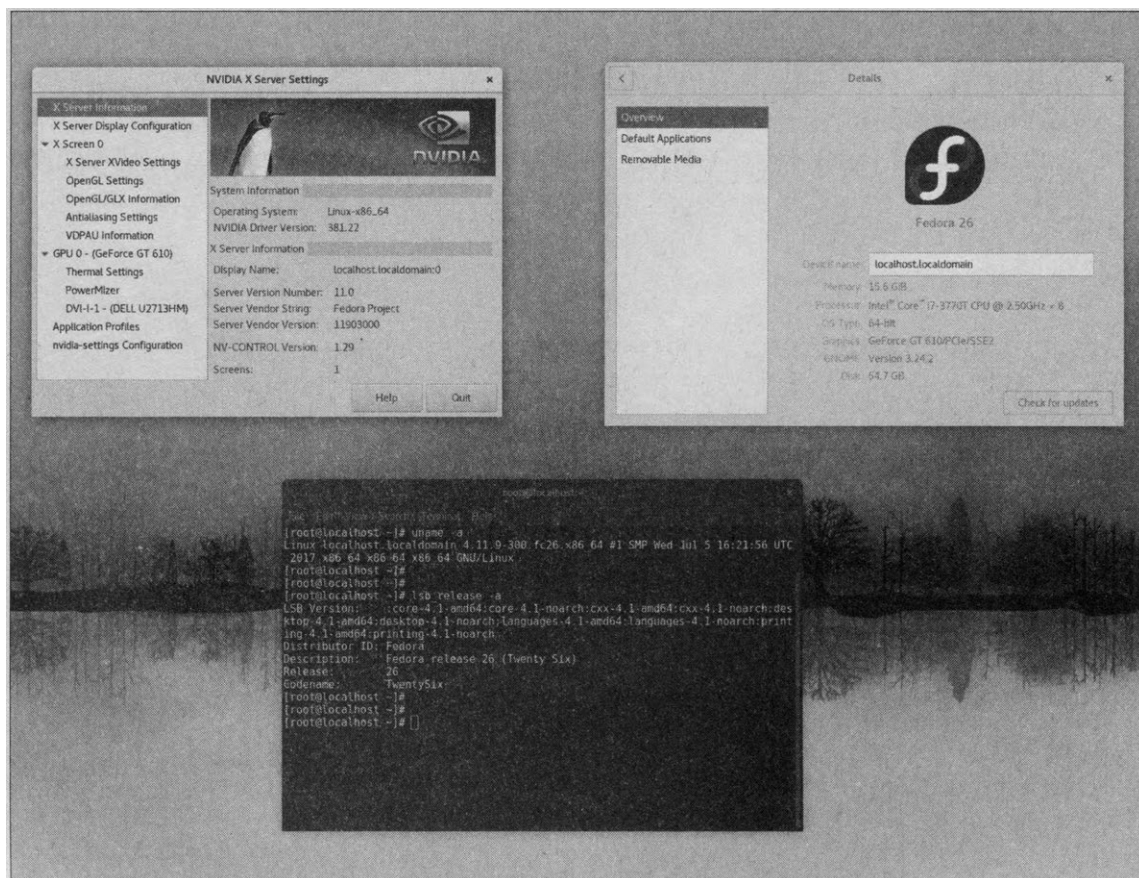
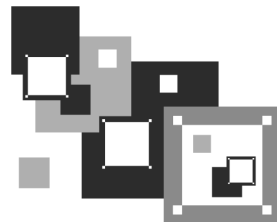


Рис. 13.6. Fedora 26: драйверы NVIDIA установлены

ГЛАВА 14



Офисные пакеты

14.1. Выбор офисного пакета

Для Linux, как и для любой другой операционной системы, предполагающей использование на так называемых *настольных компьютерах* (стационарные персональные компьютеры, ноутбуки, нетбуки), доступно несколько офисных пакетов, и пользователь вправе выбрать тот, который ему больше нравится. Наш небольшой обзор мы начнем со стандартного офисного пакета LibreOffice, доступного по умолчанию в большинстве дистрибутивов Linux.

14.1.1. LibreOffice

Пакет LibreOffice основан на пакете OpenOffice.org (о нем мы поговорим в *разд. 14.3*), в том или ином варианте входит в состав всех современных дистрибутивов Linux и представляет собой полноценный офисный пакет, содержащий средства для работы с текстом, электронными таблицами и презентациями, — как раз с тем, ради чего в большинстве случаев и устанавливают Microsoft Office. LibreOffice поддерживает форматы файлов Microsoft Office, в том числе и его последних версий (2007/2010/2013).

Внешне LibreOffice похож на Microsoft Office 2003, что в некоторых случаях даже хорошо, учитывая, что интерфейс 2003 весьма прост, и лишь необходимость поддержки новых форматов документов вынуждает пользователей использовать последние версии Microsoft Office. Здесь же вы получаете два в одном: и привычный интерфейс (рис. 14.1), и поддержку новых форматов (рис. 14.2).

Программы, входящие в состав LibreOffice, не сложнее аналогичных программ из других офисных пакетов. Да, они несколько своеобразные, и к ним нужно немного привыкнуть. Не стоит ожидать также, что LibreOffice — это точная копия MS Office 2003, хоть он во многом и похож на него.

Учитывая полную русификацию LibreOffice, с ним без проблем смогут разобраться даже самые начинающие пользователи.

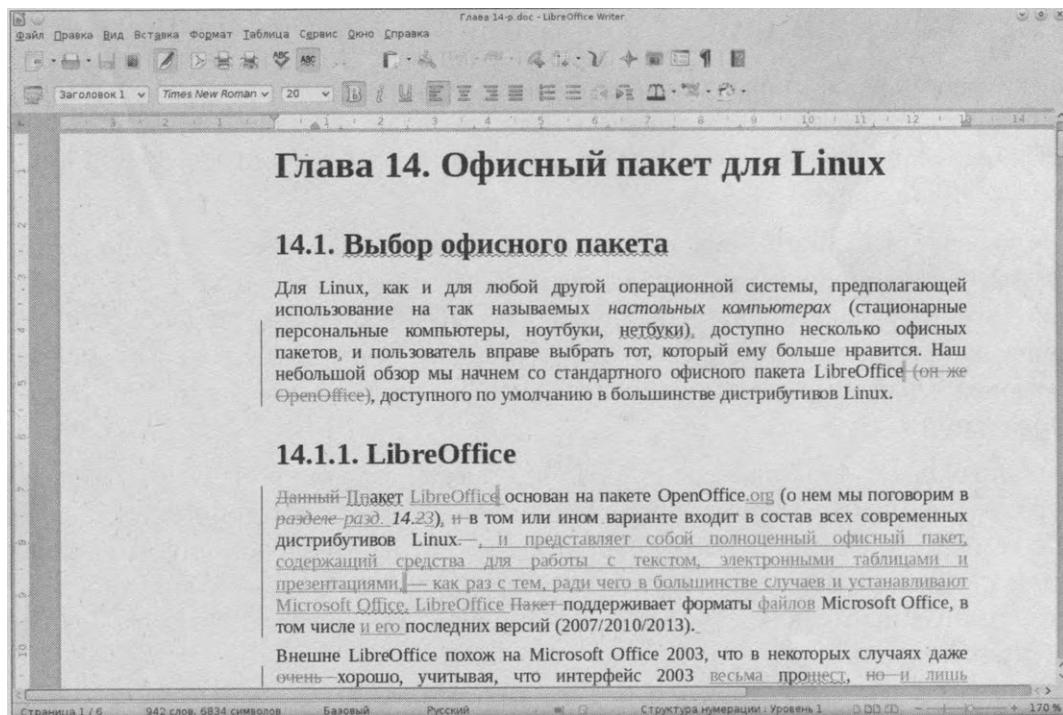


Рис. 14.1. Программа LibreOffice Writer—аналог MS Word

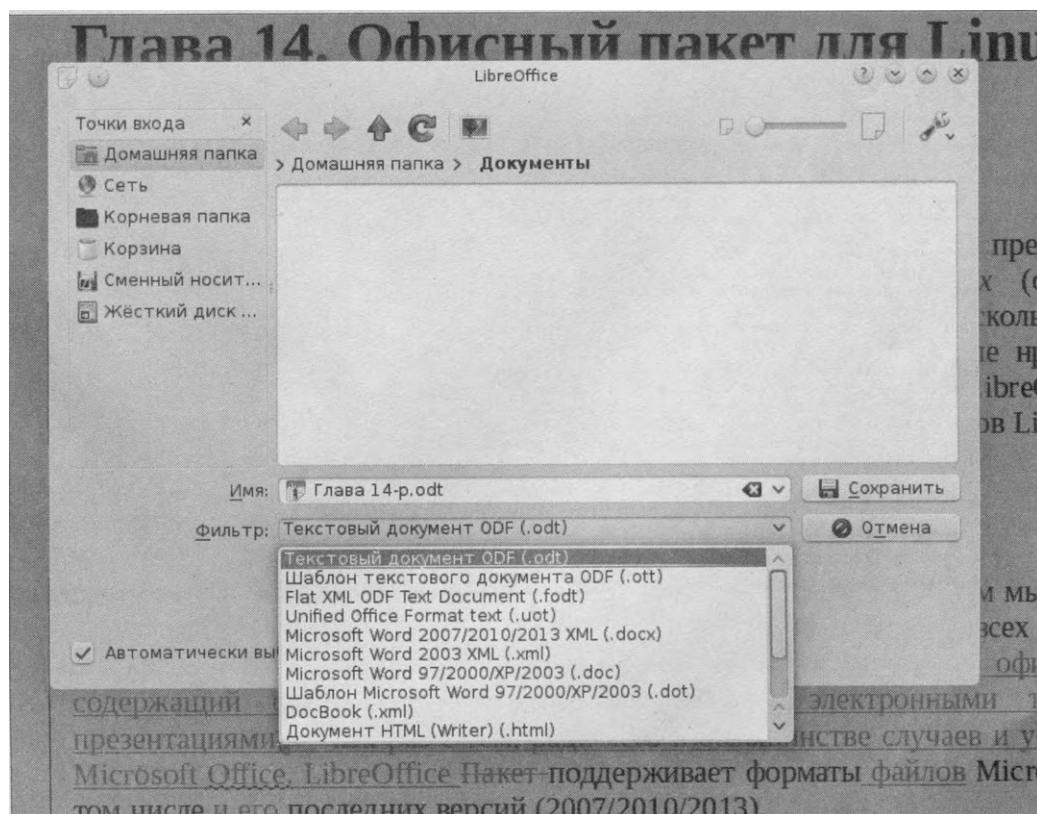


Рис. 14.2. Программа LibreOffice Writer: поддерживаемые форматы

14.1.2. Calligra Suite

Если вы работали со старыми дистрибутивами, то наверняка помните неказистый офисный пакет KOffice, созданный командой KDE и устанавливаемый вместе с графической средой KDE. В 2010 году от проекта KOffice отделилась ветка, названная Calligra Suite.

В состав пакета Calligra Suite входят программы Words, Sheets и Stage, которые являются функциональными эквивалентами Microsoft Word, Excel и PowerPoint. Имеются в этом пакете еще и оболочка для работы с базами данных, векторный графический редактор, программа для рисования блок-схем, а также программа управления проектами (есть там и более мелкие программы — вроде нотного редактора и пр.).

Как видите, пакет Calligra Suite достаточно богат по функционалу, вот только его программы не очень удобные, я бы даже сказал — просто неудобные, если сравнивать с тем же LibreOffice: в меню, например, достаточно мало команд, и большая их часть перенесена в панель инструментов с вертикальными вкладками (рис. 14.3) — такая организация меню требует некоторой привычки. Оставляет желать лучшего и локализация пакета.

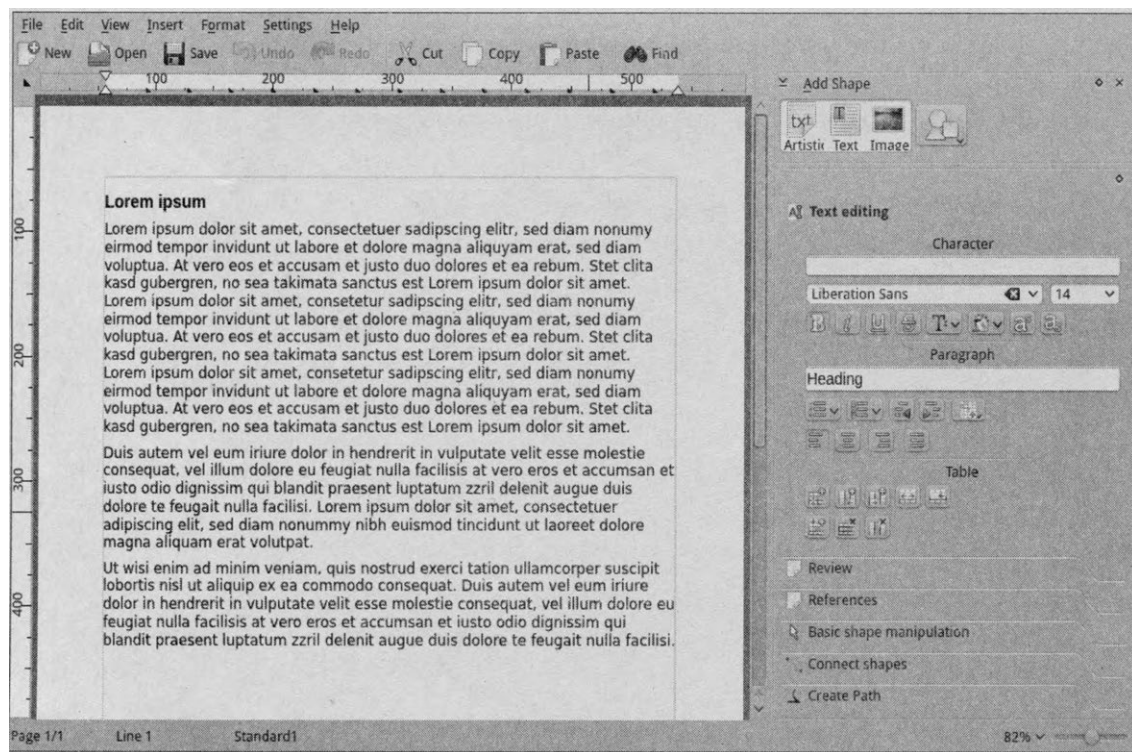


Рис. 14.3. Программа Calligra Words — аналог MS Word

Да и некоторые функции в Calligra Suite работают немного не так, как вы бы ожидали. Например, попробуйте создать в Calligra Sheets график по какому-либо набору данных. Сначала вы обнаружите, что сам график строится довольно долго, а

если выяснится, что вы допустили ошибку, то отредактировать таблицу данных для него не удастся, — придется удалять график и строить его заново. Раздражает также и принятый в Sheets способ перемещения между ячейками — вы не можете просто использовать клавиши управления курсором, и приходится каждый раз нажимать клавишу <Enter>. Похоже, что разработчики Calligra пытались в своем офисном пакете создать «изюминку», и у них это получилось...

14.1.3. Kingsoft Office

Этот офисный пакет наверняка знаком пользователям Android, поскольку является лучшим офисным пакетом для смартфонов и планшетов. Существуют версии этого офисного пакета также и для Linux и Windows. Единственным дистрибутивом Linux, в котором установлен этот пакет «из коробки», является Linux Deepin. Скачать офисный пакет Kingsoft Office можно по адресу: <http://www.kingsoftstore.com/software/kingsoft-office-freeware>.

В состав пакета Kingsoft Office входят текстовый процессор, электронная таблица и программа для создания презентаций, поэтому Kingsoft Office называют также WPS Office. Здесь WPS — аббревиатура входящих в состав офисного пакета компонентов: Writer, Presentation и Spreadsheet.

Внешне Kingsoft Office напоминает последние версии Microsoft Office (рис. 14.4), но вы можете переключиться и на классический интерфейс. Любопытно, что выбрать скин можно для каждого компонента пакета по отдельности.

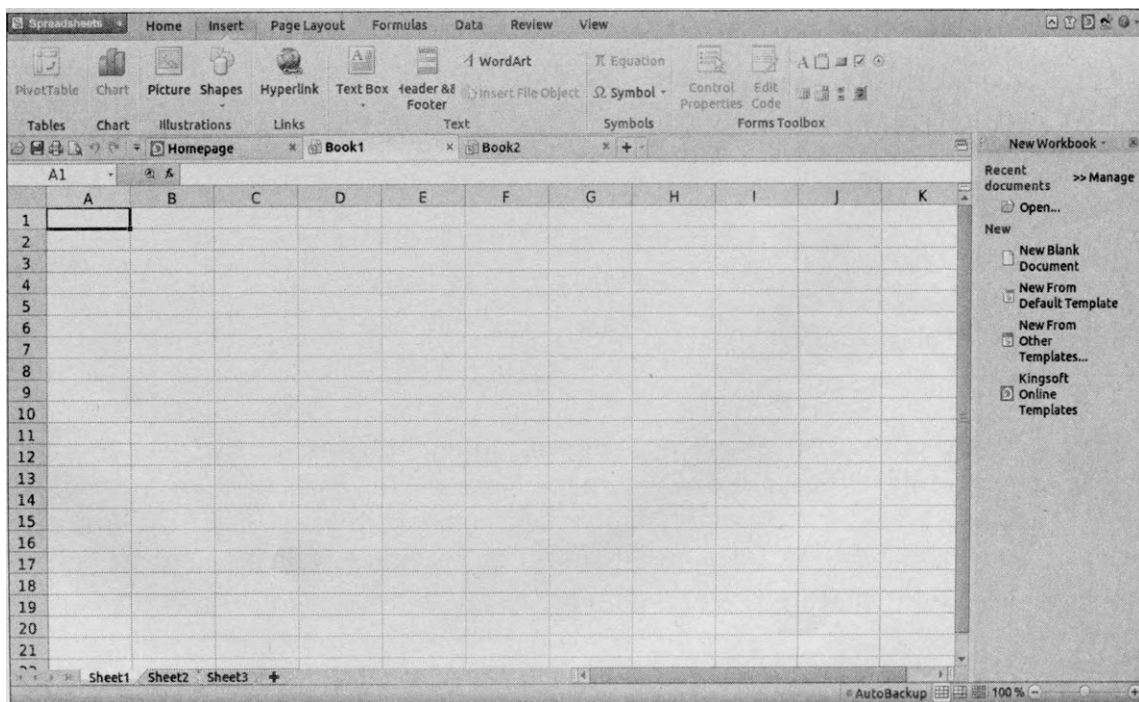


Рис. 14.4. Программа Kingsoft Office Spreadsheet — аналог MS Excel

Программы пакета Kingsoft Office весьма удобны в работе, а их интерфейс наиболее дружелюбный из рассмотренных в этой главе офисных пакетов. Особых недостатков у него нет (если не считать отсутствия локализации). Он поддерживает как собственный формат файлов, так и формат файлов MS Office, однако не поддерживает формат ODF, поэтому вы не сможете открыть в нем документы, созданные в LibreOffice. Впрочем, к недостаткам это отнести сложно, поскольку в большинстве случаев из соображений совместимости все документы, как правило, сохраняются в формате MS Office. А наличие Android-версии делает его еще более привлекательным.

* * *

Конечно, LibreOffice (OpenOffice.org), WPS Office и Calligra Suite— далеко не единственные офисные пакеты. Например, есть еще офисный пакет SoftMaker Office и ряд других, однако они не нашли широкого распространения. Стандартом де-факто в мире Linux является офисный пакет OpenOffice.org и его клон LibreOffice.

14.2. Кроссплатформенная совместимость

При выборе офисного пакета для организации следует учитывать два фактора: совместимость с Microsoft Office и совместимость с остальным компьютерным парком. Если вы выбираете офисный пакет для себя любимого и только для одного компьютера, можете взять любой офисный пакет. А вот если вы выбираете пакет программ для предприятия, важно, чтобы этот пакет можно было установить на все его компьютеры. Информация о поддерживаемых операционных системах теми или иными пакетами приведена в табл. 14.1.

Таблица 14.1. Информация о кроссплатформенной совместимости офисных пакетов

Офисный пакет	Linux	Windows	MacOS	Android
LibreOffice/OpenOffice.org	+	+	+	-
Kingsoft (WPS) Office	+	+	-	+
Calligra Suite	+	-	-	-

Наиболее универсальным вариантом, как можно видеть, для настольных компьютеров является LibreOffice. Пакет Calligra из-за отмеченных ранее его особенностей вы использовать вряд ли станете. Ради справедливости нужно отметить, что поддержка Windows и Mac OS в нем предусмотрена, но она сильно ограничена, поэтому в таблице и стоят соответствующие прочерки. Если же вам нужна поддержка Android, то следует посмотреть в сторону Kingsoft Office. Впрочем, для Android существует приложение AndrOpen Office, которое является первой в мире попыткой портировать OpenOffice на Android, но оно пока далеко от идеала.

14.3. Вкратце об OpenOffice.org

Изначально этот офисный пакет был разработан компанией Sun и назывался StarOffice. Я еще успел застать его первые версии, и поверьте — на фоне того, что в то время было написано для Linux, он стал настоящим прорывом.

Позже, когда компания Sun была куплена компанией Oracle, пакет переименовали в Oracle Open Office.

Непосредственно OpenOffice.org появился как Open-Source версия (версия с открытым исходным кодом) пакета StarOffice в 2002 году. Пакет LibreOffice отделился от OpenOffice.org в 2010 году. По сути, это тот же OpenOffice, но с небольшими изменениями.

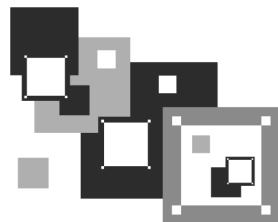
В 2011 году компания Oracle (которая является собственником OpenOffice.org) объявила о прекращении поддержки коммерческой версии пакета и передала проект организации ASF (Apache Software Foundation). Именно поэтому с 2011 года OpenOffice.org называется Apache OpenOffice.

Как вы уже, наверное, поняли, OpenOffice.org, LibreOffice и Apache OpenOffice — это братья-близнецы. Выбор того или иного пакета в большей степени будет зависеть только от того, какой пакет входит в состав вашего дистрибутива. Если в вашем дистрибутиве по умолчанию используется LibreOffice (как, например, в Debian 8), вряд ли вы станете устанавливать Apache OpenOffice и наоборот.

Есть и другие клоны OpenOffice.org, например, NeoOffice — это специальный форк OpenOffice.org, доступный только для Mac OS.

Эта книга, хоть и рассчитана на разную аудиторию: от новичков до профессионалов, но трудно назвать абсолютным новичком пользователя, который сумел и не побоялся установить Linux на свой компьютер. Поэтому, думаю, ему будет не сложно разобраться, какие кнопки в офисном пакете нужно нажимать, чтобы отформатировать текст или создать диаграмму. Именно поэтому я, прислушавшись к критике читателей, решил удалить подробное описание офисного пакета из пятого издания этой книги, чем сэкономил несколько десятков страниц. А если вы считаете, что оно, все же, в книге присутствовать должно, свяжитесь со мной на форуме сайта dkws.org.ua, а я подумаю, что с этим можно сделать.

ГЛАВА 15



Графический редактор GIMP

Графический редактор GIMP, особенно его вторая версия,— достойный Linux-аналог известной программы Photoshop.

В большинстве случаев работа обычных пользователей с тем же Photoshop — поскольку созданием профессиональных шедевров двумерной графики они, как правило, не занимаются, — сводится к нескольким несложным операциям с фотографиями: изменению размера, повороту, кадрированию и размытию отдельных их участков. Именно эти операции мы здесь и рассмотрим. Кстати, в фотостудиях их выполнение не столь уж и дешево. Например, печать стандартной фотографии открыточного формата с цифрового носителя в среднем стоит 8-10 рублей, а за кадрирование с вас потребуют рублей 40. Если фотография одна, то это не слишком обременительно, а вот если их 10, то неразумно платить лишние деньги за то, что можно сделать самому с помощью GIMP, потратив 5-10 минут.

Цель этой главы — коротко познакомить вас с программой GIMP. Если вы заинтересовались, то для более подробного знакомства рекомендую свою книгу «GIMP 2 — бесплатный аналог Photoshop для Windows/Linux/Mac OS, 2-е изд.», ее страница на сайте издательства: <http://bhv.ru/books/book.php?id=186881>.

15.1. Начало работы

Ранее при первом запуске GIMP запрашивал, сколько оперативной памяти можно выделить под нужды приложения. В версии GIMP 2.8.x, которая входит в состав всех современных дистрибутивов Linux, размер кэша установлен по умолчанию в 1024 Мбайт (рис. 15.1). Конкретная величина требуемого кэша зависит от объема имеющейся оперативной памяти и от размера фотографий, с которыми вам приходится работать, однако установленного по умолчанию резерва, как правило, вполне достаточно. Если вы по каким-то соображениям захотите изменить размер кэша, сделать это можно с помощью команды меню **Правка | Параметры | Окружение**.

После запуска программы вы увидите три окна GIMP: панель инструментов, основное окно (находится по центру экрана и содержит меню) и окно **Слои, Каналы, Контуры** (рис. 15.2).

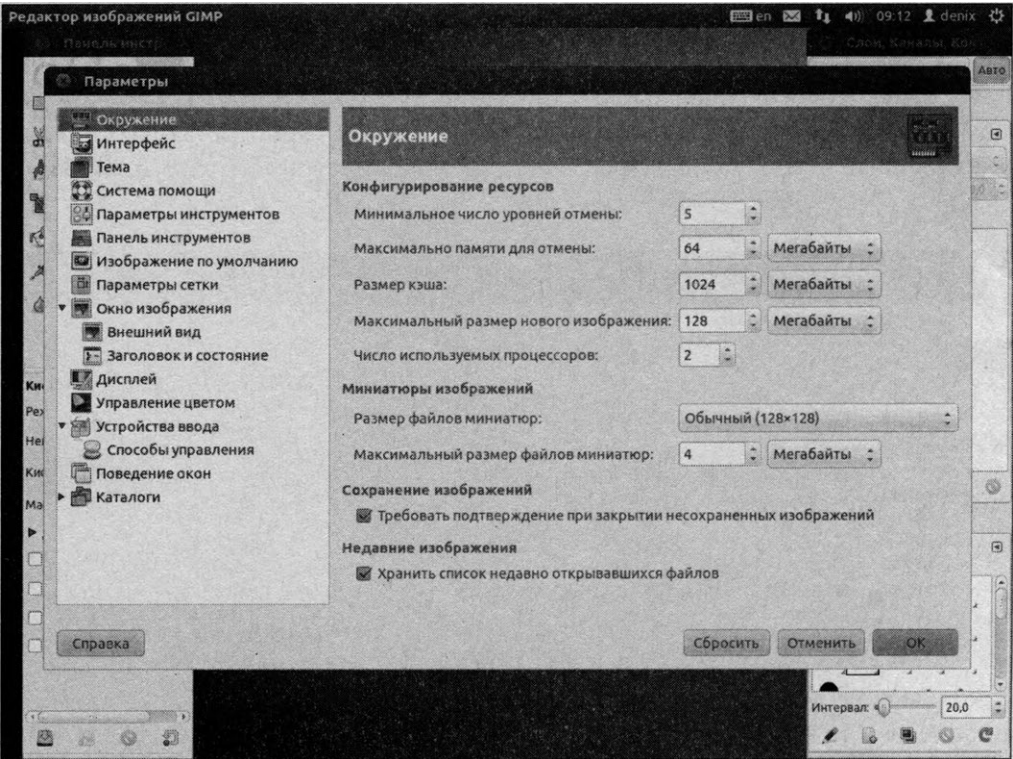


Рис. 15.1. Ubuntu: настройка кэша GIMP

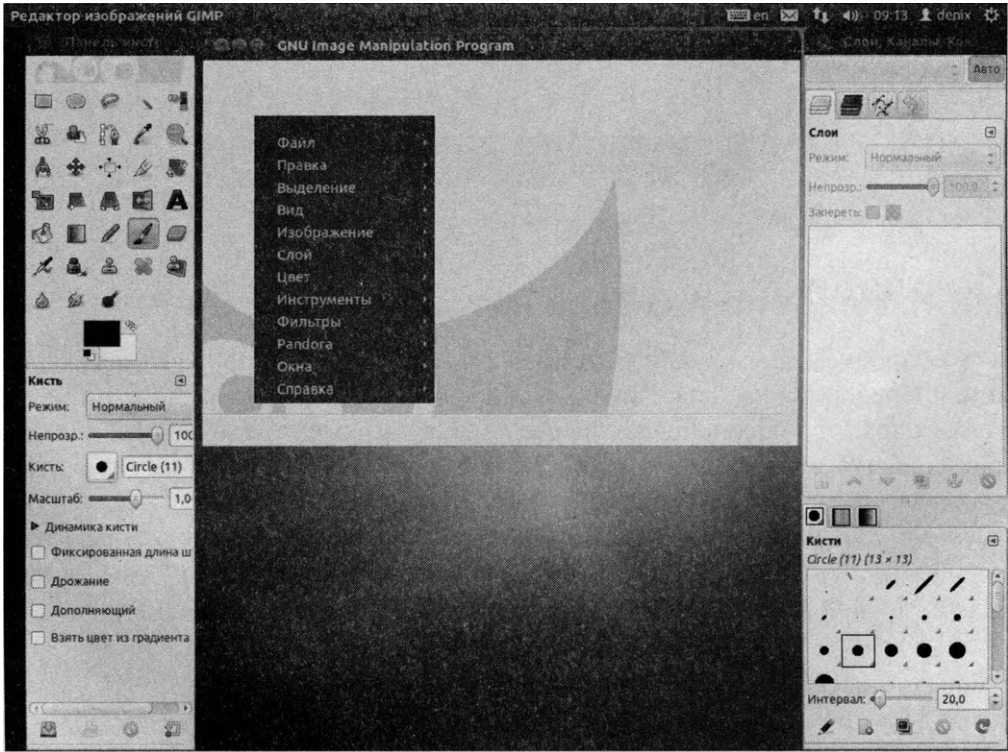


Рис. 15.2. Ubuntu: GIMP в работе

15.2. Обработка фотографий

Чтобы открыть фотографию, выполните команду меню **Файл | Открыть** или просто нажмите комбинацию клавиш **<Ctrl>+<O>**. Окно открытия файла содержит область предварительного просмотра, что позволяет быстро выбрать нужный снимок (рис. 15.3).

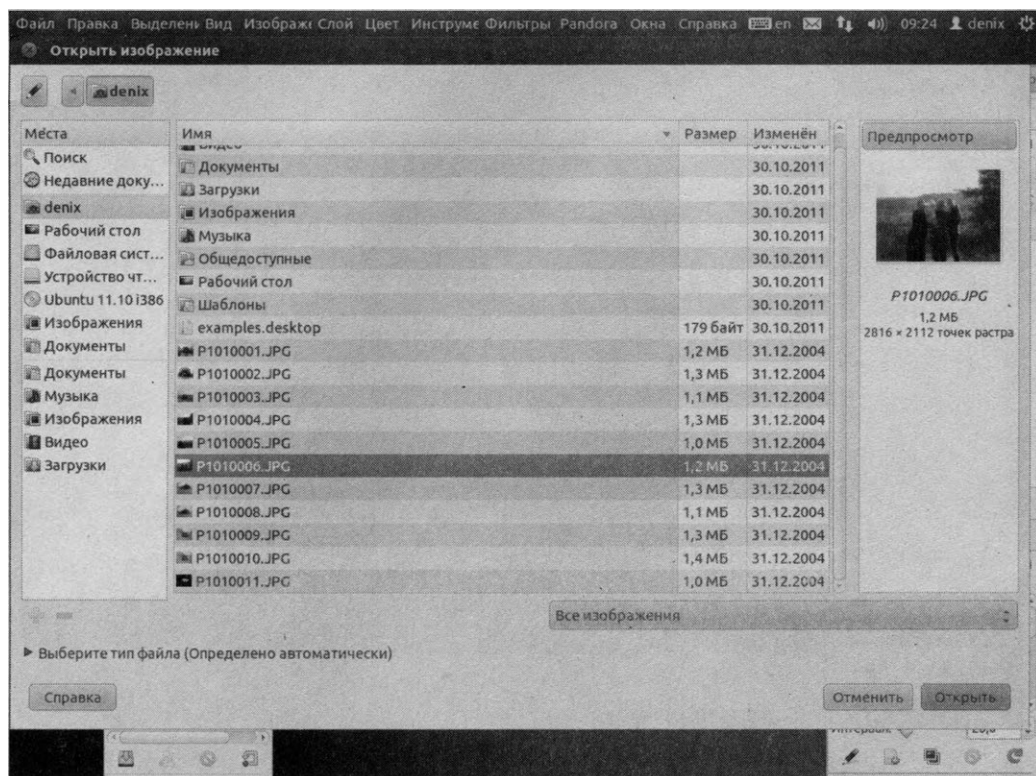


Рис. 15.3. Ubuntu: открытие фотографии в GIMP

15.2.1. Изменение размера (масштабирование)

Давайте попробуем для начала изменить размер изображения. Это весьма важная операция. Предположим, вы снимаете цифровым фотоаппаратом с матрицей на 6 мегапикселей, что обеспечивает размер файла фотографии в 3-4 Мбайт. Для печати фотоснимка это, конечно, хорошо. А вот если вы захотите отправить такой файл кому-то по Интернету для просмотра на компьютере, получатель будет не очень доволен. Во-первых, размер файла для пересылки великоват, а, во-вторых, просматривать картинку окажется неудобно— придется уменьшать ее масштаб, чтобы фотография поместилась на экране целиком. Уменьшив размер до пересылки, мы автоматически и прямо пропорционально уменьшим и размер файла.

Итак, приступим к изменению размера изображения, которое в GIMP называется *масштабированием*. После открытия файла картинка появится в новом окне. Щелкните по ней правой кнопкой мыши и из открывшегося меню выберите команду **Изображение | Размер изображения** (рис. 15.4).

В окне масштабирования установите новый размер фотографии в пикселах (рис. 15.5) или же выберите опцию **Процент** (из списка единиц измерения, который находится справа от поля **Высота**) и введите новый размер фотографии в процентах от оригинала.

Затем нажмите кнопку **Изменить** — размер фотографии будет изменен.

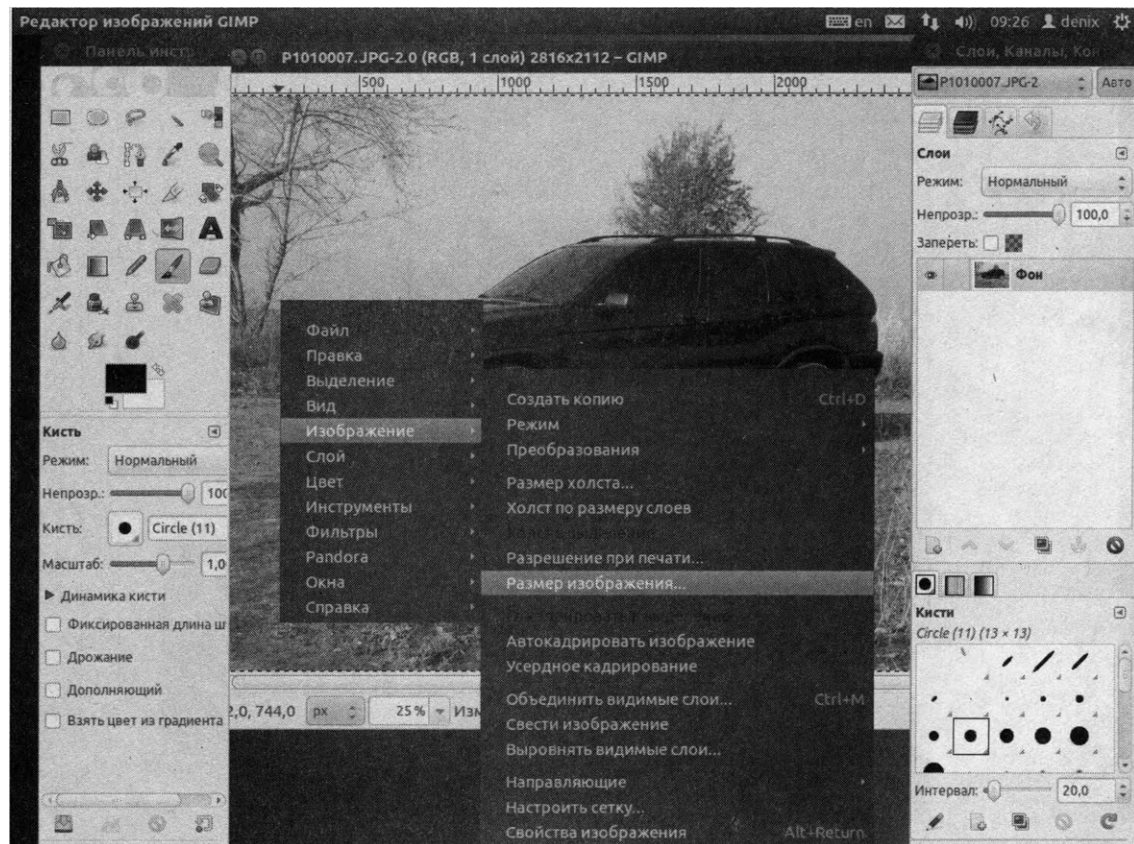


Рис. 15.4. Ubuntu: выбор в GIMP команды изменения размера изображения

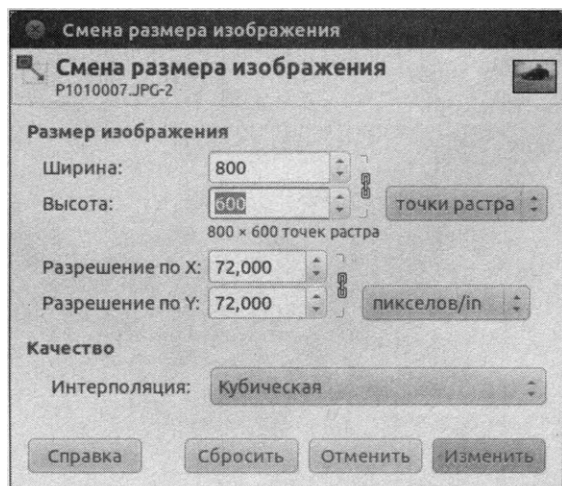


Рис. 15.5. Ubuntu:
окно масштабирования
GIMP

15.2.3. Кадрирование (обрезка)

Рассмотрим теперь операцию кадрирования. Она заключается в вырезании части изображения: сначала вы выделяете нужную вам область, затем выполняете операцию, после чего все, что находится за пределами выделенной вами области, будет удалено.

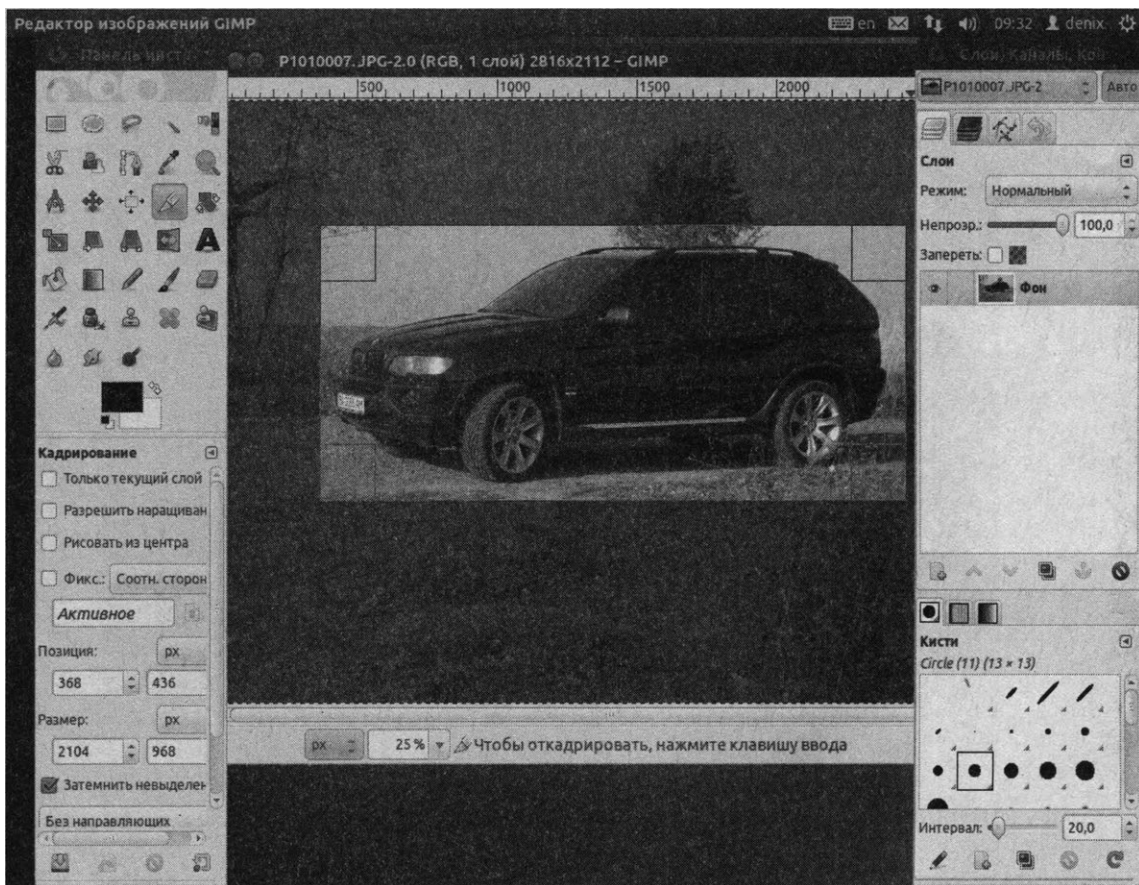


Рис. 15.8. Ubuntu: выделение в окне GIMP области для кадрирования

Для кадрирования нажмите комбинацию клавиш **<Shift>+<C>** — указатель мыши примет форму скальпеля. Выделите им прямоугольную область (рис. 15.8) и установите на панели инструментов (рис. 15.9) дополнительные параметры кадрирования, если в них будет необходимость. Чтобы обрезать выделенную часть изображения, нажмите клавишу **<Enter>** или щелкните по выделению двойным щелчком. Результат кадрирования представлен на рис. 15.10.

Если у вас что-то не получилось, нажмите комбинацию клавиш **<Ctrl>+<Z>** для отмены последней операции.

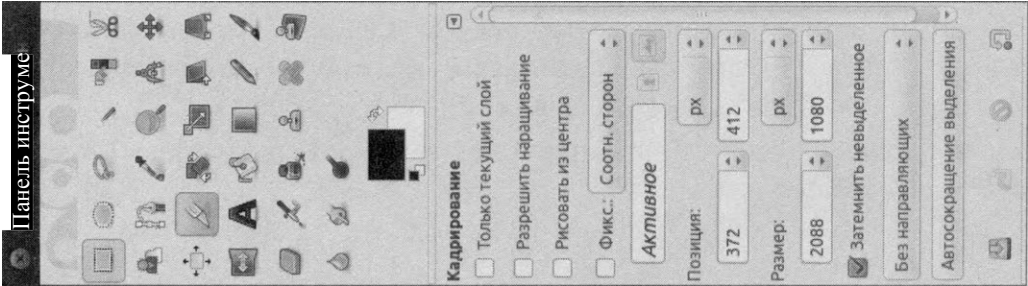


Рис. 15.9. Ubuntu: параметры

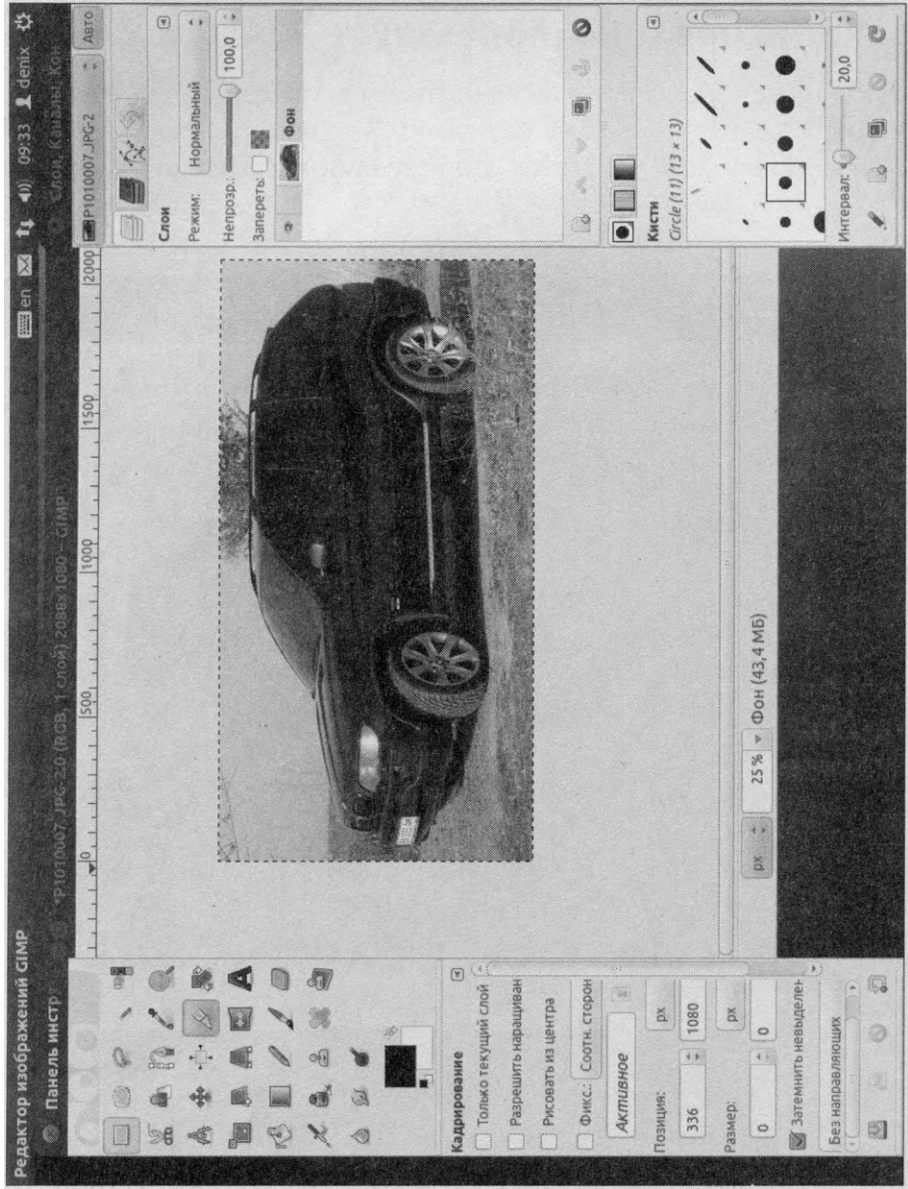


Рис. 15.10. Ubuntu: результат кадрирования в окне GIMP

15.2.4. Инструмент *Размывание-Резкость*

Иногда нужно «размыть» некоторые участки картинки или же, наоборот, придать некоторым участкам больше резкости.

Но чаще все-таки используется размывание — для сокрытия некоторых участков фотографии, которые совсем необязательно видеть посторонним. Например, довольно часто можно встретить объявления о продаже автомобилей с фотографиями, на которых «размыт» государственный номер.

Для размывания можно использовать инструмент Размывание-Резкость (рис. 15.11) — активируйте его, выберите кисть (обычно используется круглая кисть), установите режим (резкость или размывание) и скорость. Теперь вам остается только «размыть» участок изображения. Результат размывания представлен на рис. 15.12.

ВИДЕОУРОК ПО GIMP

На страничке <http://dkws.org.ua/novice/> вы найдете небольшой пятиминутный видеоролик по GIMP, в котором продемонстрированы описанные здесь основные операции.

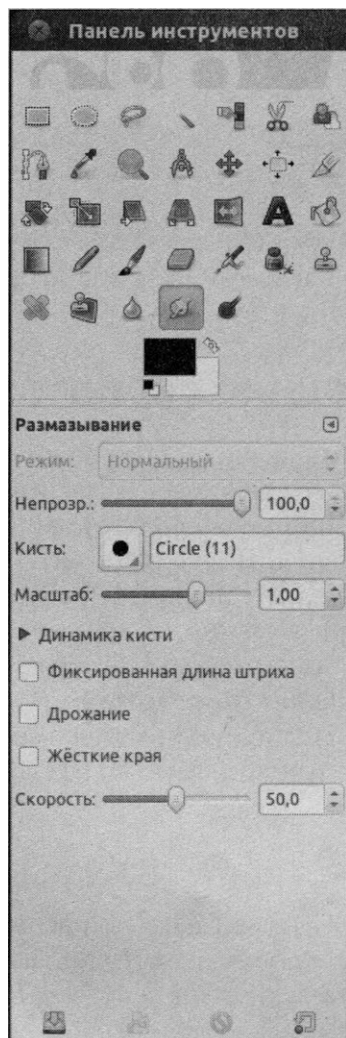


Рис. 15.11. Ubuntu: параметры инструмента GIMP Размывание-Резкость

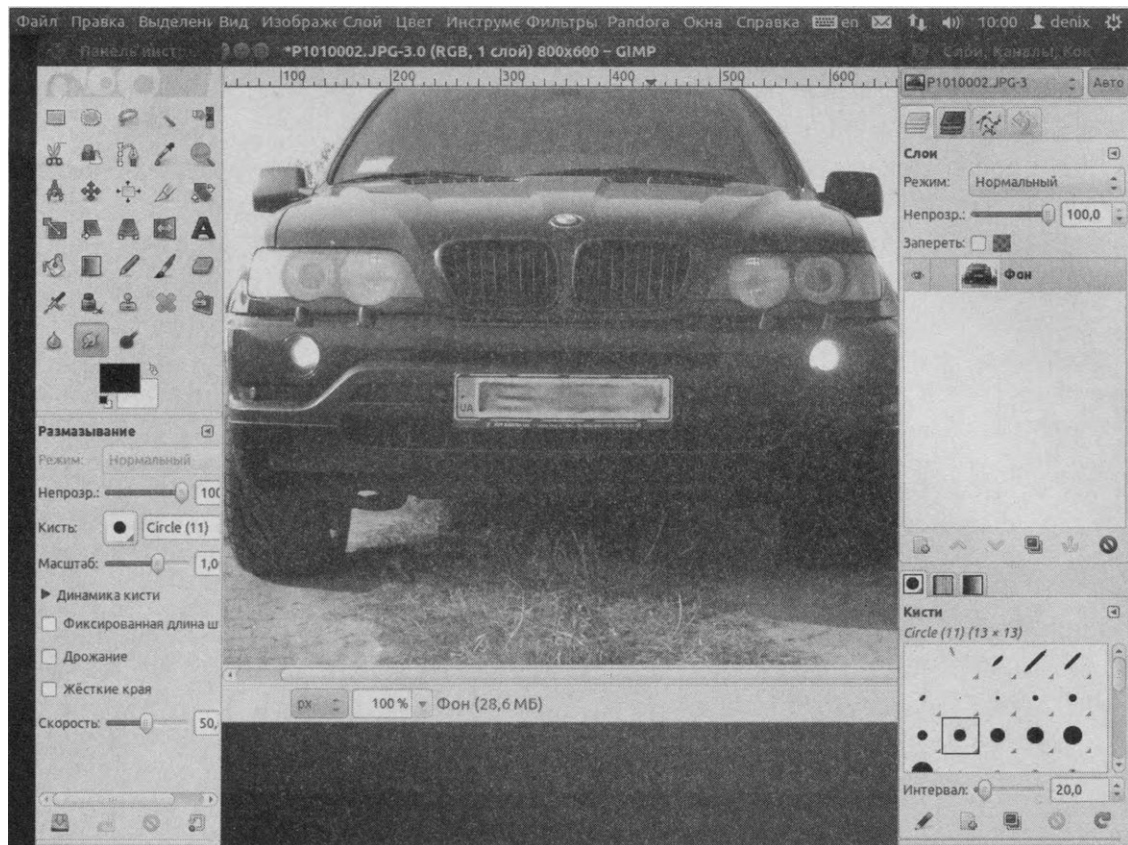


Рис. 15.12. Ubuntu: результат размывания в окне GIMP

15.3. Работа в GIMP с помощью скриптов

Стоит отметить, что кроме обычного редактирования фотографий GIMP позволяет изменять изображения с помощью сценариев (скриптов). Загрузите любое изображение, щелкните на нем правой кнопкой мыши и выберите команду меню **Фильтры | Скрипт-Фу** — вы увидите, что в состав GIMP входит много различных интересных скриптов. Если же вам чего-то не хватает, поищите требуемое в Сети или создайте самостоятельно — в Интернете при желании вы найдете руководство по созданию собственных скриптов и уже готовые их коды. Много разных скриптов можно также скачать по адресу <http://gug.sunsite.dk/scripts.php>. Особо останавливаться мы на этом не станем — поэкспериментируйте с имеющимися скриптами, и результат вас не разочарует: лучше один раз увидеть, чем 100 раз услышать.

15.4. Windows-версия GIMP

Далеко не у всех получается полностью перейти на Linux — пока еще для Linux не созданы аналоги всех Windows-программ. Например, в Linux нет полноценных CAD-систем, популярная программа бухгалтерского учета «1С» запускается только в эмуляторе, не говоря уже об отсутствии для Linux полноценных игр.

Что делать, если вы практически всегда работаете в Windows, например, из-за той же CAD-системы, но вам понадобилось отредактировать фотографии? Переустанавливать Linux из-за пары фотографий не хочется. Использовать пиратские версии Photoshop тоже. А лицензионные стоят хороших денег. Но выход есть — это Windows-версия программы GIMP, которую можно бесплатно скачать по адресу: <http://gimp-win.sourceforge.net/>.

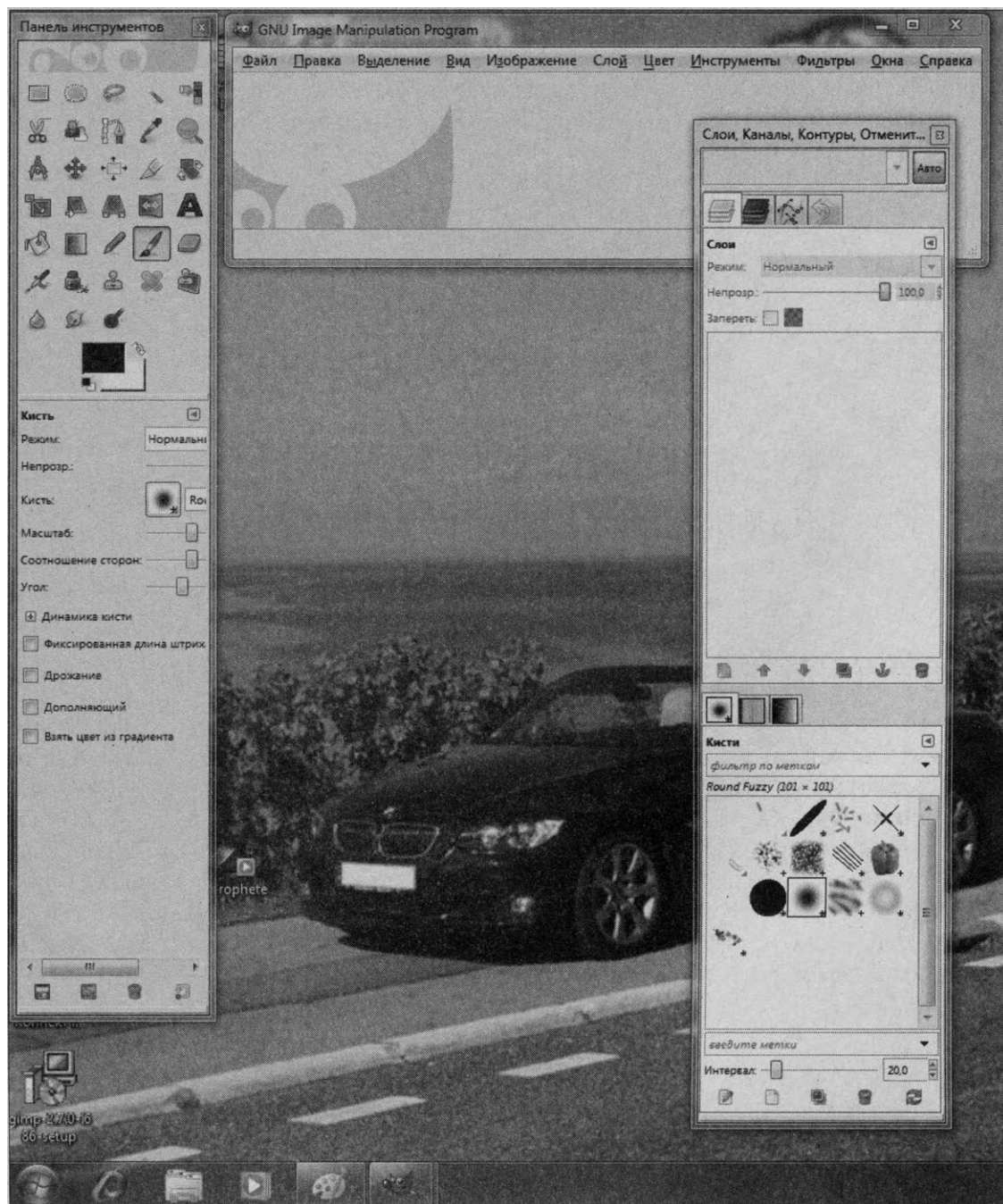
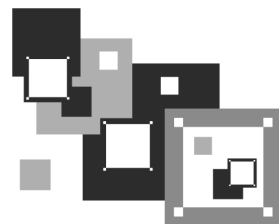


Рис. 15.13. Windows-версия GIMP

Про Windows-версию GIMP (рис. 15.13) вам нужно знать следующее:

- она абсолютно бесплатна, поэтому вы можете ее использовать безо всяких ограничений;
- она аналогична Linux-версии по функциональности;
- для работы Windows-версии нужна операционная система Windows 2000/XP/Vista/7 и старше. На более древних версиях Windows GIMP работать не станет;
- антивирус Касперского в программе установки GIMP ошибочно определяет наличие вируса, поэтому на момент установки GIMP этот антивирус лучше отключить. Не беспокойтесь — это ошибка антивируса Касперского, никаких вирусов в дистрибутиве GIMP нет.

ГЛАВА 16



Лазерные диски и программы для их «прожига»

16.1. Что нужно для записи CD и DVD?

Что же необходимо для комфортной записи CD и DVD? Давайте подумаем вместе.

- *Компьютер* — это само собой разумеется. С одной стороны, ультрасовременный компьютер никогда не помешает, а с другой, CD и даже DVD можно записывать далеко не на самых мощных компьютерах.
- *Свободное место на жестком диске* — это самое важное условие. Ведь перед «прожитом» на винчестере компьютера создается *образ* диска, который будет потом записан на болванку. Учитывая, что системе необходимо место под этот образ, для записи одного CD на жестком диске потребуется минимум 700 Мбайт свободного пространства (если вы записываете обычный диск объемом 700 Мбайт).

При записи DVD требуемый объем свободного пространства зависит от его емкости (4,7; 9,0 или 17 Гбайт). Так, для записи двустороннего двухслойного DVD вам понадобится около 18 Гбайт свободного места на винчестере, если, конечно, вы планируете записать DVD полностью.

- *Привод для записи CD/DVD* — перед записью убедитесь, что ваш привод поддерживает носитель выбранного типа. Иногда встречаются так называемые *комбинированные* приводы — они могут читать и CD, и DVD, а вот записывают только CD. Перед покупкой привода нужно уточнить, может ли он записывать DVD, чтобы отсутствие этой возможности потом не стало для вас неприятным сюрпризом. Если у вас привод, который не может записывать ни CD, ни DVD, лучше приобрести тот, что пригоден для записи дисков обоих типов, даже если вы не планируете пока записывать DVD. Разница в цене небольшая, а дополнительные возможности никогда не помешают.
- *Программное обеспечение для записи дисков* — в Linux есть программы, позволяющие записывать диски как из командной строки, так и в графическом режиме, что, безусловно, намного удобнее. Какую программу выбрать — личное дело каждого, а в этой книге будет описана очень удобная, на мой взгляд, программа K3b.

- *Чистые диски* — главное, не забывать заранее покупать так называемые «болванки». Обычные компакт-диски бывают двух типов: CD-R и CD-RW. Первые служат для однократной записи, вторые можно перезаписывать многократно. А вот о дисках DVD стоит поговорить особо.

16.2. Отдельно о DVD

DVD заслуживают отдельного разговора. А как может не заслужить этого с виду обычный компакт-диск, но емкостью от 4,5 до 17 Гбайт? Итак, отвлечемся немного и подробнее поговорим о DVD.

Ранее под аббревиатурой DVD подразумевали Digital Video Disc, т. е. цифровой видеодиск. Но поскольку на DVD можно записывать не только видео, но и музыку, фотографии, а также обычные файлы, со временем DVD «переименовали» в Digital Versatile Disc — цифровой универсальный диск.

Размеры диска и центрального отверстия в нем для CD и DVD совпадают, однако, если присмотреться, заметно, что DVD немного толще.

16.2.1. История создания DVD

Нужно отметить, что технология DVD развивалась значительно быстрее, чем в свое время технология CD. Объяснить это можно тем, что компакт-диск тогда был чем-то новым и, возможно, ему не так доверяли, как другим проверенным носителям данных (дискетам и магнитным лентам). Да и стоили первые CD-приводы совсем не дешево. А когда появился DVD, весь мир уже знал, что такое компакт-диск. Поэтому DVD был воспринят как компакт-диск большого объема. По сути, так оно и есть.

Если бы не Голливуд, то, возможно, сейчас у нас не было DVD, или он появился бы значительно позже. Именно Голливуд подтолкнул в 1994 году крупнейшие компании (Sony, Phillips и Toshiba) к созданию нового формата записи и хранения данных. Во-первых, сами понимаете, оптический диск гораздо надежнее, чем магнитная лента видеокассеты, а 700 Мбайт, помещавшихся на обычном CD, для качественной записи фильма слишком мало. Во-вторых, в то время на Западе (как, наверное, сейчас у нас) процветало видеопиратство. Доходило до анекдотических ситуаций — фильм еще не вышел, а у пиратов уже была в продаже кассета с ним. Видеокассеты не предусматривали никакой защиты от копирования, поэтому потребовался цифровой формат, позволяющий защитить информацию (прежде всего, видео) от нелегального копирования и хранить на одном носителе большие объемы данных.

Спустя два года, в 1996 году, были разработаны первые спецификации DVD: DVD-ROM (для хранения данных) и DVD-Video (для видеофильмов). Именно поэтому DVD сначала и назывались цифровыми видеодисками (Digital Video Disc).

Еще через два года появилась спецификация DVD-RW, а также организация DVD Forum, призванная координировать действия всех производителей DVD и

приводов для них. В 1998 году в состав этой организации входило более 120 компаний, так или иначе связанных с разработкой DVD. В этом же году вышел в свет стандарт DVD-Audio, а емкость обычных DVD была увеличена до 4,7 Гбайт.

В 2000 году были созданы первые проигрыватели DVD-Audio. Стоили они дорого, и поначалу купить их можно было только в США, — в Европе они появились в продаже ближе к концу года. В это же время были разработаны и более быстрые DVD-приводы для компьютера.

В 2002 году состоялся анонс формата Blu-ray, позволяющего хранить до 50 Гбайт информации на одном диске. Хотя приводы и диски этого формата сейчас в продаже имеются, они не пользуются серьезной популярностью среди пользователей, даже несмотря на то, что их цена уже вполне приемлема. Подробную информацию о формате Blu-ray можно найти в Википедии по адресу: http://ru.wikipedia.org/wiki/Blu-ray_Disc

Основная проблема этого формата заключается в том, что он немного опоздал со своим появлением. Blu-ray хоть и был анонсирован в 2002 году, но вышел в 2006-м, и поначалу широкого распространения не получил из-за высокой стоимости устройств чтения/записи, а также самих дисков. А сейчас все идет к тому, что от оптических дисков пользователи отказываются все чаще и чаще. А зачем, когда есть флешка? Флешку можно подключить к чему угодно: к автомобильной магнитоле, к домашнему DVD-проигрывателю и даже просто к телевизору. И записать данные на флешку гораздо удобнее, чем на лазерный диск.

Ранее CD/DVD-диски служили для хранения данных (у самого множество дисков с резервными копиями десятилетней давности), музыки и видео. Затем приоритет их использования сместился в сторону музыки и видео, т. к. не все устройства поддерживали работу с флешками. А что же резервные копии? Их сейчас домашние пользователи (как и маленькие предприятия) хранят на внешних жестких дисках, а предприятия (средние и крупные) — на сетевых хранилищах (NAS) с RAID.

Таким образом, в 2016-м году эта глава выглядит жизнеописанием вымерших динозавров. Именно поэтому мною было принято решение несколько ее подсократить. Информация о форматах дисков останется (см. далее), а вот из всего обилия программ для их «прожига» мы познакомимся только с K3B (для KDE) и Brasero (для GNOME), — это отличные программы, и нет смысла рассматривать какие-либо их аналоги, при том, что сами они работают без нареканий. Программа Brasero несколько простовата, зато K3B по своему функционалу способна полностью заменить Nero for Linux, поэтому описания проприетарной программы Nero в этом издании вы не найдете.

16.2.2. Преимущества и недостатки DVD

У всего есть свои преимущества и свои недостатки. Есть они и у DVD. Начнем с преимуществ:

- *большая емкость диска* — лишнего места не бывает! Но, с другой стороны, 4,7 Гбайт хорошо для записи фильма в цифровом качестве (или коллекции

фильмов в MPEG-4). Для хранения данных такой объем слишком велик. Мне пока еще не удавалось сразу заполнить всю DVD-болванку своими документами. В лучшем случае диск заполняется постепенно,— просто дописываешь в «конец» всю нужную информацию на протяжении какого-то времени. И все же один DVD удобнее, чем шесть обычных CD;

- *поддержка различных видеоформатов* — DVD-Video совместим с экранными форматами 4:3 и 16:9. Опять-таки— это важно для видео, а для других целей несущественно;
- *многоканальный звук* — на DVD можно записать до 8 различных аудиопотоков. Это очень ценная возможность, например, для диска, где есть 3 аудиопотока: оригинал (без перевода), перевод на русский язык и перевод «от Гоблина». Фактически мы получаем как бы три фильма в одном. Ведь в случае с CD пришлось бы записывать все это на три разных диска: на одном был бы фильм в «оригинале», на втором — фильм с переводом, а на третьем — небольшая пародия, которую, тем не менее, интересно посмотреть, особенно если до этого видел обычный фильм. Так же имеет место и запись аудиопотоков на разных языках — ведь диск может распространяться, например, по всей Европе, где в каждой стране свой язык;
- *поддержка до 9 различных углов зрения для камер* — хотите посмотреть на понравившуюся сцену под другим углом, например, с противоположной стороны? DVD-позволяет и это. Главное, чтобы такая возможность была предусмотрена при записи фильма;
- *совместимость со звуковыми форматами Dolby Digital, Dolby Digital Pro и Dolby Surround (многоканальный звук 5+1)* — если у вас есть домашний кинотеатр, то вы оцените это преимущество. Если же смотреть фильм на компьютере, то особой разницы между звуком DVD-фильма и звуком фильма в MPEG-4 вы не почувствуете;
- *интерактивное управление* — даже на DVD-проигрывателе (не говоря уже о компьютере) вы можете полностью управлять просмотром фильма. Нет, это не просто пауза, останов, воспроизведение и перемотка. Это вызов меню фильма, переключение звуковых каналов, управление последовательностью сцен, изменение камер обзора, чтение записанных на DVD текстов, вызов субтитров, просмотр записанных картинок и т. д.;
- *поддержка «закладок»* — предположим, вы смотрели фильм, но вам потребовалось куда-то срочно уехать (или просто захотелось спать). Тогда можно сделать «закладку», а потом продолжить просмотр с того же самого места. Согласитесь, удобно, а на CD такого нет;
- *надежность, дешевизна и компактность DVD* в сравнении с VHS-кассетами;
- *дешевизна DVD-проигрывателей*— сейчас DVD-проигрыватель, который в состоянии читать даже MPEG-4, стоит от 800 рублей. Обычный DVD (без поддержки MPEG-4) — еще дешевле.

А недостатка всего два:

- *некоторые проблемы с совместимостью* — существуют два несовместимых между собой формата DVD: DVD-R/-RW и DVD+R/+RW. Не все проигрыватели могут воспроизводить оба типа дисков. Есть трудности и с поддержкой редкого формата DVD-RAM;
- *все DVD-проигрыватели привязаны к региону, в котором они продаются*, — это сделано в целях борьбы с видеопиратами. Поэтому покупать проигрыватель нужно там, где вы живете. Были случаи, когда проигрыватель, привезенный, скажем, из Азии, не мог читать диски, которые распространяются на территории РФ. При записи DVD-Video прописывается регион, в котором должен распространяться диск. DVD-проигрыватель прежде всего считывает с диска код региона, и если он не совпадает с кодом самого проигрывателя, просмотреть мы ничего не можем. Обидно, но ради этого и разрабатывался DVD.

16.2.3. Форматы и маркировка DVD-дисков

Как уже было отмечено, DVD — это всего лишь улучшенная модификация CD. При разработке технологии DVD решили пойти не по качественному пути, а по количественному — просто повысили плотность записи. Конечно, при этом были разработаны более совершенные методы коррекции ошибок, дополнительные способы оптимизации дискового пространства, но суть от этого не меняется.

Приведем классификацию DVD-дисков:

- DVD-5 — односторонний однослойный диск емкостью 4,7 Гбайт;
- DVD-9 — односторонний двухслойный 8,54 Гбайт;
- DVD-10 — двусторонний однослойный 9,4 Гбайт;
- DVD-18 — двусторонний двухслойный 17 Гбайт.

Наиболее распространены диски DVD-5 и DVD-10. Диски DVD-9 встречаются реже, а DVD-18 вообще сложно найти в продаже. Большинство современных проигрывателей способны работать с односторонними одно- и двухслойными дисками (для чтения второй стороны диск нужно перевернуть). Иногда еще встречаются аппараты, предназначенные для чтения только однослойных DVD. Впрочем, приводы DVD постоянно совершенствуются.

Теперь перечислим форматы DVD:

- **DVD-ROM** — базовый формат для массового производства дисков, например, дисков с фильмами. Этот формат поддерживает файловые системы UDF и ISO 9660 (как для обычных CD), однако порядок физического размещения файлов задается спецификацией DVD-Audio и DVD-Video;
- **DVD-Video** — предназначен для хранения фильмов и представляет собой «логическую надстройку» над DVD-ROM. Формат задает порядок расположения файлов на диске. Кроме записи фильмов и сопровождающих их звуковых потоков, на такие диски допускается записывать картинки (которые можно будет

просматривать с помощью средств навигации DVD-проигрывателя), субтитры на разных языках и диалоговые окна. Не возбраняется записать и любые другие файлы — они будут проигнорированы домашним DVD-проигрывателем, однако доступ к ним можно получить, вставив диск в компьютер.

Особенности DVD-Video:

- на односторонний однослойный диск в этом формате можно записать 133 минуты фильма со звуком. Если фильм не умещается на таком носителе, можно выбрать DVD большей емкости;
 - многоканальный звук (до 8 каналов);
 - surround-звук — это отдельный канал для баса;
 - стандартные экранные форматы 4:3 (обычное телевидение) и 16:9 (широкоформатное видео);
 - защита от нелегального копирования;
 - кодирование регионов распространения;
 - субтитры на 32 языках;
 - интерактивное управление;
- **DVD-Audio** — формат для записи высококачественного звука. Известно, что звук в формате MP3 при воспроизведении на профессиональном проигрывателе звучит хуже, чем в формате AudioCD. Так вот, качество звукового потока формата DVD-Video намного лучше, чем AudioCD, а звучание DVD-Audio превосходит даже DVD-Video. Чувствуете, насколько хорош этот формат? Появился он не так давно — в 1999 году. Правда, в течение первого года DVD-Audio существовал только в лаборатории — проигрыватели, поддерживающие этот формат, появились в 2000 году, а еще через год мир увидел первый коммерческий диск DVD-Audio. На сегодняшний день DVD-Audio признан лучшим аудиоформатом. Высокое качество звучания достигается благодаря сжатию без потерь (алгоритм LPCM) — таким образом все пространство в 4,7 Гбайт используется исключительно для звука, что позволяет сохранить оригинальное качество звучания;
- **DVD-R** — это однократно записываемый диск, позволяющий записывать все, что угодно: музыку, документы, фильмы, картинки или все сразу. Лишь бы у вас был привод, поддерживающий запись DVD. Существуют две разновидности DVD-R: обычный (для некоммерческого использования) и для продюсеров (DVD-Authoring, позволяющий создавать мастер-диски и обладающий возможностями защиты от нелегального копирования). Технически разница заключается в различной длине волны лазера при записи (635 нм для обычных DVD и 650 нм — для DVD-Authoring). Кроме того, для записи DVD-Authoring нужен специальный привод, не совместимый с обычным. Но оба типа приводов могут читать оба типа дисков. Этот факт нужно учитывать при покупке привода DVD-RW или при покупке «болванок» (если привод у вас уже есть) — нет смысла приобретать более дорогой DVD-Authoring, поскольку все равно вы не сможете его записать. Впрочем, не думайте, что на DVD-R можно записывать только

файлы. Из чистого DVD-R вы сможете создать диск любого формата: DVD-Video, DVD-Audio, DVD-ROM, но без защиты диска от нелегального копирования;

- **DVD-RW и DVD-RAM** — перезаписываемые диски. На такой диск вы можете записать информацию, затем стереть все, потом заново записать и т. д. — как и в случае с дисками CD-RW. Перезаписываемые диски маркируются DVD-RW, иногда встречается маркировка DVD-RAM, — отличие у них в числе циклов перезаписи: DVD-RW можно перезаписывать сотни раз, а DVD-RAM — сотни тысяч раз. Хотя DVD-RAM намного надежнее, но процесс перезаписи такого диска весьма длительный (в среднем, на запись диска нужен 1 час). Есть и неоспоримое преимущество — для записи таких дисков не требуется создавать образ на жестком диске, можно сразу писать прямо на носитель. Это очень важно — ведь не всегда на жестком диске есть 5 (или более) Гбайт свободного места. Так что, если на винчестере свободно хотя бы 200 Мбайт, вы сможете записать диск DVD-RAM полностью. С другой стороны, DVD-RAM подходит только для компьютера, поскольку пока нет DVD-проигрывателей, которые читают диски этого формата.

В целом, перезаписываемые диски не столь надежны, как DVD-R. Если вам нужно записать диск для многократного чтения (например, фильм, который вы потом одолжите всем своим друзьям, и они по несколько раз его посмотрят), то лучше выбрать DVD-R, поскольку есть вероятность, что когда он к вам вернется, его все еще можно будет прочитать. А вот если вы хотите перенести файлы из офиса домой или наоборот, то DVD-RW — лучшее решение (не DVD-RAM, а именно DVD-RW, поскольку в случае с DVD-RAM вам придется уйти с работы на час позже, ожидая, пока диск запишется);

- **DVD+R/+RW** — новый формат. Получил знак + в маркировке, чтобы подчеркнуть его превосходство над старыми форматами. Преимущество этого формата заключается в более высокой скорости чтения и записи. Например, для DVD+R скорость записи на момент появления этого формата составляла 4^x, в то время как обычные диски записывались максимум со скоростью 2^x. Помните, что устаревшие приводы для чтения (и DVD-проигрыватели) не способны работать с дисками этого формата, поэтому, если вы покупали свой DVD до 2003 года (или даже в 2003 году, когда появился этот формат), скорее всего, он не будет читать такие диски. Что же касается приводов для записи DVD, то раньше они могли записывать диски или только с «минусом», или только с «плюсом». Современные приводы умеют записывать оба формата. Сейчас можно смело покупать диски и с «плюсом», и с «минусом». Если же вам больше нравится классика, покупайте диски DVD-R — они стоят немного дешевле DVD+R, а скорость их записи составляет на сегодняшний день 16^x и более.

16.2.4. Регионы DVD-Video

О регионах мы уже упоминали, рассмотрим этот вопрос подробнее. Регионы придумали для защиты от несанкционированного распространения дисков. Отчасти это так. Но основная цель — это управление рынком сбыта. Например, состоялась

премьера фильма, и его начали тиражировать на DVD. Понятно, что в Китае, России, в некоторых других регионах диски будут стоить дешевле — мы просто не будем их покупать по европейским ценам. Чтобы в ту же Европу или Америку не импортировали дешевые диски из бедных регионов (где они продаются по более низкой цене), ввели коды регионов, — европейцы просто не смогут посмотреть диски, предназначенные для сбыта в России, и наоборот.

Всего существует 8 основных регионов:

1. США и Канада.
2. Европа, Япония и Южная Африка.
3. Тайвань и Южно-Восточная Азия.
4. Южная и Центральная Америка (в т. ч. Мексика), Австралия и Новая Зеландия.
5. Россия, Пакистан, Центральная и Северная Африка.
6. Китай.
7. Не задан.
8. Используется авиалиниями.

Существует и так называемый *нулевой* регион, который предназначен для некоммерческих записей. Диски, принадлежащие этому «региону», можно просмотреть на любом проигрывателе.

Помните, что регион DVD-диска (как и проигрывателя) нельзя изменить. Регион DVD-проигрывателя устанавливается по первому воспроизведенному диску. Если вы купили DVD-проигрыватель, которым до вас никто не пользовался (абсолютно новый), то, вставив в него диск какого-либо региона, вы переведете проигрыватель в режим, при котором он сможет воспроизводить диски только этого региона. Что же касается компьютерных DVD-приводов, то они официально позволяют 5 раз менять код региона. Нужно быть очень внимательным, чтобы не забыть общее число изменений, — ведь в случае ошибки придется покупать новый DVD-привод. Впрочем, в Интернете можно найти специальные программы, решающие и эту проблему. А можно просто скопировать содержимое DVD на жесткий диск и после этого изменить значение региона.

16.2.5. Некоторые рекомендации относительно DVD

Одни разработчики DVD заявляют, что их диски могут хранить информацию до 50 лет, другие называют цифру в 100 лет. Я им не верю по одной простой причине: первый DVD появился в 1996 году, следовательно, самому «старому» DVD на момент написания этих строк — 20 лет. О каких 100 годах может идти речь, если только за последние годы информационные технологии кардинально изменились, и я не уверен, что еще через 10 лет вы вообще сможете прочесть даже современные DVD, не говоря уже о первых. Не знаю, как будут выглядеть сменные носители будущего, но уж точно не так, как сегодня. Возможно, сменных носителей вовсе не будет, поскольку необходимость в них отпадет. Высокоскоростной Интернет и сегодня доступен любому желающему — намного проще передать файл по Сети, чем

записывать на Flash (или DVD) и через весь город отправлять его адресату (или курьеру, который бы доставил его в другой город), улавливаете логику?

Учитывая механический износ, а также неблагоприятное воздействие окружающей среды (вдруг вы случайно оставите диск на солнышке), рекомендуется перезаписывать диски один раз в год. Диски, которыми вы практически не пользуетесь, можно перезаписывать реже — раз в два года. Но помните, что лучше, когда диск лежит в прохладном (только не в морозилке!) помещении, без прямого воздействия солнечных лучей.

Рекомендуется все свои CD-диски переписать на несколько DVD — так вам будет проще их контролировать. Если, например, у вас было 25 компакт-дисков, то после такой реорганизации станет всего 4-5 DVD. Удобнее, правда?

16.3. Программа K3b

В состав многих дистрибутивов входит очень удобная и простая программа K3b, предназначенная для записи компакт-дисков и DVD. Ее возможности сравнимы с популярной Windows-программой Nero. Программа входит в состав репозитория openSUSE, Fedora и др., а в openSUSE устанавливается по умолчанию.

КАЖДАЯ ПРОГРАММА — ДЛЯ СВОЕЙ СРЕДЫ

Программа K3b предназначена для работы в графической среде KDE, поскольку она использует ее библиотеки. Если на вашем компьютере установлена графическая среда GNOME, то для установки K3b вам придется также установить и библиотеки KDE, которые в таком случае, по сути, больше ни для чего не понадобятся. Так что, если вы предпочитаете графическую среду GNOME, и использование K3b для вас не принципиально, лучше установить программу, написанную с использованием библиотек GNOME, например GnomeBaker или Brasero, — так вы сэкономите место на диске.

Итак, если программы K3b у вас еще нет, установите все пакеты K3b*, кроме пакета k3b-dev, предназначенного для разработчиков.

На рис. 16.1 представлено основное окно K3b, в нижней части которого можно выбрать предполагаемое действие:

- **Новый проект с данными** — записывает CD/DVD с данными;
- **Новый проект Audio CD** — позволяет записать аудиодиск;
- **Копирование диска** — создает копию диска.

Нажав кнопку **Больше действий**, вы увидите список дополнительных действий программы K3b (рис. 16.2):

- **Новый проект с данными** — как уже было отмечено, создает диск с данными;
- **Продолжить мультисессию** — позволяет дозаписать данные на диск, если вы не закрыли сессию в прошлый раз;
- **Новый проект Audio CD** — создает аудиодиск;
- **Новый проект универсального CD** — создает проект диска, позволяющего хранить как данные, так и аудиодорожки. Раньше на таких дисках распростра-

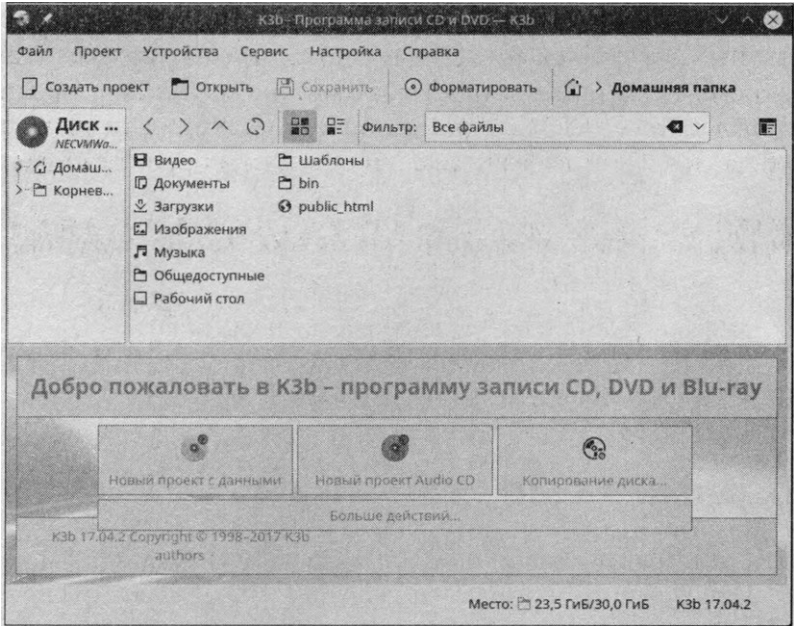


Рис. 16.1. openSUSE: основное окно K3b

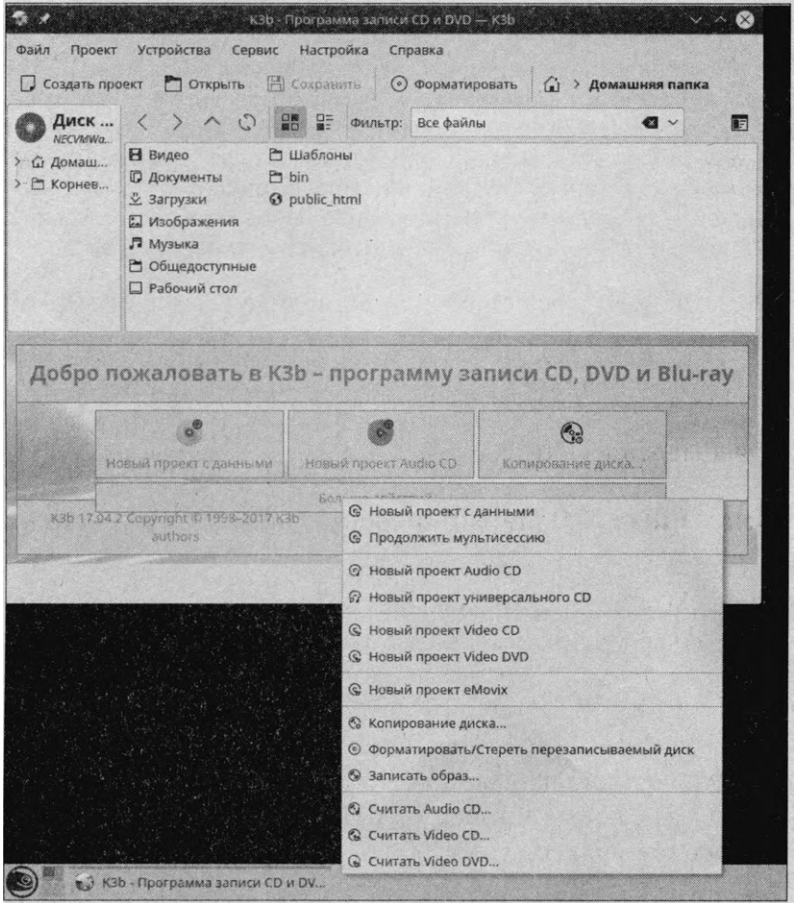


Рис. 16.2. openSUSE: список дополнительных действий K3b

нялись альбомы некоторых исполнителей. В области аудио были композиции, а в области данных — фотографии и другая дополнительная информация. Область данных можно было просмотреть на компьютере, а музыку прослушать на любом CD-проигрывателе. Таким образом, проект универсального диска (рис. 16.3) в рабочей области имеет две секции: секцию звука и проект данных K3b;

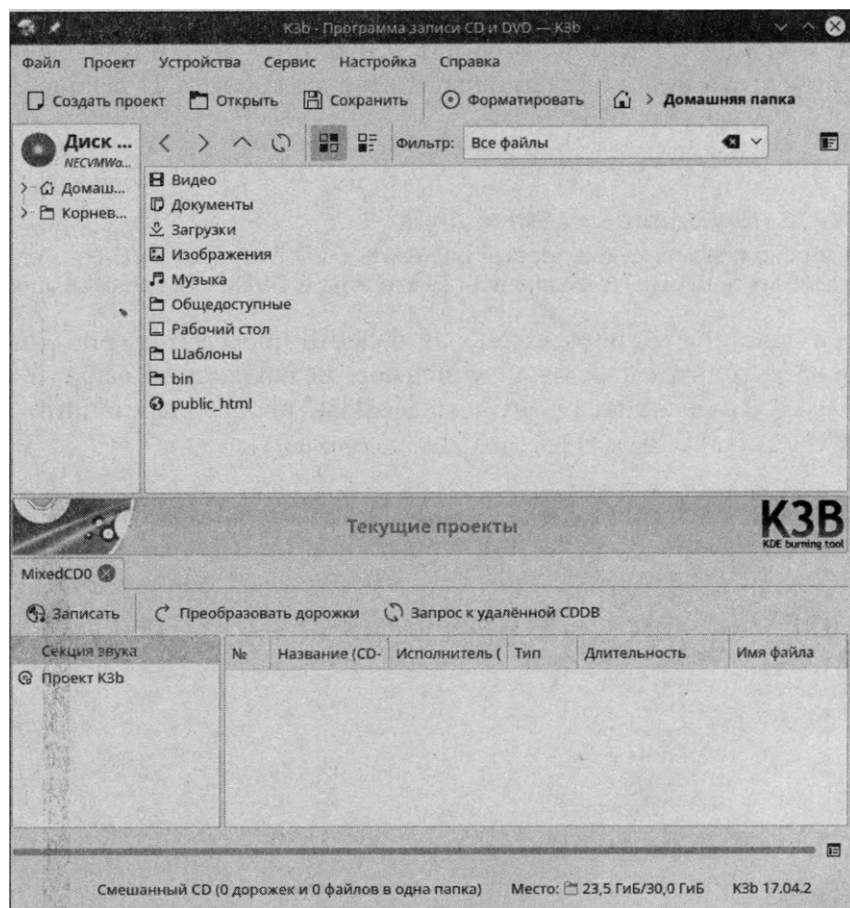


Рис. 16.3. openSUSE: в K3b создан проект универсального диска

- **Новый проект Video CD/Новый проект Video DVD** — создает видеодиск в форматах CD/DVD;
- **Новый проект eMovix** — eMovix представляет собой небольшой дистрибутив, основанный на Slackware, и содержит средства воспроизведения фильмов, записанных на этом же диске. То есть, это действие создает загрузочный диск с мини-дистрибутивом, проигрывателем и фильмом. Дистрибутив поддерживает форматы DivX, MPEG-1, MPEG-2, MPEG-4, RealVideo и много других. Само программное обеспечение занимает всего 8 Мбайт, поэтому все остальное место на «болванке» будет доступно для фильма. Иногда такой проект полезно создать, чтобы была полная уверенность, что фильм удастся просмотреть на любом компьютере даже при отсутствии кодеков. Для воспроизведения фильма нужно просто загрузиться с диска eMovix;

- **Копирование диска** — название действия говорит само за себя;
- **Форматировать/Стереть перезаписываемый диск** — это действие тоже не нуждается в комментариях;
- **Записать образ** — записывает образ диска на «болванку»;
- **Считать Audio CD/Считать Video CD/Считать Video DVD** — помните, раньше были популярны программы-грабберы, позволяющие сохранить дорожки звукового диска на винчестер? В состав K3b входят целых три граббера, позволяющих поместить на винчестер содержимое звукового CD и видеодисков (CD и DVD).

Попробуем записать DVD с данными.

ВОЗЬМИТЕ ПЕРЕЗАПИСЫВАЕМЫЙ ДИСК

Настоятельно рекомендую в первый раз взять DVD-RW, а не DVD-R — если при записи вы ошибетесь, DVD-RW можно всегда стереть, а DVD-R — только выбросить.

После выбора действия откроется рабочая область программы (рис. 16.4). В верхней части окна находится файловый менеджер, позволяющий выбрать файлы для записи на DVD. Чтобы записать нужные файлы, просто перетащите их мышью в нижнюю область (рис. 16.5).

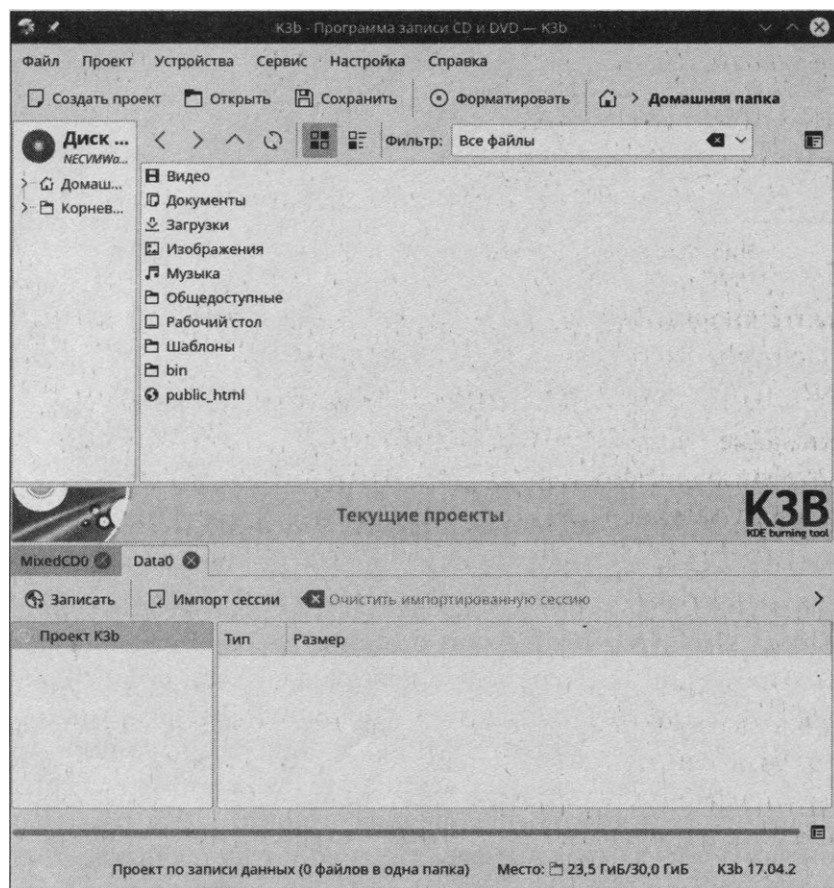


Рис. 16.4. openSUSE: рабочая область K3b

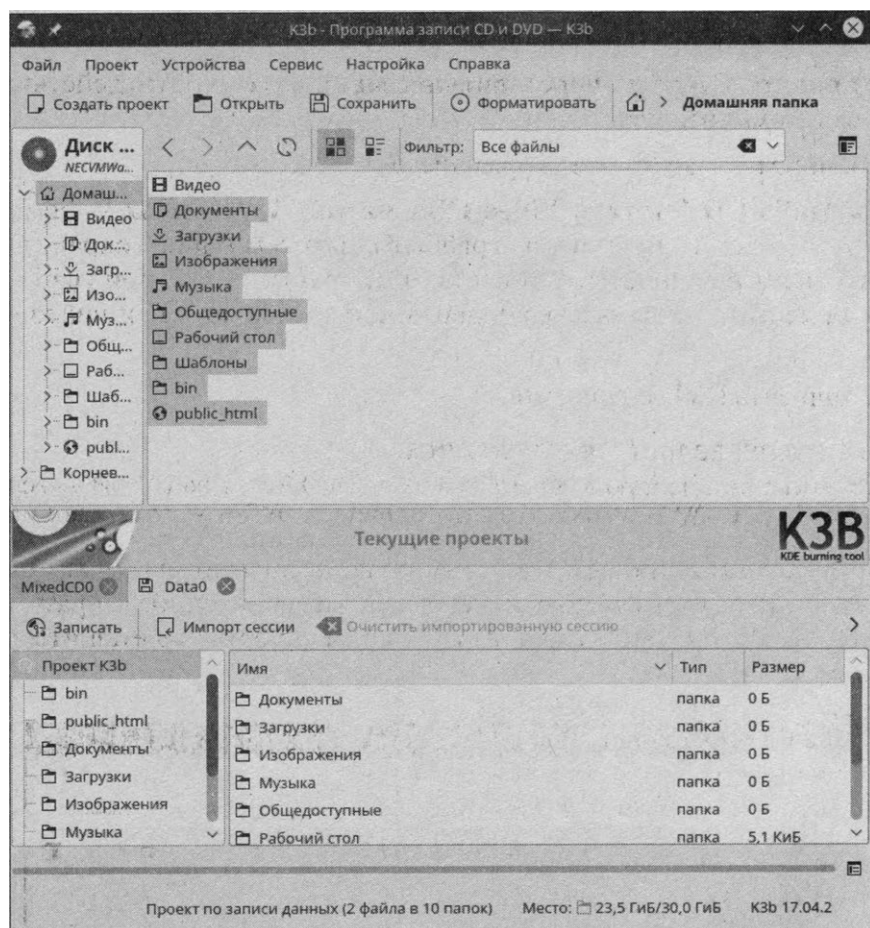


Рис. 16.5. openSUSE: все готово для начала «прожига» диска

Теперь нажмите кнопку **Записать** — откроется окно, позволяющее установить параметры записи (рис. 16.6). Рекомендую выбрать только скорость записи, не полагаясь на значение **Автоматически**.

Обратите внимание на поле **Записать диск** — сейчас в дисковомод находится пустой диск, но если в дисковод будет помещен уже заполненный DVD-RW, программа предложит вам сначала его очистить.

Записанный DVD/CD-RW с *закрытой сессией* (т. е. без возможности дозаписи) можно также очистить с помощью опции **Быстрое форматирование** команды **Сервис | Форматировать** (рис. 16.7).

Вернемся к окну записи диска (см. рис. 16.6). Если вы сейчас не будете записывать диск полностью, но в целях экономии «болванок» планируете в скором времени дозаписать данные на этот диск (я раньше так делал, когда создавал резервные копии данных), перейдите на вкладку **Разное** (рис. 16.8) и в разделе **Режим мультисессии** выберите опцию **Начать мультисессию**. В этом случае у вас потом будет возможность дозаписать данные на этот диск — выберите тогда режим **Продолжить мультисессию**.

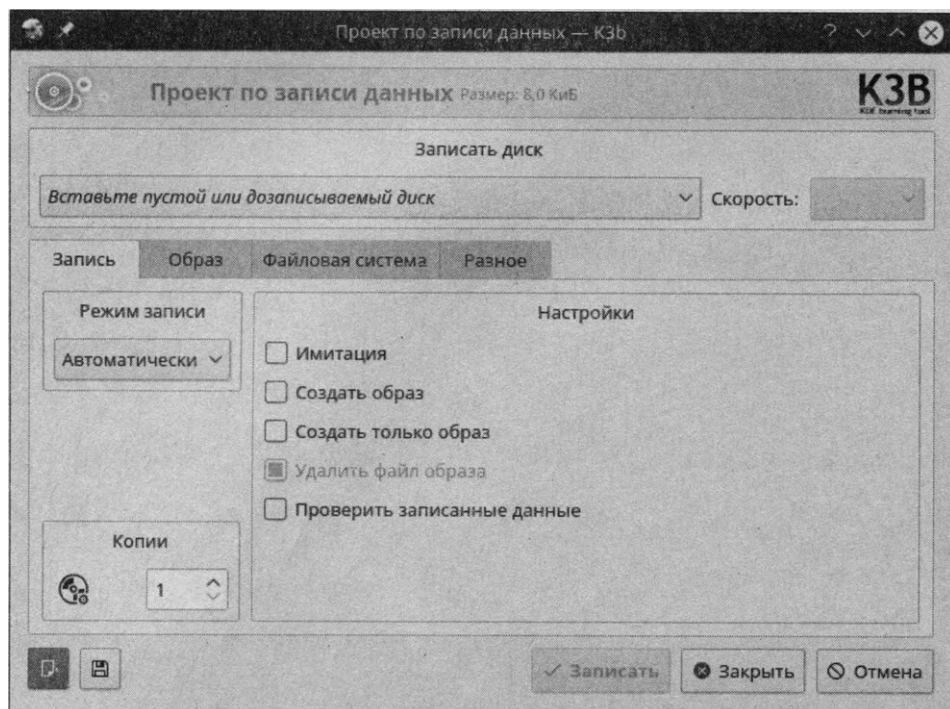


Рис. 16.6. openSUSE: осталось нажать кнопку **Записать**

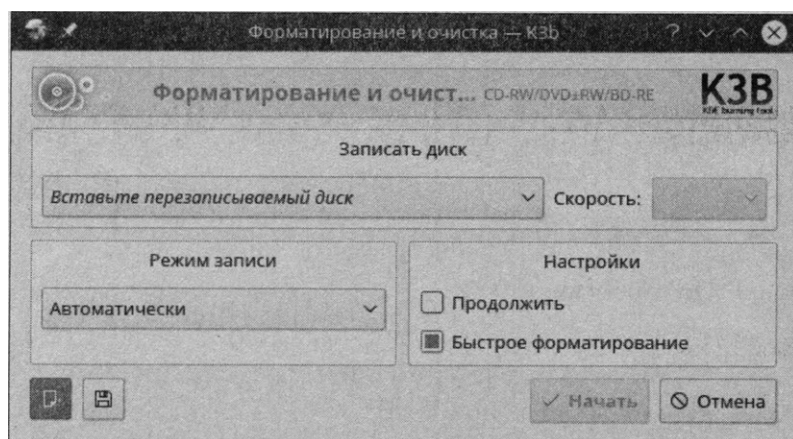


Рис. 16.7. openSUSE: форматирование DVD-RW

На вкладке **Образ** можно задать имя файла и расположение ISO-образа, который создается перед записью диска. По умолчанию образ создается в каталоге `/tmp`, но вы можете изменить папку, например, указав свой домашний каталог (по умолчанию пользователь в Linux может записывать только в свой домашний каталог). Изменение расположения ISO-образа полезно, если на корневом разделе осталось мало дискового пространства, а каталог `/home` монтируется к другому разделу, на котором место есть. Чтобы каждый раз не указывать расположение ISO-образа, нужно указать в настройках программы каталог для временных файлов (см. далее).

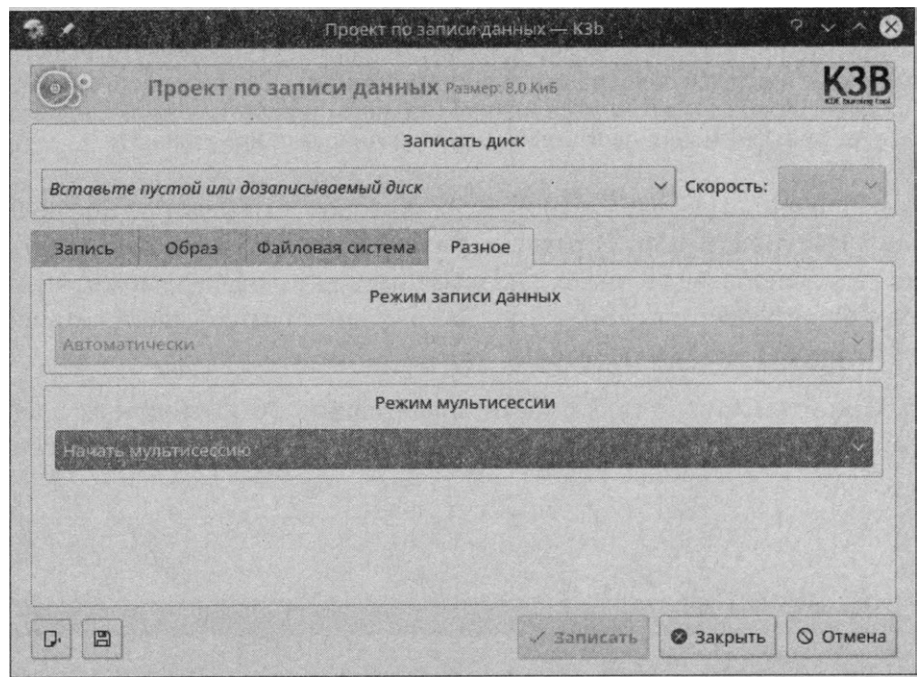


Рис. 16.8. openSUSE: режим многосессионной записи K3b

На вкладке **Файловая система** (рис. 16.9) можно установить метку тома и выбрать тип файловой системы (впрочем, обычно этого делать не требуется). Теперь для записи диска осталось только нажать кнопку **Записать**.

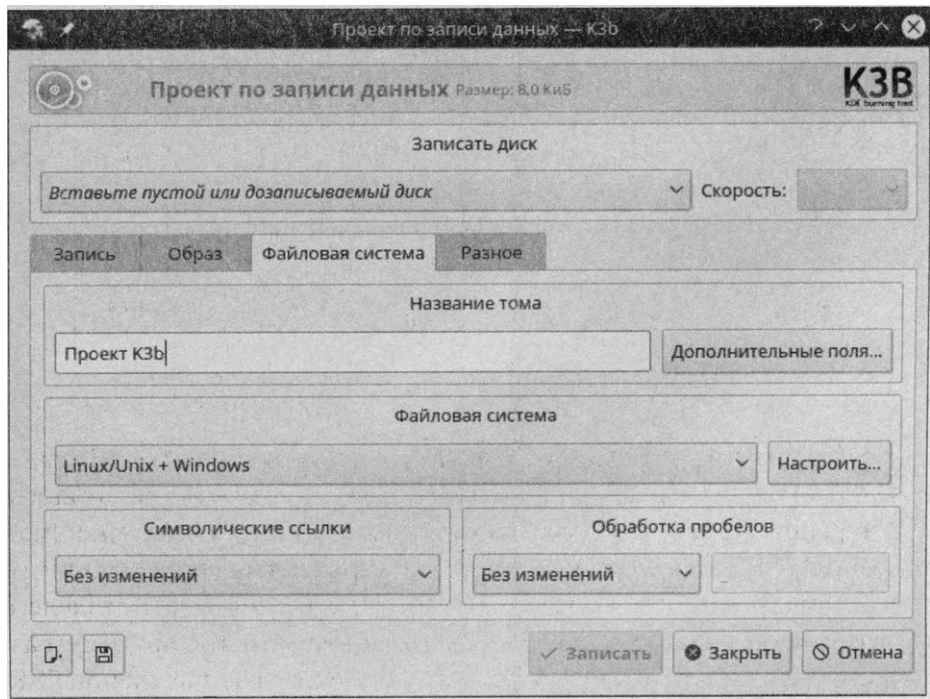


Рис. 16.9. openSUSE: вкладка K3b Файловая система

ЗАКРЫТИЕ СЕССИИ

Помните, что при закрытии сессии дописать информацию на диск DVD/CD-RW уже невозможно — придется сначала его полностью стереть, а если у вас диск DVD/CD-R, то вы вообще не сможете ничего на него записать. Закрытие сессии имеет смысл, если вы записали диск целиком и не планируете его изменять.

Рассмотрим теперь окно параметров K3b, которое вызывается командой меню **Настройка | Настроить K3b**. В разделе **Разное** (рис. 16.10) можно задать каталог для временных файлов — на диске, где находится каталог /tmp, может не быть достаточно свободного места, поэтому иногда приходится задать иной каталог, к которому подмонтирован другой носитель.

В разделе **Приводы** можно посмотреть информацию об имеющихся в вашей системе приводах CD/DVD. Как можно видеть на рис. 16.11, в системе имеется пишу-

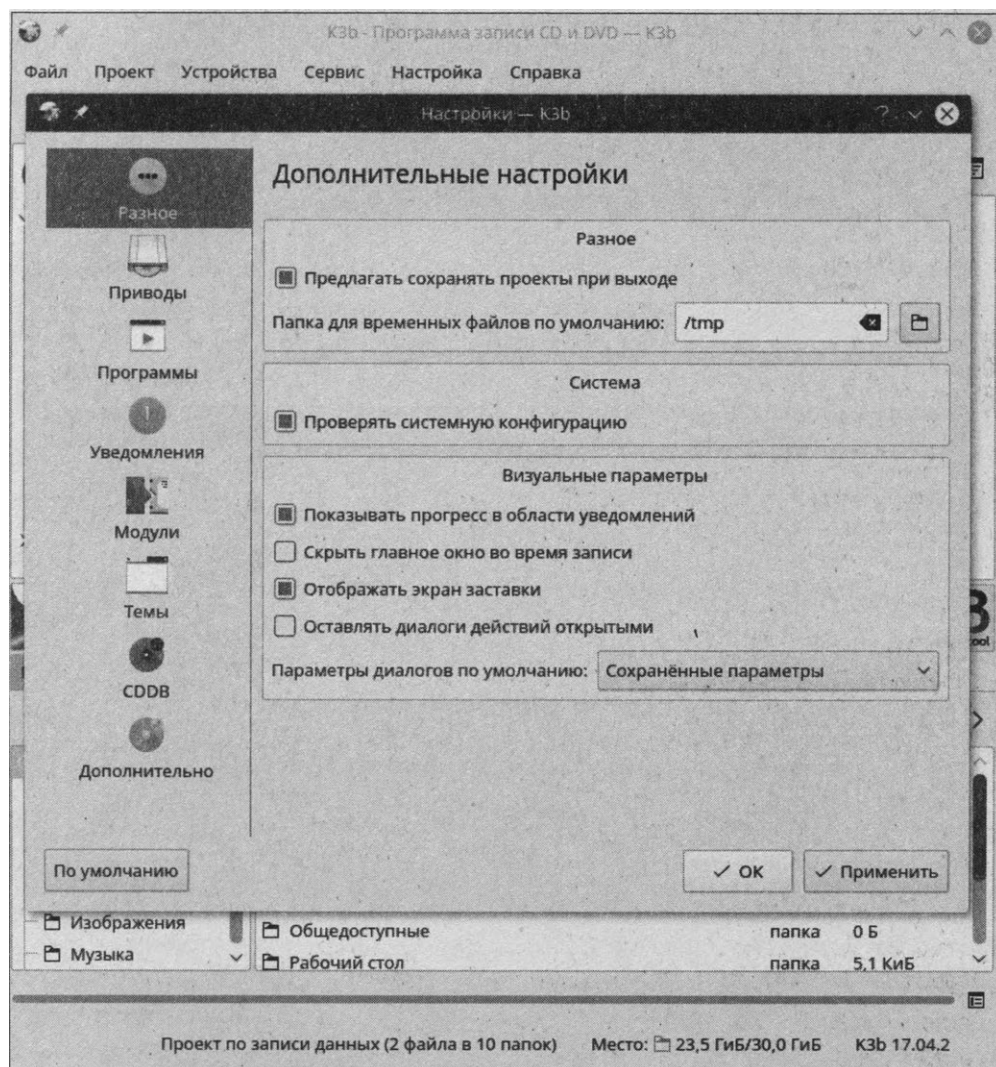


Рис. 16.10. openSUSE: параметры K3b, раздел **Разное**

щий привод VMware. В данном случае это не какой-то там виртуальный привод, а физическое устройство, просто Linux сейчас запущена в виртуальной машине (см. главу 18), и если в свойствах виртуальной машины в качестве CD/DVD-привода выбрать параметр **Физический диск**, то можно производить запись CD/DVD прямо из виртуальной машины.

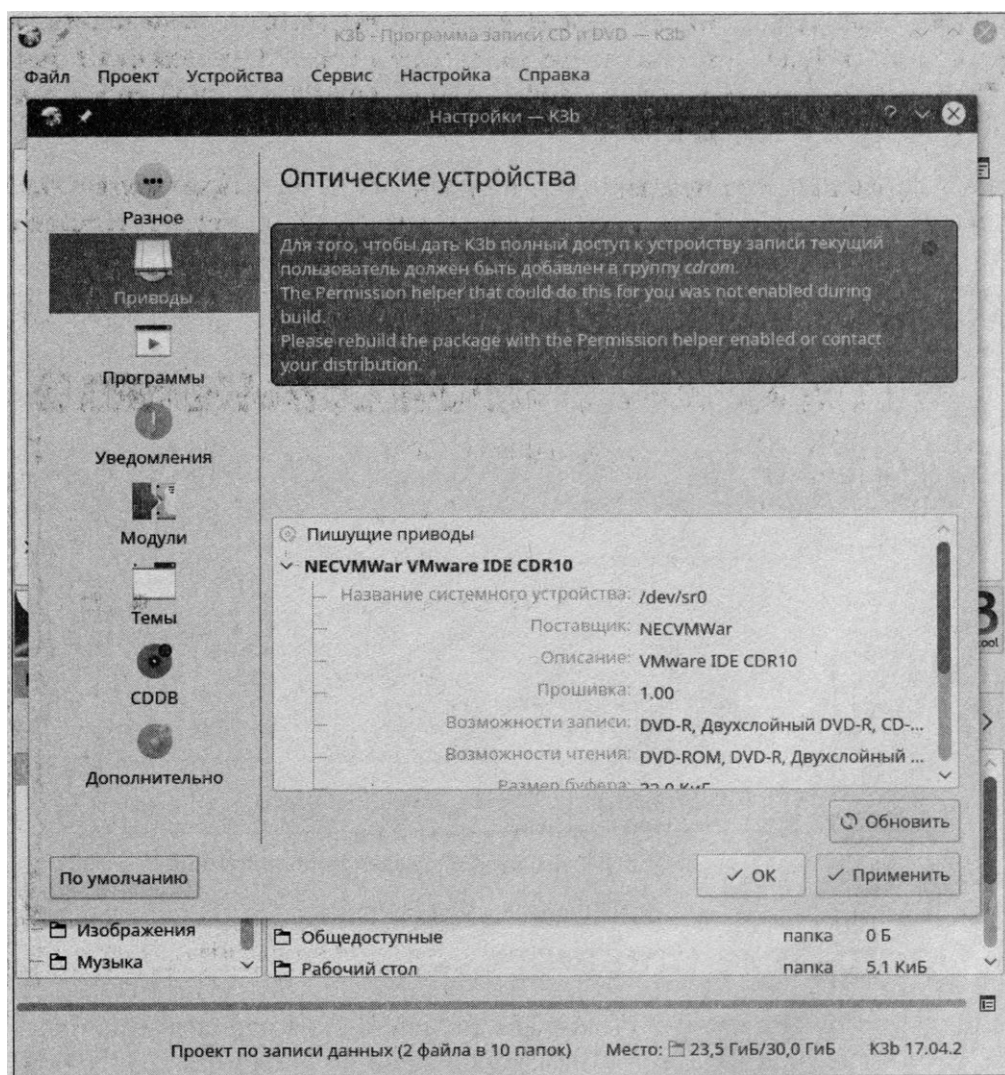


Рис. 16.11. openSUSE: параметры K3b, раздел **Приводы**

В разделе **Дополнительно** (рис. 16.12) можно (если вам это необходимо) включить параметры **Не извлекать диск** после завершения записи и **Автоматически очищать CD-RW и DVD-RW**. Если первый параметр — дело вкуса, то второй довольно опасен — диск будет очищен без предупреждения. А что, если на диске были важные данные?

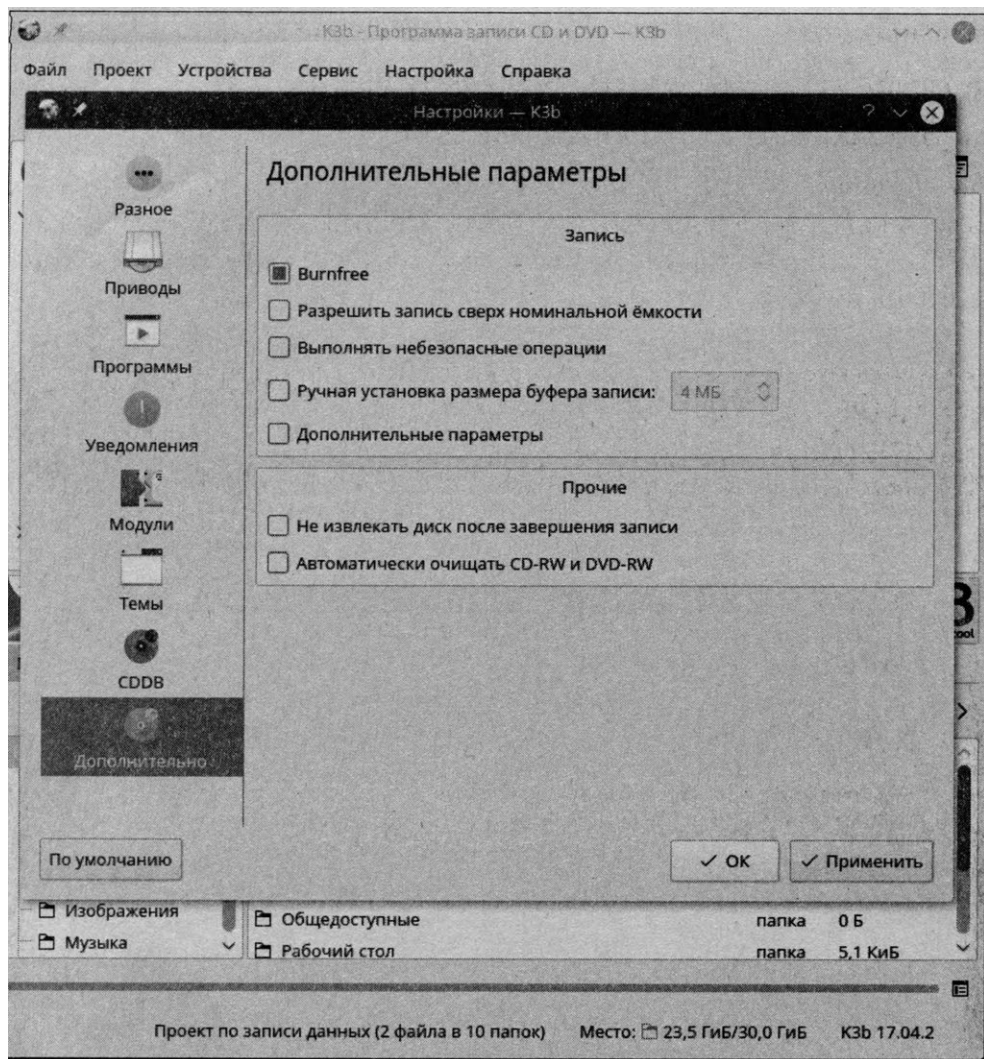


Рис. 16.12. openSUSE: параметры K3b, раздел Дополнительно

16.4. Программа Brasero

В Ubuntu и ряде других дистрибутивов для записи дисков используется программа Brasero. В отличие от K3b, эта программа основана на библиотеках GNOME, и ее лучше не устанавливать, если вы работаете с библиотеками KDE.

УСТАНОВКА ПРОГРАММЫ BRASERO

В Ubuntu 17.04 программа не устанавливается по умолчанию, и для ее установки нужно ввести команду: `sudo apt install brasero`.

Для записи CD/DVD в Brasero запустите программу и укажите тип проекта (рис. 16.13):

- **Звуковой диск** — используется для создания диска формата Audio CD, который можно воспроизвести на компьютере, музыкальном центре и автомагнитоле;

- **Диск с данными** — служит для создания диска с данными;
- **Видеодиск** — позволяет создать DVD Video или SVCD;
- **Копирование диска** — создает копию диска;
- **Записать образ** — делает копию диска, но не записывает ее на CD/DVD, а сохраняет в виде файла-образа на жестком диске. Позже образ можно записать на CD/DVD. Этот тип проекта полезен, если нужно скопировать диск, а чистой «болванки» под рукой нет.

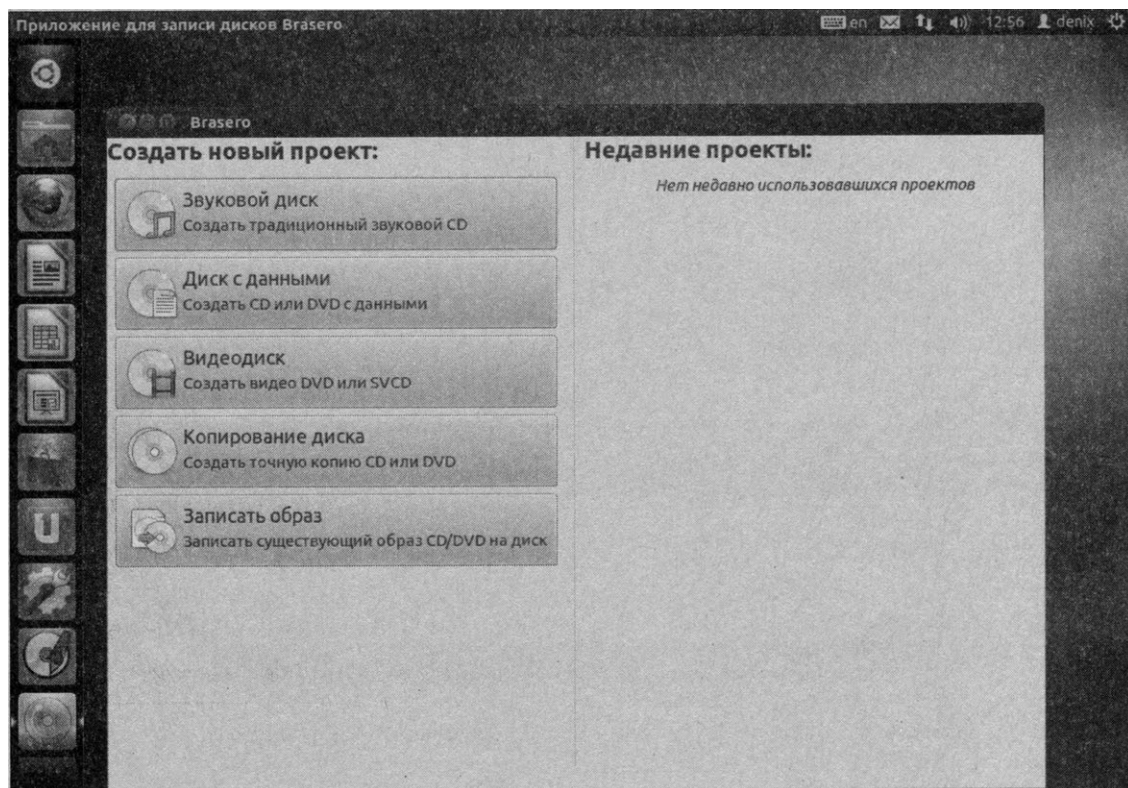


Рис. 16.13. Ubuntu: выбор типа проекта в Brasero

Выберем опцию **Диск с данными**— вы увидите (рис. 16.14) основное окно программы Brasero (оно зависит от типа проекта).

Просто перетаскивайте файлы, которые вы хотите записать на диск, в окно Brasero (можно также нажать кнопку **Добавить** и выбрать необходимые файлы) и нажмите кнопку **Записать**. Кстати, слева от кнопки **Записать** имеется список выбора привода (если у вас их несколько), в этом же списке можно выбрать **Файл образа** для создания ISO-образа, который можно будет потом записать на «болванку».

После нажатия кнопки **Записать** откроется окно (рис. 16.15), в котором придется еще раз нажать кнопку **Записать**. При этом, если вы не вставили диск в привод, Brasero предложит создать ISO-образ записываемого диска.

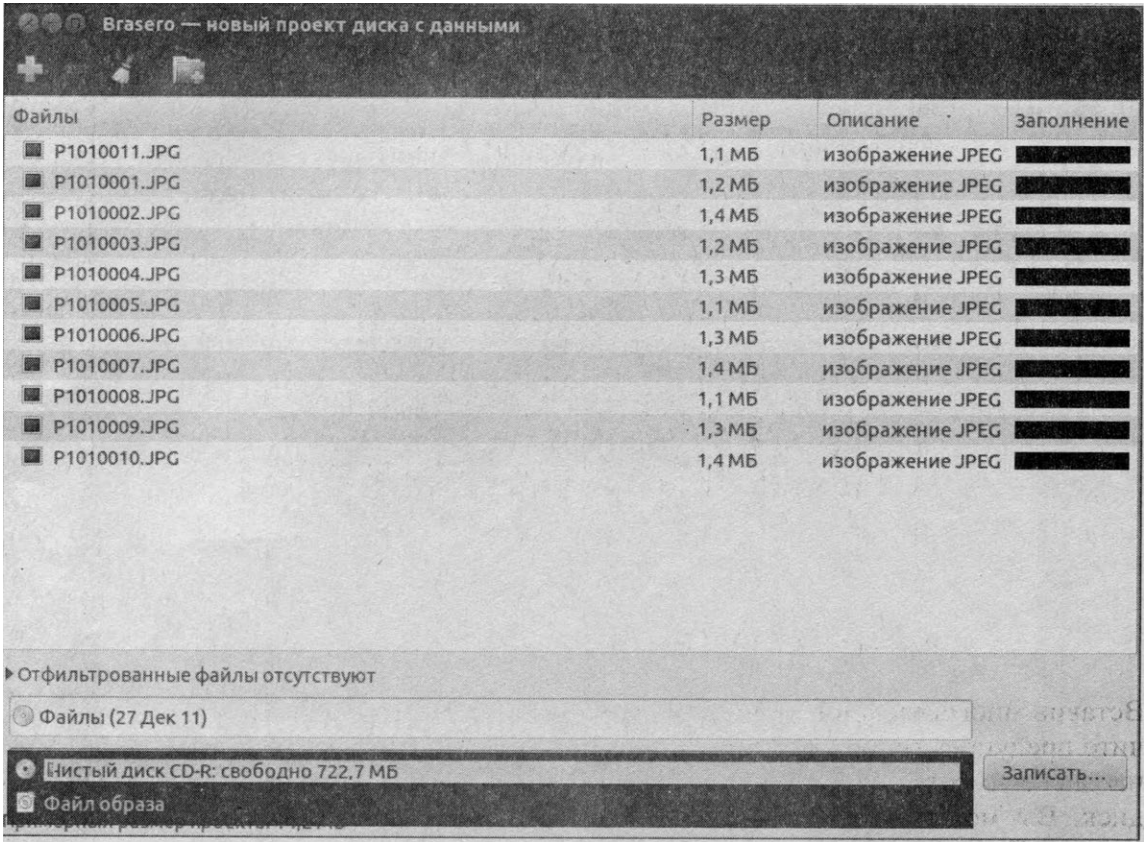


Рис. 16.14. Ubuntu: основное окно Brasero

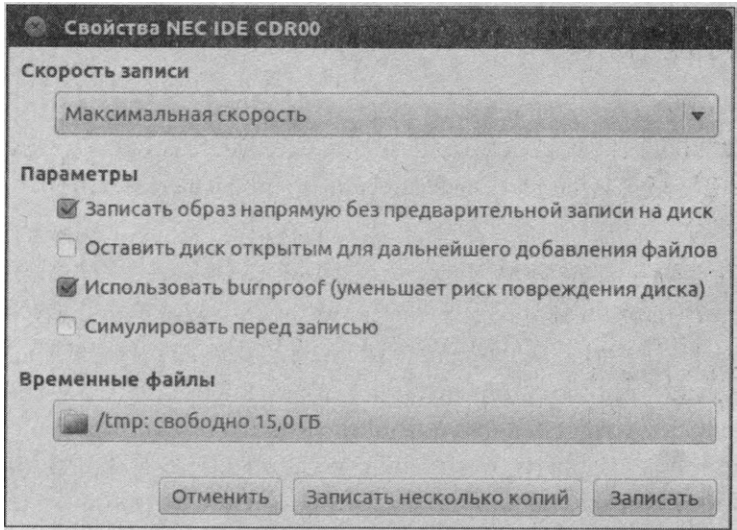


Рис. 16.15. Ubuntu: в Brasero все готово для записи — нажмите кнопку **Записать**

Здесь же можно изменить скорость записи (иногда для повышения качества записи рекомендуется снизить ее скорость), тогда просто выберите ее из списка **Скорость записи** (рис. 16.16).

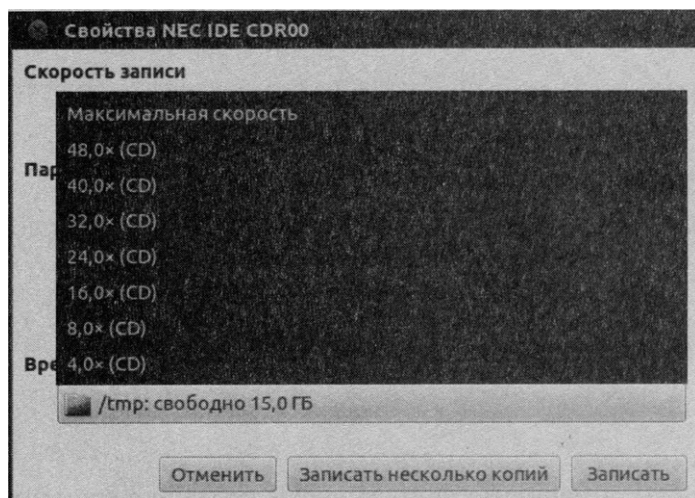


Рис. 16.16. Ubuntu: изменение скорости записи в программе Brasero

Вставив многосессионный диск (на котором уже были записаны файлы), вы получите предупреждение, что ранее записанные файлы не будут доступны. А вот если на диске имеется запись, и сессия закрыта, то Brasero предложит вам очистить диск. Вы можете или отказаться от записи (кнопка **Отмена**), или сменить диск (кнопка **Сменить диск**), или очистить его (кнопка **Очистить диск**). После очистки диска начнется процесс записи, а по окончании записи диск будет извлечен из привода.

16.5. Запись CD/DVD из консоли

Иногда нет возможности запустить графическую программу из-за того, что X-сервер не доступен (произошел сбой, или просто не установлена графическая подсистема X.Org). Тогда можно воспользоваться текстовыми программами записи CD/DVD типа cdburner.

ОСОБЫЕ ОПЕРАЦИИ ПРИ РАБОТЕ С ФАЙЛОВОЙ СИСТЕМОЙ

Подобные ситуации весьма редкие, поэтому их описание вынесено в материал «Особые операции при работе с файловой системой», который вы найдете в папке *Дополнения* сопровождающего книгу электронного архива (см. *приложение*), а также по адресу: <http://www.dkws.org.ua/novice/pdf/fs.pdf>.

16.6. Чтение «битых» компакт-дисков

К сожалению, компакт-диски иногда портятся. Чаще всего причиной становится чисто механическое повреждение— например, царапина. Прочитать все данные с такого диска полностью уже нельзя, но если потеря некоторых данных некритич-

на (например, это диск с фильмом), можно попытаться извлечь оставшуюся информацию.

Сначала нужно создать образ компакт-диска как есть (с пропуском ошибок):

```
# dd if=/dev/cdrom of=~/cd.iso conv=noerror,sync
```

Потом подмонтировать созданный образ к каталогу `/mnt/iso` (если такого каталога не существует, создайте его):

```
mount -o loop ~/cd.iso /mnt/iso
```

Затем скопировать фильм из этого каталога в домашний каталог:

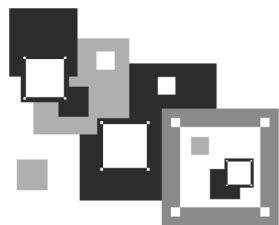
```
cp /mnt/iso/film.avi ~
```

После чего размонтировать и удалить образ:

```
# umount /mnt/iso  
# del ~/cd.iso
```

В результате в вашем домашнем каталоге появится файл с фильмом `film.avi`. Качество фильма иногда (в местах, где были ошибки) будет резко ухудшаться (возможно, пропадет звук или картинка), но это все же лучше, чем полная потеря данных. Заметьте, в Windows средствами самой операционной системы, не прибегая к помощи посторонних программ, мы бы вообще не скопировали этот файл с компакт-диска.

ГЛАВА 17



Популярные программы для работы с Интернетом

17.1. Браузер Firefox

Как пользоваться этим браузером, надеюсь, знают все— по умолчанию им комплектуются многие дистрибутивы, поэтому здесь мы поговорим только об его усовершенствовании.

Как известно, браузер Firefox из-за различных лицензионных «препятствий» не поддерживает Java-апплеты и Flash-ролики. Что касается Java, то его бум уже прошел, — сейчас редко встречаются сайты, разработанные с использованием Java, а вот Flash-ролики присутствуют чуть ли не на каждом втором сайте. Еще полбеды, когда вы не видите шапку сайта или какую-то ее часть, разработанную в виде Flash-ролика, но вот когда целый сайт построен с использованием Flash, то вы там вообще ничего не сможете увидеть, кроме сообщения, что для просмотра этого сайта вам нужно установить Macromedia Flash Player. К сожалению, многие дизайнеры, занимающиеся разработкой Flash-сайтов, напрочь забывают об обычных их HTML-версиях, доступных абсолютно всем пользователям без ограничений. Поэтому, чтобы ничего ни на каких сайтах не упустить, желательно установить в браузере поддержку Flash-роликов.

ЕЩЕ О ТЕХНОЛОГИИ FLASH

В последнее время наблюдается активный отказ от технологии Flash. Возможно, это издание будет последним, в котором объясняется, как установить Flash в Linux.

Если при инсталляции дистрибутива была выбрана установка стороннего программного обеспечения, тогда Flash у вас уже установлен. Чтобы убедиться в этом, откройте браузер и в строке адреса введите:

`about:plugins.`

На рис. 17.1 видно, что в Ubuntu 17.04 плагин Shockwave Flash установлен. Если при установке системы вы забыли отметить соответствующий флажок, то для установки Flash нужно выполнить команду:

```
sudo apt-get install flashplugin-installer
```

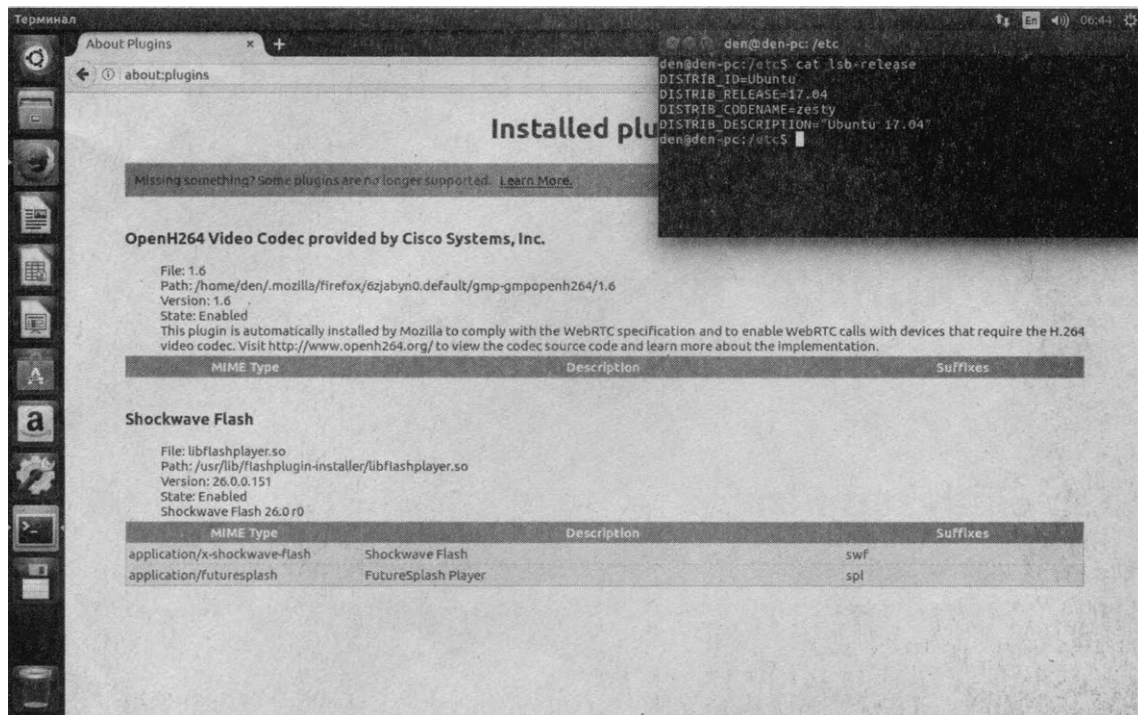


Рис. 17.1. Ubuntu 17.04: плагин Shockwave Flash установлен

Также можно установить и браузер Google Chrome (http://help.ubuntu.ru/wiki/google_chrome), который по умолчанию умеет воспроизводить Flash-контент.

Теперь разберемся, как установить поддержку Flash в Fedora 26. Перейдите по следующей ссылке:

<https://get.adobe.com/flashplayer/>

и скачайте RPM-пакет. Открывать его в программе, как предложит Firefox, не нужно. Просто сохраните его (рис. 17.2), затем перейдите в папку загрузок и щелкните на загруженном плагине двойным щелчком — откроется окно с информацией о плагине (рис. 17.3). Нажмите в нем кнопку **Установить** — перед запуском установки у вас будет запрошен пароль пользователя (установку нужно производить от имени пользователя, включенного в файл `sudoers`, а это, как правило, пользователи, созданные во время установки системы).

После установки плагина перезапустите Firefox и откройте страницу с информацией о плагинах — на ней вы увидите информацию об установленном плагине (рис. 17.4).

Раньше установка Flash Player была гораздо сложнее — приходилось производить множество лишних действий: подключать репозитории, устанавливать определенным образом пакеты, редактировать конфигурацию самого плагина. Сейчас же все гораздо проще — скачали и установили один пакет.

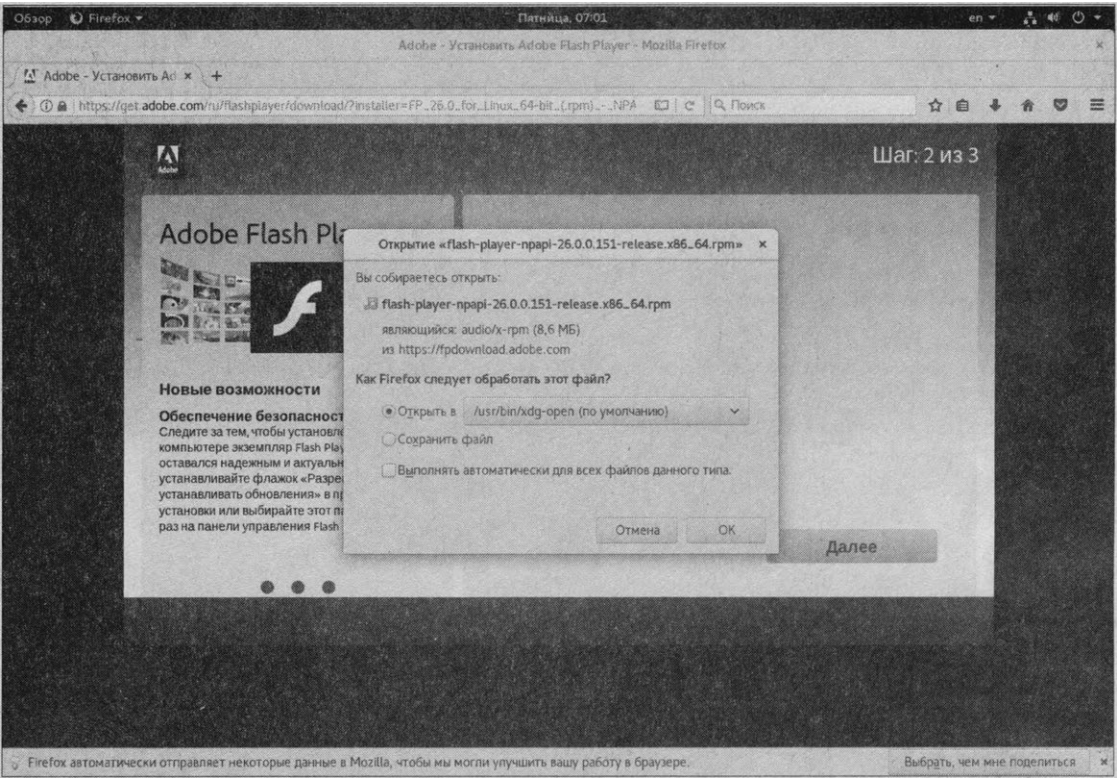


Рис. 17.2. Fedora 26: загрузка плагина Flash Player

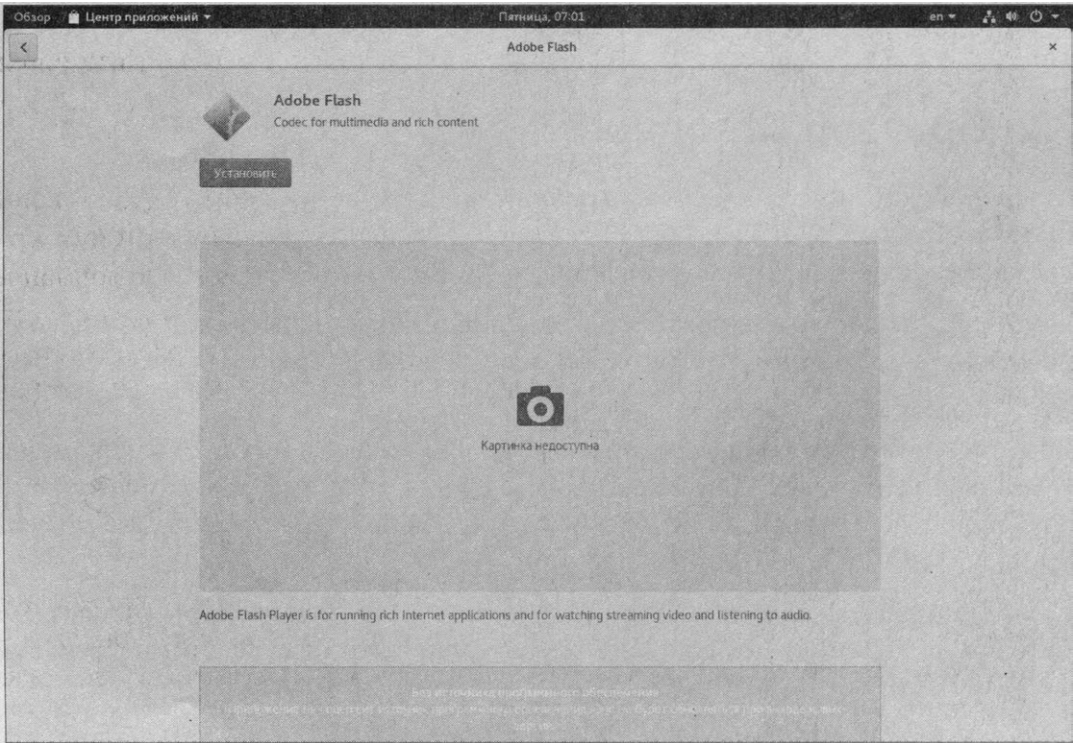


Рис. 17.3. Fedora 26: информация о плагине

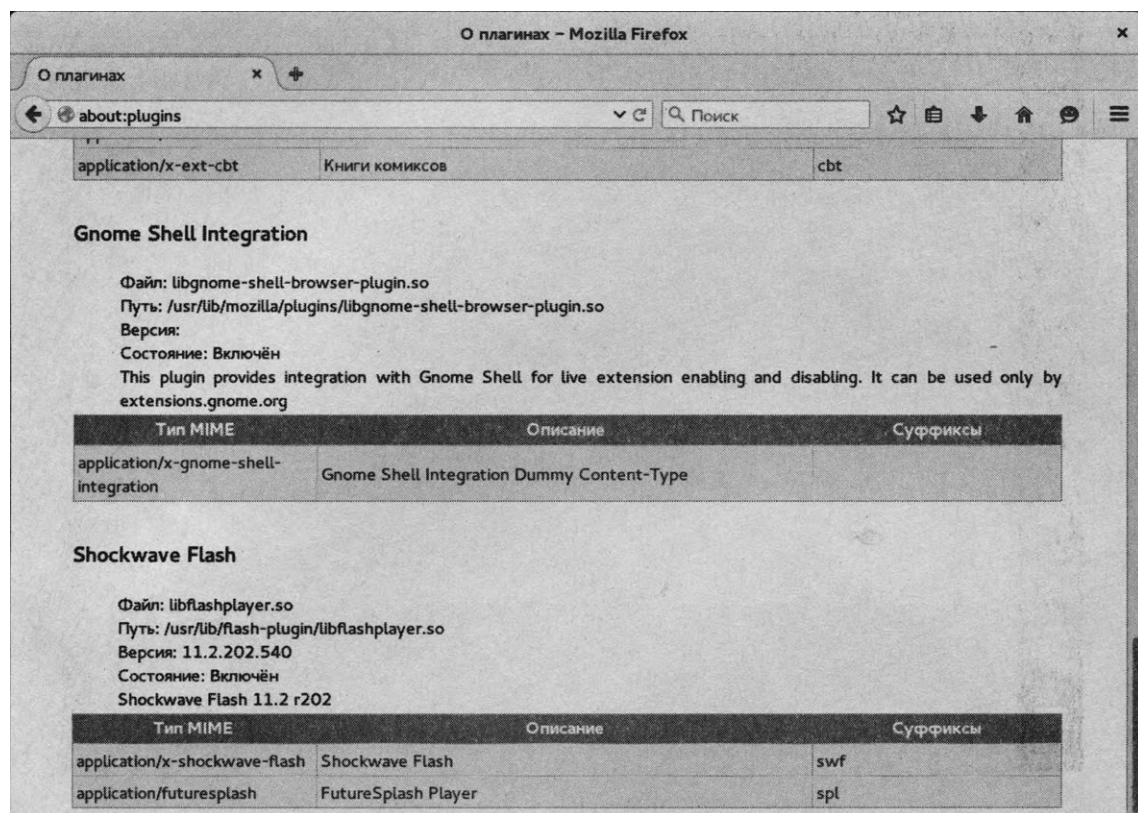


Рис. 17.4. Fedora 26: плагин Shockwave Flash установлен

17.2. Браузер Chromium

Браузер Google Chrome становится все популярнее и популярнее. Так, по данным <http://www.itrew.ru> (2017 год) браузер Google Chrome с огромным отрывом (63%) опережает своих конкурентов. Второе место у Firefox — 14,5%.

Понятно, что Windows-пользователь, попавший в Linux, испытывает определенный дискомфорт — мало того, что интерфейс другой, так и браузер другой. Здесь все другое!

Чтобы пользователям Windows жизнь в Linux казалась проще, я рекомендую им установить браузер Chromium (рис. 17.5)— это тот же самый браузер, что и Chrome, но с открытым исходным кодом: оба браузера похожи, как две капли воды.

Браузер Chromium входит в состав репозитория большинства современных дистрибутивов, поэтому его не составит труда установить стандартными средствами (рис. 17.6).

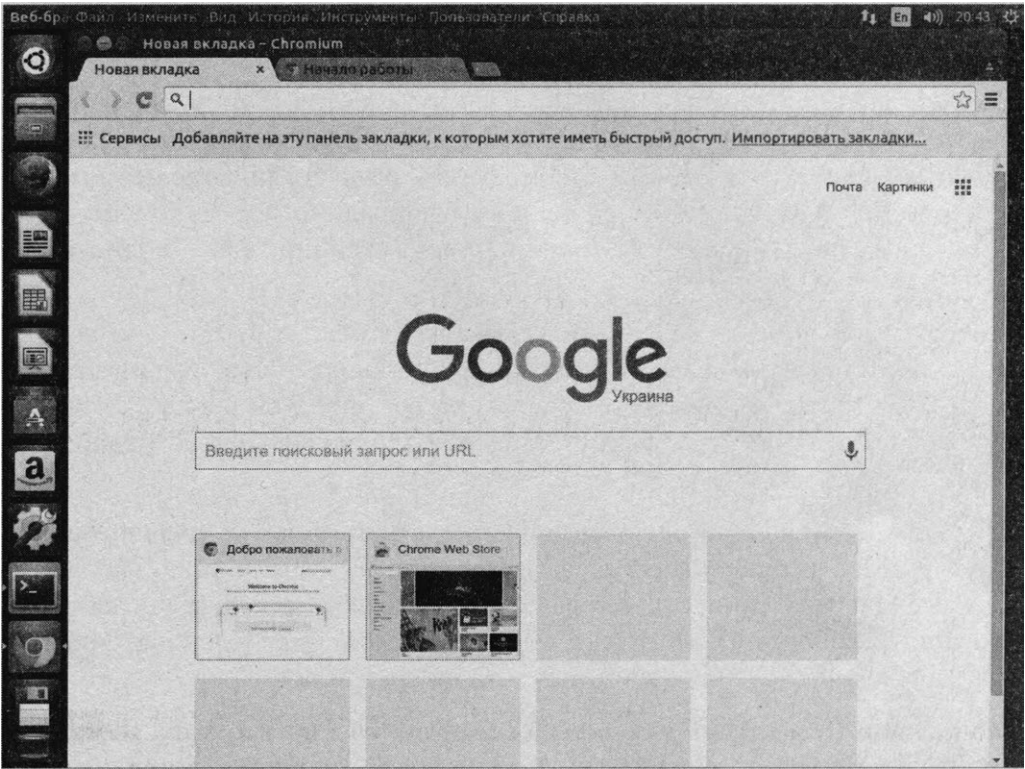


Рис. 17.5. Браузер Chromium

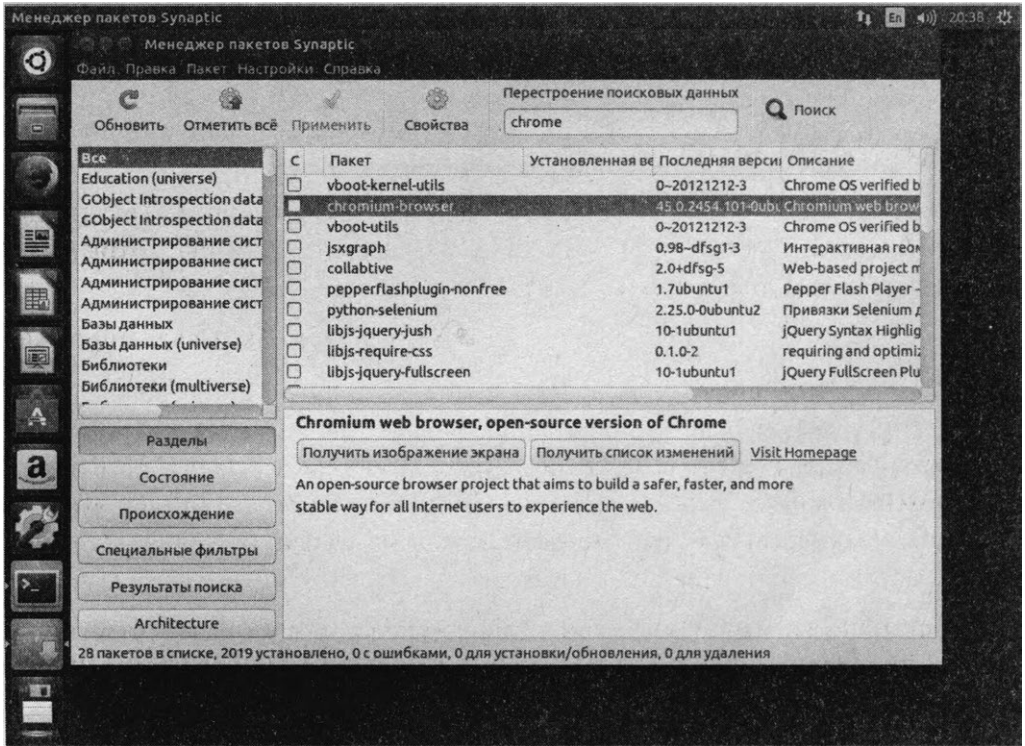


Рис. 17.6. Ubuntu: установка браузера Chromium с помощью менеджера пакетов Synaptic

17.3. Почтовый клиент

Для Linux разработано очень много почтовых клиентов. Ранее выбор почтового клиента навязывала графическая среда: если при установке системы вы выбирали KDE, то устанавливался почтовый клиент KMail, разработанный с использованием тех же библиотек (Qt), что и KDE, а если вы выбирали GNOME, то автоматически устанавливался почтовый клиент Evolution, использующий те же библиотеки, что и GNOME (GTK).

Теоретически можно было в GNOME установить KMail и в KDE— Evolution, но практически никто так не делал, поскольку при установке «чужого» почтового клиента «тянулись» тяжелые библиотеки «чужой» графической среды, которые занимали много места на диске и, фактически, использовались только одной программой.

Сейчас же практически во всех современных дистрибутивах устанавливается удобный почтовый клиент Mozilla Thunderbird, чем-то напоминающий The Bat! Такое решение весьма оправданно: во-первых, пользователям не приходится загружать лишние библиотеки, а, во-вторых, обеспечивается определенная унификация, облегчающая переход с одного дистрибутива на другой.

В использовании Thunderbird нет никаких секретов: вы просто запускаете программу, вводите параметры доступа к почтовому ящику (имя пользователя, имя сервера, пароль) и работаете со своей электронной почтой. Поэтому подробно рассматривать Thunderbird в этой книге мы не станем— не думаю я, что пользователь, сумевший установить и настроить Linux, не разберется со столь простым почтовым клиентом.

А если вы ранее использовали Evolution или KMail и успели к ним привыкнуть, то можете их установить самостоятельно, — они никуда не делись, и все еще присутствуют в репозиториях дистрибутивов.

17.4. Skype

В предыдущих изданиях этой книги в качестве клиента для обмена мгновенными сообщениями рассматривался Pidgin, который в основном использовался как ICQ-клиент. Вот только незадача— популярные ранее сервисы мгновенного обмена сообщениями, в том числе и ICQ, больше уже не пользуются популярностью у пользователей, — все переключались на более современные приложения вроде Skype, Viber и пр. Оно и понятно — здесь можно не только обмениваться мгновенными сообщениями, но и совершать видеозвонки. И все это тоже бесплатно.

Нужно отметить, что установка Skype в Linux более чем проста, — достаточно зайти на **www.skype.com** и скачать версию для своего дистрибутива. Несмотря на то, что в списке на сайте самой старшей Ubuntu оказалась только версия 12.04 (рис. 17.7), загруженный DEB-пакет нормально установился в Ubuntu 17.04— последней версии на момент написания этих строк.

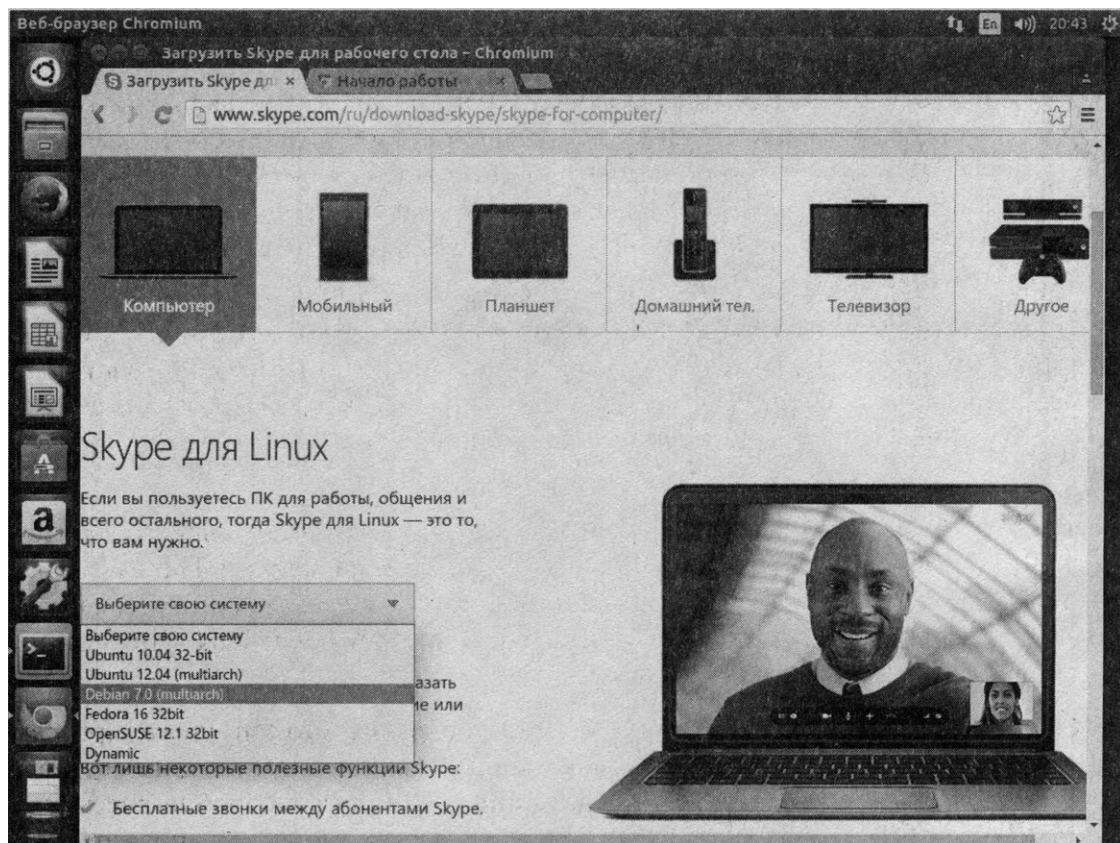


Рис. 17.7. Ubuntu: загрузка Skype

Интересно, что при загрузке DEB-пакета с сайта Skype браузер Chromium определяет его как вредоносный (рис. 17.8). Отчасти могу с ним согласиться, учитывая, сколько памяти потребляет современная версия Skype.

Далее все просто: перейдите в папку загрузок и щелкните на пакете двойным щелчком — откроется окно, в котором нужно нажать кнопку **Установить** (рис. 17.9). Установив пакет, можете запустить Skype (рис. 17.10). Вот оно и свершилось — Microsoft добралась до Linux ☺.

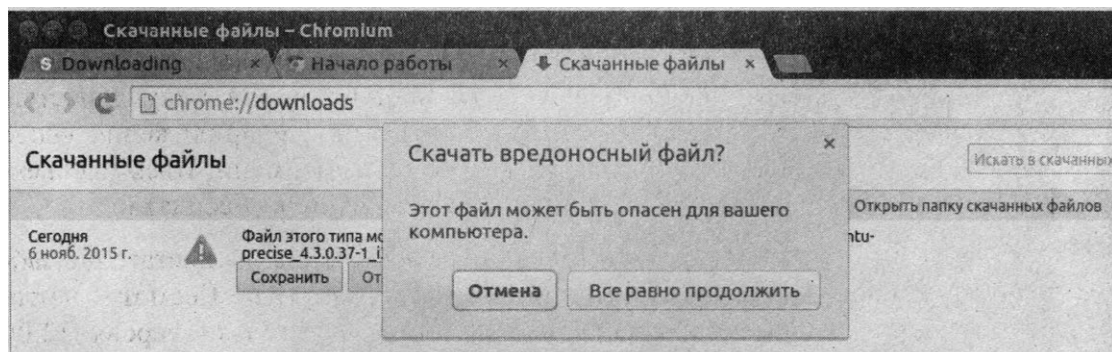


Рис. 17.8. Ubuntu: браузер определил пакет со Skype как вредоносный — игнорируйте это предупреждение

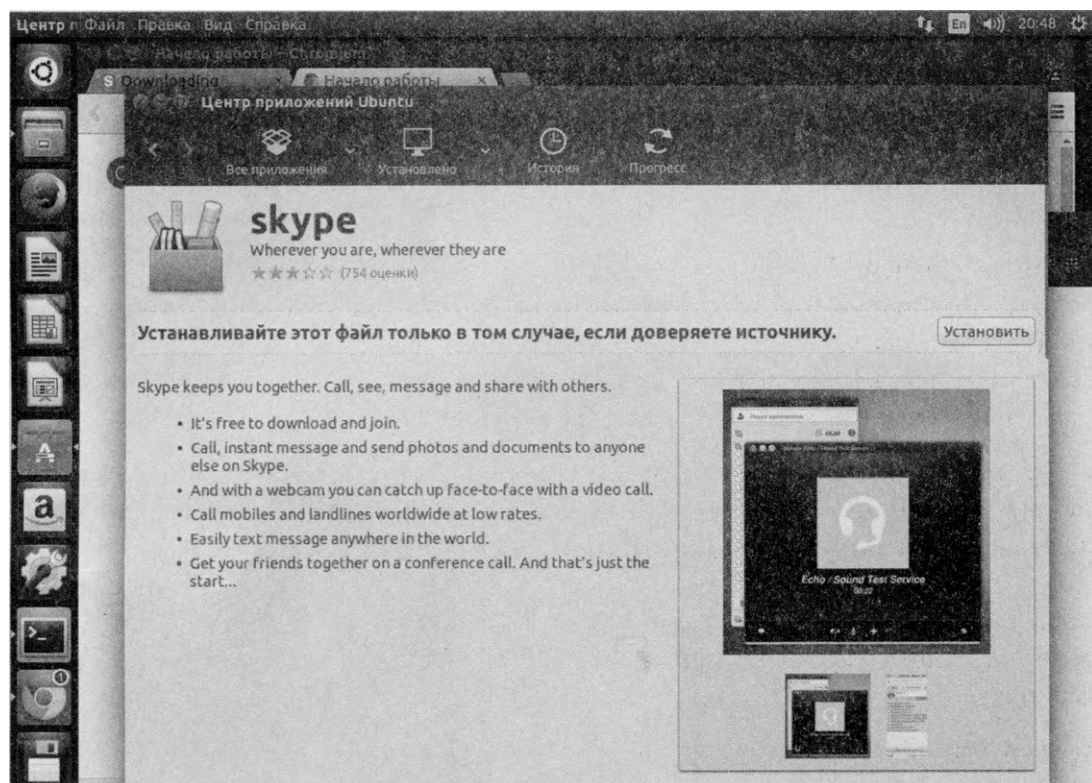


Рис. 17.9. Ubuntu: установка Skype

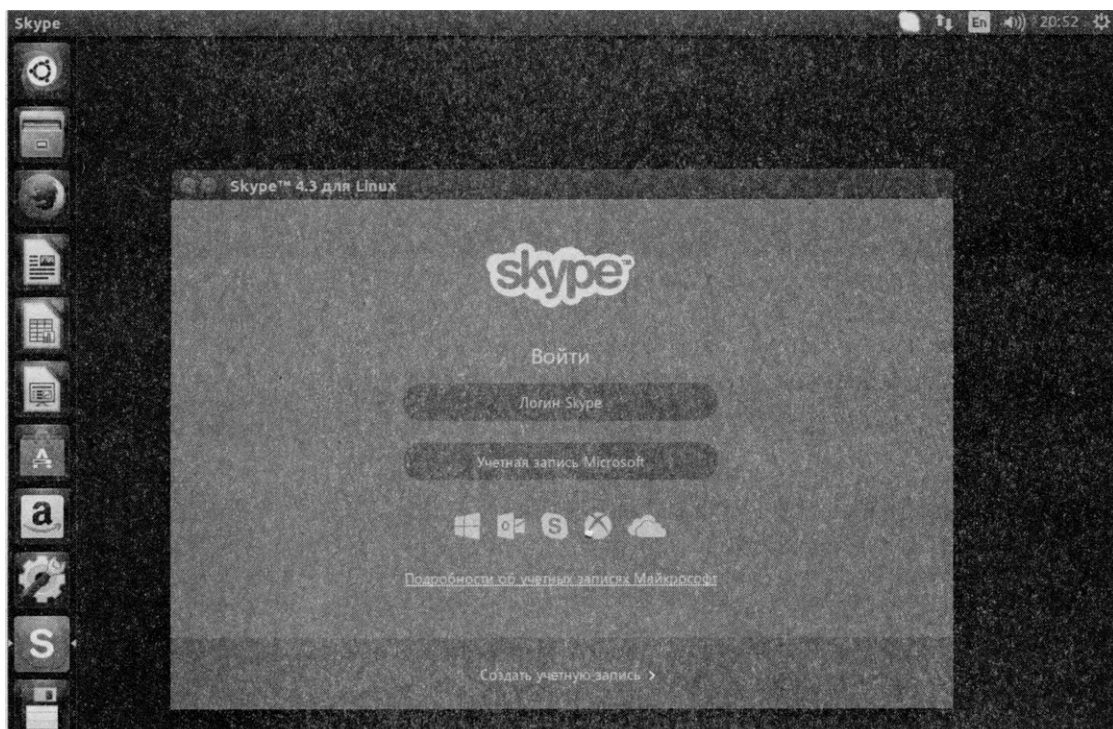


Рис. 17.10. Ubuntu: Skype запущен

17.5. FTP-клиенты

Для Linux разработано очень много FTP-клиентов. Кроме того, работу с FTP поддерживают практически все браузеры Linux. Впрочем, FTP-возможности браузеров ограничены и не дотягивают до возможностей даже самого простого FTP-клиента.

Основной задачей FTP-клиента является обмен файлами с FTP-сервером (с помощью FTP-клиента можно не только скачать файл, но и закачать его на сервер). Стандартным для многих операционных систем является простенький текстовый клиент `ftp`. Зная, как работать с этим клиентом, вы в любой операционной системе будете чувствовать себя «в своей тарелке».

Нужно отметить, что в некоторых современных дистрибутивах — например, в Fedora 26, команда `ftp` недоступна, поскольку пакет `ftp` по умолчанию в них не установлен. Для установки этого пакета в Fedora надо ввести команду:

```
sudo dnf install ftp
```

После установки пакета для открытия соединения с любым FTP-сервером введите команду:

```
ftp <имя или адрес FTP-сервера>
```

Можно также просто ввести команду `ftp`, а в ответ на приглашение:

```
ftp>
```

ввести команду:

```
open <имя или адрес FTP-сервера>
```

Лично мне больше нравится первый вариант, поскольку он позволяет сэкономить время.

В процессе подключения к серверу вам будет предоставлена возможность ввести имя пользователя и пароль (рис. 17.11).

Подключившись к серверу, вы можете ввести команду `help`, чтобы просмотреть список доступных команд. Для получения справки по той или иной команде введите `help <имя_команды>`. Наиболее популярные команды FTP-клиента приведены в табл. 17.1.

Из графических FTP-клиентов самым лучшим, на мой взгляд, FTP-клиентом является FileZilla, версии которого существуют как для Linux (рис. 17.12), так и для Windows. Linux-версия FileZilla входит в состав многих современных дистрибутивов — для ее установки нужно установить пакет `filezilla`.


```

den@den:~$
Файл Правка Вид Поиск Терминал Справка
Выполнение транзакции
  Установка      : ftp-0.17-67.fc22.i686      1/1
  Проверка       : ftp-0.17-67.fc22.i686      1/1

Установлено:
  ftp.i686 0.17-67.fc22

Выполнено!
[den@den ~]$ ftp ftp.dkws.org.ua
Connected to ftp.dkws.org.ua (91.203.4.50).
220 ProFTPD 1.3.5 Server ready.
Name (ftp.dkws.org.ua:den): dkws
331 Необходим пароль для пользователя dkws
Password:
230 Пользователь dkws подключён
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> put dmesg.txt
local: dmesg.txt remote: dmesg.txt
227 Entering Passive Mode (91,203,4,50,140,77).
150 Открываю режим BINARY данных для dmesg.txt
226 Передача завершена
101560 bytes sent in 0,0816 secs (1244,23 Kbytes/sec)
ftp>

```

Рис. 17.11. Fedora: команда ftp

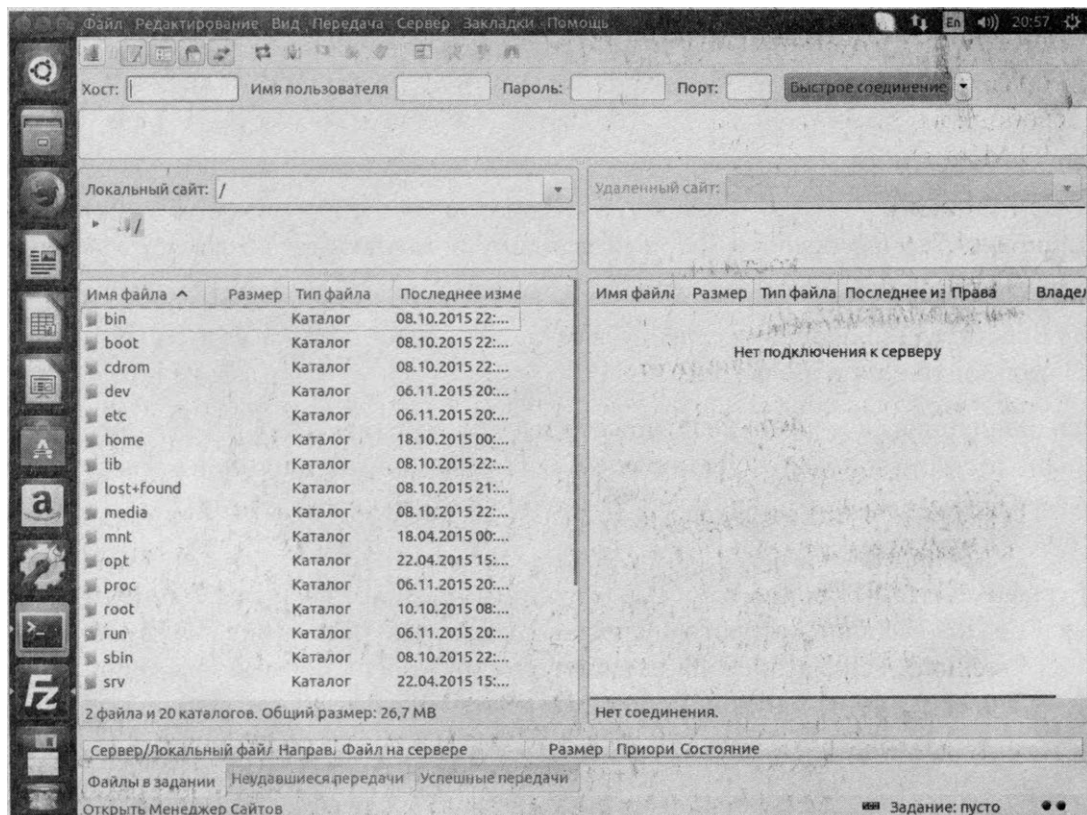


Рис. 17.12. Ubuntu: FTP-клиент FileZilla

Таблица 17.1. Некоторые команды FTP-клиента

Команда	Описание
ls	Вывод содержимого каталога
get	Загрузить файл с сервера
put	Загрузить файл на сервер
mget	Получить несколько файлов с сервера. Допускается использование масок файлов — например: *.rpm
mput	Загрузить несколько файлов на сервер
cd	Изменить каталог
mkdir	Создать каталог
rmdir	Удалить пустой каталог
delete	Удалить файл

17.6. P2P-клиенты

В январе 1999 года Шон Фэннинг, восемнадцатилетний студент одного из американских вузов, написал программу, позволяющую обмениваться МРЗ-файлами по Интернету. Программа была создана, так сказать, для внутреннего пользования — Шон делал ее для себя и своих друзей. Автор программы даже и не подозревал, что совершит настоящий прорыв в компьютерных технологиях.

Что же сделал Шон Фэннинг? Им была создана так называемая *пиринговая сеть* (Peer-to-peer, peer2peer, P2P). Не нужно путать ее с обычной одноранговой сетью, которая существовала с самого начала развития сетевых технологий. Но у пиринговой сети есть и кое-что общее с обычной одноранговой сетью — каждый участник такой сети может выступать и как клиент, и как сервер. Все мы привыкли, что есть сервер, с которого скачиваются файлы. А здесь каждый пользователь может предоставить доступ к собственным файлам (не ко всем, а только к избранным), и все остальные участники пиринговой сети смогут скачать эти файлы. Ясно, что этот пользователь должен находиться в сети, иначе файлы скачать будет невозможно.

Существуют две модели пиринговых сетей: централизованные и децентрализованные. Первая пиринговая сеть была как раз централизованной. В этом случае файлы хранятся на компьютерах пользователей, но их поиск (как и регистрация новых пользователей сети) осуществляется через центральный сервер. Понятно, что если его прикрыть, то вся пиринговая сеть будет разрушена. Поэтому следующий виток в развитии пиринговых сетей — это децентрализованные сети. Здесь нет выделенного сервера, и нейтрализация одного из компьютеров сети никак не скажется на функционировании всей сети в целом. Теоретически, чтобы закрыть такую сеть, нужно нейтрализовать все ее компьютеры. Впрочем, «чистые» децентрализованные сети встречаются редко. Намного чаще появляются пиринговые сети, представ-

ляющие собой нечто среднее между централизованной и децентрализованной моделями.

В пиринговых сетях можно найти очень много интересной информации: музыку, видео, ключи к программам (я вам этого не говорил), а также сами программы. Конечно, ни сами программы, ни ключи к ним Linux-пользователям не нужны, поскольку в большинстве случаев в пиринговых сетях выложены Windows-программы, а программы для Linux и так можно вполне официально бесплатно скачать и установить. Так что, в основном вы будете использовать такие сети для закачки музыки и видео (если ваше интернет-соединение это позволяет).

Наиболее популярным пиринговым протоколом является BitTorrent. Для работы с ним во всех дистрибутивах Linux используется программа Transmission (рис. 17.13), которая устанавливается по умолчанию. Именно с помощью этой программы можно качать с «торрентов» музыку, фильмы и видео.

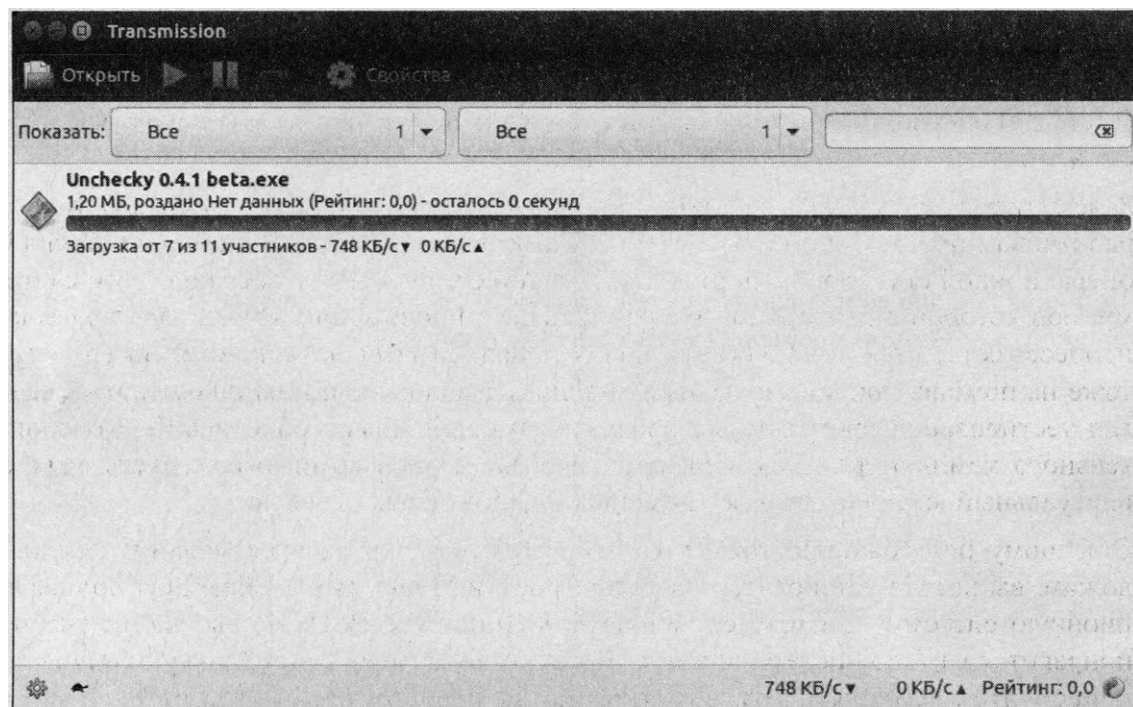


Рис. 17.13. Ubuntu: программа Transmission



Виртуальная машина VirtualBox

18.1. Зачем нужна виртуальная машина?

Наверняка многие из вас знакомы с *виртуальной машиной* VMware. Прелесть виртуальной машины заключается в том, что вы можете установить в ней любую операционную систему, работая при этом в основной операционной системе. Другая (гостевая) операционная система будет запущена в отдельном окне эмулятора, и вы сможете работать с ней в обычном режиме.

Зачем нужна виртуальная машина? По большому счету она нужна в основном разработчикам программного обеспечения. Так они могут работать в своей привычной операционной системе, а в виртуальной машине запускать ту операционную систему, под которой они хотят протестировать свое приложение. Иногда для проверки процесса сетевого взаимодействия может понадобиться еще один компьютер — тут тоже на помощь придет виртуальная машина. Налицо экономия и комфорт — ведь для тестирования программных продуктов и сетей можно обойтись без дополнительного компьютера. Да и переключение на «другой компьютер», пусть даже и виртуальный, осуществляется с помощью одного щелчка мышью.

Обычному пользователю тоже может пригодиться виртуальная машина. Предположим, вы хотите установить новый дистрибутив Linux или вообще другую операционную систему — например, Windows 8/10 или FreeBSD. Но вы еще не знаете, понравится вам эта система или нет. Тогда установите ее в виртуальную машину и попробуйте с ней поработать. Заметьте, вам не придется изменять разметку диска и размер разделов, а также создавать новые разделы, чтобы установить вторую операционную систему. При использовании виртуальной машины на вашем жестком диске будет создан файл *образа* жесткого диска, служащий в качестве жесткого диска виртуального компьютера. Если установленная операционная система вам не понравится, смело удаляйте файл образа — снова изменять разметку жесткого диска и рисковать работоспособностью своего компьютера в случае, если что-то при установке пойдет не так, не понадобится.

Честно говоря, каждый новый дистрибутив Linux я сначала устанавливаю в виртуальную машину, а только затем на физический компьютер. Во-первых, я вижу, какие проблемы могут возникнуть при установке, и если они возникнут в вирту-

альной машине, ничего страшного не случится, сами понимаете. Во-вторых, я могу сделать снимки экрана (скриншоты) процесса установки операционной системы, чтобы потом показать их друзьям.

Конечно, для работы эмулятора нужен соответствующий компьютер — понадобится как минимум 1 Гбайт оперативной памяти: 512 Мбайт останется для основной операционной системы, а 512 Мбайт будет отдано виртуальной машине. Конечно, это самый минимум. Для реальной работы с виртуальной машиной, а не только для ее запуска, понадобится как минимум 2 Гбайт оперативной памяти: один — для основной ОС, один — для виртуальной. Опять-таки многое зависит еще и от гостевой ОС. Одно дело запускать в виртуальной машине Linux, которая нетребовательная к ресурсам, совсем другое, если вы надумаете запустить, скажем, Windows 10. Если у вашего компьютера мало оперативной памяти, а нужно запускать в виртуальной машине Windows, в качестве гостевой ОС выбирайте 32-разрядные ее сборки — они гораздо менее требовательны к ресурсам.

Место на жестком диске также зависит от устанавливаемой ОС, и пара десятков свободных гигабайтов у вас должна быть.

Во время работы виртуальной машины производительность основной операционной системы, понятно, понизится. Гостевая ОС будет также работать медленнее, чем на реальном компьютере, но все-таки она будет работать!

«НАСТОЛЬНАЯ» ВИРТУАЛИЗАЦИЯ

В этой главе рассматривается решение так называемой «настольной» виртуализации, предназначенное для использования конечным пользователем. В *главе 38* будет рассмотрено решение виртуализации для сервера — OpenVZ.

18.2. Установка эмулятора VirtualBox

Для установки эмулятора виртуальной машины VirtualBox нужно установить пакет `virtualbox`, а все остальные пакеты менеджер пакетов установит автоматически. Если пакет `virtualbox` не входит в состав дистрибутива, и его нет в репозиториях дистрибутива, вы всегда сможете его скачать с сайта разработчиков <http://www.virtualbox.org/>. Программа абсолютно бесплатная и распространяется по лицензии GPL.

После установки VirtualBox перезагрузите компьютер, чтобы был загружен модуль ядра `vboxdrv`, или введите от имени `root` команду:

```
modprobe vboxdrv
```

Затем добавьте в группу `vboxusers` всех пользователей, которым разрешено использовать VirtualBox.

Запустите эмулятор с помощью соответствующей команды меню GNOME/KDE или выполните команду:

```
/usr/bin/VirtualBox
```

18.3. Создание новой виртуальной машины

В открывшемся окне менеджера виртуальных машин (рис. 18.1) нажмите кнопку **Создать** (New)— вы увидите окно мастера создания новой виртуальной машины. Первый шаг здесь сугубо информационный, поэтому просто нажмите кнопку **Далее** (Next), выберите гостевую операционную систему (ту, которую хотите установить) и введите название для новой виртуальной машины (рис. 18.2).

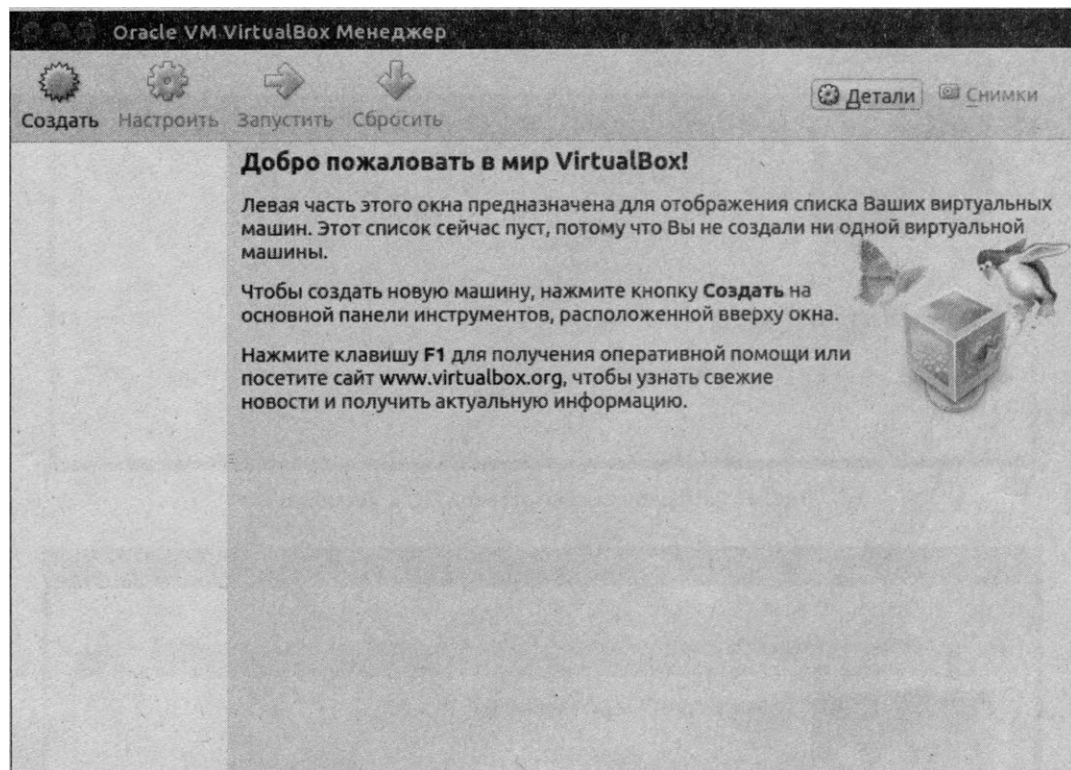


Рис. 18.1. Ubuntu: менеджер виртуальных машин VirtualBox

Нажатие кнопки **Далее** (Next) откроет следующее окно VirtualBox, где нужно установить для этой виртуальной машины размер оперативной памяти. Обычно VirtualBox самостоятельно определяет рекомендуемый размер ОЗУ, исходя из выбранной операционной системы и объема физической оперативной памяти. В моем случае VirtualBox порекомендовал 512 Мбайт. Я выбрал Windows 7 (в последней на момент написания этих строк версии VirtualBox уже присутствует также поддержка Windows 10), а для нее минимальный размер ОЗУ равен 512 Мбайт, т. е. меньше — нельзя. Поскольку у моего компьютера всего 1 Гбайт оперативной памяти, то больше тоже нельзя, иначе ничего не останется для реальной операционной системы. Отсюда рекомендуемый размер оперативной памяти — 512 Мбайт (рис. 18.3).

Следующий шаг— это выбор файла образа жесткого диска (рис. 18.4). Поскольку мы ранее не создавали такие файлы, то выберите опцию **Создать новый виртуальный жесткий диск** (Create new hard disk). В дальнейшем для использования

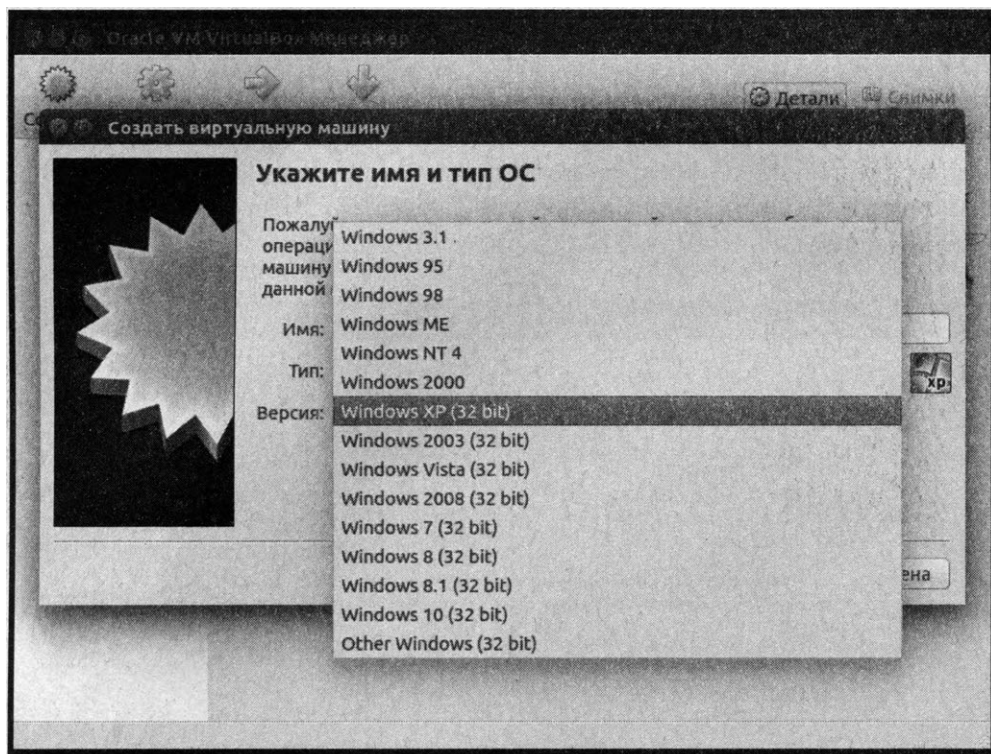


Рис. 18.2. Ubuntu: выбор гостевой ОС в VirtualBox

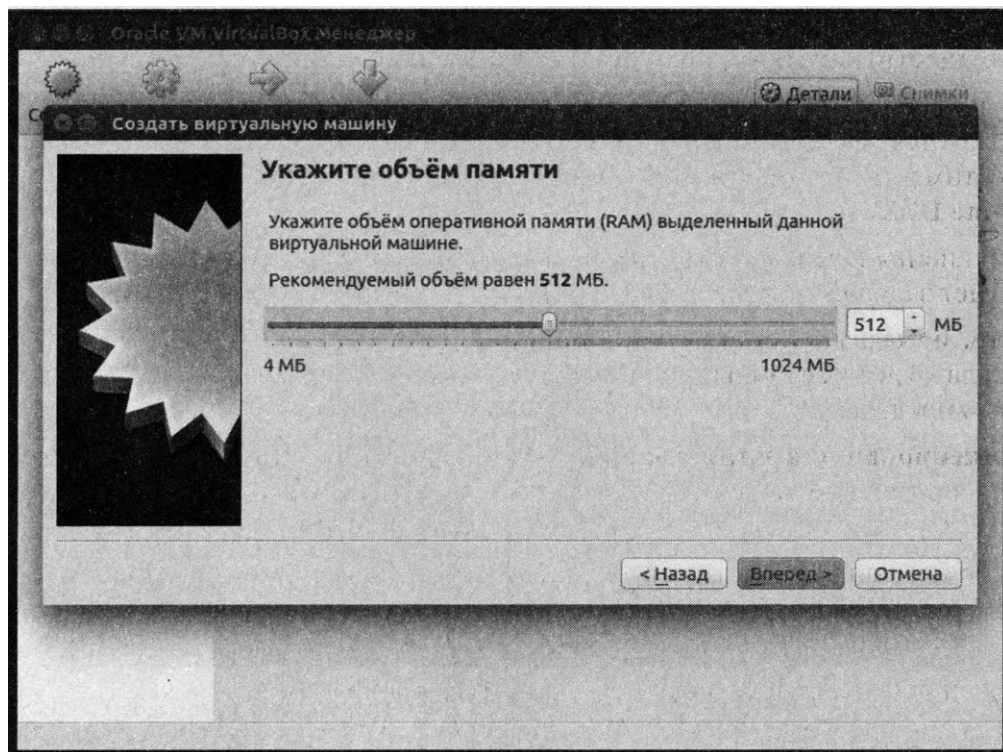


Рис. 18.3. Ubuntu: установка размера оперативной памяти в VirtualBox

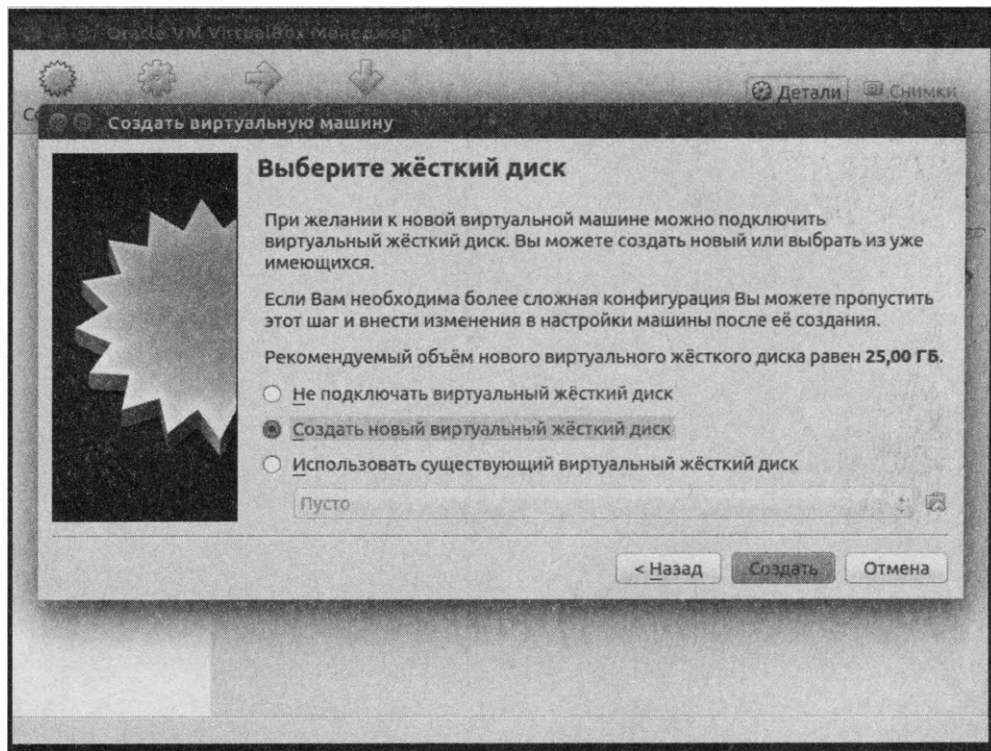


Рис. 18.4. Ubuntu: создание нового виртуального жесткого диска в VirtualBox

уже существующего файла можно выбрать его из списка по варианту **Использовать существующий виртуальный жесткий диск** (Use existing hard disk).

Следующий шаг — выбор формата виртуального диска (рис. 18.5). По умолчанию используется формат **VDI (VirtualBox Disk Image)**. Если нужно обеспечить совместимость с виртуальной машиной VMware, выберите **VMDK (Virtual Machine Disk)**.

Нажав кнопку **Вперед** (Next), мы попадем в окно (рис. 18.6), где нужно определить, как будет задан размер жесткого диска:

- **Динамический виртуальный жесткий диск** (Dynamically allocated) — размер файла будет увеличиваться по мере необходимости, но не превысит заданного вами предела;
- **Фиксированный виртуальный жесткий диск** (Fixed size) — размер четко зафиксирован вне зависимости от того, сколько места реально занимает операционная система.

Обычно первый вариант наиболее приемлем — ведь вы можете установить размер, например, 20 Гбайт, а реально операционная система займет из них всего лишь 8. Выходит, что 12 Гбайт просто не будут использоваться.

С другой стороны, установив фиксированный размер, вы можете не беспокоиться, что другие приложения «скушают» свободное место, и гостевой ОС ничего не останется, — ведь нужное дисковое пространство уже зарезервировано. Поразмыслив, выберем первый вариант и нажмем кнопку **Вперед**.

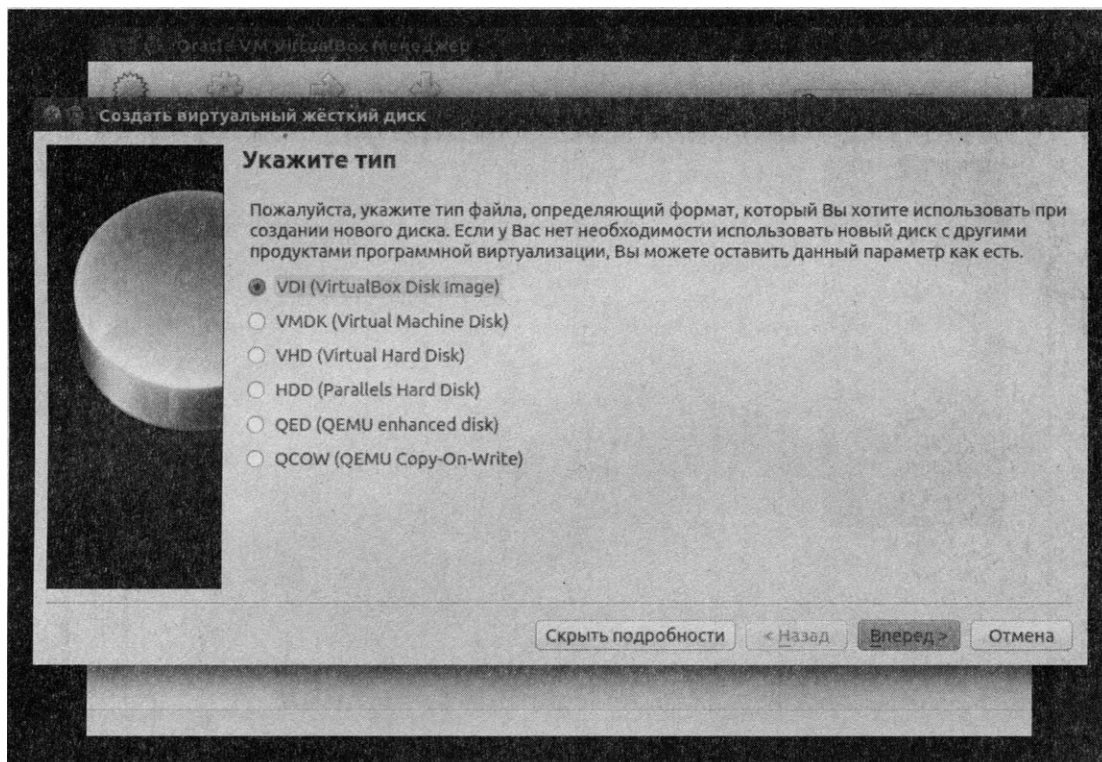


Рис. 18.5. Ubuntu: выбор формата файла виртуального диска в VirtualBox

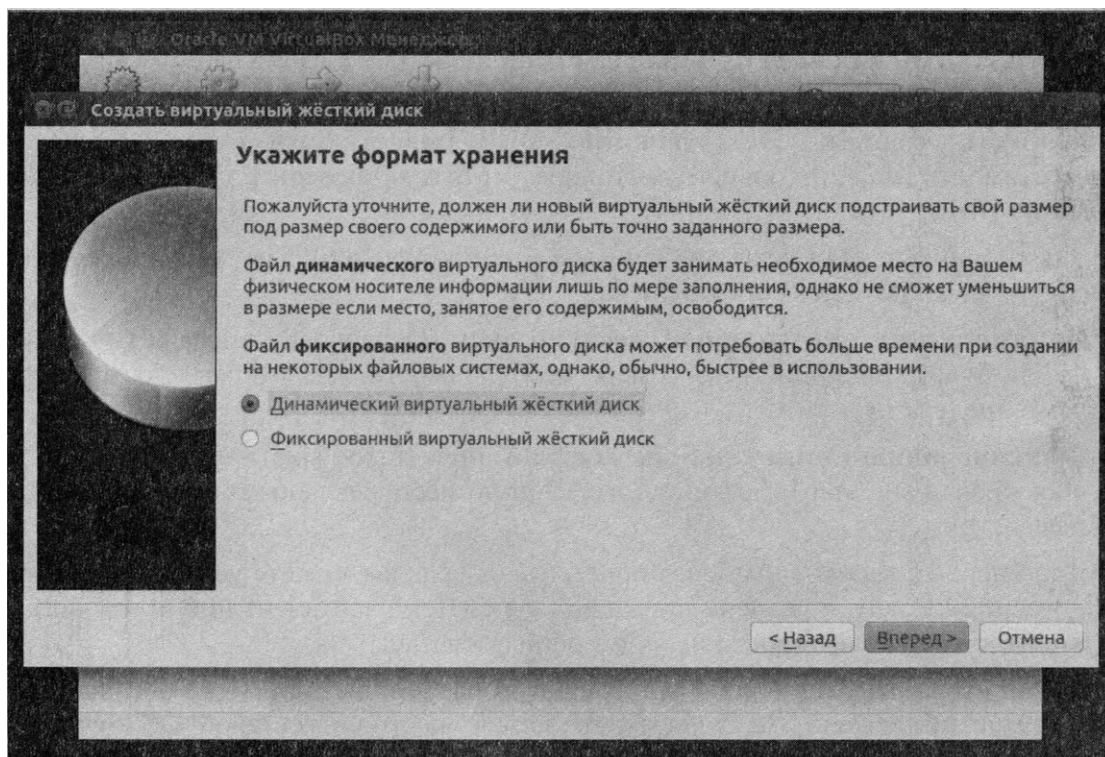


Рис. 18.6. Ubuntu: выбор метода резервирования дискового пространства в VirtualBox

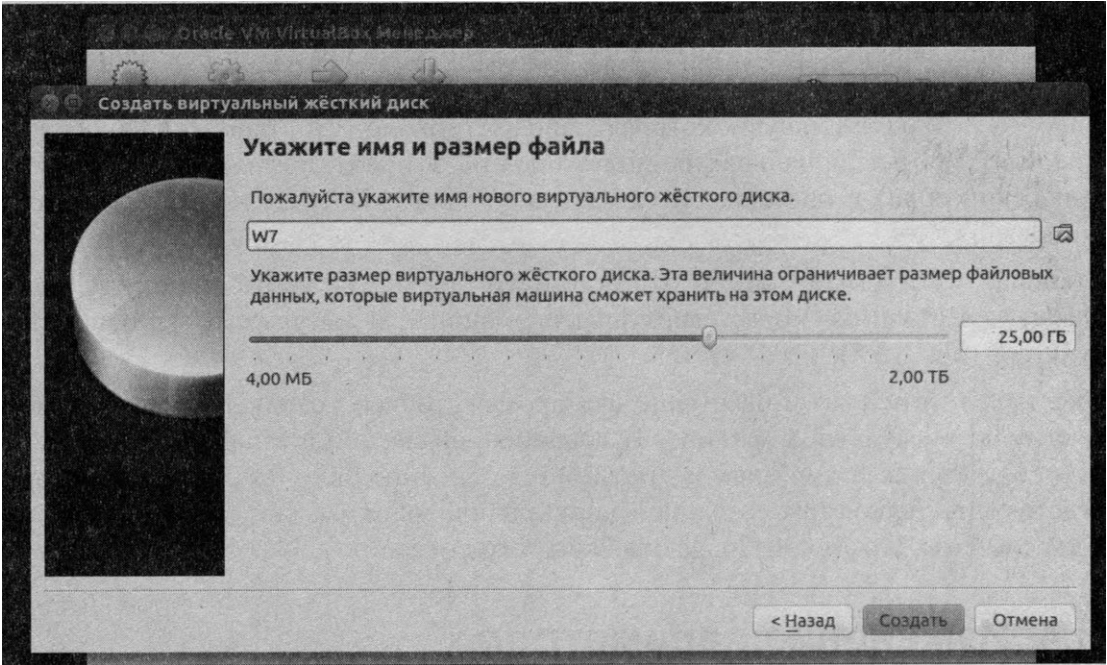


Рис. 18.7. Ubuntu: задание размера виртуального диска в VirtualBox

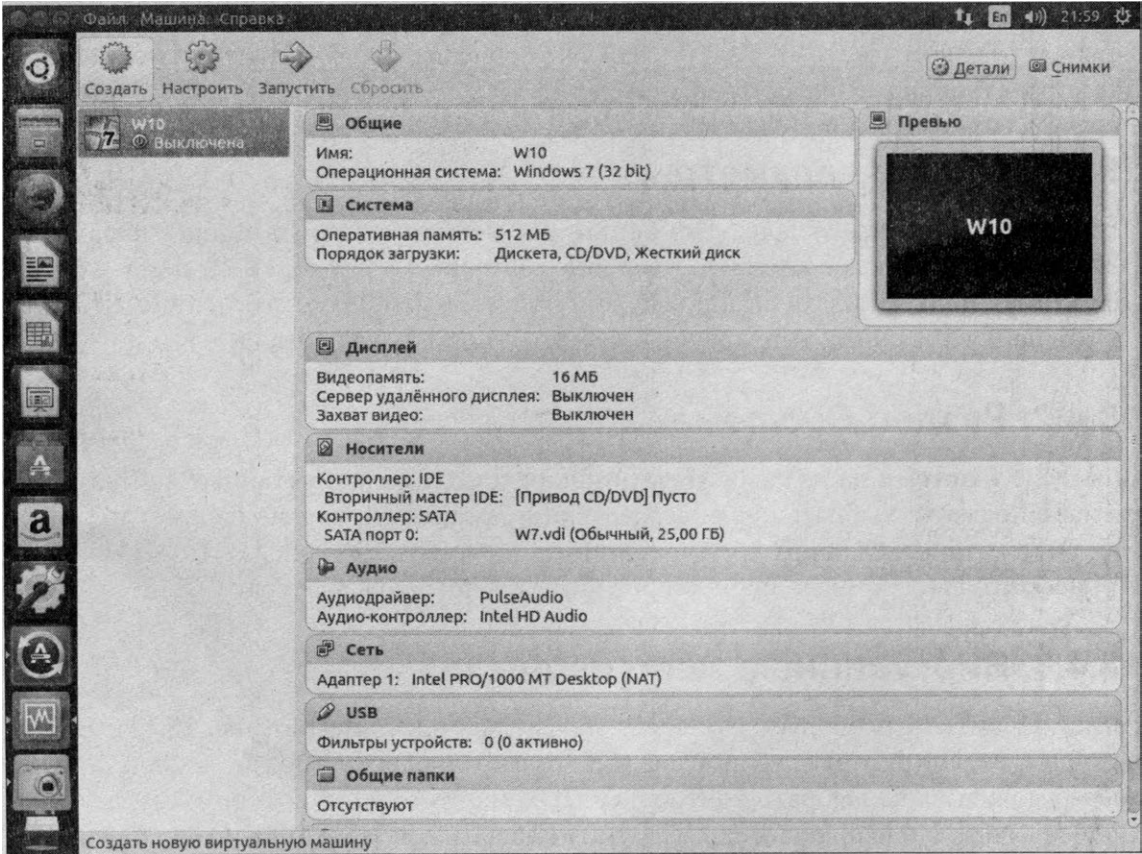


Рис. 18.8. Ubuntu: созданная виртуальная машина в окне VirtualBox

Сразу после выбора метода резервирования места на диске нужно установить размер виртуального диска (рис. 18.7). В моем случае эмулятор рекомендует установить размер 25 Гбайт. Очевидно, он полагает, что я буду устанавливать Windows 7 Максимальную, которая занимает около 15 Гбайт. Однако та же Windows 7 Профессиональная занимает около 8-9 Гбайт, поэтому я и выбрал динамический образ в надежде, что сэкономлю место на диске — ведь диск-то не резиновый.

Установив требуемый размер виртуального диска, нажмите кнопку **Создать** (Create)— VirtualBox создаст виртуальную машину, и вы увидите основное окно эмулятора (рис. 18.8).

Ранее перед этим шагом было еще два промежуточных: сводки по создаваемому диску и по виртуальной машине. В современной версии VirtualBox этого нет — сразу создается жесткий диск и открывается основное окно эмулятора, в котором представлены параметры созданной виртуальной машины: тип гостевой операционной системы, объем памяти, размер жесткого диска и т. д.

18.4. Изменение параметров виртуальной машины

Не спешите нажимать кнопку **Запустить** (Start) для запуска созданной виртуальной машины (далее в тексте — ВМ) — нажмите сначала кнопку **Настроить** (Settings) для корректировки ее параметров.

18.4.1. Общие параметры

В разделе **Общие** на вкладке **Основные** вы можете изменить общие параметры ВМ: название, тип гостевой ОС, версию гостевой ОС (рис. 18.9), а на вкладке **Дополнительно** — задать различные дополнительные параметры: например, определить папку для снимков с экрана, включить общий буфер обмена и т. д.

18.4.2. Раздел система

В разделе **Система** на вкладке **Материнская плата** можно установить объем оперативной памяти, выбрать порядок загрузки и установить другие параметры — для работы некоторых ОС приходится, например, включать EFI (рис. 18.10). На вкладке **Процессор** задается число процессоров виртуального компьютера.

18.4.3. Виртуальные жесткие диски

Раздел **Носители** позволяет изменить образы жестких дисков (рис. 18.11). В предыдущей версии VirtualBox допускалось только добавлять образы дисков, а в новой — можно даже выбрать контроллер (IDE или SATA), к которому будет «подключен» виртуальный носитель. По умолчанию здесь к SATA-контроллеру будет подключен всего один образ — тот, который вы организовали при создании ВМ.

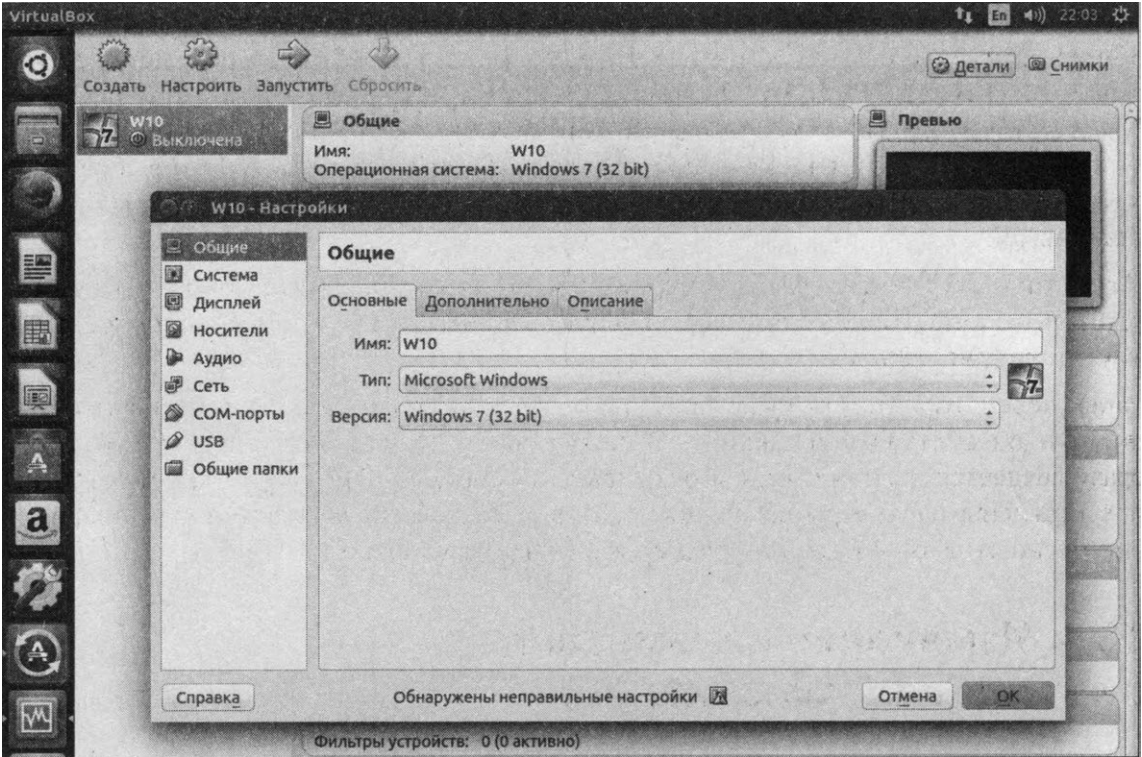


Рис. 18.9. Ubuntu: общие параметры виртуальной машины в VirtualBox

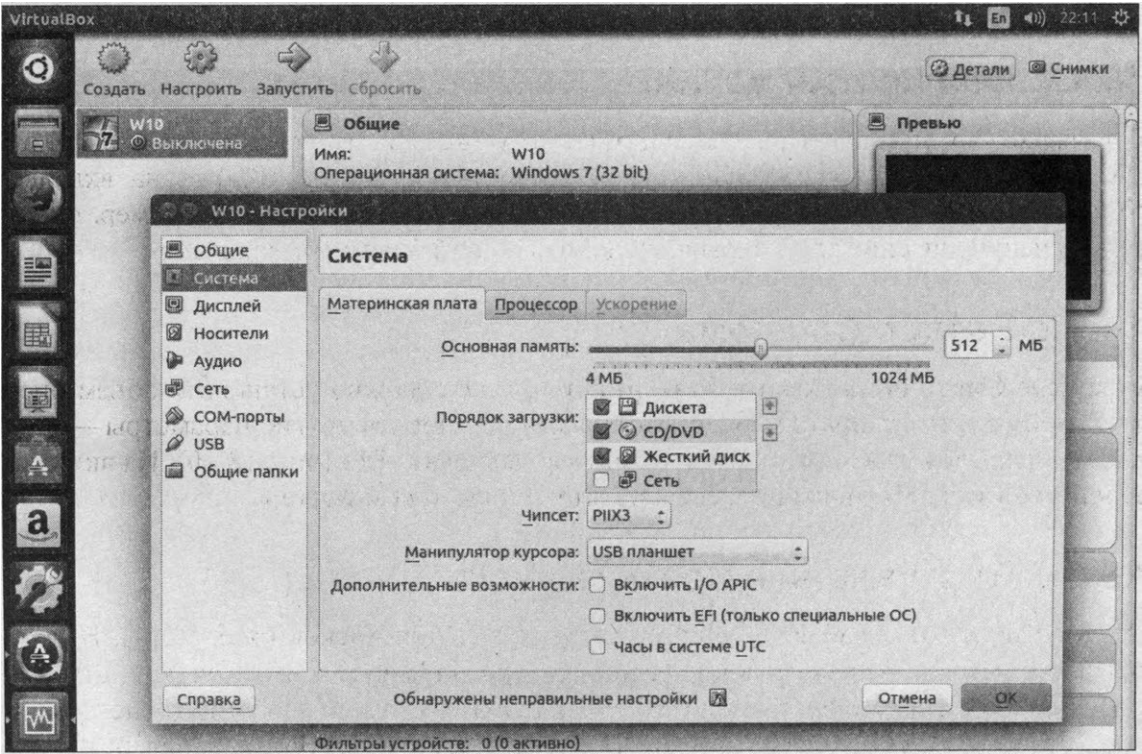


Рис. 18.10. Ubuntu: раздел Система в VirtualBox

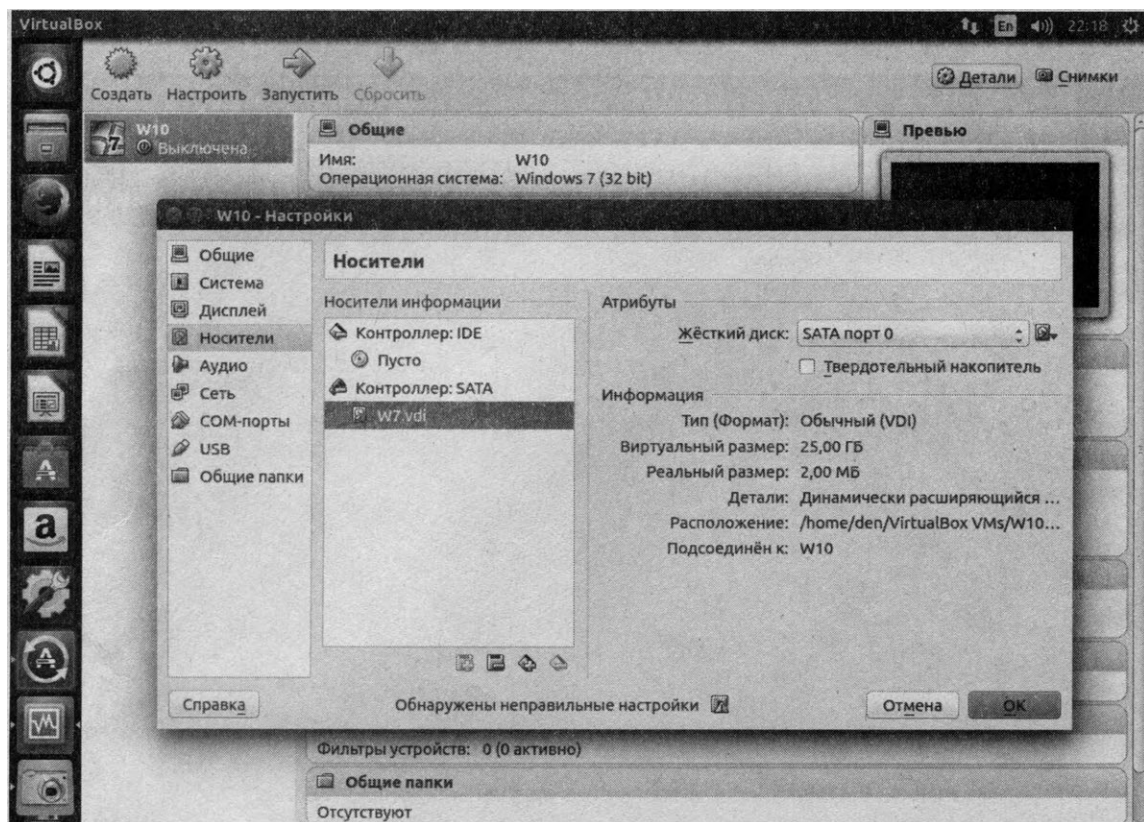


Рис. 18.11. Ubuntu: параметры жестких дисков в VirtualBox

Также ранее в VirtualBox существовал раздел **CD/DVD-ROM**, в котором можно было установить параметры виртуального привода DVD. Сейчас эти параметры устанавливаются в разделе **Носители** — по умолчанию DVD-привод подключен к IDE-контроллеру и соответствует физическому приводу DVD.

Впрочем, обычно при работе с виртуальными машинами никто не устанавливает гостевую операционную систему с записанного на болванку дистрибутива — в большинстве случаев установочный образ диска гостевой ОС мы загружаем из Интернета и записываем на винчестер, поэтому хотелось бы использовать именно его, не расходуя лишнюю болванку. Для этого щелкните по диску **Пусто** (Empty) в дереве носителей (рис. 18.12) и нажмите кнопку выбора диска, а из открывшегося меню выберите команду **Выбрать образ оптического диска** (Choose a virtual CD/DVD disk file) — откроется диалоговое окно выбора файлов, в котором нужно выбрать записанный на винчестер образ диска.

18.4.4. А нужен ли звук?

Обычно звук в виртуальной машине не нужен, но если сильно хочется, то почему бы и нет? Для установки параметров звука перейдите в раздел **Аудио** (рис. 18.13). Linux обычно использует звуковую систему ALSA, поэтому в параметрах ВМ нужно выбрать именно ее.

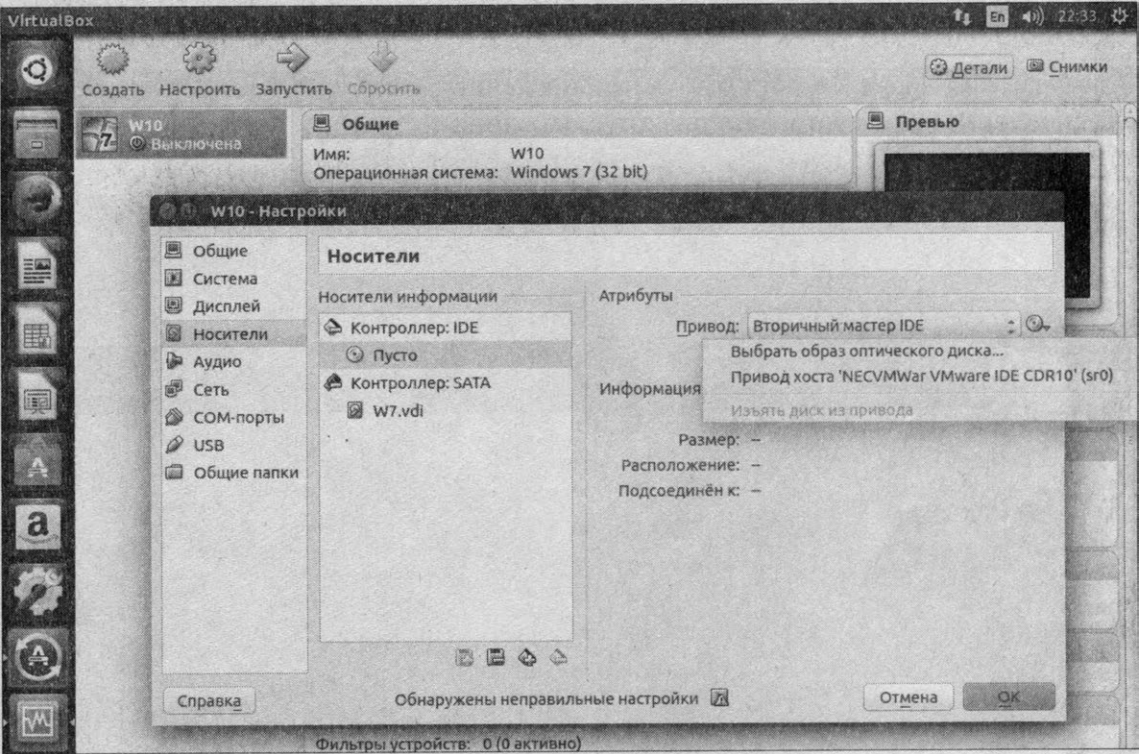


Рис. 18.12. Ubuntu: выбор образа оптического диска в VirtualBox

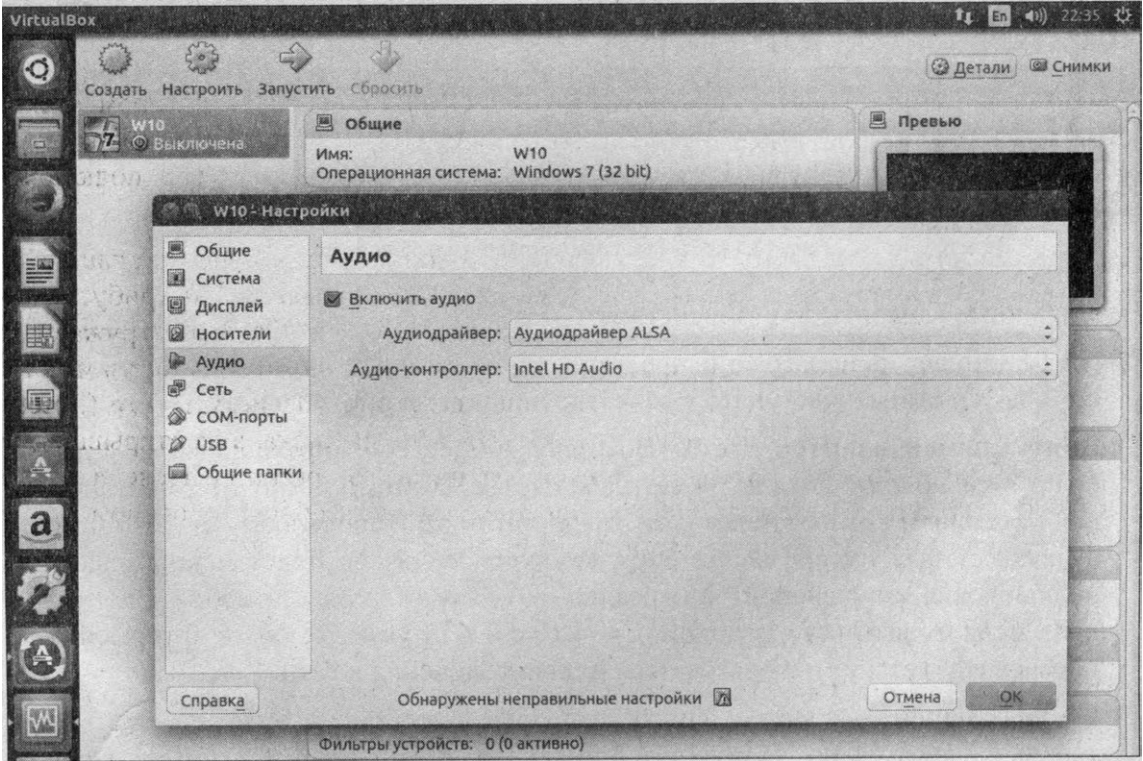


Рис. 18.13. Ubuntu: параметры звука в VirtualBox

18.4.5. Параметры сети

В разделе **Сеть** (рис. 18.14) вы можете определить, как будет гостевая операционная система взаимодействовать по сети с основной.

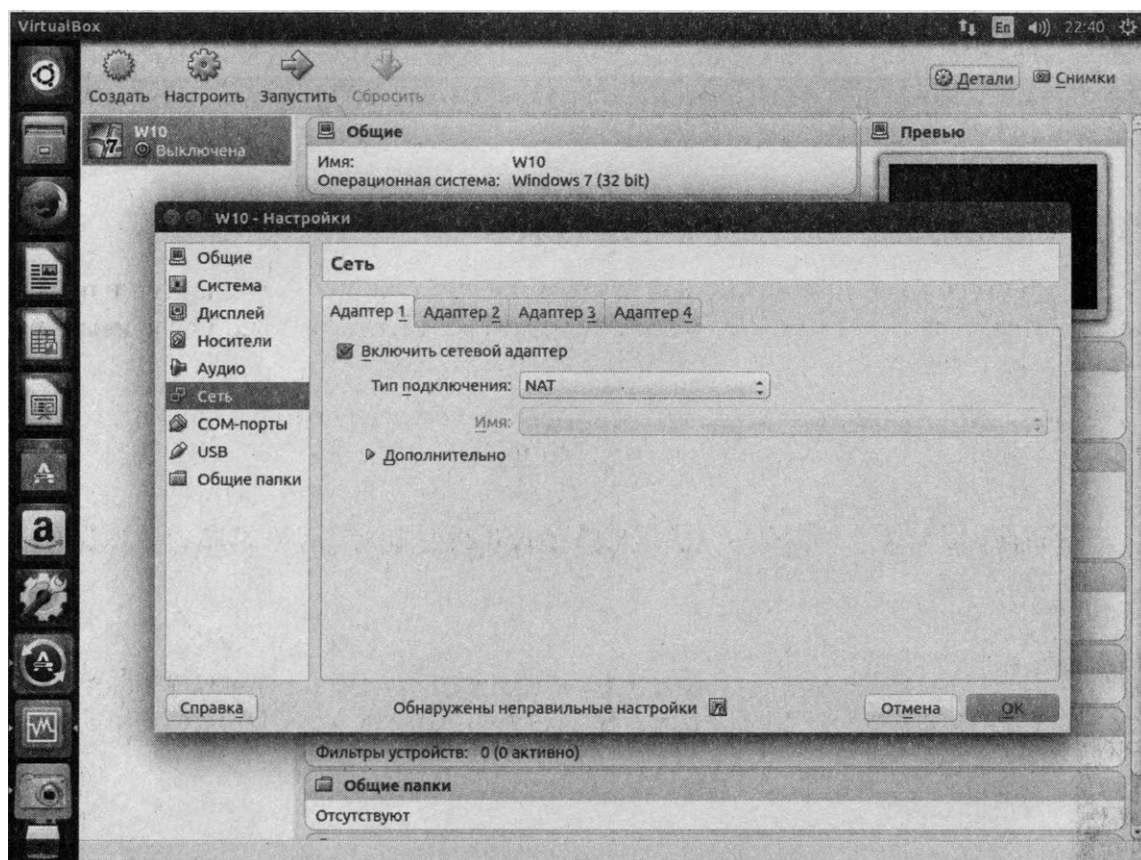


Рис. 18.14. Ubuntu: параметры сети в VirtualBox

Вот основные варианты взаимодействия:

- **NAT** — будет использовано преобразование сетевых адресов, т. е. реальный компьютер станет выступать в качестве шлюза для виртуального;
- **Виртуальный адаптер хоста (Host-only Adapter)**— виртуальный компьютер будет подключен к локальной сети как самый обычный компьютер и станет виден остальным компьютерам сети. Недостаток этого способа — необходимость в двух сетевых интерфейсах: один сетевой интерфейс будет использован для подключения к локальной сети реальным компьютером, а второй — виртуальным. Если вы выберете эту опцию, то вам следует указать, какой интерфейс (например, eth1) станет использоваться для подключения к сети;
- **Не подключен** — у виртуального компьютера вообще не будет сетевого адаптера и, следовательно, не получится никакого сетевого взаимодействия с основным компьютером;

- **Внутренняя сеть** — виртуальный компьютер будет подключен к собственной внутренней сети, о которой ничего не будет знать основной компьютер. Так что, никакого сетевого взаимодействия с основным компьютером не получится и в этом случае.

Если у вас всего один сетевой адаптер, тогда оптимальным является первый вариант — NAT.

Остальные вкладки раздела **Сеть** позволяют добавить в **ВМ** дополнительные виртуальные сетевые адаптеры.

18.4.6. Последовательные порты

Раздел **COM-порты** позволяет определить, как ВМ будет получать доступ к последовательным портам физического компьютера (рис. 18.15). Обычно последовательные порты отключены.

Остальные параметры настройки нам не интересны, и теперь вы можете нажать кнопку **ОК** для возврата в основное окно VirtualBox.

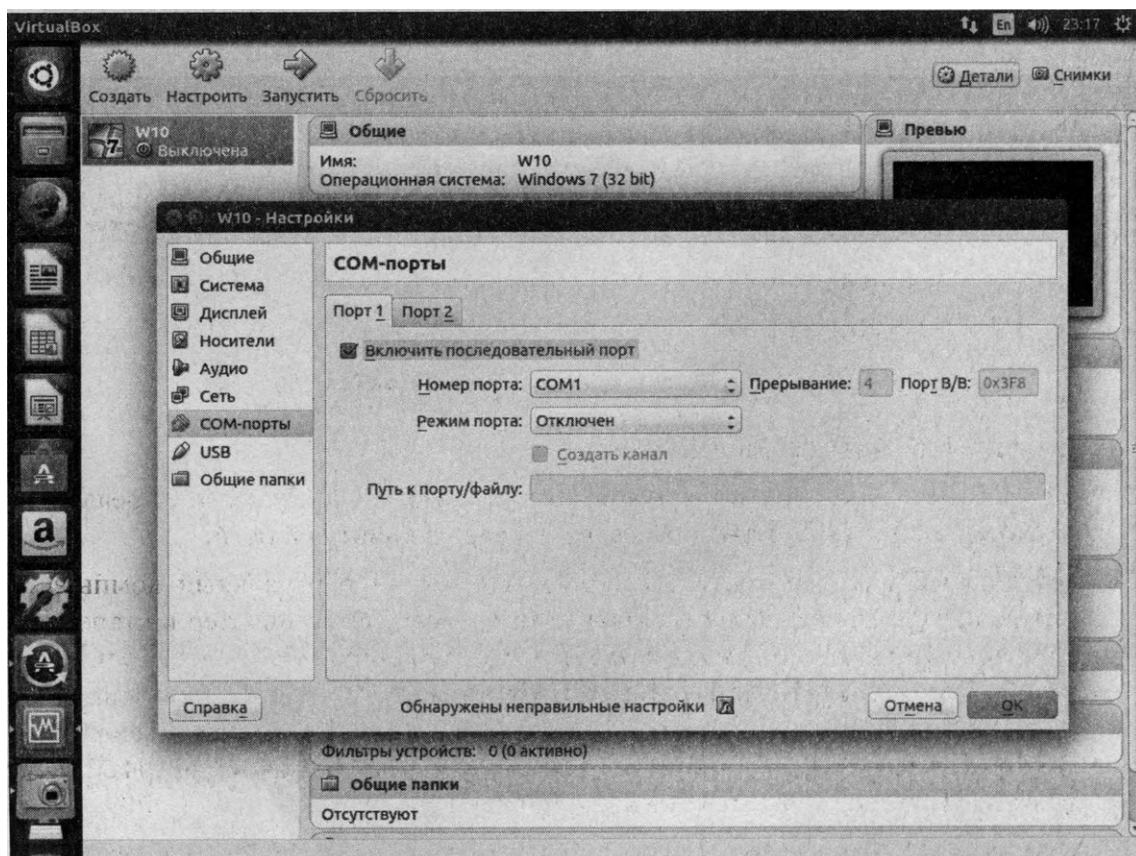


Рис. 18.15. Ubuntu: использование последовательных портов в VirtualBox

18.5. Запуск виртуальной машины и установка гостевой операционной системы

Нажмите кнопку **Запустить** (Start) — начнется запуск виртуальной машины (рис. 18.16). Теперь в ее окне все нажатия клавиш будут перехватываться и передаваться виртуальному компьютеру (в том числе и комбинация клавиш <Alt>+<Tab>) — пока вы не нажмете «горячую» клавишу (хост-клавишу). Этой «горячей» клавишей по умолчанию является правый <Ctrl> — обычно это самый удобный вариант, и он не требует изменений.

Собственно, на этом все. Осталось только установить гостевую операционную систему и приступить к ее использованию.

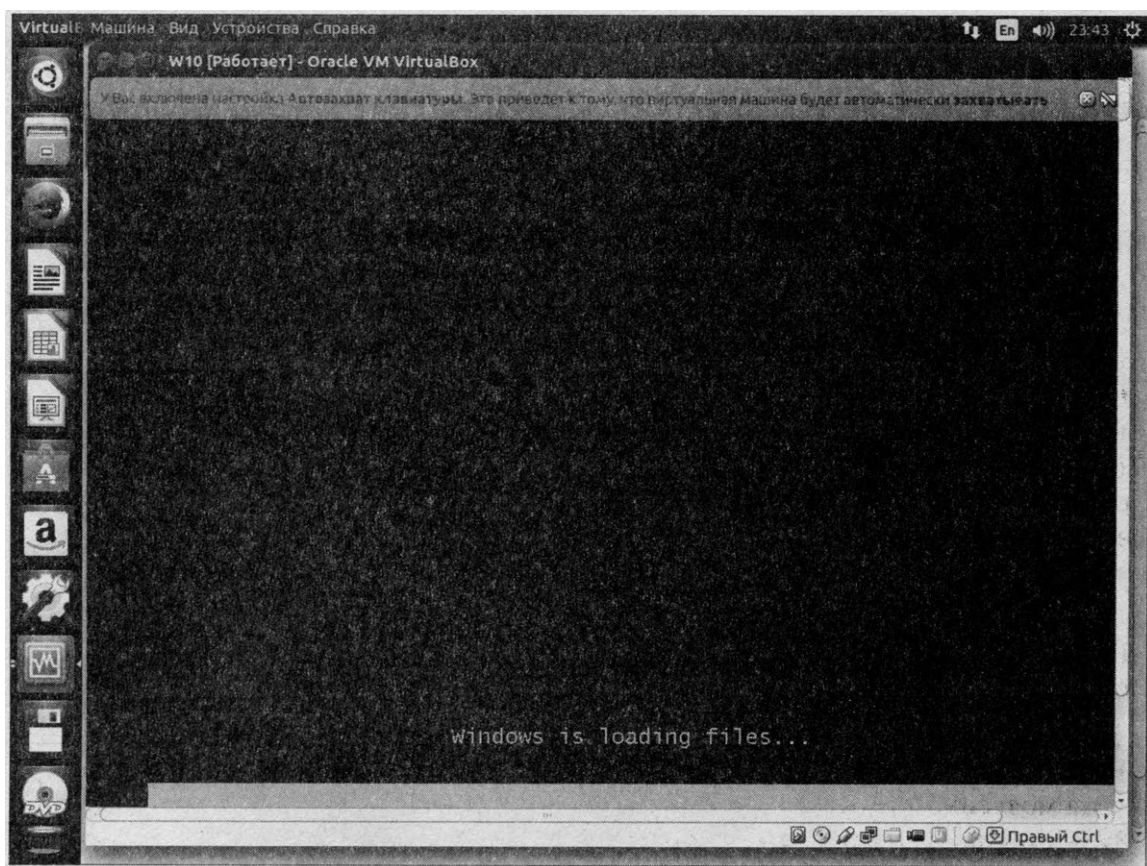
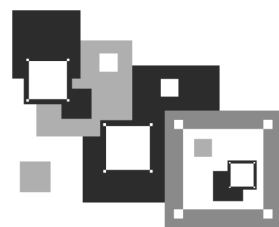


Рис. 18.16. Ubuntu: запуск виртуальной машины в VirtualBox

ГЛАВА 19



Эмулятор Wine: запуск Windows-игр в Linux

19.1. Эмуляторы, эмуляторы...

Как все мы знаем, в мире практически нет достойных Linux-игр. А те, что есть, можно пересчитать по пальцам. В мире Windows все иначе — игрушек намного больше. Вот и хочется иногда поиграть в любимую игрушку в любимой операционной системе, не запуская Windows. Понятно, что исполняемые файлы Windows не запускаются в Linux, поэтому линуксоидам остается одно — искать эмулятор Windows.

Различные эмуляторы виртуального компьютера, вроде VMware или VirtualBox, для этой цели непригодны — все они работают по принципу установки гостевой операционной системы: вы устанавливаете Windows, которая работает в эмуляторе, а потом в «виртуальной» Windows запускаете игру. Понятно, что страдает производительность, да и пропадает в этой затее весь смысл — ведь хочется отказаться от «пиратской» Windows и работать с чистой совестью. А в случае с подобным эмулятором уж проще перезагрузиться в Windows и запустить игру там — будет и удобнее, и быстрее.

Итак, нам нужен эмулятор, позволяющий запускать Windows-приложения без установки самой Windows. Таким эмулятором является бесплатный эмулятор Wine. Но вот беда — Wine не позволяет запускать игры. Все, что можно запустить с его помощью, — это обычные приложения, не использующие DirectX.

Эмулятор Wine — далеко не новинка мира OpenSource. Проект Wine был основан Бобом Амстадтом (Bob Amstadt) в 1993 году, т. е. 23 года назад! Развивался он сначала медленно (тогда просто не было острой необходимости в запуске из-под Linux Windows-приложений), а потом стал стремительно набирать обороты, — начали даже появляться основанные на Wine дистрибутивы с «прозрачной» поддержкой Windows-приложений.

В какой-то момент эмулятором Wine заинтересовалась компания TransGaming Technologies, и вскоре появился эмулятор Winex, позволяющий запускать Windows-игры. Первая его версия еще распространялась бесплатно, но ее функциональность оставляла желать лучшего: некоторые игры не запускались, некоторые работали нестабильно, в некоторых были проблемы со звуком или изображением. Да и работал эмулятор откровенно медленно.

Но компания TransGaming не останавливалась на достигнутом и постоянно совершенствовала свой эмулятор. И вот, начиная с четвертой версии (это произошло в 2004 году), эмулятор был переименован в Cedega и стал намного проще в использовании. Теперь в нем запускается большинство игр (проще указать, какие не запускаются, чем перечислить запускаемые), и можно действительно играть, а не наслаждаться фактом запуска игры под Linux.

Все бы хорошо, но, как всегда, есть одно «но» — эмулятор Cedega не бесплатный. Месячная подписка (лицензия) стоит 5 долларов, а лицензия на год — 50 долларов. Помню, как-то попробовал взломанную версию Cedega— эмулятор работал достойно, но использовать «пиратское» программное обеспечение, да еще и в Linux...

Впрочем, разработчики Wine тоже не стояли на месте. Некоторое время назад я снова установил Wine и обнаружил, что он теперь поддерживает DirectX, а, следовательно, в нем можно запускать игры — разумеется, абсолютно законно и бесплатно. Что мы и попытаемся сделать.

19.2. Установка Wine

Установите Wine— или через Synaptic (рис. 19.1), или с помощью apt-get как вам удобнее.

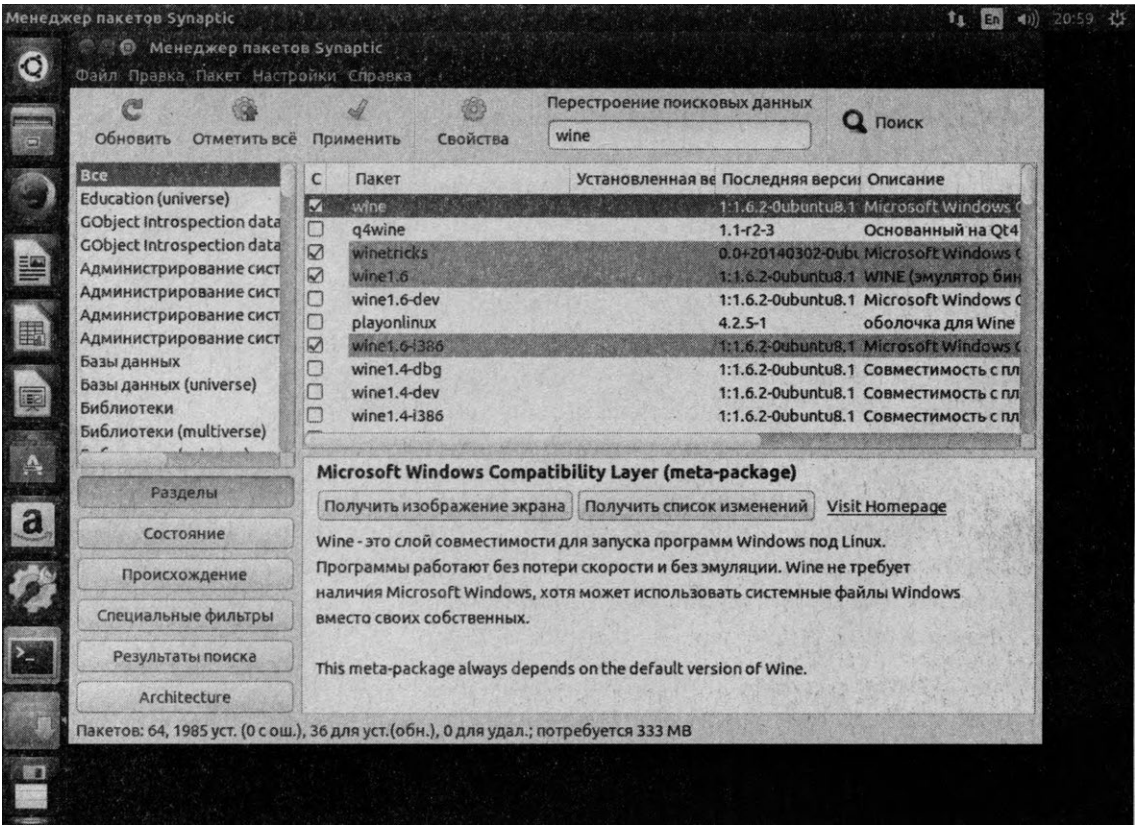


Рис. 19.1. Ubuntu: установка Wine

РАБОТАЕМ В UBUNTU

В этой главе рассматривается работа с Wine на примере Ubuntu, но в других дистрибутивах все будет происходить точно так же, кроме самого процесса установки Wine, — для установки пакета wine придется использовать менеджер пакетов дистрибутива.

После установки Wine в меню **Приложения** появится группа **Wine**. В ней вы найдете:

- группу **Programs** — сюда помещаются установленные в Wine Windows-программы. По умолчанию доступно только приложение Notepad — да, это знаменитый Блокнот;
- опцию **Browse C:\ Drive** — просмотреть «виртуальный» диск C: (на него и устанавливаются Windows-программы). Отредактировав конфигурационный файл Wine (по умолчанию ~/.wine), можно создать и другие диски, но, как правило, в этом нет необходимости;
- опцию **Configure Wine** — настроить Wine;
- опцию **Uninstall Wine Software** — удалить установленные Windows-программы.

Эту группу вы найдете в любом дистрибутиве Linux, но не в Ubuntu, — начиная с Ubuntu 11, с появлением интерфейса Unity, главное меню системы видоизменено, и обычных программных групп в нем нет. Поэтому проще всего ввести wine в поле поиска — тогда вы увидите значки, относящиеся к Wine (рис. 19.2).

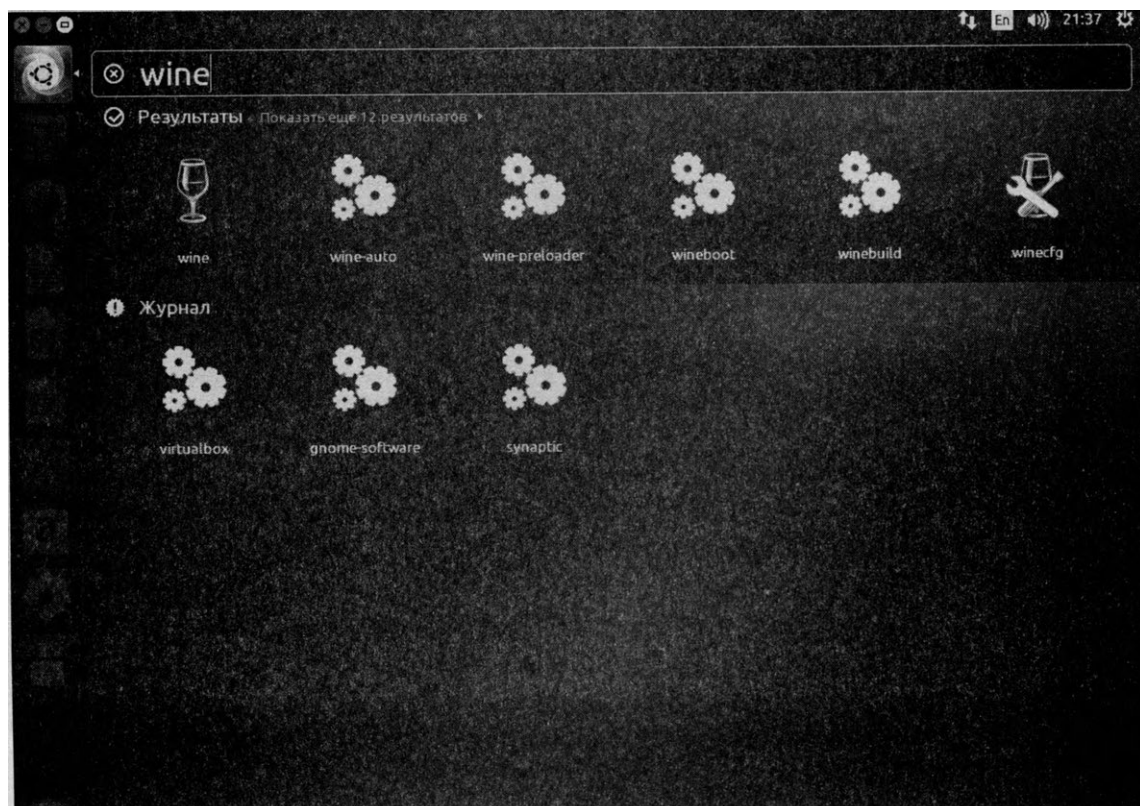


Рис. 19.2. Ubuntu: значки группы Wine

19.3. Настройка Wine и прозрачного запуска Windows-приложений

Перед тем как приступить к установке Windows-приложений, Wine необходимо настроить. Выберите команду **Настройка Wine** (Configure Wine) или запустите программу **winecfg** (см. рис. 19.2). В открывшемся окне (рис. 19.3) на вкладке **Приложения** вы можете выбрать версию Windows. Пока еще Wine не поддерживает самую новую версию — Windows 10, однако поддержка Windows 7 и 8 имеется, чего на данный момент более чем достаточно.

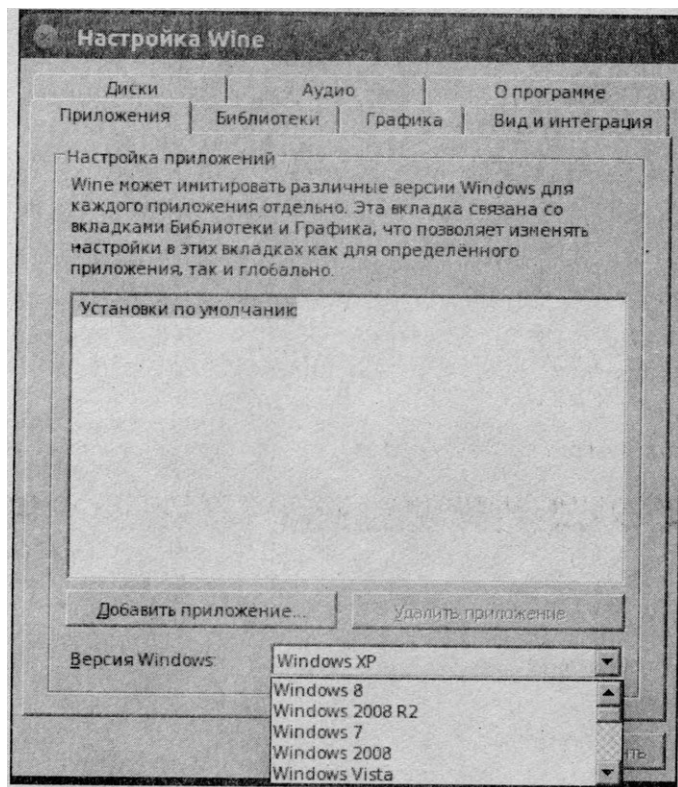


Рис. 19.3. Ubuntu: выбор версии Windows в Wine

Загляните на вкладку **Графика** — там можно увидеть, что Wine поддерживает DirectX, Direct3D и даже Pixel Shader (если, конечно, Pixel Shader поддерживается вашей видеокартой).

Теперь перейдите на вкладку **Аудио** (рис. 19.4) и нажмите кнопку **Проверить звук**. Если звука не слышно (хотя у меня все работало по умолчанию), выберите другой драйвер и снова проверьте звук.

Закройте окно настроек — оно нам более не понадобится. И осталось только обеспечить прозрачный запуск Windows-приложений — чтобы Wine по щелчку на EXE-файле в файловом менеджере запускался автоматически и начинал игру. Единственное условие — EXE-файл должен быть исполнимым, иначе Wine откажется его запустить.

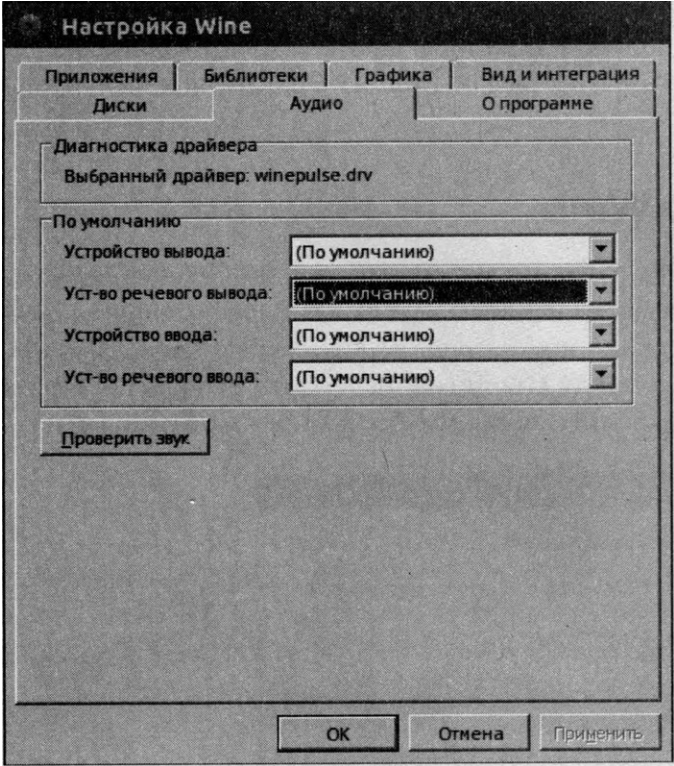


Рис. 19.4. Ubuntu: проверка звука в Wine

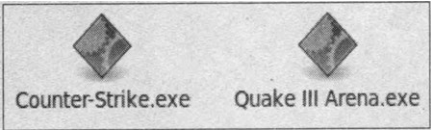


Рис. 19.5. Ubuntu: каталог с Windows-играми

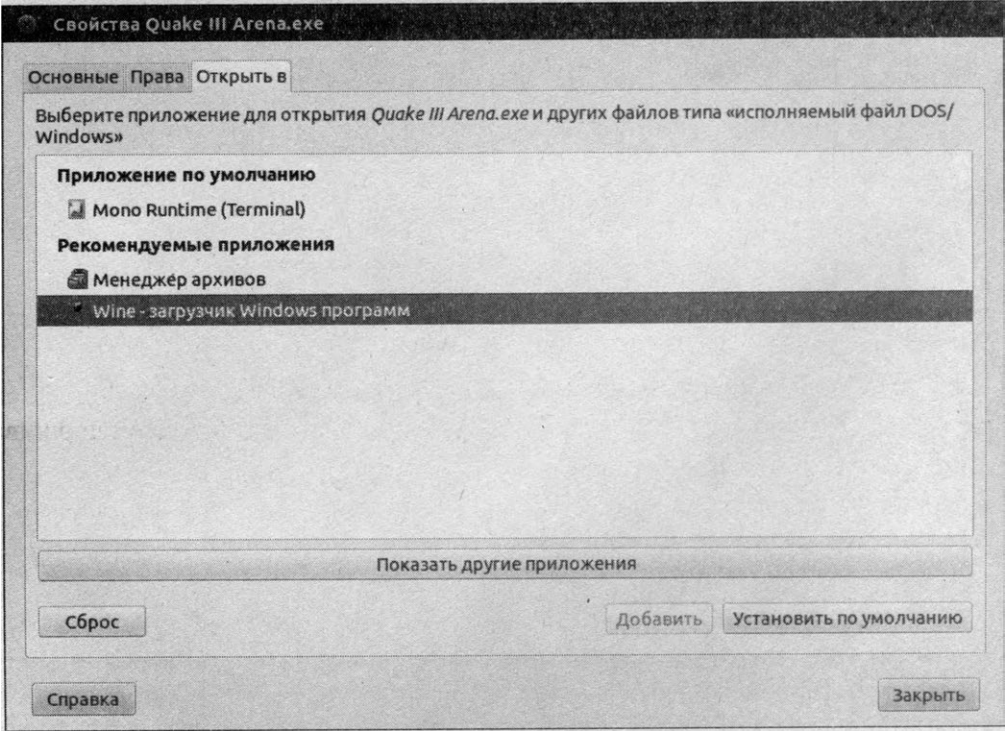


Рис. 19.6. Ubuntu: связываем EXE-файлы с Wine

Чтобы это условие обеспечить, откройте каталог с EXE-файлами игр (рис. 19.5), щелкните на EXE-файле правой кнопкой мыши и выберите команду **Свойства**. Затем перейдите на вкладку **Открыть в программе** (рис. 19.6), выберите там Wine и нажмите кнопку **Заккрыть**.

19.4. Использование Wine

Поскольку мы связали Wine с EXE-файлами, то для запуска Windows-программы достаточно щелкнуть на ее EXE-файле двойным щелчком. Если вы при запуске программы увидите сообщение, показанное на рис. 19.7, щелкните на EXE-файле правой кнопкой мыши, выберите команду **Свойства**, перейдите на вкладку **Права** (рис. 19.8) и установите флажок **Разрешить исполнение файла как программы**.

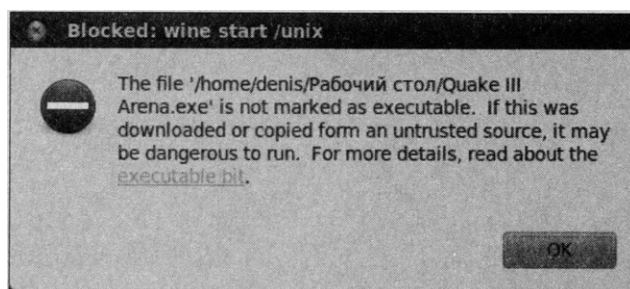


Рис. 19.7. Ubuntu: файл не помечен как исполнимый

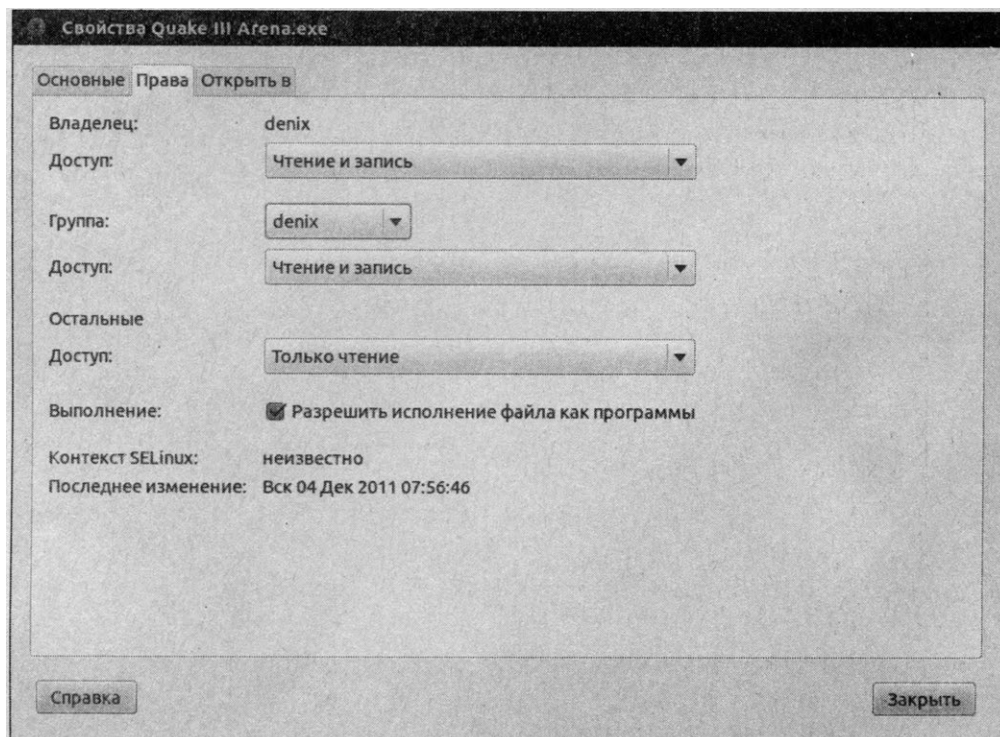


Рис. 19.8. Ubuntu: изменение прав доступа к файлу

После этого все должно заработать нормально — программа запустится. Для примера на рис. 19.9 показан запуск инсталлятора игры Quake III Arena, а на рис. 19.10 — завершение ее установки.

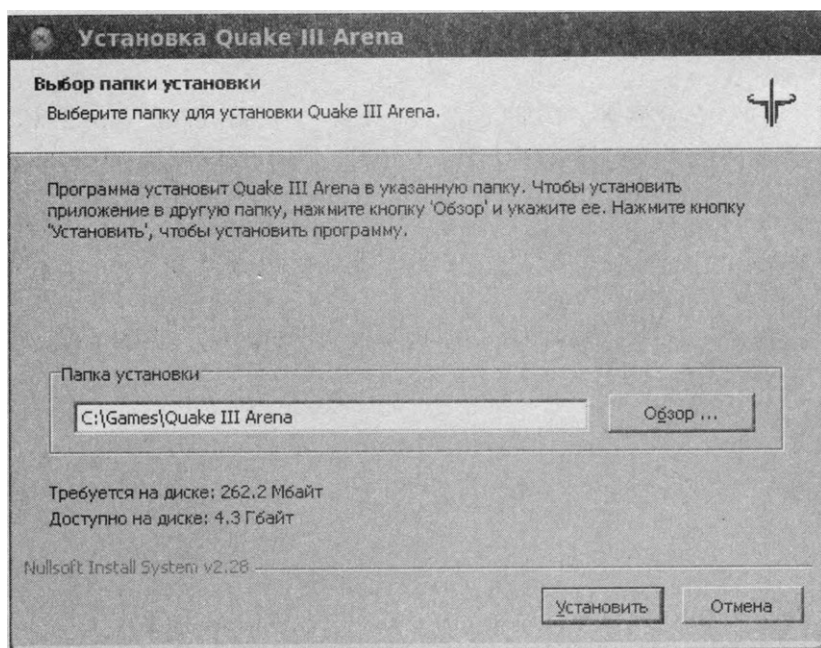


Рис. 19.9. Ubuntu: установка Quake III Arena в Wine

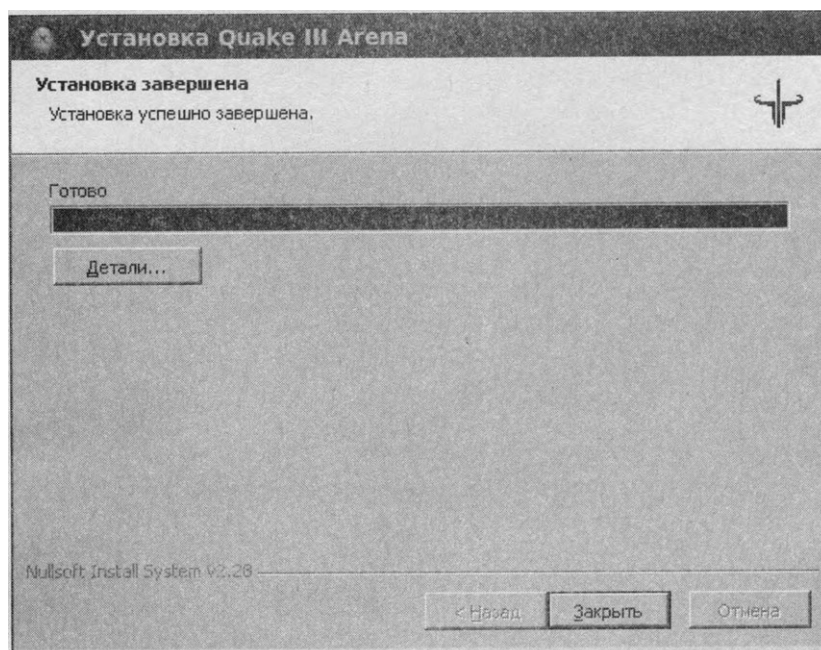
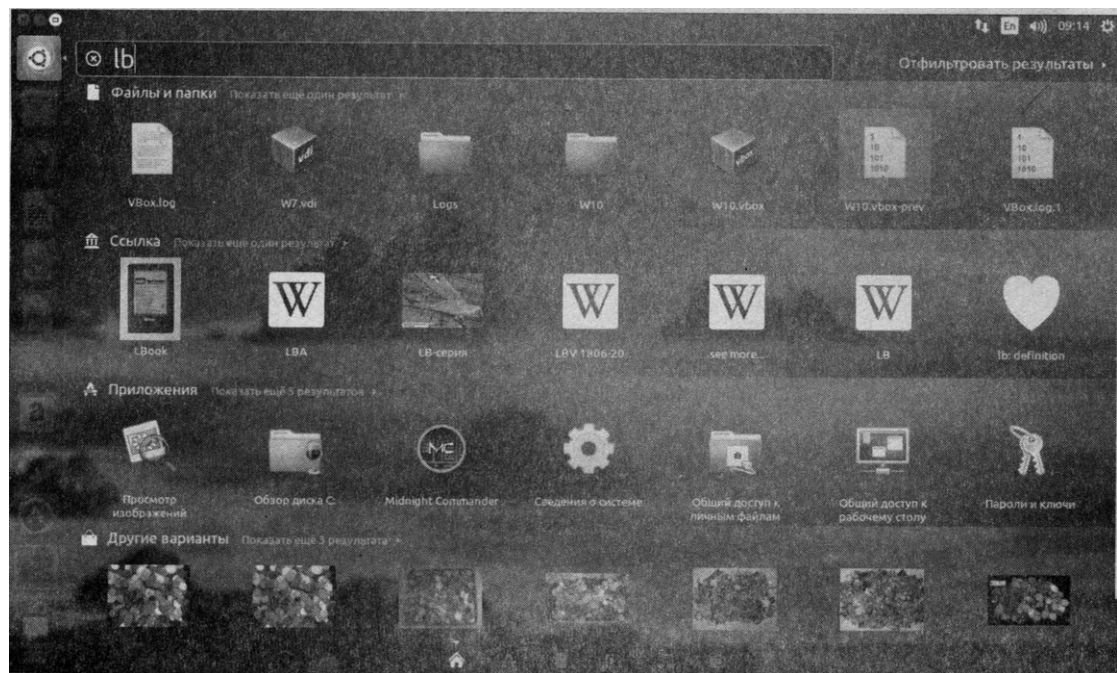
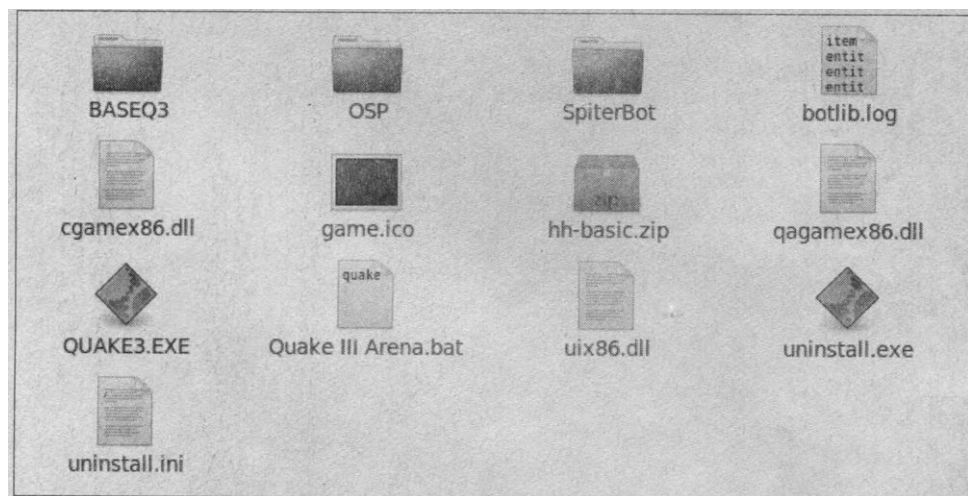


Рис. 19.10. Ubuntu: установка Quake III Arena в Wine завершена

Установленную программу можно найти с помощью экрана поиска приложений. Если вы там ее не найдете, нажмите на этом экране кнопку **Обзор диска C:** (рис. 19.11, а), и откроется каталог виртуального диска C:, в который вы установили программу (рис. 19.11, б).



а



б

Рис. 19.11. Ubuntu: а — экран поиска приложений; б—каталог с установленной игрой

Все, что вам остается — это запустить программу. Я сделал несколько иллюстраций (рис. 19.12-19.14), чтобы вы могли убедиться, что Windows-игры в Linux — это реальность.

Чтобы игра перешла в полноэкранный режим (рис. 19.15), нужно включить этот режим в настройках игры, а не в настройках Wine.

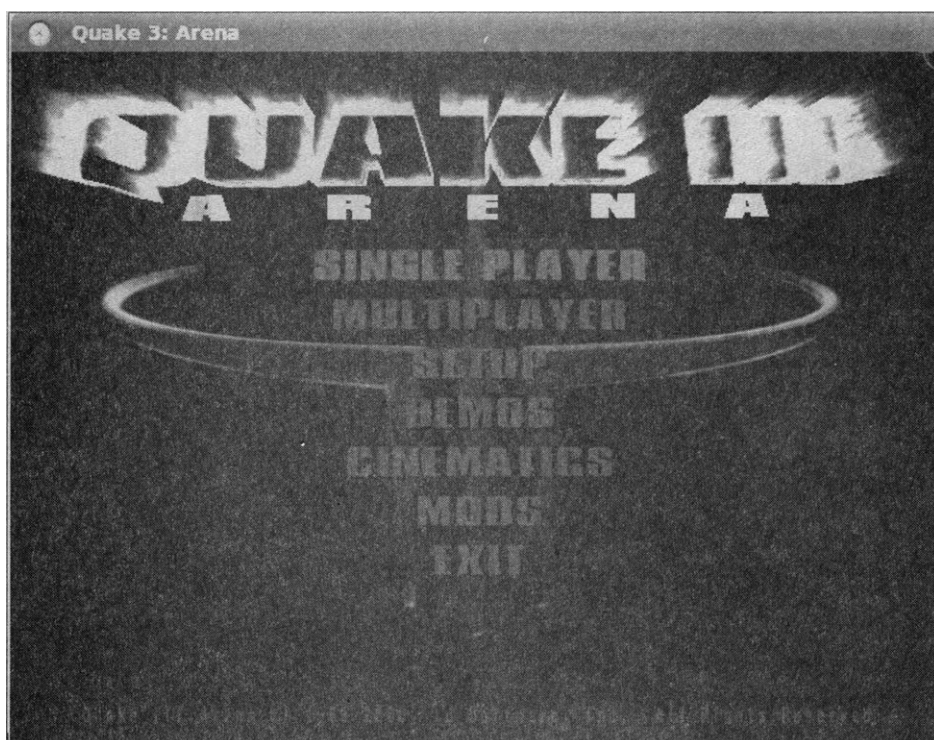


Рис. 19.12. Ubuntu: основное меню игры в Wine

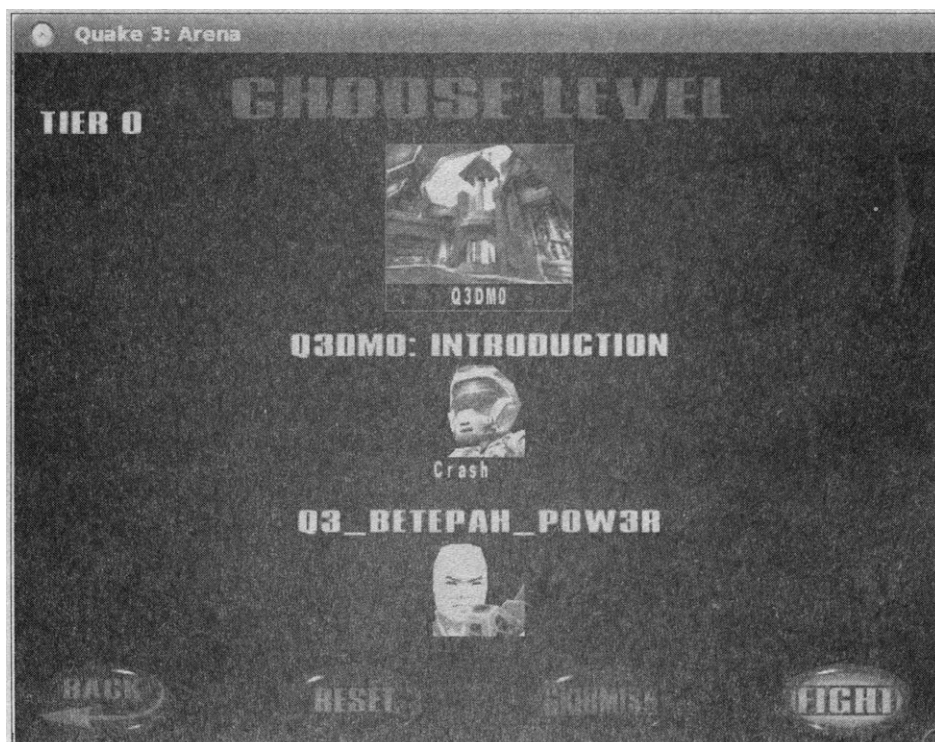


Рис. 19.13. Ubuntu: выбор уровня игры в Wine



Рис. 19.14. Ubuntu: игра запущена в Wine

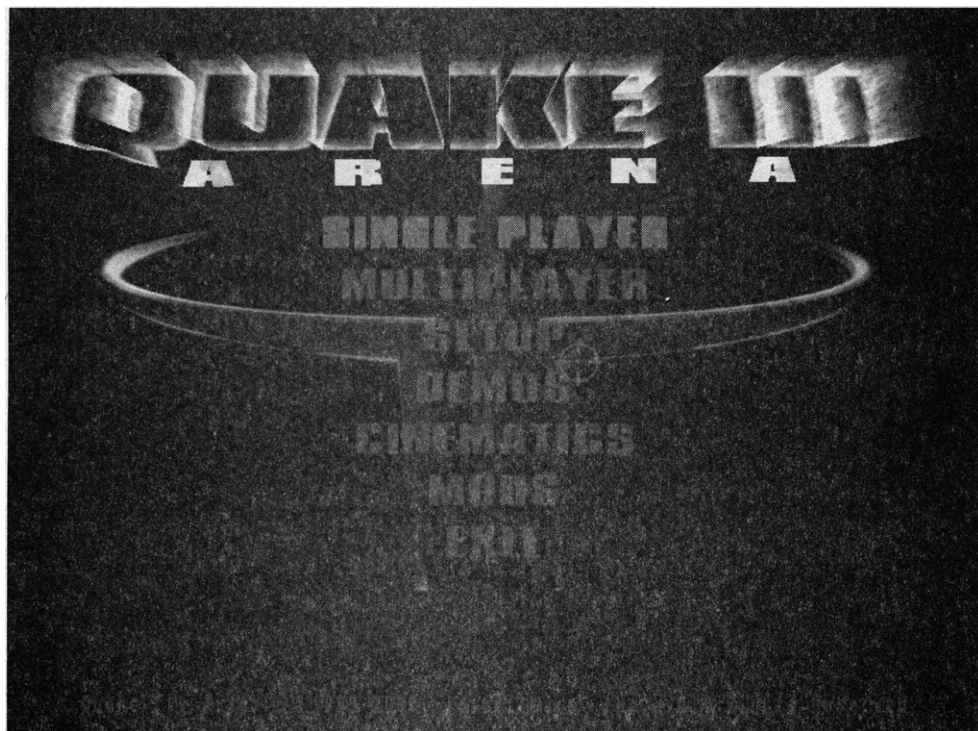
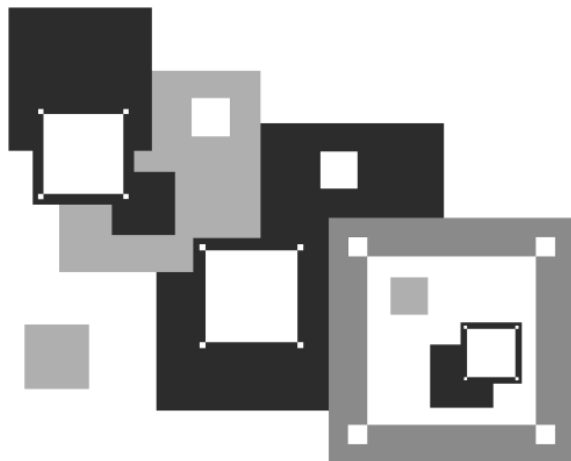


Рис. 19.15. Ubuntu: игра запущена в Wine в полноэкранном режиме

ПАРАМЕТР РАЗРЕШИТЬ МЕНЕДЖЕРУ ОКОН УПРАВЛЯТЬ ОКНАМИ WINE

Некоторые пользователи рекомендуют отключить параметр **Разрешить менеджеру окон управлять окнами Wine**. Не стоит этого делать, поскольку потом менеджер окон Ubuntu не сможет закрыть окно, когда вы нажмете на заветный крестик. А если Windows-игра зависнет, то, чтобы избавиться от окна, вам придется «убивать» процесс Wine или даже перезагружать X-сервер.

И в завершение хочется сказать пару слов о производительности Windows-игр в Linux. Как ни крути, а Wine — это, все-таки, эмулятор, поэтому производительность игры будет ниже, чем в родной ОС. Но на мощных современных компьютерах с 8 Гбайт и более оперативной памяти разницы почти не замечается. Ради эксперимента я запускал Linux в эмуляторе VMware, а затем в работающем из-под эмулятора Linux запускал эмулятор Wine и уже в нем — Windows-игру. Да, игра запустилась. Да, тормозило. В стрелялки и гонки не поиграешь, но в стратегию и в ролевые игры (типа «Diablo»), как оказалось, вполне можно играть даже в двойном эмуляторе, а просто в Wine — и подавно.

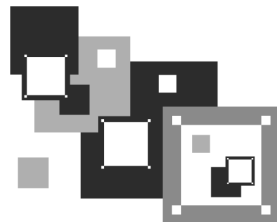


ЧАСТЬ V

Системные трюки, или Linux изнутри

Пятая часть книги посвящена различным «системным трюкам»: перекомпиляции ядра и передаче ему различных параметров, управлению системами инициализации и процессами, настройке загрузчиков, межпроцессному взаимодействию— в общем, всему тому, без чего не может и дня прожить настоящий линуксоид. Впрочем, не нужно думать, что Linux — такая неудобная операционная система, что в ней вы каждый день будете перекомпилировать ядро, — кстати, когда-то компиляция ядра была любимым видом «спорта» энтузиастов Linux. Нет, каждый пользователь найдет в Linux то, что искал: кто-то предпочтет спокойно работать, а кто-то не успокоится, пока не узнает, как устроена система изнутри. Таким пользователям и посвящена эта часть книги, хотя некоторые главы (особенно 21 и 25) нужно прочитать всем пользователям в обязательном порядке.

ГЛАВА 20



Ядро

20.1. Процесс загрузки ядра

Загрузка Linux начинается с вывода сообщений ядра: информационных (об имеющемся оборудовании, поддерживаемых протоколах и технологиях и т. д.) и диагностических (например, об ошибках). Однако современные компьютеры настолько быстры, что вы просто не успеваете проследить за их появлением на экране. Не беда, все сообщения всегда можно прочитать после загрузки системы с помощью команды:

```
# dmesg | less
```

Итак, сначала нам сообщат версию ядра и версию компилятора gcc, с помощью которого ядро было откомпилировано. Как можно видеть, у нас имеется дистрибутив Fedora 26 и версия ядра 4.11.8-300.fc26.x86_64:

```
[ 0.000000] Linux version 4.11.8-300.fc26.x86_64
```

```
(mockbuild@bkernel02.phx2.fedoraproject.org) (gcc version 7.1.1 20170622 (Red Hat 7.1.1-3)
(GCC)) #1 SMP Thu Jun 29 20:09:48 UTC 2017
```

Затем выводится карта физической памяти, предоставляемая BIOS:

```
[ 0.000000] e820: BIOS-provided physical RAM map:
```

```
[ 0.000000] BIOS-e820: [mem 0x0000000000000000-0x000000000009f7ff] usable
```

```
[ 0.000000] BIOS-e820: [mem 0x0000000000091800-0x000000000009ffff] reserved
```

```
[ 0.000000] BIOS-e820: [mem 0x00000000000ca000-0x00000000000cbfff] reserved
```

```
[ 0.000000] BIOS-e820: [mem 0x00000000000dc000-0x00000000000fffff] reserved
```

```
[ 0.000000] BIOS-e820: [mem 0x0000000000100000-0x00000000003fefffff] usable
```

```
[ 0.000000] BIOS-e820: [mem 0x0000000003fef000-0x0000000003fefeffff] ACPI data
```

```
[ 0.000000] BIOS-e820: [mem 0x0000000003feff00-0x0000000003fefffff] ACPI NVS
```

```
[ 0.000000] BIOS-e820: [mem 0x0000000003ff0000-0x0000000003ffffff] usable
```

```
[ 0.000000] BIOS-e820: [mem 0x00000000e0000000-0x00000000efffffff] reserved
```

```
[ 0.000000] BIOS-e820: [mem 0x00000000fec00000-0x00000000fec0ffff] reserved
```

```
[ 0.000000] BIOS-e820: [mem 0x00000000fee00000-0x00000000fee00fff] reserved
```

```
[ 0.000000] BIOS-e820: [mem 0x00000000fffe0000-0x00000000ffffffff] reserved
```

```
[ 0.000000] NX (Execute Disable) protection: active
```

Обратите внимание: функция запрета исполнения вредоносного кода (**NX protection**) включена.

ЗАПРЕТ ИСПОЛНЕНИЯ ВРЕДОНОСНОГО КОДА

Современные процессоры поддерживают функцию NX-защиты (NX — сокращение от No execute). Это защита областей памяти, которая используется для предотвращения распространения вирусов, «троянских коней» и других вредоносных программ. Довольно часто вредоносные программы нарочно вызывают переполнение буфера, после чего записывают свой код в область данных и передают ему управление. Функция NX-защиты как раз предотвращает развитие такого сценария на аппаратном уровне. Если ваше ядро и процессор поддерживают NX-бит, то вы увидите примерно такое сообщение: **Using x86 segment limits to approximate NX protection.**

Далее ядро проверит наличие DMI (Direct Media Interface). Правда, этого момента я не понял. DMI — это изобретение Intel, и представляет собой шину соединения южного и северного мостов материнской платы. По сути, DMI должно относиться только к процессорам Intel, а откуда ядро нашло DMI на материнской плате для процессора AMD Athlon X2, мне представить трудно. Видимо, нашелся какой-то аналог, и ядро сочло его за DMI. В любом случае для нас это не существенно, зато интересно, что ядро полностью выводит модель материнской платы вместе с версией BIOS, что поможет, если вы потеряли диск с драйверами для Windows XP ©:

[0.000000] DMI present.

[0.000000] DMI: MICRO-STAR INTERANTIONAL CO.,LTD MS-7367/MS-7367,
BIOS V1.0 06/11/2007

Далее выводится информация о диапазонных регистрах памяти (MTRR). Впрочем, если вы не профи в железе, то эта информация вам интересной не покажется:

[0.000000] MTRR default type: uncachable

[0.000000] MTRR fixed ranges enabled:

[0.000000] 00000-9FFFF write-back

[0.000000] A0000-EFFFF uncachable

[0.000000] F0000-FFFFFF write-protect

[0.000000] MTRR variable ranges enabled:

[0.000000] 0 base 0000000000 mask FF80000000 write-back

[0.000000] 1 disabled

[0.000000] 2 disabled

[0.000000] 3 disabled

[0.000000] 4 disabled

[0.000000] 5 disabled

[0.000000] 6 disabled

[0.000000] 7 disabled

ПОЛЕЗНЫЕ ССЫЛКИ

Допускаю, что если вы не уделяли особого внимания теории аппаратных средств ПК, то все эти термины (PAE, NX, MTRR) вам мало о чем говорят. Описывать их в книге не вижу смысла, поскольку книга эта о Linux, а не об аппаратных средствах. Но, к счастью,

есть Википедия, где эти термины, хоть и поверхностно, но описаны, а для общего развития большего и не нужно:

- <http://ru.wikipedia.org/wiki/PAE>;
- http://ru.wikipedia.org/wiki/NX_bit;
- <http://ru.wikipedia.org/wiki/MTRR>,

Здесь приведены ссылки на русские версии страниц, но если вы владеете английским, прочитайте эти же страницы на английском языке — информации получите больше.

Следующая строка:

[0.0000001 found SMP MP-table at [c00ff780] ff780

скажет нам, что найдена таблица SMP (Symmetrical Multiprocessing) MP (Multiprocessing). Это означает, что у нас или двухпроцессорный компьютер, или у нашего компьютера двухъядерный процессор.

Далее ядро сообщает объем оперативной памяти:

[0.000000] 1163MB HIGHMEM available.

[0.000000] 883MB LOWMEM available.

Складываем, округляем результат до ближайшей степени двойки и получаем 2048 Мбайт, или 2 Гбайт, — вроде бы, все правильно.

Мы не будем далее рассматривать абсолютно все сообщения ядра — их слишком много, а обратим внимание только на значимые.

Следующая строка сообщает нам параметры ядра, которые были переданы при загрузке системы:

**[0.000000] Kernel command line: BOOT_IMAGE= /vmlinuz-4.11.8-300.fc26.x86_x64
root=/dev/sdal ro rhgb quiet LANG=ru_RU.UTF-8**

Параметр `ro` говорит ядру о необходимости смонтировать корневую файловую систему в режиме «только чтение» (в процессе загрузки она будет перемонтирована в режим «чтение/запись» — `rw`). Корневая файловая система задается меткой `LABEL`. Помните, мы говорили о способах адресации разделов? Чтобы идентифицировать раздел, можно указать его короткое имя — вроде `/dev/sda1`, длинное имя или метку. Fedora сейчас использует как раз первый вариант. Впрочем, способ указания имени в зависимости от версии Fedora различен, — в более ранних версиях разделы идентифицировались как раз по последнему варианту (метка), а в 26-й версии, видимо, разработчики решили не усложнять пользователям жизнь и используют обычные имена. Имя устройства зависит и от разметки диска. Например, при использовании LVM имя `root`-устройства может выглядеть так:

```
root=/dev/mapper/fedora-root rd.lvm.lv=fedora/root rd.lvrn.lv=fedora/swap
```

Современные дистрибутивы производят загрузку в графическом режиме. Если ранее сначала выводились диагностические сообщения ядра, а затем сообщения системы инициализации, то сейчас вы едва успеете заметить сообщения ядра, как появится красивый графический индикатор, информирующий вас, сколько времени осталось до полной загрузки системы. Если вы предпочитаете видеть диагностиче-

ские сообщения, а не графический индикатор загрузки, тогда уберите параметр `rhgb` — это можно сделать в настройках загрузчика Linux (см. главу 21).

Параметр `quiet` вообще выключает сообщения, выводимые ядром, — если ядру передан этот параметр, сообщений ядра при загрузке системы вы вообще не увидите. Зато их можно потом получить командой `dmesg`.

Следующая серия значимых сообщений — инициализация первого процессора, точнее, первого ядра процессора:

```
[ 0.000000] Initializing CPU#0
[ 0.000000] allocated 8387584 bytes of pagecgroup
[ 0.000000] please try 'cgroup_disable=memory' option if you don't want memory cgroups
[ 0.000000] Initializing HighMem for node 0 (000373fe:0007ffd0)
```

И подробная информация об использовании памяти:

```
[ 0.000000] Memory: 2051560k/2096960k available (4189k kernel code, 44948k reserved,
2691k data, 592k init, 1191752k highmem)
```

Итак, у нас «всего» 2 096 960 Кбайт, из которых доступно 2 051 560 Кбайт, а в скобках приведен отчет о том, куда ядро израсходовало память: сколько килобайт зарезервировано для кода ядра, данных и т. д.

После этого вы увидите информацию о выделении виртуальной памяти ядра:

```
[ 0.000000] virtual kernel memory layout:
[ 0.000000] fixmap : 0xffa95000 — 0xfffff000 (5544 kB)
[ 0.000000] pkmap : 0xff400000 — 0xff800000 (4096 kB)
[ 0.000000] vmalloc : 0xf7bfe000 — 0xf0fe000 (120 MB)
[ 0.000000] lowmem : 0xc0000000 — 0xf73fe000 ( 883 MB)
[ 0.000000] .init: 0xc0ab9000 — 0xc0b4d000 ( 592 kB)
[ 0.000000] .data : 0xc081754a — 0xc0ab8200 (2691 kB)
[ 0.000000] .text: 0xc0400000 — 0xc081754a (4189 kB)
```

Пропустим пару бесполезных строк и перейдем к частоте процессора и значению `BogoMIPS`:

```
[ 0.000000] tsc: Detected 2194.599 MHz processor.
[ 0.000127] Calibrating delay loop (skipped) preset value.. 4389.19 BogoMIPS (lpj=2194599)
```

Из всего этого можно сделать вывод, что процессор у нас частотой 2194 МГц и производительностью в 4389,19 BogoMIPS.

BogoMIPS

Ядро вычисляет производительность процессора в так называемых BogoMIPS. Здесь MIPS — аббревиатура от Millions of Instructions Per Second, а Bogo — происходит от bogus (фальшивый, поддельный). Префикс Bogo ставит под сомнение актуальность вычисленной ядром величины, поэтому Линус Торвальдс (кстати, BogoMIPS — это его изобретение) и назвал ее *фальшивой*. В Интернете можно найти довольно точное определение BogoMIPS: «сколько миллионов раз в секунду процессор может ничего не делать». О производительности процессора по BogoMIPS можно судить лишь косвен-

но. Понятно, что чем производительнее процессор, тем больше будет сделано «пустых» операций, но одно дело, когда процессор просто ничего не делает, и совсем другое, когда он работает под «нагрузкой», т. е. выполняет арифметические и мультимедийные инструкции. Например, Duron 1,6 ГГц показывал результат в 3193,85 BogoMIPS, а Athlon X2 4200 — «всего» 4389,19, несмотря на большую частоту и два ядра. На практике же Athlon X2 намного быстрее, чем Duron 1,6 ГГц.

Сразу после вычисления бесполезных BogoMIPS инициализируется система контроля доступа SELinux. SELinux может быть или выключена, или работать в одном из двух режимов: принудительном (permissive) или режиме предупреждений. В нашем случае SELinux инициализируется и переходит в принудительный режим:

```
[ 0.004114] Security Framework initialized
```

```
[ 0.004126] SELinux: Initializing.
```

```
[ 0.004145] SELinux: Starting in permissive mode
```

СИСТЕМА УПРАВЛЕНИЯ ДОСТУПОМ SELINUX

Описание системы управления доступом SELinux вы найдете в папке *Дополнения* сопровождающего книгу электронного архива (см. *приложение*).

Еще через несколько строк вы увидите полное наименование процессора:

```
CPU0: AMD Athlon(tm) 64 X2 Dual Core Processor 4200+ stepping 02
```

Далее ядро Linux перейдет к инициализации второго процессора или второго ядра процессора (в зависимости от того, какая у вас машина: многопроцессорная или многоядерная):

```
[ 0.004999] Initializing CPU#1
```

```
[ 0.093074] NMI watchdog enabled, takes one hw-pmu counter.
```

```
[ 0.093103] Brought up 2 CPUs
```

```
[ 0.093106] Total of 2 processors activated (8778.70 BogoMIPS).
```

Потом будет выведена дата и время загрузки:

```
[ 0.095261] RTC time: 10:15:50, date: 01/13/12
```

А затем — строки, относящиеся к инициализации шины PCI и распределению PCI-ресурсов (их мы подробно рассматривать не станем).

Далее я позволю себе еще существеннее сократить представление вывода ядра, поскольку полный вывод занимает целых 18 страниц, и я не думаю, что вам будет интересна каждая его строчка. Вместо этого мы рассмотрим, как ядро предоставляет информацию о жестком диске, приводе DVD, мышке, клавиатуре и некоторых других устройствах.

Начнем с жесткого диска (устройство /dev/sda):

```
[ 1.641443] ata0.00: ATA-7: SAMSUNG HD251HJ, 1AC01113, max UDMA7
```

```
[ 1.641446] ata0.00: 488397168 sectors, multi 16: LBA48 NCQ (depth 31/32), AA
```

```
[ 1.641452] ata0.00: SB600 AHCI: limiting to 255 sectors per cmd
```

```
[ 1.647882] ata0.00: SB600 AHCI: limiting to 255 sectors per cmd
```

```
[ 1.647886] ata0.00: configured for UDMA/133
```

```
[ 1.648134] scsi 0:0:0:0: Direct-Access ATA SAMSUNG HD251HJ 1AC0 PQ: 0 ANSI: 5
[ 1.648334] sd 0:0:0:0: [sda] 488397168 512-byte logical blocks: (250 GB/232 GiB)
```

На компьютере установлен жесткий диск (SATA) небольшого размера— всего 250 Гбайт, производства Samsung. Ядро выводит полную информацию о диске, включая модель, режим работы и геометрию.

А вот информация о DVD-приводе LG (устройство /dev/sr0):

```
[ 2.577336] ata5: PATA max UDMA/100 cmd 0x1f0 ctl 0x3f6 bmdma 0xff00 irq 14
[ 2.577340] ata6: PATA max UDMA/100 cmd 0x170 ctl 0x376 bmdma 0xff08 irq 15
[ 2.731559] ata5.01: ATAPI: HL-DT-ST DVDROM GSA-4167B, DL11, max UDMA/33
[ 2.737449] ata5.01: configured for UDMA/33
[ 2.743264] scsi 4:0:1:0: CD-ROM HL-DT-ST DVDROM GSA-4167B DL11 PQ: 0 ANSI: 5
[ 2.747420] sr0: scsi3-mmc drive: 78x/78x writer dvd-ram cd/rw xa/form2 cdda tray
[ 2.747423] cdrom: Uniform CD-ROM driver Revision: 3.20
[ 2.747567] sr 4:0:1:0: Attached scsi CD-ROM sr0
```

Информация об USB-мышке выводится так:

```
[ 2.194399] input: USB Optical Mouse as /devices/pci0000:00/0000:00:13.2/usb4/4-2/4-2:1.0/input/input2
[ 2.194609] generic-usb 0003:1BCF:0007.0001: input,hiddev0,hidraw0: USB HID v1.10 Mouse [USB Optical Mouse] on usb-0000:00:13.2-2/input0
```

А вот клавиатура самая обычная (PS/2):

```
[ 2.187238] i8042: PNP: PS/2 Controller [PNP0303:PS2K] at 0x60,0x64 irq 1
[ 2.207257] input: AT Translated Set 2 keyboard as
/devices/platform/i8042/serio0/input/input3
```

Также может выводиться информация о различных периферийных устройствах — например, о Web-камере:

```
[ 58.516805] Linux video capture interface: v2.00
[ 58.518197] udevd[508]: renamed network interface eth0 to p6pl
[ 59.206723] uvcvideo: Found UVC 1.00 device Webcam C110 (046d:0829)
```

Информация о сетевой плате (устройство eth0):

```
[ 57.002826] r8169 Gigabit Ethernet driver 2.3LK-NAPI loaded
[ 57.002847] r8169 0000:02:00.0: PCI INT A -> GSI19 (level, low) -> IRQ 19
[ 57.002877] r8169 0000:02:00.0: setting latency timer to 64
[ 57.002935] r8169 0000:02:00.0: irq 42 for MSI/MSI-X
[ 57.004154] r8169 0000:02:00.0: ethO: RTL8168b/8111b at 0xf8410000, 00:19:db:c7:el:b5,
XID 18000000 IRQ 42
```

На устройство /dev/sda7 добавлен своп-раздел размером 530 104 Кбайт:

```
[ 99.788922] Adding 530108k swap on /dev/sda7. Priority:-1 extents:1 across:530108k
```

Далее выводится еще несколько не очень интересных сообщений, и управление передается системе инициализации Linux.

20.2. Параметры ядра

Параметры ядра позволяют управлять его поведением. Как уже говорилось, мы можем передать ядру параметры непосредственно при загрузке, используя меню загрузчика, или же прописать параметры ядра в файлах конфигурации загрузчика. Первый случай подходит для «одноразового» использования того или иного параметра, а второй — если параметр нужен для корректной работы системы. Поэтому, чтобы не указывать его каждый раз при загрузке Linux, намного проще прописать его в файле конфигурации загрузчика.

Если вы используете загрузчик GRUB/GRUB2, то передать параметры ядру можно так: сначала надо выбрать образ (метку), а затем нажать клавишу <e> — появится поле, в котором вы можете отредактировать параметры ядра, указанные в файлах конфигурации загрузчика.

Внешний вид меню загрузчика GRUB/GRUB2 определяется его конфигурацией: на рис. 20.1, а показано, как выглядит меню загрузчика GRUB2 в Ubuntu 17.04, а на рис. 20.1, б — в Fedora 26. Как видите, в разных дистрибутивах меню может выглядеть по-разному.

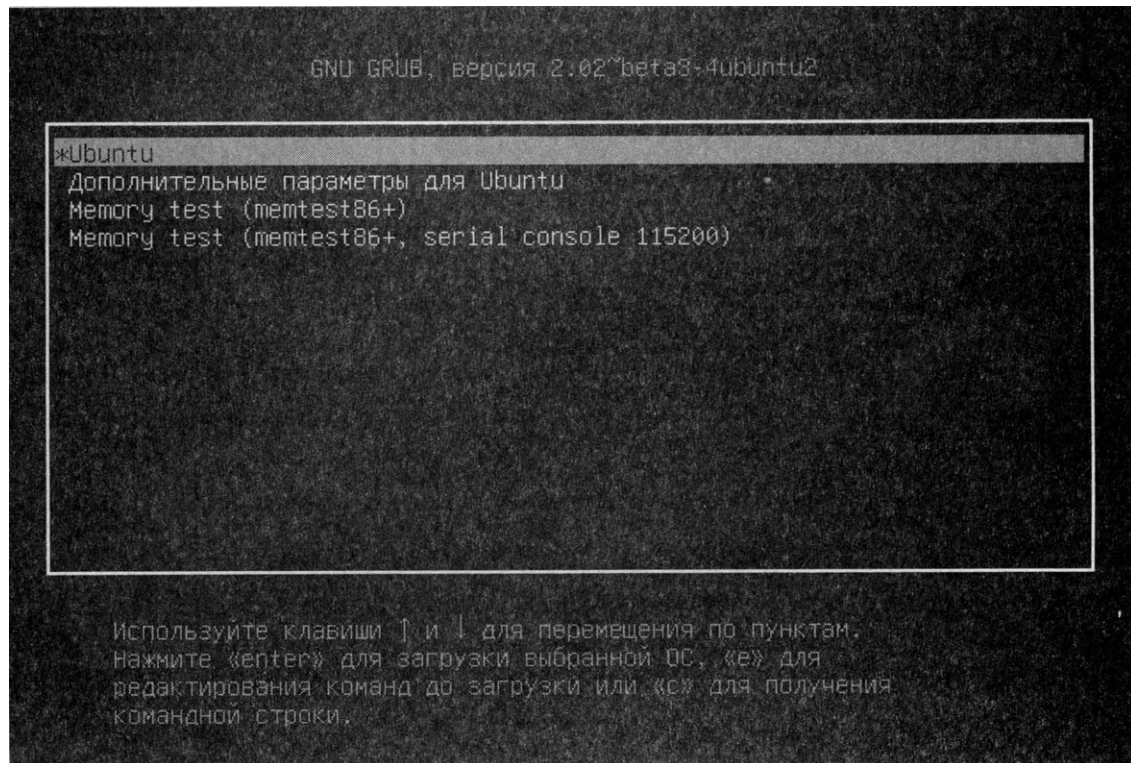


Рис. 20.1. Меню загрузчика GRUB2: а — в Ubuntu 17.04

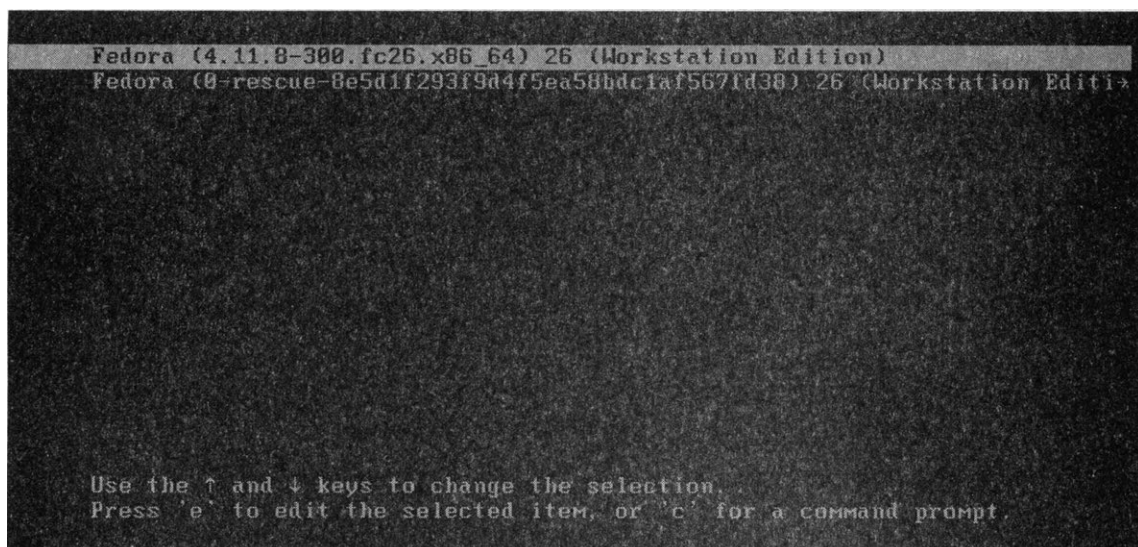


Рис. 20.1. Меню загрузчика GRUB2: б — в Fedora 26

На рис. 20.2, а показан процесс редактирования параметров ядра в Ubuntu. Параметры ядра указываются после служебного слова **linux** — вы можете добавить свои собственные параметры или же отредактировать имеющиеся. На рис. 20.2, б показан процесс редактирования параметров ядра в Fedora 26. Принцип, как видите, тот же, но используется другая версия ядра.

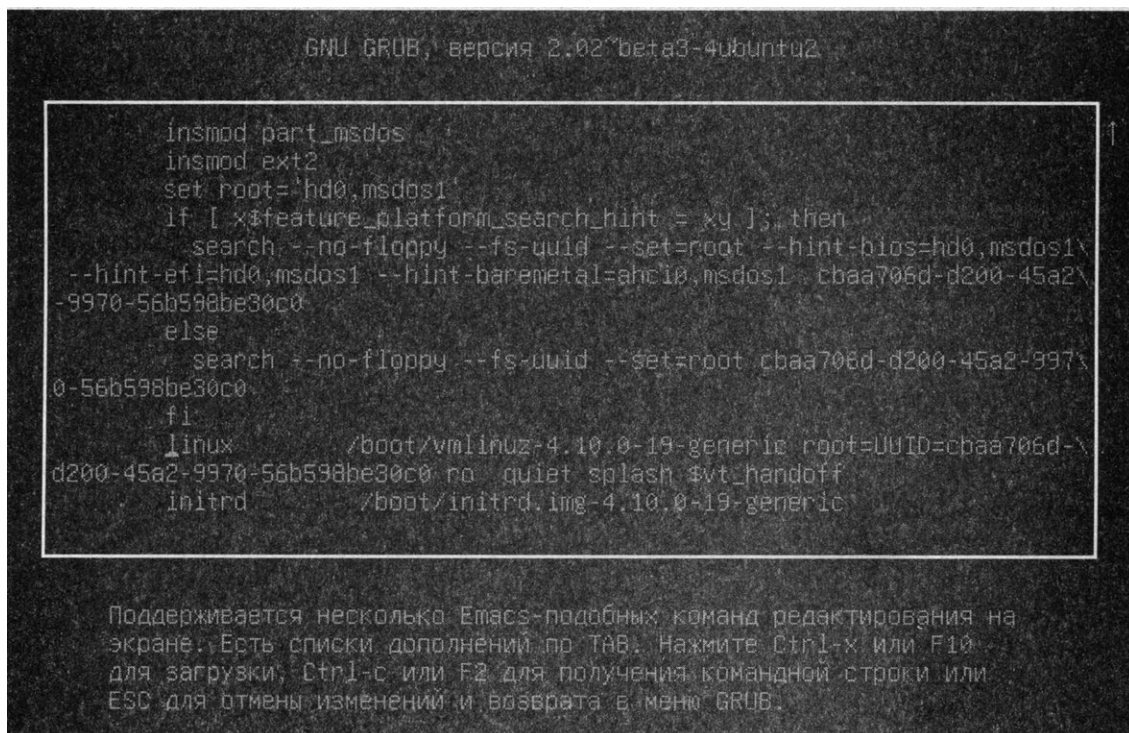


Рис. 20.2. Редактирование параметров ядра: а — в Ubuntu 17.04

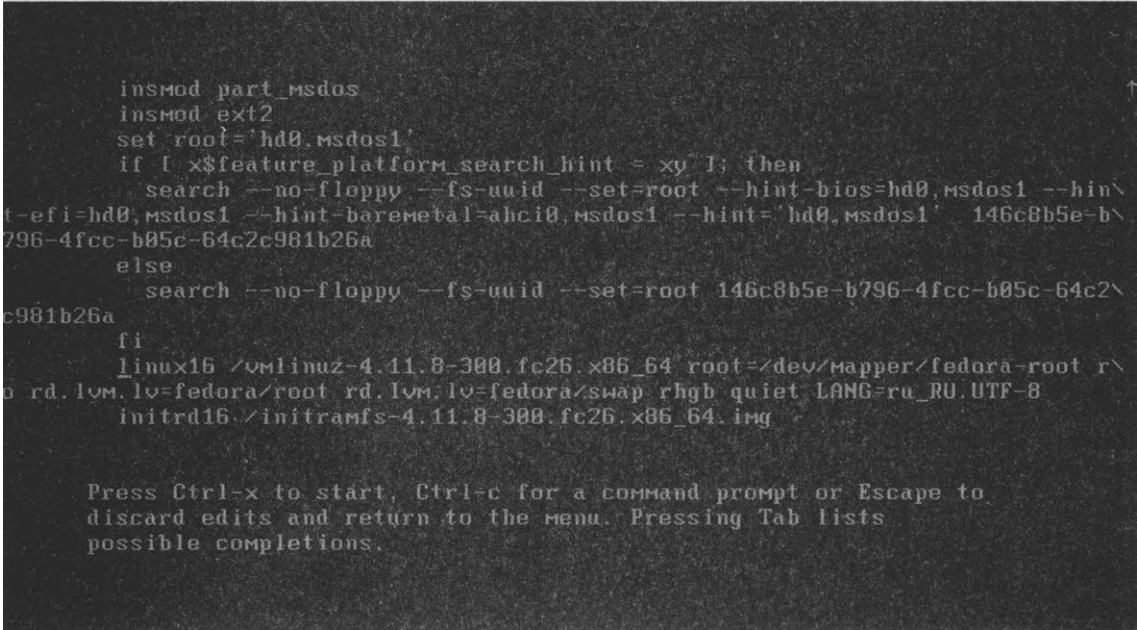


Рис. 20.2. Редактирование параметров ядра: б— в Fedora 26

После редактирования параметров ядра для продолжения загрузки нужно нажать клавишу <F10> или комбинацию клавиш <Ctrl>+<X>. Помните, что переданные таким образом ядру параметры не сохраняются. О том, как сохранить параметры ядра в файле конфигурации загрузчика, мы поговорим в *главе 21*.

Параметров ядра очень много, и чтобы не перегружать вас излишней информацией, в табл. 20.1 собраны самые полезные.

Таблица 20.1. Некоторые параметры ядра Linux

Параметр	Описание
root=устройство	Позволяет указать корневую файловую систему. Например, root=/dev/sda5
rootdelay=N	Ждать N секунд перед монтированием корневой файловой системы
rootflags=флаги	Задаёт флаги корневой файловой системы (см. главу 4)
rootfstype=тип	Задаёт тип корневой файловой системы. Полезен, если не удалось определить автоматически
rootwait	Ожидание корневой файловой системы. Ядро будет ждать, пока не появится устройство с корневой файловой системой. Полезно, когда корневая файловая система расположена на съёмном носителе, например на флешке
ro	Монтирует корневую файловую систему в режиме "только чтение". Используется по умолчанию. После проверки файловой системы программой fsck корневая файловая система перемонтируется в режим rw

Таблица 20.1 (продолжение)

Параметр	Описание
<code>rw</code>	Монтирует корневую файловую систему в режиме «чтение/запись». При использовании этого параметра нельзя запускать программы типа <code>fsck</code> . Перед запуском <code>fsck</code> нужно перемонтировать корневую файловую систему в режиме <code>ro</code>
<code>mem=</code>	<p>Определяет объем памяти, установленной в компьютере. Иногда ядро неправильно определяет объем оперативной памяти. Вы можете помочь ему в этом, указав параметр <code>mem</code>. Только указывать его нужно правильно, например:</p> <pre>mem=768M</pre> <p>После числа обязательно должна следовать буква <code>M</code>, иначе ядро «подумает», что объем оперативной памяти 768 байтов.</p> <p>В современных дистрибутивах дела с памятью обстоят лучше. Скорее всего, параметр <code>mem</code> указывать вы не будете, но есть шанс столкнуться с иной неприятной ситуацией. Компьютерная индустрия не стоит на месте, и вы можете купить компьютер с оперативной памятью более 4 Гбайт, а потом обнаружить, что ваш дистрибутив видит только первые 4 Гбайт. В этом случае вам нужно перекомпилировать ядро с поддержкой PAE (Physical Address Extension) или же установить ядро, изначально поддерживающее PAE</p>
<code>init =</code>	Позволяет задать программу инициализации. По умолчанию используется программа <code>/sbin/init</code> , но вы можете задать другую
<code>reboot=</code>	Позволяет задать тип перезагрузки компьютера. Возможные значения: <code>cold</code> и <code>warm</code> , т. е. «холодная» или «горячая» перезагрузка
<code>single</code>	Однопользовательский режим для администрирования системы — например, в случае отказа
<code>nodmraid</code>	Отключает программные RAID-массивы, организованные на уровне BIOS
<code>noapic</code>	Полезен, если вы при загрузке увидите сообщение: kernel panic — not syncing: IO-APIC + timer doesn't work! Подробнее об этом параметре вы можете прочитать по адресу: http://www.dkms.org.ua/phpbb2/viewtopic.php?topic=2973&forum=5
<code>no pcmcia</code>	Отключает PCMCIA-карты (для ноутбуков). Полезен, если вы подозреваете, что у вас проблемы с PCMCIA-картой
<code>nodma</code>	Отключается DMA (Direct Memory Access, прямой доступ к памяти) для всех IDE-устройств
<code>noapm</code>	Отключает APM (Advanced Power Management) — расширенное управление питанием
<code>nousb</code>	Отключает поддержку USB
<code>noscsi</code>	Отключает поддержку SCSI
<code>pci=noacpi</code>	Не использовать ACPI для управления PCI-прерываниями
<code>acpi=off</code>	Полностью отключает ACPI (Advanced Configuration and Power Interface). Полезен на некоторых ноутбуках, когда не удается установить (а потом загрузить) Linux

Таблица 20.1 (окончание)

Параметр	Описание
edd=off	Отключает EDD (Enhanced Disk Drive). Если при загрузке Linux вы видите сообщение Probing EDD , и загрузка на этом останавливается, тогда вам поможет параметр ядра <code>edd=off</code>
boot delay=N	Сообщения ядра выводятся так быстро, что вы не успеваете их прочитать? С помощью этого параметра вы можете установить задержку в N секунд перед выводом следующего сообщения ядра
elevator=планировщик	Позволяет выбрать планировщик ввода/вывода. Подробно о нем мы поговорим в <i>главе 30</i>
vga=режим	Позволяет задать VGA-режим. Подробнее см. файл Documentation/svga.txt. Можно также задать значение ask, чтобы ядро спросило, какой режим нужно использовать: <code>vga=ask</code>
quiet	«Тихий» режим, отключает большинство сообщений ядра

ДОПОЛНИТЕЛЬНЫЕ ПАРАМЕТРЫ ЯДРА

Повторю: у ядра очень много параметров, и нет смысла приводить здесь их все. Параметры, с которыми, возможно, вам доведется столкнуться на практике, представлены в табл. 20.1. С дополнительными параметрами ядра вы можете ознакомиться по адресу: <http://www.mjmwired.net/kernel/Documentation/kernel-parameters.txt>.

20.3. Компиляция ядра в дистрибутиве Ubuntu

Linux, в отличие от многих других операционных систем, позволяет обычному пользователю проникнуть в святая святых — в собственное ядро. Любой желающий может загрузить исходные коды ядра и откомпилировать ядро операционной системы.

Вообще, перекомпиляция ядра — весьма специфическая операция. Раньше ее приходилось делать довольно часто — практически каждый Linux-пользователь со стажем хотя бы раз в жизни перекомпилировал ядро. Зачем? Например, чтобы включить дополнительные функции. Или наоборот, выключить поддержку некоторых устройств и некоторые ненужные функции — так ядро окажется компактнее, и система будет работать быстрее.

Сейчас я уже и не знаю, зачем может понадобиться перекомпиляция ядра. Это настолько в наше время редкая операция, что даже исходные коды ядра перестали поставляться на дистрибутивных дисках, — исходники ядра теперь можно установить из репозитория дистрибутива или же скачать с сайта www.kernel.org.

Далее в этом разделе будет рассмотрена сборка ядра на примере дистрибутива Ubuntu 17.04. Но прежде чем продолжить, хочу отметить, что для компиляции ядра вам понадобится довольно много дискового пространства. Во-первых, нужно установить дополнительное программное обеспечение (1,65 Гбайт). Во-вторых, скачать

сами исходные тексты ядра, которые в распакованном виде занимают немало. В-третьих, в процессе компиляции создается множество файлов, которые и занимают львиную долю используемого объема. Поэтому после завершения компиляции ядра не забудьте ввести команду `make clean`— для очистки дискового пространства.

20.3.1. Установка дополнительных пакетов

Первым делом нужно установить пакеты, которые нам понадобятся для компиляции ядра (рис. 20.3):

```
$ sudo apt-get install git build-essential ncurses-base ncurses-dev fakeroot
kernel-package xz-utils
```

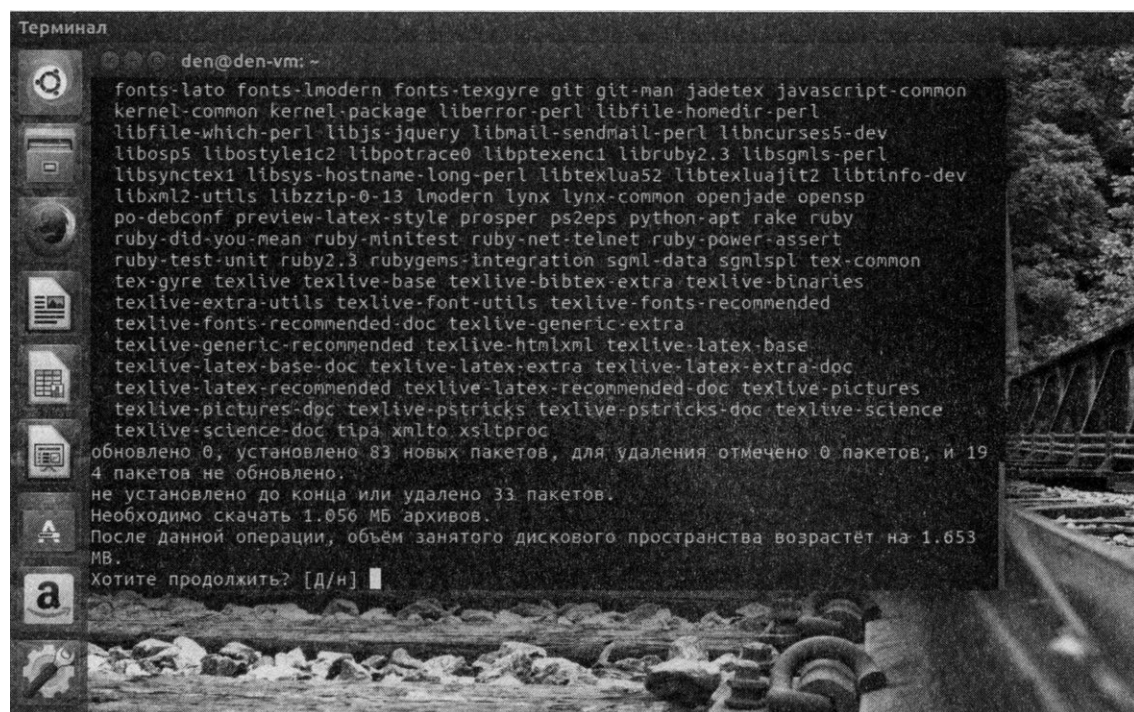


Рис. 20.3. Ubuntu 17.04: установка дополнительного программного обеспечения

20.3.2. Загрузка исходных текстов ядра

Затем надо загрузить исходные тексты ядра. Для этого введите команду:

```
$ wget https://cdn.kernel.org/pub/linux/kernel/v4.x/linux-4.12.8.tar.xz
```

На момент написания этих строк версия ядра 4.12.8 была последней, хотя, возможно, уже появилась и новая версия. Просто откройте браузер и просмотрите содержимое каталога <https://cdn.kernel.org/pub/linux/kernel/v4.x/>. Впрочем, если вы сделаете, как здесь предлагается, то все равно установите более новую версию ядра, чем 3.19, которую предлагает Ubuntu 17.04 по умолчанию.

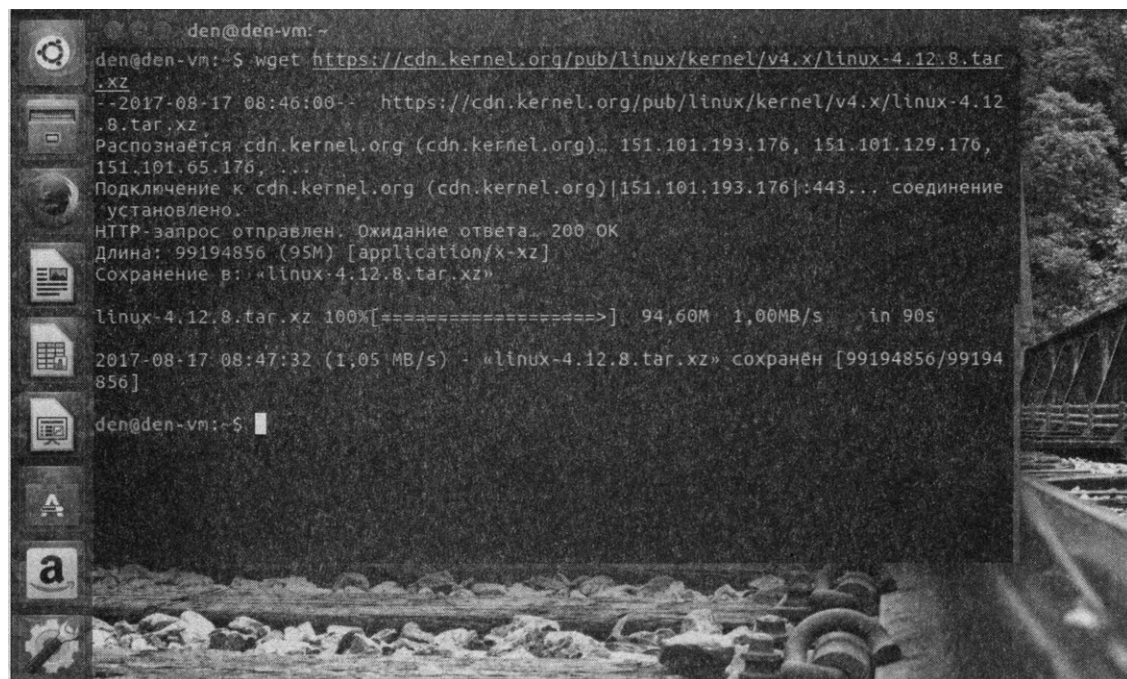


Рис. 20.4. Ubuntu 17.04: загрузка исходников ядра

После загрузки архива нужно его распаковать:

```
$ tar xvf linux-4.12.8.tar.xz
$ cd linux-4.12.8
```

20.3.3. Настройка ядра

Настало время настроить ядро — создать свою собственную его конфигурацию. Ведь вы перекомпилируете ядро не просто так — наверняка вам нужно включить дополнительные возможности или же, наоборот, отключить какие-либо функции, чтобы сделать ядро компактнее. Впрочем, если вы хотите откомпилировать ядро эксперимента ради, путеводителем вам послужит табл. 20.2, в которой описаны основные разделы опций ядра.

Итак, поскольку вы уже последней выполненной командой перешли в каталог `linux-4.12.8`, введите теперь команды:

```
$ cp /boot/config-$(uname -r) .config
$ make menuconfig
```

Первая команда копирует текущую конфигурацию ядра в файл `.config`, а вторая — запускает конфигуратор ядра. Существует и графическая версия конфигулятора, но, на мой взгляд, `make menuconfig` — наиболее удобный (рис. 20.5).

Меню конфигулятора содержит как разделы с опциями (куда можно попасть, нажав клавишу с подсвеченной в меню буквой или клавишу `<Enter>` на выделенном разделе), так и отдельные опции конфигурации. Назначение разделов и опций корневого раздела конфигулятора ядра представлены в табл. 20.2.

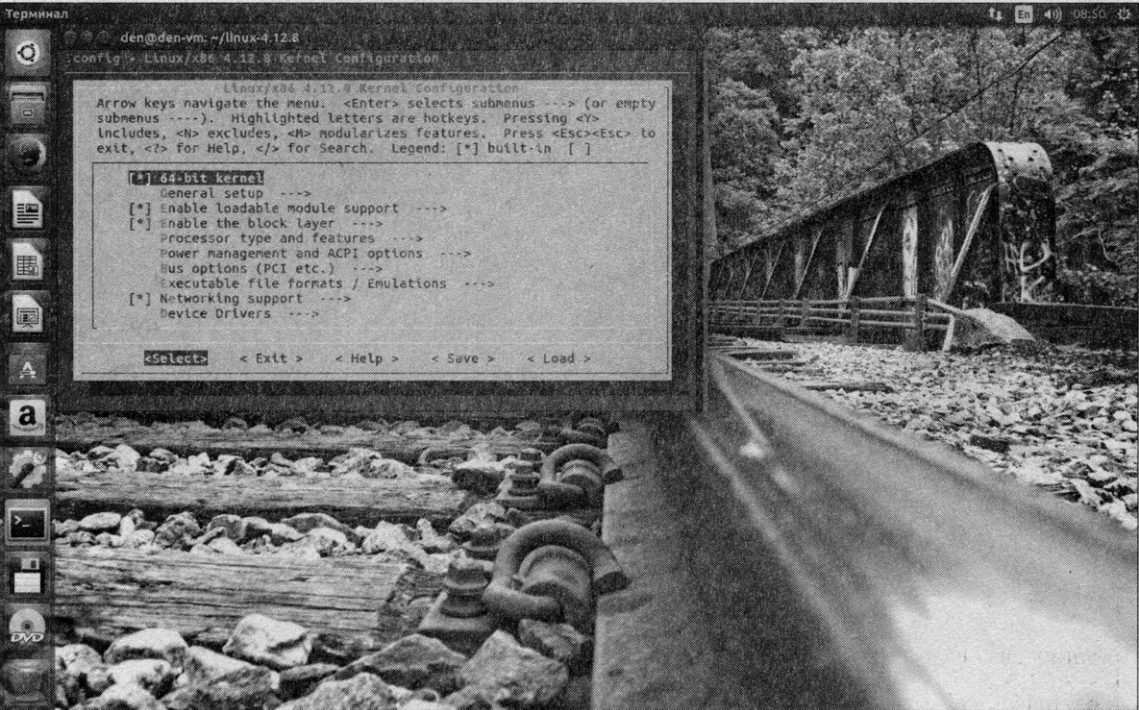


Рис. 20.5. Ubuntu 17.04: конфигуратор make menuconfig

Таблица 20.2. Обзор корневого раздела конфигуратора ядра

Раздел/опция	Описание
64-bit kernel	Если включена, скомпилированное ядро будет 64-битным
General setup	Общие параметры — например, поддержка своп-памяти, межпроцессного взаимодействия System V, Sysctl. Если не знаете, для чего нужна та или иная опция, выделите ее и нажмите клавишу <F1>. А уж если не знаете английского, то до его изучения лучше опции не выключать!
Enable loadable module support	Поддержка загружаемых модулей. Драйверы устройств в Linux разработаны в виде модулей ядра. Здесь вы можете указать, нужна ли вам поддержка модулей. Отключать поддержку модулей на обычных машинах не рекомендуется. Если же вы хотите построить мало обслуживаемый сервер, работающий по принципу «построил и забыл», отключение поддержки загружаемых модулей позволит даже повысить безопасность сервера, поскольку злоумышленник не сможет добавить свой код в ядро путем загрузки модуля. Однако в этом случае ядро будет очень громоздким, потому что вам придется все нужные вам функции, которые могли бы быть реализованы в виде модулей, компилировать в ядро
Enable the block layer	В этом разделе вы можете включить поддержку больших блочных устройств размером более 2 Тбайт
Processor type and features	Здесь вы можете выбрать тип вашего процессора и включить/выключить различные функции процессора

Таблица 20.2 (окончание)

Раздел/опция	Описание
Power management and ACPI options	Опции управления питанием (ACPI, APM)
Bus options	Здесь вы можете включить/выключить поддержку различных системных шин, а также определить их функции
Executable file formats/Emulations	Параметры поддержки форматов исполнимых файлов
Networking support	Сетевые опции ядра
Device drivers	Драйверы устройств. Здесь вы можете определить, какие устройства должна поддерживать ваша система, а какие — нет
Firmware drivers	Драйверы микропрограммного обеспечения (поддержка различных BIOS)
File systems	Здесь вы можете определить, какие файловые системы должна поддерживать ваша система, а какие — нет
Kernel Hacking	Различные параметры, относящиеся непосредственно к ядру
Security options	Параметры безопасности
Cryptographic API	Параметры криптографии (поддержка различных алгоритмов шифрования данных)
Virtualization	Параметры виртуализации
Library routines	Поддержка различных библиотечных функций (но если заглянуть в этот раздел, то вы увидите, что все эти функции связаны с вычислением контрольной суммы CRC)

Опции ядра могут быть либо включены, либо выключены. Если опция выключена, то ее код исключается из ядра (не будет учитываться при его компиляции), а если — включена, то код ее будет включен в состав ядра. Но есть еще третье состояние опции — М. Это означает, что опция будет включена в ядро как *модуль*. После сборки ядра и модулей все опции, скомпилированные в режиме М, будут «лежать» на диске, пока не понадобятся ядру. А как только это произойдет, нужный модуль будет загружен.

Для включения той или иной опции нужно выделить ее и нажать клавишу <Y>, для отключения — выделить и нажать клавишу <N>. Если опция нужна как модуль (как модуль можно включить не все опции), нажмите клавишу <M>.

При выходе из конфигулятора он спросит вас, хотите ли вы сохранить изменения в конфигурации ядра (рис. 20.6)? Конечно, хотим!

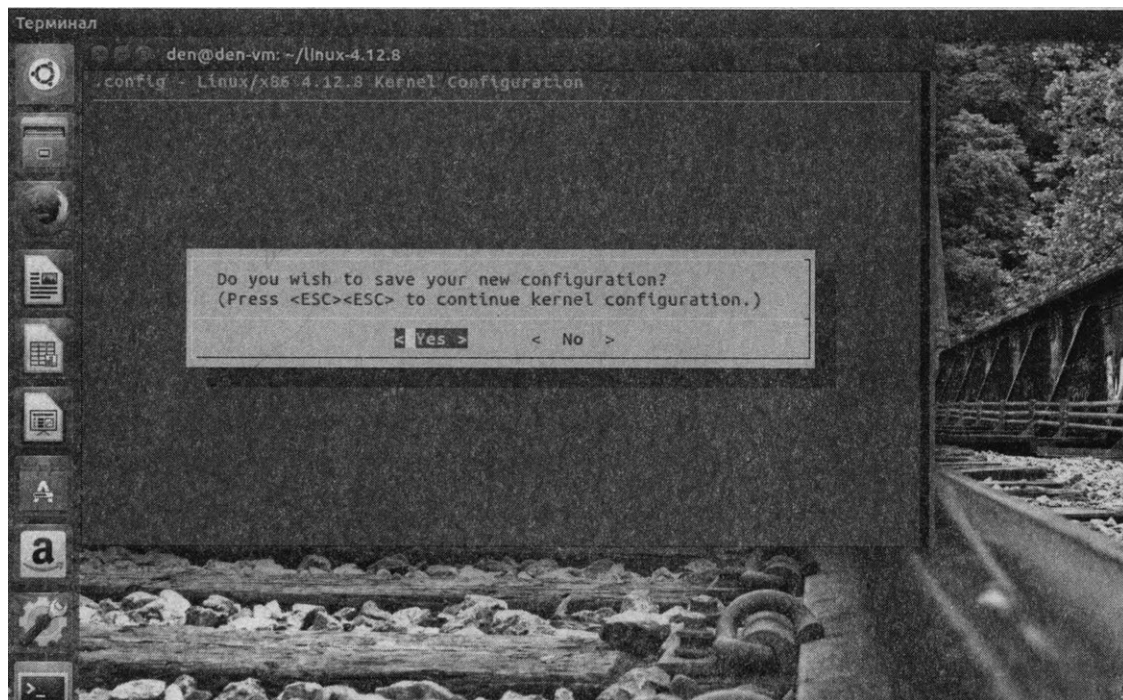


Рис. 20.6. Ubuntu 17.04: сохранить изменения в конфигурации ядра?

20.3.4. Компиляция ядра

После настройки ядра конфигуратор сообщит, что для построения ядра нужно ввести команду `make` (рис. 20.7), а для вывода справки — `make help`.

Спешить с вводом команды `make` мы пока не будем — это можно сделать всегда. Гораздо правильнее сначала очистить дерево исходного кода и сбросить параметры `kernel-package`, а затем собрать ядро, используя команду `fakeroot`, что позволит откомпилировать ядро от имени обычного пользователя, а не `root` (рис. 20.8 и 20.9):

```
$ make-kpkg clean
```

```
$ fakeroot make-kpkg --initrd --revision=1.0.DEN kernel_image kernel_headers
```

Разберемся, какие параметры мы передаем команде `make-kpkg` (именно она компилирует ядро):

- `--initrd` — создает `initrd`-образ;
- `--revision` — версия вашего ядра (можете указать здесь все, что вам хочется);
- `kernel_image` — создает Debian-пакет, содержащий образ ядра и все модули, сконфигурированные в файле `.config` (файл конфигурации ядра, созданный командой `make menuconfig`);
- `kernel_headers` — создает Debian-пакет, содержащий образ заголовков ядра Linux.

Позволю себе еще несколько замечаний относительно предлагаемого мною решения:

- мы не просто компилируем ядро на этой машине, как нам предложил конфигуратор (если бы мы ввели команду `make`), а создаем пакет с ядром, который может

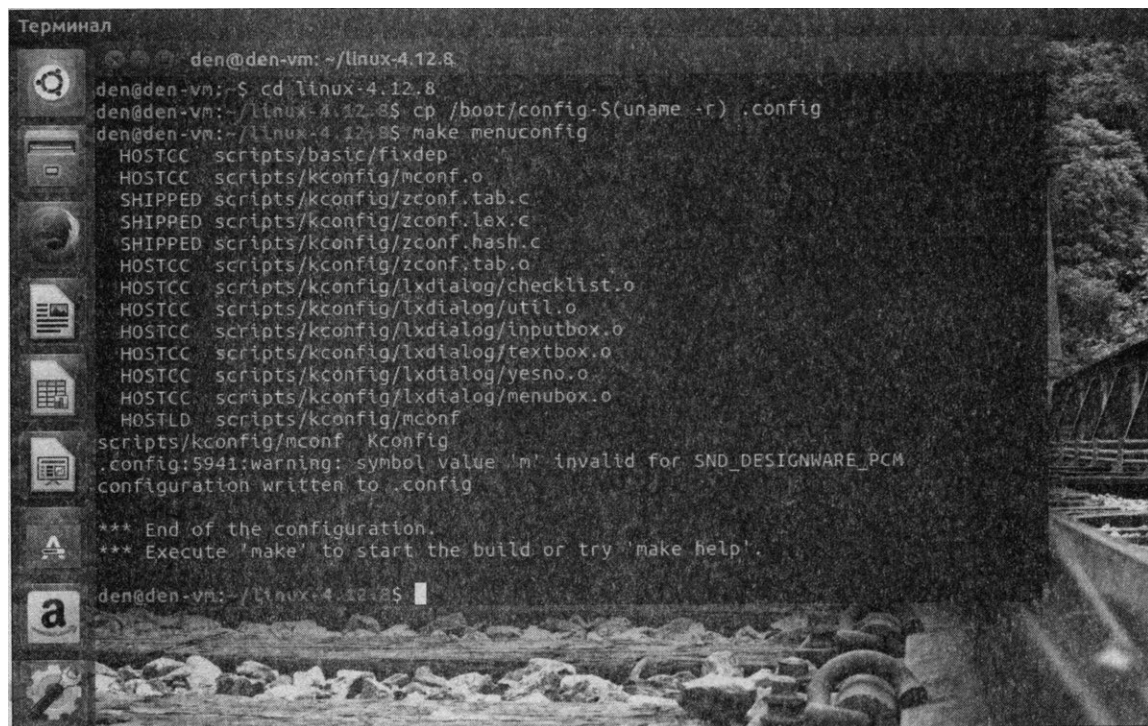


Рис. 20.7. Конфигурация сохранена

быть установлен на нескольких однотипных машинах, где нужно такое же ядро. Это существенно экономит время, поскольку не придется «собирать» ядро на каждой из машин;

- мы используем команду `fakeroot`, чтобы откомпилировать ядро от имени обычного пользователя, а не `root`. Если вы заметили, мы также не задействуем каталог `/usr/src/linux`, как требовалось ранее. Все действия происходят в домашнем каталоге пользователя, поэтому собрать собственное ядро может любой пользователь, и для этого ему не нужны права `root`, и он даже не должен быть вписан в файл `sudoers`. По сути, права `root` понадобятся вам только при установке полученных пакетов.

Время, необходимое для сборки ядра, зависит от производительности компьютера и конфигурации ядра. Так, на четырехъядерной машине с 4 Гбайт оперативной памяти компиляция ядра заняла около двух часов, а на двухъядерной машине с 2 Гбайт оперативки — примерно 4,5 часа. Если машина слабее, то процедура эта, соответственно, займет больше времени. Так что, запасайтесь терпением. В любом случае, у вас есть как минимум час свободного времени, чтобы заняться чем-либо полезным.

По окончании процесса компиляции в вашем домашнем каталоге будет создано два Debian-пакета: `linux-headers` и `linux-image`. Точное название этих пакетов зависит от версии ядра, архитектуры и указанного названия релиза.

Файлы получились довольно большими: пакет с ядром (`linux-image`) — 42 Мбайт, а пакет с заголовками (`linux-headers`) — 770 Мбайт.

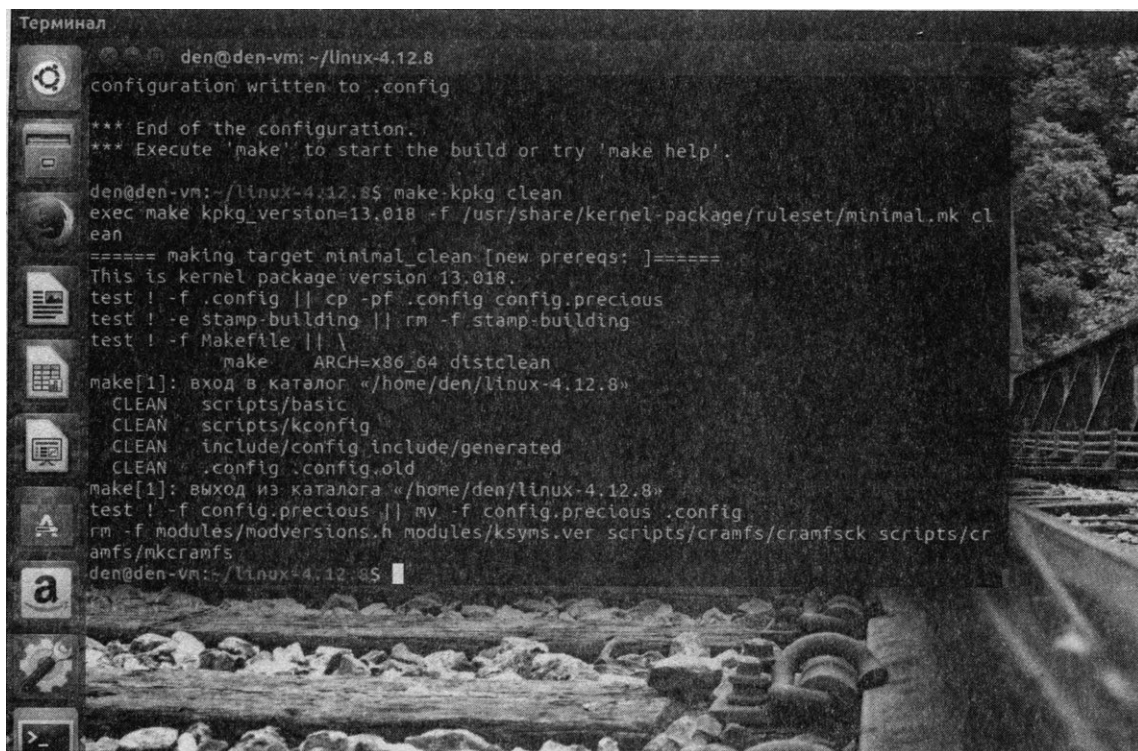


Рис. 20.8. Ubuntu 17.04: команда make-kpkg clean в действии

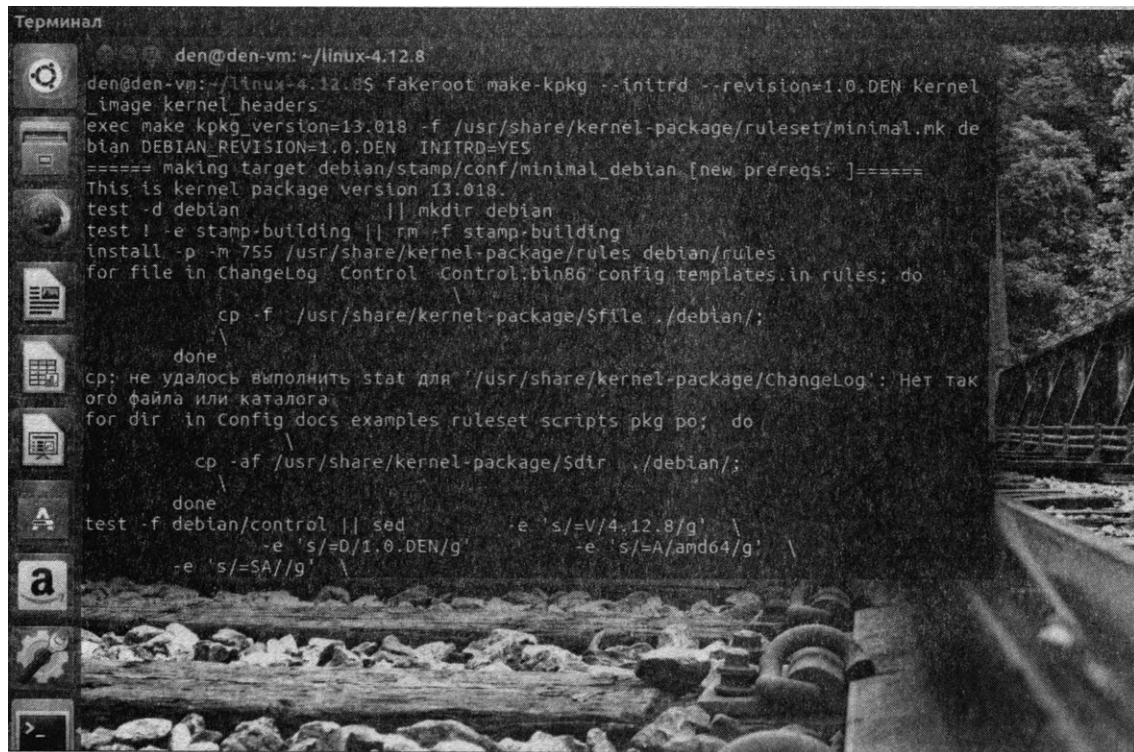


Рис. 20.9. Ubuntu: компиляция ядра

Сначала нужно установить пакет `linux-headers`, а затем — `linux-image`:

```
$ cd ~  
$ sudo dpkg -i linux-headers-4.12.8_1.0.DEN_i386.deb  
$ sudo dpkg -i linux-image-4.12.8_1.0.DEN_i386.deb
```

После чего перезагрузить компьютер:

```
$ reboot
```

Чтобы убедиться, что загрузилось именно скомпилированное ядро, введите команду:

```
$ uname -a
```

В выводе команды должна присутствовать примерно такая строка:

```
Linux den-vm 4.12.8-1.0.DEN #1 SMP Thu Aug 17 23:33:44 UTC 2017 i86_x64 GNU/Linux
```

Поздравляю! Вы успешно справились с перекомпиляцией ядра.

И в завершение этого раздела мне бы хотелось вернуться к разговору о дисковом пространстве, необходимом для компиляции ядра. Ранее было сказано, что для сборки ядра его потребуется «довольно много». На самом деле, если до компиляции ядра объем занятого на диске пространства составлял 5,5 Гбайт, то после компиляции эта величина возросла до 18 Гбайт,— получается, что 12,5 Гбайт понадобилось на компиляцию ядра, и из них только 1,65 Гбайт — на дополнительные пакеты.

Соответственно, файловая система `/home` должна содержать примерно 10,9 Гбайт доступного пространства:

18 Гбайт (после компиляции) - 5,5 Гбайт (до компиляции) - 1,65 Гбайт (пакеты,
установленные в корневой каталог) = (примерно) 10,85 Гбайт

и все это дисковое пространство должно быть доступно в вашем домашнем каталоге.

Именно поэтому после компиляции ядра для освобождения дискового пространства вам нужно перейти в каталог `linux-4.12.8` и ввести команду:

```
$ make-kpkg clean
```

Эта команда очищает дерево исходного кода от скомпилированных файлов — они нам более не нужны, поскольку уже включены в состав созданных пакетов. После очистки дерева исходников используется всего 9 Гбайт, но никак не 18.

20.4. RT-ядро

Помню, как-то в качестве операционной системы установил QNX — систему реального времени (RT, Real Time). Работала она очень быстро, еще бы — реакции от компьютера можно было ожидать в реальном времени. Существует очень простой способ превратить Linux в такую операционную систему, и есть надежда, что после этого Linux будет *реагировать в предсказуемое время на появление непредсказуемых событий* — это одно из определений системы реального времени.

Что такое система реального времени, мы разбираться здесь и сейчас не станем. Если вы про нее ничего не знаете, и не уверены, нужна ли она вам, посмотрите в Википедии:

http://ru.wikipedia.org/wiki/Операционная_система_реального_времени.

Я лишь расскажу, как превратить в систему реального времени Linux. Оказывается, все очень просто. Запустите менеджер пакетов и найдите RT-ядро. В зависимости от дистрибутива пакет с ядром реального времени может называться по-разному: например, в ALT Linux он называется `kemel-image-rt-up`, в Ubuntu — `linux-image-rt`. Просто установите этот пакет и перезагрузите компьютер, а при перезагрузке выберите новое ядро.

Если результат вас не впечатлил, тогда загрузите исходный код ядра и перекомпилируйте его. При компиляции ядра в `menuconfig` включите следующие опции:

- ☐ General setup · Choose SLAB allocator;
- ☐ Block layer -- Default I/O scheduler (CFQ);
- ☐ Processor type and - [*] Tickless System (Dynamic Ticks);
- ☐ Processor type and features RCU implementation type: (Preemptible RCU);
- ☐ Processor type and features Preemption Mode (Complete Preemption (Real-Time)).

Если вы уже настраивали ядро, то наверняка все эти опции видели. Правда, от системы с RT-ядром вы не всегда получите желаемый результат. Мне приходилось видеть в Интернете отзывы, что система после установки этого ядра «тормозила» еще больше, чем с обычным ядром. На чудо надеяться тоже не нужно — однопроцессорная машина не заработает под управлением RT-ядра быстрее.

20.5. Особенности компиляции ядра в других дистрибутивах Linux

В этой главе мы описали сборку собственного ядра в дистрибутиве Ubuntu. Инструкции, описанные здесь, действительны для любого дистрибутива, но есть некоторые нюансы.

Например, в Gentoo в каталоге `/boot` не появятся файлы `initrd` (диск в памяти, т. е. RAM Disk), пока вы не введете команду:

```
# genkernel initrd
```

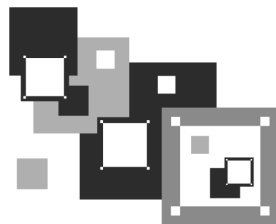
Подробно процесс сборки ядра в этом дистрибутиве описан на следующей странице (кстати, на русском языке): **<http://www.gentoo.org/doc/ru/genkernel.xml>**.

С особенностями компиляции ядра в Fedora можно познакомиться по адресу: **http://fedoraproject.org/wiki/Building_a_custom_kernel/ru**.

Особенности сборки ядра в openSUSE описаны на страницах:

- ☐ **http://ru.opensuse.org/How_To_Compile_A_Kernel_-_The_SuSE_Way**;
- ☐ **<http://www.bloged.org/2008/10/opensuse-11.html>**.

ГЛАВА 21



Загрузчики Linux

21.1. Основные загрузчики

Главное назначение загрузчика — запуск выбранной пользователем операционной системы. Наиболее популярными загрузчиками сейчас являются GRUB и GRUB2, которые мы здесь подробно рассмотрим.

ЗАГРУЗЧИК LILO

В старых дистрибутивах по умолчанию использовался загрузчик LILO, описание которого из этого издания исключено. Впрочем, если он вам зачем-то нужен, информацию о нем всегда можно найти в Интернете. Кроме того, описывающий загрузчик LILO (раздел главы 21 из 2-го издания книги) вы найдете в папке *Дополнения* сопровождающего книгу электронного архива (см. приложение).

Кроме LILO и GRUB/GRUB2 некоторые дистрибутивы могут включать собственные загрузчики — например, в ASPLinux использовался ASPLoader. Подобные загрузчики мы также рассматривать не станем, поскольку в большинстве случаев в дистрибутивах, использующих собственные загрузчики, имеется возможность установки GRUB/GRUB2 или того же LILO.

Время не стоит на месте. В свое время загрузчик GRUB (GRand Unified Bootloader) пришел на смену LILO, поскольку последний не поддерживал загрузку с разделов, начинающихся после 1024-го цилиндра. Об этой проблеме знал тогда, наверное, каждый Linux-пользователь, — ведь всего несколько лет назад, пока все дистрибутивы не перешли на GRUB, она была весьма актуальной. Загрузчик GRUB оказался также более гибким, чем LILO, — благодаря иной схеме загрузки операционных систем GRUB «понимал» больше файловых систем, нежели LILO, а именно: FAT/FAT32, ext2, ext3, ReiserFS, XFS, BSDFS и др.

Точно такая же участь постигла и GRUB — на его место пришел GRUB2, умеющий загружаться с файловой системы ext4, что просто-таки необходимо современному дистрибутиву. GRUB2 — это не просто набор патчей для GRUB, а полностью новая разработка, созданная с «нуля». Именно поэтому у GRUB2 совершенно другой формат конфигурационного файла (хотя лучше бы оставили старый).

Разработка загрузчика GRUB сейчас полностью прекращена, к нему выпускаются лишь патчи. Впрочем, даже в 2017 году можно установить обычный GRUB, если

очень хочется, — он присутствует в составе популярных дистрибутивов, хотя по умолчанию в них используется GRUB2.

21.2. Конфигурационные файлы GRUB и GRUB2

21.2.1. Конфигурационный файл GRUB

Конфигурационным файлом GRUB является файл `/boot/grub/grub.conf` (в старых версиях: `/boot/grub/menu.lst` (впрочем, `menu.1st` в новых версиях — это ссылка на файл `grub.conf`). Рассмотрим пример этого файла (листинг 21.1).

РЕДАКТИРОВАНИЕ КОНФИГУРАЦИОННЫХ ФАЙЛОВ ЗАГРУЗЧИКА

Думаю, не стоит и говорить о том, что конфигурационные файлы загрузчика нужно редактировать только с правами `root`. В некоторых дистрибутивах конфигурационный файл `grub.conf` обычный пользователь не может даже просмотреть. И это правильно, поскольку в этом файле могут содержаться незашифрованные (если администратор поленился их зашифровать) пароли загрузчика.

Листинг 21.1. Файл `/boot/grub/grub.conf`

```
# Следующие параметры будут описаны далее:
boot=/dev/hda
default=0
timeout=10
fallbacks
splashimage= (hd0, 1) /grub/mysplash.xpm.gz

# по умолчанию скрывает меню (для того чтобы увидеть меню,
# нужно нажать <ESC>)
#hiddenmenu

# Главное загрузочное устройство GRUB (можно не указывать)
#groot=(hd0, 1)

# Опции загрузчика по умолчанию (более подробно см. man menu.lst)
# defoptions=quiet splash

# опции ядра по умолчанию
# kopt=root=/dev/hda2 ro

# Предпочитаемые цвета
#color cyan/blue white/blue

title MDK
    root (hd0,1)
```

```
kernel /vmlinuz-2.6.14-1.1263 ro root=/dev/hda2
initrd /initrd-2.6.14-1.1263.img
```

```
title WinXP
    rootnoverify (hd0,0)
    makeactive
    chainloader+1
```

РАЗЛИЧИЯ В ИМЕНАХ УСТРОЙСТВ

Как вы уже успели заметить, в листинге 21.1 (параметр `boot` и далее) до сих пор используются устаревшие номера устройств `/dev/hd*`, в то время, когда во многих современных дистрибутивах даже IDE-диски именуются как `/dev/sd*` (исключение могут составить разве что некоторые дистрибутивы, например openSUSE, где даже в современных версиях используется обычный GRUB, а не GRUB2). Так что, здесь все правильно — во времена использования GRUB еще применялась старая схема именования жестких дисков. Поэтому если вам попался дистрибутив с загрузчиком GRUB, то в большинстве случаев IDE-диски будут называться `/dev/hd*`. В тех же современных дистрибутивах, где используется загрузчик GRUB2, IDE-диски называются `/dev/sd*`.

При работе с GRUB вам поначалу будет трудно разобраться с именами разделов диска. GRUB вместо привычных `/dev/hd*` (или `/dev/sd*` — для SCSI-дисков) использует свои собственные имена. Перевести имя `/dev/hd*` в имя в формате GRUB просто. Во-первых, опускается `/dev/`. Во-вторых, устройства отсчитываются не с буквы «а», как в Linux, а с нуля. Разделы на дисках отсчитываются не с единицы, а тоже с нуля, причем номер раздела указывается через запятую. Потом все имя берется в скобки. Так, раздел `/dev/hda1` в GRUB будет выглядеть как `(hd0,0)`, а раздел `/dev/hdb2` — как `(hd 1,1)`. Впрочем, об именах разделов в GRUB мы еще поговорим, но чуть позже, а сейчас разберемся со структурой файла `/boot/grub/grub.conf`.

Параметр `boot` указывает загрузочное устройство, а параметр `default` — загрузочную метку по умолчанию. Метка начинается параметром `title` и продолжается до следующего `title`. Нумерация меток начинается с 0. Параметр `timeout` задает количество секунд, по истечении которых будет загружена операционная система по умолчанию.

Параметр `default` полезно использовать с параметром `fallback`. Первый задает операционную систему по умолчанию, а второй — операционную систему, которая будет загружена в случае, если с загрузкой операционной системы по умолчанию произошла ошибка.

Задать графическое изображение позволяет параметр `spiashimage`. Чуть позже мы разберемся, как самостоятельно создать такое изображение.

Параметр `rootnoverify` указывается для Windows (точнее, для всех операционных систем не типа Linux). Параметр `chainloader` указывается для операционных систем, поддерживающих цепочечную загрузку. Если Windows на вашем компьютере установлена в неактивном разделе, с которого Windows загружаться не может, перед параметром `chainloader` нужно указать параметр `makeactive`.

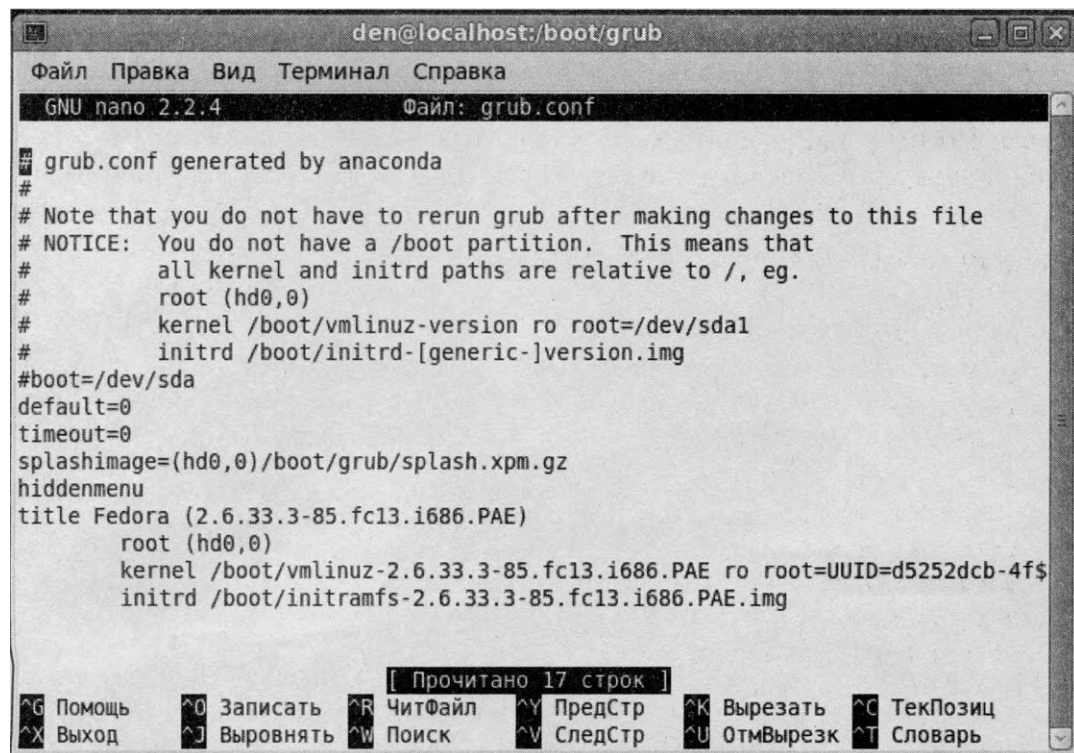


Рис. 21.1. Fedora: редактирование файла grub.conf

На рис. 21.1 изображен процесс редактирования конфигурационного файла загрузчика grub.conf.

21.2.2. Конфигурационный файл GRUB2

Основным конфигурационным файлом загрузчика GRUB2 является файл `/boot/grub/grub.cfg`. Он довольно большой (и с каждым годом становится все больше и больше), поэтому в книге я приводить его не стану. Если вам захочется его увидеть, вы можете сделать это на своем компьютере.

Конфигурационный файл `/boot/grub/grub.cfg` не редактируется вручную, поскольку для его создания используется утилита `/usr/sbin/grub-mkconfig`, которая генерирует его файл на основе шаблонов, хранящихся в каталоге `/etc/grub.d`, и настроек из файла `/etc/default/grub`.

Впрочем, при особом желании и понимании того, что делаете, можно редактировать этот файл и без каких-либо утилит.

В конфигурационном файле `/boot/grub/grub.cfg` содержится описание поведения GRUB2 (что нам совсем не интересно), а также элементов загрузочного меню. Собственно, ради элементов загрузочного меню часто и возникает необходимость отредактировать этот файл. Например, вы установили еще один дистрибутив Linux, и его инсталлятор не «прописал» новый дистрибутив в файле конфигурации загрузчика (или вы сами отказались от этого при установке).

Открыв файл `grub.cfg`, вы заметите, что его синтаксис весьма напоминает синтаксис `bash`-сценариев. Как уже было отмечено ранее, параметры GRUB2 задаются в файле `/etc/default/grub`, а сами элементы меню описаны в файлах, хранящихся в каталоге `/etc/grub.d`. Потом утилита `grub-mkconfig` «собирает» из этих файлов единый файл `grub.cfg`, корректируя поведение загрузчика на основании параметров из `/etc/default/grub`.

Рассмотрим описание типичного элемента меню:

```
menuentry 'Ubuntu' --class ubuntu --class gnu-linux --class gnu --class os
$menuentry_id_option 'gnulinux-simple-0cfcfdcf-d3e4-4755-a0ea-5d44470dee4f' {
    recordfail
    load_video
    gfxmode $linux_gfx_mode
    insmod gzio
    if [ x$grub_platform = xxen ]; then insmod xzio; insmod lzopio; fi
    insmod partjnsdos
    insmod ext2
    set root= 'hd0,msdos1'
    if [ x$feature_platform_search_hint = xy ]; then
search --no-floppy --fs-uuid --set=root --hint-bios=hd0,msdos1 --hint-efi=hd0,msdos1 --hint-baremetal=ahci0,msdos1 0cfcfdcf-d3e4-4755-a0ea-5d44470dee4f
    else
        search --no-floppy --fs-uuid --set=root 0cfcfdcf-d3e4-4755-a0ea-5d44470dee4f
    fi
    linux /boot/vmlinuz-3.19.0-15-generic root=UUID=0cfcfdcf-d3e4-4755-a0ea-5d44470dee4f ro quiet splash $vt_handoff
    initrd /boot/initrd.img-3.19.0-15-generic
}
```

В кавычках после `menuentry` находится описание элемента меню — можете заменить этот текст на все, что вам больше нравится. После названия элемента меню следуют различные необязательные параметры — в других дистрибутивах (не Ubuntu), они, как правило, могут не указываться.

Далее следуют команды GRUB2. Например, команда `insmod ext2` загружает модуль `ext2`. Но это не модуль ядра Linux! Это модуль GRUB2 — файл `ext2.mod`, находящийся в каталоге `/boot/grub`.

Команда `set root` устанавливает загрузочное устройство. Формат имени устройства такой же, как в случае с GRUB.

ВНУТРЕННЕЕ ИМЯ УСТРОЙСТВА GRUB

Мы знаем, что даже ATA-диски в новых дистрибутивах имеют имена вида `/dev/sda*`. Но команда `set root` загрузчика GRUB2 содержит имя `hd`. Это не опечатка! Это внутреннее имя устройства GRUB, а не имя системного устройства.

После служебного слова `linux` задается ядро (файл ядра) и параметры, которые будут переданы ядру. Служебное слово `initrd` указывает на файл `initrd`.

Теперь рассмотрим файл `/etc/default/grub`, содержащий параметры GRUB2 (листинг 21.2). Поскольку этот файл вы будете редактировать чаще, чем `grub.cfg`, то комментарии для большего удобства я перевел на русский язык.

Листинг 21.2. Файл `/etc/default/grub`

```
# Если вы измените этот файл, введите команду 'update-grub'
# для обновления вашего файла /boot/grub/grub.cfg.

# Элемент по умолчанию, нумерация начинается с 0
GRUB_DEFAULT=0
# Чтобы увидеть меню GRUB, надо или закомментировать следующую
# опцию, или установить значение больше 0, но в этом случае
# нужно изменить значение GRUB_HIDDEN_TIMEOUT_QUIET на false
GRUB_HIDDEN_TIMEOUT=0
GRUB_HIDDEN_TIMEOUT_QUIET=true
# Тайм-аут (в секундах)
GRUB_TIMEOUT="10"
# Название дистрибутива - вывод команды lsb_release или просто Debian
GRUB_DISTRIBUTOR='lsb_release -i -s 2> /dev/null | | echo Debian'
# Параметры ядра по умолчанию
GRUB_CMDLINE_LINUX_DEFAULT="quiet splash"
GRUB_CMDLINE_LINUX=""

# Раскомментируйте для отключения графического терминала
# (только для grub-pc)
#GRUB_TERMINAL=console

# Разрешение графического терминала
#GRUB_GFXMODE=640x480 0

# Раскомментируйте следующую опцию, если вы не хотите передавать
# параметр "root=UUID=xxx" ядру Linux
# GRUB_DISABLE_LINUX_UUID=true

# Раскомментируйте, если нужно отключить генерацию элемента меню
# режима восстановления
#GRUB_DISABLE_LINUX_RECOVERY="true"
# Раскомментируйте, чтобы получить гудок при запуске GRUB
#GRUB_INIT_TUNE="480 440 1"
```

После изменения файла `/etc/default/grub` не забудьте выполнить команду `update-grub` для обновления вашего файла `/boot/grub/grub.cfg`. Команда `update-grub` — это просто сценарий, вызывающий утилиту `grub-mkconfig` и передающий ей параметр — имя выходного файла (по умолчанию `/boot/grub/grub.cfg`). Ничто не мешает вам вызывать утилиту `grub-mkconfig` вручную, но использовать команду `update-grub` более удобно.

РЕДАКТИРОВАНИЕ ВРУЧНУЮ ФАЙЛА GRUB.CFG

Чуть ранее было сказано, что файл `grub.cfg` не следует редактировать вручную. Отчасти это так и есть — если не знаешь, что делаешь, лучше использовать утилиту `grub-mkconfig` или команду `update-grub`, но когда понимаешь о чем речь... В любом случае, при желании его редактировать можно, но только если вы уверены в своих силах...

Таким образом, при редактировании конфигурации GRUB2 нужно придерживаться одной стратегии из двух возможных:

- согласно первой вы редактируете файл `grub.cfg` вручную и не используете команд вроде `grub-mkconfig` или `update-grub`;
- вторая стратегия заключается в использовании вспомогательных программ, но тогда не нужно редактировать файл `grub.cfg` вручную, иначе при последующем изменении файла `grub.cfg` утилитой `grub-mkconfig` или командой `update-grub` все изменения, внесенные вручную, будут уничтожены. Поступая согласно второй стратегии, нужно редактировать файлы из каталога `/etc/grub.d` (там содержатся файлы, формирующие загрузочное меню) и файл `/etc/default/grub`, содержащий общие параметры GRUB2.

ВЫЗОВ КОМАНДЫ GRUB-MKCONFIG

По умолчанию команда `grub-mkconfig` генерирует конфигурационный файл на консоль, поэтому вызывать ее нужно так:

```
sudo grub-mkconfig > /boot/grub/grub.cfg
```

21.3. Команды установки загрузчиков

Установить GRUB/GRUB2, если вы это еще не сделали, можно следующей командой:

```
/sbin/grub-install <устройство>
```

Например:

```
/sbin/grub-install /dev/sda
```

После изменения конфигурационного файла переустанавливать загрузчик, как это требовалось в случае с устаревшим LILO, не нужно.

21.4. Установка собственного фона загрузчиков GRUB и GRUB2

Вы хотите создать собственный фон для загрузчика GRUB? Это очень просто: создайте или найдите в Интернете понравившуюся вам картинку, уменьшите ее до размера 640x480 и конвертируйте в формат XPM. Все это можно сделать одной командой:

```
# convert image.jpg -colors 14 -resize 640x480 image.xpm
```

Затем сожмите картинку с помощью команды `gzip`:

```
# gzip image.xpm
```


Скопируйте сжатую картинку в каталог `/boot/grub` и пропишите в конфигурационном файле `/boot/grub/grub.conf`:

```
splashimage=(hd0,1) /grub/image.хрш.gz
```

Теперь разберемся, как установить графический фон в GRUB2. Убедитесь, что установлен пакет `grub2-splashimages` — этот пакет содержит графические заставки для GRUB2, которые будут установлены в каталог `/usr/share/images/grub`. Если вам не нравятся стандартные картинки, множество фонов для GRUB2 вы можете скачать с сайта <http://www.gnome-look.org/> или создать вручную, как было только что показано. Кстати, GRUB2 уже поддерживает форматы PNG и TGA, поэтому можно не конвертировать файл фона в формат XPM.

Итак, будем считать, что картинка у нас уже выбрана. Осталось только установить ее как фон. Для этого откройте файл темы GRUB2 — он находится в каталоге `/etc/grub.d` (в Ubuntu и Debian этот файл называется `/etc/grub.d/05_debian_theme`).

НАЗВАНИЕ ФАЙЛА ТЕМЫ GRUB2

В других дистрибутивах файл темы GRUB2 может называться иначе. Точное его название для каждого дистрибутива указать сложно.

Найдите в файле темы строку:

```
for i in {/boot/grub,/usr/share/images/desktop-base}/moreblue-orbit-  
grub.{png,tga} ; do
```

Замените ее на следующую строку:

```
for i in {/boot/grub,/usr/share/images/desktop-  
base,/usr/share/images/grub}/имя_файла.{png,tga} ; do
```

Как видите, мы просто прописали выбранную вами картинку. Далее нужно обновить GRUB2:

```
sudo update-grub
```

21.5. Постоянные имена устройств

Как было отмечено ранее, все современные дистрибутивы перешли на так называемые постоянные («длинные») имена. Раньше, когда еще никто не знал о длинных именах, запись в файле `grub.conf` могла выглядеть так:

```
kernel /boot/vmlinuz26 root=/dev/hda1 vga=0x318 ro
```

Эта запись указывает имя ядра (`/boot/vmlinuz26`). Все, что после него, — параметры, которые будут переданы ядру. Один из них (параметр `root`) указывает имя корневой файловой системы. Здесь оно приведено еще в старом формате. Сейчас вы такие имена в `grub.conf` не увидите (если, конечно, сами не пропишете). Варианты указания длинных имен выглядят так:

```
root=/dev/disk/by-uuid/2d781b26-0285-421a-b9d0-d4a0d3b55680  
root=/dev/disk/by-id/scsi-SATA_WDC_WD1600JB-00_WD-WCANM7959048-part5  
root=LABEL=
```

Какой вариант будет использоваться у вас, зависит от дистрибутива. Например, в Fedora применяют третий способ, а в openSUSE — второй.

21.6. Восстановление загрузчика GRUB/GRUB2

Что делать, если вы переустановили Windows, а она установила в MBR свой загрузчик, и теперь вы не можете загрузить Linux? Не переустанавливать же еще и Linux из-за такой мелочи!

Для восстановления загрузчика GRUB/GRUB2 нужно загрузиться с LiveCD (подойдет любой LiveCD с любым дистрибутивом Linux) и ввести следующие команды:

```
mkdir /old
mkdir /old/dev
mount /dev/sdaN /old
```

Все команды следует вводить от имени root, для чего использовать команды `su` или `sudo`.

В частности, в LiveCD Ubuntu нужно вводить все команды с использованием команды `sudo` — например, так:

```
sudo mkdir /old
sudo mkdir /old/dev
```

Разберемся, что означают эти команды:

- первая из них создает каталог `/old`, который будет использоваться в качестве точки монтирования;
- вторая — создает в этом каталоге подкаталог `dev`, который пригодится для монтирования `devfs` — псевдофайловой системы;
- третья — служит для монтирования корневой файловой системы дистрибутива Linux, установленного на жестком диске в разделе `/dev/sdaN` (где *N* — номер раздела), к каталогу `/old`. Предположим, что на вашем компьютере дистрибутив Linux был установлен в раздел `/dev/sda5`. Тогда вам нужно ввести следующую команду:

```
mount /dev/sda5 /old
```

После этого надо подмонтировать каталог `/dev` к каталогу `/old/dev` — это делается с помощью все той же команды `mount`, но с параметром `--bind`:

```
mount --bind /dev /old/dev
chroot /old
```

Команда `chroot` заменяет корневую систему нашего LiveCD на корневую систему дистрибутива, установленного на винчестере. Вам остается лишь ввести команду:

```
/sbin/grub-install /dev/sda
```

Эта команда установит загрузчик GRUB/GRUB2 так, как он был установлен до переустановки Windows. После установки загрузчика следует перезагрузить компьютер командой `reboot`.

ДОПОЛНИТЕЛЬНАЯ ИНФОРМАЦИЯ

Дополнительную информацию о восстановлении загрузчика GRUB вы можете получить на моем форуме: <http://www.dkws.org.ua/phpbb2/viewtopic.php?t=3275>.

21.7. Загрузка с ISO-образов

Предположим, вы скачали ISO-образ новой версии Ubuntu, но у вас нет «болванки», чтобы записать на нее образ и загрузиться с полученного диска. Могу вас обрадовать: «болванка» вам для этого не понадобится — GRUB2 умеет использовать ISO-образы в качестве загрузочных устройств. Просто пропишите ISO-образ в конфигурационном файле GRUB2 и перезагрузите компьютер. Новая загрузочная метка появится в меню GRUB2, и, если ее выбрать, система загрузится с ISO-образа.

Итак, создайте в каталоге `/boot` подкаталог `iso` (название, сами понимаете, может быть любым) и загрузите в него ISO-образ дистрибутива. Теперь вам осталось лишь отредактировать конфигурационный файл `/boot/grub/grub.cfg`, добавив в него вот такую загрузочную запись (выделенный полужирным шрифтом текст нужно записать в одну строку):

```
menuentry "Ubuntu LiveCD" {
    loopback loop /boot/iso/ubuntu.iso
    linux (loop)/casper/vmlinuz boot=casper iso-
scan/filename=/boot/iso/ubuntu.iso noeject noprompt --
    initrd (loop)/casper/initrd.lz
}
```

Перезагружаемся и выбираем пункт меню **Ubuntu LiveCD**.

21.8. Установка пароля загрузчика

Теперь самое время защитить наш загрузчик. По умолчанию любой желающий может изменить параметры ядра. Достаточно злоумышленнику передать ядру параметры `rw`, `single` или `rw, init=/bin/bash`, и после загрузки он сможет сделать с системой все, что захочет, — например, изменить пароль `root`. А получив `root`-доступ, сможет настроить систему так, как ему это выгодно, или полностью уничтожить ее (хотя это можно было бы сделать и на этапе загрузки).

Поэтому мы должны защитить загрузчик паролем. Загрузка операционных систем будет осуществляться без пароля, однако если кто-то захочет изменить параметры ядра, то у него ничего не получится — загрузчик попросит ввести пароль. Для самых «продвинутых» доброжелателей, которые смогут подключить жесткий диск к Windows-системе и с помощью Total Commander просмотреть конфигурационный файл GRUB, мы закодируем наш пароль с помощью алгоритма MD5 — это самый

стойкий алгоритм шифрования на сегодняшний день. Поэтому, даже если злоумышленник и просмотрит конфигурационный файл загрузчика, пароль он все равно не узнает.

21.8.1. Загрузчик GRUB

Введите команду `grub`. Появится приглашение:

```
grub>
```

В ответ на приглашение введите команду:

```
md5crypt
```

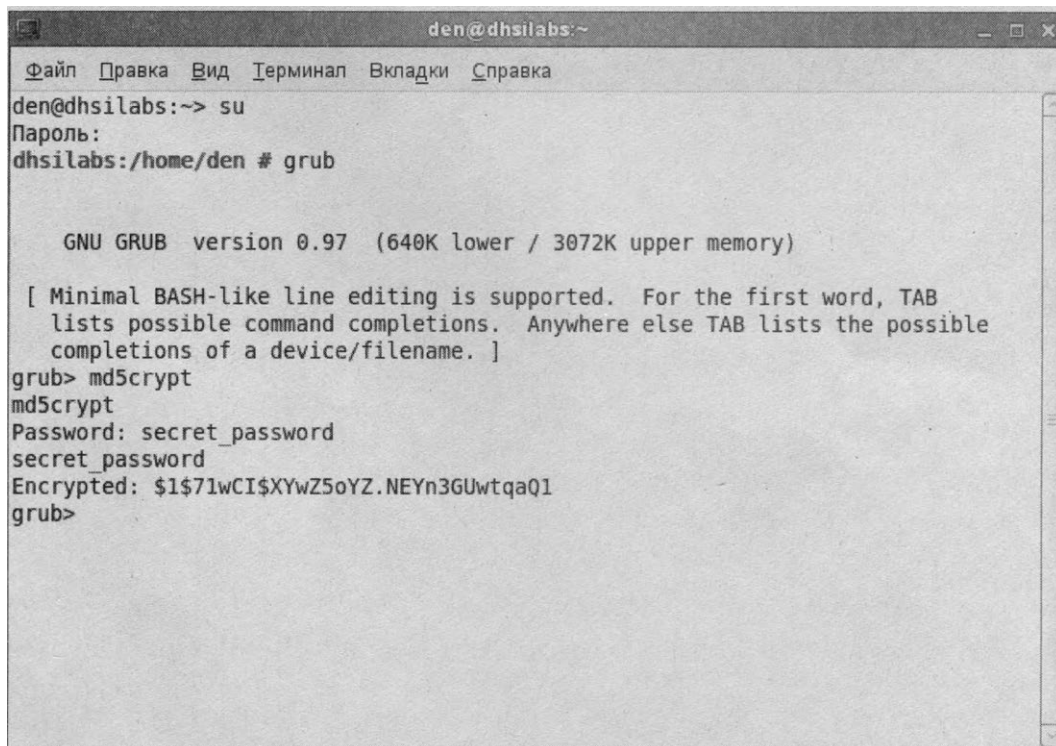
Программа запросит у вас пароль (придумайте и введите пароль в ответ на запрос), закодирует его и выведет на экран шифр введенного вами пароля:

Password: *****

Итак, вы получили зашифрованный пароль (рис. 21.2). Перепишите этот шифр (а еще лучше выделите его и выполните команду меню терминала **Правка | Копировать**) и введите команду `Quit`.

На всякий случай сделайте копию конфигурационного файла загрузчика:

```
sudo cp /boot/grub/grub.conf /boot/grub/grub.conf_backup
```

A screenshot of a terminal window titled 'den@dhsilabs:~'. The window has a menu bar with 'Файл', 'Правка', 'Вид', 'Терминал', 'Вкладки', and 'Справка'. The terminal shows the user switching to root with 'su', then entering the 'grub' command. Inside the GRUB prompt, the user enters 'md5crypt', followed by a password 'secret_password'. The terminal displays the encrypted password: '\$1\$71wCI\$XYwZ5oYZ.NEYn3GUwtqaQ1'. The prompt returns to 'grub>'.

```
den@dhsilabs:~> su
Пароль:
dhsilabs:/home/den # grub

GNU GRUB version 0.97 (640K lower / 3072K upper memory)

[ Minimal BASH-like line editing is supported. For the first word, TAB
  lists possible command completions. Anywhere else TAB lists the possible
  completions of a device/filename. ]
grub> md5crypt
md5crypt
Password: secret_password
secret_password
Encrypted: $1$71wCI$XYwZ5oYZ.NEYn3GUwtqaQ1
grub>
```

Рис. 21.2. openSUSE: шифрование пароля GRUB

Теперь откройте файл `/boot/grub/grub.conf` (или `/boot/grub/menu.lst`— в зависимости от дистрибутива) в любом текстовом редакторе (рис. 21.3):

```
gksudo gedit /boot/grub/grub.conf
```

или

```
gksudo gedit /boot/grub/menu.lst
```

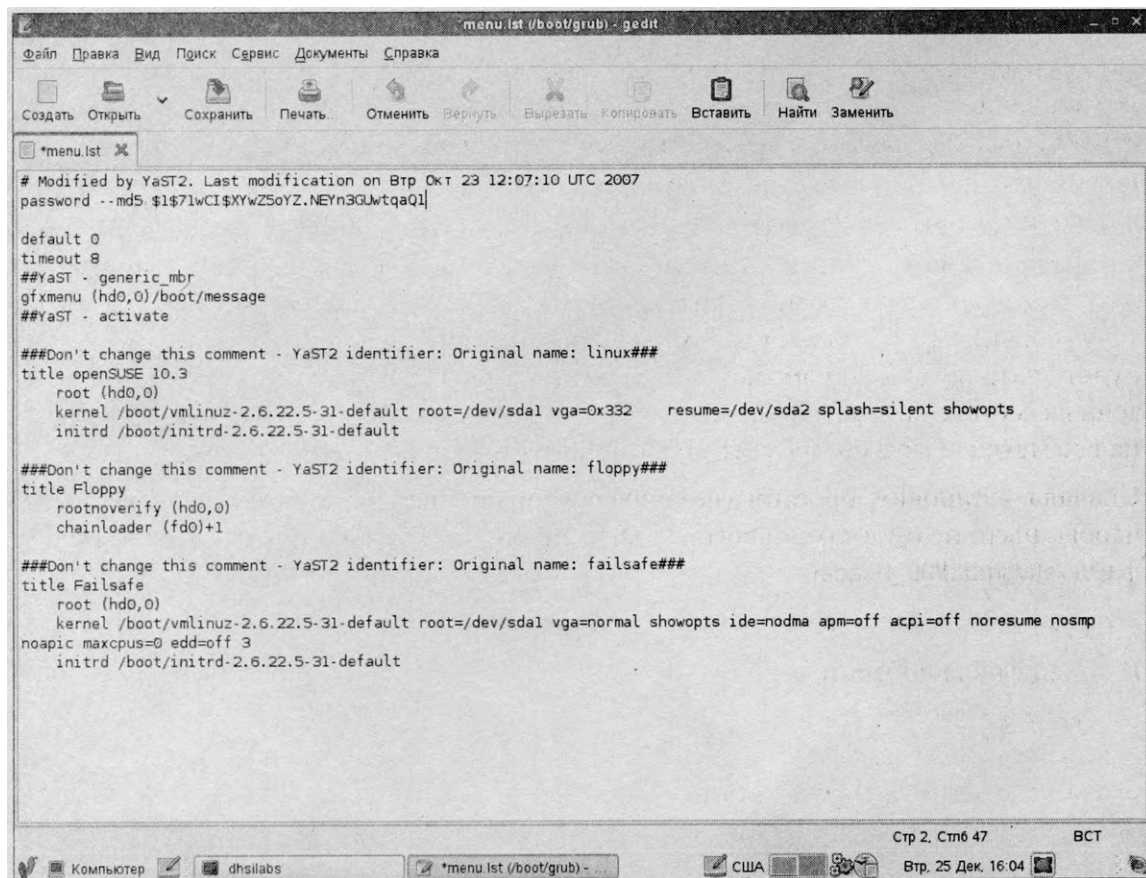


Рис. 21.3. openSUSE: редактирование файла `/boot/grub/menu.lst`

Найдите секцию пароля:

```
## password ['--md5'] passwd
# If used in the first section of a menu file, disable all interactive editing
# control (menu entry editor and command-line) and entries protected by the
# command 'lock'
# e.g. password topsecret
#     password --md5 $1$gLhU0/$aW78kHK1QfV3P2b2znUoe/
# password topsecret
```

После нее вставьте строку:

```
password --md5 ваш-шифр
```

Параметр `ваш-шифр` здесь — это тот шифр, который вы получили в ответ на введенный пароль.

Таким образом мы задали пароль, с помощью которого можно редактировать загрузочное меню GRUB. И пока не будет указан заданный пароль, GRUB не разрешит редактировать загрузочное меню.

21.8.2. Загрузчик GRUB2

По сравнению с GRUB, загрузчик GRUB2 одновременно и проще в обращении, и сложнее в настройке. Настраивать GRUB2 придется реже, но к его сложной настройке надо привыкнуть, — практически все современные дистрибутивы перешли на GRUB2.

В GRUB можно было задать общий пароль для всех загрузочных меток, а также установить пароль только на некоторые загрузочные метки. В GRUB2 можно сделать то же самое, но кроме самого пароля понадобится указать еще и имя пользователя (логин), что усложняет злоумышленнику взлом системы, поскольку ему нужно будет знать не только пароль, но и кому он принадлежит. Защита отдельных загрузочных меток, как правило, используется редко, — чаще устанавливается пароль на все метки сразу, что и будет продемонстрировано далее.

Сначала установим простой (незашифрованный) пароль, а затем зашифруем его, чтобы никто не смог его прочитать, загрузившись с LiveCD. Прежде всего откройте файл `/etc/grub.d/00_header`:

```
sudo nano /etc/grub.d/00_header
```

В конец файла добавьте строки:

```
cat « EOF
set superusers="den"
password den 1234
EOF
```

Здесь имя пользователя `den`, пароль мы придумали для примера такой: `1234`.

Теперь обновите GRUB2:

```
sudo update-grub
```

Можно также напрямую редактировать `grub.cfg` — файл конфигурации GRUB2. В него следует добавить вот такие строки:

```
set superusers="user1"
password user1 password1
password user2 password2
```

Обратите внимание, что командами `password` заданы два пользователя: `user1` и `user2` с паролями `password1` и `password2` соответственно. Но пользователь `user1` является суперпользователем, т. е. может редактировать загрузочные метки GRUB2, а обычный пользователь (`user2`) может только загружать метки. Таким образом,

у пользователя `user1` получится передать ядру новые параметры, а пользователь `user2` сможет только загрузить Linux с параметрами по умолчанию.

Можно даже задать условие, что метку Windows будет загружать только пользователь `user2`:

```
menuentry "Windows" --users user2 {
    set root=(sd,2)
    chainloader +1
}
```

Теперь разберемся с шифрованием пароля. Команда `password` поддерживает только незашифрованные пароли. Если вы хотите использовать зашифрованные пароли, нужно применить команду `password_pbkdf2`. Например:

`password_pbkdf2 den зашифрованный_пароль`

Получить зашифрованный пароль можно командой:

`grub-mkpasswd-pbkdf2`

Программа запросит у вас пароль (придумайте и введите пароль в ответ на запрос), закодирует его и выведет на экран хэш (шифр) введенного вами пароля:

Your PBKDF2 is grub.pbkdf2.зашифрованный_пароль

Пример такого шифра:

```
grub.pbkdf2.sha512.10000.9290F727ED06C38BA4549EF7DE25CF5642659211B7FC076F2D28FE
FD71784BB8D8F6FB244A8CC5C06240631B97008565A120764C0EE9C2CB0073994D7 9080136.887C
FF169EA8 3352 35D8 00424 2AA7D6187A41E3187 DF0CE14E256D85ED97A97357AAA8FF0A3871AB9EE
FF458392F4 62F4 95487387F685B7472FC6C29E293F0A0
```

Весь этот хэш нужно скопировать в конфигурационный файл GRUB2:

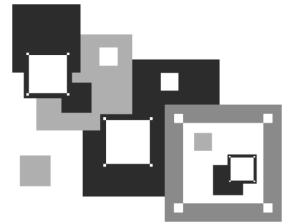
```
password_pbkdf2 den
grub.pbkdf2.sha512.10000.9290F727ED06C38BA4549EF7DE25CF5642659211B7FC076F2D28FE
FD71784BB8D8F6FB244A8CC5C06240631B97008565A120764C0EE9C2CB0073994D79080136.887C
FF169EA8335235D8004242AA7D6187A41E3187DF0CE14E256D85ED97A97357AAA8FF0A3871AB9EE
FF458392F4 62F4 95487387F685B7472FC6C29E293F0A0
```

Если вы не использовали файл `00_header`, а редактировали непосредственно файл `grub.cfg`, то команду `update-grub` вводить не нужно!

Дополнительную информацию по вопросам шифрования загрузчика GRUB2 вы сможете получить по адресам:

- <http://ubuntuguide.net/how-to-setup-boot-password-for-grub2-entries;>
- <http://grub.enbug.org/Authentication.>

ГЛАВА 22



Системы инициализации

22.1. Начальная загрузка Linux

В этой книге мы уже упоминали о начальной загрузке компьютера, поэтому сейчас начнем с того момента, когда загрузчик BIOS нашел загрузочное устройство, — например, жесткий диск. Далее загрузчик BIOS считывает первый (нулевой) сектор диска и передает ему управление. На этом работа загрузчика BIOS заканчивается.

В первом секторе находится главная загрузочная запись (Master Boot Record, MBR), состоящая из трех частей: первичного загрузчика, таблицы разделов диска (partition table) и флага загрузки.

Сначала из первой части MBR вызывается первичный загрузчик — что и как он делает, прописано в нем самом. Работу первичного загрузчика мы рассмотрим на примере двух загрузчиков: LILO и GRUB/GRUB2.

ЕЩЕ РАЗ О ЗАГРУЗЧИКЕ LILO

Загрузчик LILO, хоть и устарел безнадежно (см. главу 21), но очень прост и поэтому идеально подходит для использования в «академических целях» — для изучения самого процесса загрузки. А GRUB/GRUB2 — это современные загрузчики, без понимания принципов работы которых сегодня просто нельзя.

Загрузчик LILO состоит из двух частей: первая содержится в MBR, а вторая размещена на диске в виде файла `/boot/boot.b`. По аналогии с LILO загрузчик GRUB/GRUB2 (далее — просто GRUB) тоже состоит из двух частей: `stage 1` и `stage2`. Первая часть (`stage1`) помещается в MBR, а вторая хранится на диске в каталоге `/boot/grub`. Фактическое расположение `stage2` указывается при установке GRUB примерно вот такой командой:

```
grub> install (hd0,4)/boot/grub/stage1 (hd0) (hd0,4)/boot/grub/stage2 p (hd0,4)
/boot/grub/menu.conf
```

Кроме `stage1` и `stage2` у загрузчика GRUB есть еще несколько промежуточных частей — `* stage 1_5` — помогающих загрузчику найти `stage2` и выполняющих другие подготовительные действия, — в частности, обеспечивающих поддержку разных файловых систем.

Задача первой части — запуск вторичного загрузчика (второй части), который и производит дальнейшую загрузку системы. Дело в том, что первая часть ничего не

знает о файловых системах, поэтому местонахождение второй части записано в ней в «физических координатах»: явно указаны цилиндр, головка и сектор жесткого диска.

Вторая часть загрузчика более интеллектуальна. Она уже «знает», что такое файловая система, и что карта размещения файлов записана в файле `/boot/System.map`. По аналогии с картой размещения файлов имеется в GRUB и карта устройств — файл `/boot/grub/device.map`. Оба этих файла используются для поиска ядра и образа виртуального диска.

ВИРТУАЛЬНЫЙ ДИСК

Для чего нужен виртуальный диск? Представим, что мы еще не установили Linux, а только собираемся это сделать. Вставляем загрузочный диск, и загрузчик запускает не просто инсталлятор — на самом деле запускается операционная система Linux, ясно виден процесс загрузки ядра, а потом уже запускается программа установки. Но ядру нужно же откуда-то прочитать модули поддержки устройств и файловой системы — ведь корневая файловая система еще не создана. Вот все эти модули и находятся на виртуальном диске. Виртуальный диск загружается в память, ядро монтирует его, как обычную файловую систему, и загружает с него все необходимые модули. После этого виртуальный диск размонтируется и — в случае нормальной загрузки, а не установки Linux, — вместо него монтируется обычная корневая файловая система.

Работа с виртуальным диском основана на технологии `initrd` (INITial Ram Disk). Файл образа виртуального диска находится в каталоге `/boot` и носит название `initrd-<версия ядра>`.

В процессе запуска ядра монтируется корневая файловая система и запускается программа `init`, которая и выполняет дальнейшую инициализацию системы. Программа `init` — часть `init`, самой надежной и распространенной системы инициализации Linux. Увы, она уже устарела. Даже самые консервативные дистрибутивы, такие как Debian, отказались от нее. Но нужно заметить к чести самой `init`, «жила» она очень долго (в Debian она продержалась до седьмой версии, а в Fedora — до 14-й).

Кроме системы инициализации `init` (см. *разд. 22.2*) существуют и другие системы — в частности: `initng`, `upstart` и `systemd`:

- система `initng` позволяет существенно ускорить запуск Linux, но, к сожалению, она так и осталась экспериментальной и не прижилась в дистрибутивах Linux.

Заинтересовавшиеся могут прочитать о системе `initng` в моей статье по адресу : <http://www.dkws.org.ua/article.php?id=12> — на тот случай, если вам захочется создать собственный дистрибутив на ее основе;

- система `upstart` была специально разработана для дистрибутива Ubuntu, но ее при желании можно установить в любом дистрибутиве (некоторые идеи этой системы используются также в `systemd`).

Описание системы `upstart` в это издание не вошло, поскольку она не использовалась нигде, кроме старых версий Ubuntu (и дистрибутивов-клонов), а последние версии Ubuntu основаны на системе `systemd`. Но если вам по каким-либо причинам необходимо знакомство с системой `upstart`, вы можете обратиться или

к предыдущим изданиям этой книги (вы с легкостью найдете их на Play Market), или же к сторонним источникам в Интернете;

- система `systemd` (см. *разд. 22.3*) — современная система инициализации, заменившая `init` в последних версиях дистрибутивов Fedora, openSUSE, Ubuntu и некоторых других.

НЕМНОГО ИСТОРИИ

Прежде чем перейти к рассмотрению систем инициализации, позволю себе небольшой исторический экскурс, чтобы вы понимали, who is who. С самого начала (т. е. со времен UNIX, когда о Linux еще никто не слышал) существовало две системы инициализации: SysV (использовалась, начиная с System V) и BSD (разработанная для собственной версии UNIX университетом Беркли). Во всех Red Hat-совместимых дистрибутивах (Red Hat, Mandrake, Fedora Core, Mandriva, Fedora, openSUSE и др.) использовалась система инициализации SysV, т. е. привычная всем нам программа `init`. Но время шло, новые компьютеры становились существенно мощнее старых, а Linux продолжала загружаться на быстрых компьютерах примерно с той же скоростью, что и на медленных... Вот тогда и задумались о смене системы инициализации. Были предложены различные варианты систем: `initng` (так и осталась экспериментальной), `upstart` (действует на «старых» Ubuntu, а в современных заменена на `systemd`) и `systemd`, которая стала применяться в Fedora, начиная с ее 15-й версии. Основная цель всех этих систем — сделать запуск Linux быстрее. С тем, как они это осуществляют, мы разберемся позже. А начнем рассмотрение систем инициализации мы, все-таки, с традиционной системы `init`.

22.2. Система инициализации `init`

Сначала я вообще хотел удалить описание системы `init` из этого издания, но понял, что сбрасывать со счетов ее еще рано. Ведь не все спешат отказываться от старых дистрибутивов, — так, многие до сих пор используют Debian 7, — зачем менять то, что прекрасно работает? Да и не всегда есть возможность установить самую последнюю версию дистрибутива — например, при использовании виртуализации OpenVZ вообще придется ограничиться дистрибутивами с версией ядра 2.6, а в таких дистрибутивах будет доступна только `init`. Вообще, если вы работаете со старыми версиями дистрибутивов, то без `init` — никак (на многих серверах, организованных даже всего несколько лет назад, работает именно `init`, поскольку общая миграция на `systemd` произошла не так давно). Система инициализации `init` не была и не будет удалена из этой книги, поскольку — это классика. И на ее примере проще показать, чем лучше система `systemd`, — контраст уж больно заметен.

Итак, программа `init` читает конфигурационный файл `/etc/inittab` и запускает другие процессы, согласно инструкциям этого файла (листинг 22.1).

Листинг 22.1. Файл `/etc/inittab`

```
id:5:initdefault:

# Инициализация системы
si::sysinit:/etc/rc.d/rc.sysinit
10:0:wait:/etc/rc.d/rc 0
```

```
11:1:wait:/etc/rc.d/rc 1
12:2:wait:/etc/rc.d/rc 2
13:3:wait:/etc/rc.d/rc 3
14 : 4 .-wait:/etc/rc. d/rc 4
15:5:wait:/etc/rc.d/rc 5
16:6:wait:/etc/rc.d/rc 6

# Что делать при нажатии CTRL-ALT-DELETE
ca::ctrlaltdel:/sbin/shutdown -t3 -r now

# От UPS была получена команда, что пропало питание.
# Немного ждем и выключаем компьютер
pf: :powerfail:/sbin/shutdown -f -h +2 "Power Failure; System Shutting Down"

# От UPS получена команда, что питание возобновилось
# Отменяем shutdown
pr:12345rpowerokwait:/sbin/shutdown -c "Power Restored; Shutdown Cancelled"

# Запуск gettys
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

# Однопользовательский режим
:S:wait:/bin/sh
```

Одна из главных инструкций файла `/etc/inittab` выглядит так:

```
id:<число>:initdefault:
```

Эта инструкция задает уровень запуска по умолчанию. Уровень запуска определяет, какие действия будут выполнены программой `init` (т. е., какие процессы будут запущены). Всего предусмотрено шесть уровней запуска:

- ☐ 0 — останов системы (ясно, что в качестве уровня по умолчанию этот уровень быть не может);
- ☐ 1 — однопользовательский режим (в него можно перейти сразу при загрузке, передав ядру параметр `single`);
- ☐ 2 — многопользовательский режим без поддержки сети;
- ☐ 3 — многопользовательский режим с поддержкой сети;
- ☐ 4 — не используется;
- ☐ 5 — многопользовательский графический режим с загрузкой X11 и поддержкой сети;
- ☐ 6 — перезагрузка системы.

В большинстве случаев в качестве уровня запуска по умолчанию устанавливается 3 или 5.

22.2.1. Команда *init*

Перейти на тот или иной уровень можно и после загрузки системы. Для этого используется команда:

```
# /sbin/init <уровень_запуска>
```

Напомню, что решетка (#) перед командой означает, что команда должна быть выполнена от имени пользователя root.

«Вычислив» уровень запуска, *init* поочередно запускает сценарии из каталога */etc/rc.d/rcX.d*, где *X* — это номер уровня запуска. Если зайти в один из этих каталогов, например в */etc/rc.d/rc3.d*, то можно увидеть ссылки формата:

```
S<номер><имя>
```

Параметр *<номер>* определяет порядок запуска сценария (например, *siOnetwork* запустится раньше, чем *S11internet*), а параметр *<имя>* — задает имя сценария. Сами сценарии находятся в каталоге */etc/rc.d/init.d*.

Ссылки, начинающиеся на символ *S*, — это ссылки запуска (от *start*), при запуске соответствующих сценариев им будет передан аргумент *start*. Например, если *init* обнаружила в */etc/rc.d/rc3.d* файл *S10network*, то она выполнит команду:

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

Если имя ссылки начинается на букву *K* (от *kill*), то это ссылка останова сервиса, — например: *K0iservice*. Эта ссылка указывает на команду:

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

Вы можете запустить любой сценарий из каталога *init.d* непосредственно, передав ему параметры *start* (запуск), *stop* (останов) и другие (зависит от сервиса).

22.2.2. Команда *service*

А можете воспользоваться командой *service*:

```
# service <имя_сервиса> <start|stop|...>
```

Здесь *<имя_сервиса>* — это имя файла в каталоге */etc/rc.d/init.d*.

В openSUSE имеется удобная команда:

```
rc<имя_сервиса> <start|restart|stop>
```

Так, для запуска Apache можно использовать команду:

```
# rcapache start
```

22.2.3. Редакторы уровней запуска

Редактировать уровни запуска можно вручную, а можно и с помощью программ-конфигураторов :

- в Fedora — конфигуратором `system-config-services` (см. *разд. 22,3.6*);
- в openSUSE — конфигуратором YaST;
- в Ubuntu до версии 9.04 применялся конфигуратор `services-admin`, а с версии 9.10 следует использовать конфигуратор `bum`, устанавливаемый отдельно:

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

22.2.4. Параллельная загрузка сервисов, или как сделать старый `init` быстрее

Система инициализации `init` довольно неповоротлива. А все из-за того, что она запускает сервисы последовательно — в имени ссылки на сервис даже есть номер, задающий порядок запуска сервиса. Запустив сервис А, `init` ждет, пока он запустится, и только после этого запускает сервис Б. Но ведь сервисы А и Б можно запускать параллельно — возможности современных процессоров это позволяют. В результате можно достичь существенного сокращения времени загрузки. Так, на моей тестовой системе (правда, запущенной в VMware), я получил сокращение загрузки до 20 секунд, — мелочь, а приятно.

Откройте файл `/etc/rc.d/rc` и найдите строку вида:

```
$i start
```

Возможно, там будет такая строка (все зависит от дистрибутива и версии `init`):

```
exec $i start
```

После команды `start` добавьте символ `&` — строка запуска сервисов станет выглядеть так:

```
$i start &
```

или так:

```
exec $i start &
```

Символ `&` здесь разрешает запуск программы в фоновом режиме — при этом следующая команда будет выполнена без ожидания завершения предыдущей.

Однако эта схема работает не всегда. Представим, что нужно запустить сервисы А, Б, В и Г. Однако сервис В — довольно нерасторопный и запускается медленно, а сервис Г зависит от сервиса В. Получается, что сервис Г не сможет быть корректно запущен, пока не запустится сервис В. В результате время загрузки системы только увеличится. Что делать? Или отказаться от сервиса Г, если он вам не так и нужен, или же использовать более совершенную систему инициализации, параллельно запускающую сервисы на основе информации о зависимости сервисов. Такой системой является `cinit`, прочитать о настройке которой можно по адресу:

<http://nico.schottelius.org/documentations/speeches/metarheinmain-chaosdays-110b/cinit/view>.

22.3. Система инициализации `systemd`

22.3.1. Идеальная система инициализации

А теперь начнем наше знакомство с новой системой инициализации. Но прежде еще раз остановимся на моментах, которые не устраивали нас (как пользователей Linux) в старой системе `init`.

Как мы знаем, `init` (точнее, процесс системы инициализации) — это процесс с UID 1, он первым запускается ядром и выступает родителем для всех процессов, у которых нет собственного родителя. Основная задача такого процесса (помимо всех остальных задач) — инициализация системы. А значит, он должен это сделать быстро. Как запустить систему быстро? Во-первых, запускать не все, что можно, а только самое необходимое. Во-вторых, запускать сервисы параллельно. Существует еще одна система инициализации, запускающая сервисы параллельно, — это `upstart` (о ней мы говорили в начале главы — она, правда, больше не используется, но от этого ее возможности не стали хуже).

Давайте разберемся теперь, как запустить минимум сервисов, а запуск всех остальных отложить до тех пор, пока они не понадобятся. В некоторых случаях мы знаем, какие сервисы нам понадобятся заранее. Как правило, это `syslog`, `dbus` и т. д. Но представьте, что на своем ноутбуке мы хотим передавать и принимать файлы по Bluetooth. Нужен нам демон `bluetoothd`? Да, потому что мы хотим использовать Bluetooth. Но в данный момент, пока мы ничего не передаем и не принимаем, — он не нужен. То есть, пока не включен Bluetooth-адаптер (и пока одно из пользовательских приложений не захочет с ним общаться через D-Bus), загружать `bluetoothd` не требуется. Аналогично и для системы печати `cups` — хоть принтер и подключен к компьютеру, но сервис `cupsd` нужен лишь тогда, когда происходит печать. То же и для сетевых сервисов — пока никто не обращается к вашему Web- или FTP-серверу, нет необходимости их запускать, что экономит не только лишние секунды при запуске системы, но и снизит нагрузку на процессор в процессе работы системы. Да и память сэкономит.

Теперь поговорим о параллельном запуске нужных нам сервисов. Современные процессоры настолько мощны, что система инициализации может попытаться запустить все нужные сервисы одновременно. Этим она полностью загрузит имеющиеся ресурсы, но и сократит время запуска системы.

Однако запустить все и сразу нельзя — необходимо синхронизировать запуск сервисов. Иначе получится, что сервису Б требуется сервис А, который еще не запустился. «Вес» сервисов разный, действия, выполняемые при запуске, — тоже разные. Даже если вы сначала запустите сервис А, а потом — Б, сервис Б может запуститься быстрее базового сервиса А. Приведу для конкретности некоторые примеры: практически всем службам нужен `syslog` (иначе, как они будут вести протоколирование?), поэтому им необходимо дождаться его запуска. Многим службам (например, тому же Avahi) нужен D-Bus, поэтому, пока D-Bus не будет запущен, сервису Avahi приходится ждать.

В результате на практике получается, что большая часть сервисов запускается все равно последовательно, а не параллельно, и выигрыш времени по сравнению с `init` — мизерный.

Как снять ограничения синхронизации? Для этого нам надо понять, что нужно сервису Б от сервиса А, и как он вообще проверяет, что сервис А запущен? Оказывается, сервису Б всего лишь нужен *сокет сервиса А* (*socket service*). А что такое сокет сервиса? Это всего лишь файл. Например, все сервисы, которым нужен D-Bus, ждут возможности подключения к файлу `/var/run/dbus/system__bus_socket`. Всем, кому нужен `syslog`, ждут возможности подключения к устройству `/dev/log` и т. д.

По сути, все, что нужно, — это сделать доступными сокеты сервисов до запуска самих сервисов. А когда сервис фактически запустится, мы можем передать ему сокет с помощью команды `exec ()`.

Получается, что для параллельного запуска всех демонов нам сначала необходимо создать для них все сокеты, а потом параллельно запустить все демоны.

Что произойдет в нашей ситуации, когда сервис Б требует запуска сервиса А? Ничего страшного — сервис Б станет в очередь и будет ждать, пока сервис А запустится. Главное, что сокет сервиса открыт.

А что, если сервис А (пусть это будет `syslog`) требуется сразу нескольким сервисам: Б, В, Г и Д? Тоже не страшно. Каждый из этих сервисов отправит свое сообщение в буфер сокета `/dev/log`, затем запустится `syslog` и обработает все эти сообщения.

Как видите — все гениальное просто. Но не нужно думать, что это какое-то совсем новое изобретение. Подобная система инициализации работает в Mac OS — там она называется `launchd`. Но поскольку не все знакомы с Mac OS, эти идеи Apple известны не многим.

СУПЕРСЕРВЕР INETD

Впрочем, сама идея — еще старше. Подобным образом работал древнейший суперсервер `inetd`. Если вы его помните — хорошо, но останавливаться на нем мы здесь не будем.

22.3.2. `systemd` — основные понятия

Система инициализации `systemd` контролирует всю систему — отсюда ее название. В настоящее время она используется в последних версиях дистрибутивов, рассматриваемых в этой книге: Fedora, openSUSE и Ubuntu.

Система `systemd` построена на концепции *модулей* (*units*). У каждого модуля есть свое имя и тип. Например, модуль типа `nsed.service` управляет сервисом (демоном) `nsed`. Основными типами модулей являются:

- *service* (сервис)— демоны, которые можно запустить, перезапустить, остановить. Для совместимости с SysV (пока еще не все привыкли к `systemd`) в системе есть возможность чтения традиционных сценариев управления демонами. Как обычно, они находятся в каталоге `/etc/init.d`. Ради справедливости нужно отметить, что в openSUSE 12.1 содержимое каталога `init.d` такое же, как и было раньше (при использовании `init`), поэтому сразу и не заметишь, что используется но-

вая система инициализации. А вот в Fedora 16¹ сценариев в `init.d` гораздо меньше и сразу видно — что-то тут не так;

- *socket* (сокет) — реализует сокет, расположенный в файловой системе или Интернете. Поддерживаются сокеты `AF_INET`, `AF_INET6`, `AF_UNIX`. У каждого сокета есть связанный с ним сервис. Например, при попытке установки соединения с сокетом `nscd.socket` будет запущен сервис `nscd.service`. Вам это ничего не напоминает? А я вспоминаю старый суперсервер `inetd` и его более новую версию `xinetd` — они работали именно так;
- *device* (устройство) — реализует устройство в дереве устройств. Если устройство описано через правила `udev`, то его можно представить в `systemd` как модуль типа `device`;
- *mount* (точка монтирования) — реализует точку монтирования в файловой системе. Демон `systemd` контролирует все точки монтирования, их подключение и отключение. Теперь файл `/etc/fstab` не главный, а служит дополнительным источником информации о точках монтирования, хотя вы по-прежнему можете описывать в нем свои точки монтирования;
- *automount* (автоматическая точка монтирования) — реализует автоматическое монтирование файловой системы. Такой модуль имеет соответствующий ему модуль типа `mount`, который будет запущен, как только файловая система станет доступной;
- *target* (цель) — служит для логической группировки модулей других типов. Этот тип модуля очень важен, но в то же время он ничего не делает, а просто группирует другие модули. В `systemd` больше нет уровней запуска (которые были в `init`), вместо них используются цели. Например, цель `multi-user.target` описывает, какие сервисы (точнее модули, а не только сервисы) должны быть запущены во многопользовательском режиме. По сути, цель `multi-user.target` аналогична 3-му уровню запуска;
- *snapshot* (снимок) — также ничего не делает, а только ссылается на другие модули. Снимки используются в двух случаях: первый случай — временный перевод системы в какое-то состояние (например, в однопользовательский режим) и последующий возврат из этого состояния, второй — поддержка режима `suspend`. Многие демоны не могут правильно переходить в этот режим, поэтому в ряде случаев их лучше остановить и запустить заново, после того как система проснется.

22.3.3. Основные особенности systemd

Перечислим основные особенности `systemd`:

- позволяет контролировать для каждого процесса: среду исполнения, ограничение ресурсов, рабочий каталог, корневой каталог, `umask`, параметр `nice`, ID пользователя и группы и многое другое;

¹ openSuSE 12.1 и Fedora 16 — первые версии дистрибутивов openSUSE и Fedora с `systemd`.

- синтаксис файлов конфигурации `systemd` очень похож на синтаксис файлов `.desktop` и поэтому будет знаком многим Linux-пользователям;
- П наличие совместимости со сценариями `SysV` (правда, не могу пока сказать — временной или постоянной). Если есть старая (например, `/etc/init.d/avahi`) и новая (`/etc/systemd/system/avahi.service`) конфигурации, то используется новая¹;
- для удобства и обратной совместимости поддерживается и обрабатывается файл `/etc/fstab`;
- совместимость с `/dev/initctl`. На практике это означает, что многие системные команды вроде `poweroff`, `shutdown` будут работать с новой системой `systemd`;
- монтирование виртуальных файловых систем, установка имени узла — все это и многое другое делается теперь без `shell`-сценариев;
- состояние сервиса может контролироваться через `D-Bus`.

Конечно, это далеко не все особенности `systemd`, но остальные знать и не обязательно. Далее мы рассмотрим основные отличия `init` и `systemd`, а потом уже перейдем к практическому использованию последней.

22.3.4. Сравнение `init`, `upstart` и `systemd`

Сотрудник компании Red Hat Леннарт Поттеринг опубликовал развернутое сравнение трех основных систем инициализации: `init`, `upstart` и `systemd`. С оригиналом статьи можно ознакомиться по адресу: <http://Opointer.de/blog/projects/why.html>. Информации там довольно много, поэтому самое основное я вычленил и представил в табл. 22.1.

Таблица 22.1. Сравнение систем инициализации

Возможность	Init	Upstart	Systemd
Сервисы			
Совместимость с SysV	+	+	+
Управление SysV-сервисами как родными сервисами	+	-	+
Управление сервисами с помощью /dev/initctl	+	-	+
Контролируемый останов сервисов	-	-	+
Перезапуск сервисов с сериализацией состояния	+	-	+
Отключение сервисов без редактирования файлов	+	-	+
Отправка сигналов сервисам	-	-	+
Перезапуск сервисов при их крахе без потери соединения	-	-	+

¹ Даже в самых новых версиях дистрибутивов — например, в той же Ubuntu 17.04, поддерживается совместимость с `init`, и если вы зайдете в каталог `/etc/init.d`, то увидите там немало соответствующих сценариев.

Таблица 22.1 (продолжение)

Возможность	Init	Upstart	Systemd
Поддержка сервисов типа «instantiated»	-	+	+
Показывает все процессы, принадлежащие сервису	-	-	+
Идентификация процессов сервиса	-	-	+
Активация сервисов на основе сокетов	-	-	+
Система			
Запуск без shell/bash-сценариев	-	-	+
Сервисы ранней стадии загрузки написаны на C	-	-	+
Поддержка D-Bus	-	+	+
Упреждающее чтение данных с диска	-	-	+
Активация на основе «железа» компьютеров	-	-	+
Настройка зависимостей устройств с использованием правил udev	-	-	+
Активация по времени	-	-	+
Управление точками монтирования, в том числе и автмонтирование	-	-	+
Запуск fsck	-	-	+
Управление квотами	-	-	+
Управление swap-разделами	-	-	+
Управление локалью, изменение настроек клавиатуры и консоли	-	-	+
Поддержка контейнеров (как замена chroot())	-	-	+
Сохранение снимков состояния системы (snapshots)	-	-	+
Средства создания, удаления и чистки временных файлов	-	-	+
Ядро и протоколирование			
Поддержка статической загрузки модулей ядра	-	-	+
Поддержка перезапуска ядра на лету (kexec())	-	-	+
Минимальный демон протоколирования на базе kmsg для встраиваемых систем	-	-	+
Поддержка протоколирования в utmp/wtmp	+	+	+
Поддержка раннего протоколирования через /dev/log	-	-	+
Безопасность			
Интеграция с Linux Control Groups (cgroups)		-	+
Генерация событий аудита для запускаемых сервисов	-	-	+
Поддержка PAM	-	-	+

Таблица 22.1 (окончание)

Возможность	Init	Upstart	Systemd
Поддержка SELinux	-	-	+
Управление зашифрованными разделами и дисками (LUKS)	-	-	+
Сохранение и восстановление последовательности генератора случайных чисел (random seed)	-	-	+
Интеграция с PolicyKit	-	-	+
Обработка паролей к LUKS и SSL-сертификатам	-	-	+
Сеть			
Управление loopback	-	-	+
Управление уникальным ID компьютера	-	-	+
Управление динамическим именем компьютера	-	-	+
Совместимость с inetd	-	-	+

Исходя из данных таблицы, видно, что `systemd` — современная система инициализации для современного дистрибутива Linux. При этом, если взглянуть на некоторые возможности, которые есть у `init` и отсутствуют у `upstart`, становится понятно, почему `init` так долго оставалась «в деле».

22.3.5. Немного практики

Теперь, когда вы знакомы с основами `systemd` и с типами ее модулей, можно посмотреть на нее изнутри, а именно — глазами пользователя дистрибутива Fedora 26. В качестве подопытного дистрибутива я выбрал Fedora не просто так— `systemd` появилась в нем раньше, чем в других дистрибутивах. В том же `openSUSE` все будет примерно так же, за исключением более богатого состава каталога `/etc/init.d`, но, по сути, эти сценарии относятся к старой системе `SysV` и не имеют отношения к `systemd`.

Начнем с файла `/etc/inittab`, который в случае с `systemd` представляет собой обычный текстовый файл (рис. 22.1) — простой набор комментариев, никак не влияющий на поведение системы (листинг 22.2).

Листинг 22.2. Файл `/etc/inittab`

```
# inittab is no longer used.
#
# ADDING CONFIGURATION HERE WILL HAVE NO EFFECT ON YOUR SYSTEM.
#
# Ctrl-Alt-Delete is handled by /usr/lib/systemd/system/ctrl-alt-del.target
#
```

systemd uses 'targets' instead of runlevels. By default, there are two main targets:

```
#
# multi-user.target: analogous to runlevel 3
# graphical.target: analogous to runlevel 5
#
# To view current target, run
# systemctl get-default
#
# To set a default target, run:
# systemctl set-default TARGET.target
#
```

Первые две строки здесь сообщают, что этот файл больше не используется systemd, и добавление в него конфигурации никак не повлияет на систему. Прошу заметить, что в других дистрибутивах — например, в Ubuntu, этого файла может вообще не быть.

Далее (в строке 3) сообщается, что реакция на комбинацию клавиш <Ctrl>+<Alt>+<Delete> содержится в файле /usr/lib/systemd/system/ctrl-alt-del.target. Но по умолчанию нажатие комбинации <Ctrl>+<Alt>+<Delete> обрабатывается на уровне GNOME 3 и теперь трактуется как завершение сеанса, а не просто перезагрузка.

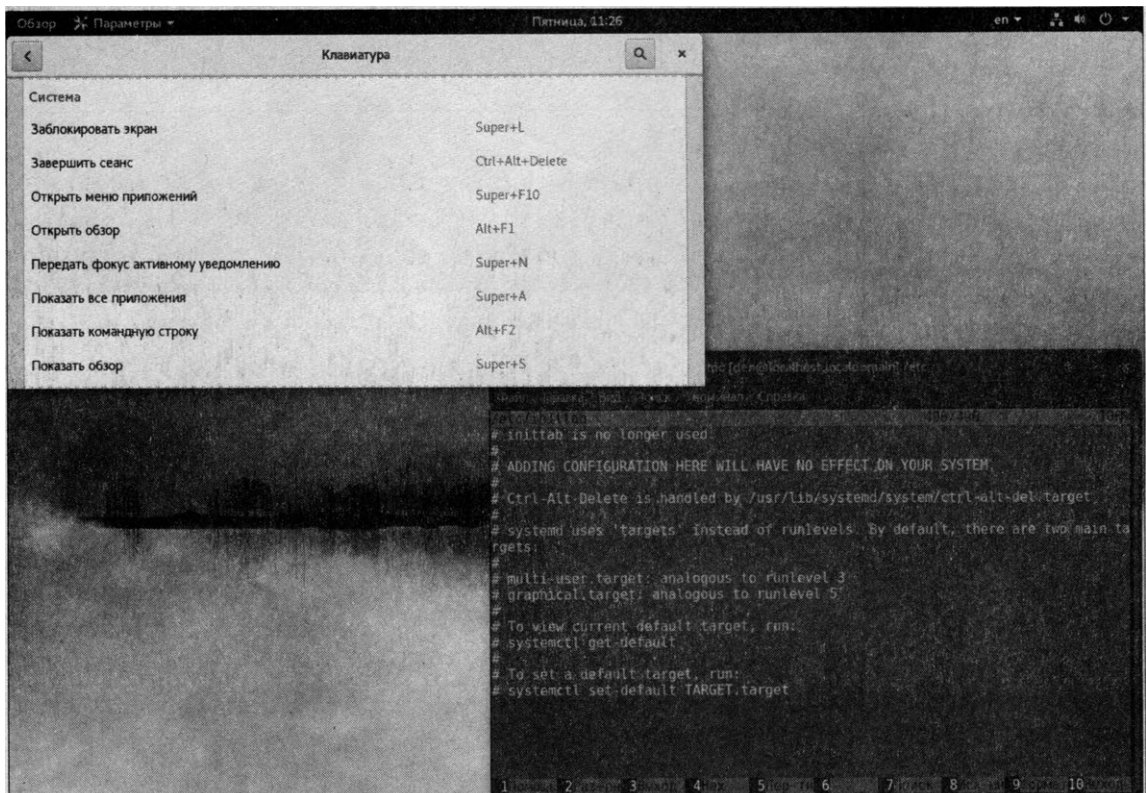


Рис. 22.1. Fedora 26: реакция в GNOME 3 на комбинацию клавиш <Ctrl>+<Alt>+<Delete>

Откройте **Параметры системы**, запустите апплет **Клавиатура**, перейдите на вкладку **Комбинации клавиш**, а затем в группу **Система** (рис. 22.1) — вы увидите описание действия (**Завершить сеанс**) и соответствующую ему комбинацию клавиш. По ее нажатию откроется окошко, предлагающее перезагрузить компьютер или завершить его работу (рис. 22.2). На мой взгляд, такое поведение системы более предпочтительно, чем просто перезагрузка. Чтобы запретить реакцию на эту комбинацию клавиш, щелкните по действию и нажмите клавишу <Пробел>.

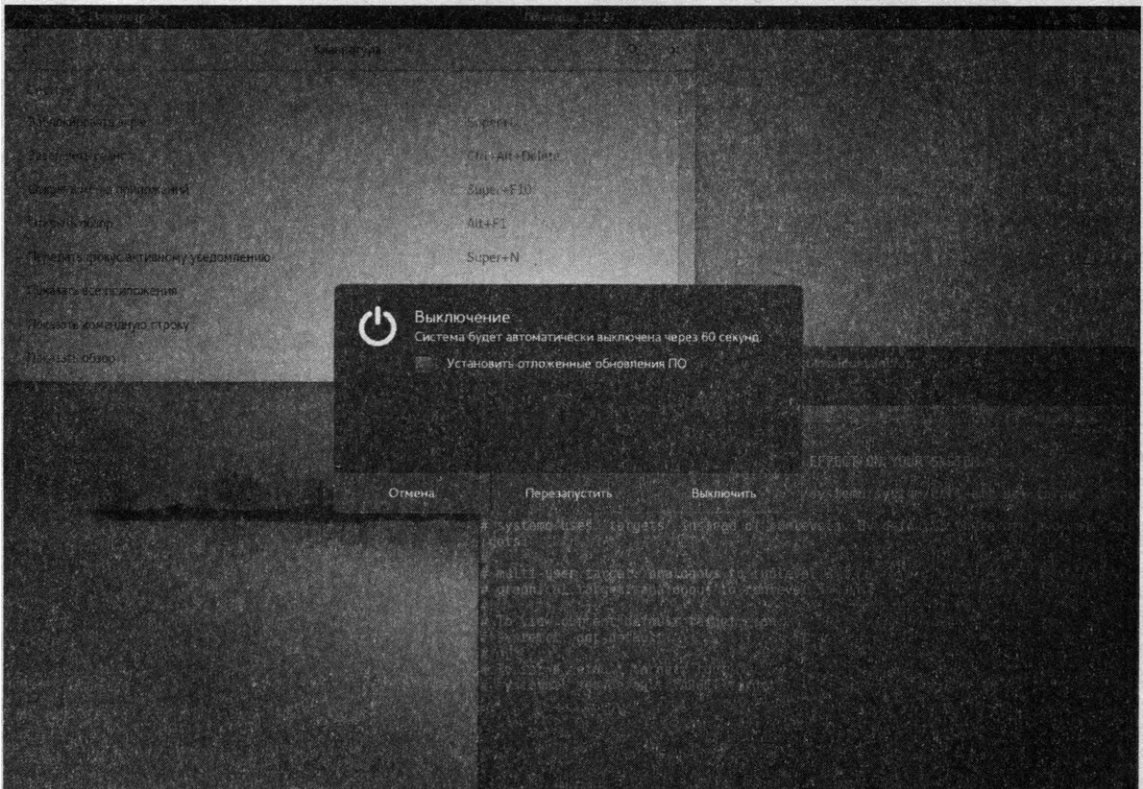


Рис. 22.2. Fedora 26: система отреагировала на нажатие <Ctrl>+<Alt>+<Delete>

Вернемся к файлу `inittab`. В строке 4 там сказано, что вместо уровней запуска `systemd` использует цели. Существуют две основные цели: `multi-user.target` — аналогичная уровню запуска 3, и `graphical.target` — аналогичная 5-му уровню запуска.

Чтобы установить цель по умолчанию, нужно использовать команду:

```
# systemctl set-default TARGET.target
```

По сути, эта команда создает ссылку:

```
# ln -s /lib/systemd/system/имя_цели.target /etc/systemd/system/default.target
```

По умолчанию `default.target` ссылается на `/lib/systemd/system/runlevel5.target`. В свою очередь, `runlevel5.target` — это просто ссылка на `graphical.target`.

Исследуем `systemd` дальше. Зайдите в каталог `/etc/systemd/system` — в нем вы найдете ряд каталогов вида <имя цели>.wants. В этих каталогах имеются ссылки на дру-

гие модули системы `systemd`, и все они ссылаются на файлы в каталоге `/lib/systemd/system`. Зайдите в каталог `multi-user.target.wants` — в нем вы найдете список ссылок на файлы `NetworkManager.service`, `abrt-ccpp.service`, `atd.service` и т. д. Вам это ничего не напоминает? Похоже на содержимое 3-го уровня запуска. Как видите, никаких сложностей.

Далее предлагаю вам продолжить знакомство с системой самостоятельно. Там все очень просто, даже проще, чем вы думаете. Откройте, к примеру, файл `graphical.target` из `/lib/systemd/system` (листинг 22.3).

Листинг 22.3. Файл `graphical.target`

```
[Unit]
Description=Graphical Interface
Documentation=man:systemd.special(7)
Requires=multi-user.target
Wants=display-manager.service
After=multi-user.target rescue.service rescue.target display-manager.service
Conflicts=rescue.service rescue.target
AllowIsolate=yes
```

Рассмотрим самые важные параметры этого файла:

- параметр `Requires` указывает, что описываемая цель требует выполнения цели `multi-user.target` (заменяет 3-й уровень запуска);
- описываемая цель должна быть выполнена после `multi-user.target` (параметр `After`);
- параметр `Conflicts` указывает на цель (или цели — в этом случае они перечисляются через пробел), с которой конфликтует описываемая цель.

Теперь откройте цель `multi-user.target` и посмотрите, от какой цели зависит она. Вы обнаружите, что она зависит от `basic.target`, которая, в свою очередь, зависит от других целей. Просмотрите файлы целей— файл за файлом. Так вы существенно глубже сможете понять новую систему инициализации.

22.3.6. Команды системного администратора

Многие администраторы привыкли использовать команды `service` и `chkconfig`. Они по-прежнему работают в мире `systemd`, но правильнее все же использовать команды `systemd`, представленные в табл. 22.2.

Таблица 22.2. Команды администратора `systemd`

Команда <code>init</code>	Команда <code>systemd</code>	Описание
<code>service s start</code>	<code>systemctl start s.service</code>	Разовый запуск сервиса <code>s</code>
<code>service s stop</code>	<code>systemctl stop s.service</code>	Остановить сервис <code>s</code>
<code>service s restart</code>	<code>systemctl restart s.service</code>	Перезапуск сервиса <code>s</code>

Таблица 22.2 (окончание)

Команда init	Команда systemd	Описание
system s status	systemctl status s.service	Статус сервиса s
ls /etc/rc.d/init.d	ls /lib/systemd/system/*.service /etc/systemd/system/*.service	Получение списка служб
chkconfig s on	systemctl enable s.service	Включает запуск сервиса после перезагрузки (сервис s будет загружаться автоматически)
chkconfig s off	systemctl disable s.service	Отключает автоматический запуск сервиса s
chkconfig s	systemctl is-enabled s.servive	Проверяем, запускается ли s автоматически
chkconfig s --list	ls /etc/systemd/system/*.wants/ s.service	Выводит список целей (уровней запуска), на которых сервис будет запускаться автоматически

Если вас удручают все эти команды, и хочется быстрого и простого средства управления службами, то лучше управляющего службами конфигуратора system-config-services (рис. 22.3) пока еще никто не придумал.

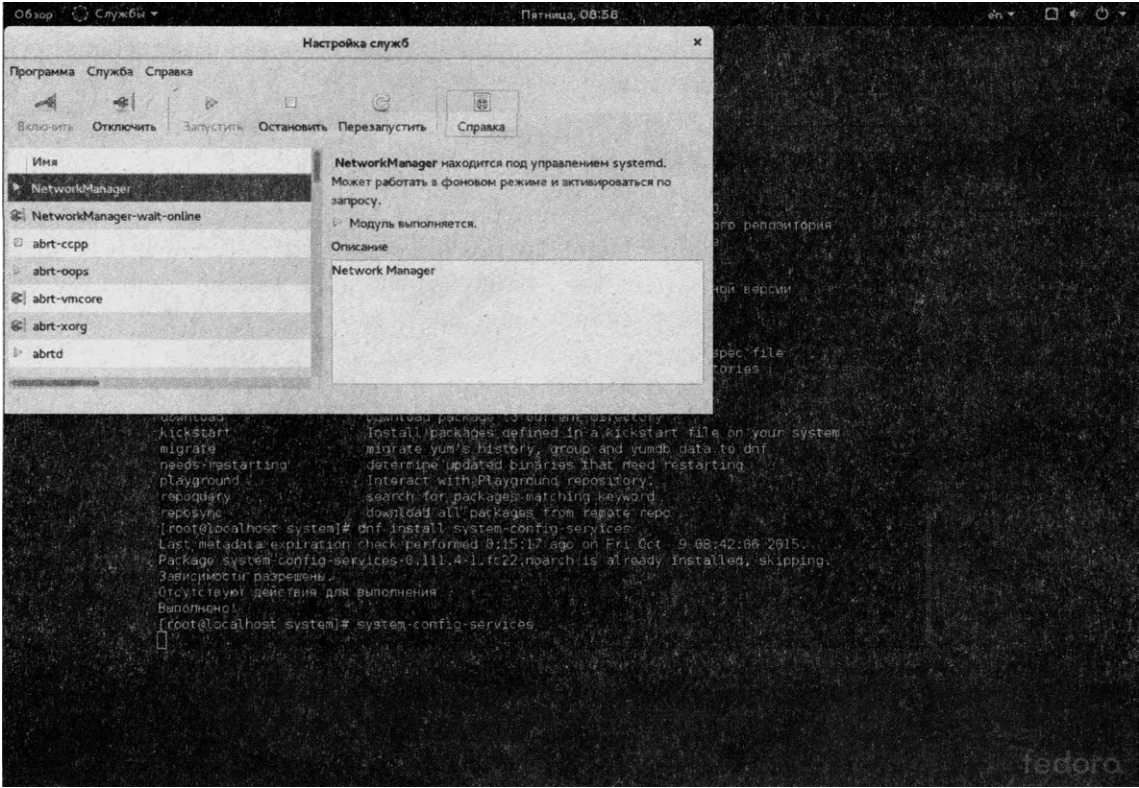


Рис. 22.3. Fedora 22: конфигуратор system-config-services

В старых версиях Fedora такой конфигуратор присутствовал по умолчанию, а в Fedora 22-25 для его установки нужно ввести команду:

```
$ sudo dnf install system-config-services
```

К сожалению, в версии 26 этот конфигуратор был упразднен. Вместо него предлагается только использовать команду `systemctl`.

22.4. Система инициализации Slackware

Система инициализации Slackware отличается от привычной системы `init`, используемой в SysV-системах. Она больше похожа на систему инициализации BSD-систем, хотя некоторые сходства с SysV все же есть.

ЕЩЕ НЕМНОГО ИСТОРИИ

В историческом отступлении в начале главы уже были упомянуты системы инициализации: SysV и BSD. Уточним здесь некоторые связанные с ними моменты. Считается, что UNIX «родилась» в 1969 году. В то время над проектом работали ряд сотрудников компании Bell Labs (одно из подразделений AT&T). Позже UNIX заинтересовались другие организации и, в частности, институт Беркли (Калифорния, США). В 1975 году появилась слегка модифицированная версия UNIX от института Беркли, которая получила название BSD (Berkeley Software Distribution), а версия от AT&T (Bell Labs) стала называться System V (SysV). Обе системы были очень похожи друг на друга, но в то же время имели и свои особенности. Например, BSD содержала собственную систему инициализации, которая очень напоминает ту, что сейчас используется в Slackware Linux.

Если говорить о сходстве систем инициализации в стиле SysV и в стиле BSD, то у обеих систем присутствуют уровни запуска, имеется и файл `/etc/inittab` — таблица инициализации (см. ранее). Однако имена файлов системы инициализации BSD-стиля немного отличаются от имен файлов SysV-стиля.

Система инициализации Slackware построена таким образом, что вне зависимости от уровня запуска первым всегда запускается сценарий `/etc/rc.d/rc.S`. Он монтирует псевдофайловые системы `/proc`, `sysfs` и `devfs`, запускает систему `hotplug` (драйвер устройств, обеспечивающий их «горячее» подключение, т. е. подключение без выключения компьютера, — например USB-устройств), подключает разделы свопинга, монтирует и проверяет корневую файловую систему, монтирует другие файловые системы и т. д. Как видите, сценарий `/etc/rc.d/rc.S` выполняет большую часть действий по инициализации системы. Обычно этот файл не требует изменения, но иногда, все же, его приходится редактировать. Например, если вы создали файл подкачки и хотите, чтобы он подключался при загрузке системы, то команду `swapon <имя_файла>` нужно добавить в файл `/etc/rc.d/rc.S` после команды `/sbin/swapon -a`.

Сценарий `/etc/rc.d/rc.S` проверяет наличие файла `/etc/rc.d/rc.modules.local`, обеспечивающего загрузку модулей при старте системы. При условии, что файл `rc.modules.local` существует, он запускается. В противном случае происходит поиск файла `/etc/rc.d/rc.modules<-версия ядра>`, а если и его нет, тогда сценарий `/etc/rc.d/rc.S` пытается запустить файл `/etc/rc.d/rc.modules`. Один из этих файлов должен существо-

вать, иначе система будет загружена без модулей, а это означает, что не будут работать некоторые устройства и поддерживаться некоторые файловые системы.

Кроме файла `/etc/rc.d/rc.modules.local` или другого файла загрузки модулей (см. ранее) также используется файл `/etc/rc.d/rc.netdevice` — он служит для загрузки модулей сетевых карт (точнее, сетевых интерфейсов).

Как уже было отмечено, файл `/etc/rc.d/rc.S` запускается вне зависимости от уровня запуска. Кроме этого файла в каталоге `etc/rc.d` вы найдете серию файлов `rc.N`, где *N* — номер уровня запуска. Эти файлы запускаются в зависимости от выбранного уровня запуска — например, на третьем уровне запуска будет запущен файл `/etc/rc.d/rc.3`. Каждый такой файл подготавливает систему к работе на выбранном уровне запуска. Уровень запуска по умолчанию, как и в случае с системой инициализации в стиле SysV, задается в файле `/etc/inittab`.

Сценарий `/etc/rc.d/rc.inet1` отвечает за инициализацию сетевых интерфейсов и построение таблицы маршрутизации. Конфигурация сетевых интерфейсов хранится в файле `/etc/rc.d/rc.inet1.conf`. Вот фрагмент этого файла:

```
IPADDR [0] ="192.168.1.1"
NETMASK[0] ="255.255.255.0"
USE_DHCP[0]=""
DHC_P_HOS TNAME[0]=""
```

Сценарий `/etc/rc.d/r.inet2` управляет запуском сетевых служб и подключением сетевых файловых систем. Именно в этом файле происходит попытка монтирования файловых систем NFS и smbfs. Также из этого файла происходит запуск сетевых служб. Сценарии для запуска сетевых служб называются `/etc/rc.d/rc.<название службы>` например, `/etc/rc.d/rc.sshd` — сценарий запуска SSH-сервера. Однако некоторые сетевые сервисы, например `sendmail` и `samba`, в силу своих особенностей запускаются из файлов `rc.N`.

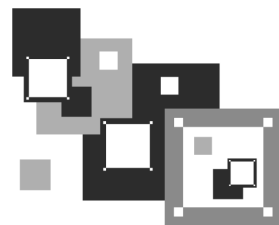
Иногда нужно обеспечить запуск сетевой службы, для которой нет собственного RC-файла. Тогда ее запуск можно или описать в файле `/etc/rc.d/rc.local` (что довольно просто), или создать собственный RC-файл и добавить его вызов в один из файлов `rc.N`. Шаблон собственного RC-файла приведен в листинге 22.4.

Листинг 22.4. Шаблон RC-файла для запуска сетевой службы

```
#!/bin/bash
start()
{
    echo "Service started"
    service_start
}
stop()
{
    echo "Service stoped"
    killall service
}
```

```
case $1 in
  start)
    start ;;
  stop)
    stop ;;
  restart)
    stop
    sleep 2
    start ;;
  *)
    echo "Usage: service start|stop | restart"
esac
```

ГЛАВА 23



Процессы

23.1. Аварийное завершение процесса

Каждому процессу в Linux присваивается уникальный номер— идентификатор процесса (PID. Process ID). Зная ID процесса, вы можете управлять процессом, а именно — завершить процесс или изменить его приоритет. Принудительное завершение процесса необходимо, если процесс завис и его нельзя завершить обычным образом. А изменение приоритета может понадобиться, если вы хотите, чтобы процесс доделал свою работу быстрее.

Предположим, у вас зависла какая-то программа — например, пусть это будет файловый менеджер **mc**. Хотя это и маловероятно (не помню, чтобы он когда-нибудь зависал), но для примера пусть будет так. Принудительно завершить («убить») процесс можно с помощью команды **kill**. Формат ее вызова следующий:

```
kill [параметры] PID
```

PID здесь — это идентификатор процесса (Process ID), который присваивается процессу системой и уникален для каждого процесса. Но мы знаем только имя процесса (имя команды), но не знаем идентификатор процесса. Узнать идентификатор процесса позволяет команда **ps**. Предположим, что **mc** находится на первой консоли. Поскольку он завис, вы не можете более использовать консоль, и вам нужно переключиться на вторую консоль (что можно сделать клавиатурной комбинацией **<Alt>+<F2>**). Зарегистрировавшись на второй консоли, введите команду **ps** — она выведет список процессов, запущенных на второй консоли, — это будут **bash** и сам **ps** (рис. 23.1).

Чтобы добраться до нужного нам процесса (**mc**), который запущен на первой консоли, введите команду **ps -a** или **ps -i root**. В первом случае вы получите список процессов, запущенных вами, а во втором — список процессов, запущенных от вашего имени (здесь мы предполагаем, что вы работаете под именем **root**).

Обратите внимание— вы сами запустили процессы **mc** и **ps** (рис. 23.2), а от вашего имени (**root**) система запустила еще множество процессов. Следует заметить, что команда **ps** выводит также имя терминала (**tty1**), на котором запущен процесс. Это очень важно— если на разных консолях у вас запущены одинаковые процессы, можно легко ошибиться и завершить не тот процесс.

```
Welcome to openSUSE 13.2 "Harlequin" - Kernel 3.16.6-2-desktop (tty2).

linux-wa49 login: root
Password:
Have a lot of fun...
linux-wa49:~ # ps
  PID TTY          TIME CMD
 2730 tty2      00:00:00 bash
 2819 tty2      00:00:00 ps
linux-wa49:~ #
```

Рис. 23.1. Список процессов на текущей консоли

```
Welcome to openSUSE 13.2 "Harlequin" - Kernel 3.16.6-2-desktop (tty2).

linux-wa49 login: root
Password:
Have a lot of fun...
linux-wa49:~ # ps
  PID TTY          TIME CMD
 2730 tty2      00:00:00 bash
 2819 tty2      00:00:00 ps
linux-wa49:~ # ps -a
  PID TTY          TIME CMD
 2818 tty1      00:00:00 mc
 2820 tty2      00:00:00 ps
linux-wa49:~ # _
```

Рис. 23.2. Определение PID программы tc

Теперь, когда мы знаем PID нашего процесса, мы можем его «убить»:

```
# kill 2484
```

Перейдите на первую консоль после выполнения этой команды — `mc` на ней уже не будет. Если выполнить команду `ps -a`, то в списке процессов `tc` тоже не будет.

Проще всего вычислить PID процесса с помощью следующей команды:

```
# ps -ax | grep <имя>
```

Например, `# ps -ax | grep firefox`.

Вообще-то все эти действия, связанные с вычислением PID процесса, мы рассмотрели только для того, чтобы познакомиться с командой `ps`.

Так что, если вы знаете только имя процесса, гораздо удобнее использовать команду:

```
# killall <имя процесса>
```

Но имейте в виду, что такая команда завершит все экземпляры этого процесса. А вполне может быть, что у вас на одной консоли находится `mc`, который нужно «убить», а на другой — нормально работающий `mc`. Команда `killall` «убьет» оба процесса.

При выполнении команд `kill` и `killall` нужно помнить, что если вы работаете от имени обычного пользователя, они могут завершить только те процессы, которые принадлежат вам. А если вы работаете от имени пользователя `root`, то можете завершить любой процесс в системе.

23.2. Программа `top`: кто больше всех расходует процессорное время?

Иногда бывает, что система ужасно тормозит, — весь день работала нормально, а вдруг начала притормаживать.

Если вы даже не догадываетесь, из-за чего это случилось, вам нужно использовать команду `top` (рис. 23.3) — она выводит список процессов с сортировкой по процессорному времени. То есть, на вершине списка будет находиться процесс, который занимает больше процессорного времени, чем сама система. Вероятно, из-за него и происходит эффект «торможения».

На рис. 23.3 показано, что больше всего процессорного времени (0,662%) занимает программа `kworker`.

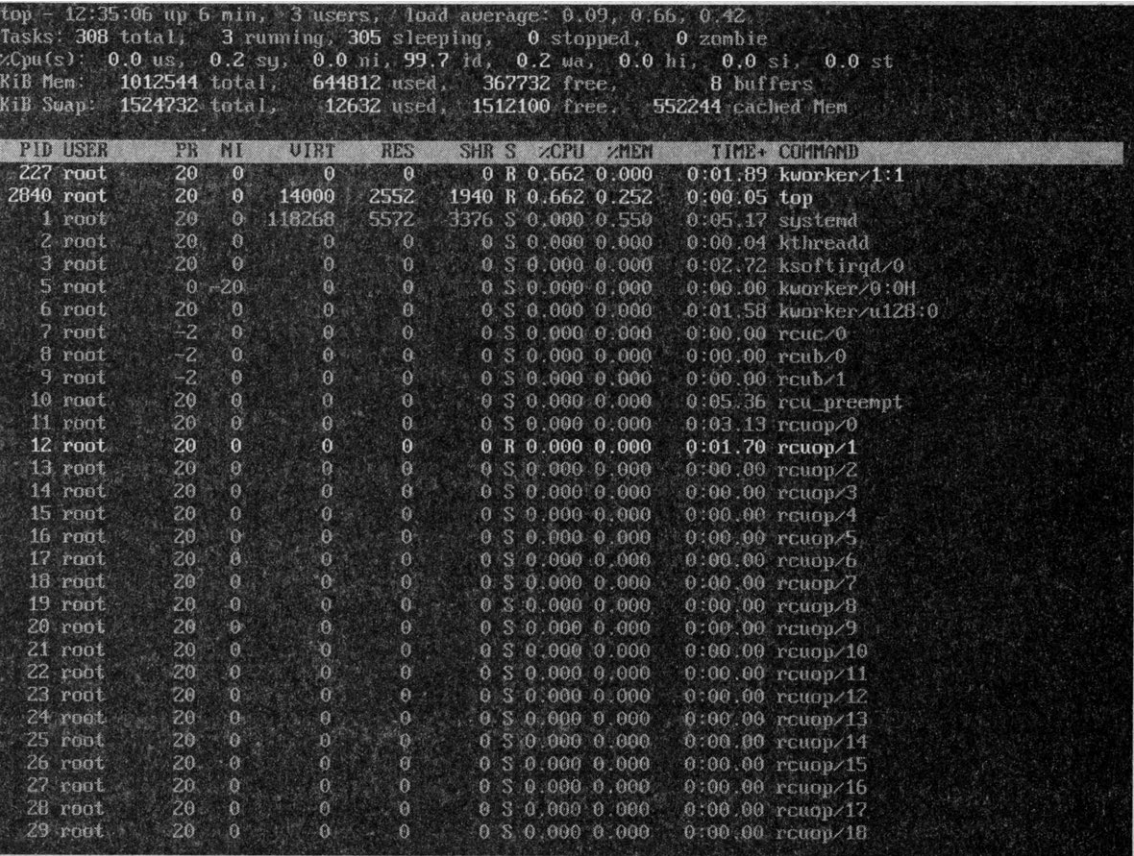


Рис. 23.3. Вывод команды `top`

Выйти из программы `top` можно, нажав клавишу `<Q>`. Кроме команды `<Q>` действуют следующие клавиши:

- `<U>` — показывает только пользовательские процессы (т. е. те процессы, которые запустил пользователь, под именем которого вы работаете в системе);
- `<D>` — изменяет интервал обновления;
- `<F>` — изменяет столбец, по которому сортируются задачи. По умолчанию задачи сортируются по столбцу `%CPU`, т. е. по процессорному времени, занимаемому процессом;
- `<H>` — получить справку по остальным командам программы `top`.

Назначение столбцов программы `top` указано в табл. 23.1.

Таблица 23.1. Назначение столбцов программы `top`

Столбец	Описание
PID	Идентификатор процесса
USER	Имя пользователя, запустившего процесс
PR	Приоритет процесса
NI	Показатель nice (см. <i>разд. 23.3</i>)
VIRT	Виртуальная память, использованная процессом (в Кбайт)
RES	Размер процесса, не перемещенный в область подкачки (в Кбайт). Этот размер равен размерам сегментов кода и данных, т. е. <code>RES = CODE + DATA</code>
S	Состояние процесса: <ul style="list-style-type: none">• R — выполняется;• S — «спит» (режим ожидания), в этом состоянии процесс выгружен из оперативной памяти в область подкачки;• D — «непрерываемый сон» (uninterruptible sleep), из такого состояния процесс может вывести только прямой сигнал от оборудования;• T — процесс в состоянии трассировки или остановлен;• Z (зомби) — специальное состояние процесса, когда сам процесс уже завершен, но его структура еще осталась в памяти
%CPU	Занимаемое процессом процессорное время
%MEM	Использование памяти процессом
TIME+	Процессорное время, израсходованное с момента запуска процесса
COMMAND	Команда, которая использовалась для запуска процесса (обычно имя исполнимого файла процесса)

23.3. Изменение приоритета процесса

Предположим, что вы работаете с видео, и вам нужно перекодировать файл из одного видеоформата в другой. Конвертирование видео занимает много процессорного времени, а хотелось бы все сделать как можно быстрее и уйти пораньше домой. Тогда вам поможет программа `nice` — она позволяет запустить любую программу с указанным приоритетом. Ясно— чем выше приоритет, тем быстрее будет выполняться программа. Формат вызова команды следующий:

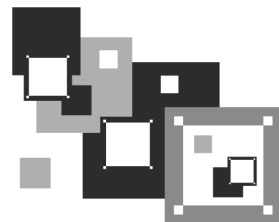
```
nice -n <приоритет> команда аргументы
```

Максимальный приоритет задается числом `-20`, а минимальный — числом `19`. Приоритет по умолчанию равен `10`.

Если процесс уже запущен, тогда для изменения его приоритета можно использовать команду `renice`:

```
renice -n <приоритет> -p PID
```

ГЛАВА 24



Псевдофайловые системы sysfs и proc

Особую роль в Linux играют так называемые *псевдофайловые* системы. Слово «псевдо», как мы знаем, означает «почти», т. е. псевдофайловая система — не совсем файловая система в прямом смысле этого слова. Псевдофайловые системы также называются *виртуальными*, поскольку работают они на уровне виртуальной файловой системы (Virtual File System layer). Для большинства пользователей виртуальная файловая система выглядит как обычная файловая система— можно открыть тот или иной файл и посмотреть, что в нем записано, можно записать информацию в файл. Ради интереса зайдите в каталог `/proc` (это каталог псевдофайловой системы `proc`) и посмотрите на размер любого файла— например, `/proc/filesystems`. Его размер будет равен 0, как и остальных файлов этой файловой системы, но если открыть сам файл, то вы увидите, что информация в нем есть. Это объясняется тем, что содержимое файла формируется при обращении к нему, т. е. «на лету». Другими словами, виртуальная файловая система находится в оперативной памяти, а не на жестком диске. Информация попадает в файл на основании сведений, полученных от ядра.

В большинстве современных дистрибутивов используются виртуальные файловые системы `sysfs` и `proc`. Откройте файл `/etc/fstab`, и вы увидите строки монтирования этих файловых систем:

```
sysfs /sys sysfs defaults 0 0
proc /proc proc defaults 0 0
```

24.1. Виртуальная файловая система sysfs

Виртуальная (псевдофайловая) система `sysfs` экспортирует в пространство пользователя информацию о ядре Linux, об имеющихся в системе устройствах и их драйверах. Впервые `sysfs` появилась в ядре версии 2.6. Зайдите в каталог `/sys` — названия подкаталогов говорят сами за себя:

- **block** — содержит каталоги всех блочных устройств, имеющихся в системе в текущее время (под устройством подразумевается совокупность физического устройства и его драйвера). Когда вы подключаете Flash-диск, то в любом случае в каталоге `/sys/devices/` появляется новое устройство, но в каталоге `/sys/block` это

устройство появится только при наличии соответствующих драйверов (в указанном случае — драйвера `usb-storage`);

- **bus** — перечень шин, поддерживаемых ядром (точнее, зарегистрированных в ядре). В каждом каталоге шины есть подкаталоги **devices** и **drivers**. В каталоге **devices** находятся ссылки на каталоги всех устройств, которые описаны в системе (т. е. находящихся в каталоге `/sys/devices`);
- **class** — по этому каталогу можно понять, как устройства формируются в классы. Для каждого устройства в каталоге **class** есть свой отдельный каталог (под устройством, как и в случае с каталогом **block**, подразумевается совокупность устройства и его драйвера);
- **devices** — содержит файлы и каталоги, которые полностью соответствуют внутреннему дереву устройств ядра;
- **drivers** — каталоги драйверов для загруженных устройств. Подкаталог **drivers** каталога шины содержит драйверы устройств, работающих на этой шине.

24.2. Виртуальная файловая система `proc`

Виртуальная (псевдофайловая) система `proc` — это специальный механизм, позволяющий получать информацию о системе, ядре или процессе и изменять параметры ядра и его модулей «на лету» (кстати, ее название — это сокращение от `process`).

Файлы, позволяющие получать информацию, мы можем только просмотреть, а файлы, с помощью которых можно изменять некоторые параметры системы, — просмотреть и, если нужно, изменить.

Просмотреть информационный файл можно командой `cat`:

```
cat /proc/путь/<название_файла>
```

Записать значение в один из файлов `proc` можно так:

```
echo "данные" > /proc/путь/название_файла
```

24.2.1. Информационные файлы

В табл. 24.1 представлены некоторые (самые полезные) информационные `proc`-файлы — с их помощью вы можете получить информацию о системе.

Таблица 24.1. Информационные `proc`-файлы

Файл	Описание
<code>/proc/version</code>	Содержит версию ядра
<code>/proc/cmdline</code>	Список параметров, переданных ядру при загрузке
<code>/proc/cpuinfo</code>	Информация о процессоре
<code>/proc/meminfo</code>	Информация об использовании оперативной памяти (почти то же, что и команда <code>free</code>)

Таблица 24.1 (окончание)

Файл	Описание
/proc/devices	Список устройств
/proc/filesystems	Файловые системы, которые поддерживаются системой
/proc/mounts	Список подмонтированных файловых систем
/proc/modules	Список загруженных модулей
/proc/swaps	Список разделов и файлов подкачки, которые активны в текущий момент

24.2.2. Файлы,
позволяющие изменять параметры ядра

Каталог /proc/sys/kernel содержит файлы, с помощью которых вы можете изменять важные параметры ядра. Конечно, все файлы мы рассматривать здесь не будем, а остановимся лишь на тех, которые используются на практике (табл. 24.2).

Таблица 24.2. Файлы каталога /proc/sys/kernel

Файл	Каталог
/proc/sys/kernel/ctrl-alt-del	Если этот файл содержит значение 0, то при нажатии клавиатурной комбинации <Ctrl>+<Alt>+ будет выполнена так называемая «мягкая перезагрузка», когда управление передается программе системы инициализации и последняя «разгружает» систему, как при вводе команды <code>reboot</code> . Если этот файл содержит значение 1, то нажатие <Ctrl>+<Alt>+ равносильно нажатию кнопки Reset. Сами понимаете, значение 1 устанавливать не рекомендуется
/proc/sys/kernel/domainname	Здесь находится имя домена — например, dkws.org.ua
/proc/sys/kernel/hostname	Содержит имя компьютера, например <code>den</code>
/proc/sys/kernel/panic	При критической ошибке ядро «впадает в панику» — работа системы останавливается, а на экран выводится надпись kernel panic и текст ошибки. Указанный файл содержит значение в секундах, в течение которого система подождет, пока пользователь прочитает это сообщение, после чего компьютер будет перезагружен. Значение 0 (по умолчанию) означает, что перезагружать компьютер вообще не нужно
/proc/sys/kernel/printk	Этот файл позволяет определить важность сообщения об ошибках. По умолчанию файл содержит значения 6 4 17. Это означает, что сообщения с уровнем приоритета 6 и ниже (чем ниже уровень, тем выше важность сообщения) будут выводиться на консоль. Для некоторых сообщений об ошибках уровень приоритета не задается. Тогда нужно установить уровень по умолчанию. Это как раз и есть второе значение — 4. Третье значение — это номер самого максимального приоритета, а последнее значение задает, значение по умолчанию для первого значения. Обычно изменяют только первое значение, чтобы определить, какие значения должны быть выведены на консоль, а какие — попасть в журнал демона <code>syslog</code>

24.2.3. Файлы, изменяющие параметры сети

В каталоге `/proc/sys/net` вы найдете файлы, изменяющие параметры сети (табл. 24.3).

Таблица 24.3. Файлы каталога `/proc/sys/net`

Файл	Описание
<code>/proc/sys/net/core/message_burst</code>	Опытные системные администраторы используют этот файл для защиты от атак на отказ (DoS). Один из примеров DoS-атаки — когда система заваливается сообщениями атакующего, а полезные сообщения системой игнорируются, потому что она не успевает реагировать на сообщения злоумышленника. В указанном файле содержится значение времени (в десятых долях секунды), необходимое для принятия следующего сообщения. Значение по умолчанию — 50 (5 секунд). Сообщение, попавшее в «перерыв» (в эти 5 секунд), будет проигнорировано
<code>/proc/sys/net/core/message_cost</code>	Чем выше значение в этом файле, тем больше сообщений будет проигнорировано в перерыв, заданный файлом <code>message_burst</code>
<code>/proc/sys/net/core/netdev_max_backlog</code>	Задаёт максимальное число пакетов в очереди. По умолчанию — 300. Используется, если сетевой интерфейс передает пакеты быстрее, чем система может их обработать
<code>/proc/sys/net/core/optmem_max</code>	Задаёт максимальный размер буфера для одного сокета

24.2.4. Файлы, изменяющие параметры виртуальной памяти

В каталоге `/proc/sys/vm` вы найдете файлы, с помощью которых можно изменить параметры виртуальной памяти:

- в файле `buffermem` находятся три значения (разделяются пробелами): минимальный, средний и максимальный объем памяти, которую система может использовать для буфера. Значения по умолчанию: 2 10 60;
- в файле `kswapd` тоже есть три значения, которые можно использовать для управления подкачкой:
 - первое значение задает максимальное количество страниц, которые ядро будет пытаться переместить на жесткий диск за один раз;
 - второе значение — минимальное количество попыток освобождения той или иной страницы памяти;
 - третье значение задает количество страниц, которые можно записать за один раз. Значения по умолчанию: 512 32 8.

24.2.5. Файлы, позволяющие изменить параметры файловых систем

Каталог `/proc/sys/fs` содержит файлы, изменяющие параметры файловых систем. В частности:

- файл **file-max** задает максимальное количество одновременно открытых файлов (по умолчанию: 4096);
- в файле **inode-max** содержится максимальное количество одновременно открытых индексных дескрипторов — *инодов* (максимальное значение также равно 4096);
- в файле **super-max** находится максимальное количество используемых супер-блоков;

ПОЯСНЕНИЕ

Поскольку каждая файловая система имеет свой суперблок, легко догадаться, что количество подмонтируемых файловых систем не может превысить значение из файла **super-max**, которое по умолчанию равно 256, чего в большинстве случаев вполне достаточно. Наоборот, можно уменьшить это значение, чтобы никто не мог подмонтировать больше файловых систем, чем нужно (если монтирование файловых систем разрешено обычным пользователям).

- в файле **super-nr** находится количество открытых суперблоков в текущий момент. Этот файл нельзя записывать, его можно только читать.

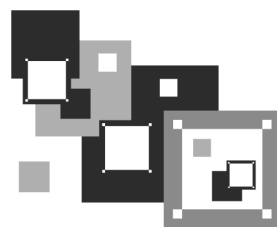
24.3. Сохранение произведенных изменений

Итак, вы изменили некоторые параметры системы с помощью файлов каталога `/proc`. Чтобы сохранить измененные параметры, их следует прописать в файле `/etc/sysctl.conf`. Вот только формат этого файла следующий: надо отбросить `/proc/sys/` в начале имени файла, все, что останется, записать через точку, а затем через знак равенства указать значение параметра. Например, для изменения параметра `/proc/sys/vm/buffermem` нужно в файле `etc/sysctl.conf` прописать строку:

```
vm.buffermem = 2 11 60
```

Если в вашем дистрибутиве нет файла `/etc/sysctl.conf`, тогда пропишите команды вида `echo "значение" > файл` в сценарий инициализации системы.

ГЛАВА 25



Команды Linux, о которых нужно знать каждому линуксоиду

В этой главе мы рассмотрим команды, которые нужно знать каждому пользователю Linux. Для большего удобства команды разбиты на группы: общие команды, команды для работы с текстом, команды для работы в Интернете и команды системного администратора.

25.1. Общие команды

25.1.1. Команда *arch*— вывод архитектуры компьютера

Эта команда поможет узнать тип аппаратной платформы— например: i386, i586, i686 и др.

Пример использования:

```
$ arch  
i686
```

25.1.2. Команда *clear*— очистка экрана

Команда *clear* очищает экран при работе в консоли (терминале).

Пример использования:

```
$ clear
```

25.1.3. Команда *date*

Команда *date* служит для вывода текущей даты. Эта команда может применяться также для установки даты, если запущена от имени администратора.

Пример использования:

```
$ date  
# date 1609161707
```

Первая команда выводит дату, а вторая — устанавливает дату (при условии, что команда запущена от имени *root*) 16 сентября (1609) 2016 года (16) и время 17:17.

Как видите, установка даты осуществляется в формате MMddhhmmYY (MM — месяц, dd — число, hh — часы, mm — минуты, YY — год).

Команда `date` может вывести дату и в указанном вами формате. Чтобы посмотреть варианты возможных форматов даты, введите команду `man date`.

25.1.4. Команда *echo*

Команда `echo` выводит текстовую строку, указанную в качестве аргумента — например:

```
$ echo "Hello world!"
```

Hello world!

Обычно эта команда используется в сценариях командного интерпретатора для вывода сообщений на экран.

25.1.5. Команда *exit* — выход из системы

Для завершения сеанса работы в системе (при условии, что вы работаете в консоли) нужно использовать команду `exit`. Не забывайте отдавать эту команду, уходя со своего рабочего места, — если не завершить сеанс работы, во время вашего отсутствия за компьютером кто угодно сможет работать в системе под вашим именем.

25.1.6. Команда *man* — вывод справки

Команда `man` служит для получения справки о любой команде системы. Например, команда `man ls` выведет справку об использовании команды `ls`, которая выводит содержимое каталога. О том, как правильно пользоваться самой справочной системой, вам расскажет команда `man man`.

25.1.7. Команда *passwd* — изменение пароля

С этой командой мы уже знакомы. Она обеспечивает изменение пароля пользователя, который ее запустил. Суперпользователь `root` имеет право изменить пароль любого пользователя:

```
# passwd имя_пользователя
```

25.1.8. Команда *startx* — запуск графического интерфейса X.Org

Linux может запускаться на разных уровнях запуска. На пятом уровне запуска графический интерфейс X.Org (старое название: X Window) запускается автоматически (если он вообще был установлен). На третьем же уровне запуск графического интерфейса не производится. Если он вам, тем не менее, нужен, то его можно запустить с помощью команды `startx`. Никаких параметров не требуется.

25.1.9. Команда *uptime* — информация о работе системы

Команда *uptime* (рис. 25.1) выводит статистическую информацию о работе системы: сколько времени прошло с момента последней перезагрузки (собственно, это и есть время «uptime»), сколько пользователей в текущий момент подключено к системе, и среднюю загрузку системы за последние 1, 5 и 15 минут.

```
[den@localhost ~]$ uptime  
13:02:26 up 6 min, 3 users, load average: 0.57, 0.81, 0.44  
[den@localhost ~]$
```

Рис. 25.1. Команда *uptime*

25.1.10. Команда *users* — информация о пользователях

Команда *users* выводит информацию о пользователях, подключенных к системе в текущий момент.

На рис. 25.2 видно, что пользователь *den* подключился к системе двумя способами: вошел в консоли и в графическом режиме (или по FTP, *ssh*, *telnet* — способы подключения к системе могут быть разными).

```
[den@localhost ~]$ users  
den den  
[den@localhost ~]$
```

Рис. 25.2. Команда *users*

25.1.11. Команды *w*, *who* и *whoami* — информация о пользователях

Эти три родственные команды выводят следующую информацию (рис. 25.3):

- команда *w* — список пользователей, подключенных к системе, виртуальный терминал, с которого работает пользователь, время входа в систему для каждого пользователя, статистику использования системы (*IDLE* — время простоя, *JCPU* — использование процессора), выполняемые каждым пользователем задачи;
- команда *who* — список пользователей, подключенных к системе, время и дату входа каждого пользователя;
- команда *whoami* — имя пользователя, который ввел команду.

```
[den@localhost ~]$ w
13:04:08 up 7 min, 3 users, load average: 0,18, 0,60, 0,41
USER      TTY      LOGIN@   IDLE   JCPU   PCPU WHAT
den       pts/0    12:59    4:54   0.00s  1.14s kded [kdeinit] --new-startup
den       pts/1    13:02    0.00s  0.15s  0.01s w
[den@localhost ~]$ who
den       pts/0    2007-07-23 12:59
den       pts/1    2007-07-23 13:02
[den@localhost ~]$ whoami
den
[den@localhost ~]$ █
```

Рис. 25.3. Команды `w`, `who` и `whoami`

25.1.12. Команда `xf86config` — настройка графической подсистемы

Текстовый конфигуратор системы X.Org. Использовать его нужно, только если в вашем дистрибутиве нет более удобных графических или псевдографических конфигураторов.

25.2. Команды для работы с текстом

25.2.1. Команды `diff` и `cmp` — сравнение файлов

Команда `diff` служит для сравнения двух файлов. Формат вызова команды:

`diff` параметры файл1 файл2

В выводе команды отличающиеся строки помечаются символами `>` и `<`:

- строка из первого файла помечается символом `<`;
- строка из второго файла — символом `>`.

Самые полезные параметры команды `diff` приведены в табл. 25.1.

Таблица 25.1. Некоторые параметры команды `diff`

Параметр	Описание
-b	Игнорируются пробельные символы в конце строки
-B	Игнорируются пустые строки
-e	Используется при создании сценария для редактора ed. Этот сценарий превращает первый файл во второй
-w	Игнорируются пробельные символы
-y	Вывод в два столбца
-r	Используется для сравнения файлов в подкаталогах. Вместо первого файла указывается первый каталог, вместо второго файла — соответственно, второй каталог

Команда `cmp` также служит для сравнения двух файлов: если файлы идентичны, то стар вообще ничего не выводит, а вот если файлы отличаются, то `cmp` выводит номер строки и номер символа в строке, откуда начинается различие.

Команда `cmp` более универсальна, поскольку она может использоваться для сравнения как текстовых, так и двоичных файлов. В отличие от нее, команда `diff` и ее аналоги умеют сравнивать только текстовые файлы.

Формат вызова команды `cmp` следующий:

```
cmp [параметры] файл1 файл2
```

Параметры команды `cmp` указаны в табл. 25.2.

Таблица 25.2. Параметры команды `cmp`

Параметр	Описание
<code>-c</code>	Вывод отличающихся символов
<code>-i n</code>	Игнорировать первые <code>n</code> символов
<code>-l</code>	Вывод позиций всех отличий, а не только первого
<code>-s</code>	Не выводить информацию на экран, при этом код возврата будет следующим: 0 — файлы одинаковые; 1 — файлы отличаются; 2 — ошибка при открытии одного из файлов

25.2.2. Команды *grep* и *egrep* — текстовый фильтр

Предположим, у вас есть файл протокола `/var/log/messages`, и вы хотите вывести все сообщения, связанные с демоном `pppd`. Понятно, что вручную выделить все нужные сообщения будет весьма трудно. Но с помощью `grep` эту задачу можно автоматизировать:

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

Команда `cat /var/log/messages` передаст содержимое файла `/var/log/messages` на стандартный ввод команды `grep`, которая, в свою очередь, выделит строки, содержащие строку `ppp`.

КОМАНДА *tac*

Вообще-то, просматривать журналы удобнее с помощью команды `tac`, которая выводит строки файла в обратном порядке, — ведь сообщения дописываются в конец журнала, следовательно, если выводить строки в обратном порядке, то сначала мы получим самые новые сообщения, а потом уже все остальные:

```
tac /var/log/messages | grep ppp
```

Команда `egrep` похожа на команду `grep`, но считается более быстрой и более функциональной. Если файлы не заданы, то программа читает текст из стандартного ввода.

Формат вызова команды `egrep`:

`egrep [параметры] строка файлы`

Параметры команды `egrep` приведены в табл. 25.3.

Таблица 25.3. Параметры команды `egrep`

Параметр	Описание
<code>-A n</code>	Вывод <code>n</code> строк после строки, в которой есть искомая строка
<code>-B n</code>	Вывод <code>n</code> строк перед строкой, содержащей искомую строку
<code>-b</code>	Выводит для каждой строки файла, где есть искомая строка, ее положение в файле
<code>-c</code>	Выводит количество совпадений, но не выводит сами совпадения
<code>-C</code>	Выводит две строки до и две строки после строки, которая содержит искомую строку
<code>-e строка</code>	Используйте этот параметр, если искомая строка начинается с символа «-»
<code>-f файл</code>	Производит поиск искомых строк, которые имеются в указанном файле
<code>-h</code>	Выводит строки, содержащие искомую строку, но не выводит имена содержащих их файлов
<code>-i</code>	Игнорировать регистр букв
<code>-n</code>	Выводит номера строк (и сами строки), содержащих искомую строку
<code>-s</code>	Не выводить сообщения об ошибке, если некоторые файлы не могут быть открыты
<code>-w</code>	Поиск совпадения целого слова с искомой строкой
<code>-X</code>	Поиск совпадения целой строки с искомой строкой

Пример использования:

```
egrep "ppp [11]" *
```

Эта команда ищет строку, заключенную в кавычки, во всех файлах в текущем каталоге.

25.2.3. Команды *more* и *less* — постраничный вывод

Большой текстовый файл намного удобнее просматривать с помощью команд `less` или `more`. Программа `less` удобнее, чем `more`, если она есть в вашей системе:

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

25.2.4. Команды *head* и *tail* — вывод начала и хвоста файла

Команда `head` выводит первые десять строк файла, а `tail` — последние десять. Количество строк может регулироваться с помощью параметра `-n`.

Пример использования:

```
head -п 10 /var/log/messages
tail -n 15 /var/log/messages
```

25.2.5. Команда *wc* — подсчет слов в файле

Команда *wc* используется:

- для подсчета слов в текстовом файле:
`wc /var/log/messages`
- для подсчета количества строк (если задан параметр `-l`):
`wc -l /var/log/messages`
- для подсчета количества символов (параметр `-c`):
`wc -c /var/log/messages`

25.2.6. Команды *vi*, *nano*, *ee*, *mcedit*, *pico* — текстовые редакторы

Со времен первых версий UNIX в современные системы перекочевал текстовый редактор *vi*. То, что ему больше тридцати лет, — видно сразу. Более неудобного редактора я не знаю! Согласен, что тогда это был прорыв, но сегодня редактор смотрится уж очень архаично.

Некоторые гурманы (я бы их назвал мазохистами) говорят, что к нему нужно привыкнуть. Может и так, но сначала нужно изучить длинное руководство (*man*) и выучить команды редактора наизусть, поскольку как такового интерфейса пользователя у этого редактора практически нет — то, что есть, сложно назвать интерфейсом. Однако в этой книге мы рассмотрим *vi*, хотя бы вкратце. Тому есть две причины. Первая — это критики. Мол, как это в главе, посвященной командной строке, не будет «классики»? Вторая — существуют системы, где по непонятным мне причинам до сих пор используется по умолчанию *vi*, а другие редакторы недоступны. Да, можно изменить переменную окружения `EDITOR`, но нет никакой гарантии, что в системе будет установлен какой-нибудь другой редактор.

На рис. 25.4 представлен редактор *vi*, в который загружен файл `/etc/passwd`.

Итак, приступим к рассмотрению редактора *vi*. Он может работать в трех режимах:

- основной (визуальный) режим — в нем и осуществляется редактирование текста;
- командный режим — в нем осуществляется ввод специальных команд для работы с текстом (если сравнивать *vi* с нормальным редактором, то этот режим ассоциируется с меню редактора, где есть команды типа **Сохранить**, **Выйти** и т. д.);
- режим просмотра — используется только для просмотра файла (если надумаете использовать этот режим, вспомните про команду `less`).

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin)/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101:/var/lib/libuuid:/bin/sh
syslog:x:101:102:/home/syslog:/bin/false
messagebus:x:102:106:/var/run/dbus:/bin/false
nslcd:x:103:7:NSLCD system user:/var/run/nslcd:/bin/false
avahi-autoipd:x:104:110:Avahi autoip daemon:/var/lib/avahi-autoipd:/bin/false
/etc/passwd" [readonly] 33 lines, 1617 characters
```

Рис. 25.4. Редактор vi

После запуска редактора вы можете переключать режимы (как, будет сказано позже), но выбрать режим можно и при запуске редактора:

```
vi файл
vi -e файл
vi -R файл
```

Первая команда запускает vi и загружает файл. Вторая — запускает vi в командном режиме и загружает файл. Третья — запускает режим просмотра файла. Если указанный файл не существует, то он будет создан. По умолчанию активируется именно командный режим, поэтому в ключе -e смысла нет.

После запуска vi главное — знать, как из него выйти. Ведь в нем не будет привычной строки меню, редактор также не реагирует на привычные комбинации клавиш вроде <Alt>+<X>. Комбинация <Ctrl>+<C> тоже не поможет.

В табл. 25.4 приведены основные команды редактора vi.

Таблица 25.4. Основные команды редактора vi

Команда	Описание
: q!	Выход без сохранения
: w	Сохранить изменения
:w <файл>	Сохранить изменения под именем <файл>
: wq	Сохранить и выйти
:q	Выйти, если нет изменений

Таблица 25.4 (окончание)

Команда	Описание
i	Перейти в режим вставки символов в позицию курсора
a	Перейти в режим вставки символов в позицию после курсора
o	Вставить строку после текущей
O	Вставить строку над текущей
X	Удалить символ в позицию курсора
dd	Удалить текущую строку
u	Отменить последнее действие

Команды, которые начинаются с двоеточия, будут отображены в нижней строке, остальные просто выполняются, но не отображаются. Как уже было отмечено, у редактора `vi` есть два основных режима (режим просмотра мы не учитываем): режим команд и режим редактирования (визуальный). Переключение в режим команд осуществляется нажатием клавиши `<Esc>`. Нажатие клавиш `<i>`, `<a>` и других переключает редактор в режим вставки, когда набираемые символы трактуются именно как символы, а не как команды. Для переключения обратно в командный режим надо снова воспользоваться клавишей `<Esc>`. В некоторых случаях (например, когда вы пытаетесь передвинуть курсор левее первого символа в строке) переход в командный режим осуществляется автоматически.

Теперь немного практики. Введите команду:

```
$ vi file.txt
```

Нажмите клавишу `<i>`, чтобы переключиться в режим вставки. Наберите любой текст, но постарайтесь не ошибаться, поскольку исправление ошибок в `vi` — дело, требующее отдельного разговора. Затем нажмите клавишу `<Esc>` и введите команду `:wq`. После выхода из редактора введите команду:

```
cat file.txt
```

Так вы убедитесь, что файл создан, и в нем сохранен введенный вами текст. Теперь приступим к дальнейшему рассмотрению редактора. Если ввести не команду `i`, а команду `a`, то вы тоже перейдете в режим вставки, но с одним отличием — вводимый текст будет вставляться не перед символом, в котором находится курсор, а после него. Также в режим вставки можно перейти командами `o` и `O`. В первом случае добавится пустая строка после текущей строки, а во втором — перед текущей строкой, и весь дальнейший ввод будет восприниматься именно как ввод текста, а не команд.

Чтобы удалить символ, нужно перейти в режим команд и над удаляемым символом нажать клавишу `<x>`. Да, клавиши `<Backspace>` и `<Delete>` тут не работают. Точнее, `<Backspace>` работает, но для удаления последней непрерывно введенной последовательности символов. Например, у нас есть текст: `vi` — текстовый редактор. Вы

перейдете в режим вставки и измените текст так: `vi` – неудобный текстовый редактор. Нажатие клавиши `<Backspace>` удалит слово `неудобный`, но не сможет удалить дефис и другие символы.

Чтобы удалить строку, в которой находится курсор, нужно использовать команду `dd`. Помните, что `vi` считает строкой не то, что вы видите на экране, а последовательность символов до первого символа новой строки (`\n`). Если строка длиннее 80 символов, то она переносится на две экранные строки и визуально выглядит как две строки, а не как одна.

Чтобы перейти в конец строки (клавиши `<Home>` и `<End>` тоже не работают, как вы успели заметить, если уже запускали `vi`), нужно ввести команду `$`. При навигации курсор перемещается не по экранным линиям, а как раз по строкам текста.

Для отмены последней операции служит команда `u`. Вот только истории изменений нет, да и по команде `u` отменяется вся предыдущая команда целиком. Например, вы создали файл, перешли в режим вставки (командой `i`) и ввели весь текст Большой медицинской энциклопедии. Если вы введете команду `u`, то она отменит всю предыдущую команду, т. е. удалит весь введенный вами текст. Так что, будьте осторожны.

Все — азы `vi` я вам преподнес. Но не думаю, что вы будете им пользоваться. Если есть желание продолжить знакомство, введите команду:

```
man vi
```

А мы тем временем познакомимся с другими текстовыми редакторами. Самый удобный из известных мне текстовых редакторов — `nano` (рис. 25.5). Раньше он назывался `pine` и входил в состав почтового клиента `pine`.



Рис. 25.5. Редактор nano

Внизу (под текстом) имеется подсказка по комбинациям клавиш для управления редактором. Символ ^ означает здесь клавишу <Ctrl> — т. е., для выхода из редактора нужно нажать комбинацию клавиш <Ctrl>+<X>, а для сохранения текста — <Ctrl>+<O>.

В некоторых системах (например, в FreeBSD) вместо nano используется редактор ee. Он похож на nano, разве что подсказки в нем выводятся до текста (вверху экрана), а не после него, однако идея та же. Весьма удобен и редактор joe.

В пакет mc (файловый менеджер) входит довольно-таки удобный редактор mcedit, который запускается в mc при нажатии клавиши <F4> (рис. 25.6). Flo вы можете запустить редактор и отдельно от mc:

```
mcedit <имя файла>
```

Кстати, редакторы joe, nano и ee запускаются таким же путем:

```
joe <имя файла>
```

```
nano <имя файла>
```

```
ee <имя файла>
```

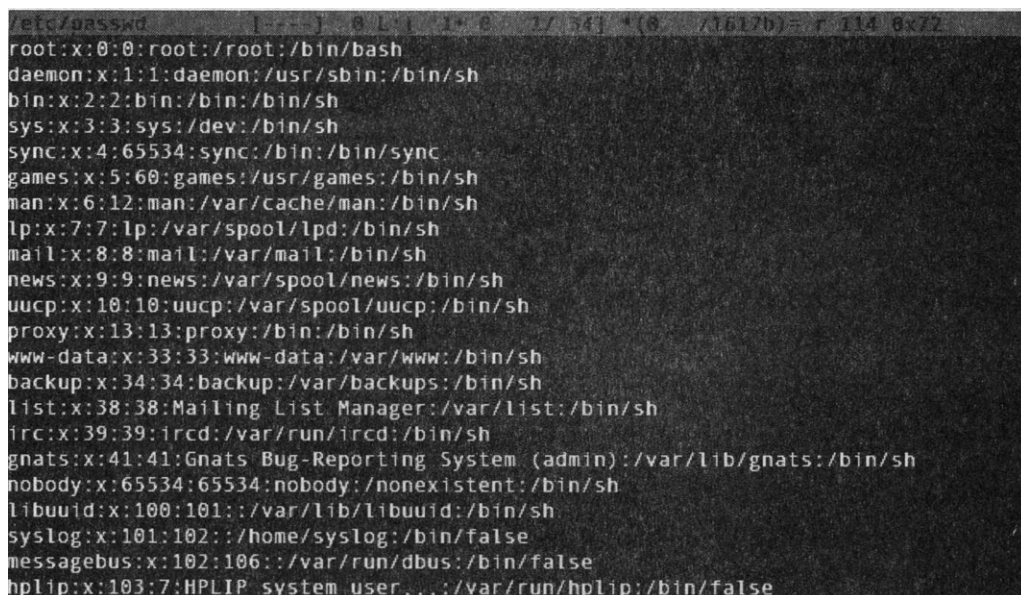


Рис. 25.6. Редактор mcedit

25.3. Команды для работы с Интернетом

25.3.1. Команда *ftp* — стандартный FTP-клиент

Для открытия соединения с любым FTP-сервером введите команду:

```
ftp <имя или адрес FTP-сервера>
```

Подключившись к серверу, вы можете ввести команду *help*, чтобы просмотреть список доступных команд (рис. 25.7). Работа с командой *ftp* подробно описана в разд. 17.5, и нет никакого смысла его здесь дублировать.

```

331 Password required
Password:
230 Logged in, proceed
Remote system type is UNIX.
ftp> help
Commands may be abbreviated.  Commands are:

!                cr                mdir             proxy            send
$                delete            mget             sendport         site
account          debug            mkdir            put              size
append           dir              mls              pwd              status
ascii            disconnect      mode             quit             struct
bell             form            modtime          quote            system
binary           get             mput            recv             sunique
bye              glob            newer            reget           tenex
case             hash            nmap             rstatus         trace
ccc             help            nlist           rhel             type
cd              idle            ntrans          rename           user
cdup            image           open            reset            umask
chmod           lcd             passive          restart          verbose
clear           ls              private          rmdir           ?
close           macdef          prompt           runique
cprotect        mdelete         protect          safe
ftp>

```

Рис. 25.7. Список команд FTP-клиента

Кроме ftp, для Linux разработаны и другие текстовые FTP-клиенты:

- NcFTP (<http://www.ncftp.com>);
- lukemftp (<ftp://ftp.netbsd.org/pub/NetBSD/misc/lukemftp/>);
- lftp (<http://ftp.yars.free.net/projects/lftp/>) и др.

Все они не входят в состав дистрибутивов, и их нужно устанавливать самостоятельно. Но стоит ли это делать — решать вам. Ведь они функционально подобны стандартному клиенту ftp, а если и обладают двумя-тремя дополнительными функциями, то они, возможно, вам и не понадобятся. Например, NcFTP умеет докачивать файлы, а lftp — загружать одновременно несколько файлов. В любом случае вы можете изучить документацию по тому или иному FTP-клиенту (ее легко найти в Интернете), а потом решить, стоит его использовать или нет.

25.3.2. Команда *lynx* — текстовый браузер

Если графический режим недоступен (например, на сервере), а по Сети побродить хочется, командой *lynx* можно вызвать текстовый браузер *lynx*. В некоторых дистрибутивах вместо *lynx* используются браузеры *links* и *elinks*, но суть остается та же — просмотр страниц Интернета в текстовом режиме.

25.3.3. Команда *mail* — чтение почты и отправка сообщений

Команда *mail* — это простейший клиент для чтения и отправки почты. Он позволяет читать только ту почту, что принята вашей системой. Если же нужно принять почту с других POP3-серверов, следует использовать иные почтовые клиенты, которые могут работать в консоли, — например, *mutt* или *pine*.

Для чтения предназначенных вам сообщений введите команду `mail` без параметров. Если хотите написать кому-то письмо, передайте в качестве параметра электронный адрес этого человека:

```
mail ivanov@firma.ru
```

25.4. Команды системного администратора

25.4.1. Команды *free* и *df* — информация о системных ресурсах

Команда `free` выводит информацию об использовании оперативной и виртуальной памяти, а `df` — об использовании дискового пространства. На рис. 25.8 видно, что в системе установлено всего 384 Мбайт ОЗУ, из них 247 Мбайт занято и 138 Мбайт — свободно. На жестком диске `/dev/sda1` всего 2,8 Гбайт дискового пространства, из них свободно — 1,66 Гбайт.

```
[root@localhost den]# free
              total        used        free      shared    buffers     cached
Mem:          385628      247604      138024           0       30584      116056
-/+ buffers/cache:      100964      284664
Swap:         168640           0       168640
[root@localhost den]# df
Файловая система    Разм  Исп  Дост  Исп% смонтирована на
/dev/sda1            2,8G  1,1G  1,6G  42% /
[root@localhost den]#
```

Рис. 25.8. Команды `free` и `df`

25.4.2. Команда *md5sum* — вычисление контрольного кода MD5

Для проверки подлинности некоторых файлов, передаваемых через Интернет, используется алгоритм MD5 (точнее, контрольный код, вычисленный с использованием этого алгоритма). Разработчик программы выкладывает в Интернете пакет с этой программой и на своем сайте публикует контрольный код. Вы скачиваете пакет и вычисляете его контрольный код. Если коды различаются, то файл при передаче был поврежден (или это другая версия пакета, которая, возможно, была подложена злоумышленником с целью ввода вражеского кода в вашу систему).

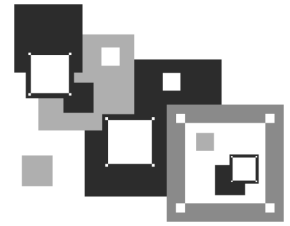
Использовать команду нужно так:

```
md5sum файл
```

25.4.3. Команды *ssh* и *telnet* — удаленный вход в систему

Подробнее эти команды рассмотрены в главе 32, а пока, если есть желание, можете почитать соответствующую страницу руководства (`man`).

ГЛАВА 26



Конфигурационные файлы Linux

26.1. Каталог /etc

Все пользователи Windows наверняка слышали о «святой святых» Windows — реестре. Реестр это огромная бинарная база данных, в которой хранятся все настройки системы: параметры самой системы и параметры всех современных Windows-приложений (старые Windows-программы хранят настройки в INI-файлах).

Каталог `/etc` в Linux чем-то похож на реестр Windows. Он тоже содержит все настройки системы (кроме пользовательских, поскольку пользовательские настройки хранятся в домашнем каталоге пользователя, равно как и в Windows пользовательская часть реестра хранится в домашнем каталоге того или иного пользователя), но при этом в каталоге `/etc` они прописаны не в бинарных, а в текстовых файлах. А поскольку файлы текстовые, то вы можете редактировать их любым текстовым редактором, и вам не потребуется для этого какой-то определенный редактор (вроде `regedit` в Windows), что существенно упрощает работу с системными файлами и повышает надежность системы. Что случится с Windows, если реестр будет поврежден? Думаю, все знают. А вот если даже удалить какой-либо из конфигурационных файлов каталога `/etc`, система продолжит работу как ни в чем не бывало! Конечно, работать она будет не совсем так, как до удаления этого файла, но все же она, в отличие от Windows, не «рухнет» в «синий экран смерти».

В этой главе мы рассмотрим содержимое каталога `/etc` на примере дистрибутива Fedora 26. Понятно, что в других дистрибутивах могут иметься и иные файлы/каталоги конфигурации, а некоторые файлы конфигурации, возможно, будут называться иначе. Но рассмотреть каталог `/etc` всех дистрибутивов здесь просто физически невозможно — получилась бы отдельная книга «Конфигурационные файлы Linux», да и нет в этом особой необходимости, — ведь большинство файлов конфигурации различных дистрибутивов весьма схожи между собой.

Рассмотрены здесь будут также далеко не все каталоги с файлами конфигурации, а только те, на которые нужно обратить ваше внимание. К тому же, конкретный набор конфигурационных файлов и каталогов зависит от установленного программного обеспечения. Например, в моем компьютере установлен Web-сервер Apache, поэтому в нем имеется каталог `/etc/httpd`, содержащий файлы конфигурации Web-

сервера. У вас Web-сервер может быть не установлен, поэтому такого каталога у вас не будет, зато будут свои конфигурационные каталоги и файлы, которых нет у меня.

26.2. Каталог /etc/NetworkManager

В этом каталоге содержатся файлы конфигурации соединений, настроенных с помощью программы NetworkManager, а также основной файл конфигурации программы — NetworkManager.conf.

В подкаталоге system-connections каталога /etc/NetworkManager находятся файлы, описывающие сетевые соединения. Имя файла соответствует имени соединения, введенному в настройках программы. Вряд ли вы будете редактировать эти файлы вручную — для этого обычно используется интерфейс программы NetworkManager. Но для общего развития вы можете просмотреть эти файлы. И хотя они доступны только пользователю root, открыв их, вы будете удивлены, обнаружив, что все пароли (например, от Wi-Fi или DSL-соединений) хранятся здесь в открытом (не-зашифрованном) виде...

На рис. 26.1 приведен файл конфигурации моего DSL-соединения. Как видите, его формат очень прост.

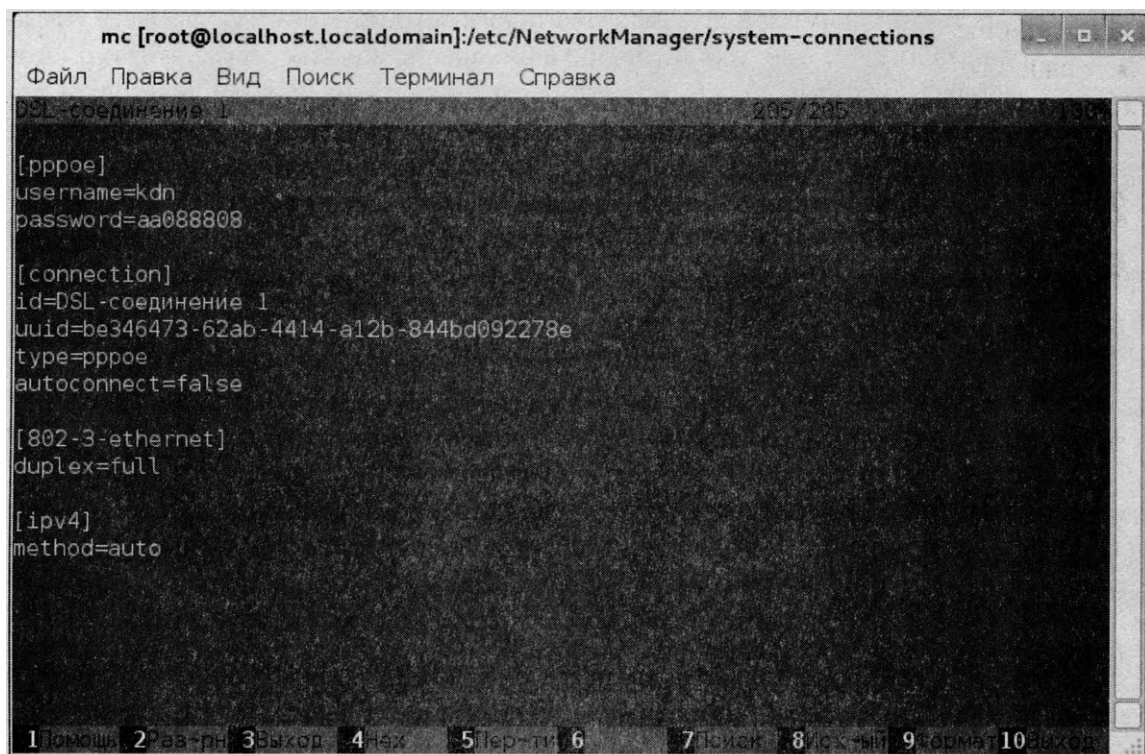


Рис. 26.1. Fedora: файл конфигурации DSL-соединения, созданный с помощью NetworkManager

26.3. Каталог `/etc/abrt`

Этот каталог содержит файлы конфигурации утилиты `abrt` (Automatic bug detection and reporting tool), использующейся для создания и отправки разработчикам отчетов об ошибках. Вряд ли вам придется когда-то редактировать файл конфигурации `abrt` — только по той причине, что вы не будете вызывать `abrt` вручную.

26.4. Каталог `/etc/alsa`

В каталоге `/etc/alsa` находятся параметры системы ALSA (Advanced Linux Sound Architecture) — она обеспечивает поддержку звука в современных дистрибутивах Linux. Файлы конфигурации из каталога `/etc/alsa` вы никогда не будете редактировать вручную — сложно, да и нет в этом необходимости, когда есть графические конфигураторы. Именно в данном случае применение конфигураторов оправданно, поскольку изучение таких конфигурационных файлов нецелесообразно с точки зрения расходования времени.

26.5. Каталоги `/etc/audit` и `/etc/auditd`

Эти каталоги содержат конфигурационные файлы демона аудита — `auditd` и его диспетчера событий — `auditd` (audit event dispatcher). Главным конфигурационным файлом является `/etc/audit/auditd.conf`, из которого можно узнать, что файл журнала этого демона называется `/var/log/audit/audit.log`. Остальные параметры (хотя их там и не много) обычно в редактировании не нуждаются.

26.6. Каталог `/etc/avahi` — файлы конфигурации демона Avahi

Демон Avahi реализует архитектуру Apple ZeroConf, также известную под именами «Rendezvous» и «Bonjour». Демон регистрирует локальные IP-адреса и статические сервисы с помощью mDNS/DNS-SD и предоставляет API (программный интерфейс) локальным программам, позволяя им использовать записи кэша mDNS.

Конфигурационные файлы этого демона находятся в каталоге `/etc/avahi`. Основной конфигурационный файл называется `avahi-daemon.conf`. В файле `/etc/avahi/hosts` прописано соответствие локальных IP-адресов именам компьютеров (по сути, это аналог файла `/etc/hosts`, но для Avahi).

Демон Avahi на практике используется довольно редко, хотя входит в состав многих дистрибутивов. Основная причина его непопулярности в том, что локальная сеть без него и так прекрасно работает! Одним словом, вы вряд ли будете редактировать конфигурационные файлы каталога `/etc/avahi`, поскольку практически сразу после установки системы отключите сам демон.

26.7. Каталог /etc/blkid

В файле `/etc/blkid/blkid.tab` описывается соответствие меток (`LABEL`) разделов идентификаторам `UUID`. Пример этого файла (с моего компьютера) представлен в листинге 26.1.

Листинг 26.1. Пример файла `/etc/blkid/blkid.tab`

```
<device DEVNO="0x080a" TIME="1201886636" LABEL="ETC" UUID="2853-9445"
TYPE="vfat">/dev/sda10</device>
<device DEVNO="0x080c" TIME="1201886636" LABEL="VIDEO" UUID="0861-77A9"
TYPE="vfat">/dev/sda12</device>
<device DEVNO="0x080b" TIME="1201886636" LABEL="FILES" UUID="D05B-B520"
TYPE="vfat">/dev/sdall</device>
<device DEVNO="0x0808" TIME="1201886636" LABEL="SOFT" UUID="B4F4-3620"
TYPE="vfat">/dev/sda8</device>
<device DEVNO="0x0809" TIME="1201886636" LABEL="WORK" UUID="304C-5B60"
TYPE="vfat">/dev/sda9</device>

<device DEVNO="0x0807" TIME="1201886636" TYPE="swap" UUID="6b96d2d4-
d2e3-49f2-abce-c78f8d3df532">/dev/sda7</device>
<device DEVNO="0x0806" TIME="1202800468" LABEL="/" UUID="475f247f-
a419-4ae5-94b1-fada22c232b9" SEC_TYPE="ext2" TYPE="ext3">/dev/sda6</
device>
```

26.8. Файлы конфигурации планировщиков задач

Во многих современных дистрибутивах можно обнаружить файлы `/etc/anacrontab`, `/etc/crontab`, `/etc/at.allow`, `/etc/at.deny`, а также серию каталогов `/etc/cron*`. Все эти файлы и каталоги, кроме `at.allow` и `at.deny`, рассматриваются в *главе 30*.

В файл `at.allow` заносят список команд, которые можно вставить в очередь планировщика `at`. Если файл пуст, то планировщику `at` разрешается запускать любые команды. В файл `at.deny` заносят команды, которые нельзя выполнять с помощью планировщика `at`.

Понятно, что ограничения, накладываемые файлом `at.allow`, более строгие, чем ограничения файла `at.deny`. Так, если вы в `at.allow` занесете одну команду, планировщику будет разрешено выполнять только эту команду, а все остальные команды, вне зависимости от файла `at.deny`, будут запрещены. Поэтому файл `at.allow` по умолчанию даже не существует. Намного проще составить список запрещенных команд — файл `at.deny`. Если вы хотите использовать файл `at.allow`, то его нужно создать самостоятельно:

```
# touch at.allow
```

26.9. Каталог /etc/cups

В каталоге /etc/cups содержатся параметры системы CUPS (Common Unix Printing System). В основном конфигурационном файле этой системы — /etc/cups/printers.conf описаны установленные в системе принтеры. В моей системе этот файл выглядит так, как показано в листинге 26.2.

Листинг 26.2. Файл /etc/cups/printers.conf

```
<Printer Lexmark_E321>
Info Lexmark International Lexmark E321
DeviceURI usb://Lexmark/E321
State Idle
StateTime 1202226025
Accepting Yes
Shared Yes
JobSheets none none
QuotaPeriod 0
PageLimit 0
KLimit 0
OpPolicy default
ErrorPolicy stop-printer
</Printer>
```

Как можно видеть, у меня установлен принтер Lexmark, подключенный к компьютеру по USB. Его текущее состояние — простой (**S**tate **I**dle), т. е. ничего не печатается, к тому же принтер является общим (**S**hared **Y**es).

Файл конфигурации /etc/cups/cupsd.conf определяет настройки сервера печати. Пример этого файла вместе с комментариями представлен в листинге 26.3.

Листинг 26.3. Файл конфигурации /etc/cups/cupsd.conf

```
# Уровень протоколирования: info или debug (см. гл. 27)
LogLevel info

# Группы пользователей, к которым принадлежит администратор
SystemGroup sys root

# Прослушивать соединения на компьютере localhost, порт 631
Listen localhost:631
# Файл сокета
Listen /var/run/cups/cups.sock

# Показывать общие принтеры другим компьютерам сети
Browsing On
BrowseOrder allow,deny
BrowseAllow all
```

```

# Метод аутентификации, если она нужна
DefaultAuthType Basic

# Ограничиваем непосредственный доступ к серверу
# Разрешить доступ только локальному компьютеру
<Location />
    Order allow,deny
    Allow localhost
</Location>

# Ограничить доступ к панели управления
<Location /admin>
    Encryption Required
    Order allow,deny
    Allow localhost
</Location>

# Ограничить доступ к конфигурационным файлам
<Location /admin/conf>
    AuthType Default
    Require user @SYSTEM
    Order allow,deny
    Allow localhost
</Location>

# Политики по умолчанию
<Policy default>
    # Операции над заданиями печати доступны только администратору
    # и владельцу задания
    <Limit Send-Document Send-URI Hold-Job Release-Job Restart-Job Purge-Jobs
Set-Job-Attributes Create-Job-Subscription Renew-Subscription
Cancel-Subscription Get-Notifications Reprocess-Job Cancel-Current-Job
Suspend-Current-Job Resume-Job CUPS-Move-Job>
        Require user @OWNER @SYSTEM
        Order deny,allow
    </Limit>

# Все административные задачи (например, добавление принтера) 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 могут отменить
# или аутентифицировать задание печати
<Limit Cancel-Job CUPS-Authenticate-Job>
    Require user @OWNER @SYSTEM
    Order deny,allow
</Limit>

CLimit All>
    Order deny,allow
</Limit>
</Policy>
```

26.10. Файл /etc/fonts/fonts.conf

Этот файл содержит настройки подсистемы шрифтов: описывает каталоги со шрифтами, каталоги с кэшем шрифтов, описывает аналоги шрифтов (если требуемый шрифт недоступен, то вместо него будет использоваться аналог). Формат этого файла несложен, к тому же он тщательно прокомментирован. Очень сомневаюсь, что вам придется когда-либо его редактировать.

26.11. Каталог /etc/gdm

Каталог `/etc/gdm` содержит файлы конфигурации и инициализационные файлы менеджера дисплея GNOME (GDM, GNOME Display Manager). Вы не будете редактировать файлы из этого каталога — вам достаточно только знать, что в нем находится. Изменение сценариев `gdm` возможно лишь при условии, что вы знаете, что делаете, — т. е. в тех случаях, когда вы хотите (и можете) изменить ход инициализации GDM.

26.12. Файлы конфигурации популярных сетевых служб

Файлы конфигурации популярных сетевых служб, таких как Web-сервер Apache, заслуживают отдельного разговора, а поэтому рассматриваются в других главах книги (табл. 26.1).

Таблица 26.1. Главы книги, в которых рассматриваются файлы конфигурации сетевых служб

Файл конфигурации	Служба	Глава
/etc/ssh/sshd_config	SSH-сервер	32
/etc/httpd/conf/httpd.conf	Web-сервер Apache	33

Таблица 26.1 (окончание)

Файл конфигурации	Служба	Глава
/etc/proftpd/proftpd.conf	FTP-сервер ProFTPD	34
/etc/bind/named.conf	DNS-сервер	35
/etc/squid/squid.conf	Прокси-сервер SQUID	36
/etc/exports	NFS (Network File System)*	*
/etc/samba/smb.conf	Samba: доступ к Windows-сети	38

*) Сетевая файловая система.

Описание сетевой файловой системы (Network File System) вы найдете в папке *Дополнения* сопровождающего книгу электронного архива (см. *приложение*).

26.13. Каталог /etc/logrotate.d

Файлы протоколов рано или поздно станут неприлично большими. Что-либо найти в таком файле будет сложно, да и система станет работать чуть медленнее из-за увеличения размера журналов. Для решения этой проблемы в Linux используется утилита logrotate. Основная задача logrotate — ротация журналов. Например, у нас есть журнал /var/log/messages. Когда он станет огромным, logrotate переименует его в messages.1, а вместо него создаст пустой файл messages. Когда файл messages опять заполнится, программа переименует файл messages.1 в messages.2, а messages в messages.1, и т. д.

В каталоге /etc/logrotate.d описаны действия, необходимые для ротации тех или иных журналов. Вам не нужно редактировать эти файлы! Вы можете отредактировать основной конфигурационный файл logrotate — /etc/logrotate.conf (листинг 26.4).

Листинг 26.4. Файл /etc/logrotate.conf

```
# Как часто нужно выполнять ротацию журналов:
# weekly — каждую неделю,
# daily — каждый день,
# monthly — ежемесячно
weekly

# Сколько предыдущих журналов хранить? Для домашнего компьютера это число
# можно уменьшить до 2, а для сервера — увеличить до 8-10
rotate 4

# После ротации создать пустой файл журнала
create

# Использовать дату в качестве суффикса для журнала после ротации
dateext
```

```
# Сжимать файлы журналов (обычно используется gzip)
#compress
# Каталог, содержащий указания по ротации.
# Редактировать файлы из этого каталога не нужно!
include /etc/logrotate.d

# В каталоге /etc/logrotate.d нет указаний по ротации журналов *tmp,
# поэтому они описываются прямо в файле конфигурации
/var/log/wtmp {
    monthly
    create 0664 root utmp
    rotate 1
}

/var/log/btmp {
    missingok
    monthly
    create 0600 root utmp
    rotate 1
}
```

26.14. Каталог /etc/mail

Здесь находятся файлы конфигурации почтового агента sendmail. Кто-то считает sendmail устаревшим, а кто-то до сих пор с успехом его использует.

В любом случае в этой книге мы не рассматриваем sendmail, поэтому не станем описывать и его файлы конфигурации.

26.15. Каталог /etc/ntp

Этот каталог содержит файлы конфигурации сервера времени. Сервер времени подробно рассматривается в моей книге «Серверное применение Linux»¹. А вообще, файлы из этого каталога редактируются довольно редко. Все, что требуется, — это в файле /etc/ntp/ntpservers прописать серверы времени, откуда нужно получать точное время.

26.16. Каталог /etc/openldap

Каталог /etc/openldap содержит файлы конфигурации сервера каталогов OpenLDAP (Lightweight Directory Access Protocol). Рассмотрение этого сервера выходит за рамки нашей книги — если вам интересно, рекомендую прочитать статью:

<http://www.nixp.ru/articles/openldap>.

¹ Колисниченко Д. Н. Серверное применение Linux. — 3-е изд. — СПб.: БХВ-Петербург, 2011, <http://bhv.ru/books/book.php?id=188723>.

26.17. Каталог/etc/openvpn

Этот каталог содержит файлы конфигурации виртуальной частной сети OpenVPN (Open Virtual Private Network). О настройке виртуальной частной сети вы можете прочитать в моей книге «Серверное применение Linux», ссылка на которую приведена в *разд. 26.15*.

26.18. Каталоги /etc/pam.d и /etc/security

Каталоги /etc/pam.d и /etc/security содержат файлы конфигурации модулей аутентификации PAM. Все конфигурационные файлы из этих каталогов рассмотрены в *главе 29*.

26.19. Каталог/etc/ppp

Этот каталог содержит файлы конфигурации демона pppd, отвечающего за установку PPP-соединений (в том числе и PPPoE-соединений):

- **chap-secrets** — пароли PPP-соединений при условии, что используется CHAP-аутентификация (пароли хранятся в открытом виде, а передаются по сети в зашифрованном);
- **pap-secrets** — пароли PPP-соединений, при условии, что используется PAP-аутентификация (пароли хранятся и передаются по сети в открытом виде);
- **firewall*** — набор правил брандмауэра для PPP-соединений;
- **ip-up** — действия при установке соединения;
- **ip-down** — действия при завершении соединения;
- **options** — параметры PPP-соединения;
- **pppoe-server-options** — параметры сервера PPPoE;
- **resolv.conf** — содержит IP-адреса DNS-серверов при работе по PPP-соединению.

26.20. Каталог /etc/rc.d

Каталог /etc/rc.d содержит сценарии инициализации системы. Подробно этот каталог рассматривается в *главе 22*.

26.21. Каталог/etc/sane.d

Этот каталог содержит конфигурационные файлы xsane — программы для работы со сканером.

КОНФИГУРАЦИЯ СКАНЕРА

Подробно конфигурация сканера рассмотрена в материале, с которым вы можете ознакомиться по адресу: http://www.dkws.org.ua/novice/pdf/printer_scanner.pdf.

26.22. Каталог /etc/selinux

Это конфигурационный каталог системы управления доступом SELinux. В этом издании система SELinux не рассматривается.

СИСТЕМА УПРАВЛЕНИЯ ДОСТУПОМ SELINUX

Описание системы управления доступом SELinux (*глава 33* из 2-го издания книги) вы найдете в папке *Дополнения* сопровождающего книгу электронного архива (см. *приложение*).

26.23. Каталог /etc/skel

Это довольно-таки интересный каталог. Все мы знаем, что при создании нового пользователя в каталоге /home создается его домашний каталог. Но если сразу после создания пользователя просмотреть его домашний каталог, то вы обнаружите, что он не пуст, — в нем уже есть файлы, хотя пользователь еще не заходил в систему, и незапущенные им программы не могли создать свои файлы конфигурации в его домашнем каталоге. Оказывается, при создании нового пользователя в его домашний каталог копируется содержимое каталога skel.

26.24. Каталог /etc/sysconfig

Каталог /etc/sysconfig содержит конфигурационные файлы системы. В нем очень много конфигурационных файлов, и все их мы рассматривать не станем. Каждый файл тщательно закомментирован, поэтому вам не составит труда разобраться с ними самостоятельно. Вот некоторые файлы и подкаталоги из этого каталога:

- **network-scripts** — каталог содержит скрипты настройки сетевых интерфейсов, описывающие параметры сетевых интерфейсов (в том числе конфигурацию протокола IP: IP-адрес, адрес шлюза, сетевую маску);
- **networking/devices** — каталог содержит файлы конфигурации сетевых интерфейсов;
- **autofs** — файл содержит конфигурацию демона autofs, отвечающего за автоматическое монтирование сменных носителей;
- **clock** — файл содержит конфигурацию системных часов (часовой пояс и другие параметры);
- **crond** — используется для передачи аргументов планировщику crond;
- **firstboot** — при первом запуске директива `RUN FIRSTBOOT` из этого файла принимает значение `YES`, после чего устанавливается значение `NO`. Если вы хотите заново запустить мастер начальной настройки, установите `RUN FIRSTBOOT` в `YES`;
- **grub** — некоторые параметры загрузчика GRUB;
- **hwconf** — содержит аппаратную конфигурацию компьютера, вручную не редактируется;
- **iptables-config** — содержит параметры брандмауэра iptables;

- **iptables** — в этом файле хранятся правила брандмауэра iptables;
- **irda** — параметры инфракрасного приемопередатчика;
- **keyboard** — параметры клавиатуры, в частности, ее раскладка;
- **network** — некоторые сетевые параметры, например доменное имя компьютера;
- **nfs** — параметры сетевой файловой системы NFS;
- **rsyslog** — используется для передачи параметров демону протоколирования rsyslogd;
- **samba** — некоторые параметры Samba;
- **system-config-*** — параметры конфигураторов системы.

26.25. Каталог/etc/X11

В этом каталоге содержатся настройки системы X.Org. Основной конфигурационный файл этой системы `xorg.conf` был подробно описан в *главе 13*, поэтому не вижу смысла рассматривать его еще раз.

26.26. Конфигурационные файлы yum/dnf

Файл `yum.conf` содержит параметры менеджера пакетов yum, а в каталоге `yum.repos.d` описаны репозитории yum. Формат файла `yum.conf` и файлов репозитория рассматривается в *главе 7* — если вы ее пропустили, то самое время прочитать. Поскольку, начиная с версии 22, вместо yum используется менеджер пакетов dnf, этот каталог из состава Fedora должен быть исключен, а настройки менеджера пакетов будут храниться только в каталоге `/etc/dnf`. В настоящее время (версия 26) конфигурация репозитория хранится в файле `/etc/yum.repos.d`, хотя уже несколько лет как используется менеджер dnf.

26.27. Основные конфигурационные файлы сети

Назначение файлов `aliases`, `hosts.conf`, `hosts`, `hosts.allow`, `hosts.deny`, `iftab`, `motd`, `resolv.conf`, `services` и `xinetd.conf` рассматривается в *главе 8*.

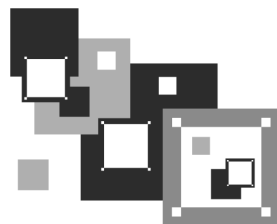
26.28. Остальные конфигурационные файлы каталога /etc

В табл. 26.2 представлены подробно не рассмотренные пока файлы конфигурации из каталога `/etc`. Еще раз замечу, что файлы тщательно прокомментированы, поэтому вам даже не придется читать по ним какую-либо дополнительную документацию.

Таблица 26.2. Некоторые конфигурационные файлы

Файл	Описание
bashrc	Содержит описание некоторых функций командного интерпретатора <code>bash</code> . Обычно этот файл никогда не редактируется
filesystems	В файле находится список поддерживаемых ядром файловых систем. Сугубо информационный файл, вам не нужно его редактировать
fstab	Описывает файловые системы, монтируемые при запуске операционной системы. Подробно описан в <i>главе 4</i>
group	Содержит информацию о группах пользователей (см. <i>главу 6</i>)
issue	Текст сообщения, выводимого перед <i>локальной</i> регистрацией пользователя в системе
issue.net	Текст сообщения, выводимого перед <i>удаленной</i> регистрацией пользователя в системе
man_db.conf	Файл конфигурации справочной системы <code>man</code>
modprobe.conf	Содержит список автоматически загружаемых модулей. В некоторых дистрибутивах этот файл еще существует, поэтому он представлен в этой таблице именно как файл. В Fedora 16 этот файл упразднен, а все настройки, которые были в нем, перенесены в каталог <code>/etc/modprobe.d</code> . Также загляните в каталог <code>modules-load.d</code> — в нем описываются дополнительные модули
mtab (или /proc/self/mounts)	В этом файле вы найдете список смонтированных в текущий момент файловых систем
networks	Описывает сети и подсети
passwd	Хранит информацию о пользователях (см. <i>главу 6</i>)
protocols	Содержит список поддерживаемых протоколов
shells	Содержит список командных интерпретаторов, установленных в системе
sudoers	Определяет, кому можно использовать команду <code>sudo</code> (см. <i>главу 6</i>)
sysctl.conf	Системная конфигурация ядра

ГЛАВА 27



Протоколирование системы

В любой UNIX-системе, коей является и Linux, имеются так называемые *демоны протоколирования* (далее просто «демоны»). Демоны записывают в протоколы (журналы, логи) сообщения, генерируемые ядром, сервисами, пользовательскими программами.

Если вы уже работали с Linux, то знаете, что ранее протоколирование системы осуществляли демоны `syslogd` и `rsyslogd`. Сейчас же все устроено немного иначе — во многих современных дистрибутивах протоколированием системы занимается сама система инициализации `systemd`, а точнее — ее сервис `systemd-journald.service`. При этом запрос системного лога (журнала) организуется через утилиту `journaltl`.

Протоколирование через сервис `systemd-journald.service` впервые появилось в Fedora 20, перешли на `systemd` и все современные дистрибутивы, среди которых Fedora, CentOS, Debian и Ubuntu (начиная с 15.04). Однако в дистрибутивах, основанных на системе инициализации, отличной от `systemd`, может использоваться демон `syslogd`, — именно поэтому его описание не удалено из книги (см. *разд. 27.2*),

Вот, что нужно знать о протоколировании в современных дистрибутивах:

- все журналы по-прежнему хранятся в каталоге `/var/log`;
- серверы (WWW, FTP и пр.) могут создавать собственные каталоги/файлы журналов в каталоге `/var/log`;
- для просмотра системных журналов используется утилита `journaltl` — привычные файлы вроде `/var/log/messages` более недоступны. Конечно, вы можете параллельно установить демон `rsyslogd`, и он будет прекрасно работать в паре с `journaltl`. Однако у `journaltl` гораздо больше возможностей — например, с помощью опции `-b` можно просмотреть логи текущей или предыдущей загрузки. Некоторые возможности утилиты `journaltl` рассмотрены в этой книге, а с остальными вы сможете ознакомиться в справочной системе;

УТИЛИТА JOURNALCTL

Полное описание утилиты `journaltl` можно найти в руководстве `man` или по адресу <http://www.freedesktop.org/software/systemd/man/journalctl.html>, а мы же рассмотрим здесь практические примеры ее использования.

- существует графическая утилита просмотра журналов — `gnome-system-log` (рис. 27.1). Она не устанавливается по умолчанию, и чтобы ее установить, введите команду:

```
sudo dnf install gnome-system-log
```

Впрочем, учитывая, что просмотр основных журналов осуществляется через утилиту `journalctl`, особого эффекта от использования `gnome-system-log` вы не ощутите.

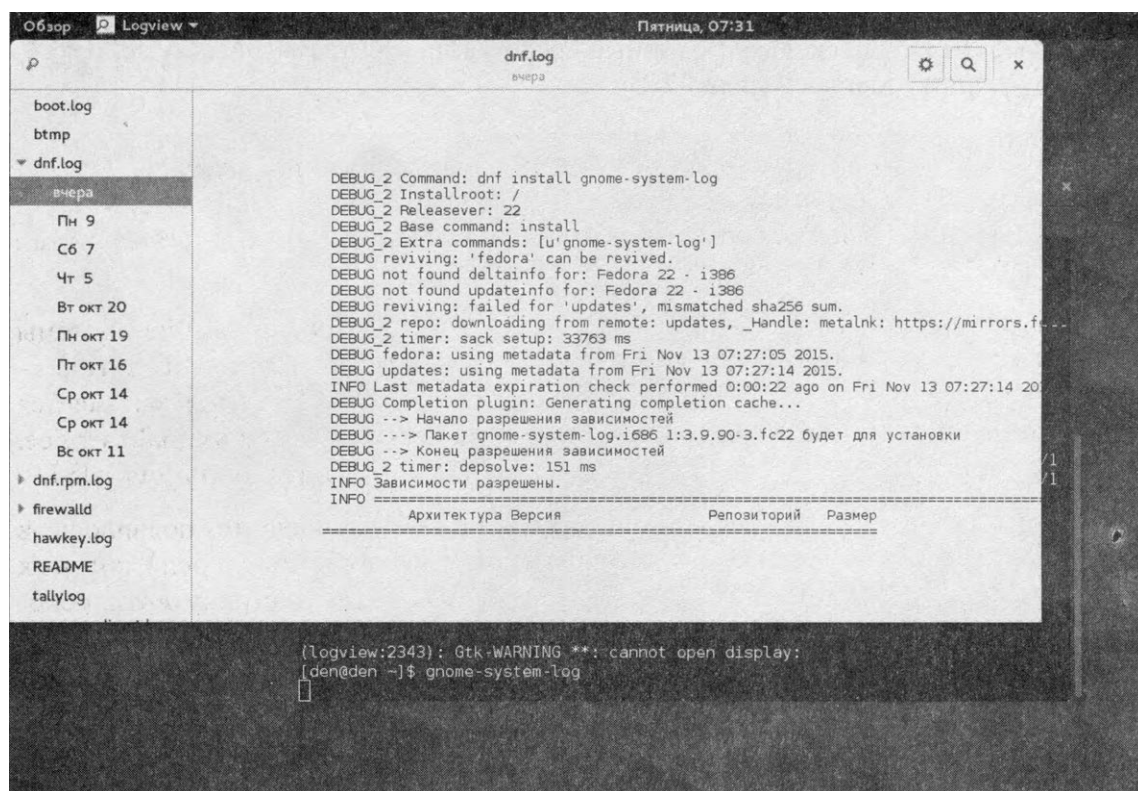


Рис. 27.1. Fedora: утилита `gnome-system-log`

27.1. Протоколирование по-новому: `journalctl`

27.1.1. Установка времени

Первое, что нужно сделать, — это установить правильный часовой пояс. Основным недостаток `syslogd/rsyslogd` заключался в том, что эти демоны сохраняли записи в журнале без учета часового пояса, и было непонятно, когда именно произошло то или иное событие. В `journalctl` этот недостаток устранен — можно использовать как местное время, так и UTC.

Для выбора часового пояса служит команда:

```
$ timedatectl set-timezone <часовой пояс>
```


Просмотреть список часовых поясов можно командой:

```
$ timedatectl list-timezones
```

Просмотреть информацию о текущем часовом поясе можно командой:

```
$ timedatectl status
```

27.1.2. Просмотр и фильтрация логов

Для просмотра логов введите команду `journalctl` — будет выведен огромный список различных записей. Использовать команды постраничного вывода вроде `more` необходимости нет, поскольку подобные средства просмотра журнала уже встроены в саму утилиту `journalctl` (рис. 27.2).

```
den@den:~
Файл Правка Вид Поиск Терминал Справка
-- Logs begin at Чт 2015-10-08 21:20:24 EEST, end at Пт 2015-11-13 07:42:11 EET.
окт 08 21:20:24 localhost.localdomain systemd-journal[84]: Runtime journal is us
окт 08 21:20:24 localhost.localdomain systemd-journal[84]: Runtime journal is us
окт 08 21:20:24 localhost.localdomain kernel: Initializing cgroup subsys cpuset
окт 08 21:20:24 localhost.localdomain kernel: Initializing cgroup subsys cpu
окт 08 21:20:24 localhost.localdomain kernel: Initializing cgroup subsys cpuacct
окт 08 21:20:24 localhost.localdomain kernel: Linux version 4.0.4-301.fc22.i686
окт 08 21:20:24 localhost.localdomain kernel: Disabled fast string operations
окт 08 21:20:24 localhost.localdomain kernel: e820: BIOS-provided physical RAM m
окт 08 21:20:24 localhost.localdomain kernel: BIOS-e820: [mem 0x0000000000000000
окт 08 21:20:24 localhost.localdomain kernel: BIOS-e820: [mem 0x00000000000009f800
окт 08 21:20:24 localhost.localdomain kernel: BIOS-e820: [mem 0x0000000000000ca000
окт 08 21:20:24 localhost.localdomain kernel: BIOS-e820: [mem 0x000000000000dc000
окт 08 21:20:24 localhost.localdomain kernel: BIOS-e820: [mem 0x0000000000100000
окт 08 21:20:24 localhost.localdomain kernel: BIOS-e820: [mem 0x0000000002fef0000
окт 08 21:20:24 localhost.localdomain kernel: BIOS-e820: [mem 0x0000000002feff000
окт 08 21:20:24 localhost.localdomain kernel: BIOS-e820: [mem 0x0000000002ff00000
окт 08 21:20:24 localhost.localdomain kernel: BIOS-e820: [mem 0x000000000e0000000
окт 08 21:20:24 localhost.localdomain kernel: BIOS-e820: [mem 0x000000000fec00000
окт 08 21:20:24 localhost.localdomain kernel: BIOS-e820: [mem 0x000000000fee00000
окт 08 21:20:24 localhost.localdomain kernel: BIOS-e820: [mem 0x000000000fffe0000
окт 08 21:20:24 localhost.localdomain kernel: Notice: NX (Execute Disable) prote
окт 08 21:20:24 localhost.localdomain kernel: SMBIOS 2.4 present.
lines 1-23
```

Рис. 27.2. Просмотр журнала утилитой `journalctl`

Обратите внимание на самую первую строку — она говорит, с какого момента начинается ведение логов. Как правило, это дата установки системы. Понятно, что с самого начала будет очень много записей, и их как-то нужно фильтровать.

Текущая и предыдущие загрузки

Чтобы просмотреть системные логи с момента текущей загрузки, используйте параметр `-b`:

```
$ journalctl -b
```

А вот просмотреть логи за предыдущую загрузку немного сложнее — увидеть список предыдущих загрузок можно командой (рис. 27.3):

```
$ journalctl --list-boots
```

```

den@den:~
Файл Правка Вид Поиск Терминал Справка
[den@den ~]$ journalctl --list-boots
-16 8080ac6426ad46a995522931251fe99d Чт 2015-10-08 21:20:24 EEST-Чт 2015-10-08 2
-15 80d2c07ff0894ea897fec33b6e674a40 Чт 2015-10-08 22:51:34 EEST-Чт 2015-10-08 2
-14 360e6435822c4c9f8b5c23f225fe4f41 Пт 2015-10-09 08:31:28 EEST-Пт 2015-10-09 0
-13 8aff193584d349589cc52ba7cfd89b65 Вс 2015-10-11 13:57:47 EEST-Вс 2015-10-11 1
-12 01c95be23cbb487993bb1c17fb24e018 Пн 2015-10-12 18:57:35 EEST-Пн 2015-10-12 2
-11 5f3127fc18ca42fa9d05c4ca1129b7f3 Чт 2015-10-15 07:31:46 EEST-Чт 2015-10-15 0
-10 b21bbb09d7634691b23cf5cca58e56a1 Сб 2015-10-17 07:45:08 EEST-Сб 2015-10-17 0
-9 35928c19c3134cf18004a4e3dda635ff Вт 2015-10-20 18:55:00 EEST-Вт 2015-10-20 1
-8 df3bc2fa60444cc2ac03e45c96188d79 Вт 2015-10-20 19:58:50 EEST-Вт 2015-10-20 2
-7 e5a84b90a4a54d0dbbf42679bb6a2b9d Ср 2015-10-21 19:38:43 EEST-Ср 2015-10-21 2
-6 800c8cbcd5dd4f2da1bf8bc717147c6f Вс 2015-10-25 08:36:30 EET-Вс 2015-10-25 08
-5 9a0ab16b819f4c209a433a5f7275fe24 Пт 2015-11-06 19:09:46 EET-Пт 2015-11-06 19
-4 3daba2224eff42ed9e9a16dd5b979a11 Вс 2015-11-08 13:33:47 EET-Вс 2015-11-08 14
-3 2102b901b4ed4cc482043f4cf6f175c4 Вс 2015-11-08 14:26:27 EET-Вс 2015-11-08 14
-2 c3c6a026e9a1440ab62f548f22834c84 Вс 2015-11-08 14:35:24 EET-Вс 2015-11-08 16
-1 c92ced611adf4aeaa069d89f7881c888 Вт 2015-11-10 17:04:25 EET-Вт 2015-11-10 17
0 35fe55473714467c98f9ad29288f0993 Пт 2015-11-13 07:21:08 EET-Пт 2015-11-13 08
lines 1-17/17 (END)

```

Рис. 27.3. Предыдущие загрузки: для каждой загрузки выводится ее идентификатор и дата

А чтобы просмотреть журнал предыдущей загрузки, следует указать ее номер — например: 0 — текущая загрузка, -1 — предыдущая и т. д.):

```
$ journalctl -b -1
```

Фильтр по дате

Фильтрацию журналов можно выполнить и по дате — для этого используются опции `--since` и `--until`. Например, для просмотра логов, начиная с 11.11.15 7:00, введите команду:

```
$ journalctl --since "2015-11-11 07:00:00"
```

Если вы указали опцию `--since`, но не указали дату, будет взята текущая дата. Если указана дата, но не указано время, будет взято время 00:00:00.

Еще несколько примеров:

```

$ journalctl --since yesterday
$ journalctl --since 09:00 --until now
$ journalctl --since 10:00 --until "1 hour ago"

```

Первая команда показывает логи, начиная со вчерашнего дня и по текущий момент. Вторая — отображает журналы, начиная с 9 утра и по текущий момент. Третья — начиная с 10 утра и до прошлого часа (т. е. если сейчас 19:00, то до 18:00).

Фильтр по сервису

Выполнить фильтрацию можно и по определенному сервису (когда нужно просмотреть не все журналы, а только определенной службы) — например:

```
$ journalctl -u nginx.service
```

Тип фильтрации можно комбинировать — например, следующая команда выводит журналы nginx, начиная со вчерашнего дня:

```
$ journalctl -u nginx.service -since today
```

Фильтр по пути

Просмотреть журналы какого-то процесса можно путем указания пути к нему:

```
$ journalctl /usr/bin/docker
```

Фильтр по процессу или пользователю

Можно отфильтровать журнал по PID процесса:

```
$ journalctl _PID=<ИД процесса>
```

А также — и по UID пользователя:

```
$ journalctl _UID=33
```

Узнать UID пользователя можно так:

```
$id -u <имя пользователя>
```

Просмотр сообщений ядра

Для просмотра сообщений ядра используйте опции `-к` или `--dmesg`:

```
$ journalctl -к
```

Эта команда покажет все сообщения ядра для текущей загрузки. Такую команду можно комбинировать с опцией `-b`, чтобы просмотреть сообщения ядра во время предыдущей загрузки:

```
$ journalctl -к -b -2
```

Фильтр по уровню ошибки

В `journalctl`, как и в `syslogd`, используется та же классификация уровней ошибок:

- ☐ 0 — EMERG (система неработоспособна);
- ☐ 1 — ALERT (требуется немедленное вмешательство);
- ☐ 2 — CRIT (критическое состояние);
- ☐ 3 — ERR (ошибка);
- ☐ 4 — WARNING (предупреждение);
- ☐ 5 — NOTICE (просто обратите внимание);

- 6 — INFO (информационное сообщение);
- 7 — DEBUG (отложенная печать).

Например:

```
$ journalctl -p err -b
```

Эта команда выводит все ошибки при текущей загрузке.

27.1.3. Журналы в реальном времени

Журналы системы можно просматривать в реальном времени. Для этого используется опция `-f`:

```
$ journalctl -f
```

27.1.4. Централизованное хранение логов

Некоторые администраторы предпочитают демон `rsyslogd` только потому, что он позволяет хранить логи на другом (центральном) компьютере. Однако система инициализации `systemd` также позволяет «собирать» логи со всех серверов на каком-то одном сервере сети. Для решения этой задачи используются следующие ее компоненты: `systemd-journal-remote`, `systemd-journal-upload` и `systemd-journal-gatewayd`:

- команда `systemd-journal-remote` позволяет принимать логи с удаленных узлов и сохранять их, например:

```
$ systemd-journal-remote --url https://host:19531/
```

На каждом из этих узлов должен быть запущен демон `systemd-journal-gatewayd`. В результате приведенной команды журналы с узла `host` будут сохранены в каталоге `/var/log/journal/host/remote-host.journal`;

- команда `systemd-journal-remote` также позволяет «собирать» имеющиеся на локальной машине журналы в отдельный каталог, например:

```
$ journalctl -o export | systemd-journal-remote -o /tmp/dir -
```

- команда `systemd-journal-upload` служит для загрузки журналов с локальной машины в удаленное хранилище:

```
$ systemd-journal-upload --url https://server:19531/
```

27.2. Демоны `syslogd` и `rsyslogd`

Как уже было сказано ранее, во многих современных дистрибутивах протоколированием системы занимается сама система инициализации `systemd`, однако в дистрибутивах, основанных на системе инициализации, отличной от `systemd`, в качестве основного демона протоколирования продолжает использоваться `syslogd`. Вот работу с ним в таких условиях мы здесь и рассмотрим.

Демон `syslogd` имеется практически на всех UNIX-системах — от самых старых до самых новых, правда, в современных дистрибутивах применяются модифицированные версии `syslogd`: `rsyslogd` или `syslog-ng`. Первый из них получил большее распространение, ему мы и уделим здесь основное внимание. Секрет популярности `rsyslogd` — в файле конфигурации, синтаксис которого идентичен синтаксису файла настроек демона `syslogd`. Это очень удобно. Во-первых, не приходится изучать новый синтаксис, во-вторых, подобие формата файла упрощает миграцию на `rsyslogd`, — достаточно просто переименовать файл конфигурации и запустить новый демон протоколирования.

Иногда пользователи отключают сервис `syslogd` (или `rsyslogd`). Настоятельно рекомендую не делать этого. Ведь у Linux довольно развита функция самодиагностики, и в случае возникновения сбоя по содержимому журналов вы сможете понять, в чем причина сбоя, и устранить ее. Во всяком случае с записями в журнале это будет проще сделать, чем без них.

Основным файлом конфигурации демона `syslogd` является файл `/etc/syslog.conf`, а основным файлом конфигурации демона `rsyslogd` — файл `/etc/rsyslog.conf`. Формат обоих файлов следующий:

селектор [; селектор] действие

В некоторых системах — например, в Ubuntu (до версии 15.04), файл `/etc/rsyslog.conf` — общий, а конкретные настройки, относящиеся к протоколированию, прописаны в отдельных файлах, лежащих в каталоге `/etc/rsyslog.d`. Формат всех этих файлов аналогичен формату файла `rsyslog.conf`. Кстати, в openSUSE вам понадобятся права `root` даже для того, чтобы лишь прочитать файл `/etc/rsyslog.conf`.

Параметр селектор определяет, какие сообщения должны быть запротоколированы. Вот список наиболее часто использующихся селекторов:

- `auth, security` — все, что связано с регистрацией пользователя в системе;
- `authpriv` — отслеживает программы, изменяющие привилегии пользователей, — например, программу `su`;
- `cron` — сообщения планировщиков заданий;
- `kern` — сообщения ядра;
- `mail` — сообщения почтовых программ;
- `news` — сообщения новостного демона;
- `uucp` — сообщения службы Unix-to-Unix-CoPy. Она уже давно не используется, но файл конфигурации демона все еще содержит упоминание о ней;
- `syslog` — сообщения самого демона `syslogd`;
- `user` — сообщения пользовательских программ;
- `daemon` — сообщения различных сервисов;
- `*` — все сообщения.

При указании селектора можно определить, какие сообщения нужно протоколировать:

- `debug` — отладочные сообщения;
- `info` — информационные сообщения;
- `err` — ошибки;
- `warning` — предупреждения (некритические ошибки);
- `crit` — критические ошибки;
- `alert` — «тревожные» сообщения, требующие вмешательства администратора;
- `emerg` — очень важные сообщения (произошло что-то такое, что мешает нормальной работе системы);
- `notice` — замечания.

Впрочем, обычно селекторы указываются так:

`название_селектора.*`

Это означает, что будут протоколироваться все сообщения селектора. Вот еще несколько примеров:

- `daemon.*` — протоколируются все сообщения сервисов;
- `daemon.err` — регистрировать только сообщения об ошибках сервисов.

Теперь перейдем к параметру `действие` — это второе поле файла конфигурации. В большинстве случаев `действие` — это имя файла журнала, в который нужно записать сообщение селектора. Если перед именем файла стоит дефис (-), то после каждой записи в журнал демон не станет выполнять синхронизацию файла, т. е. осуществлять системный вызов `fsync()`. Это повышает производительность системы, поскольку сообщений обычно много, и если после каждого выполнять синхронизацию журнала, система будет работать медленно.

Фрагмент конфигурационного файла `rsyslog.conf` (`syslog.conf`) приведен в листинге 27.1.

Листинг 27.1. Фрагмент файла конфигурации `/etc/rsyslog.conf` (дистрибутив Fedora)

```
# Сообщения ядра протоколируются на консоль
#kern.* /dev/console

# Протоколировать все сообщения (кроме почты) в /var/log/messages
*.info;mail.none;authpriv.none;cron.none /var/log/messages

# Сообщения селектора authpriv записываются в файл /var/log/secure
authpriv.* /var/log/secure

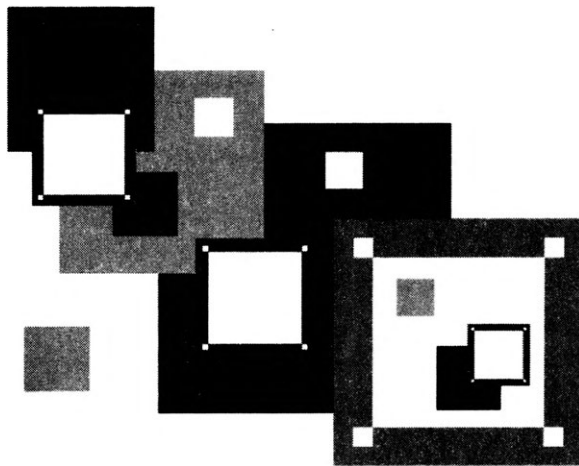
# Сообщения почты (их будет много, если запущен агент МТА вроде postfix)
# записываются в файл maillog
mail.* -/var/log/maillog
```

```
# Сообщения планировщиков заданий записываются в cron
cron.*                                /var/log/cron
```

```
# Особо критичные сообщения выводятся на экран всех работающих в данный момент #
пользователей (вместо имени файла указана звездочка)
*.emerg *
```

```
# Сообщения UUCP и сообщения сервера новостей записываются в /var/log/spooler
uucp,news.crit                       /var/log/spooler
```

```
# Загрузочные сообщения записываются в boot.log
local17.*                            /var/log/boot.log
```

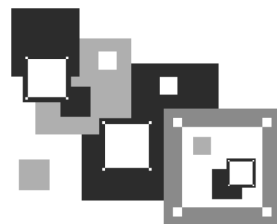


ЧАСТЬ VI

Linux на сервере

Операционная система Linux с успехом работает на многих серверах, поэтому книга не была бы полной без рассмотрения серверных возможностей Linux. В этой части мы уделим внимание самым популярным сетевым сервисам Linux.

ГЛАВА 28



Обеспечение безопасности сервера

Гибкость Linux — это источник ее проблем. Она настолько гибка, что с одинаковой легкостью предоставляет свои возможности как законному администратору, так и злоумышленнику. Любой, кто получит физический доступ к серверу (т. е. к его клавиатуре и монитору), может захватить root-доступ всего за несколько секунд, если будет знать, что делать. В этой главе мы поговорим о том, как защитить наш сервер от подобных вмешательств.

28.1. Защита от «восстановления пароля root»

28.1.1. Параметр ядра *single*

Любой желающий может подойти к компьютеру, перезагрузить его и передать ядру Linux параметр `single`. В результате система будет загружена в однопользовательском режиме, а злоумышленник без лишних вопросов получит root-доступ. Время, необходимое на варварскую перезагрузку системы (нажатием кнопки Reset), — несколько миллисекунд, затем еще 15-30 секунд до появления загрузочного меню, еще несколько секунд на ввод параметров ядра и секунд 20-30 на загрузку Linux в однопользовательском режиме. Грубо говоря, через минуту злоумышленник сможет делать с вашей системой все, что захочет. Сможет даже с помощью команды `passwd root` изменить пароль root. Вот вам и одна из самых безопасных систем! С другой стороны, описанная тактика используется для восстановления пароля root в случае, если администратор системы страдает легкой формой склероза.

Однако делу можно помочь. Например, настроить систему так, чтобы она запрашивала пароль при загрузке в однопользовательском режиме. Но мы не будем этого делать. Почему? Да потому что это — не панацея. Злоумышленник может передать ядру другой параметр: `init=/bin/bash`. Параметр `init` задает программу инициализации системы. По умолчанию загружается программа `/sbin/init`, но, используя параметр `init`, можно запустить любую программу, и в описываемом случае будет выполнена команда `/bin/bash` — запущен командный интерпретатор. Вы уже догадались, что программа эта будет запущена с правами root. По умолчанию корневая файловая система монтируется в режиме `ro` (только чтение), поэтому, чтобы получить полный контроль над системой, злоумышленнику достаточно перемонтиро-

```

insmod part_msdos
insmod ext2
set root='hd0,msdos1'
if [ x$feature_platform_search_hint = xy ]; then
    search --no-floppy --fs-uuid --set=root --hint-bios=hd0,msdos1 --hin\
t-efi=hd0,msdos1 --hint-baremetal=ahci0,msdos1 --hint='hd0,msdos1' 146c8b5e-b\
796-4fcc-b05c-64c2c981b26a
else
    search --no-floppy --fs-uuid --set=root 146c8b5e-b796-4fcc-b05c-64c2\
c981b26a
fi
linux16 /vmlinuz-4.11.8-300.fc26.x86_64 root=/dev/mapper/fedora-root r\
o rd.lvm.lv=fedora/root rd.lvm.lv=fedora/swap rhgb init=/bin/bash LANG=ru_RU.U\
TF-8
initrd16 /initramfs-4.11.8-300.fc26.x86_64.img

Press Ctrl-x to start, Ctrl-c for a command prompt or Escape to
discard edits and return to the menu. Pressing Tab lists
possible completions.

```

Рис. 28.1. Fedora 26: передаем параметр ядра

```

[ 6.318322] scsi target2:0:0: FAST-40 WIDE SCSI 80.0 MB/s ST (25 ns, offset 1
27)
[ 6.318492] [drm] global init.
[ 6.324160] [TIM] Zone kernel: Available graphics memory: 1018984 kiB
[ 6.324161] [TIM] Initializing pool allocator
[ 6.324166] [TIM] Initializing DMA pool allocator
[ 6.324382] [drm] Supports vblank timestamp caching Rev 2 (21.10.2013).
[ 6.324382] [drm] No driver support for vblank timestamp query.
[ 6.325716] [drm] Screen Objects Display Unit initialized
[ 6.325989] [drm] width 1280
[ 6.326003] [drm] height 768
[ 6.326018] [drm] bpp 32
[ 6.464781] [drm] FIFO max 0x00200000 min 0x00001000 cap 0x000007ff
[ 6.583569] sd 2:0:0:0: [sda] 67108864 512-byte logical blocks: (34.4 GB/32.0
GiB)
[ 6.583756] sd 2:0:0:0: [sda] Write Protect is off
[ 6.583958] sd 2:0:0:0: [sda] Cache data unavailable
[ 6.583960] sd 2:0:0:0: [sda] Assuming drive cache: write through
[ 6.591325] sda: sda1 sda2
[ 6.591450] sd 2:0:0:0: Attached scsi generic sg1 type 0
[ 6.592743] sd 2:0:0:0: [sda] Attached SCSI disk
[ 6.634551] [drm] Using command buffers with DMA pool.
[ 6.634674] [drm] DX: no.
[ 6.823398] random: fast init done
[ 7.005068] fbcon: svgadrmfb (fb0) is primary device
[ 7.065622] Console: switching to colour frame buffer device 160x48
[ 7.078039] e1000 0000:02:01:0 eth0: (PCI:66MHz:32-bit) 00:0c:29:38:87:d5
[ 7.078049] e1000 0000:02:01:0 eth0: Intel(R) PRO/1000 Network Connection
[ 7.081731] [drm] Initialized vmwgfx 2.12.0 20170221 for 0000:00:0f:0 on minor 0
[ 7.084589] e1000 0000:02:01:0 ens33: renamed from eth0
[ 7.497485] EXT4-fs (dm-0): mounted filesystem with ordered data mode. Opts: (null)
[ 7.947452] systemd-journald[149]: Received SIGTERM from PID 1 (systemd).
[ 8.045056] bash: 12 output lines suppressed due to ratelimiting
bash-4.4# _

```

Рис. 28.2. Fedora 26: предоставлены права root

вать файловую систему в режиме `rw` (см. руководство `man mount`) — и снова можно творить с системой все, что заблагорассудится.

ИЗМЕНЯЕМ ПАРОЛЬ ROOT...

В папке *Видео* сопровождающего книгу электронного архива (см. *приложение*) содержится видеофайл, показывающий, как в современном дистрибутиве (на примере Debian 8) войти под именем пользователя `root` без пароля и установить новый пароль.

Эта «особенность» не устранена даже в самых последних дистрибутивах. Так, в Fedora 26 можно указать параметр ядра `init=/bin/bash` (рис. 28.1), и система будет загружена с правами `root` (рис. 28.2).

28.1.2. Пароль загрузчика GRUB

Защитить систему лучше с помощью загрузчика GRUB. Чтобы никто без вашего ведома не мог изменить параметры ядра Linux, следует установить пароль на изменение этих параметров. Тогда после установки пароля любую операционную систему можно будет загрузить без пароля, а вот при попытке изменения параметров ядра Linux GRUB запросит пароль.

О том, как установить пароль загрузчика, было рассказано в *разд. 21.8*. Цель этого небольшого раздела — напомнить вам о возможности установки пароля загрузчика, чтобы вы не забыли о нем, когда будете настраивать сервер.

28.1.3. Осторожно: LiveCD

Но это еще не все. Злоумышленник может загрузиться с LiveCD, после чего с помощью команды `chroot` заменить корневую файловую систему LiveCD файловой системой сервера и опять получить над ним полный контроль. Для предотвращения таких действий не забудьте установить пароль на вход в BIOS Setup, что не позволит злоумышленнику изменить порядок загрузки и загрузиться с LiveCD.

Впрочем, корпуса системных блоков современных компьютеров можно открыть даже без отвертки, причем за пару секунд, и еще за несколько секунд переустановить джампер, стирающий все настройки BIOS, в том числе и установленный пароль. Вы можете ничего не заметить, а злоумышленник тем временем внедрит *backdoor*-программу, позволяющую удаленно контролировать ваш сервер! Что ж, на такой случай желательно использовать специальные корпуса с замками, — замок, конечно, тоже можно взломать, но тут факт взлома будет, как говорится, налицо.

28.2. Защита от перезагрузки

От грубой перезагрузки защиты нет. Да, можно в BIOS Setup отключить кнопку `Reset`, а если такой опции там нет, то просто физически отключить разъем кнопки `Reset` в системном блоке. Но толку особого от этого не будет — если кто-то получит физический доступ к серверу, то ничего ему не мешает вытащить из розетки кабель питания. Вот и весь секрет...

Но мы можем блокировать хотя бы программную перезагрузку, осуществляемую с помощью клавиатурной комбинации `<Ctrl>+<Alt>+`.

Для этого в старых дистрибутивах, основанных на системе инициализации `init`, откройте файл `/etc/inittab` и найдите в нем строку:

```
ca:ictrlaltdel:/sbin/shutdown -r -t 4 now
```

Эту строку нужно закомментировать, а еще лучше — изменить так:

```
ca::ctrlaltdel:/bin/true
```

Как вы уже догадались, в таком случае система при нажатии комбинации клавиш `<Ctrl>+<Alt>+` никаких действий производить не станет. А для перезагрузки компьютера можно будет использовать команды `reboot` или `shutdown`.

Такой трюк подойдет для систем, использующих систему инициализации `init` (openSUSE, старые Fedora, CentOS, ранние версии Red Hat), однако в Ubuntu вы не найдете файла `inittab`, поэтому реакция на комбинацию клавиш `<Ctrl>+<Alt>+` отключается в Ubuntu иначе: откройте файл `/etc/init/control-alt-delete.conf` (рис. 28.3), найдите в нем строку:

```
exec shutdown -r now "Control-Alt-Delete pressed"
```

и закомментируйте ее с помощью символа `#`:

```
#exec shutdown -r now "Control-Alt-Delete pressed"
```

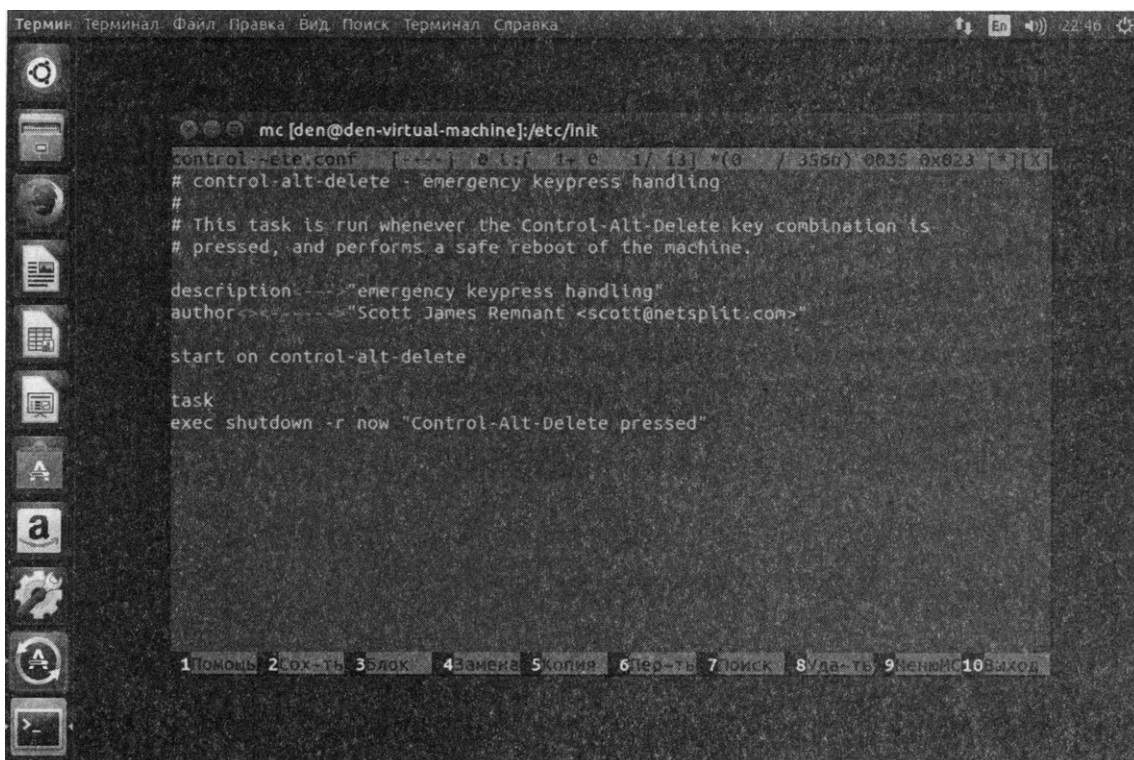


Рис. 28.3. Ubuntu: редактирование файла `/etc/init/control-alt-delete.conf`

В дистрибутивах, основанных на системе инициализации `systemd`, — например, в последних версиях Fedora, реакция на нажатие комбинации клавиш `<Ctrl>+<Alt>+` задается в файле `/lib/systemd/system/reboot.target` (рис. 28.4). При этом надо иметь в виду, что упомянутый в разд. 22.3.5 файл `/lib/systemd/system/ctrl-alt-del.target` является просто ссылкой на файл `/lib/systemd/system/reboot.target`.

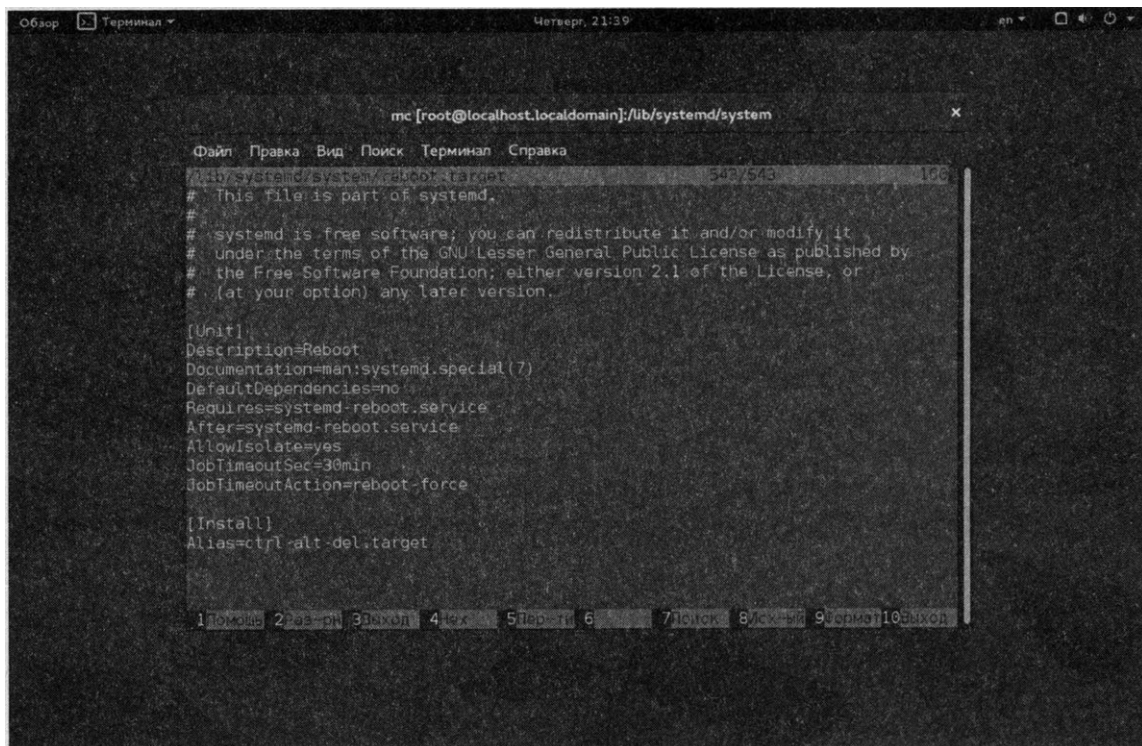


Рис. 28.4. Fedora: редактирование файла `/lib/systemd/system/reboot.target`

В разд. 22.3.5 также приведен способ изменить реакцию на нажатие комбинации клавиш `<Ctrl>+<Alt>+` — на уровне GNOME 3 (см. рис. 22.1 и 22.2).

28.3. Отключение учетной записи root: нестандартный метод

Довольно часто сервер настраивается по принципу — настроил и забыл. Да, после правильной настройки о нем забудете вы, но только не злоумышленники, которым не будет давать покоя ваша учетная запись `root`. Именно она — как учетная запись с максимальными правами — представляет наибольший интерес для злоумышленника.

Предлагаемый метод отключения учетной записи `root` довольно-таки варварский, но не более варварский, чем команда `rm -rf /`, которую может ввести «доброжелатель», получив доступ к `root`. Метод неудобен тем, что для входа в систему как `root` нужно перезагружать компьютер.

Прежде всего разрешим одному обычному пользователю (это ваша учетная запись, под которой вы работаете каждый день) выполнять некоторые команды от имени root. Для этого откройте файл `/etc/sudoers` и добавьте в него строку типа:

```
den localhost = NOPASSWD: /bin/kill, /sbin/reboot, /sbin/halt
```

Такая строка разрешает пользователю den выполнять команды `/bin/kill`, `/sbin/reboot`, `/sbin/halt` с привилегиями root на машине localhost без ввода пароля.

Можете дополнительно обезопасить систему, изменив приведенную строчку так:

```
den localhost = PASSWD: /bin/kill, /sbin/reboot, /sbin/halt
```

Тогда при вводе указанных команд будет запрошен пароль пользователя den (а не root!). Команды `kill`, `reboot` и `halt` надо будет вызывать через `sudo`:

```
sudo /bin/kill <PID>
sudo /sbin/reboot
sudo /sbin/halt
```

Еще раз отмечу, что сначала нужно разрешить обычному пользователю перезагружать компьютер, иначе после отключения учетной записи root и клавиатурной комбинации `<Ctrl>+<Alt>+` единственным способом перезагрузки останется кнопка Reset.

Вот теперь нужно открыть файл `/etc/passwd` и заменить строку:

```
root:x:0:0:root:/root:/bin/bash
```

следующей строкой:

```
root:x:0:0:root:/root:/bin/true
```

После этого в качестве командной оболочки пользователя root будет использоваться программа `/bin/true` — она запускается, возвращает истинное значение (для сценариев) и завершает работу (помните рассказ про «заглушки» из *разд. 5.11*). Сохраните файл `/etc/passwd`, введите команду `exit` и попробуйте войти как root — у вас ничего не получится. Теперь даже если злоумышленник и узнает пароль root, он не сможет им воспользоваться.

Для проверки войдите в систему как обычный пользователь и введите команду `su`. По правилам работы с этой командой у вас будет запрошен пароль root, но также будет запущена и оболочка root — команда `/bin/true`, которая немедленно завершит сеанс root.

А сейчас самое интересное — узнаем, как получить обратно root-доступ, когда он нам понадобится. Для этого нужно перезагрузить компьютер и передать ядру параметр `init=/bin/bash` — поскольку мы знаем пароль загрузчика, для нас это не составит проблемы (см. рис. 28.1 и 28.2).

После загрузки системы нужно перемонтировать корневую файловую систему в режиме `rw` (чтение/запись) командой типа:

```
mount -w -o remount /dev/sdal /
```

Конечно, фрагмент `/dev/sdal` нужно заменить в этой команде на имя раздела, на который установлена Linux. Уаля! Мы получили полный контроль над системой.

Теперь запускаем наш любимый `mc`, открываем файл `/etc/passwd` и изменяем оболочку `root`— в этот раз на `/bin/bash`, после чего перезагружаем машину командой `reboot`.

Теперь вы можете полноценно войти в систему как `root`. Но после завершения работы не забудьте вернуть все назад, как было.

28.4. Отключение учетной записи `root` средствами KDM и GDM

Довольно часто «доброжелатели» знают, что такое `root`, но не знают ни одной UNIX-команды — следовательно, не смогут причинить системе особого вреда в консоли, но могут натворить невесть что, зарегистрировавшись в графическом режиме (если, конечно, сервер загружается на пятом уровне запуска).

Так вот, KDM (K Display Manager) позволяет запретить пользователю `root` вход в графическом режиме. Конечно, для того чтобы приведенный далее совет работал, нужно, чтобы KDM был установлен как основной менеджер дисплея (обычно это так, если установлена графическая среда KDE).

Итак, откройте файл `/etc/kde/kdm/kdmrc` (для KDE3.5) или файл `/etc/alternatives/kdm4-config` (для KDE 4). Найдите в нем строку `AllowRootLogin=true` и замените ее строкой `AllowRootLogin=false`. Сохраните файл и завершите сеанс работы пользователя. После этого вы не сможете войти в систему под именем `root` в графическом режиме.

В KDE Plasma 5 по умолчанию используется графический менеджер SDDM, который сам пресекает попытки входа как `root`, поэтому изменять в нем какие-либо настройки вам не придется.

Если вы выбрали графическую среду GNOME, то в качестве графического менеджера в ней используется GDM. В нем также можно запретить вход пользователю `root`. Для этого откройте конфигурационный файл GDM (`/etc/X11/gdm/gdm.conf`) и добавьте в него команду (или отредактируйте ее, если она уже есть):

```
AllowRoot=false
```

В новой версии GDM, которая поставляется с GNOME 3, нужно редактировать файл `/etc/gdm/custom.conf`. В секцию `[security]` этого файла следует добавить эту же строку:

```
AllowRoot=false
```

Подробно о конфигурационном файле `custom.conf` можно прочитать по адресу: <http://projects.gnome.org/gdm/docs/2.14/configuration.html?pagewanted=all>.

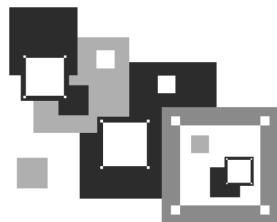
28.5. Системы управления доступом

Все описанные здесь действия довольно полезны, особенно учитывая, что о них часто забывают. Но все же они не дают полной защиты сервера. Если вы заботитесь о безопасности сервера, настоятельно рекомендую установить и настроить одну из *систем управления доступом* (LIDS, GrSecurity, SELinux, Тошоуо). К сожалению, рассмотрение системы управления доступом выходит за рамки этой книги, но в Интернете вы найдете всю необходимую информацию.

СИСТЕМА УПРАВЛЕНИЯ ДОСТУПОМ SELINUX

Описание системы управления доступом SELinux (*глава 33* из 2-го издания книги) вы найдете в папке *Дополнения* сопровождающего книгу электронного архива (см. *приложение*).

ГЛАВА 29



Модули аутентификации PAM

PAM (Pluggable Authentication Modules) — это подключаемые модули безопасности, предоставляющие администраторам дополнительные методы подтверждения подлинности пользователя. Механизм PAM разработан довольно давно — сначала он был экспериментальным, но потом прочно прижился в Linux, и многие администраторы не представляют, что бы делали без него.

Модули PAM позволяют использовать несколько схем аутентификации. Практически все приложения, нуждающиеся в проверке подлинности пользователя (POP, SSH и др.), применяют PAM.

Используя дополнительные модули PAM, можно изменить способ аутентификации. Обычно пользователь вводит имя пользователя и пароль для входа в систему. С помощью PAM можно организовать аутентификацию по сетчатке глаза, отпечаткам пальцев или по голосу. Наиболее простой в реализации способ — сканирование отпечатков пальцев. PAM-модули для его воплощения давно существуют, а сканеры отпечатков пальцев стоят относительно недорого.

АУТЕНТИФИКАЦИЯ ПО ОТПЕЧАТКАМ ПАЛЬЦЕВ

Если вас интересуют вопросы аутентификации по отпечаткам пальцев, посетите страницу: <http://www.dkws.org.ua/phpbb2/viewtopic.php?p=11698>.

Файлы конфигурации PAM находятся в каталоге `/etc/pam.d`, а библиотеки (модули) PAM, реализующие дополнительные функции аутентификации, хранятся в каталоге `/lib/security` (или `/lib64/security` — для 64-разрядных систем). В этой главе мы рассмотрим, как можно увеличить безопасность системы с помощью PAM.

29.1. Каталог `/etc/pam.d`

В каталоге `/etc/pam.d` размещаются файлы настроек, задающие набор модулей PAM, которые должна использовать та или иная программа. Имя файла совпадает с названием программы — например, для SSH-сервера это файл `sshd`, для программы `login` — файл `login` и т. д.

Формат записи всех таких файлов одинаковый:

Тип-Модуля Тип-Контроля Путь-К-Модулю Параметры-Модуля

Здесь поле `тип-модуля` задает тип модуля, а поле `тип-контроля` — режим контроля модуля, т. е. определяет, как система будет реагировать на удачное или неудачное завершение работы модуля (табл. 29.1). Назначение остальных двух полей, думаю, понятно: это путь к модулю и передаваемые ему параметры (зависят от самого модуля).

Таблица 29.1. Типы модулей PAM

Тип	Описание
Тип модуля	
auth	Модуль аутентификации. Проверяет наличие пользователя в системе, запрашивает его имя, разрешает или запрещает доступ в ту или иную группу, независимо от содержимого файла <code>/etc/groups</code> , а также может предоставлять пользователям определенные привилегии. Все зависит от самого модуля и от его параметров
account	Не занимается аутентификацией, а позволяет контролировать распределение ресурсов системы для тех или иных пользовательских аккаунтов
session	Связан с сеансом пользователя, а также с событиями, которые происходят перед обращением пользователя к той или иной службе. Например, такие модули могут вести записи в системных журналах
password	Модуль занимается проверкой пароля на подлинность, на стойкость к подбору и т. д.
Тип контроля	
required	Удача этого модуля требуется для всей аутентификации в целом. Неудача модуля с подобным флагом не проявится для пользователя, пока не выполнятся все оставшиеся модули
requisite	То же самое, что и <code>required</code> , но в случае неудачи управление сразу же будет передано приложению
sufficient	Неудача этого модуля не считается фатальной для всей последующей аутентификации. Но удачное его завершение считается достаточным для признания всей аутентификации успешной
optional	Этот модуль не критичен для аутентификации и используется как дополнительный. Модули с таким типом контроля могут, например, проверять силу пароля и выводить предупреждение о его слабости

29.2. Дополнительные файлы конфигурации

29.2.1. Содержимое каталога `/etc/security`

В каталоге `/etc/security` находятся дополнительные файлы конфигурации:

- `access.conf`— управляет доступом к системе, позволяет задать не только, кто и куда может зайти, но и откуда. Эти настройки обслуживает модуль `pam_access`;
- `console.perms`— определяет права, получаемые привилегированными пользователями к консоли во время входа в систему и возвращаемые при выходе. Эти настройки обслуживает модуль `pam_console`;

- **group.conf** — позволяет указать, какой группе будет принадлежать служба, запущенная определенным пользователем в определенное время с определенного терминала. Эти настройки обслуживают модули **pam_time** и **pam_group**;
- **limits.conf** — позволяет ограничить системные ресурсы. Эти настройки обслуживает модуль **pam_limits** (см. *разд. 29.2.2*);
- **pam_env.conf** — здесь можно ограничить возможности пользователей изменять определенные переменные среды. Эти настройки обслуживает модуль **pam_env**;
- **time.conf** — позволяет ограничивать вход пользователей в систему по времени. Например, с его помощью можно запретить вход администратора с первой консоли по выходным. Эти настройки обслуживает модуль **pam_time**.

29.2.2. Файл **access.conf**: ограничение доступа к системе

Файл **access.conf** позволяет ограничить доступ пользователей к вашему компьютеру.

Предположим, что пользователям **den** и **admin** разрешено администрировать сервер, а остальным пользователям — нечего даже делать у консоли сервера (и это правильно!).

Тогда в файл **access.conf** нужно добавить строку:

```
-:ALL EXCEPT root den admin:ALL
```

Такая запись означает: запретить регистрацию в системе всем пользователям, кроме пользователей **root**, **den** и **admin** — эти три пользователя могут регистрироваться с любой консоли и с любого IP-адреса (если регистрация происходит по SSH).

Довольно часто администратор работает за своим компьютером — отдельной рабочей станцией — и будет регистрироваться на сервере только с этого компьютера. В таком случае в файл **access.conf** можно добавить строки следующего вида:

```
-:ALL EXCEPT root: 192.168.1.2  
-:ALL: LOCAL
```

Здесь **192.168.1.2** — адрес рабочей станции администратора, вход на сервер будет разрешен только с этого IP-адреса. Вторая строка запрещает локальный доступ всех пользователей и даже пользователя **root**. Будьте осторожны с этой строкой — если что-то случится с компьютером **192.168.1.2** или, вообще, с сетью, вы не сможете зайти в систему.

В листинге 29.1 приведены полезные примеры, которые можно использовать в файле **access.conf**.

Листинг 29.1. Полезные примеры ограничения доступа (файл **/etc/security/access.conf**)

```
# Для включения той или иной возможности нужно раскомментировать  
# соответствующую ей строку  
  
# Запрещает регистрацию всех пользователей, кроме root, на первой консоли (tty1)  
# -:ALL EXCEPT root:tty1
```

```
# Запрещает локальную регистрацию в системе всем пользователям, кроме
# пользователей den и admin
#-:ALL EXCEPT den admin:LOCAL
#
# Запрещает локальную регистрацию всех пользователей, кроме
# членов группы admins
#-:ALL EXCEPT (admins):LOCAL
#
# Запрещает пользователям user1 и user2 любой вход в систему.
# Остальные пользователи могут входить в систему любым способом
# (консоль, SSH, FTP и др.)
#-:wsbscaro wsbsecre wsbspac wsbsym wscosor wstaiwde:ALL
#
# Пользователь root может регистрироваться только со следующих IP:
#+ : root : 192.168.200.1 192.168.200.4 192.168.200.9
#+ : root : 127.0.0.1
#
# Пользователь root может регистрироваться только из сети 192.168.201.
#+ : root : 192.168.201.
#
# Запрещает доступ пользователя root в систему
#- : root : ALL
```

Но одного редактирования файла **access.conf** для задействия его опций недостаточно. Чтобы система и SSH при проверке подлинности пользователей использовала PAM, нужно в файлы **/etc/pam.d/system-auth** и **/etc/pam.d/sshd** добавить следующую строку:

```
account required /lib/security/pam_access.so
```

Модули для 64-разрядных систем

Не забывайте, что для 64-разрядных систем соответствующие модули находятся в каталоге **/lib64/security**!

29.2.3. Файл limits.conf: ограничение на используемые системные ресурсы

Один из видов атак (атака на отказ, DoS) заключается в загрузке программой злоумышленника всех системных ресурсов, в результате чего система оказывается не в состоянии выполнять полезные действия и обслуживать других пользователей. Для защиты от таких атак можно использовать файл **/etc/security/limits.conf**, позволяющий установить лимиты на использование системных ресурсов. Формат файла следующий:

```
домен тип элемент значение
```

Здесь поле домен — это имя пользователя или имя группы (группа указывается так: @имя). Если нужно, чтобы ограничение распространялось на всех пользователей и на все группы, следует указать *.

Второе поле задает тип ограничения:

- `soft` — «мягкое» ограничение, которое еще можно незначительно превысить;
- `hard` — «жесткое» ограничение, превысить которое уже нельзя.

В качестве поля элемент можно использовать следующие значения:

- `core` — ограничивает размер файла ядра (в Кбайт);
- `data` — максимальный размер сегмента данных (в Кбайт);
- `fsize` — максимальный размер файла (в Кбайт);
- `nofile` — максимальное число одновременно открытых файлов;
- `nproc` — количество процессов, которые может запустить пользователь;
- `stack` — максимальный размер стека (в Кбайт);
- `cpu` — максимальное процессорное время (в минутах);
- `maxlogins` — максимальное количество регистраций пользователя (в Linux по умолчанию разрешается одному пользователю войти в систему неограниченное количество раз: с разных консолей, по SSH, FTP и т. д.);
- `priority` — приоритет, с которым будут выполняться процессы пользователя/группы.

Последнее поле значение задает сам лимит для выбранного элемента — например:

- для группы `students` установлен жесткий лимит на количество процессов, равный 30:

```
@students hard nproc 30
```

- пользователю `ftp` вообще запрещено запускать какие-либо процессы:

```
ftp hard nproc 0
```

29.2.4. Файл `time.conf`: регистрация только в рабочее время

Целесообразно разрешить пользователям регистрироваться в системе только в рабочее время — вне рабочего времени им делать в системе нечего.

Откройте файл `/etc/security/time.conf` и добавьте в него следующую строку:

```
login;tty* & !ttyp*; !root & den & ; !A10800-1800
```

После этого откройте файлы `/etc/pam.d/system-auth` и `/etc/pam.d/sshd` и добавьте в них строку:

```
account required /lib/security/pam_time.so
```

Таким образом регистрация в системе всем пользователям (кроме пользователей root и den) разрешена только в рабочее время (с 8-00 до 18-00). Пользователи root и den могут регистрироваться в любое время суток.

29.3. Список PAM-модулей

Итак, мы рассмотрели основные файлы конфигурации и уже успели познакомиться с некоторыми PAM-модулями. Осталось разобраться, какие модули вообще существуют и для чего они используются. Параметры конкретного модуля вы сможете узнать из справочной системы man, а в табл. 29.2 описаны основные модули и тип контроля для каждого из них.

Таблица 29.2. Модули PAM

Модуль	Тип контроля	Описание
pam_cracklib	password	Проверяет пароль на стойкость. К полезным параметрам модуля можно отнести следующие: minlen=N (минимальная длина пароля), dcredit=N (минимальное количество цифр), lcredit=N (количество строчных букв), ocredit (количество прописных букв и других символов). Подробно этот модуль рассмотрен в разд. 29.4
pam_deny	Любой	Всегда закрывает доступ
pam_env	auth	Контролирует сохранность переменных среды
pam_ftp	auth	Предназначен для организации анонимного доступа по FTP. Получив имя пользователя anonymous, ждет что-то напоминающее e-mail в качестве пароля
pam_group	auth	Предназначен для обслуживания файла groups, conf
pam_lastlog	auth	Сообщает о времени и месте входа в систему. Обновляет файл /var/log/wtmp
pam_limits	session	Обслуживает файл limits.conf
pam_listfile	auth	Предназначен для организации доступа на основе конфигурационных файлов-списков — например, /etc/ftpaccess
pam_mail	auth	Сообщает о наличии почты, если таковая имеется
pam_nologin	auth	Стандартная реакция на наличие файла /etc/nologin. Если он присутствует, в систему может зайти только root, а остальным будет выдано на экран содержимое этого файла
pam_permit	Любой	Всегда означает успешную аутентификацию. Использование этого модуля опасно
pam_pwdb	Любой	Замещает модули серии pam_unix. Использует интерфейс библиотеки libpwdb (пользовательские базы данных), что повышает независимость системы аутентификации от способа хранения пользовательских данных

Таблица 29.2 (окончание)

Модуль	Тип контроля	Описание
pam_radius	session	Аутентификация через RADIUS-сервер
pam_rhosts_auth	auth	Анализирует содержимое файлов <code>hosts.equiv</code> и <code>.rhosts</code> , используемых для аутентификации таких служб, как <code>rlogin</code> и <code>rsh</code>
pam_root_ok	auth	Используется в случае, когда администратору необходимо получать доступ к сервису без введения пароля
pam_securetty	auth	Обслуживает файл <code>/etc/securetty</code> . Пользователь <code>root</code> может зайти с консоли, указанной в этом файле
pam_time	account	Этот модуль обслуживает файл <code>time.conf</code> (см. разд. 29.2.4)
pam_warn	auth или password	Ведет записи в системных журналах
pam_wheel	auth	Некая эмуляция BSD в Linux. В BSD получить права <code>root</code> может только пользователь группы <code>wheel</code> (в Linux она называется <code>root</code>). Можно отметить параметр <code>group</code> , задающий определенную группу вместо группы с GID 0

29.4. Борьба с простыми паролями

Пользователи частенько стараются облегчить себе жизнь и устанавливают очень простые пароли, которые очень просто и взламываются — точнее, подбираются. В файле `/etc/pam.d/system-auth` администратор может указать, каким он хочет видеть безопасный пароль.

Характеристики пароля задаются следующими параметрами:

- `minlen=N` — минимальная длина пароля (нужно как минимум 6 символов — это исключит короткие и легко подбираемые пароли);
- `dcredit` — если задан этот параметр, то допустимое минимальное количество символов будет уменьшено на величину этого параметра при условии, что в пароле есть хотя бы одна цифра.

Суть этого параметра в следующем. Допустим, вы установили длину пароля в 10 символов, а пользователь задал пароль, содержащий одну цифру. В таком случае значение параметра `dcredit` (равное количеству цифр в пароле) окажется равным 1, и минимальная длина пароля будет уменьшена на 1, т. е. станет равна 9;

- `ucredit` — то же самое, что и `dcredit`, но определяется наличием буквы в верхнем регистре;
- `lcredit` — то же самое, что и `dcredit`, но определяется наличием буквы в нижнем регистре (применяется редко, но не позволяет пользователям перехитрить систему и использовать короткий пароль, состоящий из букв в верхнем регистре);

- `ocredit` — то же самое, что и `dcredit`, но определяется наличием специального символа;
- `retry=N` — количество попыток ввода пароля, после чего учетная запись будет заблокирована.

Параметры пароля устанавливаются так:

```
password required /lib/security/pam_cracklib.so параметры
```

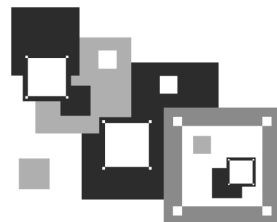
Например:

```
password required /lib/security/pam_cracklib.so retry=3 minlen=10 dcredit ocredit
```

Здесь мы установили три попытки ввода пароля, минимальную длину пароля в 10 символов и «льготные» параметры `dcredit` и `ocredit`. Согласно нашим правилам, подойдут следующие пароли:

- `secretpassword` — длина 14 символов, что удовлетворяет условиям параметра `minlen=10`;
- `password7` — длина 9 символов, но поскольку используется параметр `dcredit`, и в пароле имеется одна цифра, такой пароль допускается;
- `top_passwd` — длина 9 символов, но поскольку используется параметр `ocredit`, и в пароле есть один специальный символ, такой пароль допускается.

ГЛАВА 30



Оптимизация системы. Автоматизация выполнения задач

30.1. Оптимизация подкачки

Оперативная память — это весьма критичный для Linux ресурс. Даже более критичный, чем частота процессора, поэтому нехватка оперативной памяти в Linux ощущается очень остро — иногда работать становится просто невыносимо.

При установке Linux создается раздел подкачки (swap, своп), который задействуется, если системе не хватает физической оперативной памяти, — на него сгружается неиспользуемая в текущий момент информация, а в освобожденную таким образом оперативную память с жесткого диска подгружаются необходимые процессору данные. Ясно, что система с разделом подкачки работает медленнее, чем с модулем оперативной памяти, но все же она работает быстрее и стабильнее, нежели вообще без раздела подкачки.

Сама операционная система Linux не очень требовательна к памяти — для нормальной работы даже шлюза небольшой сети вполне хватит 64 Мбайт оперативной памяти. Не верите? Посмотрите на рис. 30.1 — из 128 Мбайт использовано всего 33 Мбайт, а раздел подкачки (**Swap**) вообще не задействован.

```
[root@localhost ~]# free
              total        used         free       shared    buffers     cached
Mem:          126284       33244        93040           0        4584       16152
-/+ buffers/cache:      12508        113776
Swap:         240964           0        240964
[root@localhost ~]# _
```

Рис. 30.1. Команда free — сведения об использовании оперативной памяти

Но это только в том случае, если не запущена система X.Org. После ее запуска Linux превращается в настоящего «обжору», съедающего десятки мегабайтов памяти. Сама X.Org тоже не особенно требовательна к памяти, чего не скажешь о графических интерфейсах GNOME и ЮЗЕ, — при их использовании для комфортной работы необходимо минимум 1 Гбайт ОЗУ. Так что, ваша система может работать, мягко говоря, не очень быстро только потому, что ей не хватает оперативной памяти.

Попытаемся определить, хватает ли оперативной памяти вашему компьютеру,— запустите те программы, с которыми вы чаще всего работаете: LibreOffice Writer, LibreOffice Calc, xmms, GIMP — не все сразу, а только те, которые вы часто используете одновременно. Затем введите команду `free` и посмотрите, сколько мегабайтов оперативной памяти у вас свободно. Обратите внимание и на «остаток» области подкачки. Если и там, и там осталось всего несколько мегабайтов памяти, значит, вам пора купить еще один модуль ОЗУ.

Временно, пока вы его не купили, можно в помощь разделу подкачки создать специальный файл подкачки, что несколько повысит производительность системы. Хочу, однако, обратить ваше внимание на то, что это мера временная, ведь производительность жесткого диска существенно ниже производительности оперативной памяти, следовательно, даже если вы добавите к области подкачки файл подкачки объемом в 1 Гбайт, это не сравнится с одним настоящим модулем памяти на 512 Мбайт. С другой стороны, быстрый SSD-накопитель может компенсировать нерасторопность файла/раздела подкачки. Хотя, если ваш компьютер оборудован SSD-накопителем, вы вряд ли испытываете нехватку оперативной памяти.

В *разд. 30.2* мы научимся создавать файл подкачки. Но одного добавления своп-файла мало. Нужно еще оптимизировать работу системы свопинга с помощью коэффициента подкачки. Значение этого коэффициента хранится в файле `/proc/sys/vm/swappiness`. Минимальное значение коэффициента — 0, максимальное — 100, значение по умолчанию — 70.

Очень важно правильно выбрать оптимальное значение коэффициента подкачки. Если вы в основном работаете с небольшими программками и часто переключаетесь между ними, можно установить значение меньше 50, например 40 или даже 30. В этом случае переключение между приложениями будет мгновенным, но замедлится их работа. Впрочем, поскольку эти приложения небольшого размера, то вы этого не заметите.

Если же вы большую часть времени работаете на протяжении дня с громоздкими приложениями, — например, с LibreOffice, или занимаетесь обработкой изображений в GIMP, вам лучше установить значение коэффициента, превышающее 70, — например, 80 или даже 85. В этом случае переключение между приложениями станет медленным, зато ваше основное приложение будет работать быстро.

Изменить значение коэффициента можно с помощью команды:

```
# echo "значение" > /proc/sys/vm/swappiness
```

Например:

```
# echo "50" > /proc/sys/vm/swappiness
```

30.2. Создание файла подкачки

Если при установке Linux вы пожадничали и создали раздел подкачки недостаточного размера, делу можно помочь даже без переразметки жесткого диска, — существует возможность создать файл подкачки, который будет использоваться в паре с разделом подкачки.

Итак, создадим файл `/swap_file` размером 512 Мбайт:

```
# dd if=/dev/zero of=/swap_file bs=1k count=524288
```

Файл `/swap_file` пока еще нельзя назвать файлом подкачки, поскольку мы его не отформатировали как файл подкачки. Сделаем это:

```
# mkswap /swap_file 524288
```

Теперь осталось активировать только что созданный файл подкачки:

```
# swapon /swap_file
```

Последнюю команду нужно добавить в сценарии автозапуска для того, чтобы не вводить ее при каждом запуске системы (см. главу 21).

30.3. Настройка планировщика ввода/вывода

Производительность многозадачной системы в целом сильно зависит от правильного планирования процессов системы. Сейчас мы попытаемся с помощью параметра ядра `elevator` установить нужный нам алгоритм работы ядра, что позволит существенно повысить производительность системы. Допустимы следующие значения этого параметра:

- `none` — значение по умолчанию;
- `as` — упреждающее планирование;
- `cfq` — «честная очередь»;
- `deadline` — планирование крайних сроков.

Для домашнего компьютера больше подойдут значения `as` и `cfq`:

- в первом случае (значение `as`) ядро будет пытаться «угадать» ход программы, а именно: какую операцию ввода/вывода программа «захочет» выполнить в следующий раз. Если ядро будет правильно «угадывать», то производительность системы существенно увеличится. Ясно, что работа этого алгоритма очень зависит от логики программы;
- во втором случае (значение `cfq`) ядро будет равномерно планировать операции ввода/вывода. Этот алгоритм будет работать лучше первого в случае с запутанной логикой программы, когда невозможно предугадать ее следующую операцию;
- последнее значение (`deadline`) больше подходит для сервера, чем для рабочей станции, поэтому существенного прироста от него не ждите.

При загрузке передать параметр ядра можно так:

```
linux elevator=3Na4eHne
```

Чтобы не вводить параметр каждый раз при загрузке, добавьте его в файл конфигурации загрузчика (см. главу 21).

30.4. Двухканальный режим памяти

Существенно повысить производительность операционной системы (причем, не только Linux, но и любой другой) можно путем использования двухканального режима работы памяти. В этом режиме пропускная способность модулей DDR2/DDR3 в два раза выше, чем в одноканальном режиме.

Первым делом убедитесь, что ваша материнская плата поддерживает двухканальный режим памяти (в этом вам поможет руководство по материнской плате), — впрочем, практически все современные материнские платы поддерживают двухканальный режим. Косвенно о поддержке этого режима можно судить по количеству слотов оперативной памяти — если их количество четное (2 или 4), то, скорее всего, двухканальный режим поддерживается.

Затем нужно установить в слоты четное количество модулей (2 или 4) оперативной памяти одинаковой емкости и с одинаковыми характеристиками (частота, тайминги). Лучше всего установить одинаковые модули (одного и того же производителя) — тогда вам гарантирована полная совместимость.

К слову, если вам нужно 8 Гбайт оперативной памяти, то имеет смысл установить два модуля по 4 Гбайт, а не один модуль емкостью 8 Гбайт. В первом случае (когда есть два модуля) будет активирован двухканальный режим памяти, а во втором (один модуль) — нет.

30.5. Автоматизация выполнения задач

Очень часто приходится периодически выполнять одни и те же действия. Например, каждый день проверять обновление антивируса (или раз в неделю — в зависимости от того, как часто выходят для него обновления) или каждые 30 минут — почту. Можно выполнять эти действия самому, но этот вариант не лучший. Представьте, что ваш рабочий день будет начинаться с команды запуска программы обновления антивируса, а каждые 30 минут вы будете вводить программу проверки почты. Во-первых, это не очень удобно, а во-вторых, можно легко забыть выполнить ту или иную команду. Например, в пятницу вечером вы можете забыть выполнить команду создания резервной копии, а в понедельник утром что-то случится с сервером, и вы не досчитаетесь всего пользовательского каталога. Не очень приятно, правда?

30.5.1. Планировщик `crond`

В Linux есть специальный демон `crond`, позволяющий выполнять программы по расписанию. Откройте конфигурационный файл демона `crond` — `/etc/crontab` (листинг 30.1).

Листинг 30.1. Пример файла /etc/crontab

```

SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
HOME=/

# run-parts
01 * * * * root nice-nl9run-parts-report/etc/cron.hourly
02 4 * * * root nice-nl9run-parts-report/etc/cron.daily
22 4 * * 0 root nice-nl9run-parts-report/etc/cron.weekly
42 4 1 * * root nice-nl9run-parts-report/etc/cron.monthly

```

Параметр `SHELL` задает имя программы-оболочки, параметр `PATH` — путь поиска программ, `MAILTO` — имя пользователя, которому будет отправлен отчет о выполнении расписания, а `HOME` — домашний каталог `crond`.

Но самое главное — не эти параметры, а сама таблица расписаний, занимающая в нашем случае последние четыре строки листинга. Согласно этой таблице каждый час будут выполняться программы из каталога `/etc/cron.hourly`, каждый день — из каталога `/etc/cron.daily`, каждую неделю — из каталога `/etc/cron.weekly`, а раз в месяц — из каталога `/etc/cron.monthly`.

Предположим, вам нужно каждый день выполнять команду `updateav ftp://server.ru/bases/`. В каталоге `/etc/cron.daily` создайте файл `update_av` следующего содержания:

```

#!/bin/bash
update_av ftp://server.ru/bases/

```

Этот файл представляет собой небольшой `bash`-сценарий (сценарий командного интерпретатора). Теперь сделаем его исполнимым:

```
# chmod +x update_av
```

Правда, удобно?

Но иногда нам бывает нужно создать более гибкое расписание. Например, мы хотим, чтобы одна программа выполнялась в 7:00, а другая в 7:20. Тут простым добавлением сценария в каталог `/etc/cron.daily` уже не отделаешься. Чтобы создать такое расписание, вам придется изучить формат записей таблицы расписаний:

минуты (0–59) часы (0–23) день (1–31) месяц (1–12) день_недели (0–6, 0 — Вс)
команда

Чтобы реализовать наше расписание, следует добавить в файл `/etc/crontab` следующие строки:

```

0 7 * * * /usr/bin/command1 arguments
20 7 * * * /usr/bin/command2 arguments

```

Первая команда будет запускаться каждый день в 7 часов утра, а вторая — тоже каждый день, но в 7:20.

Зная формат файла `crontab`, мы можем отредактировать стандартную таблицу расписаний (см. листинг 30.1). Обратите внимание — команды, выполняемые ежедневно, будут запускаться в 4 часа утра. Это, конечно, удобно, но они не будут выполнены, если вы выключаете сервер на ночь. Поэтому давайте установим другое время, например 8 часов утра:

```
02 8 *** root nice -n 19 run-parts --report /etc/cron.daily
```

Аналогичная ситуация и с еженедельным запуском. Программы будут запущены не только в 4:22 утра, но еще и в воскресенье. Однако на выходные вы точно выключаете свой сервер (впрочем, это зависит от политики организации — в некоторых организациях на выходные все компьютеры и не выключают). Поэтому целесообразно назначить запуск на понедельник в 8 часов 22 минуты:

```
22 8 ** 1 root nice -n 19 run-parts --report /etc/cron.weekly
```

С ежемесячным запуском вроде бы все нормально — программы будут выполняться в 4:42 первого числа каждого месяца. Хотя время лучше изменить на 8:42:

```
42 8 1 ** root nice -n 19 run-parts --report /etc/cron.monthly
```

30.5.2. Планировщик `anacron`

Планировщик `anacron` — непосредственный родственник `crond`, дальнейшее его развитие. Главное преимущество `anacron` заключается в том, что он, в отличие от `crond`, учитывает время, когда компьютер был выключен. Планировщик `crond` родом из UNIX, а эта операционная система устанавливалась только на серверах, которые всегда включены. Предположим, что вам нужно каждый понедельник в 7 часов утра рассылать некоторую информацию вашим сотрудникам. Вы настроили `crond` так, чтобы он запускал сценарий отправки сообщений каждый понедельник в 7 утра. Но вот беда — в 6 часов утра выключили электричество, а включили его, скажем, в 7:20. Но 7:20 — это не 7:00, следовательно, `crond` не выполнит задание по отправке сообщений, а ваши сотрудники не получают важную информацию.

`Anacron` работает не так. Если он обнаружил, что некоторые задания не выполнены по тем или иным причинам (выключение электричества, перезагрузка компьютера), он обязательно выполнит их. Поэтому ваши сотрудники получают информацию, но с небольшой задержкой. Все же лучше, чем получить важную информацию лишь в следующий понедельник.

Но и у `anacron` есть свои недостатки. В частности, пользователи не могут создавать свои собственные расписания, а файл `/etc/anacrontab` может редактировать только `root`. К тому же, более старый `crond` является и более гибким в настройке — например, вы можете точно указать часы и минуты, а в случае с `anacron` можно задать только период, когда будет выполнена команда.

Формат файла `/etc/anacrontab` выглядит так:

Период	Задержка	ID	Команда
--------	----------	----	---------

Например:

1	5	cron.daily	run-parts /etc/cron.daily
7	10	cron.weekly	run-parts /etc/cron.weekly
30	75	cron.monthly	run-parts /etc/cron.monthly

30.5.3. Разовое выполнение команд — демон atd

Иногда нужно просто выполнить определенные команды в определенное время (однократно), поэтому редактировать для этого таблицу crontab не совсем уместно. Такую задачу можно решить более рационально. Убедитесь, что у вас установлен и запущен демон atd. После этого введите команду:

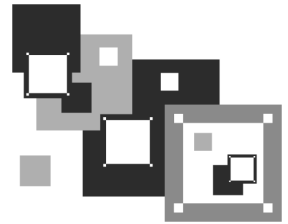
```
at <время> [дата]
```

Затем просто вводите команды, которые вы хотите выполнить в указанное время. Для завершения ввода нажмите комбинацию клавиш <Ctrl>+<D>. Время указывается в АМ/РМ-формате— например, если вам нужно выполнить команды в 14:00, то вы должны ввести команду: `at 2pm`.

Просмотреть очередь заданий можно командой `atq`, а удалить какое-либо задание — командой `atrm`.

В целях повышения безопасности в файл `/etc/at.deny` можно добавить команды, которые запрещены для выполнения планировщиком at.

ГЛАВА 31



Маршрутизация. Настройка брандмауэра

Маршрутизация— это процесс перенаправления пакета по сетям, находящимся между отправителем и получателем. Представьте, что вам нужно поехать в гости к другу в город, в котором вы никогда не были. Понимаю, что на дворе XXI век и GPS-навигатор — теперь аксессуар не только Джеймса Бонда, но все же о навигаторах на минуту забудем. Итак, сначала вам надо выяснить, как проехать в город, в котором живет ваш друг. Если вы живете в относительно большом городе, то первым делом нужно узнать, куда выехать из своего города, — можно, конечно, выехать в любом направлении, но потом, возможно, придется делать лишний крюк, чего бы не хотелось. Поэтому выясняем у встречного таксиста начальное направление. Выбравшись из своего города и зная примерное направление, вы себе спокойно едете, пока не начнете сомневаться в правильности маршрута. Тогда вы остановитесь на придорожной АЗС или у поста ДПС и узнаете, куда вам ехать дальше. Возможно, придется проехать еще через несколько городов, и в каждом из них вам нужно будет уточнить маршрут. Можно и не спрашивать — если есть знаки. Одним словом, либо человек, либо дорожный знак укажут вам дорогу. Когда вы приедете в город друга, вам нужно будет узнать, где находится улица, на которой он живет. А когда вы окажетесь на нужной улице, наверняка попросите прохожих подсказать, где находится дом с искомым номером.

Маршрутизация пакетов выполняется примерно так же. В приведенном примере с путешественником «пакетом» были именно вы, а роль маршрутизаторов играли люди, которые подсказывали вам, куда ехать.

В TCP/IP-сетях информация о маршрутах имеет вид *правил*— например, чтобы добраться до сети А, нужно отправить пакеты через компьютер Д. Ничего удивительного и необычного — примерно так же выглядит и информация о маршрутах на дороге: чтобы доехать до города А нужно проехать через город Д. Кроме набора маршрутов есть также и стандартный маршрут— по нему отправляют пакеты, предназначенные для отправки в сеть, маршрут к которой явно не указан. Компьютер, на который отправляются такие пакеты, называется *шлюзом по умолчанию* (default gateway). Получив пакет, шлюз решает, что с ним сделать: или отправить дальше, если ему известен маршрут в сеть получателя пакета, или же уничтожить пакет, как будто бы его никогда и не было. В общем, что сделать с пакетом — это личное дело шлюза по умолчанию, все зависит от его набора правил маршрутизации. А наше дело маленькое — отправить пакет на шлюз по умолчанию.

Данные о маршрутах хранятся в таблице маршрутизации ядра Linux. Каждая запись этой таблицы содержит несколько параметров: адрес сети назначения, сетевую маску и т. д. Если пакет не удалось отправить ни по одному маршруту (в том числе и по стандартному), отправителю пакета передается ICMP-сообщение «сеть недоступна» (network unreachable).

31.1. Таблица маршрутизации ядра. Установка маршрута по умолчанию

Для просмотра таблицы маршрутизации служат команды `netstat -r` и `netstat -rn`. Можно также по старинке воспользоваться командой `route` без параметров. Разница между командами `netstat -r` и `netstat -rn` заключается в том, что параметр `-rn` запрещает поиск доменных имен в DNS, поэтому все адреса будут представлены в числовом виде (подобно команде `route` без параметров). А вот разница между выводом `netstat` и `route` заключается в представлении маршрута по умолчанию (`netstat` выводит адрес 0.0.0.0, а `route` — метку `default`) и в названии полей самой таблицы маршрутизации.

Какой командой пользоваться — дело вкуса. Раньше я использовал `route` и для просмотра, и для редактирования таблицы маршрутизации. Теперь для просмотра таблицы я отдаю команду `netstat -rn`, а для ее изменения — команду `route`.

На рис. 31.1 показан вывод команд `netstat -rn` и `route`. В выводе каждой команды можно видеть две сети: 192.168.181.0 и 169.254.0.0 — и обе на интерфейсе `eth0`.

```

denix@denix-desktop: ~
Файл Правка Вид Терминал Справка
denix@denix-desktop:~$ netstat -rn
Таблица маршрутизации ядра протокола IP
Destination Gateway Genmask Flags MSS Window irtt Iface
192.168.181.0 0.0.0.0 255.255.255.0 U 0 0 0 eth0
169.254.0.0 0.0.0.0 255.255.0.0 U 0 0 0 eth0
0.0.0.0 192.168.181.2 0.0.0.0 UG 0 0 0 eth0
denix@denix-desktop:~$ route
Таблица маршрутизации ядра протокола IP
Destination Gateway Genmask Flags Metric Ref Use Iface
192.168.181.0 * 255.255.255.0 U 1 0 0 eth0
link-local * 255.255.0.0 U 1000 0 0 eth0
default 192.168.181.2 0.0.0.0 UG 0 0 0 eth0
denix@denix-desktop:~$

```

Рис. 31.1. Команды `netstat -rn` и `route`

Такая ситуация сложилась из-за особенностей NAT/DHCP виртуальной машины VMware, в которой была запущена Linux для снятия снимков с экрана. В реальных условиях вы, как правило, увидите по одной подсети на одном интерфейсе. С другой стороны, рис. 31.1 демонстрирует поддержку VLAN, когда один интерфейс может использоваться двумя подсетями. Шлюзом по умолчанию здесь является компьютер с адресом 192.168.181.2, о чем и свидетельствует таблица маршрутизации в выводе каждой команды.

Поля таблицы маршрутизации объясняются в табл. 31.1.

Таблица 31.1. Поля таблицы маршрутизации

Поле	Описание
Destination	Адрес сети назначения
Gateway	Шлюз по умолчанию
Genmask	Маска сети назначения
Flags	Поле Flags содержит флаги маршрута: <ul style="list-style-type: none"> • U — маршрут активен; • H — маршрут относится не к сети, а к хосту; • G — данная машина является шлюзом, поэтому при обращении к ней нужно заменить MAC-адрес машины получателя на MAC-адрес шлюза (если MAC-адрес получателя почему-то известен); • D — динамический маршрут, установлен демоном маршрутизации; • M — маршрут, модифицированный демоном маршрутизации; • C — запись кэширована; • ! — запрещенный маршрут
Metric	Метрика маршрута, т. е. расстояние к цели в хопх (переходах). Один хоп (переход) означает один маршрутизатор
Ref	Количество ссылок на маршрут. Не учитывается ядром Linux, но в других операционных системах, — например, в FreeBSD, вы можете столкнуться с этим полем
Use	Содержит количество пакетов, прошедших по этому маршруту
Iface	Используемый интерфейс
MSS	Максимальный размер сегмента (Maximum Segment Size) для TCP-соединений по этому маршруту
Window	Размер окна по умолчанию для TCP-соединений по этому маршруту
irtt	Протокол TCP гарантирует надежную доставку данных между компьютерами. Такая гарантия обеспечивается повторной отправкой пакетов, если они были потеряны. При этом ведется счетчик времени: сколько нужно ждать, пока пакет дойдет до назначения и придет подтверждение о получении пакета. Если время вышло, а подтверждение-таки не было получено, то пакет отправляется еще раз. Это время и называется round-trip time (время «путешествия туда-обратно»). Параметр irtt — это начальное время rtt . В большинстве случаев подходит значение по умолчанию, но для некоторых медленных сетей, например для сетей пакетного радио, значение по умолчанию слишком короткое, что вызывает ненужные повторы. Параметр irtt можно увеличить командой <code>route</code> . По умолчанию его значение — 0

Добавить маршрут в таблицу маршрутизации можно статически (с помощью команды `route`), динамически или комбинированно (например, статические маршруты добавляются при запуске системы, а динамические — по мере работы системы). Статические маршруты добавляются, как правило, командой `route`, запущенной из сценария инициализации системы. Например, следующая команда задает шлюз по умолчанию для интерфейса `eth0`:

```
# route add default gw 192.168.181.2 eth0
```

Неприятно, но после перезагрузки системы добавленная нами запись исчезнет из таблицы маршрутизации. Можно добавить такую команду в сценарии инициализации системы, но такой подход не считается корректным. Более корректный способ установки шлюза по умолчанию в Fedora, Red Hat и других совместимых с ними дистрибутивах (CentOS, RHEL) заключается в корректировке файла `/etc/sysconfig/network` с указанием в переменной `GATEWAY` IP-адреса шлюза по умолчанию (листинг 31.1).

Листинг 31.1. Файл `/etc/sysconfig/network`: основные сетевые параметры в Fedora

```
NETWORKING=yes
FORWARD_IPV4=yes
HOSTNAME=den.dkws.org.ua
GATEWAY=0.0.0.0
```

ПРИМЕЧАНИЕ

С некоторыми конфигурационными файлами, приведенными в этой главе, вы уже знакомы, но здесь они рассматриваются в разрезе маршрутизации.

Параметр `NETWORKING` здесь определяет, будет ли включена поддержка сети (`yes` — поддержка сети включена, `no` — выключена). Параметр `FORWARD_IPV4` определяет, будет ли включено перенаправление пакетов. На компьютере, являющемся шлюзом, этот параметр должен быть включен (значение `yes`), на остальных компьютерах сети — выключен (значение `no`).

Параметр `HOSTNAME` задает имя узла, ну а `GATEWAY` — шлюз по умолчанию. Если компьютер является шлюзом, то обычно для этого параметра устанавливается IP-адрес `0.0.0.0`.

В openSUSE для задания шлюза по умолчанию нужно отредактировать файл `/etc/route.conf` или `/etc/sysconfig/network/routes` (в более современных версиях openSUSE), добавив в него строку вида:

```
default адрес [маска] [интерфейс]
```

Например:

```
default 192.168.181.2
```

Маску и интерфейс указывать необязательно. В этом же файле можно указать все остальные маршруты, т. е., по сути, этот файл хранит таблицу маршрутизации.

Маршрут по умолчанию, как правило, указывается последним. Пример файла конфигурации `/etc/sysconfig/network/routes` (`/etc/route.conf`) приведен в листинге 31.2.

Листинг 31.2. Файл `/etc/route.conf`

```
#
# /etc/sysconfig/network/routes (/etc/route.conf)
#
# Этот файл содержит описание статических маршрутов
#
# Назначение Шлюз          Маска          Устройство
#
192.168.0.0    0.0.0.0      255.255.255.128  eth0
default       192.168.0.1
```

Кроме файла `route.conf` в дистрибутивах SUSE вы можете также редактировать файл `/etc/rc.config`, содержащий всю информацию об имеющихся сетевых интерфейсах. Здесь важно отметить, что речь идет о старых версиях SUSE (конфигурационные файлы современных версий openSUSE мы рассматривали в *главе 26*).

В Debian и Ubuntu нужно редактировать файл `/etc/network/interfaces` — шлюз по умолчанию задается в нем параметром `gateway` (листинг 31.3). Синтаксис этого файла подробно описан в моей статье по адресу:

<http://dkws.org.ua/index.php?page=show&file=a/ubuntu/network-interfaces>,

поэтому здесь я позволю себе лишь несколько комментариев. Как можно видеть, в файле `/etc/network/interfaces` производится конфигурация интерфейса `eth0`, IP-адрес задается статически (`static`), присваивается IP-адрес `192.168.1.11`, маска `255.255.255.0`. Шлюз по умолчанию — это компьютер с IP-адресом `192.168.1.1`.

Листинг 31.3. Файл `/etc/network/interfaces`

```
iface eth0 inet static
address 192.168.1.11
netmask 255.255.255.0
gateway 192.168.1.1
```

31.2. Изменение таблицы маршрутизации.

Команда *route*

Нам уже знакома команда `route`, но использовали ее мы для просмотра таблицы маршрутизации. А сейчас мы научимся с ее помощью изменять эту таблицу.

Маршрутизация осуществляется на сетевом уровне модели OSI. Когда маршрутизатор получает пакет, предназначенный для другого узла, IP-адрес получателя пакета сравнивается с записями в таблице маршрутизации. Если есть хотя бы частич-

ное совпадение с каким-то маршрутом из таблицы, пакет отправляется по IP-адресу шлюза, связанного с этим маршрутом.

Если совпадений не найдено (т. е. вообще нет маршрута, по которому можно было бы отправить пакет), тогда пакет отправляется на шлюз по умолчанию, если таковой задан в таблице маршрутизации. Как уже отмечалось ранее, если шлюза по умолчанию нет, отправителю пакета посылается ICMP-сообщение "сеть недоступна" (network unreachable).

Команда `route` за один вызов может добавить или удалить только один маршрут. Другими словами, вы не можете сразу добавить или удалить несколько маршрутов. Формат вызова `route` следующий:

```
# route [операция] [тип] адресат gw шлюз [метрика] [dev интерфейс]
```

ИСПОЛЬЗУЙТЕ КОМАНДЫ `SUDO` ИЛИ `SU`

Команды добавления/удаления маршрута требуется вводить от имени пользователя `root`. Однако в современных системах входить под этим именем необязательно — для получения `root`-доступа можно использовать команды `sudo` или `su`.

- Параметр операция может принимать два значения: `add` (добавить маршрут) и `del` (удалить маршрут).
- Параметр тип необязательный — он задает тип маршрута: `-net` (маршрут к сети), `-host` (маршрут к узлу) или `default` (маршрут по умолчанию).
- Параметр адресат содержит адрес сети (если задается маршрут к сети), адрес узла (при добавлении маршрута к сети) или вообще не указывается (если задается маршрут по умолчанию).
- Параметр шлюз задает IP-адрес (или доменное имя) шлюза.
- Последние два параметра: метрика и `dev` необязательны:
 - параметр метрика задает максимальное число переходов (через маршрутизаторы) на пути к адресату (в Linux, в отличие от других ОС, этот параметр необязательный);
 - параметр `dev` имеет смысл указывать, если в системе установлено несколько сетевых интерфейсов и требуется указать, через какой именно сетевой интерфейс нужно отправить пакеты по заданному маршруту.

Команда удаления маршрута выглядит так:

```
# route del адрес
```

В других UNIX-системах имеется параметр `-f`, удаляющий все маршруты (`route -f`), но в Linux такого параметра нет. Следовательно, для очистки всей таблицы маршрутизации вам придется ввести серию команд `route del`.

Изменять таблицу маршрутизации нужно только зарегистрировавшись на компьютере локально. При удаленной регистрации (например, по `ssh`) легко удалить ошибочно маршрут, по которому вы вошли в систему. О последствиях такого действия, думаю, можно не говорить.

Примеры использования команды `route`:

```
route add -net 192.76.16.0 netmask 255.255.255.0 dev eth0
```

Добавляет маршрут к сети 192.76.16.0 (сеть класса C, о чем свидетельствует сетевая маска, заданная параметром `netmask`) через устройство `eth0`. Шлюз не указан, просто все пакеты, адресованные сети 192.76.16.0, будут отправлены на интерфейс `eth0`.

```
route add -net 192.16.16.0 netmask 255.255.255.0 gw 192.76.16.1
```

Добавляет маршрут к сети 192.16.16.0 через маршрутизатор 192.76.16.1. Сетевой интерфейс указывать не обязательно, но можно и указать при особом желании.

```
route add default gw gatel
```

Добавляет маршрут по умолчанию. Все пакеты будут отправлены компьютеру с именем `gatel`. Обратите внимание: мы указываем доменное имя узла вместо IP-адреса.

```
route add -net 10.1.0.0 netmask 255.0.0.0 reject
```

Добавляет запрещающий маршрут. Отправка пакетов по этому маршруту (в сеть 10.1.0.0) запрещена.

Итак, мы добавили необходимые маршруты, пропинговали удаленные узлы — все работает. Теперь нужно сохранить установленные маршруты, чтобы они были доступны при следующей загрузке системы. Для этого в openSUSE нужно отредактировать файл `/etc/sysconfig/network/routes` (`/etc/route.conf` — в старых версиях). Мы уже рассматривали этот файл (см. главу 26 и листинг 31.2), поэтому переходим сразу к другим дистрибутивам.

В старых версиях Fedora/CentOS (и других Red Hat-совместимых дистрибутивах) статические маршруты хранятся в файле `/etc/sysconfig/static-routes`. Строки в этом файле имеют вид:

```
any net адрес_сети netmask маска gw адрес_шлюза
```

Здесь `any` означает любой интерфейс. Можно указать конкретный интерфейс, например:

```
eth0 net 192.168.2.0 netmask 255.255.255.0 gw 192.168.1.1
```

Файл `/etc/sysconfig/static-routes` по умолчанию отсутствует, при необходимости его нужно создать самостоятельно.

В новых версиях Fedora статические маршруты описываются в файлах из каталога `/etc/sysconfig/network-scripts/`. Типичное имя файла статического маршрута выглядит так: `/etc/sysconfig/network-scripts/route-ifname`, где `ifname` — имя интерфейса. Например, `/etc/sysconfig/network-scripts/route-eth0`.

Маршрут по умолчанию задается так:

```
default via 192.168.1.1 dev interface
```

Если у вас несколько сетевых интерфейсов, и все они используют один и тот же маршрут по умолчанию, тогда целесообразно его определить в файле `/etc/sysconfig/network`.

Если нужно определить маршрут к удаленной сети, то делается это с помощью строк формата:

```
адрес_сети/маска via IP-адрес шлюза [dev имя_интерфейса]
```

Например:

```
10.10.10.0/24 via 192.168.1.1 dev eth0
```

Здесь ясно, что пакеты к сети 10.10.10.0 пойдут через шлюз 192.168.1.1 по интерфейсу eth0.

Дополнительную информацию о настройке статических маршрутов в Fedora 22 можно найти в документации по адресу:

https://docs.fedoraproject.org/en-US/Fedora/22/html/Networkmg_Guide/sec-Configuring_Static_Routes_in_ifcfg_files.html

Рекомендую ее внимательно изучить — вы найдете в ней много интересного и не только относительно настройки маршрутизации.

В Debian/Ubuntu статические маршруты прописываются вместе с конфигурацией сетевого интерфейса в файле `/etc/network/interfaces`. С помощью параметров `up` и `down` этого файла можно задать команды, которые будут выполняться при «поднятии» (`up`) и «закрытии» (`down`) интерфейса. После параметров `up` и `down` может следовать любая Linux-команда. Обычно это команда `route`. Например, при запуске интерфейса eth0 будет добавлен статический маршрут к сети 192.168.3.0 через шлюз 192.168.1.2:

```
up route add -net 192.168.3.0 netmask 255.255.255.0 gw 192.168.1.2
```

Можно также добавить маршрут по умолчанию:

```
up route add default gw 192.168.1.2
```

При «закрытии» интерфейса нужно удалить маршруты, которые использовали этот интерфейс, для этого служит параметр `down`:

```
down route del default gw 192.168.1.2
```

```
down route del -net 192.168.3.0
```

Подробное описание файла `/etc/network/interfaces` вы найдете по адресу <http://www.dkws.org.ua/index.php?page=show&file=a/ubuntu/network-interfaces>.

31.3. Включение 1Рy4-переадресации, или превращение компьютера в шлюз

Основное предназначение шлюза (маршрутизатора) — это пересылка (forwarding) пакетов. Чтобы включить пересылку пакетов протокола IPv4 (IPv4 forwarding), нужно записать значение 1 в файл `/proc/sys/net/ipv4/ip_forward`:

```
# echo 1 > /proc/sys/net/ipv4/ip_forward
```


Но включить пересылку мало — нужно еще сохранить это значение, иначе при перезагрузке будет восстановлено значение по умолчанию (0). Для этого следует в файл `/etc/sysctl.conf` добавить строку:

```
net.ipv4.ip_forward=0
```

В некоторых дистрибутивах — например, в openSUSE, можно воспользоваться конфигуратором, вызываемым командой **YaST | Сетевые настройки | Маршрутизация**. В иных — переадресацию можно включить путем редактирования конфигурационных файлов сети — например, в Fedora (см. листинг 31.1).

ПСЕВДОФАЙЛОВАЯ СИСТЕМА PROCS

Подробно о настройке системы с помощью псевдофайловой системы `procs` рассказано в главе 24.

31.4. Настройка брандмауэра

Брандмауэр (он же `firewall`, бастион/межсетевой экран) предназначен для защиты внутренней сети (или всего одного компьютера, напрямую подключенного к Интернету) от вторжения извне. С помощью брандмауэра вы можете контролировать доступ пользователей Интернета к узлам вашей внутренней сети, а также контролировать доступ локальных пользователей к ресурсам Интернета — например, вы можете запретить им посещать определенные узлы с целью экономии трафика.

Прежде чем перейти к настройке межсетевого экрана, определимся с терминологией и, в частности, с понятием *шлюз*. Шлюзом называется компьютер, предоставляющий компьютерам локальной сети доступ к Интернету. Шлюз выполняет как бы маршрутизацию пакетов. Но не нужно путать шлюз с обычным маршрутизатором. Маршрутизатор осуществляет простую пересылку пакетов, поэтому его можно использовать для соединения сетей одного типа, — например, локальной и локальной, глобальной и глобальной. А шлюз служит для соединения сетей разных типов — например, локальной и глобальной, как в нашем случае. Конечно, сейчас можно встретить маршрутизаторы с функцией шлюза, но это уже, скорее, аппаратные шлюзы, чем простые маршрутизаторы. Тем не менее, часто термины «маршрутизатор» и «шлюз» употребляются как синонимы, хотя это не совсем так.

Сложность в соединении сетей разных типов заключается в различной адресации. Как мы знаем, в локальной сети обычно используются локальные адреса, которые не допустимы в Интернете, например: `192.168.*.*` (сеть класса C), `10.*.*.*` (сеть класса A) и `172.16.*.*-172.31.*.*` (класс B). Поэтому шлюз должен выполнить преобразование сетевого адреса (NAT, Network Address Translation). Суть такого преобразования в следующем. Предположим, у нас есть шлюз и локальная сеть с адресами `192.168.*.*`. Реальный IP-адрес (который можно использовать в Интернете) есть только у шлюза, пусть это `193.254.219.1`. У всех остальных компьютеров — локальные адреса, поэтому при всем своем желании они не могут обратиться к интернет-узлам.

У нашего шлюза два сетевых интерфейса. Один из них, пусть `ppp0`, используется для подключения к Интернету. Его IP-адрес, как уже было отмечено, `93.254.219.1`.

Для подключения к локальной сети используется другой сетевой интерфейс — eth0 (сетевая плата) с IP-адресом 192.168.1.1.

Все узлы нашей локальной сети используют в качестве шлюза компьютер с адресом 192.168.1.1.. Это означает, что все запросы будут переданы на узел 192.168.1.1. Запросы передаются в виде:

Назначение: IP-адрес узла Интернета

Источник: адрес компьютера локальной сети, пусть 192.168.1.10

Наш шлюз принимает запрос и перезаписывает его так:

Назначение: IP-адрес узла Интернета

Источник: 193.254.219.1

То есть, шлюз подменяет адрес источника, устанавливая в качестве этого адреса свой реальный IP-адрес, иначе бы любой интернет-узел не принял бы запрос с локального адреса. Получив ответ от узла, он направляет его нашему узлу:

Назначение: 192.168.1.10

Источник: IP-адрес узла Интернета

Нашему локальному узлу «кажется», что он получил ответ непосредственно от узла Интернета, а на самом деле ответ приходит от шлюза.

Теперь, когда мы разобрались с теорией, самое время перейти к практике.

31.4.1. Цепочки и правила

Основная задача брандмауэра — это фильтрация пакетов, которые проходят через сетевой интерфейс. При поступлении пакета брандмауэр анализирует его и затем принимает решение: принять пакет (ACCEPT) или избавиться от него (DROP). Брандмауэр может выполнять и более сложные действия, но часто ограничиваются именно этими двумя.

Прежде чем брандмауэр примет решение относительно пакета, пакет должен пройти по цепочке правил. Каждое правило состоит из условия и действия (цели). Если пакет соответствует условию правила, то выполняется указанное в правиле действие. Если пакет не соответствует условию правила, он передается следующему правилу. Если же пакет не соответствует ни одному из правил цепочки, выполняется действие по умолчанию.

Вроде бы все понятно, но чтобы лучше закрепить знания, рассмотрим табл. 31.2, демонстрирующую принцип работы цепочки правил.

Таблица 31.2. Цепочка правил

Номер правила	Условие	Действие (цель)
1	Пакет от 192.168.1.0	ACCEPT
2	Пакет от 192.168.0.0	DROP
3	Пакет для 192.168.2.0	ACCEPT
DEFAULT	*	DROP

Предположим, что пакет пришел из сети 192.168.4.0 для узла 192.168.1.7 (это наша сеть). Пакет не соответствует первому правилу (отправитель не из сети 192.168.1.0), поэтому он передается правилу 2. Пакет не соответствует и этому правилу. Пакет адресован компьютеру 192.168.1.7, а не компьютеру из сети 192.168.2.0, поэтому он не соответствует третьему правилу. Брандмауэру остается применить правило по умолчанию — пакет будет отброшен (действие `drop`).

Цепочки правил собираются в три основные таблицы:

- `filter` — таблица фильтрации, основная таблица;
- `nat` — таблица NAT, используется при создании пакетом нового соединения;
- `mangle` — используется, когда нужно произвести специальные действия над пакетом.

БРАНДМАУЭРЫ: СТАРЫЙ И НОВЫЙ

Ранее брандмауэр в Linux поддерживал только цепочки правил и назывался `ipchains`, сейчас брандмауэр поддерживает и цепочки правил, и таблицы цепочек и называется `iptables`. Это примечание сделано, чтобы вы понимали разницу между старым брандмауэром `ipchains` (ядра 2.2 и ниже) и новым `iptables` (ядра 2.4 и выше).

Если необходимо, вы можете создать собственные таблицы. В состав таблицы входят три цепочки:

- `INPUT` — для входящих пакетов;
- `OUTPUT` — для исходящих пакетов;
- `FORWARD` — для пересылаемых (транзитных) пакетов.

Над пакетом можно выполнить следующие действия:

- `<имя цепочки>` — пакет будет отправлен для обработки в цепочку с указанным именем;
- `ACCEPT` — принять пакет;
- `DROP` — отбросить пакет, после этого пакет удаляется, больше над ним не выполняются какие-либо действия;
- `MASQUERADE` — скрыть IP-адрес пакета.

Это не все действия, но пока нам больше знать ни к чему. На рис. 31.2 изображена схема обработки пакета. Входящий пакет (на схеме `ПАКЕТ IN`) поступает в цепочку `PREROUTING` таблицы `mangle`. После чего (если он не был отброшен правилами таблицы `mangle`) пакет обрабатывается правилами цепочки `PREROUTING`, но таблицы `nat`. На этом этапе проверяется, нужно ли модифицировать назначение пакета (этот вид NAT называется Destination NAT, DNAT).

Затем пакет может быть направлен либо в цепочку `INPUT` (если получателем пакета является этот компьютер), либо в цепочку `FORWARD` (если пакет нужно передать другому компьютеру).

Если получатель компьютера — сам шлюз (на нем может быть запущен, например, почтовый или Web-сервер), то пакет сначала обрабатывается правилами цепочки

INPUT таблиц `mangle` и `filter`. Если пакет не был отброшен, он передается приложению (например, почтовому серверу). Приложение получило пакет, обработало его и отправляет ответный пакет. Этот пакет обрабатывается цепочкой OUTPUT таблиц `mangle`, `nat` и `filter`. Далее пакет отправляется на цепочку POSTROUTING и обрабатывается правилами таблиц `mangle` и `nat`.

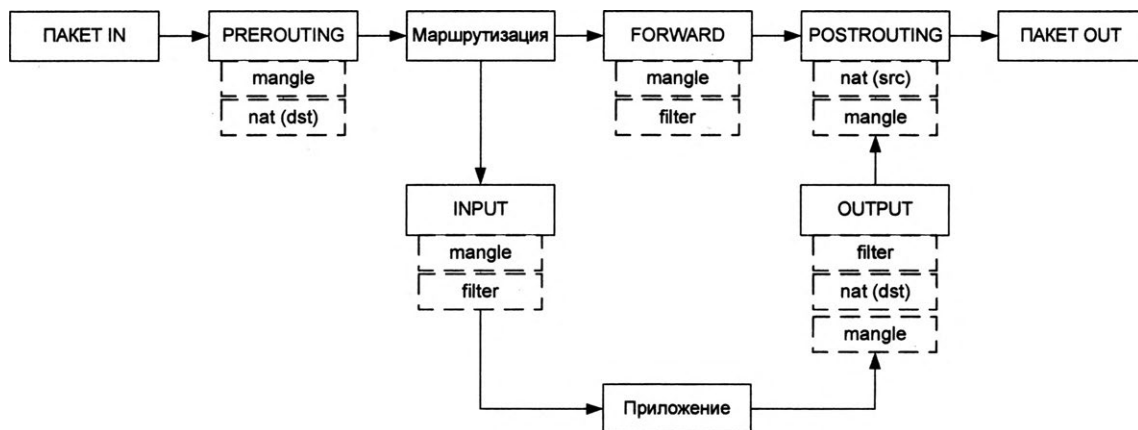


Рис. 31.2. Схема обработки пакета

Если пакет нужно передать другому компьютеру, то он обрабатывается правилами цепочки `forward` таблиц `mangle` и `filter`, а после этого к нему применяются правила цепочки `POSTROUTING`. На этом этапе используется подмена источника пакета (этот вид NAT называется Source NAT, SNAT).

После всех правил пакет «выжил»? Тогда он становится исходящим пакетом (на схеме пакет `оит`) и отправляется в сеть.

НЕСКОЛЬКО СЛОВ О БРАНДМАУЭРЕ NFTABLES

Прежде, чем мы перейдем к изучению брандмауэра `iptables`, нужно сказать пару слов о брандмауэре `nftables`, — он призван стать заменой для `iptables`. Поддержка `nftables` была добавлена в ядро еще в версии 3.13 в 2014 году. Вот только незадача — разработчики дистрибутивов не спешат на него переходить. Запускаю Ubuntu 15.04 — в ней старый, добрый `iptables`, а когда вводишь команду `nft` (основная команда `nftables`) система предлагает сначала установить пакет `nftables`. Полагаю, это из-за того, что ядро у Ubuntu пока старое, — 3.19 (этому я, честно, удивился, поскольку обычно в Ubuntu все самое новое, — видимо, начали заботиться о стабильности). Что ж, запускаю Fedora Server 22, в котором используется ядро версии 4.0.4, — но и там тоже `iptables`... Если вы надумаете-таки использовать брандмауэре `nftables`, тогда выбирайте дистрибутив с самой новой версией ядра, — чем выше, тем больше ошибок в `nftables` будет исправлено. Впрочем, пока я бы не рекомендовал переходить на `nftables`, но вполне возможно, что его описание появится в следующем издании этой книги.

31.4.2. Брандмауэр iptables

Теперь, когда мы разобрались с правилами и цепочками, самое время научиться использовать брандмауэр `iptables`. Для себя сразу определитесь, что вы настраиваете: можно настраивать просто брандмауэр, защищающий локальный компьютер от

всевозможных атак, а можно настраивать шлюз сети, предоставляющий всем остальным компьютерам сети доступ к Интернету. В последнем случае нужно включить IP-переадресацию (IPv4-forwarding). О том, как это сделать, было сказано ранее. В большинстве случаев хватит вот такой команды:

```
sudo sysctl -w net.ipv4.ip_forward="1"
sudo echo 1 > /proc/sys/net/ipv4/ip_forward
```

Для изменения правил брандмауэра нужны полномочия root, поэтому все команды iptables следует вводить или через команду sudo (для этого ваш пользователь должен иметь право использовать sudo), или с предварительно полученными полномочиями root (команда su).

Для добавления правила в цепочку служит команда:

```
sudo iptables -A цепочка правило
```

Например:

```
sudo iptables -A INPUT правило
```

Эта команда добавит правило в цепочку INPUT таблицы filter (это таблица по умолчанию). Если вы желаете добавить правило в другую таблицу, нужно указать ее в параметре -t:

```
sudo iptables -t таблица -A цепочка правило
```

Например, для таблицы nat:

```
sudo iptables -t nat -A INPUT правило
```

Действие по умолчанию задается ключом -P:

```
sudo iptables -P INPUT DROP
```

Обычно по умолчанию устанавливаются вот такие действия:

```
sudo iptables -P INPUT DROP
sudo iptables -P FORWARD ACCEPT
sudo iptables -P OUTPUT DROP
```

Параметры фильтрации пакетов сведены в табл. 31.3, но прежде, чем обратиться к ней, рассмотрим этапы (фазы) установки TCP-соединения. Соединение устанавливается в три этапа:

- сначала первый компьютер отправляет второму компьютеру SYN-пакет, запрашивая открытие соединения;
- второй компьютер отправляет ему подтверждение SYN-пакета — ACK-пакет;
- после этого соединение считается установленным (ESTABLISHED).

Открытое, но не установленное соединение (когда компьютеры обмениваются пакетами SYN-ACK) называется новым (NEW).

Разобраться с материалом табл. 31.3, где при описании параметров указываются не полные команды iptables, а только их фрагменты, имеющие отношения к тому или иному параметру, помогут пояснения, проведенные в скобках.

Таблица 31.3. Параметры фильтрации пакетов

Параметр	Описание
--source	Позволяет указать источник пакета. Можно указывать как доменное имя компьютера (den.dkws.org.ua), так и его IP-адрес (192.156.1.1) и даже набор адресов (192.168.1.0/255.255.255.0). Пример: iptables -A FORWARD -source 192.168.1.11 ...
--destination	Задаёт назначение (адрес получателя) пакета. Синтаксис такой же, как и у --source
-protocol (или -p)	Задаёт протокол. Чаще всего работают с tcp, icmp или udp, но можно указать любой протокол, определенный в файле /etc/protocols. Также можно указать all, что означает все протоколы. Примеры: iptables -A FORWARD -protocol tcp ... iptables -A FORWARD -p tcp ...
--source-port (или --sport)	Определяет порт отправителя. Эта опция может использоваться только вместе с параметром -p. Например: iptables -A FORWARD -p tcp -source-port 23 ...
--destination-port (или --dport)	Задаёт порт-назначение. Опция возможна только с параметром -p. Синтаксис такой же, как и в случае с -source-port
-state	Позволяет отфильтровать пакеты по состоянию. Параметр -state доступен только при загрузке модуля state с помощью другого параметра -m state. Состояния пакета: <ul style="list-style-type: none"> NEW — новое соединение (еще не установленное); ESTABLISHED — установленное соединение; RELATED — пакеты, которые не принадлежат соединению, но связаны с ним; INVALID — неопознанные пакеты. Пример: iptables -A FORWARD -m state -state RELATED,INVALID
-in-interface (или -i)	Определяет интерфейс, по которому прибыл пакет. Пример: iptables -A FORWARD -i eth1
-out-interface (или -o)	Определяет интерфейс, по которому будет отправлен пакет. Пример: iptables -A FORWARD -o ppp0
-tcp-flags	Производит фильтрацию по TCP-флагам (man iptables)

Ранее мы познакомились с основными действиями iptables (см. табл. 31.2). В табл. 31.4 представлены все действия iptables (цели iptables). Действие задается параметром -j.

Таблица 31.4. Цели iptables

Действие	Описание
ACCEPT	Принять пакет. При этом пакет уходит из этой цепочки и передается дальше
DROP	Уничтожить пакет

Таблица 31.4 (окончание)

Действие	Описание
REJECT	<p>Уничтожает пакет и сообщает об этом отправителю с помощью ICMP-сообщения. Параметр <code>-reject-with</code> позволяет уточнить тип ICMP-сообщения:</p> <ul style="list-style-type: none"> <code>icmp-host-unreachable</code> — узел недоступен; <code>icmp-net-unreachable</code> — сеть недоступна; <code>icmp-port-unreachable</code> — порт недоступен; <code>icmp-proto-unreachable</code> — протокол недоступен. <p>По умолчанию это действие отправляет сообщение о недоступности порта. При этом, используя сообщение <code>icmp-host-unreachable</code>, можно сбить с толку атакующего вас злоумышленника. Предположим, что вы просто решили отбрасывать неудобные вам пакеты (действие <code>DROP</code>). Но злоумышленник будет посылать и посылать вам эти пакеты, чтобы брандмауэр только и делал, что занимался бы фильтрацией и удалением этих пакетов (один из видов атаки на отказ). А если вы ответите сообщением <code>icmp-host-unreachable</code>, то злоумышленник решит, что узел недоступен, т. е. что компьютер выключен, либо он уже достиг своей цели — добился отказа компьютера. С другой стороны, помните, что это действие порождает ответный ICMP-пакет, что нагружает исходящий канал, который в некоторых случаях (например, одностороннего спутникового соединения) очень «узкий». Если злоумышленник пришлет вам 1 миллион пакетов, то вы должны будете отправить 1 миллион сообщений в ответ. Подумайте, готовы ли вы к такой нагрузке на исходящий канал</p>
LOG	Заносит информацию о пакете в протокол. Полезно использовать для протоколирования возможных атак — если вы подозреваете, что ваш узел атакуется кем-то. Также полезно при отладке настроек брандмауэра
RETURN	Возвращает пакет в цепочку, откуда он прибыл. Действие возможно, но лучше его не использовать, т. к. легко ошибиться и создать непрерывный цикл: вы отправляете пакет обратно, а он опять следует на правило, содержащее цель <code>RETURN</code>
SNAT	Выполняет подмену IP-адреса отправителя (Source NAT). Используется в цепочках <code>POSTROUTING</code> и <code>OUTPUT</code> таблицы <code>nat</code>
DNAT	Выполняет подмену адреса получателя (Destination NAT). Используется только в цепочке <code>POSTROUTING</code> таблицы <code>nat</code>
MASQUERADE	Пожо на <code>SNAT</code> , но «забывает» про все активные соединения при потере интерфейса. Используется при работе с динамическими IP-адресами, когда происходит «потеря» интерфейса при изменении IP-адреса. Применяется в цепочке <code>POSTROUTING</code> таблицы <code>nat</code>

31.4.3. Шлюз своими руками

Создать шлюз в Linux очень просто, и сейчас вы сами в этом убедитесь. Гораздо сложнее правильно его настроить, чтобы шлюз не только выполнял свою непосредственную функцию (т. е. передачу пакетов из локальной сети в Интернет и обратно), но и защищал сеть.

В последнее время очень популярны DSL-соединения, поэтому будем считать, что для подключения к Интернету используется именно DSL-соединение. Впрочем, вся

разница только в названии интерфейса — `ppp0`. Вполне может быть, что у вас иная конфигурация. Например, у вас может быть два сетевых интерфейса: `eth0` и `eth1`. Первый «смотрит» в локальную сеть, а второй — подключен к Интернету. Тогда и правила вы будете формировать исходя из того, что соединение с Интернетом происходит по интерфейсу `eth1`.

При DSL-соединении у нас тоже будет два сетевых адаптера. Первый (`eth0`) подключен к локальной сети, а к второму (`eth1`) подключен DSL-модем. Перед настройкой шлюза проверьте, действительно ли это так. Вполне может оказаться, что сетевая плата, к которой подключен DSL-модем, — это интерфейс `eth0`, а не `eth1`. Тогда вам придется или изменить названия интерфейсов при формировании правил, или просто подключить модем к другому сетевому адаптеру.

IP-адрес DSL-соединения будет динамическим (обычно так оно и есть), а вот сетевому адаптеру, обращенному к локальной сети, назначим IP-адрес `192.168.1.1`. Вы можете использовать и другой адрес (адрес должен быть локальным, если только у вас нет подсети с реальными IP-адресами).

Итак, мы настроили локальную сеть, узнали имена сетевых адаптеров, включили IP-переедресацию. Осталось только ввести команду:

```
sudo iptables -t nat -A POSTROUTING -o ppp0 -j MASQUERADE
```

Установите на всех компьютерах вашей сети IP-адрес `192.168.1.1` в качестве шлюза по умолчанию (можно настроить DHCP-сервер, чтобы не настраивать все компьютеры вручную) и попробуйте «пропинговать» с любого узла какой-нибудь сайт — оказывается, вы прочитали всю эту главу ради одной такой строчки. Так и есть. Но, сами понимаете, на этом настройка шлюза не заканчивается. Надо еще защитить вашу сеть. Как минимум, требуется установить следующие действия по умолчанию:

```
sudo iptables -P INPUT DROP
sudo iptables -P FORWARD ACCEPT
sudo iptables -P OUTPUT DROP
```

Разрешим входящие соединения на шлюз только от узлов нашей внутренней сети `192.168.1.0`:

```
sudo iptables -A INPUT -i eth0 -source 192.168.1.0/24 -match state -state NEW,ESTABLISHED -j ACCEPT
```

Надо также установить правило для цепочки `OUTPUT` — оно разрешает шлюзу отвечать компьютерам нашей локальной сети:

```
sudo iptables -A OUTPUT -o eth0 -destination 192.168.1.0/24 -match state -state NEW,ESTABLISHED -j ACCEPT
```

Будьте внимательны при указании имен интерфейсов и IP-адресов. Очень легко запутаться, а потом полчаса разбираться, почему шлюз не работает.

Нам осталось только запретить соединения из Интернета (компьютеры нашей сети смогут устанавливать соединения с серверами Интернета, зато интернет-пользователи не смогут установить соединения с компьютерами нашей сети):


```
sudo iptables -A FORWARD -i eth0 --destination 192.168.1.0/24 --match state --state ESTABLISHED -j ACCEPT
```

Итак, у нас получилась весьма простенькая конфигурация: компьютеры нашей сети могут выступать инициаторами соединения, а интернет-узлы могут передавать данные в нашу сеть только в том случае, если инициатором соединения выступил локальный компьютер.

Но это еще не все. Как вы уже догадались, поскольку мы не сохранили правила брандмауэра, при перезагрузке компьютера его придется настраивать заново. Поскольку мне нет резона описывать настройку брандмауэра (сохранение и восстановление правил) для каждого дистрибутива (пусть это будет вашим домашним заданием), рассмотрим универсальный способ. Он заключается в создании `bash`-сценария, вызывающего необходимые нам команды настройки `iptables`. После написания сценария вам останется вызвать его при загрузке системы — а для этого придется изучить строение системы инициализации в вашем дистрибутиве (см. главу 22).

Вместо того, чтобы объяснять вам, как вызвать сценарий, загружающий правила брандмауэра (с этим вы и сами разберетесь), я лучше приведу сценарий (понятно, с комментариями), реализующий более сложную конфигурацию `iptables`. Этот сценарий (листинг 31.4) будет не только выполнять все функции шлюза, но и защищать сеть от разного рода атак. Следует обеспечить автоматическую загрузку этого сценария, чтобы не запускать его каждый раз при старте системы.

Листинг 31.4. Сценарий `firewall_start`

```
# Путь к iptables
IPT="/sbin/iptables"

# Сетевой интерфейс, подключенный к Интернету
INET="ppp0"

# Номера непривилегированных портов
UPORTS="1024:65535"

# Включаем IPv4-forwarding (чтобы не думать, почему шлюз не работает)
echo 1 > /proc/sys/net/ipv4/ip_forward

# Удаляем все цепочки и правила
$IPT -F
$IPT -X

# Действия по умолчанию.
$IPT -P INPUT DROP
$IPT -P FORWARD ACCEPT
$IPT -P OUTPUT DROP
```

```
# Разрешаем все пакеты по интерфейсу lo (обратная петля)
$IPT -A INPUT -i lo -j ACCEPT
$IPT -A OUTPUT -o lo -j ACCEPT

# Запрещаем любые новые соединения с нашим компьютером
# с любых интерфейсов, кроме lo
$IPT -A INPUT -m state ! -i lo --state NEW -j DROP
$IPT -A INPUT -s 127.0.0.1/255.0.0.0 ! -i lo -j DROP

# Отбрасываем все пакеты со статусом INVALID
$IPT -A INPUT -m state --state INVALID -j DROP
$IPT -A FORWARD -m state --state INVALID -j DROP

# Принимаем все пакеты из уже установленного соединения
# Состояние ESTABLISHED
$IPT -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
$IPT -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT

# Мой провайдер использует IP-адреса из сети 10.0.0.0 для
# доступа к своим локальным ресурсам. Ничего не поделаешь,
# нужно разрешить эти адреса, иначе мы даже не сможем войти в
# билинговую систему. В вашем случае, может и не нужно будет
# добавлять следующее правило, а, может, у вас будет такая же
# ситуация, но адрес подсети будет другим
$IPT -t nat -I PREROUTING -i $INET -s 10.0.0.1/32 -j ACCEPT

# Защищаемся от SYN-наводнения (довольно популярный вид атаки)
$IPT -A INPUT -p tcp ! --syn -m state --state NEW -j DROP
$IPT -A OUTPUT -p tcp ! --syn -m state --state NEW -j DROP

# Защищаемся от UDP-наводнения
$IPT -A INPUT -p UDP -s 0/0 --dport 138 -j DROP
$IPT -A INPUT -p UDP -s 0/0 --dport 113 -j REJECT
$IPT -A INPUT -p UDP -s 0/0 --sport 67 --dport 68 -j ACCEPT
$IPT -A INPUT -p UDP -j RETURN
$IPT -A OUTPUT -p UDP -s 0/0 -j ACCEPT

# Защищаемся от ICMP-перенаправления
# Этот вид атаки может использоваться злоумышленником для
# перенаправления своего трафика через вашу машину
$IPT -A INPUT --fragment -p ICMP -j DROP
$IPT -A OUTPUT --fragment -p ICMP -j DROP

# Но обычные ICMP-сообщения мы разрешаем
$IPT -A INPUT -p icmp -m icmp -i $INET --icmp-type source-quench -j ACCEPT
$IPT -A OUTPUT -p icmp -m icmp -o $INET --icmp-type source-quench -j ACCEPT
```

```
# Разрешаем себе "пинговать" интернет-узлы
$IPT -A INPUT -p icmp -m icmp -i $INET-icmp-type echo-reply -j ACCEPT
$IPT -A OUTPUT -p icmp -m icmp -o $INET-icmp-type echo-request -j ACCEPT

# Разрешаем передачу ICMP-сообщения "неверный параметр"
$IPT -A INPUT -p icmp -m icmp -i $INET-icmp-type parameter-problem -j ACCEPT
$IPT -A OUTPUT -p icmp -m icmp -o $INET-icmp-type parameter-problem -j ACCEPT

# Запрещаем подключение к X.Org через сетевые интерфейсы.
$IPT -A INPUT -p tcp -m tcp -i $INET -dport 6000:6063 -j DROP --syn

# Указываем порты, открытые в системе, но которые должны быть
# закрыты на сетевых интерфейсах. Я пропишу только порт 5501:
$IPT -A INPUT -p tcp -m tcp -m multiport -i $INET -j DROP --dports 5501

# Разрешаем DNS
$IPT -A OUTPUT -p udp -m udp -o $INET --dport 53 --sport $UPOINTS -j ACCEPT
$IPT -A OUTPUT -p tcp -m tcp -o $INET --dport 53 --sport $UPOINTS -j ACCEPT
$IPT -A INPUT -p .udp -m udp -i $INET --dport $UPOINTS --sport 53 -j ACCEPT
$IPT -A INPUT -p tcp -m tcp -i $INET --dport $UPOINTS --sport 53 -j ACCEPT

# Разрешаем AUTH-запросы к удаленным серверам, но запрещаем такие
# запросы к своему компьютеру
$IPT -A OUTPUT -p tcp -m tcp -o $INET --dport 113 --sport $UPOINTS -j ACCEPT
$IPT -A INPUT -p tcp -m tcp -i $INET --dport $UPOINTS --sport 113 -j ACCEPT ! --
syn
$IPT -A INPUT -p tcp -m tcp -i $INET -dport 113 -j DROP

# Далее мы открываем некоторые порты, необходимые для
# функционирования сетевых служб

# FTP-клиент (порт 21)
$IPT -A OUTPUT -p tcp -m tcp -o $INET --dport 21 --sport $UPOINTS -j ACCEPT
$IPT -A INPUT -p tcp -m tcp -i $INET --dport $UPOINTS --sport 21 -j ACCEPT ! --
syn

# SSH-клиент (порт 22)
$IPT -A OUTPUT -p tcp -m tcp -o $INET --dport 22 --sport $UPOINTS -j ACCEPT
$IPT -A INPUT -p tcp -m tcp -i $INET --dport $UPOINTS --sport 22 -j ACCEPT ! --
syn
$IPT -A OUTPUT -p tcp -m tcp -o $INET --dport 22 --sport 1020:1023 -j ACCEPT
$IPT -A INPUT -p tcp -m tcp -i $INET --dport 1020:1023 -sport 22 -j ACCEPT ! --
-syn

# SMTP-клиент (порт 25)
$IPT -A OUTPUT -p tcp -m tcp -o $INET --dport 25 --sport $UPOINTS -j ACCEPT
$IPT -A INPUT -p tcp -m tcp -i $INET --dport $UPOINTS --sport 25 -j ACCEPT ! --syn
```

```
# HTTP/HTTPS-клиент (порты 80, 443)
$IPT -A OUTPUT -p top -ш top -ш multiport -o $INET --sport $UPOINTS -j ACCEPT --
dports 80,443
$IPT -A INPUT -p top -м top -м multiport -i $INET --dport $UPOINTS -j ACCEPT --
sports 80,443 ! --syn

# POP-клиент (порт 110)
$IPT -A OUTPUT -p top -м top -o $INET --dport 110 --sport $UPOINTS -j ACCEPT
$IPT -A INPUT -p top -м top -i $INET --dport $UPOINTS --sport 110 -j ACCEPT ! --
syn

# Разрешаем прохождение DHCP-запросов через iptables
# Необходимо, если IP-адрес динамический
$IPT -A OUTPUT -p udp -м udp -o $INET --dport 67 --sport 68 -j ACCEPT
$IPT -A INPUT -p udp -м udp -i $INET --dport 68 --sport 67 -j ACCEPT
```

Вот, практически, и все... Конечно, приведенное здесь описание iptables нельзя назвать полным. Но для полного описания iptables пришлось бы создать отдельную книгу под названием «Брандмауэр в Linux».

ПОЛНОЕ РУКОВОДСТВО ПО IPTABLES

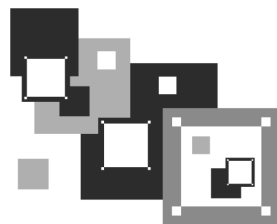
В Интернете я нашел одно из наиболее полных руководств по iptables на русском языке. Так вот, если его распечатать, оно займет 121 страницу формата А4. Учитывая размер полосы набора страницы книжного формата, которая обычно меньше А4, смело можно говорить, что объем такой книги составил бы около 200 страниц.

Адрес указанного руководства: <http://www.opennet.ru/docs/RUS/iptables/>.

Вот еще одна очень хорошая статья по iptables: <http://ru.wikipedia.org/wiki/Iptables>.

А для пользователей Debian и Ubuntu будет полезным следующее руководство: http://www.linux.by/wiki/index.php/Debian_Firewall.

ГЛАВА 32



Безопасный удаленный доступ. OpenSSH

32.1. Протокол SSH

Для организации удаленного доступа к консоли сервера ранее использовался протокол telnet, и в каждую сетевую операционную систему, будь то FreeBSD или Windows 95 (которую, впрочем, сложно напрямую назвать сетевой), включался telnet-клиент. Эта программа так и называется — telnet (в Windows — `telnet.exe`).

Подключившись с помощью telnet к удаленному компьютеру, вы можете работать с ним как обычно. В окне telnet-клиента представлена как бы консоль удаленного компьютера: вы будете вводить команды и получать результат их выполнения — все так, как если бы вы работали непосредственно за удаленным компьютером.

Но технологии не стоят на месте, и протокол telnet устарел. Сейчас им практически никто не пользуется. На смену ему пришел протокол SSH (Secure Shell), который, как видно из названия, представляет собой безопасную оболочку. Главное отличие SSH от telnet состоит в том, что в соответствии с этим протоколом все данные (включая пароли доступа к удаленному компьютеру, отдельные файлы и пр.) передаются в зашифрованном виде. Причиной создания SSH стало то, что во времена telnet участились случаи перехвата паролей и другой важной информации.

SSH использует для шифрования передаваемых данных следующие алгоритмы: Blowfish, 3DES (Data Encryption Standard), IDEA (International Data Encryption Algorithm) и RSA (Rivest-Shamir-Adelman algorithm). Самыми надежными из них считаются IDEA и RSA. Поэтому, если вы передаете действительно конфиденциальные данные, лучше использовать один из этих алгоритмов.

В состав любого дистрибутива Linux входит SSH-сервер (программа, которая и обеспечивает удаленный доступ к компьютеру, на котором она установлена) и SSH-клиент (программа, позволяющая подключаться к SSH-серверу). Для установки SSH-сервера нужно установить пакет `openssh` (это разновидность SSH-сервера), а для установки SSH-клиента — пакет `openssh-clients`.

Если у вас на рабочей станции установлена система Windows, и вам нужно подключиться к SSH-серверу, запущенному на Linux-машине, то по адресу <http://www.cs.hut.fi/ssh/> вы можете скачать Windows-клиент для SSH. Нужно отметить, что Windows-клиент, в отличие от Linux-клиента, не бесплатен.

32.2. Использование SSH-клиента

Работать с SSH-клиентом очень просто. Для подключения к удаленному компьютеру введите команду:

```
ssh [опции] <адрес_удаленного_компьютера>
```

В качестве адреса можно указать как IP-адрес, так и доменное имя компьютера. В табл. 32.1 приведены часто используемые опции команды ssh.

Таблица 32.1. Опции команды ssh

Опция	Описание
-c blowfish 3des des	Используется для выбора алгоритма шифрования при условии, что применяется первая версия протокола SSH (об этом позже). Можно указать blowfish, des или 3des
-c шифр	Задаёт список шифров, разделённых запятыми в порядке предпочтения. Опция используется для второй версии SSH. Можно указать blowfish, twofish, arcfour, cast, des и 3des
-f	Переводит SSH в фоновый режим после аутентификации пользователя. Рекомендуется использовать для запуска программы X11. Например: ssh -f server xterm
-l имя_пользователя	Указывает имя пользователя, от имени которого нужно зарегистрироваться на удалённом компьютере. Опцию использовать не обязательно, поскольку удалённый компьютер и так запросит имя пользователя и пароль
-p порт	Определяет порт SSH-сервера (по умолчанию используется порт 22)
-q	«Тихий режим» — будут отображаться только сообщения о фатальных ошибках. Все прочие предупреждающие сообщения в стандартный выходной поток выводиться не будут
-x	Отключает перенаправление X11
-X	Включает перенаправление X11. Полезно при запуске X11 -программ
-1	Использовать только первую версию протокола SSH
-2	Использовать только вторую версию протокола SSH. Вторая версия протокола более безопасна, поэтому при настройке SSH-сервера нужно использовать именно её

32.3. Настройка SSH-сервера

Если вы используете OpenSSH (а в большинстве случаев так оно и есть), все настройки SSH-сервера хранятся в одном-единственном файле: /etc/ssh/sshd_config (в старых версиях: /etc/sshd_config), а настройки программы-клиента — в файле

/etc/ssh/ssh_config (в старых версиях: /etc/ssh_config). Настройки программы клиента обычно задавать не нужно, поскольку они приемлемы по умолчанию. На всякий случай вы можете заглянуть в файл /etc/ssh_config — его формат, как и назначение опций (большая часть из них закомментирована), вы поймете и без моих подсказок.

Сейчас нас больше интересует файл sshd_config, содержащий конфигурацию SSH-сервера. Рассмотрим пример такой конфигурации (листинг 32.1). Чтобы понять назначение директив, внимательно читайте комментарии, приведенные в листинге.

Листинг 32.1. Пример файла конфигурации /etc/ssh/sshd_config

```
#      $OpenBSD: sshd_config,v 1.72 2005/07/25 11:59:40 markus Exp $

# Задаёт порт, на котором будет работать SSH-сервер. Если директива
# не указана (закомментирована), то по умолчанию используется порт 22
# Port 22

# Директива Protocol позволяет выбрать версию протокола,
# рекомендуется использовать вторую версию
# Protocol 2,1
Protocol 2

# Директива AddressFamily задаёт семейство интерфейсов, которые должен
# прослушивать SSH-сервер
# AddressFamily any

# Локальный адрес, который должен прослушиваться SSH-сервером
# ListenAddress 0.0.0.0

# Ключевой файл для протокола SSH версии 1
# HostKey for protocol version 1
HostKey /etc/ssh/ssh_host_key
# Ключевые файлы для второй версии протокола SSH
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_dsa_key

# Время жизни ключа протокола первой версии. Время можно задавать
# в секундах или в часах (постфикс h, например, 1h — это 1 час или
# 3600 секунд). По истечении указанного времени ключевой файл будет
# сгенерирован заново
# KeyRegenerationInterval 1h

# Разрядность ключа сервера в битах (только для первой версии протокола
# SSH)
# ServerKeyBits 768

# Директивы управления протоколированием (можно не изменять)
# SyslogFacility AUTH
# LogLevel INFO
```

```
# Директивы аутентификации

# Время, предоставляемое клиенту для аутентификации. Задается в секундах
# или минутах (1m = 60 секунд). Если за это время клиент не
# аутентифицировал себя, соединение будет прекращено
# LoginGraceTime 2m
# Директива разрешает (yes) удаленный доступ пользователя root
PermitRootLogin yes

# Максимальное количество попыток аутентификации
# MaxAuthTries 6

# Использование RSA (yes)
# RSAAuthentication yes
# Аутентификация с открытым ключом (при значении yes)
# PubkeyAuthentication yes
# AuthorizedKeysFile .ssh/authorized_keys

# Использование .rhosts-аутентификации с поддержкой RSA.
# Rhosts-аутентификацию использовать не рекомендуется, поэтому по
# умолчанию для этой директивы указано значение no. Если вы все-таки
# установите значение yes для этой директивы, то не
# забудьте указать в файле /etc/ssh/ssh_known_hosts IP-адреса
# компьютеров, которым разрешен доступ к SSH-серверу. Только для первой
# версии протокола
# RhostsRSAAuthentication no

# Если вы используете вторую версию протокола и хотите разрешить Rhosts-
# аутентификацию, то вам нужно включить директиву HostbasedAuthentication,
# а разрешенные узлы указываются в файле ~/.ssh/known_hosts
# HostbasedAuthentication no

# Если вы не доверяете пользовательским файлам ~/.ssh/known_hosts,
# установите значение yes для директивы IgnoreUserKnownHosts.
# Тогда будет использован только
# файл /etc/ssh/ssh_known_hosts
# IgnoreUserKnownHosts no

# Игнорировать файлы ~/.rhosts и ~/.shosts (рекомендуется установить yes)
# IgnoreRhosts yes

# Следующие директивы не рекомендуется изменять из соображений
# безопасности – они включают аутентификацию по паролю (а не IP-адресу
# компьютера, указанному в файле /etc/ssh/ssh_known_hosts)
# и запрещают использование пустых паролей
# PasswordAuthentication yes
# PermitEmptyPasswords no
# Параметры протокола аутентификации Kerberos
# Рекомендуется использовать RSA-аутентификацию
# KerberosAuthentication no
# KerberosOrLocalPasswd yes
# KerberosTicketCleanup yes
# KerberosGetAFSToken no
```



```
# Параметры GSSAPI
# GSSAPIAuthentication no
# GSSAPICleanupCredentials yes

# Использовать для аутентификации модули PAM (по умолчанию они
# не используются)
# UsePAM по
# Разрешить TCP-форвардинг
# AllowTcpForwarding yes

# Использовать порты шлюза
# GatewayPorts по

# Использовать X11-форвардинг (для запуска X11-приложений)
X11Forwarding yes

# Выводить сообщение дня (содержится в файле /etc/motd)
# PrintMotd yes

# Выводить время последней регистрации пользователя
# PrintLastLog yes

# Не обрывать TCP-соединения после выполнения команды по SSH
# TCPKeepAlive yes

# Отключение (значение по) этой опции позволяет немного ускорить работу
# SSH, поскольку DNS не будет использоваться для разрешения доменных имен
# UseDNS yes

# Остальные параметры рекомендуется оставить как есть
# UseLogin no
UsePrivilegeSeparation yes
# PermitUserEnvironment no
# Compression delayed
# ClientAliveInterval 0
# ClientAliveCountMax 3

# PidFile /var/run/sshd.pid
# MaxStartups 10

# Banner /some/path

Subsystem sftp /usr/lib/ssh/sftp-server
```

Итак, установив пакеты `openssh` и `openssh-clients`, приступим к тестированию работы SSH-сервера. Для его запуска следует использовать команду:

```
# service sshd start
```

А для останова — ту же команду, но с параметром `stop`:

```
# service sshd stop
```

В openSUSE для запуска/останова сервера используются команды (соответственно):

```
# rcssh start
# rcssh stop
```

Запустите также конфигуратор управления сервисами и убедитесь, что сервис sshd запускается при запуске системы. В Fedora таким конфигуратором является system-config-services, а в openSUSE для управления службами используется конфигуратор YaST: **YaST | Система | Системные службы**. В современных версиях Ubuntu конфигуратор, управляющий службами, отсутствует по умолчанию, и при особом желании вы можете самостоятельно установить графический конфигуратор bum (рис. 32.1).

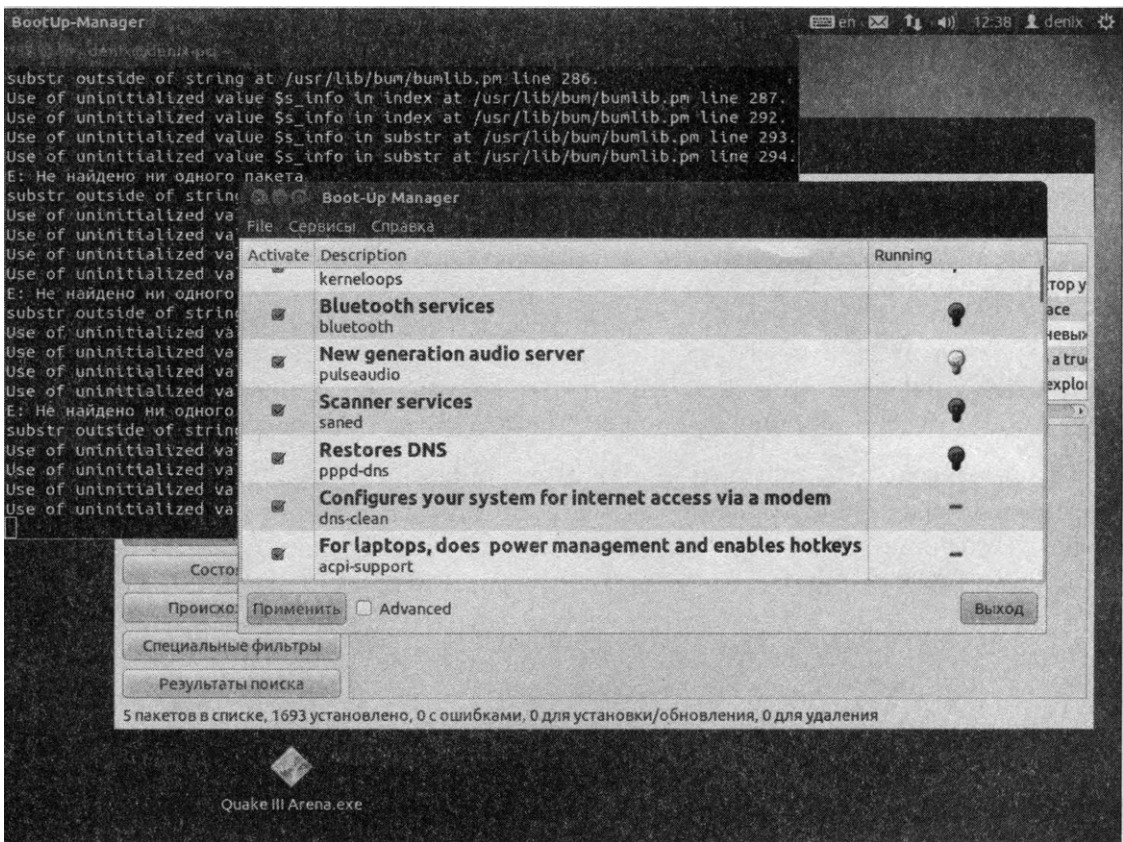


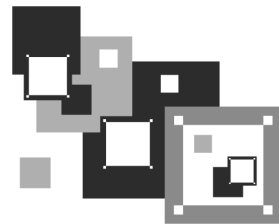
Рис. 32.1. Ubuntu: конфигуратор служб bum

После этого для подключения к локальному компьютеру можно ввести команду:

```
ssh 127.0.0.1
```

К SSH-серверу можно также подключиться и с удаленного компьютера— если сеть на локальном и удаленном компьютере настроена правильно, проблем возникнуть не должно.

ГЛАВА 33



Web-сервер. Связка Apache + PHP + MySQL

33.1. Самый популярный Web-сервер

Apache — это Web-сервер с открытым исходным кодом. История его развития началась в 1995 году — тогда Apache был всего лишь «заплаткой», устраняющей ошибки популярного в то время Web-сервера NCSA HTTPd 1.3. Считается, что отсюда произошло и название Apache (a patchy — заплатка). Сейчас Apache — самый популярный Web-сервер в Интернете.

Основные достоинства Apache — надежность, безопасность и гибкость настройки. Apache позволяет подключать различные модули, добавляющие в него новые возможности — например, можно подключить модуль, обеспечивающий поддержку PHP или любого другого Web-ориентированного языка программирования.

Но есть у Apache и недостатки — без этого никак, у любой медали всегда есть обратная сторона. Основной недостаток — отсутствие удобного графического интерфейса администратора. Да, настройка Apache осуществляется путем редактирования его конфигурационного файла. В Интернете можно найти простые конфигурации Apache, но их возможностей явно не хватает для настройки всех функций Web-сервера.

33.2. Установка Web-сервера и интерпретатора PHP. Выбор версии

Долгое время в репозиториях большинства дистрибутивов параллельно содержались две версии Apache: 1.3 и 2.2. Сейчас версия 1.x более не поддерживается, а вместо версии 2.2 обычно используется версия 2.4. Поэтому теперь можно не ломать себе голову вопросом, какую версию лучше установить.

Итак, приступим к установке Apache. Запустите менеджер пакетов (например, Synaptic, используемый в Ubuntu), произведите поиск пакета apache (в некоторых дистрибутивах нужный нам пакет называется apache2, а, например, в Fedora — httpd) и выберите пакет apache2. Менеджер пакетов сообщит вам, что нужно установить дополнительные пакеты (рис. 33.1).

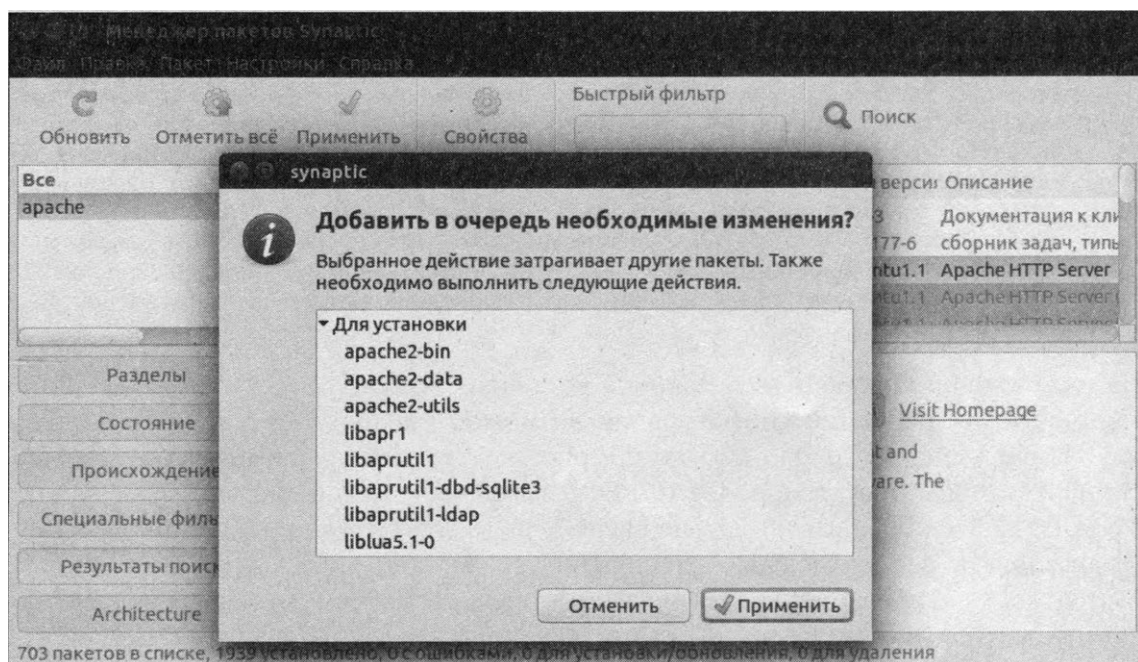


Рис. 33.1. Ubuntu: дополнительные пакеты для установки Apache

Чтобы сразу «убить двух зайцев», выберите еще и пакет `libapache2-mod-php5` — он устанавливает PHP5 и добавляет его поддержку в Apache. Менеджер снова предложит вам установить дополнительные пакеты, но теперь для PHP (рис. 33.2).

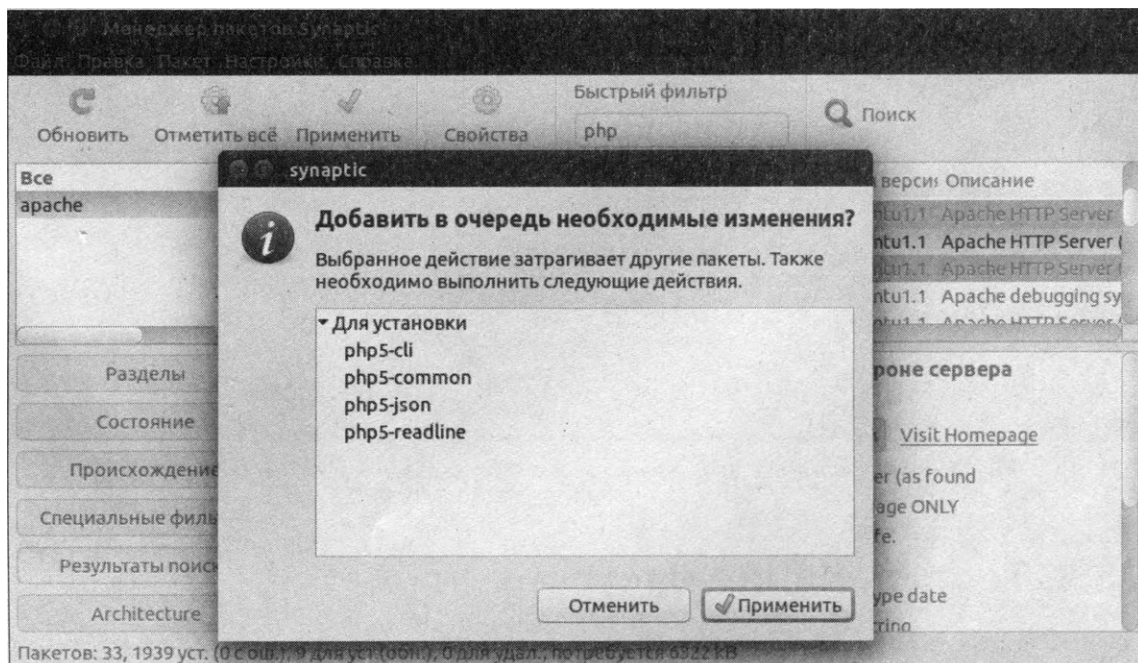


Рис. 33.2. Ubuntu: дополнительные пакеты для установки PHP5

О ВЕРСИЯХ PHP

На момент написания этих строк уже была доступной версия PHP 7.1, но поскольку версия 7.x пока редко используется в производственных окружениях, настройка Web-сервера производится на примере 5-й версии, которая пока еще является стандартом де-факто. При желании вы можете установить и 7-ю версию — просто установите пакеты, относящиеся к 7-й версии. Помните, что PHP 7 не поддерживает расширение `mysql`, поэтому не пытайтесь найти пакет `php-mysql7` в вашем дистрибутиве — его не будет. Работа с СУБД MySQL будет осуществляться или посредством расширения `PDO` или расширения `mysqli`. Если у вас есть старые сценарии, написанные с использованием расширения `mysql`, их придется переписать, иначе они не будут работать в PHP 7.

Нажмем кнопку **Применить**, и машина установит все выбранное (рис. 33.3). В зависимости от требуемой конфигурации возможно понадобится установить также пакеты `php5-imap`, `php5-gd`, `php5-mysql`, `php5-xmlrpc`, обеспечивающие поддержку, соответственно, протоколов IMAP/POP, графической библиотеки GD, СУБД MySQL и XML-RPC. Есть и другие расширения PHP, однако вряд ли имеет смысл устанавливать все возможные расширения (если, конечно, вы не настраиваете сервер хостинг-провайдера, где нужно обеспечить максимум возможностей для клиентов).

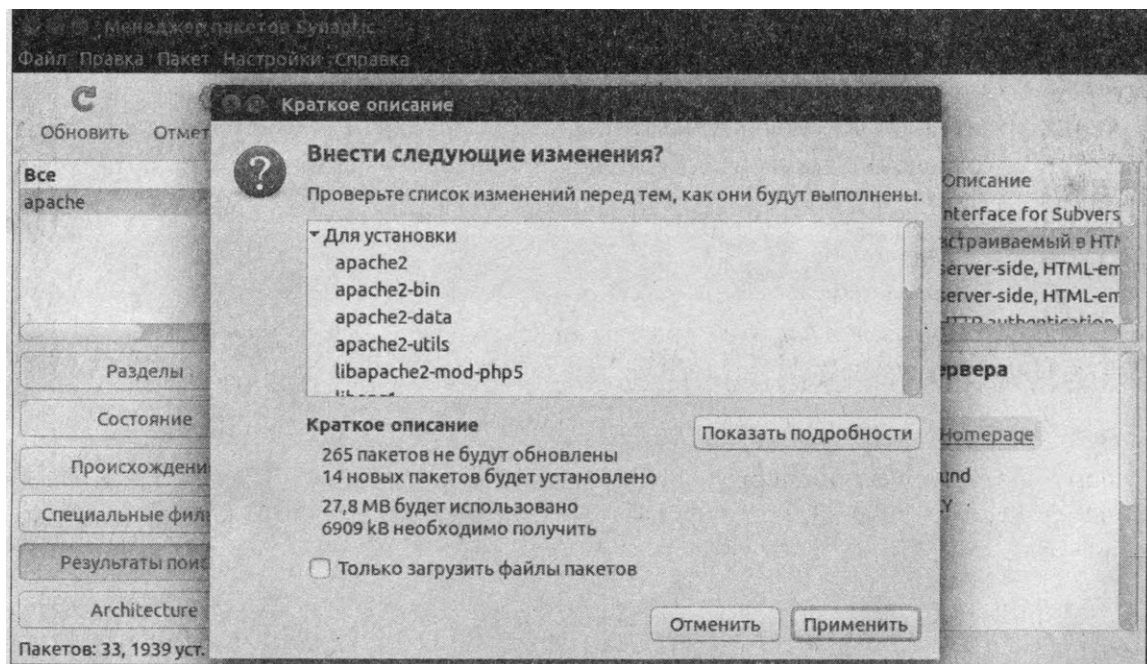


Рис. 33.3. Ubuntu: за миг до начала установки Web-сервера и всего необходимого

33.3. Тестирование настроек

Теперь протестируем Web-сервер. По идее, после установки сервер должен запускаться автоматически, но в некоторых дистрибутивах его придется запустить вручную (см. *разд. 33.5*).

Запустите сервер или убедитесь, что он запущен (см. *разд. 33.5*), откройте браузер и введите адрес:

http://localhost

Должна открыться тестовая страница Apache (рис. 33.4).

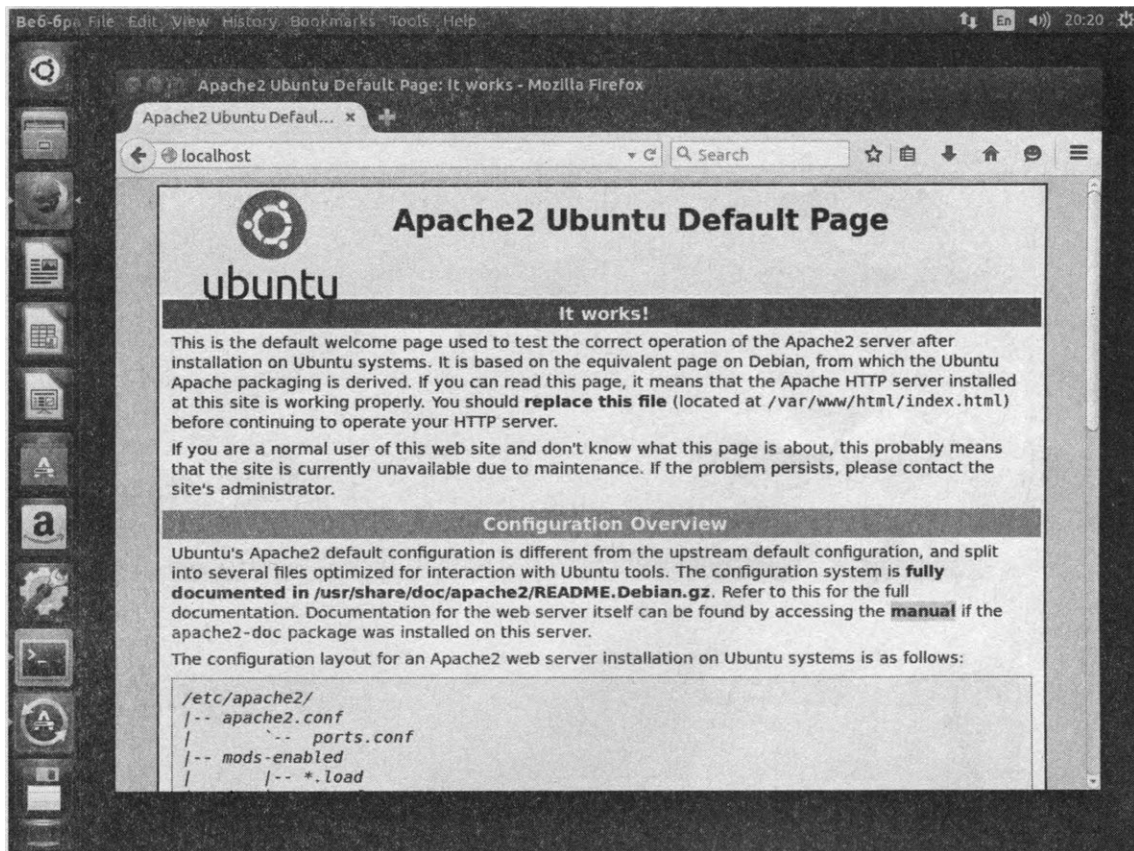


Рис. 33.4. Ubuntu: тестовая страница Apache

Теперь протестируем поддержку PHP. Для этого поместите в каталог `/var/www/html/` файл `test.php` (листинг 33.1). Учтите — чтобы создать файл в этом каталоге, нужны права root.

Листинг 33.1. Файл test.php

```
<?
phpinfo() ;

?>
```

Создав файл, введите в строке браузера следующий адрес:

http://localhost/test.php

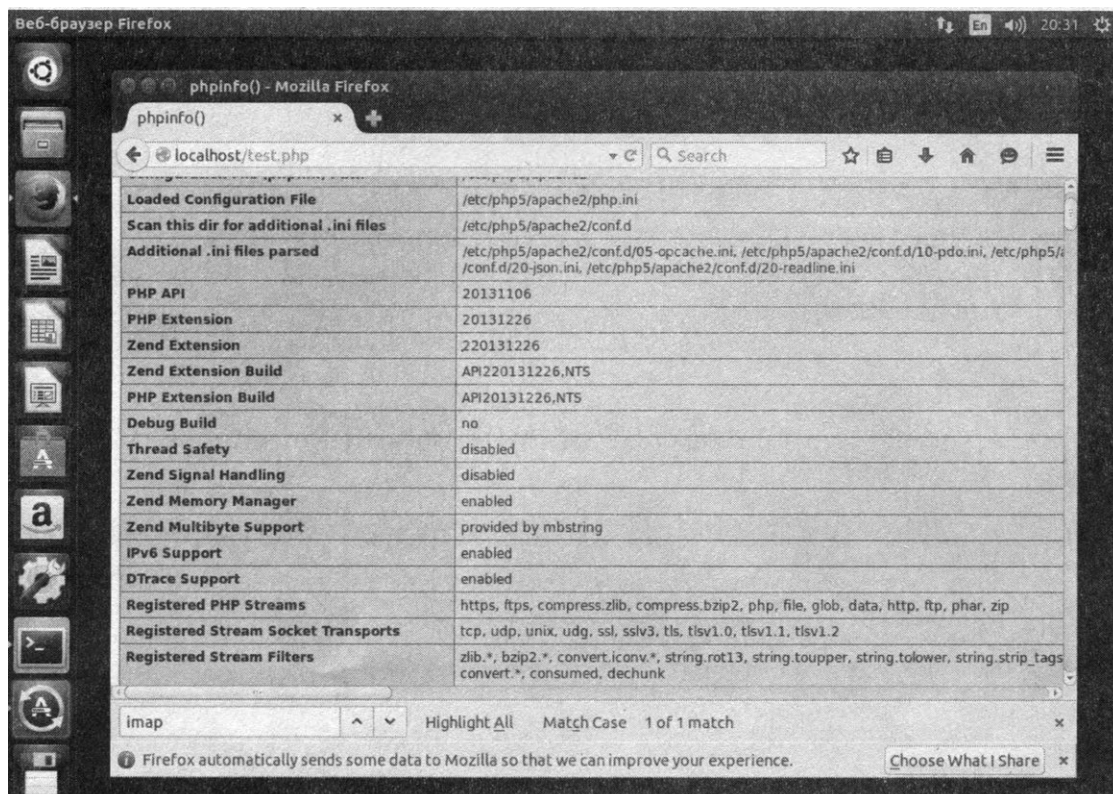


Рис. 33.5. Ubuntu: информация о Web-сервере и о PHP

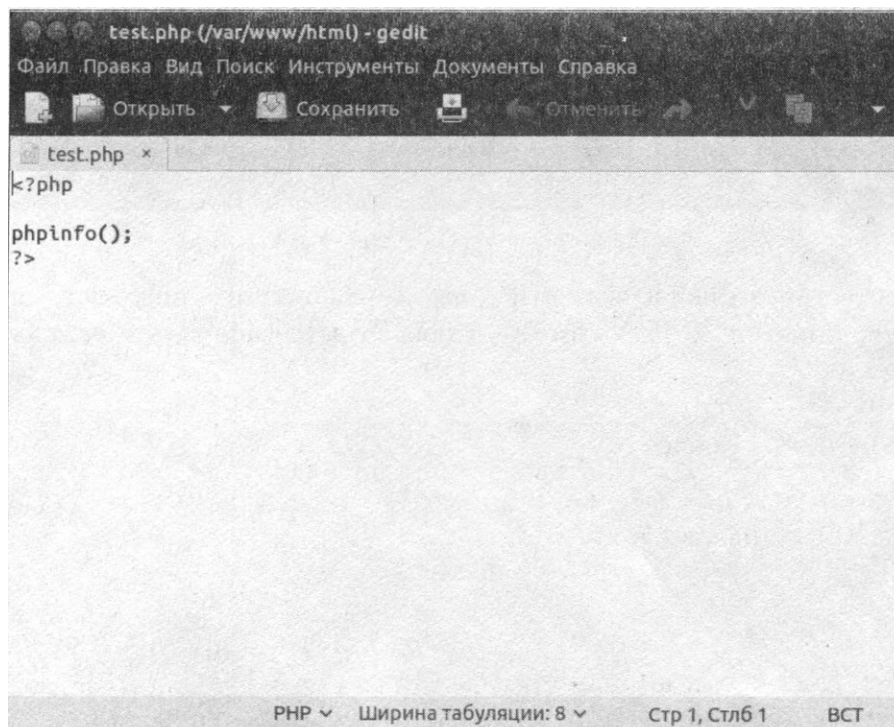


Рис. 33.6. Ubuntu: тестовый сценарий

В окне браузера вы должны увидеть информацию о своем сервере и о PHP (рис. 33.5), а на рис. 33.6 показан сам тестовый сценарий.

СОВЕТ

Если вместо отображения тестовой страницы, изображенной на рис. 33.4, браузер предложит вам сохранить файл *test.php*, перезапустите Web-сервер (см. *разд. 33.5*).

Как вы уже догадались, каталог */var/www/html* является корневым для нашего сервера, и если создать в нем файл *test.html*, то он будет доступен по адресу <http://localhost/test.html>.

33.4. Файл конфигурации Web-сервера

33.4.1. Базовая настройка

В зависимости от версии Apache и вашего дистрибутива, конфигурационные файлы Apache могут находиться в следующих каталогах: */etc/apache*, */etc/apache2*, */etc/httpd* или */etc/httpd2*. Основные конфигурационные файлы Web-сервера при этом называются *httpd.conf*, *httpd2.conf* или *apache.conf* и *apache2.conf*. Названия каталогов и файлов, содержащих слово «*apache*», характерно для дистрибутивов Debian и Ubuntu, а содержащих слово «*httpd*» — для Fedora. В любом случае найти конфигурационные файлы не сложно: ищите или *apache*, или *httpd* — и не промахнетесь!

ВНИМАНИЕ!

После каждого изменения конфигурационных файлов сервера его нужно перезапустить (см. *разд. 33.5*)!

В предыдущих версиях Linux все настройки хранились в одном огромном файле конфигурации. Сейчас этот файл чаще содержит Include-инструкции подключения других файлов конфигурации (более компактных). Все это сделано для удобства администраторов— проще работать с несколькими компактными файлами, чем с одним огромным, поэтому смотрите на все такие дополнительные файлы как на части основного файла конфигурации. Синтаксис у них такой же.

Итак, первым делом откройте основной конфигурационный файл (для определенности будем считать, что он называется *httpd2.conf*) и найдите директиву:

```
#ServerName new.host.наше
```

Ее следует раскомментировать и указать имя сервера, которое будут задавать пользователи в строке браузера. Это имя должно быть зарегистрировано в DNS-сервере вашей сети (или указано в файле */etc/hosts* каждого компьютера сети). Обычно здесь указывается имя компьютера, например:

```
ServerName user-desktop
```

После этого можно будет обращаться к серверу по адресу <http://user-desktop/>.

Кроме файла *apache2.conf* в подкаталогах каталога */etc/apache2* находятся дополнительные файлы, которые подключаются в основном файле конфигурации. Так, в файле *ports.conf* содержится описание портов, которые должен прослушивать сер-

вер, в подкаталоге `conf-enabled` — вспомогательные файлы конфигурации (и все они с «расширением» `.conf`), а описание различных модулей Apache находится в файлах из подкаталога `mods-enabled`.

33.4.2. Самые полезные директивы файла конфигурации

Понятно, что для полноценной настройки сервера одной директивы `ServerName` недостаточно. В табл. 33.1 приведены самые полезные директивы файла конфигурации Apache. Нужно отметить, что в таблице не рассматриваются некоторые директивы (например, `Port`, `BindAddress`), которые не используются во второй версии Apache.

Таблица 33.1. Директивы файла конфигурации

Директива	Описание
<code>ServerName</code> имя	Задает имя Web-сервера — оно должно быть зарегистрировано на DNS-сервере, т. е. обычно — это доменное имя сервера
<code>ServerAdmin</code> e-mail	Задает e-mail администратора сервера
<code>ServerRoot</code> каталог	Определяет каталог с конфигурационными файлами сервера
<code>PidFile</code> файл	Определяет имя файла, в котором будет храниться PID исходного процесса Web-сервера. Обычно изменять эту директиву не нужно
<code>DocumentRoot</code> каталог	Позволяет задать каталог, в котором хранятся документы Web-сервера, — это корневой каталог документов. Обычно это <code>/var/www</code>
<code>StartServers</code> N, <code>MaxSpareServers</code> N, <code>MinSpareServers</code> N, <code>MaxClients</code> N	Директивы, непосредственно влияющие на производительность сервера. Мы их рассмотрим отдельно в разд. 33.6
<code>KeepAlive</code> On Off, <code>KeepAliveTimeout</code> N	Управляют постоянными соединениями (будут рассмотрены в разб. 33.6)
<code>DirectoryIndex</code> список	Задает имена файлов, которые могут использоваться в качестве главной страницы (индекса). Значение по умолчанию <code>index.html index.cgi index.pl index.php index.xhtml</code>
<code>HostnameLookups</code> On Off	Если директива включена (On), то IP-адрес клиента перед записью в журнал будет разрешен (т. е. Web-сервер вычислит доменное имя клиента перед записью информации о попытке доступа в журнал). Выключение (Off) этой опции позволяет повысить производительность сервера, поскольку ему не нужно будет тратить время на разрешение IP-адресов в доменные имена
<code>ErrorLog</code> файл	Задает журнал ошибок
<code>TransferLog</code> файл	Задает журнал обращений к серверу

Таблица 33.1 (окончание)

Директива	Описание
Timeout N	Тайм-аут в секундах (время, на протяжении которого сервер будет ждать возобновления прерванной попытки передачи данных)
User пользовательGroup группа	Директивы User и Group задают имя пользователя и группы, от имени которых запускается Web-сервер
FancyIndexing on off	Если пользователь в запросе не укажет имя документа, а только каталог, но в нем не окажется главной страницы, заданной директивой DirectoryIndex, сервер передаст пользователю оглавление каталога. Эта директива определяет, в каком виде будет передано оглавление каталога: в более красивом, со значками каталогов и описаниями файлов (значение On), или в более простом (Off)
AddIcon картинка список	Если Fancyindexing включена, то AddIcon позволяет связать графическую картинку с типом файла, например: AddIcon /images/graphics.gif .gif, .jpeg, .bmp, .png, .tiff
DefaultIcon картинка	Позволяет задать картинку по умолчанию (AddIcon, Fancyindexing)
ErrorDocument N файл	Позволяет задать файл, содержащий сообщение об ошибке. Для ошибки с номером N, например: ErrorDocument 404 /errors/file_not_found.html
Directory, Limit, Location, Files	Это так называемые <i>блочные</i> директивы, которые нельзя описать одной строкой, поэтому о них мы поговорим отдельно (см. разд. 33.4.3)

33.4.3. Директивы *Directory, Limit, Location, Files*

Рассмотрим сначала блочные директивы *Directory* и *Limit*.

- С помощью блочной директивы *Directory* можно установить параметры отдельного каталога. Внутри директивы *Directory* могут использоваться директивы *AllowOverride, Limit, Options*. Вот пример определения параметров корневого сервера:

```
<Directory />
AllowOverride None
Options None
</Directory>
```

Значения *None* для обеих директив (*AllowOverride* и *Options*) считаются самыми безопасными. *None* для *AllowOverride* запрещает использование файлов *.htaccess*, которые могут переопределять директивы конфигурационного файла *Apache*. К тому же, *AllowOverride None* позволяет повысить производительность сервера.

Допустимые опции каталога (значения директивы Options) указаны в табл. 33.2.

Таблица 33.2. Опции каталога

Опция	Описание
None	Запрещены все опции
All	Все опции разрешены
Indexes	Если указана эта опция, при отсутствии файла, заданного Directory Index, будет выведено оглавление каталога. Если Options установлена в None (или Indexes не указана в списке опций), то оглавление каталога выводиться не будет
Includes	Разрешает использование SSI (Server Side Includes)
IncludesNoExec	Более безопасный режим SSI: разрешает SSI, но запрещает запускать из включений внешние программы
ExecCGI	Разрешает выполнение CGI-сценариев
FollowSymLink	Разрешает использование символических ссылок. Довольно опасная опция, поэтому лучше ее не использовать

- Блочная директива Limit позволяет ограничить доступ. Внутри этой директивы можно использовать директивы order, deny и allow (вообще-то, есть еще и директива require, но она очень редко используется). Директива order задает порядок выполнения директив deny и allow:

```
# сначала запретить, потом разрешить
order deny, allow
# сначала разрешить, потом запретить
order allow, deny
```

Директивы allow и deny нужно использовать так:

```
# запрещаем доступ всем
deny from all
# разрешаем доступ только нашей сети
allow from firma.ru
```

Пример использования директив Directory и Limit представлен в листинге 33.2.

Листинг 33.2. Фрагмент файла конфигурации Apache

```
<Directory />
AllowOverride None
Options None
<Limit>
    order deny, allow
    # запрещаем доступ всем
    deny from all
```

```
# разрешаем доступ только нашей сети
allow from firma.ru
</Limit>

</Directory>
```

В качестве параметра директиве `Limit` можно передать метод передачи данных (`GET`, `POST`), например:

```
<Limit GET>
<Limit POST>
```

Теперь обратимся к блочным директивам `Location` и `Files`.

- Директива `Location` очень похожа на директиву `Directory`. Только если `Directory` ограничивает доступ к каталогу, то `Location` предназначена для ограничения доступа к отдельным адресам (URL) сервера:

```
<Location URL>
директивы ограничения доступа
</Location>
```

К директивам ограничения доступа относятся `order`, `deny`, `allow`.

- Директива `Files` предназначена для ограничения доступа к отдельным файлам:

```
<Files файл>
директивы ограничения доступа
</Files>
```

Вы можете указать как отдельный файл, так и регулярное выражение, которому должны соответствовать файлы:

```
# запрещаем доступ к файлу privat.html всем, кроме нашей сети
<Files privat.html>
order deny, allow
deny from all
allow from firma.ru
</Files>

# запрещаем доступ к файлам . ht* всем
<Files ~ "^\.ht">
    Order allow,deny
    Deny from all
</Files>
```

Мы рассмотрели все самые полезные директивы конфигурационного файла Apache. Напомню, что директивы, непосредственно влияющие на производительность сервера, рассмотрены в *разд. 33.6*.

33.5. Управление запуском сервера Apache

Для управления Web-сервером можно использовать команду `service`:

- ☐ `sudo service httpd start` — запуск сервера;
- ☐ `sudo service httpd stop` — останов сервера;
- ☐ `sudo service httpd restart` — перезапуск сервера.

Понятно, что Web-сервер запускается автоматически, поэтому каждый день вам не придется вводить команду `service httpd start`.

Как уже отмечалось ранее, в разных дистрибутивах служба Web-сервера называется по-разному: `httpd`, `httpd2`, `apache` или `apache2`. В последней версии Ubuntu служба называется именно `apache2`, а в Fedora — `httpd`. Поэтому, если система сообщила, что в ней нет вызванного вами сервиса, попробуйте просто использовать другое название.

В новых версиях Ubuntu команда `service` присутствует, поэтому управлять Apache можно так, как здесь и показано. В старых версиях Ubuntu и Debian команды `service` нет, поэтому управлять Apache надо так:

```
sudo /etc/init.d/apache2 start
sudo /etc/init.d/apache2 stop
sudo /etc/init.d/apache2 restart
```

33.6. Оптимизация Apache

В конфигурационном файле сервера Apache `httpd.conf`, находящемся в каталоге `/etc/apache` или в каталоге `/etc/httpd/conf` (в зависимости от дистрибутива и версии Apache), имеется ряд директив, позволяющих оптимизировать работу сервера.

- ☐ Директива `MaxClients` позволяет ограничить число одновременно работающих клиентов.

Чтобы правильно установить это значение, нужно знать, сколько пользователей может одновременно зайти на сервер. При небольшой посещаемости вполне хватит значения 30-50, при большой загрузке количество одновременно работающих клиентов может исчисляться сотнями. Следите за посещаемостью вашего сервера и корректируйте это значение, иначе какая-то часть пользователей может остаться «за бортом», а им это очень не понравится (или же, наоборот, ресурсы сервера будут использоваться нерационально).

- ☐ Директива `StartServers` задает количество экземпляров сервера, которые будут созданы при запуске исходной копии сервера.

Для этой директивы можно установить значение, равное 10% от `MaxClients`. Устанавливать большое значение не следует во избежание нерационального использования ресурсов компьютера.

Рассмотрим обычную ситуацию. Для `MaxClients` вы установили значение 200, а для `StartServers` — 20. Запросы первых 20 клиентов будут обрабатываться очень быстро, поскольку сервисы уже запущены. Запрос 21-го клиента будет обслужен чуть медленнее, поскольку понадобится запустить еще одну копию Apache. И тем не менее, не нужно устанавливать в нашем случае (`MaxClients` = 200) для `StartServers` значение больше 20 — ведь не всегда даже 20 человек одновременно заходят на сервер. Если же на сервере постоянно находится как минимум 20 человек, тогда нужно увеличить значения и `MaxClients`, и `StartServers`.

Впрочем, бывают и исключения — например, если сервер обслуживает внутреннюю корпоративную сеть. В этом случае вы точно знаете, сколько клиентов в вашей сети, а следовательно, можете точно определить, какое значение установить для `MaxClients` и `StartServers`. Но, все равно, для `MaxClients` нужно установить чуть большее значение, чем для `StartServers` — на всякий случай:

```
MaxClients 150
StartServers 100
```

- Чтобы еще эффективнее оптимизировать работу Web-сервера, нужно понимать, как он работает: клиент посылает запрос, Web-сервер его обрабатывает и посылает клиенту ответ. После этого соединение можно закрывать и завершать копию Apache, обслуживающую это соединение. Но зачем завершать копию Web-сервера, если сейчас же на сайт зайдет другой пользователь, и опять нужно будет запускать еще одну копию сервера, что только увеличит загрузку процессора. Поэтому с помощью директивы `MaxSpareServers` можно установить максимальное число серверов, которые останутся в памяти уже после закрытия соединения с пользователем, — они будут просто ждать своего пользователя. Теоретически, чтобы сбалансировать нагрузку, значение для `MaxSpareServers` можно установить таким же, что и для `StartServers`, *е.е.* 10% от `MaxClients`
- Вы не задумывались, что если Web-сервер будет работать в режиме постоянного соединения, то это повысит его производительность? Если вы об этом подумали, то мыслите в правильном направлении. Представим, что у нас на сайте есть форум. Человек редко заходит на форум, чтобы посмотреть одну страничку, — обычно он может находиться на форуме часами. Так зачем же закрывать соединение? Чтобы потом опять тратить время и ресурсы на его открытие? Разрешить постоянные соединения можно с помощью директивы `KeepAlive`. Она задает максимальное число таких соединений:

```
KeepAlive 5
```

- А директива `KeepAliveTimeout` задает тайм-аут для постоянного соединения в секундах:

```
KeepAliveTimeout 15
```

Используя все упомянутые в этом разделе директивы, вы сможете добиться существенного повышения производительности своего Web-сервера.

33.7. Пользовательские каталоги

Если вы когда-нибудь настраивали сервер Apache, то наверняка знакомы с директивой `userDir`. Я специально ее не описал в табл. 33.1, потому что она заслуживает отдельного разговора.

По умолчанию директива `UserDir` отключена:

```
UserDir disabled
```

Включить ее можно, указав вместо `disabled` любое другое значение, — обычно указывается значение `public_html`:

```
UserDir public_html
```

Затем в пользовательском каталоге `/1"юте/<имя>` создается каталог `public_html` и в него помещаются HTML/PHP-файлы персонального сайта пользователя. Обращение к сайту пользователя происходит по URL:

`http://имя_сервера/~имя_пользователя`

Например, если при включенной директиве `UserDir` вы поместили в каталог `/home/den/public_html` файл `report.xml`, то обратиться к нему можно по адресу:

`http://server/~den/report.xml`

Недавно, настраивая сервер на базе openSUSE, я столкнулся с небольшой проблемой. Ранее, во времена огромного конфигурационного файла, достаточно было раскомментировать эту директиву в конфигурационном файле. Сейчас, когда конфигурация сервера состоит из нескольких небольших файлов, добавление этой опции в основной конфигурационный файл не привело ни к каким изменениям. Оказалось, опцию `UserDir` нужно добавить (точнее, просто раскомментировать) в файл `/etc/apache2/mod_userdir.conf`¹. А затем добавить следующую строку в самый конец файла `/etc/apache2/default-server.conf`:

```
Include /etc/apache2/mod_userdir.conf
```

После всего этого следует перезапустить сервер.

33.8. Установка сервера баз данных MySQL

33.8.1. Установка сервера

Для организации связки Apache + PHP + MySQL нам осталось установить последний компонент — сервер баз данных MySQL. Для установки MySQL-сервера установите следующие пакеты:

- ☐ `mysql-server-5.0`;
- ☐ `mysql-client-5.0`;
- ☐ `mysql-admin`.

¹ Не забывайте указывать свой путь к файлу конфигурации, поскольку каталог с файлами конфигурации может отличаться в зависимости от версии дистрибутива и самого Apache.

Первый пакет содержит последнюю версию MySQL-сервера (на данный момент — это пятая версия), во втором пакете находится MySQL-клиент, т. е. программа, которая будет подключаться к MySQL-серверу, передавать ему SQL-запросы и отображать результат их выполнения. Третий пакет содержит программу для администрирования MySQL-сервера. Все необходимые дополнительные пакеты будут установлены автоматически.

33.8.2. Изменение пароля root и добавление пользователей

Сразу после установки пакетов введите следующие команды:

```
# mysql_install_db
# mysqladmin -u root password ваш_пароль
```

ЕСЛИ ЧТО-ТО ПОЙДЕТ НЕ ТАК...

В процессе выполнения команды `mysql_install_db` вы можете получить сообщение:

[ERROR] /usr/libexec/mysqld: Can't find file: './mysql/help_relation.frm' (errno: 13)

Поможет команда `chown -r mysql /var/lib/mysql`. После ее выполнения нужно заново выполнить команду `mysql_install_db`.

Первая команда (`# mysql_install_db`) создаст необходимые таблицы привилегий, а вторая (`# mysqladmin -u root password ваш_пароль`) — задаст пароль пользователя `root` для сервера MySQL. Этот пароль вы будете использовать для администрирования сервера (заданный пароль может и должен отличаться от того, который вы используете для входа в систему). Для обычной работы с сервером рекомендуется создать обычного пользователя. Для этого введите команду:

```
mysql -u root -p mysql
```

Программа `mysql` является клиентом MySQL-сервера. В указанном случае она должна подключиться к базе данных `mysql` (служебная база данных), используя имя пользователя `root` (`-u root`). Поскольку вы только что указали пароль для пользователя `root` (до этого пароль для `root` не был задан), вам нужно указать параметр `-p`. После того как программа `mysql` подключится к серверу, вы увидите приглашение программы. В ответ на него нужно ввести следующий SQL-оператор:

```
insert into user(Host, User, Password, Select_priv, Insert_priv, Update_priv,
Delete_priv)
values ('%', 'username', password('123456'), 'Y', 'Y', 'Y', 'Y');
```

Таким оператором мы создали пользователя с именем `username` и паролем `123456`. Этот пользователь имеет право использовать SQL-операторы `select` (выборка из таблицы), `insert` (добавление новой записи в таблицу), `update` (обновление записи), `delete` (удаление записи). Если нужно, чтобы ваш пользователь имел право создавать и удалять таблицы, тогда добавьте привилегии `Create_priv` и `Drop_priv`:

```
insert into user(Host, User, Password, Select_priv, Insert_priv, Update_priv,
Delete_priv, Create_priv, Drop_priv)
values ('%', 'username', password('123456'), 'Y', 'Y', 'Y', 'Y', 'Y', 'Y');
```


ПОМНИТЕ ПРО ТОЧКУ С ЗАПЯТОЙ!

Приведенный здесь SQL-оператор можно записать в одну строку, можно разбить на несколько строк — как вам будет удобно. Но в конце каждого SQL-оператора должна быть точка с запятой!

Для выхода из программы `mysql` нужно ввести команду `quit`.

Кроме программы `mysql`, в состав MySQL-клиента входит одна очень полезная программа — `mysqlshow`, которая может вывести список таблиц, находящихся в той или иной базе данных. Кроме этого, она еще много чего может, но сейчас нам нужен пока список таблиц — чтобы вы знали, какие таблицы есть в базе данных:

```
mysqlshow -p <база данных>
```

33.8.3. Запуск и останов сервера

Для управления MySQL-сервером служит программа `/etc/init.d/mysql`. Чтобы запустить сервер, нужно передать этой программе параметр `start`, для останова — `stop`, а для перезапуска — `restart`:

```
sudo /etc/init.d/mysql start
sudo /etc/init.d/mysql stop
sudo /etc/init.d/mysql restart
```

В Fedora можно воспользоваться командой `service`:

```
# service mysql start
# service mysql stop
# service mysql restart
```

Также для управления сервером можно использовать программу `mysqladmin`, узнать больше о ней можно с помощью команды:

```
man mysqladmin
```

33.8.4. Программа MySQL Administrator

При установке сервера мы установили программу MySQL Administrator (пакет `mysql-admin`). Запустите программу командой меню **Приложения | Программирование | MySQL Administrator**. Укажите адрес сервера `localhost`, имя пользователя — `root`, пароль, который вы указали при установке сервера (рис. 33.7), и нажмите кнопку **Connect**.

Далее управлять сервером будет существенно проще (рис. 33.8). Пройдемся по основным разделам программы MySQL Administrator:

- **Server Information** — общая информация о сервере;
- **Service Control** — управление запуском сервиса MySQL (здесь вы можете перезапустить сервер);
- **Startup Parameters** — параметры, указываемые при запуске сервера;
- **User Administration** — здесь можно добавить новых пользователей MySQL и установить права пользователей;



Рис. 33.7. Ubuntu: вход на сервер MySQL

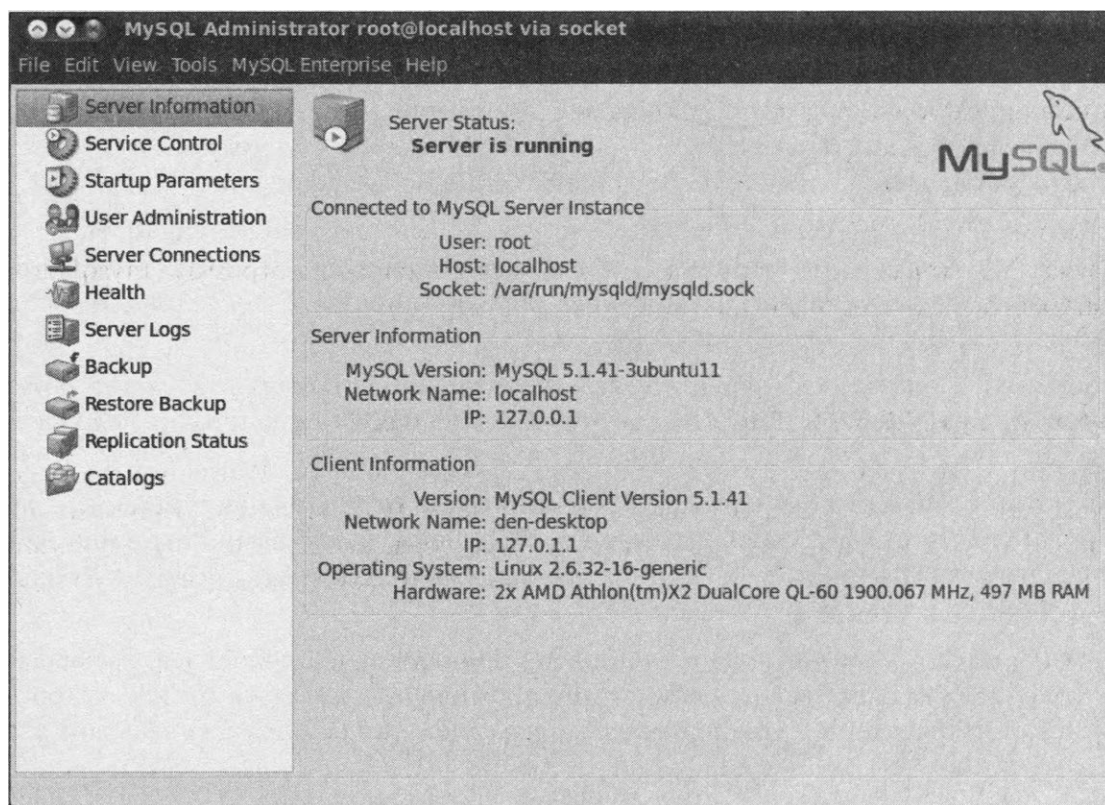


Рис. 33.8. Ubuntu: основное окно программы MySQL Administrator

- **Server Connections** — позволяет просмотреть текущие соединения с сервером;
- **Server Logs** — журналы сервера;
- **Backup** — создание резервной копии сервера;
- **Restore Backup** — восстановление из резервной копии;
- **Replication Status** — состояние репликации сервера;
- **Catalogs** — позволяет просмотреть имеющиеся базы данных и таблицы внутри них.

33.9. Обеспечение безопасности сайта от вирусов

33.9.1. Как вирусы попадают на сайт?

В последнее время замечена стойкая тенденция распространения интернет-вирусов, и в этом издании я не мог не затронуть этот вопрос. Только в текущем, 2017-м, году ко мне обратились более 20 знакомых с просьбой удалить вирус с их сайта, и при этом в нескольких случаях это были сайты крупных и известных компаний.

Первым делом нужно понять, как вирус попадает на сайт, а потом уже разрабатывать стратегию защиты сайта. Как правило, вирус проникает в сайты, разработанные с использованием готовых т. н. *систем управления содержимым* (от англ. Content Management System, CMS): Joomla!, WordPress, Magento и пр.

Код таких CMS открыт и доступен всем, в том числе и злоумышленникам. Все, что им остается, — это найти уязвимость или в коде самой CMS, или в стороннем плагине/скине, а затем найти сайт, имеющий эту уязвимость. С последним отлично справится Google или другой поисковик.

У злоумышленников, как правило, есть специальные боты, производящие поиск в сети сайтов с уязвимостями. После того, как сайт найден, происходит внедрение вируса.

Как с этим бороться? Если есть возможность, нужно отдать предпочтение не готовым CMS, а так называемым «кастомным». Код, написанный вами лично или приглашенным программистом, не будет содержать общеизвестных уязвимостей, поэтому боты просто не «увидят» ваш сайт.

Конечно, вы тоже можете допустить ошибку при написании сценариев, и в вашем коде тоже могут оказаться уязвимости. Но они будут отличаться от тех, которые имеются в стандартных CMS, поэтому стандартные методы взлома на ваш сайт уже действовать не будут.

Взломать можно любой сайт. Но тогда злоумышленнику придется взламывать именно ваш сайт, тратить время на поиск уязвимости, что не так просто сделать, не имея кода под рукой (а он будет закрыт, ясное дело).

Конечно, не всегда есть возможность по тем или иным причинам написать код самому. Тогда нужно обезопасить то, что есть.

33.9.2. Установка прав доступа

Прежде всего следует с помощью файла `.htaccess` закрыть доступ к панели управления сайтом. В случае с WordPress нужно ограничить доступ к сценарию `wp-login.php`. Делается это так:

```
<Files wp-login.php> •
order deny,allow
deny from all
allow from Bam_ip
</Files>
```

Даже если у вас динамический IP-адрес, то проще его изменить в файле `.htaccess`, чем дожидаться взлома вашего сайта.

Если жестко ограничить доступ невозможно, в качестве выхода из ситуации для защиты Web-, SSH- и почтового сервера можно использовать программу Fail2ban. Эта программа, если обнаружит подозрительное поведение, — например, попытку брутфорса пароля, просто заблокирует нарушителя. Блокировка осуществляется посредством iptables. Подробное рассмотрение Fail2ban выходит за рамки этой книги, но могу посоветовать неплохое руководство на русском языке: <https://vps.ua/wiki/install-linux-vps/security/configuring-fail2ban/>.

Вторая часто встречающаяся проблема — неправильные и слишком высокие права доступа к файлам и каталогам вашего сайта. В идеале нужно предоставить доступ «только чтение» ко всем файлам и каталогам сайта пользователю `www-data` (от его имени в современных дистрибутивах запускается Apache).

Для этого перейдите в каталог DocumentRoot (обычно это `/var/www/html`, но точный путь зависит от настроек сервера) и введите команду:

```
chown -R www-data:www-data .
```

Эта команда рекурсивно меняет владельца всех файлов и каталогов сайта на `www-data`.

Затем установите права доступа «только чтение» (500 — для каталогов и 400 — для файлов):

```
find . -type f -exec chmod 400 {} \;
find . -type d -exec chmod 500 {} \;
```

Для файлов и каталогов, которым необходим доступ «чтение запись» (600 и 700 соответственно), чтобы обеспечить нормальное функционирование сайта (а это каталоги, в которые загружаются служебные данные, изображения и т. д.), нужно изменить права так:

```
find var/ -type f -exec chmod 600 {} \;
find media/ -type f -exec chmod 600 {} \;
find var/ -type d -exec chmod 700 {} \;
find media/ -type d -exec chmod 700 {} \;
```

33.9.3. Антивирус ClamAV

Не будет лишним и применить антивирус, для чего установите пакет clamav. В результате будет создана группа и пользователь с таким же именем, поэтому в средстве мониторинга (вроде Zabbix) вы можете получить уведомление об изменении файла /etc/passwd.

После установки антивируса нужно сразу же обновить антивирусные базы:

```
# freshclam
```

Не забудьте также добавить задание cron для обновления баз, выполнив команду:

```
# crontab -e
```

Добавьте в задание строку:

```
0 0 * * * /usr/local/bin/freshclam -quiet -l /var/log/clam-update.log
```

Она обеспечит запуск обновления в 0 часов и 0 минут каждый день.

АНТИВИРУС CLAMAV

Более подробное описание антивируса ClamAV вы найдете в папке *Дополнения* сопровождающего книгу электронного архива (см. *приложение*).

Теперь о сканировании. Для сканирования можно использовать либо сканер clamscan, либо демон clam-daemon. Последний устанавливается отдельно (пакет clam-daemon).

Запускать сканер нужно так:

```
clamscan -r -i [каталог]
```

Если каталог не задан, то проверка начнется с текущего каталога. Ключ -i обеспечивает вывод информации только об инфицированных файлах. На рис. 33.9 показан вывод отчета о сканировании — как видите, антивирус обнаружил один инфицированный файл (полный путь здесь я, естественно, затер, чтобы не демонстрировать имя узла, файлы которого проверял).

При желании можно запланировать проверку (если вы не хотите по каким-то причинам использовать демон clam-daemon), скажем, раз в сутки так:

```
clamscan -r -i /var/www/html | mail user@example.com
```

Эту команду нужно добавить в задание планировщика cron.

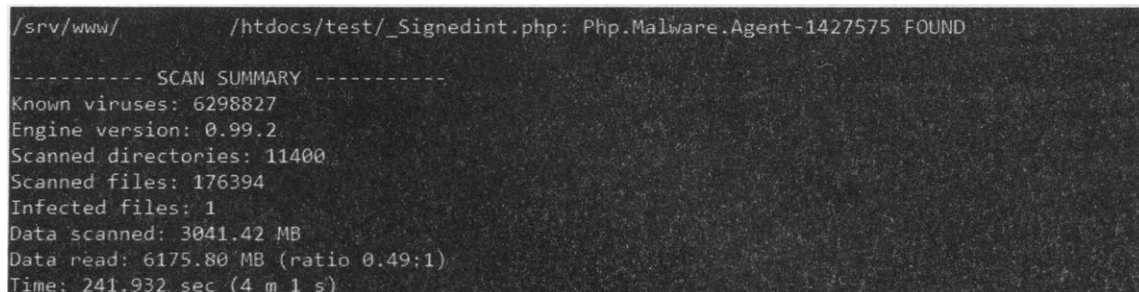


Рис. 33.9. Отчет о проверке

33.9.4. Сценарий scanner

Для собственного использования я написал простенький bash-сценарий, который ищет файлы, содержащие подозрительные PHP-инструкции, а также файлы .html и .php, изменившиеся за последние 20 дней (рис. 33.10). Не обязательно, что все найденные файлы заражены вирусами, но такие файлы нужно держать на контроле, особенно те, которые вы не изменяли.

Листинг 33.3. Сценарий scanner

```
#!/bin/bash

fld="/var/www/html"

echo "Следующие файлы содержат подозрительный код (инструкции eval,
base64_decode, preg_replace):"

grep -RE 'preg_replace\\(|eval\\(|base64_decode\\(' --include=*.*.php' $fld |
cut -d: -f 1 | sort -u

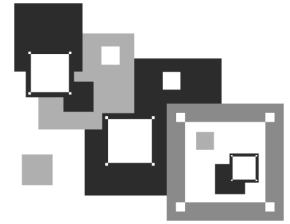
echo "Файлы, модифицированные за последние 20 дней:"

find $fld -name '*.php' -type f -mtime -20 ! -mtime -1 -printf '%TY-%Tm-%Td
%TT %p\n'
find $fld -name '*.html' -type f -mtime -20 ! -mtime -1 -printf '%TY-%Tm-%Td
%TT %p\n'
```

```
/srv/www/          /htdocs/test/analyze/salesplug.php
/srv/www/          /htdocs/test/_Signedint.php
Файлы, модифицированные за последние 20 дней:
2017-06-21 17:36:36.2327712000 /srv/www/          /htdocs/skin/Signedint.php
2017-06-21 17:44:08.5687712000 /srv/www/          /htdocs/salesplug/salesplug.php
2017-06-15 17:22:07.5527712000 /srv/www/          /htdocs/mail.php
2017-06-21 15:25:53.1727712000 /srv/www/          /htdocs/upload2.php
2017-06-21 14:23:31.4687712000 /srv/www/          /htdocs/upload/upload.php
```

Рис. 33.10. Результат работы сценария scanner

ГЛАВА 34



FTP-сервер

Сервер FTP (File Transfer Protocol) обеспечивает обмен файлами между пользователями Интернета. Осуществляется это следующим образом: на FTP-сервере размещается какой-нибудь файл, а пользователи с помощью FTP-клиента (в любой операционной системе имеется стандартный FTP-клиент— программа `ftp`) подключаются к FTP-серверу и скачивают этот файл.

Права пользователя FTP-сервера определяются его администратором. Одним пользователям разрешается загружать файлы в свои личные каталоги на сервере, другие имеют полный доступ к FTP-серверу (имеют право загружать файлы в любые каталоги — как правило, это администраторы FTP-сервера), третьи могут только скачивать публично доступные файлы. Третья группа пользователей самая большая — это так называемые *анонимные пользователи*. Чтобы не создавать учетную запись для каждого анонимного пользователя, все они работают под так называемой *анонимной учетной записью*, когда вместо имени пользователя указывается имя `anonymous`, а вместо пароля — адрес электронной почты пользователя.

В локальной сети обмен файлами можно организовать с помощью сервера Samba, имитирующего работу рабочей станции под управлением Windows, в Интернете же для обмена файлами надо использовать только FTP-сервер. С другой стороны, ничего не мешает вам организовать FTP-сервер для обмена файлами внутри локальной сети, — это дело вкуса и предпочтений администратора.

Все необходимое для организации FTP-сервера программное обеспечение входит в состав дистрибутива или же бесплатно доступно для скачивания в Интернете. Здесь мы рассмотрим самый удобный, на мой взгляд, FTP-сервер ProFTPD. Это не единственный FTP-сервер для Linux, есть еще и другие FTP-серверы, — например, `wu-ftp`, но ProFTPD является одним из самых защищенных и удобных в настройке.

34.1. Установка FTP-сервера

Для установки FTP-сервера нужно установить пакет `proftpd` (рис. 34.1). Хорошо бы также установить и конфигуратор `gproftpd`, если он доступен в вашем дистрибутиве.


```

ServerType      standalone      # автономный
DeferWelcome    off              # вывести приветствие до
                                   # аутентификации

MultilineRFC2228 on              # поддержка RFC2228
DefaultServer   on              # сервер по умолчанию
ShowSymlinks    on              # показывать символические ссылки

# настройка таймаутов
TimeoutNoTransfer 600
TimeoutStalled 600
TimeoutIdle 1200

DisplayLogin     welcome.msg    # файл с приветствием
DisplayFirstChdir .message      # отобразить этот файл при
                                   # каждой смене каталога

# запрещает использовать это выражение в FTP-командах
# (все файлы (маска *.* ) вы уже не сможете удалить, придется удалять
# поодиночке!)
DenyFilter       \*/

Port             21              # стандартный порт

MaxInstances     30              # количество копий proftpd
# пользователь и группа, от имени которых работает proftpd
User             proftpd
Group            nogroup
Umask            022 022        # см. man umask

AllowOverwrite   on              # можно перезаписывать файлы

# Журналы сервера
TransferLog /var/log/proftpd/xferlog
SystemLog /var/log/proftpd/proftpd.log

# Параметры подключаемых модулей. Изменять не нужно
<IfModule mod_tls.c>
TLSEngine off
</IfModule>

<IfModule mod_quota.c>
QuotaEngine on
</IfModule>

<IfModule mod_ratio.c>
Ratios on
</IfModule>

```

```
<IfModule mod_delay.c>
DelayEngine on
</IfModule>

<IfModule mod_ctrls.c>
ControlsEngine          on
ControlsMaxClients      2
ControlsLog              /var/log/proftpd/controls.log
ControlsInterval        5
ControlsSocket           /var/run/proftpd/proftpd.sock
</IfModule>

<IfModule mod_ctrls_admin.c>
AdminControlsEngine on
</IfModule>
```

В конфигурационном файле `proftpd.conf` вы можете использовать как обычные директивы, задающие одиночные свойства, так и блочные директивы, определяющие группы свойств (параметров). Например, директива `ServerName` — обычная, она задает одно свойство, а директива `Directory` — блочная, позволяющая задать несколько параметров для одного каталога.

Самые полезные директивы файла конфигурации сведены в табл. 34.1. С остальными вы всегда можете ознакомиться, прочитав документацию по ProFTPD.

Таблица 34.1. Директивы файла конфигурации `proftpd.conf`

Директива	Описание
AccessGrantMsg "сообщение"	Задает сообщение, которое будет отправлено пользователю при его регистрации на сервере. Можно задать грозное сообщение, напоминающее о том, что попытка несанкционированного доступа карается статьей такой-то уголовного кодекса
Allow from <i>all</i> <i>узел</i> <i>сеть</i> [<i>,узел</i> <i>сеть</i> [<i>,</i> . . .]]	Используется только в блоке <code>Limit</code> . Разрешает доступ к серверу. По умолчанию в ней задается значение <code>all</code> , которое разрешает доступ к серверу всем узлам со всех сетей
AllowAll	Разрешает доступ всем. Может использоваться в блоках Directory , Anonymous , Limit
AllowForeignAddress <i>on</i> <i>off</i>	Разрешает узлу при подключении к серверу указывать адрес, не принадлежащий ему. По умолчанию задается значение <code>off</code> (т. е. доступ запрещен), рекомендуется не изменять его. Директива может использоваться в блоках Anonymous , <Global>
AllowGroup список групп	Разрешает доступ к серверу указанным группам пользователей (группы должны быть зарегистрированы на этом сервере)
AllowOverwrite <i>on</i> <i>off</i>	Разрешает (<i>on</i>) перезаписывать существующие файлы

Таблица 34.1 (продолжение)

Директива	Описание
AllowUser список пользователей	Разрешает доступ к серверу указанным группам пользователей (пользователи должны быть зарегистрированы на этом сервере)
<Anonymous каталог>	Разрешает анонимный доступ к указанному каталогу. Указанный каталог будет корневым каталогом анонимного FTP-сервера
AuthGroupFile файл	Задаёт альтернативный файл групп. По умолчанию /etc/group
AuthUserFile файл	Задаёт альтернативный файл паролей. По умолчанию /etc/passwd
Bind IP-адрес	Выполняет привязку дополнительного адреса к FTP-серверу
DeferWelcome on off	Разрешает вывести приветствие после аутентификации (on) или до нее (off)
Deny from all узел сеть	Запрещает доступ к FTP-серверу. Используется в блоке Limit
DenyAll	Запрещает доступ всем к объектам, указанным в Directory, Anonymous, Limit
DenyUser список пользователей	Запрещает доступ указанным пользователям
DefaultRoot каталог	Определяет корневой каталог FTP-сервера. В качестве значения этого параметра полезно указать значение ~, тогда в качестве корневого каталога будет использоваться домашний каталог пользователя, который зашел на сервер
DisplayLogin файл	Указанный текстовый файл будет отображен, когда пользователь зайдет на сервер
DisplayFirstChdir файл	Отображает указанный файл при каждой смене каталога
<Directory каталог>	Задаёт параметры доступа к каталогу и его подкаталогам
<Global>	Задаёт глобальные параметры FTP-сервера
<Limit команда>	Накладывает ограничение на выполнение некоторых FTP-команд — например: READ, WRITE, STOR, LOGIN
MaxClients число сообщение	Максимальное количество одновременно работающих клиентов. Если указанное число будет превышено, FTP-сервер отобразит указанное сообщение
MaxLoginAttempts	Максимальное количество попыток регистрации на сервере. По умолчанию 3. Указывается в блоке Global
MaxInstances	Максимальное количество одновременно работающих экземпляров демона proftpd
ServerType тип	Задаёт тип запуска сервера. Значение по умолчанию — standalone (автономный запуск). Не нужно его изменять
ServerName "имя"	Задаёт имя сервера. Можете написать все, что угодно — например: My server

Таблица 34.1 (окончание)

Директива	Описание
ServerAdmin e-mail	Позволяет указать адрес электронной почты администратора сервера
ShowSymlinks on off	Разрешает показывать символические ссылки (on) или сразу результирующие файлы (off)
Order allow, deny deny, allow	Задаёт порядок выполнения директив Allow и Deny в блоке Limit
TimeoutIdle секунды	Определяет тайм-аут простоя. Если пользователь не проявит активности за указанное время, соединение будет разорвано. По умолчанию используется значение 60
TimeoutNoTransfer секунды	Тайм-аут начала передачи. Определяет, сколько времени нужно ждать до разъединения, если пользователь вошел, но не начал передачу
TimeoutStalled секунды	«Замирание» во время передачи файла. Бывает так, что клиент начал передачу (или прием) файла, но связь оборвалась. Этот тайм-аут определяет, сколько нужно ждать до разъединения в такой ситуации. Такой тайм-аут нужен, потому что бывает другая ситуация — когда у пользователя очень медленный канал
Umask маска	Задаёт права доступа для созданного файла
User имя пользователя	Пользователь, от имени которого работает демон ProFTPD

34.3. Настройка FTP-сервера

В этом разделе мы настроим реальный FTP-сервер, к которому смогут получить доступ как обычные (зарегистрированные) пользователи, так и анонимные.

Приведенная в листинге 34.1 конфигурация вполне работоспособна. Однако для создания *обычного* (не анонимного) FTP-сервера в этот конфигурационный файл нужно добавить две директивы:

```
DefaultRoot ~
MaxClients 20 "Server is full!!!"
```

Первая директива делает корневым домашний каталог пользователя (при этом пользователь не может выйти за пределы своего домашнего каталога — следовательно, он не в состоянии навредить системе, если администратор неправильно установил права доступа к каким-нибудь системным каталогам), а вторая — ограничивает число одновременно работающих клиентов во избежание перегрузки сервера. Остальные параметры вы можете задать по своему усмотрению.

Рассмотрим несколько примеров использования блоков **Directory** и **Login**:

```
<Directory upload>
  <Limit READ>
```

```

        DenyAll
    </Limit>
    <Limit WRITE>
        AllowAll
    </Limit>
</Directory>

```

Директива **Directory** определяет две директивы **Limit** для каталога **upload**: первая запрещает всем читать этот каталог, а вторая — разрешает всем записывать новые файлы в этот каталог. Каталог **upload**, таким образом, полностью оправдывает свое название — только для загрузки файлов.

А вот еще один пример, запрещающий доступ к серверу всех узлов из подсети 192.168.1.0:

```

<Limit LOGIN>
    DenyAll
    Deny from 192.168.1.
</Limit>

```

Если надо, наоборот, разрешить доступ к серверу только пользователям из сети 192.168.1.0, то нужно использовать следующий блок **Limit**:

```

<Limit LOGIN>
Order deny, allow                # порядок действия deny-allow
    DenyAll                      # запрещаем доступ всем
    Allow from 192.168.1.0        # разрешаем доступ только из сети
                                # 192.168.1.0
</Limit>

```

Для организации *анонимного* доступа на FTP-сервер нужно добавить в файл конфигурации следующую директиву **Anonymous**:

```

<Anonymous ~ftp>

```

```

User                ftp
Group               nogroup

```

```

# Определяем псевдоним "anonymous" для пользователя "ftp"
# Клиенты смогут войти под обоими именами

```

```

UserAlias           anonymousftp

```

```

# Все файлы принадлежат пользователю ftp
DirFakeUser on ftp
DirFakeGroup on ftp

```

```

# Не нужно требовать "правильную" оболочку
# "Правильной" считается оболочка, указанная в файле /etc/shells
RequireValidShell   off

```

```
# Максимальное число анонимных пользователей
MaxClients 10

# Файлы с сообщениями
DisplayLogin welcome.msg
DisplayFirstChdir .message

# Ограничим WRITE для анонимных пользователей
<Directory *>
    <Limit WRITE>
        DenyAll
    </Limit>
</Directory>

</Anonymous >
```

34.4. Оптимизация FTP-сервера

Оптимизировать ProFTPD можно по трем направлениям: ускорить авторизацию, равномерно распределить нагрузку на сервер и помочь серверу избежать перегрузки «узкого» канала.

Ускорить *авторизацию* ПОМОЖЕТ Отключение директив `IdentLookup` и `UseReverseDNS`. Первая управляет использованием протокола `ident`, но поскольку этот протокол давно не применяется, директиву можно безболезненно отключить. Вторая определяет доменное имя клиента по его IP-адресу, но это занимает некоторое время, поэтому для ускорения доступа к FTP-серверу ее также нужно отключить. Добавьте для этого в файл конфигурации `proftpd.conf` следующие строки:

```
IdentLookups off
UseReverseDNS off
```

К авторизации относится также и директива `MaxLoginAttempts`, задающая максимальное число попыток регистрации пользователя на сервере:

```
MaxLoginAttempts 3
```

Теперь приступим к *распределению нагрузки на сервер*. Первым делом нужно задать максимальное число клиентов:

```
MaxClients число
```

Понятно, что чем быстрее наш канал подключения к Интернету, тем больше клиентов сервер сможет принять.

С помощью директивы `MaxClientsPerHost` можно задать максимальное число клиентов, приходящих на сервер с одного узла:

```
MaxClientsPerHost число
```

Устанавливать для этой директивы значение 1 не следует. Представьте сеть, доступ к Интернету пользователей которой осуществляется через один сервер — шлюз. То

есть, вся эта сеть имеет только один реальный IP-адрес. Соответственно, и все пользователи такой сети выходят в Интернет под одним и тем же IP-адресом. Если установить параметр `MaxClientsPerHost` в 1, то из всей сети на наш FTP-сервер сможет зайти только один пользователь. Исходя из понимания, что все пользователи сети тоже не будут одновременно заходить на наш FTP-сервер, для директивы `MaxClientsPerHost` нужно установить чуть большее значение — например, 2 или 3.

Предположим также, что доступ к нашему FTP-серверу разрешен только зарегистрированным (а не анонимным) пользователям. Но некоторые пользователи могут «одолжить» свой логин и пароль другим, незарегистрированным на сервере пользователям, чтобы они тоже смогли одновременно с ними использовать ресурсы нашего сервера. Это не есть хорошо, и с помощью директивы `MaxClientsPerUser` мы можем задать максимальное количество FTP-клиентов от одного пользователя. Вот тут самое время установить значение 1:

```
MaxClientsPerUser 1
```

Но пользователи стараются нас обхитрить — они заходят под одним и тем же логином, но с разных узлов (например, из разных сетей). Делать это им тоже нужно запретить:

```
MaxClientsPerUser 1
```

Директива `MaxHostsPerUser`, как понятно из ее названия, ограничивает количество узлов на одного пользователя.

Нужно определить и директиву `MaxInstances`, задающую максимальное число параллельно запущенных экземпляров сервера `proftpd` (для обработки запросов каждого нового клиента запускается своя копия `proftpd`). Ее значение зависит от возможностей вашего сервера. Предположим, что для директивы `MaxClients` мы задали значение 10, т. е. одновременно могут работать 10 пользователей. Поскольку мы установили для `MaxClientsPerUser` и `MaxHostsPerUser` значение 1, то для `MaxInstances` можно установить значение 10. Но если мы разрешим использовать каждому пользователю более одного FTP-клиента или разрешим регистрироваться одновременно с разных узлов под одним и тем же логином, тогда нужно увеличить значение `MaxInstances`. Например, если для `MaxHostsPerUser` мы установили значение 2, то `MaxInstances` должен быть равен 20 (2x10). В общем, вам, учитывая три значения (`MaxClients`, `MaxClientsPerUser` и `MaxHostsPerUser`), следует высчитать максимальное значение `MaxInstances`, чтобы в моменты пиковой нагрузки все клиенты получили доступ к серверу:

```
Maxinstances 10
```

С помощью директивы `MaxLoginAttempts` можно задать, сколько раз пользователь может ввести пароль (после последней попытки сервер разорвет соединение). Рекомендуемое значение — 3.

`MaxRetrieveFileSize` — максимальный размер загружаемого файла. Эту величину можно не ограничивать, потому как размер файлов, загружаемых на сервер непосредственно вами, вы станете определять сами, а размер файлов, которые загружа-

ЮТ пользователи, будет ограничен С ПОМОЩЬЮ директивы `MaxStoreFileSize`, упомянутой далее. Так что, если никто не «зальет» на сервер файл размером, скажем, в 1 Гбайт, то никто не сможет и скачать такой файл.

`MaxStoreFileSize` — максимальный размер файла, загружаемого на сервер пользователями. Тут все зависит от «ширины» канала и места на диске — даже больше от второго, нежели от первого. Решайте сами.

Нам осталось *ограничить скорость передачи данных* — чтобы сервис FTP не узурпировал под себя весь трафик. Особенно это важно, если канал «узкий», и на сервере запущены другие сетевые сервисы, — например, тот же Apache.

Ограничить пропускную способность можно или с помощью устаревших директив `Rate*`, или с помощью новой `TransferRate`. Последнюю использовать удобно, если сервер подключен к Интернету по синхронному каналу, когда скорость приема равна скорости передачи, т. к. она одновременно ограничивает как скорость чтения, так и записи:

`TransferRate` байт-в-секунду

Если же сервер подключен по асинхронному каналу, т. е. скорости приема и передачи — разные, удобнее использовать директивы `Rate*`, потому что они могут по отдельности ограничить как скорость чтения, так и скорость записи:

- `RateReadBPS` байт-в-секунду — задает скорость чтения данных в байтах в секунду;
- `RateWriteBPS` байт-в-секунду — определяет максимальную скорость записи данных в байтах в секунду.

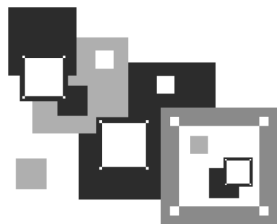
34.5. Программы `ftppwho` и `ftpcount`

Вспомогательные программы `ftppwho` и `ftpcount` помогут администратору FTP-сервера определить, какие пользователи в текущий момент зарегистрированы на сервере (`ftppwho`), и узнать общее число зарегистрированных на сервере в текущий момент пользователей (`ftpcount`). Вывод обеих программ показан на рис. 34.2.

```
den@den-desktop:~$ ftppwho
standalone FTP daemon [5176], up for 37 min
 7378 den      [ 0m10s]   0m7s idle
Service class                                -   1 user
den@den-desktop:~$ ftpcount
Master proftpd process 5176:
Service class                                -   1 user
den@den-desktop:~$
```

Рис. 34.2. Программы `ftppwho` и `ftpcount`

ГЛАВА 35



DNS-сервер

35.1. Еще раз о том, что такое DNS

Система доменных имен (DNS, Domain Name System) позволяет преобразовывать IP-адреса в доменные имена и обратно. Компьютеру намного проще работать с числами, человеку же легче запомнить символьное имя узла, чем его IP-адрес.

Система DNS имеет древовидную иерархическую структуру (рис. 35.1) — здесь мы видим корень системы DNS, домены первого уровня (ru, com, org) и домен второго уровня (firma). Доменов первого уровня (их еще называют TLD, Top Level Domains) довольно много: com, biz, org, info, gov, net, ws, домены стран (ru, ua, uk, ...) и т. д. Понятно, что доменов второго уровня еще больше, не говоря уже о доменах третьего и последующих уровней.

Список корневых серверов DNS хранится на каждом DNS-сервере (позже мы узнаем, где именно и как его обновлять).

Доменное имя компьютера имеет следующий формат:

[имя_компьютера] . [домен_ы] [домен.TLD]

Например,

ftp.sales.firma.ru

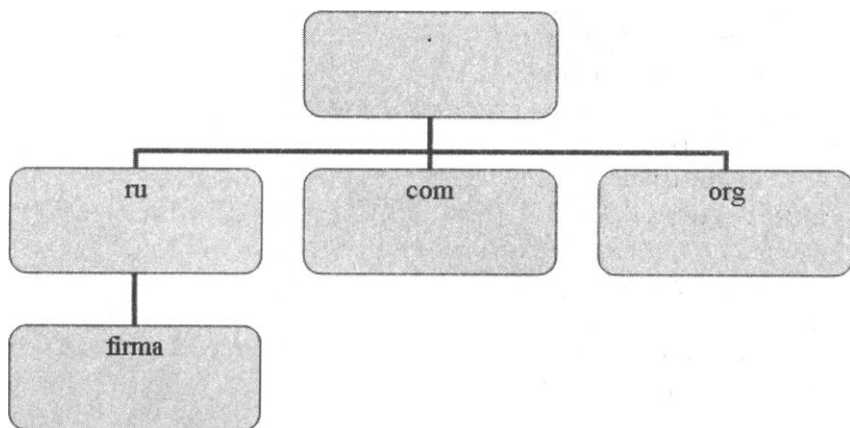


Рис. 35.1. Иерархическая структура DNS

При запросе к DNS-серверу доменное имя обрабатывается в доменном порядке. Сначала наш DNS-сервер посылает запрос к DNS-серверу домена ru: знает ли он что-нибудь о домене firma? DNS-сервер домена ru:, если домен firma найден, сообщает нам IP-адрес сервера DNS домена firma. Потом наш DNS-сервер (или наш собственный сервер имен, или же сервер имен провайдера) обращается к серверу имен домена firma.ru. Ему нужно уточнить, знает ли тот что-либо о домене sales. Получив IP-адрес DNS-сервера домена sales.firma.ru, мы можем к нему обратиться, чтобы получить IP-адрес компьютера с именем ftp.sales.firma.ru (очевидно, это FTP-сервер отдела продаж какой-то фирмы).

Приведенная схема разрешения доменного имени называется *рекурсивной*, а наш запрос — рекурсивным запросом. Конечно, саму эту схему я немного упростил, но общий смысл должен быть понятен. Ясно и то, что такой запрос занимает довольно много времени и ресурсов, поэтому целесообразно настроить кэширующий сервер DNS, даже если у вас нет собственного домена. Всю «грязную» работу (т. е. рекурсивные запросы) будут делать серверы DNS провайдера, а нашему серверу останется только кэшировать результаты запросов, — так можно повысить скорость разрешения доменных имен и, следовательно, ускорить работу Интернета в целом. Поэтому кэширующий сервер можно установить не только на шлюзе, но и на домашнем компьютере, где он также будет с успехом выполнять свою функцию.

Настройку сервера DNS мы начнем именно с кэширующего сервера DNS. Во-первых, он настраивается проще, чем полноценный сервер DNS, но зато в процессе его настройки мы познакомимся с основными конфигурационными файлами, и при настройке полноценного DNS-сервера нам будет проще. Во-вторых, не всегда есть необходимость настраивать полноценный DNS-сервер, — у вас может быть локальная сеть с выходом в Интернет, но она не обязательно должна иметь свой собственный домен.

35.2. Кэширующий сервер DNS

Наверняка все мы знакомы с так называемыми «ускорителями», или «оптимизаторами», Интернета — программами, якобы помогающими сделать Интернет намного быстрее. Как правило, это Windows-программы, распространяемые в Интернете за определенную плату. Впрочем, иногда их даже можно скачать бесплатно. В первом случае, если программа распространяется за деньги, «ускоритель» Интернета вообще ничего не делает, — пользователь его запускает, устанавливает параметры, но на самом деле никакого ускорения не происходит. Просто кто-то таким не очень честным образом зарабатывает деньги. Во втором случае, когда программа распространяется бесплатно, также не наблюдается никакого ускорения, а, наоборот, заметны падение скорости и повышенный расход трафика. Почему? Да потому что «оптимизаторы» Интернета в большинстве случаев являются вирусами-троянами. Пользователи добровольно устанавливают программу, которая потом передаст злоумышленнику секретную информацию (например, ключи от электронного кошелька). Помните, что бесплатный сыр — только в мышеловке.

А вот Linux позволяет организовать настоящий ускоритель Интернета. Впрочем, не нужно ожидать, что ваш Интернет станет работать на 70, а то и на все 100% быст-

рее, как это обещают оптимизаторы-вирусы. Ускорение обеспечит кэширующий сервер DNS. Установка такого сервера позволяет:

- сократить время разрешения доменных имен, поскольку в нашей сети заработает свой DNS-сервер— ответы на запросы о разрешении доменных имен будут приходить от локального сервера, а не от загруженного DNS-сервера провайдера;
- немного сэкономить трафик, поскольку локальный трафик не будет учитываться, чего не скажешь о трафике между вами (вашей сетью) и провайдером.

Итак, кэширующий DNS-сервер — дело нужное, поэтому не будем терять времени, установим пакет `bind9` и приступим к настройке сервера.

КЭШИРУЮЩИЙ DNS-СЕРВЕР NAMED

Пакет называется `bind9`, т. е. BIND (Berkley Internet Nameserver Deamon) версии 9, а сам сервер — `named`. В старых версиях дистрибутивов этот пакет называется просто `bind` и, скорее всего, содержит восьмую версию BIND.

Настройку кэширующего DNS-сервера мы рассмотрим на примере дистрибутива Debian, но в других дистрибутивах процесс настройки при условии использования девятой версии BIND должен осуществляться аналогично.

Основным файлом конфигурации сервера является файл `/etc/bind/named.conf`. Когда-то он был большим, но для удобства настройки его разделили на отдельные файлы, и сейчас в нем всего три строчки (листинг 35.1).

Листинг 35.1. Файл конфигурации `/etc/bind/named.conf`

```
include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.local";
include "/etc/bind/named.conf.default-zones";
```

Соответственно, в современной версии DNS-сервера нужно редактировать не основной файл конфигурации, а файлы, указанные в директивах `include`: первый файл содержит общие параметры DNS-сервера, во втором файле описаны локальные зоны, а зоны по умолчанию описаны в третьем файле. Локальная зона служит для преобразования имени `localhost` в IP-адрес `127.0.0.1` и наоборот, поэтому нас больше интересуют зоны по умолчанию (листинг 35.2).

Листинг 35.2. Файл `/etc/bind/named.conf.default-zones` (зоны по умолчанию)

```
// корневая зона, содержит корневые серверы имен
zone "." {
    type hint;
    file "/etc/bind/db.root";
};

// Зона localhost
zone "localhost" {
```

```
    type master;
    file "/etc/bind/db.local";
};

zone "127.in-addr.arpa" {
    type master;
    file "/etc/bind/db.127";
};

zone "0.in-addr.arpa" {
    type master;
    file "/etc/bind/db.0";
};

zone "255.in-addr.arpa" {
    type master;
    file "/etc/bind/db.255";
};
```

В основном в этом конфигурационном файле прописываются локальная зона и корневая, содержащая список корневых серверов DNS.

Вообще-то, собственные зоны вы можете описать и в файле `named.conf` — особой разницы нет. Но если ваш DNS-сервер описывает много зон или одну большую зону (где много компьютеров), тогда целесообразно вынести описание этих зон в отдельные файлы — вам так будет удобнее настраивать DNS-сервер.

Рассмотрим файл `/etc/bind/named.conf.options`, содержащий опции DNS-сервера (листинг 35.3). Оригинальный файл содержит комментарии на английском языке, а здесь мы покажем их на родном.

Листинг 35.3. Файл опций `/etc/bind/named.conf.options`

```
options {
    directory "/var/cache/bind";
    // Если "между" вашим DNS-сервером и форвард-серверами находится
    // брандмауэр, вы должны его настроить должным образом. О настройке
    // брандмауэра можно прочитать по адресу:
    // http://www.kb.cert.org/vuls/id/800113

    // Здесь прописываются форвард-серверы
    // forwarders {
    // 0.0.0.0;
    // };
    dnssec-validation auto;

    auth-nxdomain no;          # в соответствии с RFC1035
    listen-on-v6 { any; };
};
```

Основной рабочий каталог (`/var/cache/bind`) задается здесь параметром `directory`.

А вот далее начинается самое интересное. Напомню, что мы сейчас создаем кэширующий сервер, позволяющий ускорить процесс разрешения доменных имен. Но можно ускорить работу самого сервера, указав *форвард-серверы*. Дело в том, что в обычном режиме наш сервер сам формирует кэш, но так как сеть у нас относительно небольшая, кэш будет формироваться долго, а насколько долго — зависит от количества запросов, поступающих от клиентов сети. И если кэширующий сервер был установлен только для обслуживания своего компьютера, то вы сначала вообще не почувствуете никакой разницы, — ведь серверу, прежде чем добавить IP-адрес в кэш, нужно сначала его разрешить, и только при втором обращении к доменному имени его IP-адрес будет получен из кэша. Ускорение же на начальном этапе может быть обеспечено обращением к кэшу форвард-серверов. Как правило, в роли форвард-серверов выступают серверы провайдера, уже сформировавшие довольно большой кэш, которым мы и можем воспользоваться.

Все, что нужно для использования форвард-сервера, — это добавить его IP-адрес в блок `forwarders`:

```
forwarders {  
    # все запросы будут переадресованы к DNS-серверу  
    # провайдера 192.168.99.1  
    # если с этим сервером что-то случится, то локальный сервер  
    # попыбует найти ответ в своем кэше или обратиться к другим  
    # DNS-серверам, которые указываются в файле /etc/resolv.conf  
    192.168.99.1;  
};
```

Параметр `forwarders` задает заключенный в фигурные скобки список IP-адресов, соответствующих DNS-серверам, которым наш DNS-сервер будет переадресовывать запросы, вместо того, чтобы отвечать на них самому. IP-адреса перечисляются через точку с запятой. Адреса форвард-серверов обычно находятся в файле `/etc/resolv.conf`.

Кроме параметра `forwarders` можно использовать параметр `forward`, принимающий следующие значения:

- `only` — наш DNS-сервер никогда не должен предпринимать попыток обработать запрос самостоятельно;
- `first` — наш сервер должен пытаться сам обработать запрос, если указанные далее параметром `forwarders` серверы DNS не были найдены.

Использование параметра `forward` лишено смысла без параметра `forwarders`, и указывать параметр `forward` обычно нужно до параметра `forwarders`:

```
forward first;  
forwarders {  
    192.168.99.1;  
    192.168.99.2;  
};
```

Вот, вроде бы, и все — можно приступить к запуску сервера. Надо лишь отметить, что поскольку мы создавали кэширующий сервер, в его конфигурационном файле отсутствует блок `controls {}`. Пустой или отсутствующий блок `controls {}` нужен для того, чтобы сервер `named` не обращал внимания на отсутствие ключа `mdc.key`, требуемого для программы удаленного управления сервером `rndc`. Такой прием, правда, не вполне корректен, поскольку для останова сервера нам придется использовать команду `killall named`, но для нас это не существенно, поскольку мы не будем часто его останавливать.

Теперь можно запустить наш сервер имен:

```
sudo service bind9 start
```

Впрочем, поскольку сервер может быть уже запущен (при установке пакета он запускается автоматически), то после изменения конфигурации его следует перезапустить:

```
sudo service bind9 restart
```

Если в конфигурационных файлах нет ошибок, вы получите сообщение:

Starting domain name service... bind9 [OK]

Сам сервис носит название `bind9` (если помните, по сути, сервис — это сценарий), но процесс DNS-сервера называется `named`, поэтому, когда в файле `/var/log/messages` (или в `/var/log/daemon.log`) вы станете искать сообщения, связанные с DNS-сервером, ищите строчку `named`, а не `bind9`:

```
Aug 8 9:58:16 den named[3140]: starting BIND 9.2.3
Aug 8 9:58:16 den named[3140]: using 1 CPU
Aug 8 9:58:16 den named[3140]: loading configuration from '/etc/bind/named.conf'
Aug 8 9:58:16 den named[3140]: listening on IPv4 interface lo, 127.0.0.1#53
Aug 8 9:58:16 den named[3140]: listening on IPv4 interface eth0, 192.168.0.1#53
Aug 8 9:58:16 den named[3140]: zone 0.0.127.in-addr.arpa/IN: loaded serial
1997022700
Aug 8 9:58:16 den named[3140]: running
```

Кстати, этот вывод здесь приведен не просто так — последняя строка свидетельствует о том, что сервер запущен. Первая же запись сообщает нам версию BIND, вторая — то, что используется один процессор, далее указываются: используемый конфигурационный файл, прослушиваемые интерфейсы (`lo` и `eth0`) и порт — 53, а также загруженная локальная зона. Число в квадратных скобках (3140) — это PID процесса (идентификатор процесса), «убить» процесс в данном случае можно так:

```
# kill 3140
```

Проверить, работает ли сервер, можно и другим способом — например:

```
# ps -ax | grep named
# ps -ax | grep bind9
```

Теперь осталось в файле `/etc/resolv.conf` прописать IP-адрес собственного сервера DNS. То же самое нужно сделать на всех остальных компьютерах сети:

```
domain firms.ru
# IP адрес или 127.0.0.1
nameserver 127.0.0.1
# или IP-адрес DNS-сервера — для остальных компьютеров сети
nameserver 10.0.0.1
```

А чтобы NetworkManager не перезаписал содержимое этого файла, следует запретить его редактирование:

```
sudo chattr +i /etc/resolv.conf
```

Протестировать настройки сервера можно с помощью команды `ns lookup`:

```
# nslookup yandex.ru
Server: localhost.firma.ru
Address: 127.0.0.1
Non-authoritative answer:
Name:   yandex.ru
Address: 213.180.216.200
```

Если вы получили подобный ответ, то это означает, что наш сервер работает нормально. Обратите внимание, что ответ пришел не от DNS-сервера провайдера, а от нашего локального сервера.

ПРОБЛЕМА С ПЕРЕЗАПИСЬЮ ФАЙЛА RESOLV.CONF В UBUNTU

В Ubuntu есть небольшая проблема с перезаписью файла `resolv.conf`. Несмотря на то, что вы его перезапишете, при установке соединения или при перезагрузке он будет возвращен в исходное состояние. Можно было бы, конечно, запретить изменение файла с помощью команды `chattr`, однако я докопался до истины. В книге весь процесс для экономии места мы рассматривать не станем, но все желающие могут узнать о моей борьбе с Ubuntu по адресу:

<http://www.dkws.org.ua/index.php?page=show&file=a/ubuntu/static-dns-ubuntu9>.

35.3. Полноценный DNS-сервер

Теперь можно перейти к настройке полноценного сервера DNS, если, конечно, он вам нужен. Но сначала определимся, что такое *зона*, поскольку полноценный DNS-сервер обслуживает одну или несколько зон. Ошибочно считать зоной обслуживаемый домен — это не так. Домен — это группа компьютеров с одинаковой правой частью доменного имени. Пусть у нас есть домен `firma.ru`. Компания, которой принадлежит этот домен, довольно большая, поэтому для каждого подразделения пришлось организовать свой домен: `sales.firma.ru`, `dev.firma.ru`, `orders.firma.ru` и т. д. Для управления всем доменом `firma.ru` (и всеми его поддоменами) мы можем использовать или один-единственный DNS-сервер, или же создать независимые серверы для каждого поддомена (или только для некоторых поддоменов). Например, основной сервер будет обслуживать только домены `firma.ru` и `sales.firma.ru`, а дополнительный сервер — домены `dev.firma.ru` и `orders.firma.ru`. Домены `firma.ru` и `sales.firma.ru` образуют в этом случае одну зону, а домены `dev.firma.ru` и `orders.firma.ru` — другую. Иными словами, зона —

это часть домена, управляемая определенным DNS-сервером. Зона, которая содержит домены низшего уровня, называется *подчиненной зоной* (subordinate zone).

Итак, прежде всего нам нужно настроить удаленное управление сервером, а именно добавить секцию `controls`, которая отсутствует у нас в предыдущем примере. Выполните команду:

```
# /usr/sbin/rndc-confgen > rndc.conf
```

Откройте файл `rndc.conf` в любом текстовом редакторе — нам нужно выделить и скопировать две директивы: `controls` и `key`:

```
key "rndc-key" {
    algorithm hmac-md5;
    secret "ключ";
};
controls {
# разрешаем "удаленное" управление только с локального компьютера
    inet 127.0.0.1 port 953
    allow { 127.0.0.1; } keys { "rndc-key"; };
};
```

Скопированный блок текста следует вставить в самое начало файла `named.conf`. Понятно, что из него нужно удалить пустую директиву `controls`, если она там есть.

При настройке кэширующего сервера DNS мы в его конфигурационном файле описали две зоны: корневую и локальную. Теперь нам нужно описать еще две зоны: прямого и обратного преобразования, которые и будут обслуживать наш домен. Добавьте в файл конфигурации `named.conf.local` (в конец файла) или в сам `named.conf` (тоже в конец файла) строки:

```
zone "firma.ru" {
    type master;
    file "firma.ru";
    notify no;
};

zone "1.0.0.10.in-addr.arpa" {
    type master;
    file "10.0.0.1";
    notify yes;
}
```

Файл `firma.ru` (он должен находиться в каталоге, заданном директивой `directory`) служит для прямого преобразования, т. е. для преобразования доменных имен в IP-адреса. В листинге 35.4 представлен пример этого файла.

Листинг 35.4. Пример файла прямого преобразования

```
@          IN SOA      server.firma.ru. hostmaster.firma.ru. (
                20040603      ; серийный номер (можно узнать в
                                ; файлах с примерами)
```



```

        3600          ; обновление каждый час
        3600          ; повтор каждый час
        3600000       ; время хранения информации 1000 часов
        3600          ; TTL записи
)
    IN NS             server.fima.ru.
    IN A              10.0.0.1
    IN MX             100 server.fima.ru.
www    IN CNAME       server.fima.ru.
ftp    IN CNAME       server.fima.ru.
mail   IN CNAME       server.fima.ru.
c2     IN A           10.0.0.2
c3     IN A           10.0.0.2
localhost. IN A      127.0.0.1

```

Прежде всего обратите внимание, что в конце каждого доменного имени ставится точка — это для того, чтобы сервер не приписывал имя домена (*firma.ru*) к доменному имени. Если имя домена писать лень, можно просто указывать имя компьютера (*server* вместо *server.firma.ru*), но не ставить точку в конце доменного имени.

Разберемся с записью *IN SOA*. Она описывает начало полномочий (Start Of Authority, SOA). Первое имя после *SOA* — это имя компьютера, на котором запущен DNS-сервер. В нашем случае — это *server.firma.ru*. Затем следует e-mail администратора сервера, но поскольку символ *@* зарезервирован, вместо него ставится точка. Остальные элементы записи *SOA* прокомментированы в листинге.

Запись *NS* (*IN NS*) задает имя сервера доменных имен, а запись *A* — его IP-адрес. Запись *MX* служит для задания почтового сервера. Как мы видим, в роли почтового сервера используется все тот же наш *server.fima.ru*. Запись *100* — это приоритет почтового сервера. Приоритет используется, если указано два (или более) почтовых сервера. Чем меньше число, тем выше приоритет:

```

IN MX 100 mail1
IN MX 150 mail2

```

С помощью записи *CNAME* определяются канонические имена, т. е. псевдонимы. Как мы видим, к нашему серверу *server.fima.com* можно обратиться по следующим именам: *www.fima.ru*, *ftp.firma.ru*, *mail.firm.ru*.

Далее описаны два компьютера: *c2.fima.ru* (мы не ставили точку после *c2*, поэтому *fima.ru* сервер «допишет» автоматически) и *c3.fima.ru*, с IP-адресами *10.0.0.2* и *10.0.0.3* соответственно.

Последняя запись — это определение имени *localhost*, желательно не забыть о нем.

Теперь пора приступить к рассмотрению файла обратного соответствия, который представлен в листинге 35.5. Напомню, что этот файл используется для преобразования IP-адресов в доменные имена.

Листинг 35.5. Пример файла обратного преобразования /etc/bin9d/10.0.0.1

```

@           IN SOA      server.firma.ru. hostmaster.firma.ru. (
                20040603      ; серийный номер (можно узнать в файлах
                                ; с примерами)
                3600           ; обновление каждый час
                3600           ; повтор каждый час
                3600000        ; время хранения информации 1000 часов
                3600           ; TTL записи
)
0           IN NS       server.firma.ru
1           IN PTR      server.firmaru
2           IN PTR      c2.firma.ru
3           IN PTR      c3.firma.ru

```

В этом файле, если вы успели заметить, можно полностью не приводить IP-адрес, но доменное имя следует указывать полностью (точки в конце доменного имени не нужны). Если же вам хочется привести IP-адрес полностью, тогда указать его необходимо в обратном порядке, — например:

```
2.0.0.10    IN PTR c2.finna.ru
```

Вот, практически, и все. Можно в целях защиты сервера добавить в блок options конфигурационного файла **named.conf.options** директиву **allow-query**:

```

allow-query {
10.0.0.0/24;
localhost;
}

```

Блок **allow-query** разрешает запросы к серверу только узлам подсети 10.0.0.0 и от узла **localhost** — узлы других подсетей не смогут использовать наш сервер. И когда вы настраиваете DNS-сервер, который будет работать в локальной сети (обслуживать только клиентов нашей локальной сети), то, по большому счету, блок **allow-query** вам не нужен. Однако при настройке DNS-сервера провайдера или же сервера, работающего в сети с реальными IP-адресами, директива **allow-query** просто необходима, чтобы «чужие» узлы не смогли использовать наш сервер.

Полный файл конфигурации полноценного DNS-сервера для домена **firma.ru** представлен в листинге 35.6. Описание зон и опций я не выносил в файлы **named.conf.options** и **named.conf.local** для наглядности — чтобы вы в одном листинге увидели все настройки сервера. Однако на практике я не рекомендую хранить все настройки в одном файле, хотя это и не запрещено.

Листинг 35.6. Полная версия файла конфигурации

```

key "rndc-key" {
    algorithm hmac-md5 ;
    secret "ключ";
};

```

```
controls {
    inet 127.0.0.1 port 953
        allow { 127.0.0.1; } keys { "rndc-key"; };
};

options {
    directory "/etc/bind";

allow-query {
    10.0.0.0/24;
    localhost;
}

};

zone "." in {
    type hint;
    file "db.root";
};
zone "0.0.127.in-addr.arpa" in {
    type master;
    file "db.127";
};
zone "localhost" {
    type master;
    file "db.local";
};
zone "255.in-addr.arpa" {
    type master;
    file "db.255";
};

zone "firma.ru" {
    type master;
    file "firma.ru";
    notify no;
};

zone "1.0.0.10.in-addr.arpa" {
    type master;
    file "10.0.0.1";
    notify yes;
}
```

После настройки сервер нужно перезапустить:

```
# service named restart
```

35.4. Вторичный DNS-сервер

В идеале для поддержки домена должно быть выделено два сервера: первичный и вторичный. Вторичный используется для подстраховки, если вдруг с первичным что-то случится (например, банальная перезагрузка администратором).

Вторичный сервер DNS описывается аналогично первичному, но зона домена указывается несколько иначе:

```
zone "firma.m" {
    type slave;
    file "firma.ru";
    masters { 10.0.0.1; };
};
```

Как видим, устанавливается тип сервера— подчиненный (slave), а в блоке masters описываются первичные серверы (у нас он один).

В файл конфигурации первичного сервера нужно добавить директиву `allow-transfer`, в которой следует указать DNS-серверы, которым разрешен трансфер зоны, т. е. все вторичные серверы:

```
options {
    allow-transfer { 10.0.0.2; };
}
```

35.5. Обновление базы данных корневых серверов

Чтобы база данных корневых серверов всегда была актуальной, ее нужно регулярно обновлять. Получить ее можно по адресу **ftp://ftp.internic.net/domain/named.root**, а обновить — с помощью трех команд:

```
wget ftp://ftp.internic.net/domain/named.root
sudo cp named.root /etc/bind/db.root
sudo service bind9 restart
```

В листинге 35.7 содержится самая актуальная на момент написания этих строк версия файла `named.root`.

Листинг 35.7. Файл `named.root` (`db.root`)

```
;      This file holds the information on root name servers needed to
;      initialize cache of Internet domain name servers
;      (e.g. reference this file in the "cache . <file>"
;      configuration file of BIND domain name servers).

;
;      This file is made available by InterNIC
;      under anonymous FTP as
;      file                /domain/named.cache
```

```

;           on server           FTP.INTERNIC.NET
;      -OR-                     RS.INTERNIC.NET
;
;      last update:   May 23, 2015
;      related version of root zone: 2015052300

; formerly NS.INTERNIC.NET
;
;                               3600000      NS      A.ROOT-SERVERS.NET.
A. ROOT-SERVERS.NET. 3600000      A      198.41.0.4
A.   ROOT-SERVERS.NET. 3600000      AAAA  2001:503:ba3e::2:30
;
; FORMERLY NS1.ISI.EDU
;
;                               3600000      NS      B.ROOT-SERVERS.NET.
B.   ROOT-SERVERS.NET.3600000      A      192.228.79.201
B.   ROOT-SERVERS.NET.3600000      AAAA  2001:500:84::b
;
; FORMERLY C.PSI.NET
;
;                               3600000      NS      C.ROOT-SERVERS.NET.
C. ROOT-SERVERS.NET. 3600000      A      192.33.4.12
C. ROOT-SERVERS.NET. 3600000      AAAA  2001:500:2::c
;
; FORMERLY TERP.UMD.EDU
;
;                               3600000      NS      D.ROOT-SERVERS.NET.
D.   ROOT-SERVERS.NET.3600000      A      199.7.91.13
D. ROOT-SERVERS.NET. 3600000      AAAA  2001:500:2d::d
;
; FORMERLY NS.NASA.GOV
;
;                               3600000      NS      E.ROOT-SERVERS.NET.
E. ROOT-SERVERS.NET. 3600000      A      192.203.230.10
;
; FORMERLY NS.ISC.ORG
;
;                               3600000      NS      F.ROOT-SERVERS.NET.
F.   ROOT-SERVERS.NET.3600000      A      192.5.5.241
F.   ROOT-SERVERS.NET.3600000      AAAA  2001:500:2f::f
;
; FORMERLY NS.NIC.DDN.MIL
;
;                               3600000      NS      G.ROOT-SERVERS.NET.
G.   ROOT-SERVERS.NET.3600000      A      192.112.36.4
;

```

```

; FORMERLY AOS.ARL.ARMY.MIL
;
                                3600000      NS      H.ROOT-SERVERS.NET.
H.ROOT-SERVERS.NET. 3600000      A        128.63.2.53
H. ROOT-SERVERS.NET. 3600000      AAAA     2001:500:1::803f:235
'
; FORMERLY NIC.NORDU.NET

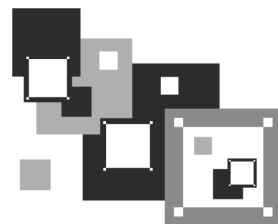
                                3600000      NS      I.ROOT-SERVERS.NET.
I. ROOT-SERVERS.NET. 3600000      A        192.36.148.17
I. ROOT-SERVERS.NET. 3600000      AAAA     2001:7fe::53
;
; OPERATED BY VERISIGN, INC.
'
                                3600000      NS      J.ROOT-SERVERS.NET.
J. ROOT-SERVERS.NET. 3600000      A        192.58.128.30
J. ROOT-SERVERS.NET. 3600000      AAAA     2001:503:c27::2:30

; OPERATED BY RIPE NCC
;
                                3600000      NS      K.ROOT-SERVERS.NET.
K. ROOT-SERVERS.NET. 3600000      A        193.0.14.129
K. ROOT-SERVERS.NET. 3600000      AAAA     2001:7fd::1
;
; OPERATED BY ICANN
;
                                3600000      NS      L.ROOT-SERVERS.NET.
L. ROOT-SERVERS.NET. 3600000      A        199.7.83.42
L. ROOT-SERVERS.NET. 3600000      AAAA     2001:500:3::42
'
; OPERATED BY WIDE

                                3600000      NS      M.ROOT-SERVERS.NET.
M. ROOT-SERVERS.NET. 3600000      A        202.12.27.33
M.ROOT-SERVERS.NET. 3600000      AAAA     2001:dc3::35
; End of file

```

ГЛАВА 36



Прокси-сервер: Squid и squidGuard

36.1. Зачем нужен прокси-сервер в локальной сети?

С помощью прокси-сервера можно очень эффективно управлять ресурсами своей сети — например, кэшировать трафик (HTTP), «обрезать» баннеры, указывать, какие файлы можно скачивать пользователям, а какие — нет, задавать максимальный объем передаваемого объекта и даже ограничивать пропускную способность пользователей определенного класса.

Основная функция прокси-сервера — это кэширование трафика. Если в сети используется прокси-сервер, можно сократить кэш браузеров клиентов практически до нуля — он уже не будет нужен, поскольку кэширование станет выполнять прокси-сервер. Тем более, что он выполняет кэширование всех клиентов сети, и уже запрошенные ранее кем-либо страницы будут доступны другим пользователям. Это означает, что если кто-то зашел на сайт **firma.ru**, то у всех остальных пользователей сети этот сайт будет открываться практически мгновенно, потому что он уже кэширован.

Даже если у вас всего один компьютер, все равно есть смысл использовать прокси-сервер, — хотя бы для того, чтобы «обрезать» баннеры, — так можно сэкономить на трафике, да и страницы начнут открываться быстрее, потому что многочисленные баннеры загружаться перестанут.

36.2. Базовая настройка Squid

Прокси-сервер Squid не сложен в настройке — во всяком случае, он не сложнее Samba и подобных сетевых сервисов. Для его установки нужно установить пакет squid, после чего у вас в системе появится новый сервис — squid.

Основной конфигурационный файл прокси-сервера Squid — **squid.conf**, он находится в каталоге **/etc/squid3/** или — для старых версий — в каталоге **/etc/squid/**. По умолчанию файл **squid.conf** не короче экватора Земли — ведь в нем перечислены все возможные директивы со всеми возможными параметрами. Понятное дело, все это

снабжено подробными комментариями. По сути, лучший справочник по директивам Squid — это его файл конфигурации. Однако не все пользователи владеют английским языком, и в листинге 36.1 приведена рабочая конфигурация этого файла, из которой удалено все лишнее.

Листинг 36.1. Файл `/etc/squid/squid.conf`

```
# порт для прослушивания запросов клиентов
# задается в формате http_port <порт> или http_port <узел>:<порт>
# последний случай подходит, если SQUID запущен на машине с несколькими
# сетевыми интерфейсами
http_port 192.168.0.1:3128

# адрес прокси провайдера, нужно уточнить у провайдера
# cach_peer proxy.your_isp.com

# объем оперативки в байтах, который будет использоваться прокси-сервером
# (85 Мбайт), - не устанавливайте более трети физического объема оперативки;
# если данная машина должна использоваться еще для чего-либо,
# можно задать в мегабайтах, но тогда между числом и мегабайтами (MB)
# обязательно должен быть пробел: cache_mem 85 MB
# в Squid3 значение по умолчанию для этого параметра - 256 Мб. Вряд ли стоит
# уменьшать это значение, а перед тем как увеличить его, подумайте,
# хватит ли у вас оперативной памяти. 10 клиентов по 256 Мб - это уже 2.5 Гб
cache_mem 256 MB
# где будет размещен кэш.
# первое число - это размер кэша в мегабайтах, не устанавливайте кэш на весь
# раздел; если нужно, чтобы он занимал весь раздел, отнимите от размера
# раздела 20% и укажите это значение. Например, если раздел 1024 Мбайт,
# то для кэша - только 820 Мбайт; второе число - количество каталогов первого
# уровня; третье - к-во каталогов второго уровня
cache_dir /usr/local/squid 1024 16 256

# максимальный размер кэшируемого объекта
# если размер объекта превышает указанный здесь, то объект не будет
# сохранен на диске
# maximum_object_size 4096 KB

# хосты, с которых разрешен доступ к прокси
acl allowed_hosts src 192.168.1.0/255.255.255.0
acl localhost src 127.0.0.1/255.255.255.255
# разрешенные порты:
acl allow_ports port 80                # http
acl allow_ports port 21                # ftp
# SSL-порты
acl SSL_ports port 443 563
```



```
# запрещаем все порты, кроме указанных в allow_ports
http_access deny !allow_ports

# запрещаем метод CONNECT для всех портов, кроме указанных в
# acl SSL_ports:
http_access deny CONNECT !SSL_ports

# запретим доступ всем, кроме тех, кому можно
http_access allow localhost
http_access allow allowed_hosts
http_access allow SSL_ports
http_access deny all

# пропишем пользователей, которым разрешено пользоваться squid
# (ppt, admin) :
ident_lookup on
acl allowed_users ppt admin
http_access allow allowed_users
http_access deny all
```

Базовый конфигурационный файл с успехом выполняет только функцию кэширования, а в следующем разделе мы поговорим о более тонкой настройке Squid.

36.3. Практические примеры

36.3.1. Управление доступом

Управление доступом осуществляется с помощью ACL (Access Control List) — списков управления доступом.

Чтобы разобраться, как работать с ACL, создадим список AllowedPorts:

```
acl AllowedPorts port 80 8080 3128
```

Имя списка — AllowedPorts, тип списка — port. Далее мы можем использовать этот список в http access для разрешения/запрещения указанных портов:

```
http_access allow AllowedPorts # разрешение портов
http_access deny AllowedPorts# запрещение портов
```

Кроме типа port часто используются следующие типы списков:

- proto — протокол (HTTP или FTP);
- method — метод передачи данных (GET или POST);
- src — IP-адреса (или диапазоны адресов) клиентов;
- dst — IP-адреса/URL сайтов, к которым обращаются клиенты.

Вы также можете создать список узлов, которым разрешен доступ к прокси:

```
acl allowed_hosts src "/etc/squid/allowed-hosts.txt"
```

Сам файл `/etc/squid/allowed-hosts.txt` будет выглядеть так:

```
# den
192.168.0. 2/255.255.255.255
# admin
192.168.0. 3/255.255.255.255
```

Отдельный файл использовать удобнее, чтобы не «засорять» основной конфигурационный файл. Обратите внимание — права доступа к файлу `allowed-hosts.txt` должны быть такие же, как и к файлу `squid.conf`.

36.3.2. Создание «черного» списка адресов

Теперь попробуем создать «черный» список интернет-адресов (URL):

```
acl blacklist url_regex adult
http_access deny blaklist
http_access allow all
```

Этот «черный» список не пропускает адреса (URL), содержащие слово `adult`. По аналогии можно было бы создать отдельный файл и записать в него все «плохие» URL (но это довольно накладно, проще использовать регулярные выражения).

36.3.3. Отказ от баннеров

С помощью ACL можно отказаться и от баннеров — принцип тот же. Для этого добавьте в файл конфигурации следующие ACL:

```
acl banners urlpath_regex "/etc/squid/banners.txt"
http_access deny banners
```

В файл `banners.txt` нужно внести URL баннерных сетей, например:

```
^http://www.clickhere.ru
^http://banner.kiev.ua
```

Создание этого файла пусть будет вашим домашним заданием — все равно все баннерные сети в книге не приведешь. Большого эффекта можно добиться, применив прокси-сервер SquidGuard, использующий уже готовые базы.

36.4. Управление прокси-сервером squid

Для запуска, перезапуска и остановки прокси-сервера нужно использовать следующие команды:

```
sudo service squid3 start
sudo service squid3 restart
sudo service squid3 stop
```

36.5. Настройка клиентов

Все браузеры на компьютерах вашей сети нужно настроить на использование порта 3128 (именно этот порт мы установили в конфигурационном файле). На рис. 36.1 изображена настройка браузера Opera.

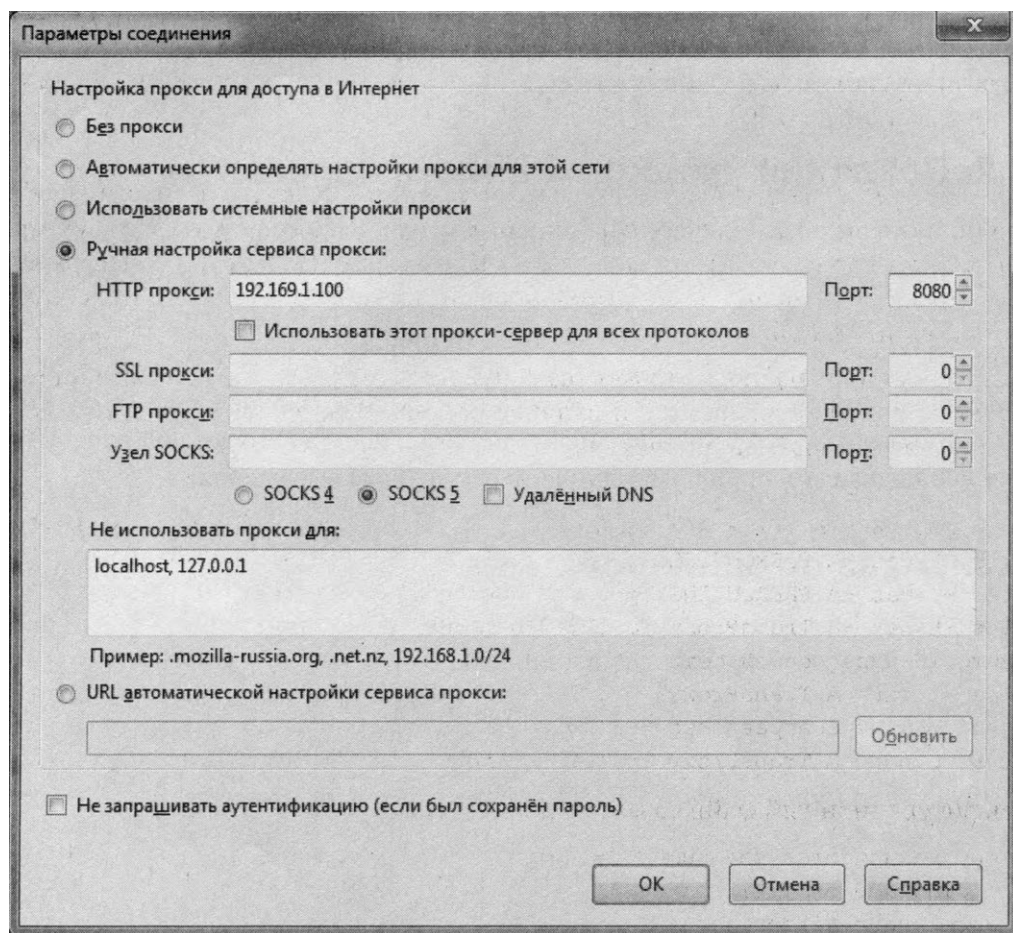


Рис. 36.1. Настройка клиента

36.6. Прозрачный прокси-сервер

С прокси-сервером часто связаны две проблемы. Первая заключается в том, что для работы через прокси-сервер нужно настраивать всех клиентов. Если сеть большая, скажем, 100 компьютеров, можете себе представить, сколько это займет времени, — ведь нужно подойти к каждому компьютеру. Даже если на настройку одного компьютера потребуется 5 минут, то всего их понадобится 500 — целый рабочий день. Но настройкой браузера может дело и не ограничиться. Ведь у пользователей могут быть и другие интернет-программы, работающие с WWW/FTP, которые также нужно будет настроить.

Проблема настройки — не самая страшная. Понятно, что если в сети организации 100 или более компьютеров, то и администратор в такой организации будет не один. А вдвоем-втроем можно настроить все 100 компьютеров за 2-3 часа.

Вторая проблема — более серьезная. Представим, что в сети у нас есть «продвинутые» пользователи (а они-таки есть), которые знают, для чего служит прокси-сервер. Они могут легко изменить его настройки и вместо работы через прокси использовать прямое соединение с Интернетом, т. е. работать в обход Squid. Вы так старались, создавая список «черных» URL (преимущественно это сайты для взрослых и всевозможные чаты/форумы), а они с помощью пары щелчков мыши свели все ваши старания к нулю.

Обе проблемы можно решить, если настроить *прозрачный* прокси-сервер, — пользователи даже не будут подозревать, что он есть. Во-первых, это решит проблемы с настройкой — вам не нужно настраивать браузеры пользователей, потому что все HTTP-запросы будут автоматически поступать на прокси-сервер. Во-вторых, прозрачный прокси обеспечит принудительное кэширование информации и, соответственно, принудительный контроль за страницами, которые посещают пользователи.

Для настройки прозрачного прокси вам нужно изменить как конфигурационный файл самого прокси-сервера, так и правила брандмауэра iptables (см. главу 31). Вот нужные для этого правила iptables:

```
iptables -t nat -new-chain TransProxy
# только порт 80 (HTTP) и 443 (SSL, https) — остальные обрабатывать # не будем
iptables -t nat-PREROUTING -p tcp -dport 80 -j TransProxy
iptables -t nat-PREROUTING -p tcp -dport 443 -j TransProxy
iptables -t nat-ATransProxy -d 127.0.0.1/8 -j ACCEPT
# укажите IP-адрес своей сети
iptables -t nat -A TransProxy -d. 192.168.1.0/24 -j ACCEPT
# все запросы перенаправляются на прокси-сервер 192.168.1.1, порт 3128
iptables -t nat -A TransProxy -p TCP -j DNAT -to 192.168.1.1:3128
```

А в конфигурационный файл **squid.conf** добавьте следующие директивы:

```
# серверу назначается реальный IP-адрес, его и нужно указать
tcp_outgoing_address ваш_реальный_IP
httpd_accel_host virtual
httpd_accel_with_proxy on
httpd_accel_uses_host_header on
```

Напомню, что iptables обычно устанавливается на шлюзе — компьютере, который предоставляет доступ к Интернету другим компьютерам сети. На этом же компьютере должен быть установлен и Squid.

36.7. squidGuard — ваше дополнительное «оружие»

Немного ранее в этой главе мы попытались для экономии трафика заблокировать все баннерные сети. Понятно, что у нас ничего не вышло, поскольку таких сетей много. Да и кроме баннерных сетей существует много информации, которую не

стоит принимать, — например, информацию порнографического содержания, рекламу, информацию о наркотиках и т. д.

Для эффективной защиты вашего трафика (а также для его экономии) лучше использовать squidGuard— дополнение к прокси-серверу Squid, содержащее базу данных запрещенного контента. И вам не придется самому заполнять эту базу данных— она уже разработана за вас. Так что, все, что требуется, — это установить squidGuard. Стандартная база squidGuard охватывает сайты, посвященные наркотикам, порно, насилию, азартным играм, а также рекламу. Закрыв доступ ко всему этому, можно сэкономить немало трафика. Поскольку squidGuard — это не отдельный сетевой сервис, а, как уже было отмечено, дополнение к прокси-серверу Squid, squidGuard не может работать без Squid.

Итак, squidGuard — штука нужная, поэтому установите пакет squidguard и отредактируйте его файл конфигурации— /etc/squidguard/squidGuard.conf. В листинге 35.2 приведен пример такого файла, и вам нужно изменить его «под себя».

Листинг 36.2. Пример файла /etc/squid/squidGuard.conf

```
# Путь к базе данных, x.x.x — номер версии squidGuard
dbhome /usr/lib/squidguard/db
logdir /var/log/squidGuard

# Дни и время работы
# s = Вс, m = Пн, t =Вт, w = Ср, h = Чт, f = Пт, a = Сб

time workhours {
    weekly s 10:00-13:00
    weekly m 08:00-13:00 14:00-18:00
    weekly t 08:00-13:00 14:00-18:00
    weekly w 08:00-13:00 14:00-18:00
    weekly h 08:00-13:00 14:00-18:00
    weekly f 08:00-13:00 14:00-18:00
    weekly a 09:20-13:00
}

# Наша сеть

# пользователи сети
src users {
ip 10.0.0.1-10.0.0.100
}

# демилитаризованная зона (внутренние серверы сети)
src dmz {
ip 10.0.1.1-10.0.1.10
}
```

```
# далее описываются базы запрещенного контента

# файл конфигурации я сократил, ведь у вас все равно есть полная
# версия, мы только рассмотрим пример описания одной базы –
# базы рекламы
dest advertising {
    domainlist advertising/domains
    urllist          advertising/urls

# вместо рекламы будет отображен файл nulbanner.png,
# размещенный на локальном Web-сервере 0x0
    redirect http://127.0.0.1/cgi-bin/nulbanner.png
}

# Списки доступа, т. е. кто и что может делать в нашей сети
acl {
# компьютерам из зоны DMZ разрешим любой контент, кроме рекламы
    dmz {

# управлять контентом можно с помощью директивы pass
# в качестве значений можно передать название базы,
# например, advertising – реклама, porn – порно и т. д. # (базы описаны
# выше)
# значение all означает весь контент, а попе – наоборот all, т. е.
# будет запрещен любой контент. Значение попе используется редко.
# Чаще используется выражение !база, например, !porn запрещает
# порнографию

        pass !advertising all

# Все запрещенные запросы будут передаваться
# на сценарий http://127.0.0.1/cgi-bin/squidGuard.cgi
        redirect http://127.0.0.1/cgi-
bin/blocked.cgi?clientaddr=%a&srcclass=%s&targetclass=%t&url=%u
    }

# Обычные пользователи сети
    users {
        # запрещаем весь ненужный контент
        pass !adult !audio-video !forums !hacking !redirector !warez
        lads !aggressive !drugs !gambling !publicite {violence !banneddestination
        !advertising all
        redirect http://127.0.0.1/cgi-
bin/blocked.cgi?clientaddr=%a&srcclass=%s&targetclass=%t&url=%u
    }

# Значение по умолчанию. Все запрещено, запросы перенаправляются
# на сценарий squidGuard.cgi
```

```
default {  
    pass none  
    redirect http://127.0.0.1/cgi-  
bin/blocked.cgi?clientaddr=%a&srcclass=%s&targetclass=%t&url=%u  
  
}  
}
```

В файле конфигурации squidGuard нет ничего сложного — вам понадобится вписать в него только IP-адреса вашей сети, а также время работы.

Наверное, вы обратили внимание, что вместо просмотра запрещенного контента браузер перенаправляется на сценарий `squidGuard.cgi`, установленный на локальном Web-сервере. Получается, что для работы squidGuard требуется Web-сервер Apache. Если, кроме как для squidGuard, Apache вам более ни для чего не нужен, просто установите его, — в настройках он нуждаться не будет, хватит конфигурации по умолчанию (см. главу 33). Также не нужно самостоятельно копировать файл `blodked.cgi` в каталог `/var/www/cgi-bin` — это происходит автоматически при установке squidGuard.

Практически все готово. Нам нужно только указать, что Squid должен использовать squidGuard. Сделать это очень просто — достаточно добавить в файл `/etc/squid3/squid.conf` строки:

```
redirector_bypass on  
redirect_program /usr/local/squidGuard/bin/squidGuard  
  
redirect_children 1
```

Осталось лишь перезапустить Squid:

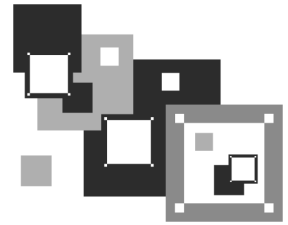
```
sudo service squid3 restart
```

После этого откройте журнал squidGuard — `/var/log/squidGuard/squidGuard.log`. В нем вы должны увидеть строку

```
squidGuard ready for requests
```

Если она есть, значит, вы все сделали правильно, и squidGuard работает.

ГЛАВА 37



Почтовый сервер

37.1. Выбор почтового сервера

Системным администраторам Linux частенько приходится настраивать собственный почтовый сервер, особенно в корпоративной среде, когда использование почтового сервера провайдера не всегда уместно или попросту недопустимо из-за существующей политики информационной безопасности.

Почтовый сервер состоит из двух компонентов: SMTP-сервера и POP-сервера. Сервер SMTP (Simple Mail Transfer Protocol) служит для приема почты от пользователя и ее передачи другим почтовым серверам. Сервер POP (Post Office Protocol) обеспечивает получение корреспонденции пользователями. Иногда вместо протокола POP используется протокол IMAP, позволяющий загружать на компьютер пользователя только заголовки пришедших ему отправок, — в отличие от протокола POP, сразу же загружающего на компьютер пользователя всю поступившую почту. Работа по протоколу IMAP позволяет пользователю предварительно просмотреть полученные заголовки и загрузить на свой компьютер лишь нужные ему отправления. Впрочем, при организации корпоративной почты выбор протокола, как правило, зависит только от предпочтений администратора сети.

Взаимодействие пользователей с почтой на их локальных компьютерах обеспечивают *почтовые клиенты* (Mail User Agent, MUA), умеющие работать как с протоколом отправки почты (SMTP), так и с протоколами получения почты (POP, IMAP).

Работу с почтой пользователя на почтовом сервере обеспечивают программы (агенты) передачи почты (MTA, Mail Transfer Agent). Ранее стандартом почтового агента де-факто считался MTA sendmail, первые версии которого были довольно дырявыми и, мягко говоря, небезопасными. Сейчас с безопасностью у sendmail все в порядке, но все равно это довольно старый и сложный в настройке почтовый агент.

MUA И MTA

Если вы ранее не были знакомы с аббревиатурами MUA и MTA, то наверняка запутались. MUA — это пользовательская программа для работы с электронной почтой, например Outlook Express, The Bat!. В процессе своей работы MUA подключается к почтовому серверу и принимает/передает электронные сообщения. Программа, установленная на сервере и принимающая сообщения от MUA, а затем передающая их на другой сервер, — это и есть MTA. Схема работы системы электронной почты показана на рис. 37.1.



Рис. 37.1. Отправка и получение письма

Корни sendmail уходят в UNIX-системы, а в Linux он стал популярным, поскольку, кроме него, в то время больше ничего не было, да и устанавливался он по умолчанию во всех дистрибутивах. Потом место под солнцем занял MTA postfix, отличающийся относительно простой настройкой, особенно по сравнению с sendmail. Впрочем, многие администраторы настолько привыкли к sendmail, что долго не спешили переходить на новый почтовый агент.

Кроме sendmail и postfix существуют еще очень достойные MTA: QMail и Exim. QMail сочетает в себе функции не только SMTP, но и POP3-сервера, что в некоторых случаях упрощает его настройку. В Интернете бытует мнение о якобы сложной настройке QMail. Этот миф полностью развеян мной в книге «Серверное применение Linux, 3-е изд.»¹, где MTA QMail описан подробно,— ничего сложного в нем нет, и нужно просто не спеша с ним разобраться. В этой же книге, чтобы не повторяться, будет рассмотрен MTA Exim версии 4.

Здесь мы также не станем углубляться в настройку POP3-сервера, обеспечивающего получение почты, — для реализации такого сервера нужно просто установить пакет cyrus-pop3d. Сразу после установки сервер готов к работе, а дальнейшая его настройка, как правило, требуется далеко не всегда.

37.2. Настройка MTA Exim

Процесс настройки четвертой версии MTA Exim, рассмотренный здесь на примере дистрибутивов Ubuntu/Ubuntu Server/Debian (для остальных дистрибутивов соответствующую информацию можно найти в Интернете), существенно отличается от

¹ См. <http://bhv.ru/books/book.php?id=188723>.

настройки его третьей версии, но сравнивать процессы настройки версий 3 и 4 я не стану, — в этом нет смысла, поскольку 3-я версия Exim более не актуальна.

Итак, установите пакет `exim4`, после чего можно приступить к его настройке.

Основной конфигурационный файл конфигурации Exim4 — `/var/lib/exim4/config.autogenerated` — весьма сложен, поэтому на первых порах гораздо проще воспользоваться специальным конфигуратором, позволяющим настроить большинство параметров Exim4. Это один из немногих случаев, когда я считаю использование конфигуратора оправданным (рис. 37.2).

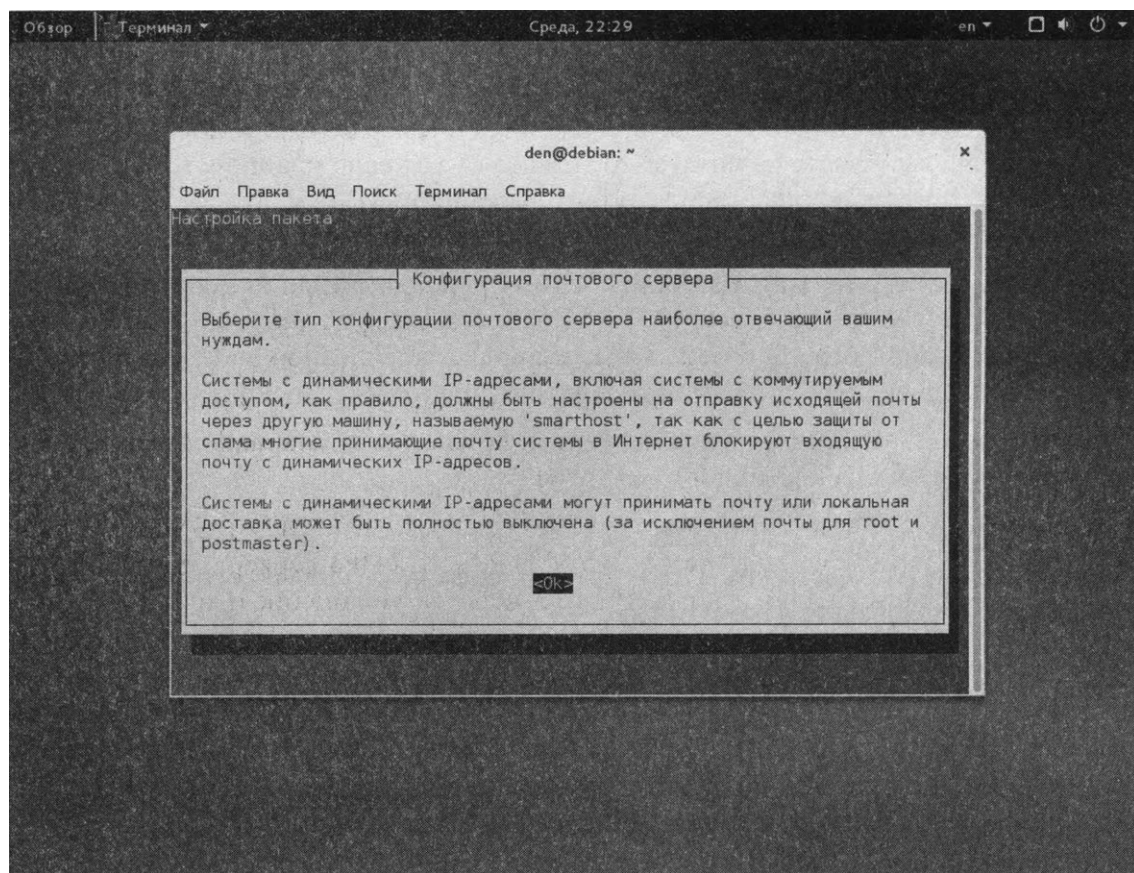


Рис. 37.2. Конфигуратор Exim4

Для запуска конфигуратора введите команду:

```
sudo dpkg-reconfigure exim4-config
```

и произведите в нем необходимые настройки (см. далее). Все параметры, настроенные в конфигураторе, будут сохранены в файле `/etc/exim4/update-exim4.conf`. Это обычный текстовый файл, и при желании его можно отредактировать в любом текстовом редакторе. Если вам понадобится исправить один-два параметра, нет смысла запускать снова конфигуратор, — достаточно отредактировать этот файл вручную.

Произведя в конфигураторе необходимые настройки, введите следующую команду, которая сгенерирует основной конфигурационный файл:

```
sudo update-exim4.conf
```

Обратите внимание, что основной конфигурационный файл `/var/lib/exim4/config.autogenerated` не следует редактировать вручную, поскольку он будет перезаписан при следующем запуске команды `update-exim4.conf`.

Осталось запустить Exim4:

```
sudo service exim4 start
```

37.3. Настройка аутентификации SMTP

Если вы рассчитываете использовать свой SMTP-сервер только в локальной сети, где «все свои», вы можете не читать этот раздел. А вот если планируется его работа в Интернете, тогда нужно настроить аутентификацию SMTP, иначе ваш сервер станет «точкой рассылки спама», — его сможет использовать любой желающий.

Далее мы рассмотрим, как настроить Exim4 для организации аутентификации SMTP-AUTH с учетом требований TLS (Transport Layer Security, протокола безопасности транспортного уровня) и SASL (Simple Authentication and Security Layer, метода добавления поддержки аутентификации в протоколы соединения).

Первым делом нужно с помощью следующей команды создать сертификат для использования TLS:

```
sudo /usr/share/doc/exim4-base/examples/exim-gencert
```

После этого следует настроить TLS — откройте файл `/etc/exim4/conf.d/main/03_exim4-config_tlsoptions` и добавьте в него строку:

```
MAIN_TLS_ENABLE = yes
```

Далее надо настроить Exim4 на использование демона `saslauthd` (сервера аутентификации SASL) — откройте файл конфигурации `/etc/exim4/conf.d/auth/30__exim4-config_examples` и раскомментируйте секции `plain_saslauthd_server` и `login_saslauthd_server`. Должно получиться так:

```
plain_saslauthd_server:
    driver = plaintext
    public_name = PLAIN
    server_condition = ${if saslauthd{{${auth2}}{{${auth3}}}{1}}{0}}
    server_set_id = $auth2
    server_prompts = :
    . ifndef AUTH_SERVER_ALLOW_NOTLS_PAS SWORDS
    server_advertise_condition = ${if eq{${tls_cipher}}{}}{*}}
    .endif
```

```
#
```

```
login_saslauthd_server:
    driver = plaintext
```

```
public_name = LOGIN
server_prompts = "Username:: : Password::"
# не отправлять системные пароли по незашифрованным соединениям
server_condition = ${if saslauthd{{sau\hl}}{$auth2}}{{1}}{0}}
server_set_id = $auth1
.ifdef AUTH_SERVER_ALLOW_NOTLS_PAS SWORDS
server_advertise_condition = ${if eq{$tls_cipher}}{{}}{{*}}
.endif
```

Обеспечьте теперь возможность почтовому клиенту соединиться с вашим новым SMTP-сервером, для чего добавьте в Exim нового пользователя:

```
sudo /usr/share/doc/exim4/examples/exim-adduser
```

Новый файл паролей нужно защитить — для этого изменим владельца файла и права доступа к нему:

```
sudo chown root:Debian-exim /etc/exim4/passwd
sudo chmod 640 /etc/exim4/passwd
```

Впрочем, при постоянном добавлении пользователей вам придется снова и снова изменять права доступа. Так что, дважды подумайте, нужно ли вам это.

Осталось обновить конфигурацию и перезапустить Exim4:

```
sudo update-exim4.conf
sudo service exim4 restart
```

37.4. Настройка демона SASL

Теперь осталось настроить сам saslauthd, чтобы обеспечить аутентификацию для Exim4. Прежде всего установите пакет sasl2-bin и отредактируйте файл /etc/default/saslauthd, заменив в нем строку:

```
START=no
```

строкой:

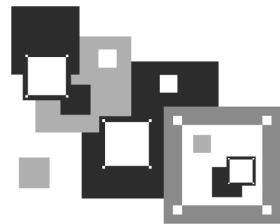
```
START=yes
```

Осталось добавить пользователя Debian-exim в группу sasl, чтобы Exim4 смог использовать сервис saslauthd и запустить сервис saslauthd:

```
sudo adduser Debian-exim sasl
sudo service saslauthd start
```

На этом все— с этого момента ваш SMTP-сервер использует аутентификацию TLS/SASL.

ГЛАВА 38



Сервис Samba

38.1. Установка Samba

Linux — отличная операционная система, но и от Windows нам не уйти. Windows будет окружать нас всегда, будь то домашняя, корпоративная сеть или интернет-кафе. Нам постоянно предстоит обмениваться документами с Windows-компьютерами — ведь далеко не все пользователи предпочитают работать в Linux. В этой книге особое внимание было уделено взаимодействию с Windows-компьютерами, и было бы нелогично не сказать о подключении Linux к сети Microsoft.

Взаимодействие с сетью Microsoft в Linux обеспечивает пакет `samba` (в старых дистрибутивах — `samba-server`). Если вы хотите использовать общие ресурсы Windows-сети, установите этот пакет, — он позволяет не только пользоваться общими ресурсами сети, но и предоставлять собственные ресурсы Windows-пользователям. Причем все происходит так, что Windows-пользователи даже не заметят разницы.

Кроме пакета `samba` вам также понадобятся пакеты, обеспечивающие поддержку сетевого протокола аутентификации Kerberos и специального демона WinBind:

```
sudo dnf install samba krb5-user winbind
```

После установки этого пакета будет установлен сервис `smb` — это и есть основной сервис Samba. Запускать и останавливать его можно командами:

```
sudo service smbd start
sudo service smbd stop
```

38.2. Базовая настройка Samba

Основной конфигурационный файл Samba — `/etc/samba/smb.conf`. Откройте его и перейдите к секции `[global]`.

Прежде всего, измените в ней параметр `workgroup` — он задает имя рабочей группы или домена NT:

```
WORKGROUP = MSHOME
```

Впрочем, имя группы у вас, скорее всего, будет другим, — его сюда и впишите. Можете также установить параметр `server string` (он ранее назывался `comment`) — это описание вашего компьютера:

```
server string = My Linux computer
```

Установите параметр `security` — если у вас сеть «клиент-сервер», то нужно выбрать параметр `server`, а если сеть одноранговая (т. е. без выделенного сервера), выберите `user` или `share`:

```
security = share
```

Имя гостевой учетной записи установите так:

```
guest account = guest
```

Следует настроить и кодировки:

```
unix charset = UTF-8
```

```
dos charset = UTF-8
```

```
display charset = UTF-8
```

Для того чтобы Samba работала быстрее, установите следующие опции (что они означают, мы разберемся чуть позже):

```
socket options = TCP_NODELAY SO_RCVBUF=8192 SO_SNDBUF=8192
```

```
dns proxy = no
```

Параметр `interfaces` указывает интерфейсы, на которых должен работать сервис `smb`, — укажите те интерфейсы, которые связывают вашу машину с Windows-сетями:

```
interfaces = 192.168.0.22/24
```

А теперь позволю себе несколько комментариев для пользователей более ранних версий Samba;

- параметр `server string` ранее назывался `comment`;
- параметры `client code page` и `character set` больше не поддерживаются — теперь вместо НИХ используются параметры `unix charset`, `dos charset` и `display charset`:
 - параметр `unix charset` задает кодировку, в которой хранятся файлы конфигурации Samba;
 - параметр `dos charset` — кодировку для Windows-клиентов;
 - параметр `display charset` — кодировку для Samba-клиентов;
- текущая версия Samba полностью поддерживает кодировку UTF-8 (как и современные версии Linux и Windows), поэтому проблем с UTF-8 возникнуть не должно, и мы спокойно задаем эту кодировку, как и показано чуть ранее.

38.3. Настройка общих ресурсов

Теперь осталось сконфигурировать ресурсы, которые вы хотите предоставить в общее пользование. Фрагмент, приведенный в листинге 38.1, нужно добавить в файл конфигурации Samba.

Листинг 38.1. Секция [public]

```
[public]
# общий каталог, комментарий для ресурса задается директивой comment
comment = Public Directory
# путь
path = /var/samba
# не только чтение
read only = no
# разрешить запись
writable = yes
# разрешить гостевой доступ
guest ok = yes
# разрешить просмотр содержимого каталога
browseable = yes
```

В этом случае общим ресурсом нашего компьютера будет каталог `/var/samba`. В него Другие пользователи смогут записывать свои файлы (`read only = no`, `writable = yes`), естественно, они смогут их и читать (`browseable = yes`). Проверка имени пользователя и пароля для доступа к ресурсу не нужна (`guest ok = yes`) — используется так называемый гостевой доступ. Комментарий `Public Directory` увидят другие пользователи Windows-сети при просмотре ресурсов вашего компьютера.

ДИРЕКТИВЫ COMMENT И SERVER STRING

Как уже было отмечено ранее, начиная с третьей версии Samba в ее конфигурационном файле произошли небольшие изменения. В секции `[global]`, описывающей глобальные параметры Samba, теперь вместо директивы `comment` используется директива `server string`, однако при описании разделяемых ресурсов (т. е. каталогов, к которым вы предоставили общий доступ) используется та же директива `comment`.

Рассмотрим еще один пример, позволяющий сделать общими домашние каталоги пользователей, — секцию `[homes]` (листинг 38.2).

Листинг 38.2. Секция [homes]

```
[homes]
comment = Home Directories
browseable = no
valid users = %S

# запись запрещена, только просмотр
writable = no
```

```
# маска при создании файлов, нужна если writable=yes
create mask = 0600
# маска при создании каталогов, нужна если writable=yes
directory mask = 0700
```

В листинге 38.3 приведен пример предоставления общего доступа к CD/DVD (будем считать, что наш CD/DVD смонтирован в каталоге `/cdrom`).

Листинг 38.3. Пример общего доступа к CD/DVD

```
[ cdrom]
    comment = Samba server's CD-ROM
    writable = no
    locking = no
# каталог /cdrom должен существовать и являться точкой монтирования CD/DVD
    path = /cdrom
    public = yes
# следующие два параметра нужны для автоматического монтирования CD/DVD
# они будут работать, если /etc/fstab содержит следующую строку:
# /dev/scd0 /cdrom iso9660 defaults,noauto,ro,user 0 0
# /dev/scd0 — имя устройства CD/DVD
# /cdrom — точка монтирования (каталог должен существовать)
    preexec = /bin/mount /cdrom
    postexec = /bin/umount /cdrom
```

38.4. Просмотр ресурсов Windows-сети

Просмотреть ресурсы Windows-сети можно с помощью программы `smbclient`, но она работает в текстовом режиме, поэтому не совсем удобна. В современных дистрибутивах ресурсы Windows-сети лучше просматривать средствами графической среды. Так, в KDE откройте файловый менеджер Dolphin, в боковой панели выберите **Сеть**, а затем — **Samba Shares** (рис. 38.1).

Если вы используете GNOME, то для просмотра ресурсов сети можно применять команду главного меню **Переход | Сеть**, что даже проще и удобнее, чем с помощью файлового менеджера Dolphin.

38.5. Оптимизация Samba

Открыв файл конфигурации `smb.conf`, вы найдете в нем параметр `wide links`. Никогда не устанавливайте его в `no` — так вы существенно снизите производительность Samba! Наоборот, если вы установите его в `yes` (если до этого параметр `wide links` был отключен), то сможете значительно повысить производительность сервиса. Дело в том, что параметр `wide links` определяет, как Samba будет следовать по символическим ссылкам. Сначала Samba следует по символической ссылке, а затем выполняет так называемый `directory path lookup` (системный вызов, определяю-

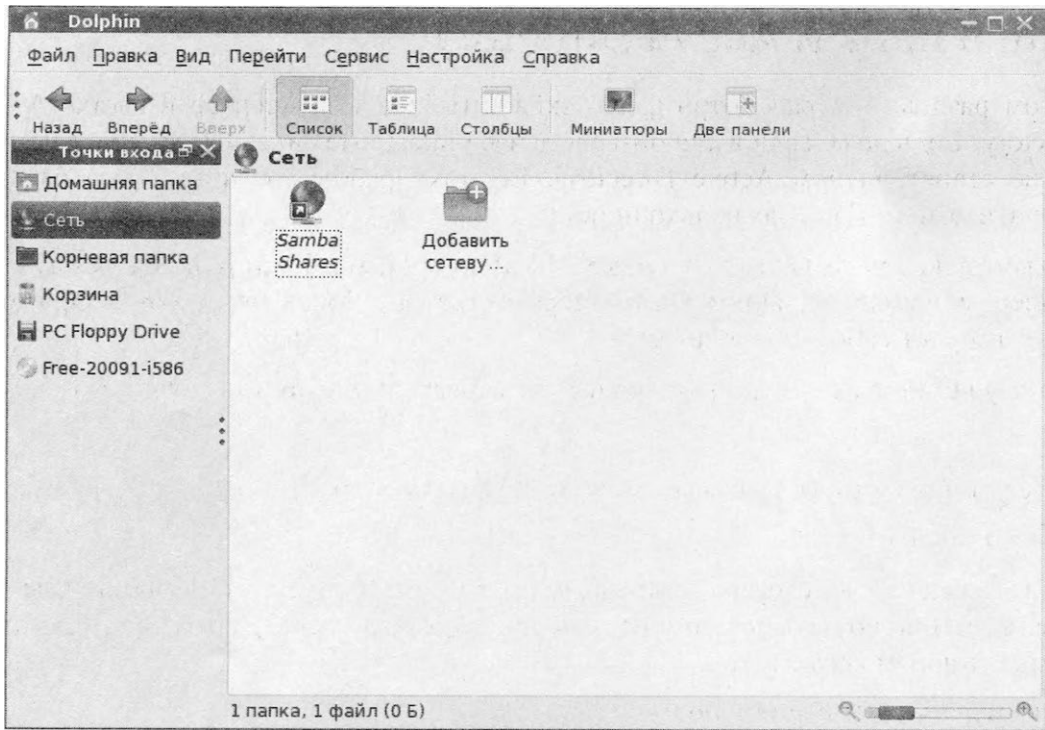


Рис. 38.1. Просмотр ресурсов сети с помощью Dolphin

щий, где завершилась ссылка). Если `wide links = no`, то Samba не будет следовать по символическим ссылкам вне экспортируемой области. Эта операция подразумевает на 6 системных вызовов больше, нежели в случае, если `wide links = yes`. Учитывая, что подобных операций делается очень много, то выключение `wide links` снижает производительность Samba приблизительно на 30%.

Протокол TCP/IP — штука тонкая. Производительность сетевых приложений во многом зависит от того, правильно ли настроен TCP/IP. Samba — настоящее сетевое приложение, которое к тому же работает по протоколу TCP/IP. При использовании TCP/IP, если размер запросов и ответов не фиксирован (как в случае с Samba), рекомендуется применять протокол TCP с опцией `TCP_NODELAY`. Для этого в файл `smb.conf` нужно добавить строку:

```
socket options = TCP_NODELAY
```

Тесты показывают, что с этими опциями Samba при больших нагрузках работает в три раза быстрее, чем без их указания. Если Samba используется в локальной сети (в большинстве случаев так оно и есть), рекомендуется еще указать и опцию

```
IP_TOS_LOWDELAY:
```

```
socket options = IPTOS_LOWDELAY TCP_NODELAY
```

При желании «выжать» из Samba еще больше, установите следующие параметры буферизации: `so_rcvbuf=8192 so_sndbuf=8192`. Например:

```
socket options = TCP_NODELAY SO_RCVBUF=8192 SO_SNDBUF=8192
```

38.6. Samba и Active Directory

В этом разделе мы рассмотрим, как включить ваш Linux-сервер в состав Active Directory. Ведь даже если вы произведете все описанные ранее настройки, ваш сервер не станет членом Active Directory. Все-таки рабочая группа — это рабочая группа, а домен AD — несколько иное.

Первым делом надо настроить синхронизацию времени. Если разница между контроллером домена и вашим компьютером составит более пяти минут, протокол Kerberos будет работать неправильно.

Поэтому установим пакет `ntp` (он содержит сервер синхронизации времени):

```
sudo dnf install ntp
```

После установки в файле `/etc/ntp.conf` нужно указать строку:

```
server dc.domain.ru
```

Так вы задаете имя сервера, с которым нужно синхронизировать время. Следует указать именно контроллер домена, а не произвольный сервер времени, пусть даже с самым точным временем.

Далее перезапустите демон `ntp`:

```
service ntp restart
```

Надо также убедиться, что система DNS работает правильно, — в противном случае `ntp` просто не сможет найти компьютер с именем **dc.domain.ru**. Если это не так, внесите информацию о DNS-серверах вашего домена в файл `/etc/resolv.conf` и перезапустите сеть:

```
service networking restart
```

После настройки синхронизации времени отредактируйте основной файл конфигурации Kerberos — `/etc/krb5.conf` (листинг 38.4). Понятно, название домена `DOMAIN.RU` следует заменить реальным.

Листинг 38.4. Файл конфигурации `/etc/krb5.conf`

```
[libdefaults]
    def_aul t_realm = DOMAIN. RU
    kdc_timesync = 1
    ccache_type = 4
    forwardable = true
    proxiable = true
    v4_instance_resolve = false
    v4_name_convert = {
        host = {
            rcmd = host
            ftp = ftp
        }
    }
```

```

        plain = {
            something = something-else
        }
    }
    fcc-mit-ticketflags = true

[realms]
    DOMAIN.RU = {
        kdc = dc
        admin_server = dc
        master_kdc = dc.damain.ru
        def_auth_t_dcmain = domain.ru
    }

[domain_realm]
    .damain.ru = DOMAIN.RU
    domain.ru = DOMAIN.RU

[login]
    krb4_convert = false
    krb4_get_tickets = false

```

Сохраните файл конфигурации и проверьте работу Kerberos:

```
kinit <пользователь>@DOMAIN.RU
```

Обратите внимание — имя домена нужно указывать заглавными буквами. Если вы не увидели сообщение об неуспешной аутентификации (в случае успеха команда `kinit` ничего не выводит), тогда можно приступить к следующему этапу настройки. В противном случае отредактируйте файл конфигурации Kerberos и устраните допущенные ошибки.

Теперь надо настроить сервис Samba — кроме названия рабочей группы задать еще и название зоны (realm), а также указать Samba, что авторизация должна проходить через службы Active Directory (ADS), а пароли должны шифроваться.

Чтобы нечаянно не повысить свой сервер до уровня контроллера домена, нужно явно отключить опции вроде `domain master`, `local master` и т.п., — т. е. все опции, которые так или иначе могут повлиять на работу AD. В листинге 38.5 приводится секция `[global]` сервера, входящего в состав AD. Остальные секции (в которых предоставляются ресурсы) вы можете заполнить по своему усмотрению.

Листинг 38.5. Секция `[global]` для Active Directory

```

[global]
    # Имена рабочей группы и домена указываются заглавными буквами
    workgroup = DOMAIN
    realm = DOMAIN.RU

    # Авторизация через AD, шифрование паролей
    security = ADS
    encrypt passwords = true

```

```
# Отключаем прокси DNS
dns proxy = no

# Наш сервер - не контроллер домена!
local master = no
domain master = no
preferred master = no
os level = 0
domain logons = no
```

Проверьте теперь файл конфигурации Samba командой `testparm`. Обратите внимание на роль сервера (`server role`):

```
Load smb config files from /etc/samba/smb.conf
Loaded services file OK.
Server role: ROLE_DOMAIN_MEMBER
```

В нашем случае роль нашего сервера— член домена. Главное, чтобы ненароком наш сервер не превратился в контроллер домена.

Запустите Samba:

```
sudo service smbd start
```

А после запуска проверьте, сможет ли пользователь войти в домен:

```
# net ads join -U user -D DOMAIN
Enter user's password:
Using short domain name — DOMAIN
Joined 'server' to realm 'domain.ru'
```

Если в ходе выполнения этой команды вы увидите сообщение **DNS update failed**, проверьте параметры DNS, если вы этого еще не сделали.

Если вы сейчас попытаетесь настроить общие ресурсы, то у вас ничего не получится. А все потому, что мы еще не успели настроить Winbind — специальный демон, служащий для связи локальной системы управления пользователями и группами Linux с сервером Active Directory. Итак, в секцию `[global]` файла конфигурации Samba добавьте следующие строки, их можно не изменять. Если же все-таки вы захотите их почему-либо изменить, то обратитесь к документации, чтобы вы понимали, что делаете:

```
idmap uid = 10000 - 40000
idmap gid = 10000 - 40000

winbind enum groups = yes
winbind enum users = yes
winbind use default domain = yes
template shell = /bin/bash
winbind refresh tickets = yes
```

После этого в файле `/etc/nsswitch.conf` найдите следующие строки:

```
passwd: compat
group:  compat
```

Их следует дополнить так:

```
passwd: compatwinbind
group:  compatwinbind
```

Осталось перезапустить Samba и Winbind в указанной последовательности:

```
sudo service winbind stop
sudo service smbd restart
sudo service winbind start
```

Можно, конечно, перезагрузить компьютер. Но это же Linux, поэтому достаточно обойтись просто перезагрузкой сервисов.

38.7. Samba в качестве контроллера домена

При желании компьютер с установленной Samba можно превратить контроллер домена — так, что обычные рабочие станции будут «думать», что их обслуживает Windows Server, но никак не операционная система Linux.

Прежде, чем мы приступим к этой процедуре, нужно упомянуть некоторые настройки, которые мы будем задавать. Название домена у нас будет COMPANY.LOC или просто COMPANY (сокращенное). Имя сервера в домене: ad, а его IP-адрес: 192.168.0.1.

Установим сначала необходимое программное обеспечение:

```
sudo apt-get install samba acl krb5-user ntp cups bind9 smbclient
```

Для поднятия домена воспользуемся утилитой `samba-tool`, которая входит в состав `samba4`:

```
sudo samba-tool domain provision --use-rfc2307 --interactive
```

Далее нужно ответить на вопросы этой утилиты:

```
Realm [COMPANY.LOC]: COMPANY.LOC
Domain [COMPANY]: COMPANY
Server Role (dc, member, standalone) [dc]: dc
DNS backend (SAMBA_INTERNAL, BIND9_FLATFILE, BIND9_DLZ, NONE) [SAMBA_INTERNAL] :
BIND9_DLZ
Administrator password:
Retype password:
```

Здесь мы сначала вводим полное и сокращенное имя домена, затем роль нашего сервера: `dc` (то есть контроллер домена), затем указываем, что для разрешения имен будет использоваться `BIND9`. Последние два вопроса — это пароль администратора.

Если в процессе настройки что-то пошло не так, то сначала нужно очистить конфигурацию Samba, а затем запустить процесс настройки заново. Для очистки конфигурации выполните команды:

```
sudo apt-get purge samba
sudo apt-get install samba
sudo rm /etc/samba/smb.conf
```

Оставим пока Samba в покое и настроим BIND9, необходимый для разрешения имен. Первым делом откройте файл `/etc/bind/named.conf.options` и приведите его к виду, показанному в листинге 38.6. Конечно, IP-адреса здесь следует заменить своими.

Листинг 38.6. Файл `/etc/bind/named.conf.options`

```
options {
    directory "/var/cache/bind";
    auth-nxdomain yes;
    forwarders { 192.168.0.2; };
    allow-transfer { none; };
    notify no;
    empty-zones-enable no;

    allow-query {
        192.168.0.0/24;
        127.0.0.0/8;
    };

    allow-recursion {
        192.168.0.0/24;
        127.0.0.0/8;
    };

    allow-update {
        192.168.0.0/24;
        127.0.0.0/8;
    };
};
```

Затем откройте файл `/etc/bind/named.conf` и в его конец добавьте строку:

```
include "/var/lib/samba/private/named.conf";
```

Теперь нужно открыть файл, указанный в приведенной команде (`/var/lib/samba/private/named.conf`), и привести его к виду, показанному в листинге 38.7.

Листинг 38.7. Файл `/var/lib/samba/private/named.conf`

```
dlz "AD DNS Zone" {
    # For BIND 9.8.0
    # database "dlopen /usr/lib/x86_64-linux-gnu/samba/bind9/dlz_bind9.so";
```

```
# For BIND 9.9.0
database "diopen /usr/lib/x86_64-linux-gnu/samba/bind9/dlz_bind9_9.so";
};
```

Чтобы разрешить BIND9, нам нужно исправить конфигурацию apparmor (если вы его используете). Для этого откройте файл `/etc/apparmor.d/usr.sbin.named` и до закрывающей скобки `}` вставьте строки:

```
# for samba4
#/var/lib/samba/private/** r,
/usr/lib/x86_64-linux-gnu/samba/bind9/** m,
/usr/lib/x86_64-linux-gnu/samba/ldb/** m,
/usr/lib/x86_64-linux-gnu/ldb/modules/ldb/** m,
/usr/lib/x86_64-linux-gnu/samba/gensec/krb5.so m,
/var/lib/samba/private/dns. keytab rwk,
/var/lib/samba/private/named. conf r,
/var/lib/samba/private/dns/** rwk,
/var/lib/samba/private/krb5.conf r,
/var/tmp/** rwk,
/dev/urandom rwk* .
```

Отредактируйте файл `/etc/resolv.conf`, чтобы добавить наш DNS-сервер:

```
nameserver 192.168.0.1
```

По сути, почти все готово. Осталось только перезапустить службы и запустить Samba4:

```
sudo service apparmor restart
sudo service bind9 restart
sudo service samba-ad-dc start
```

Попробуем войти как администратор:

```
smbclient //localhost/netlogon -UAdministrator -c 'ls'
```

Enter Administrator's password:

Domain=[ADSAMBA] OS=[Unix] Server=[Samba 4.1.6-Ubuntu]

D 0 Tue Apr 8 11:21:49 2014

DO Tue Apr 8 11:22:03 2014

46445 blocks of size 4194304. 45581 blocks available

Для полного счастья нужно настроить еще Kerberos и NTP. С Kerberos все довольно просто:

```
sudo In -s /var/lib/samba/private/krb5.conf /etc/
kinit administrator@COMPANY.LOC
```

Вторая команда получает билет (полученный билет можно просмотреть командой `klist`). Обратите внимание, что при получении билета вы увидите, что аккаунт администратора (точнее, пароль аккаунта) будет действителен в течение 41 дня:

Warning: Your password will expire in 41 days...

Изменить длительность пароля можно так:

```
sudo samba-tool domain passwordsettings set --max-pwd-age=90
```

Здесь мы установили срок действия пароля 90 дней.

Настроить NTP тоже не сложно. Откройте файл `/etc/ntp.conf` и в самый конец добавьте строки:

```
ntpsigndsocket /var/lib/samba/ntp_signd/  
restrict default mssntp
```

После этого нужно перезапустить сервис `ntp`, и вы получите полноценный контроллер домена. Не забудьте на Windows-компьютерах указать его IP-адрес в качестве адреса первичного DNS-сервера.

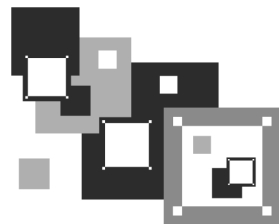
Для управления доменом можно использовать пакет `Adminpack.msi`, который можно скачать по адресу:

<http://www.microsoft.com/downloads/details.aspx?FamilyID=7D2F6AD7-656B-4313-A005-4E344E43997D&disp!aylang=ru>

<http://download.microsoft.com/download/3/e/4/3e438f5e-24ef-4637-abdl-981341d349c7/WindowsServer2003-KB892777-SupportTools-x86-ENU.exe>

Первая ссылка— это его первая часть, вторая ссылка—соответственно, вторая часть пакета. Установите все и выполните команду `dsa.msc` для управления доменом.

ГЛАВА 39



Поддержка RAID

39.1. Аппаратные RAID-массивы

Массивы RAID (Redundant Array of Independent Disk, массив независимых дисков с избыточностью) повышают надежность хранения ваших данных. Так, при объединении двух винчестеров в один RAID-массив все, что будет записано на первый винчестер, автоматически продублируется на второй. И если с первым винчестером что-то случится (а у жестких дисков есть неприятное свойство выходить из строя через 5-7 лет эксплуатации), мы сможем восстановить свои данные со второго винчестера. Приведенный пример является далеко не единственным способом организации RAID-массивов. Алгоритм работы RAID-массива определяется его уровнем — самые популярные уровни RAID-массивов представлены в табл. 39.1, а за информацией об остальных уровнях вы можете обратиться или к Википедии (<https://ru.wikipedia.org/wiki/RAID>), или к калькулятору RAID (<http://altastor.ru/apps/raid-calculator/>).

На практике обычно используются уровни 0, 1 и 5. Ранее поддержкой RAID-массивов на аппаратном уровне обладали только дорогие серверные материнские платы. Сейчас поддержку RAID можно встретить в относительно недорогих материнских платах среднего ценового диапазона. О возможности создания и поддержки аппаратных RAID-массивов вы можете прочитать в документации на материнскую плату своего компьютера.

Как следует из данных табл. 39.1, некоторые производители создают комбинированные уровни: RAID 10 (1+0), RAID 15 (1+5), RAID 50 (5+0) и пр. Суть таких комбинаций заключается в следующем: RAID 10 — это комбинация уровней 1 и 0, 15 — это уровни 1+5, т. е. зеркало «пятерок», и т. д. Такие комбинированные уровни сочетают в себе как преимущества, так и недостатки своих «родителей». Например, уровень RAID 50 — практически то же, что и RAID 5, но зато быстрее, чем RAID 5.

Кроме обычных уровней, организуются еще и расширенные уровни RAID, — тогда к наименованию уровня добавляется буква E, — например: RAID 1E, RAID 5E и т. д. Это усовершенствованные версии базовых уровней. Чтобы не описывать каждый такой уровень, обратимся к данным табл. 39.2, описывающим общие характеристики самых часто используемых уровней RAID, т. е. именно тех уровней, с которыми вы будете работать на практике.

Таблица 39.1. Уровни RAID-массивов

Уровень	Описание	Кол-во дисков	Полезный объем, %
0	Предназначен не для обеспечения надежности, а для увеличения суммарного объема диска. Так, объединив в RAID-массив два винчестера по 200 Гбайт, мы получим один диск на 400 Гбайт. Очень удобно, если мы работаем с видео (имеется в виду профессиональный видеомонтаж, а не просмотр кинофильмов)	1-16	100
00	Похож на RAID 0, но позволяет объединить в массив от 2 до 60 дисков	2-60	100
1	Простое зеркальное копирование, как было описано ранее. Все, что записано на первый жесткий диск, будет продублировано на второй. Желательно, чтобы диски были одного размера. Если это не так, то размер RAID-массива будет равен размеру меньшего диска	2-16	50
5	Самый оптимальный уровень по соотношению производительность/надежность. Использует контрольные суммы и данные записываются вместе с контрольными кодами на все диски. Если с одним из дисков что-то случилось, то данные можно восстановить с помощью контрольной суммы. Общий размер массива вычисляется по формуле $M \cdot (N - 1)$, где N — количество дисков в массиве, а M — размер наименьшего диска. Минимальное значение $N = 3$	3-16	67-94
5E	Модификация RAID 5. По сути, то же самое, что и RAID 5, но предполагает использование резервного диска. На всех дисках массива резервируют $1/N$ пространства, где N — количество дисков в массиве. Зарезервированная часть пространства используется при отказе одного из дисков. RAID 5E предоставляет лучшую производительность — ведь чтение и запись производятся параллельно с большего количества дисков	4-16	50-88
5EE	Модификация RAID 5E, обеспечивающая эффективное распределение резервного диска и быстрое время восстановления	4-16	50-88
6	Представляет собой усовершенствованный уровень 5. RAID 6 надежнее, чем RAID 5, но менее производительный. Скорость чтения информации примерно такая же, как и в случае с RAID 5, но скорость записи обычно ниже на 40-50%, не говоря уже о медленном восстановлении данных. Однако RAID 6 позволяет восстановить данные даже в случае выхода из строя двух жестких дисков. Довольно дорог в реализации, поскольку требует как минимум четыре жестких диска, а полезная емкость равна $N - 2$, где N — это количество дисков (два жестких диска отводятся для хранения контрольных сумм). Если у вас в массиве четыре жестких диска по 200 Гбайт каждый, то полезная емкость составит только 400 Гбайт из 800. Кратко RAID6 можно охарактеризовать так: дорогой, медленный, но надежный. Из-за своей дороговизны и низкой производительности применяется редко. Однако его можно использовать, если надежность превыше всего	4-16	50-88

Таблица 39.1 (окончание)

Уровень	Описание	Кол-во дисков	Полезный объем, %
10 (он же RAID 1+0)	Диски массива парами объединяются в «зеркала» (уровень RAID 1), далее зеркала объединяются в общий массив с чередованием данных (RAID 0). Отсюда и название уровня — RAID 10 или RAID 1+0. В массив RAID 10 можно объединить только парное количество дисков: от 4 до 16. Довольно надежен, т. к. используются «зеркала», но в то же время быстр, поскольку от RAID 0 унаследована производительность. Полезная емкость — в два раза меньше общей емкости массива. Более предпочтителен, чем RAID 6 там, где нужны производительность и надежность	4-16	50
1E	Расширенная (E, Enhanced) версия RAID 1. Данные чередуются блоками по всем дискам массива, а затем еще раз чередуются со сдвигом на один диск. Этот уровень позволяет объединять от 3 до 16 дисков. По надежности примерно такой, как RAID 10, но имеет большую полезную емкость и еще большую производительность	3-16	50
ДЕО	Позволяет объединять в нулевой массив массивы уровня RAID 1E. Преимуществ в скорости нет — наоборот, этот массив работает медленнее, чем RAID 1E, преимуществ в надежности тоже нет — из-за сложной реализации он менее надежный, чем RAID 1E, но смысл его организации в объединении в один большой массив до 60 (!) дисков	6-60	50
15	Представляет собой «зеркало» массивов RAID 5. Количество дисков: от 6 до 60. Отличительная особенность — очень низкая полезная емкость массива (от 33 до 48%)	6-60	33-48

Таблица 39.2. Характеристики уровней RAID

Уровень	Избыточность	Резервный диск	Чтение	Запись	Емкость
0	-	-	10	10	100
1	+	-	8	8	50
5	+	-	10	7	67-94
6	+	-	10	7	50-88
10(1+0)	+	-	9	9	50
15	+	-	10	7	33-48
1E	+	-	8	8	50
1E0	+	-	8	8	50
50	+	-	10	7	67-94
5E	+	+	10	7	50-88
5EE	+	+	10	7	50-88
00	-	-	10	10	100

ПОЯСНЕНИЕ

Здесь колонка «Избыточность» показывает, поддерживает ли уровень избыточность данных. Колонка «Резервный диск» — поддерживает ли уровень RAID резервный диск (spare disk). Колонки «Чтение» и «Запись» — оценки производительности чтения и записи по 10-балльной шкале. Колонка «Емкость» — использование емкости дисков в процентном соотношении. Для удобства эта колонка дублирует аналогичную из табл. 39.1.

39.2. Программные RAID-массивы

В Linux можно создавать программные RAID-массивы, даже если материнская плата вашего компьютера не поддерживает их на аппаратном уровне. У программных массивов есть один маленький недостаток — они работают немного медленнее аппаратных. Впрочем, у них есть и одно неоспоримое преимущество — поскольку обработка данных происходит на программном уровне, совсем необязательно, чтобы жесткие диски, входящие в состав массива, были совместимы между собой. Например, можно создать массив уровня 5, который будет состоять из дисков EIDE, SATA и SCSI — эти три разных интерфейса объединить в аппаратный массив просто невозможно.

Поддержка RAID-массивов встроена в ядро по умолчанию, поэтому вам даже не придется его перекомпилировать. При загрузке Linux вы должны увидеть следующие строки:

```
md: md driver 0.90.2 MAX_MD_DEVS=256, MD_SB_DISKS=27
```

```
md: bitmap version 3.39
```

```
...
```

```
md: Autodetecting KASH arrays,
```

```
md: autorun...
```

```
md:... autorun DONE.
```

Появление этих строк (если при загрузке вы не успели их заметить, введите команду `dmesg`) означает, что ядро поддерживает RAID. Не поддерживать RAID могут лишь компактные ядра некоторых дистрибутивов, которые мы здесь рассматривать не будем. Многие дистрибутивы (Fedora, CentOS и др.) поддерживают RAID-массивы по умолчанию.

Если же поддержки RAID в вашем дистрибутиве почему-то не оказалось, то включить ее можно в разделе **Block device** конфигулятора `make menuconfig` (см. главу 20), — не забудьте только после этого перекомпилировать ядро. Загрузив систему с новым ядром, следует установить пакет `raidtools`, содержащий необходимые нам команды: `raidhotadd`, `raidhotremove`, `mkraid` — последняя команда создает RAID-массив, первая добавляет в него диск, а вторая — удаляет диск из массива.

39.3. Создание программных массивов

Попробуем объединить в программный RAID-массив уровня 1 (зеркало) два диска: `/dev/sda` и `/dev/sdb`. Такой массив обеспечит вам избыточность хранящихся на сервере данных. Уровень не очень эффективный, зато прост в реализации, управлении, восстановлении данных и в самой простой конфигурации требует всего два жестких диска.

ВСЕ МАНИПУЛЯЦИИ — В ВИРТУАЛЬНОЙ МАШИНЕ

Во избежание потерь данных и простоя производственного сервера убедительно вас прошу: все манипуляции с RAID-массивом производите сначала в виртуальной машине. Виртуальные машины VMware и VirtualBox (см. *главу 18*) позволяют создать практически любые конфигурации дисков разных типов, которые вы можете объединить в RAID-массив. Создайте в виртуальной среде конфигурацию, подобную той, которую хотите использовать на практике, и потренируйтесь сначала в ней.

Итак, загрузитесь с LiveCD или откройте консоль в инсталляторе Linux для доступа к командной строке.

Первым делом надо разметить диски, предназначенные для организации RAID. Для этого создайте на диске `/dev/sda` два раздела: один — размером 2 Гбайт (под раздел подкачки), а второй — на все оставшееся дисковое пространство. Этот раздел будет использоваться для корневой файловой системы, т. е. для хранения данных. Создавать более сложную конфигурацию с отдельными разделами для `/home`, `/var` и `/usr` мы пока не станем. Тип обоих создаваемых разделов — Linux raid (код `fd`). Напомним, что для разметки диска можно использовать утилиты `fdisk` или `cfdisk`.

Второй жесткий диск нужно разметить так же. Как вы уже догадались, его размер должен быть идентичен первому. Для экономии времени можете воспользоваться командой `sfdisk`:

```
sudo sfdisk -d /dev/sda | sfdisk /dev/sdb
```

Эта команда скопирует разметку диска `/dev/sda` на диск `/dev/sdb`. Таким образом, у вас получится два идентичных диска.

Далее командой `mdadm` нужно создать два RAID-массива:

```
sudo mdadm --create /dev/md0 --level=1 --raid-devices=2 /dev/sda1 /dev/sdb1
sudo mdadm --create /dev/md1 --level=1 --raid-devices=2 /dev/sda2 /dev/sdb2
```

Первая команда создает RAID-массив разделов подкачки (`/dev/md0`), вторая — разделов данных (`/dev/md1`).

Первый параметр команды `mdadm` (`--create`) указывает, что мы создаем RAID-массив, за ним следует имя устройства. Параметр `--level` задает уровень RAID, параметр `--raid-devices` — количество устройств, а далее через пробел приводится список устройств, формирующих RAID-массив.

После выполнения этих команд у вас образуются два устройства: `/dev/md0`, которое мы планируем задействовать для подкачки, и `/dev/md1`, которое будет использоваться для хранения данных.

Перед установкой системы на созданные устройства нужно пометить устройство `/dev/md0` как устройство подкачки:

```
sudo mkswap /dev/md0 setup
```

Теперь можно начать установку системы так, как бы вы делали это обычно. При выборе разделов следует определить устройство `/dev/md0` для использования в качестве подкачки, а для устройства `/dev/md1` задать точку монтирования `/`. Загрузчик нужно установить в MBR.

Собственно, на этом все. Вы думали, будет сложнее?

39.4. RAID-массив только для данных

В предыдущем разделе мы создали RAID-массив, обеспечивающий избыточность всех данных сервера, в том числе и раздела подкачки. Но это нужно не всегда. Чаще из соображений экономии дискового пространства RAID-массив создается только для данных, которые обрабатываются сервером (или для точек монтирования `/var` или `/home` — в зависимости от специфики сервера).

Для организации такой конфигурации вам понадобится три диска, причем первый диск будет использоваться как для установки системы, так и для подкачки.

Итак, установите систему на один из трех дисков, как обычно. Из оставшихся двух дисков создайте RAID-массив уровня 1, как было показано ранее. На каждом из этих дисков создайте по одному разделу типа Linux `raid` и объедините оба диска в один командой `mdadm` — у вас получится устройство `/dev/md0`. Что делать дальше?

Начнем с создания файловой системы. Для создания файловой системы `ext2` (`ext3/ext4` использовать нет смысла, поскольку RAID надежнее журнала) на `/dev/md0` можно использовать команду:

```
# mke2fs -b 4096 -R /dev/md0
```

ОПЦИИ КОМАНДЫ MKE2FS

Вы также можете использовать опцию `stride=8`, но только для RAID уровней 4 и 5 (она позволяет повысить производительность). Для остальных уровней ее лучше не указывать. Опция `-b` задает размер блока (в нашем случае — 4096 байтов).

Однако файловая система `ext2` безнадежно устарела, поэтому вместо нее лучше создать `ReiserFS`:

```
# mkreiserfs /dev/md0
```

Создав файловую систему, подмонтируйте устройство `/dev/md0` — например, так:

```
# mkdir /mnt/raid
# mount /dev/md0 /mnt/raid
```

Для автоматического монтирования нашего RAID-массива при каждой загрузке системы в файл `/etc/fstab` нужно добавить строку:

```
/dev/md0 /mnt/raid reiserfs defaults 0 0
```

ПРИМЕЧАНИЕ

Если вы используете файловую систему ext2, то приведенную здесь строку надо изменить так:

```
/dev/md0 /mnt/raid ext2 defaults 0 0
```

Теперь устройство /mnt/raid готово для хранения данных.

RAID-массив /dev/md0 вы можете также смонтировать как /var или как /home. Только предварительно скопируйте данные из каталога /var или /home в один из каталогов /, затем подмонтируйте /dev/md0 как /var или /home и переместите на него скопированные ранее данные.

39.5. Сбой и его имитация

Просмотреть состояние RAID-массива можно в файле /proc/mdstat:

```
md0 : active raid1 sda1[1] sdb1[0]
      45612599 blocks [2/1] [U_]
```

Здесь мы видим, что в массив md0 входят устройства sda1 и sdb1. Количество устройств — 2, из них работает 1 ([2/1]). Какое из устройств вышло из строя? Это вам подскажет индикатор [U_] — в нашем случае второй жесткий диск /dev/sdb1 вышел из строя.

Что делать дальше? Первым делом нужно подключить новое устройство. Если сервер не поддерживает горячую замену, то его придется выключить на некоторое время, — пока вы устанавливаете новый диск.

После этого удаляем отказавшее устройство из всех массивов, в которых оно участвует:

```
sudo mdadm /dev/md0 --remove /dev/sdb1
sudo mdadm /dev/md1 --remove /dev/sdb2
```

Новый диск подготавливается как обычно — с помощью команды sfdisk:

```
sudo sfdisk -d /dev/sda | sfdisk /dev/sdc
```

Осталось только добавить новое устройство в массив:

```
sudo mdadm /dev/md0 -a /dev/sdc1
sudo mdadm /dev/md1 -a /dev/sdc2
```

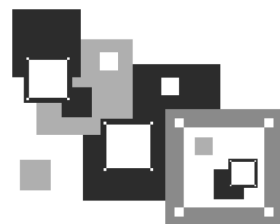
Вот и все — сервер продолжит работу, данные — в целости и сохранности на диске /dev/sda1, а при добавлении в массив второго диска они будут на него продублированы.

Если вы хотите подготовиться к выходу диска из строя, используйте команду:

```
sudo mdadm /dev/md0 --fail /dev/sdb1
```

Эта команда имитирует сбой второго жесткого диска — лучше потренироваться в ликвидации проблемы в виртуальной машине, чем искать выход при реальном сбое реального сервера.

ГЛАВА 40



Программные системы хранения данных

Каждый развивающийся проект сталкивается с выбором системы хранения данных. И любая такая система должна обеспечивать резервирование, гарантирующее сохранность данных в случае разного рода сбоев.

Очень часто при выборе хранилища данных задумываются о производительности и цене, но забывают о надежности, масштабируемости и времени восстановления после сбоя. А затем, в процессе эксплуатации, все эти факторы начинают проявлять себя.

Ранее считалось, что для обеспечения необходимой надежности нужно использовать аппаратные хранилища с резервированием вроде RAID-массивов. Поэтому с них и начнем.

ПРИМЕЧАНИЕ

Эта глава будет сугубо теоретической. Ее цель не показать, как что-то настроить, а сориентировать читателя в нужном направлении, а именно — в сторону программных систем хранения данных.

40.1. Аппаратные хранилища с резервированием

Несколько слов о том, что сейчас представляют собой аппаратные хранилища с резервированием. Как правило, это либо RAID-массивы, либо NAS (Network Attached Storage) — сетевые хранилища. Последние обычно построены на базе той же технологии RAID (о технологии RAID рассказывалось в *главе 39*). Да, в той коробочке, которая пылится в углу офиса, диски объединены в RAID-массив какого-либо уровня (обычно первого, реже — пятого).

Как технология, RAID (Redundant Array of Independent Disks) уже изжила себя. Она верой и правдой служила нам ровно 30 лет (впервые термин «RAID» прозвучал в 1987 году), но сейчас мы наблюдаем ее закат.

RAID-массивы, как мы уже знаем, представляют собой объединенные в одно целое накопители (жесткие диски, реже — SSD). Способ, которым осуществляется резервирование данных, называют *уровнем RAID*. Например, RAID 1 — это обычное

зеркалирование, когда есть два накопителя, и записываемые данные дублируются на каждый из дисков массива. Самый популярный из уровней — RAID 5. Он более экономичен — данные распределяются по всем дискам массива, минимальное количество дисков — три.

Экономичным этот уровень делает то обстоятельство, что при нем больше пространства остается именно под запись данных (особенно, по сравнению с RAID 1). Характерное свойство уровня RAID 5 заключается в том, что он обеспечивает посредственную скорость записи, зато отличное время чтения, поскольку потоки данных с нескольких накопителей массива распараллеливаются.

Основные недостатки технологии RAID и всех решений, построенных на ней, следующие:

- *плохая масштабируемость* — вы только вдумайтесь, технология создавалась 30 лет назад, когда жесткие диски стоили дорого, вращались со скоростью 3600 оборотов в минуту и позволяли записывать мегабайты (!) данных. В 1987 году типичный жесткий диск был размером 21 Мбайт. В 1997 году отличным считался размер 1-2 Гбайт. Сегодня типичный SATA-диск — это 1 Тбайт. У каждого уровня RAID есть ограничения на количество накопителей, которые можно объединить в массив. Для классического варианта RAID 1 — всего два диска. Это означает, что та коробочка в офисе уже не масштабируется, из нее выжато все. Максимум, что можно сделать, — это заменить оба жестких диска более емкими (вместо дисков в 1 Тбайт установить диски по 2 Тбайт). Для RAID 5 максимум составляет 16 дисков. Конечно, есть современные уровни вроде RAID 50 и RAID 51, позволяющие установить 60 дисков. А теперь считаем: пусть у нас есть массив RAID 50, в который мы можем установить 60 дисков по 1 Тбайт. Учитывая полезное использование емкости дисков в 67% для этого уровня, полезный объем (который можно использовать для хранения данных) составит 40 Тбайт. Дорого и нерационально;
- *время восстановления после сбоя* — представим, что у нас есть массив уровня RAID 1, состоящий из двух дисков. Если один из дисков выйдет из строя, то RAID-массив продолжит работу в аварийном режиме, ожидая замены сломавшегося диска. В это время массив уязвим, поскольку содержит одну копию данных. Как вычислить время восстановления? Это время, за которое контроллер запишет данные с рабочего диска на новый. В среднем — это 100 Мбайт/с, если контроллер не нагружен. Если массив хранит 1 Тбайт данных, то время восстановления составит 10 тыс. секунд или 2,7 часа. И это не учитывая времени, потраченного на физическую замену диска, которое может оказаться гораздо больше, чем время копирования. Например, в наличии может не оказаться нужного диска, и за ним придется куда-то поехать. А по современным меркам даже 2 часа — это много.

О ВРЕМЕНИ ВОССТАНОВЛЕНИЯ RAID 5

Отдельно хочется рассказать о скорости ребилда RAID 5. Недостатки RAID 5 проявляются при выходе из строя одного из дисков. Весь массив переходит в аварийный режим, а производительность резко падает, диски начинают греться. Если вовремя не предпринять мер, можно потерять весь массив.

Если вы никогда не сталкивались с ребилдом RAID 5, то, когда это произойдет, вы будете поражены, насколько медленным может оказаться этот процесс. Длительность процесса восстановления зависит от многих факторов: от количества дисков в массиве, от заполненности диска, а также от мощности процессора RAID-контроллера и, конечно же, от производительности самих дисков. Если же вы «счастливчик», и вам «повезло» потерять диск в разгар рабочего дня, то процесс ребилда значительно замедлится. Так, имеются сведения (<http://searchsmbstorage.techtarget.com/tip/Five-ways-to-control-RAID-rebuild-times>) о том, что на восстановление сравнительно небольшого массива из 5 дисков по 500 Гбайт ушло 24 часа.

Из всего этого можно сделать вывод, что резкий рост объемов дисков при гораздо медленном приросте скорости передачи данных с диска привел к катастрофически долгому времени восстановления RAID-массива. В результате увеличивается период времени, когда данные остаются полностью незащищенными. Резервирование якобы есть, но в то же время восстановление массива выполняется так долго, что за это время можно потерять оставшуюся часть данных. А ведь система хранения данных должна обеспечить их сохранность любой ценой.

При всем этом стоимость аппаратных RAID-решений ой как не дешева! Конечно, фанаты RAID могут привести аргумент, что можно создать программный RAID-массив, что существенно удешевит стоимость массива. Да, это так. Но в этом случае страдает производительность.

Обратите внимание, что мы постепенно смещаемся к программным решениям. Именно программные решения обеспечивают следующие преимущества:

- хорошую масштабируемость;
- приемлемую стоимость;
- высокую скорость.

Но обо всем по порядку.

40.2. Программные хранилища с резервированием

В мире аппаратных решений выбора особо нет— 30-летняя технология RAID, а также решения, построенные на базе этой технологии. По сути, выбор аппаратного решения сводится к выбору необходимого уровня RAID, а затем к покупке «железа», которое поддерживает тот или иной уровень (ну и, конечно же, к приобретению необходимого количества дисков).

А вот в случае с программными решениями выбор огромный. При этом программный RAID мы здесь рассматривать не станем, поскольку это тот же RAID, только еще более медленный, хотя и более дешевый.

Решений действительно много: Parallels Cloud Storage, MooseFS, Ceph, Lustre, GlusterFS — это только самые известные и часто используемые.

Программные решения предоставляют гораздо больше возможностей для всякого рода оптимизации. Данные распределяются по всему кластеру и по всем дискам кластера, а в случае выхода из строя одного из дисков автоматически запускается

процесс репликации. Преимущество очевидно — не нужно ждать, пока администратор заменит сломавшийся диск, не говоря уже о скорости самой репликации. На рис. 40.1 показана диаграмма времени репликации классического RAID уровня 1 и кластера PStorage, построенного на базе 7 и 14 серверов. Используются SATA-диски на 1 Тбайт, а скорость передачи данных по сети (теоретическая) составляет 1 Гбит/с.

Полагаю, результаты не нуждаются в каких-либо комментариях.

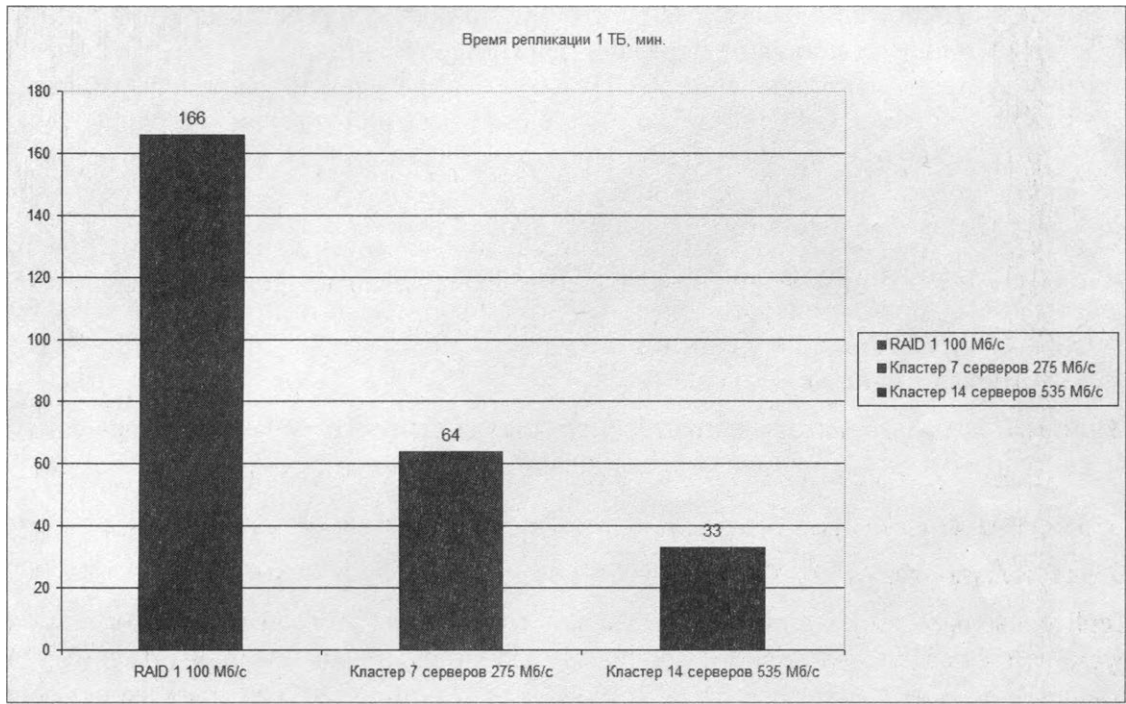


Рис. 40.1. Время репликации 1 Тбайт

If you need...	Consider...				
	Gluster	Lustre	Ceph	Cinder	Swift
NAS and Scale Out NAS	✓	✓	✓		
SAN			✓	✓ consider with Ceph Plugin or other storage systems	
Shared Filesystems	✓	✓	✓		
Object Storage	✓		✓		✓

Рис. 40.2. Сравнение разных программных систем хранения данных (иллюстрация с ресурса <http://www.yet.org/2012/12/staas/>)

Мы здесь не станем рассматривать, чем отличаются различные программные системы хранилищ. Если вам все-таки интересно, рекомендую ознакомиться с кратким сравнением (<http://www.yet.org/2012/12/staas/>), по результатам которого становится понятно, что самым универсальным вариантом на сегодняшний день является система Ceph (рис. 40.2). Она позволяет покрыть все потребности современного предприятия. Если вам нужен NAS, SAN, распределенная файловая система и хранилище объектов, то ваш выбор — Ceph.

40.3. Распределенная система хранения данных Ceph

Итак, Ceph — это распределенная система хранения данных с открытым исходным кодом, обеспечивающая высокую производительность, надежность и масштабируемость. Система Ceph хранит объекты на распределенном компьютерном кластере и предоставляет интерфейсы для хранения файлов, блоков, объектов.

Ceph построена на следующих архитектурных принципах:

- не должно быть единой точки отказа;
- система должна работать на общедоступных аппаратных средствах;
- все компоненты системы должны быть масштабируемыми;
- решение должно быть основано на открытом программном обеспечении;
- все должно быть самоуправляемым — везде, где это возможно.

Ceph позволяет достичь высокой производительности, неограниченной масштабируемости, отказаться от архаичных систем хранения данных. Ceph — это унифицированное решение для хранения данных уровня предприятия, работающее на обычных аппаратных средствах, что делает его экономически эффективной и многофункциональной системой хранения данных.

Можно долго и относительно скучно описывать архитектуру Ceph, объяснять термины, но об этом написано множество материалов, ссылки на которые будут приведены в конце этой главы.

В сопровождающем книгу электронном архиве (см. *приложение*) вы найдете два файла: 3.gif и 4.gif. Считайте, что это третья и четвертая иллюстрации к этой главе.

На рис. 3.gif показано, как работает Ceph в штатном режиме, а на рис. 4.gif — что произойдет с Ceph в случае сбоя. При выходе из строя любого узла (или диска) система Ceph не только обеспечит сохранность данных, но и сама восстановит потерянные на других узлах копии — до тех пор, пока вышедшие из строя узлы или диски не заменят рабочими. Нужно отметить, что, в отличие от RAID, ребилд происходит без секунды простоя и полностью незаметен для клиентов. Потеря одной из копий объекта приводит к переходу объекта в состояние degraded. После этого копия объекта переносится на рабочий узел и выполняется ремаппинг. Новая карта будет содержать новое расположение потерянной копии объекта. Все это выполня-

ется автоматически — без вмешательства администратора и, конечно же, незаметно для пользователя. В худшем случае задержка будет измеряться единицами секунд.

А теперь немного о производительности. Данные в Ceph квантуются очень маленькими порциями и распределяются псевдослучайно по OSD (Object Storage Device) — устройству хранения объектов. В результате реальный ввод/вывод каждого клиента система равномерно «размазывает» по всем дискам кластера, что позволяет снизить конкуренцию между клиентами за дисковый ресурс (в качестве бонуса клиент получает существенно большие лимиты по пропускной способности и IOPS).

Секрет быстродействия Ceph кроется также и в журналах. Все операции записи сначала попадают в журнал OSD, а затем, не задерживая клиента, асинхронно переносятся в постоянное хранилище. Именно поэтому рекомендуется размещение журнала на SSD, что в несколько раз ускоряет операции записи.

Нужно понимать, что Ceph (<http://ceph.com/>) — система свободная и бесплатная. Но некоторые компании, в том числе и Red Hat, могут продавать собственные решения, построенные на базе Ceph. Вы можете выбрать уже готовое решение, например, Red Hat Ceph Storage, а можете построить собственный кластер, и это обойдется гораздо дешевле (особенно, учитывая, что большая часть оборудования у вас уже есть).

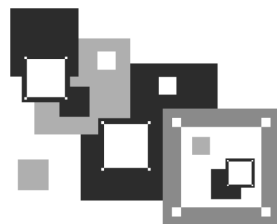
Итак, преимущества внедрения Ceph выглядят так:

- аппаратный отказ узла (диска) незаметен для пользователей, как событие;
- время простоя при отказе равно 0;
- нет единой точки отказа;
- снижения задержки ввода/вывода до уровня SSD-диска;
- доступны снимки, клонирование, инкрементные диффы;
- сверхнизкие цены на услуги благодаря возможности работы на обычном оборудовании.

40.4. Дополнительные материалы

- Подборка слайдов по производительности Ceph:
<http://slides.com/sebastienhan/ceph-performance-and-benchmarking/#>.
- Каран Сингх. Изучаем Ceph. — «Packt Publishing»:
<http://onreader.mdl.ru/LearningCeph/content/index.html>.
- Benchmark Ceph Cluster Performance:
http://tracker.ceph.com/projects/ceph/wiki/Benchmark_Ceph_Cluster_Performance.
- Описание использования Red Hat Ceph:
<http://habrahabr.ru/company/ua-hosting/blog/329618/>.

ГЛАВА 41



Средства резервного копирования. Создание ISO-образа диска

41.1. Необходимость в «живой» резервной копии

Многие из нас помнят «привидение от Нортона» — Norton Ghost®. В мире Windows — это незаменимый продукт. Здесь же мы поговорим о средствах резервного копирования для Linux — они позволяют создать не просто резервную копию системы, а LiveCD/DVD. Да, эти средства способны заменить Norton Ghost, причем абсолютно бесплатны в отличие от этого продукта Symantec.

Для начала определимся, зачем нужны средства создания LiveCD. Наша цель — резервное копирование системы, но причем здесь LiveCD? Оказывается, это очень удобно — мы убиваем вот сколько зайцев сразу:

- **создаем средство для восстановления системы** — предположим, что вы настроили свою систему, «подняли» все сетевые службы, отредактировали их конфигурационные файлы. Но завтра из-за очередного перепада напряжения сторел жесткий диск. Опять все заново настраивать? Если же вы накануне создали LiveCD, то вам нечего беспокоиться: заменили жесткий диск, загрузились с LiveCD (смотря правде в глаза и учитывая размер системы, у нас будет LiveDVD, но по старинке мы здесь и далее будем называть его LiveCD) и установили систему вместе со всеми параметрами на новый винчестер. И все! На всю эту операцию будет потрачено полчаса, от силы минут 40 вместе с установкой нового жесткого диска. Пользователи и начальство будут вам благодарны за столь оперативное «воскрешение» сервера. А теперь представьте, что вы создали обычный «бэкап» с помощью программ tar/tgz, — вам понадобится минимум 40 минут на установку системы, потом время на восстановление резервной копии плюс одна лишняя перезагрузка. Однозначно времени будет потрачено больше;
- **создаем средство для клонирования системы** — когда предприятие организует компьютерный парк, то, как правило, приобретаются однотипные компьютеры. Исключение составляют разве что серверы (они должны быть мощнее) и компьютеры начальства (на них должна стоять мощная видеокарта ☺). Вот теперь представьте, что вам нужно настроить каждый новый компьютер. А их 10,

20, 50! Можно поступить проще — настроить один компьютер, создать LiveCD и развернуть его на всех остальных компьютерах сети. Пусть настройка одного компьютера вместе с установкой системы займет полтора часа, создание ISO-образа системы — еще минут 30 (тут все зависит от производительности компьютера, потому что от вас потребуется ввести всего одну команду), затем запись этого образа на нужное количество болванок. Да, именно на «количество», потому что вам надо будет создать несколько копий LiveCD, чтобы можно было одновременно устанавливать систему на несколько компьютеров. Затем еще минут 40 ожидания, и будет настроено несколько компьютеров сразу — по числу имеющихся болванок. Удобно? Думаю, да. Без LiveCD вы бы потратили полтора часа на каждый компьютер. 10 компьютеров — это 15 часов (два рабочих дня). А так будет потрачено примерно 4 часа. Созданные «клоны» системы можно использовать и в будущем, если компьютерный парк будет расширяться;

- **приобретаем возможность создания LiveUSB** — загрузочная «живая» флешка понадобится для восстановления/клонирования ОС нетбука и других компьютеров, не имеющих привода DVD. Средства создания LiveCD позволяют создать и такую флешку.

Не нужно думать, что «бэкап» с помощью LiveCD может использоваться только для копирования/восстановления файлов самой системы. Можно копировать на диски и пользовательские данные из каталога `/home` — лишь бы их размер не превысил размера болванки DVD. Впрочем, увеличить объем резервируемой информации позволит использование двухслойных дисков (двухсторонние диски не удобны в эксплуатации).

41.2. Средства клонирования Linux

Самым мощным средством для клонирования Linux является Clonezilla. Этот продукт может не только создать LiveCD, но и развернуть систему по сети. На сайте разработчиков <http://clonezilla.org/> имеется следующая информация: за 10 минут Clonezilla SE (SE, Server Edition) развернула по сети образ объемом 5,6 Гбайт на 41 компьютер сети. Таким образом, все компьютеры были настроены всего за 10 минут. Правда, для подобной сетевой установки придется развернуть специальный сервер, но об этом позже. Кроме того, Clonezilla может использоваться для создания резервных копий компьютеров, работающих под управлением Windows и FreeBSD.

REMASTERSYS BACKUP

В 4-м издании этой книги был описан замечательный инструмент Remastersys Backup, позволяющий делать резервные копии любых Debian-совместимых дистрибутивов (в том числе Ubuntu и всех ее клонов). Однако что-то стряслось у разработчиков, и они решили прекратить свой проект, — сейчас Remastersys Backup не поддерживается, и его репозиторий закрыт. Сожалею, но это так. Если спустя некоторое время проект «оживет», я с удовольствием расскажу вам о нем.

Любителям Slackware подойдет скрипт Linux Live (<http://www.linux-live.org/>) — он позволяет создать как LiveCD, так и LiveUSB.

Подобные утилиты можно найти и для других дистрибутивов, но рассматривать их все не вижу смысла, — мы познакомимся здесь с универсальным средством Clonezilla и дистрибутивно-ориентированным Linux Live.

41.3. Clonezilla

Основные особенности Clonezilla:

- полностью бесплатна (распространяется по лицензии GPL);
- поддерживает файловые системы ext2, ext3, ext4, reiserfs, reiser4, xfs, jfs, FAT, NTFS, HFS (Mac OS), UFS (FreeBSD, NetBSD, OpenBSD), VMFS (VMware ESX), поэтому вы можете клонировать не только Linux, но и MS Windows, Mac OS (Intel), FreeBSD, NetBSD и OpenBSD;
- поддерживает LVM2 (LVM ver 1 не поддерживает);
- поддерживает GRUB версий 1 и 2;
- поддерживает Multicast для массового клонирования по сети (версия Clonezilla Server Edition при условии, что компьютеры поддерживают PXE и Wake-on-LAN);
- может сохранить не только отдельно взятый раздел, но и весь жесткий диск со всеми разделами.

Clonezilla — программа не простая, и здесь мы рассмотрим лишь один из примеров ее использования, а именно — создание LiveCD и восстановление системы с его помощью. Познакомиться же с остальными возможностями программы можно в документации к ней или на сайте разработчиков.

Итак, для создания/восстановления образа системы нужно выполнить следующие действия:

1. Скачать с сайта <http://clonezilla.org/download/sourceforge/> ISO-образ Clonezilla Live и записать его на болванку.
2. Загрузиться с болванки Clonezilla Live (рис. 41.1), выбрав команду **Clonezilla live**. Вы увидите процесс загрузки Debian (рис. 41.2)— тут все, как обычно, — нужно просто подождать. Если возникнут проблемы (например, с видеокартой), можно выбрать команду **Other modes of Clonezilla live** и попробовать другой режим загрузки Clonezilla.
3. Далее (рис. 41.3) нужно выбрать язык (русского, к сожалению, пока не предвидится). Можно также выбрать и раскладку клавиатуры (рис. 41.4), но поскольку раскладку изменять нам ни к чему, выберите вариант **Don't touch keymap**.
4. Выбрать команду **Start Clonezilla** (рис. 41.5).
5. Выбрать режим **device-image** для создания файла образа диска или раздела (рис. 41.6). Режим **device-device** служит для создания копии диска или раздела на другом диске (разделе) без создания файла образа.
6. Выбрать режим **local_dev** — локальное устройство, куда будет сохранен образ или откуда он будет прочитан в случае восстановления системы по образцу

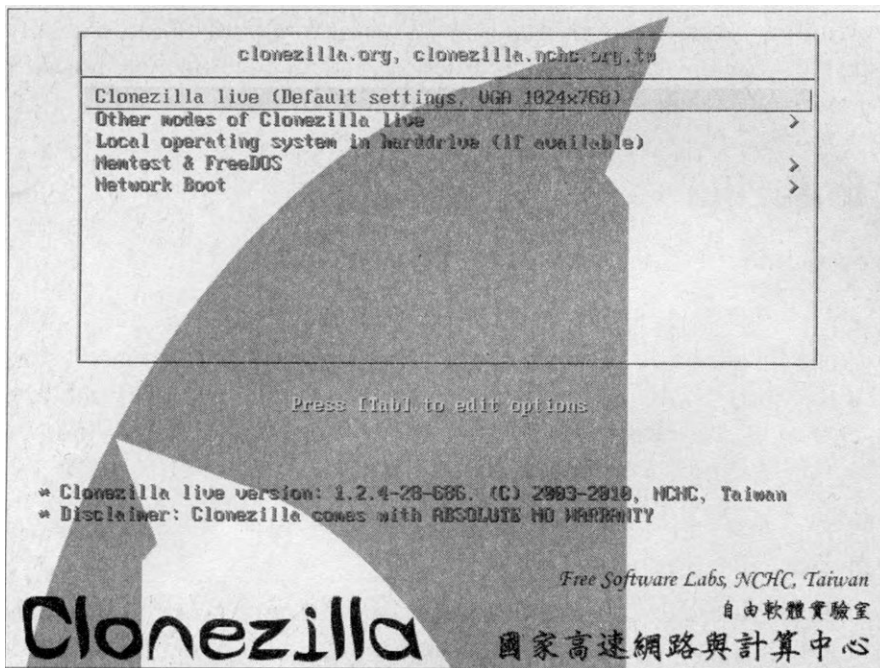


Рис. 41.1. Загрузочное меню Clonezilla Live

```
[ 2.298874] scsi 1:0:1:0: Direct-Access    ATA          VMware Virtual I 0000 PQ: 0 ANSI: 5
[ 2.340195] ata2.00: ATAPI: VMware Virtual IDE CDROM Drive, 00000001, max UDMA/33
[ 2.341583] ata2.00: configured for UDMA/33
[ 2.342129] scsi 2:0:0:0: CD-ROM             NECUMwar VMware IDE CDROM 1.00 PQ: 0 ANSI: 5
[ 2.350556] sr0: scsi3-mmc drive: 1x/1x xaforn2 cdda tray
[ 2.352065] Uniform CD-ROM driver Revision: 3.20
[ 2.358812] sd 1:0:0:0: [sda] 16777216 512-byte logical blocks: (8.58 GB/8.00 GiB)
[ 2.359612] sd 1:0:0:0: [sda] Write Protect is off
[ 2.361466] sd 1:0:0:0: [sda] Write cache: disabled, read cache: enabled, doesn't support DPO or
FUA
[ 2.362200] sda: sda1 sda2 sda3 < sda5 >
[ 2.363092] sd 1:0:1:0: [sdb] 31457280 512-byte logical blocks: (16.1 GB/15.0 GiB)
[ 2.363185] sd 1:0:1:0: [sdb] Write Protect is off
[ 2.363228] sd 1:0:1:0: [sdb] Write cache: disabled, read cache: enabled, doesn't support DPO or
FUA
[ 2.380613] sdb: sdb1
[ 2.386360] sd 1:0:1:0: [sdb] Attached SCSI disk
[ 2.387994] sd 1:0:0:0: [sda] Attached SCSI disk
[ 2.391994] sd 1:0:0:0: Attached scsi generic sg0 type 0
[ 2.393897] sd 1:0:1:0: Attached scsi generic sg1 type 0
[ 2.400551] sr 2:0:0:0: Attached scsi generic sg2 type 5
Begin: Loading essential drivers ... [ 2.593615] Atheros(R) L2 Ethernet Driver - version 2.2.3
[ 2.593830] Copyright (c) 2007 Atheros Corporation.
[ 2.612151] Broadcom NetXtreme II 5771x 10Gigabit Ethernet Driver bnx2x 1.52.1 (2009/08/12)
[ 2.632202] device-mapper: uevent: version 1.0.3
[ 2.634009] device-mapper: ioctl: 4.15.0-ioctl (2009-04-01) initialised: dm-devel@redhat.com
done.
Begin: Running /scripts/init-premount ... done.
Begin: Mounting root file system ... [ 2.745155] Uniform Multi-Platform E-IDE driver
[ 2.745836] ide_generic: please use "probe_mask=0x3f" module parameter for probing all legacy ISA
IDE ports
[ 2.882403] aufs: module is from the staging directory, the quality is unknown, you have been war
ned.
[ 2.885440] aufs 2-standalone.tree-32-20100125
[ 2.930106] loop: module loaded
[ 3.041203] squashfs: version 4.0 (2009/01/31) Phillip Lougher
```

Рис. 41.2. Debian: процесс загрузки

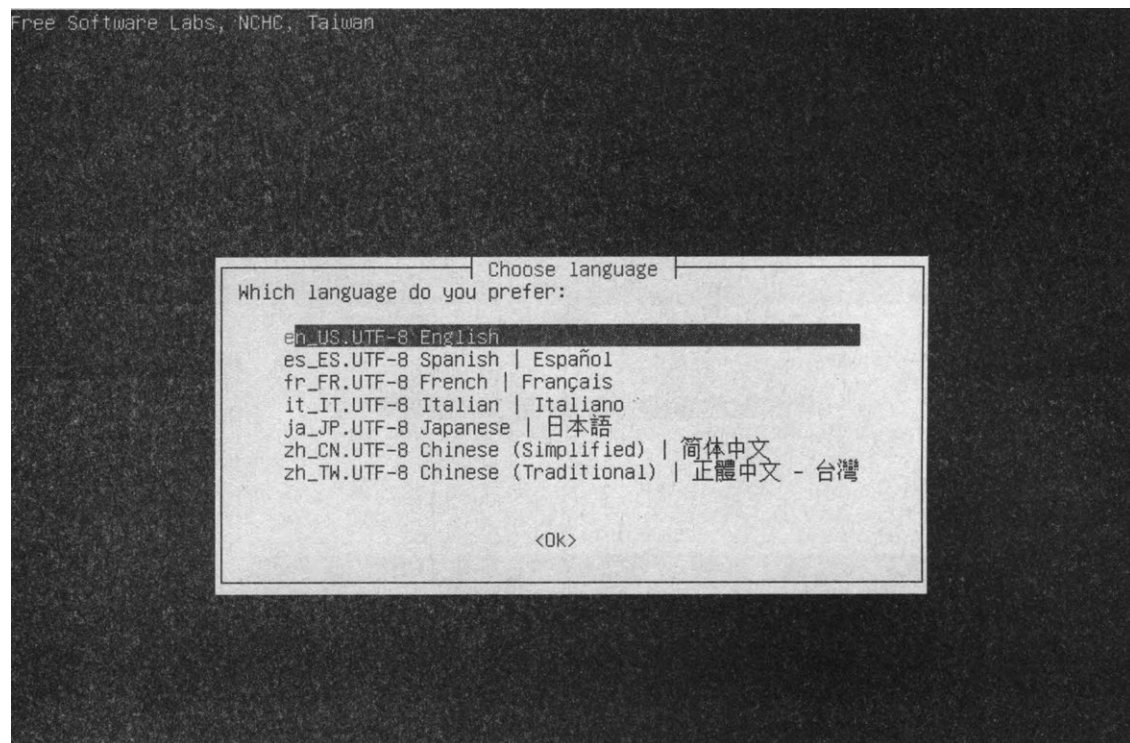


Рис. 41.3. Выбор языка Clonezilla

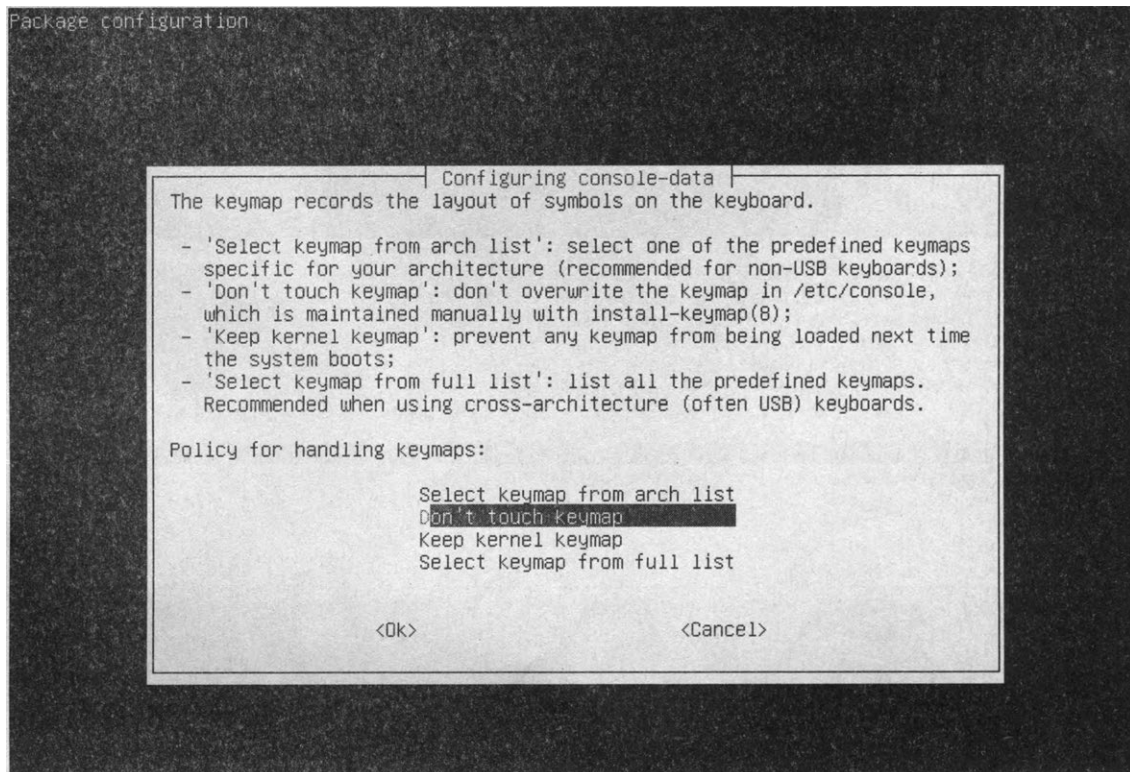


Рис. 41.4. Выбор раскладки

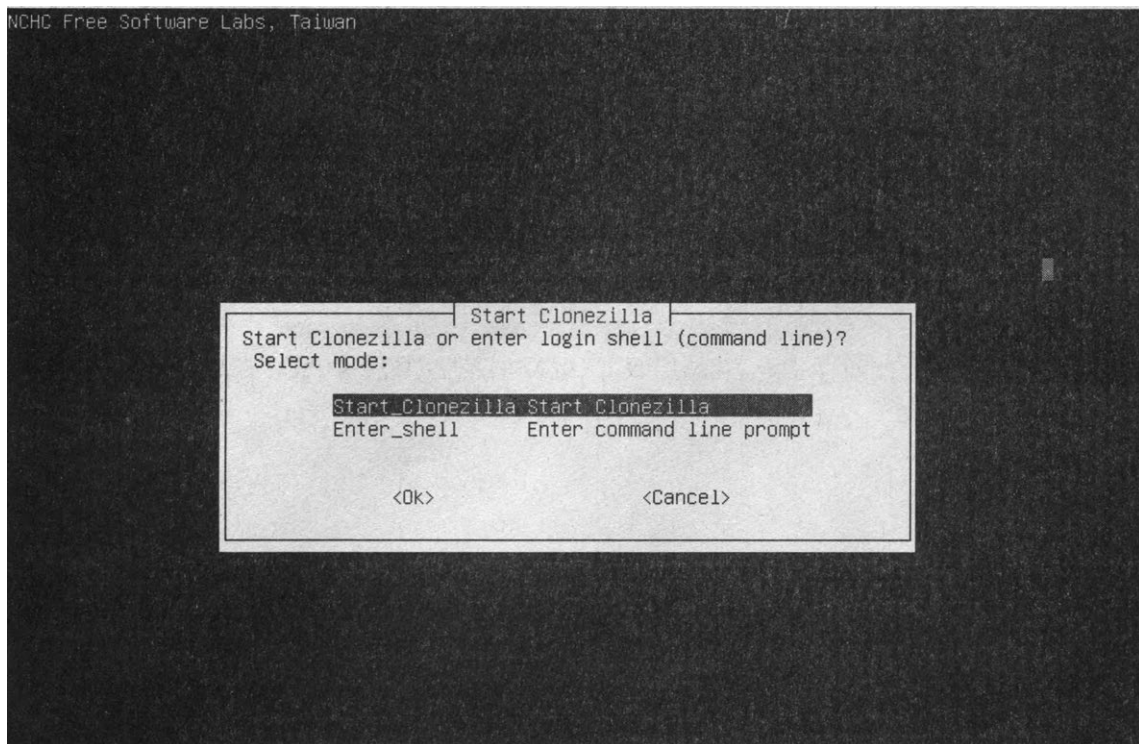


Рис. 41.5. Выберите команду Start Clonezilla

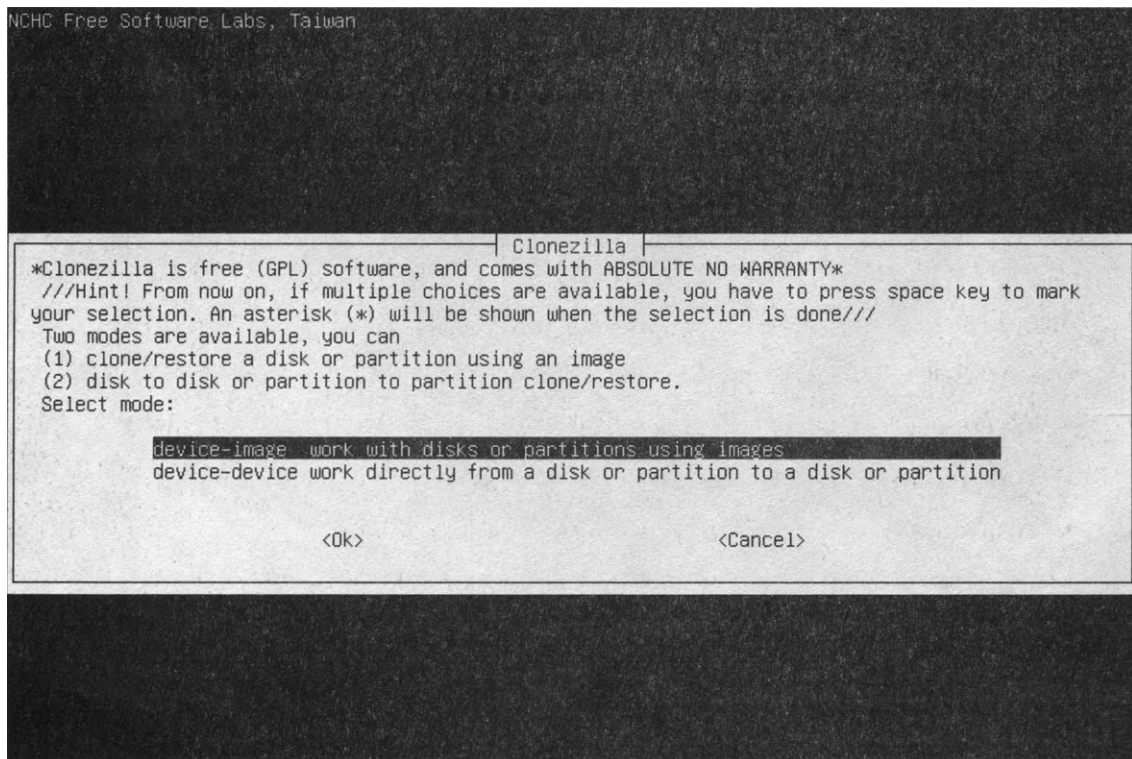


Рис. 41.6. Выберите режим device-image

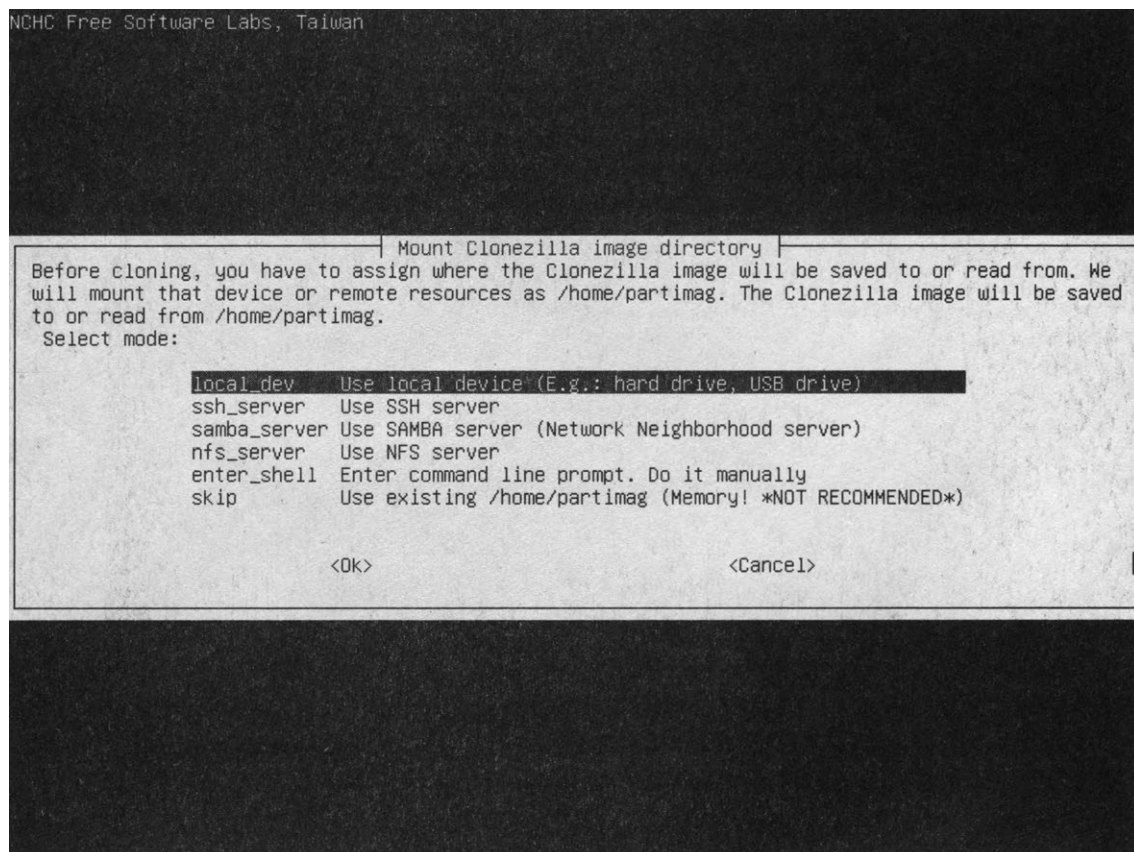


Рис. 41.7. Выбор носителя образа

(рис. 41.7). Образ также можно получить (или записать) по SSH, NFS (Network File System, а не Need For Speed!) и из сети MS Windows (**samba_server**).

7. Выбрать раздел, где будут храниться образы: если вы создаете образ, то на этот раздел он будет сохранен, а если восстанавливаете, то Clonezilla будет искать его на этом разделе.
8. Выбрать одну из команд (рис. 41.8):
 - **savedisk** — для сохранения всего диска;
 - **saveparts** — для сохранения одного или нескольких разделов диска;
 - **restoredisk** — для восстановления образа диска на локальный диск;
 - **restoreparts** — для восстановления образа раздела;
 - **recovery-iso-zip** — для создания «живого» диска восстановления.
9. Если вы выбрали команду восстановления образа, то далее следует выбрать образ, который нужно для этого взять (рис. 41.9).
10. Ввести устройство (имена устройств соответствуют именам устройств в Linux), на которое нужно развернуть образ (рис. 41.10). Будьте внимательны, чтобы не развернуть образ раздела на весь диск, — потеряете остальные разделы!

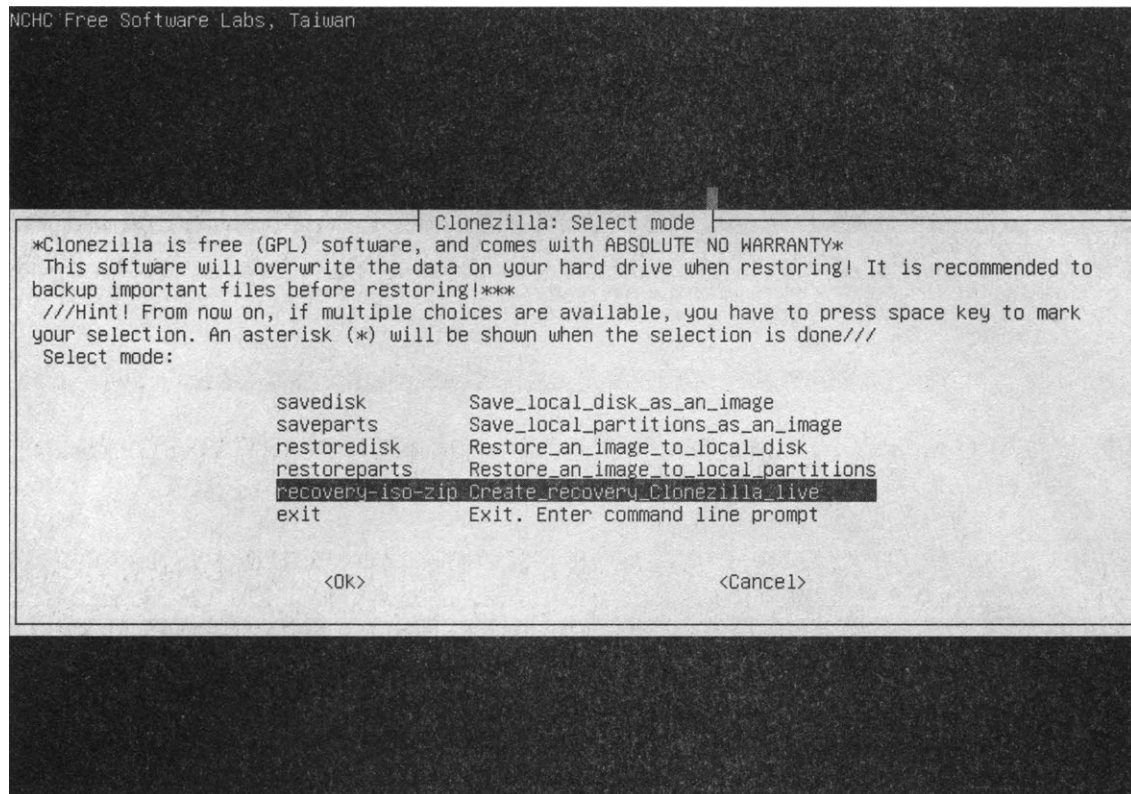


Рис. 41.8. Создать образ или восстановить?

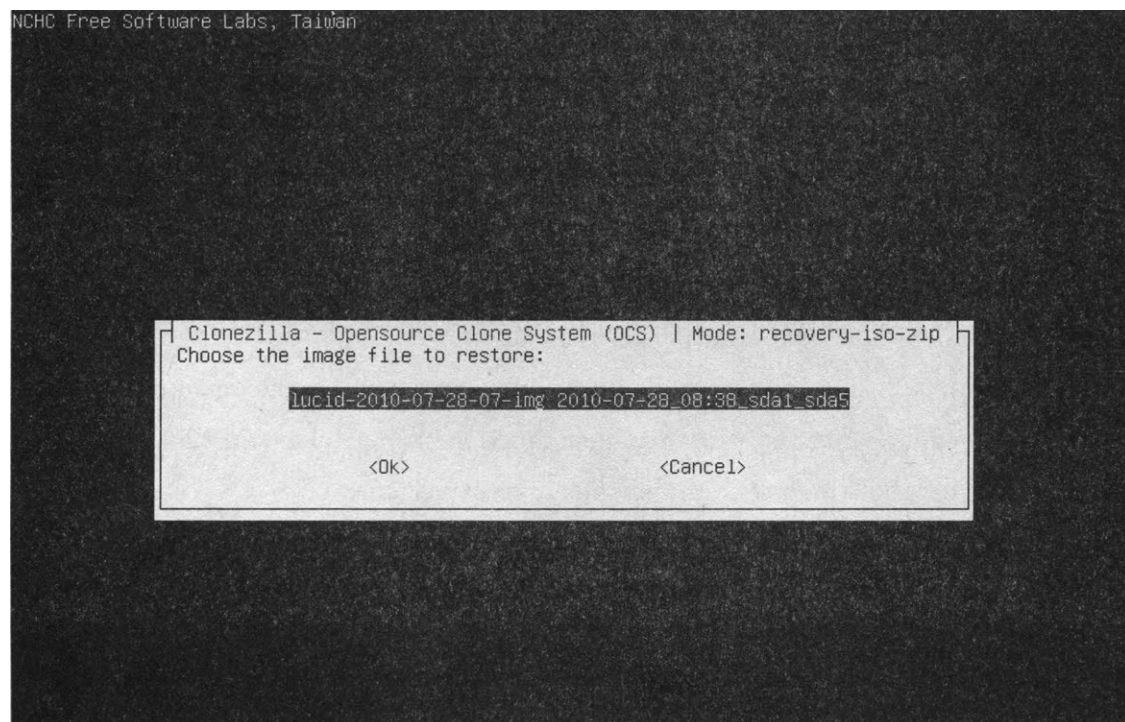


Рис. 41.9. Выбор образа для восстановления

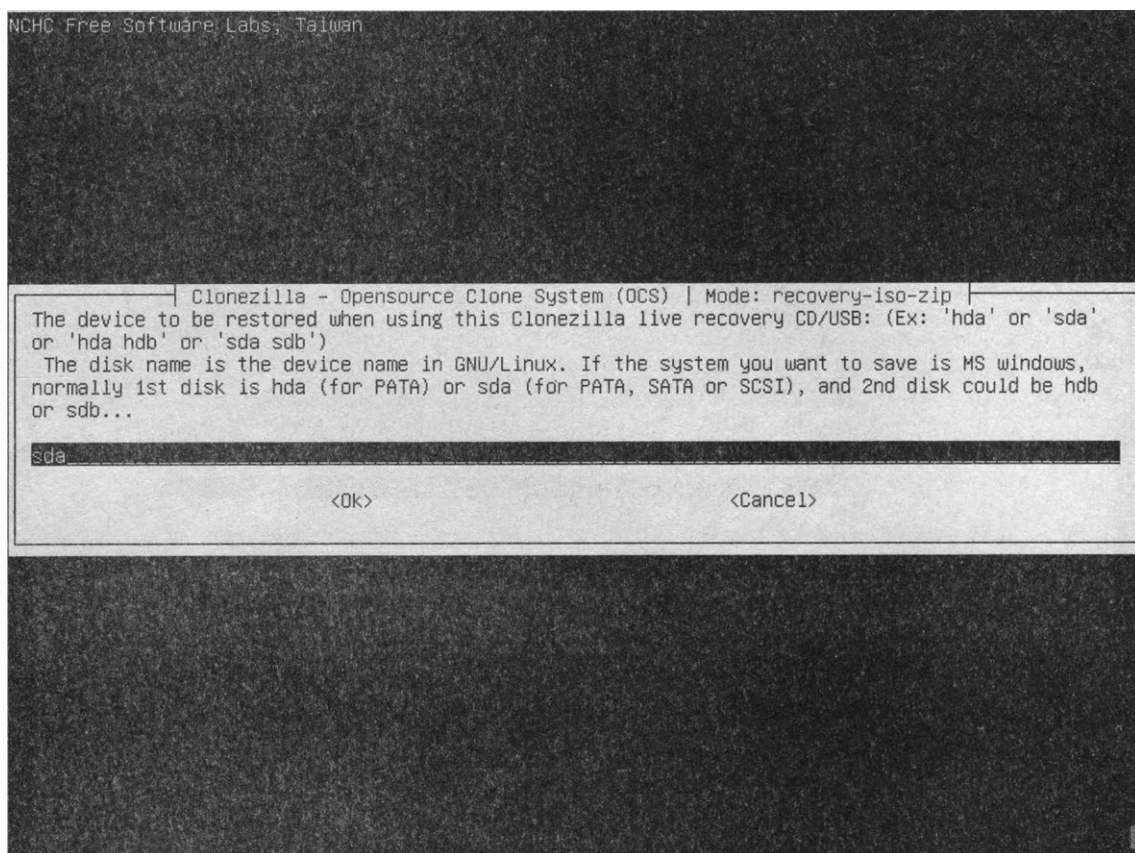


Рис. 41.10. На какое устройство «развернуть» образ

11. Если вы выбрали команду **recovery-iso-zip** (см. рис. 41.8) для создания LiveDVD/USB, то нужно также выбрать режим (рис. 41.11):

- **iso** — будет создан образ для записи на DVD;
- **zip** — образ для записи на LiveUSB;
- **both** — будут созданы оба файла, которые можно использовать впоследствии как для создания LiveDVD, так и для создания LiveUSB. Созданный файл (файлы) будет сохранен в каталоге `/home/partimag` (рис. 41.12).

На рис. 41.13 показан процесс создания LiveCD, а из рис. 41.14 видно, что этот процесс успешно завершен.

Вот и все! Как видите, это довольно просто. Программа работает с устройствами (дисками, разделами) напрямую, поэтому при создании/восстановлении образа все равно, под какой операционной системой работает компьютер.

Если у вас есть необходимость в серверной версии (Clonezilla Server Edition), найти руководство по ее использованию вы можете по адресу: <http://clonezilla.org/clonezilla-server-edition/>.

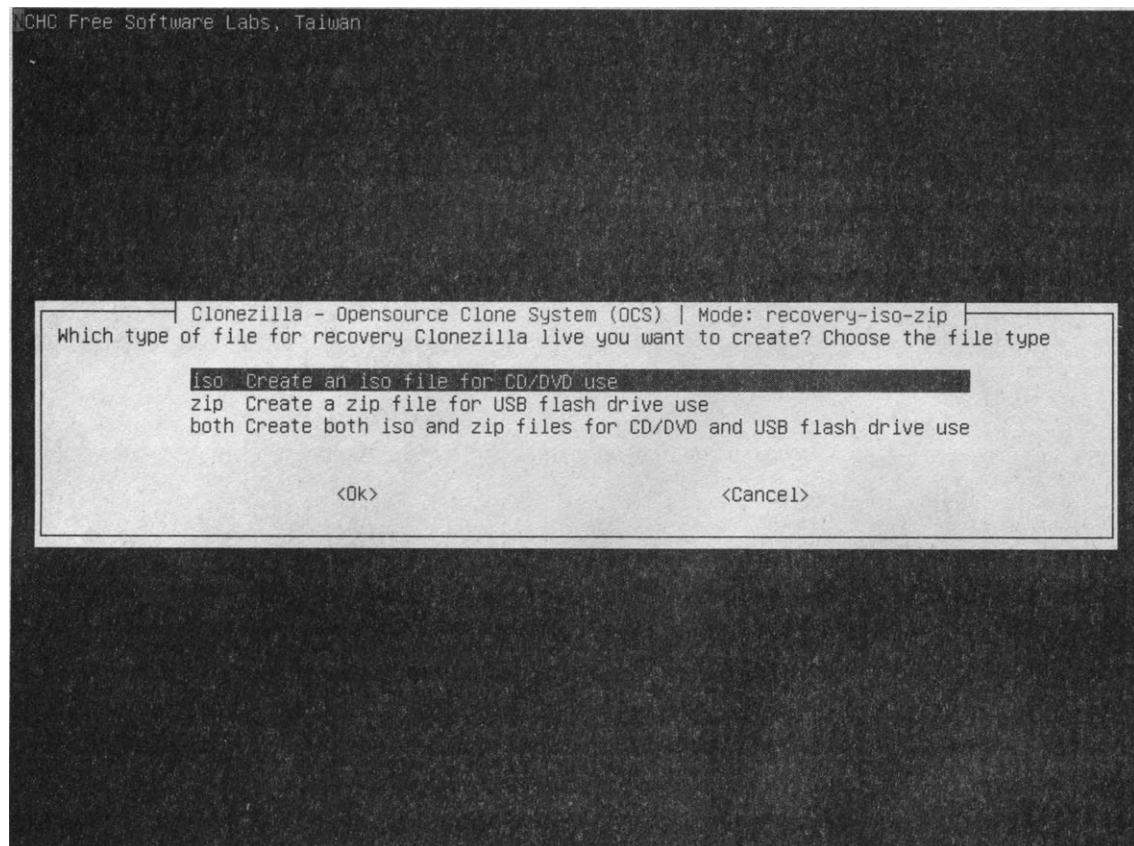


Рис. 41.11. Выбор режима команды recovery-iso-zip

```
PS. Next time you can run this command directly:
ocs-iso -g en_US.UTF-8 -t -k NONE -e "-g auto -e1 auto -e2 -c -r -j2 -p true restoredisk lucid-2010-
07-28-07-img sda" lucid-2010-07-28-07-img
This command is also saved as this file name for later use if necessary: /tmp/ocs-iso-zip-2010-07-28-
09-04
The output iso/zip file will be in this dir: /home/partimag
Press "Enter" to continue..._
```

Рис. 41.12. Созданный файл будет сохранен в каталоге /home/partimag

```

PS. Next time you can run this command directly:
ocs-iso -g en_US.UTF-8 -t -k NONE -e "g auto -e1 auto -e2 -c -r -j2 -p true restoredisk lucid-2010-
07-28-07-img sda" lucid-2010-07-28-07-img
This command is also saved as this file name for later use if necessary: /tmp/ocs-iso-zip-2010-07-28
-09-04
The output iso/zip file will be in this dir: /home/partimag
Press "Enter" to continue...
Found a Clonezilla live media... Will use that as a template...
Creating clonezilla ISO with image(s) lucid-2010-07-28-07-img from /home/partimag...
The output file name is: clonezilla-live-lucid-2010-07-28-07-img.iso.
Copying the system files to working dir... This might take a few minutes... done!
Estimated target ISO file "clonezilla-live-lucid-2010-07-28-07-img.iso" size: 338 MB
Trying to find the boot params from template live cd...
Adding isolinux menus for clonezilla live with img lucid-2010-07-28-07-img...
Adding syslinux menus for clonezilla live with img lucid-2010-07-28-07-img...
Preparing syslinux, syslinux.exe, makeboot.sh, and makeboot.bat in dir utils...
Warning: -follow-links does not always work correctly; be careful.
I: -input-charset not specified, using utf-8 (detected in locale settings)
Size of boot image is 4 sectors -> No emulation
 2.91% done, estimate finish Wed Jul 28 09:05:51 2010
 5.81% done, estimate finish Wed Jul 28 09:05:51 2010
 8.72% done, estimate finish Wed Jul 28 09:05:51 2010
11.62% done, estimate finish Wed Jul 28 09:05:51 2010
14.53% done, estimate finish Wed Jul 28 09:05:51 2010
17.43% done, estimate finish Wed Jul 28 09:05:56 2010
20.34% done, estimate finish Wed Jul 28 09:05:55 2010
23.24% done, estimate finish Wed Jul 28 09:05:55 2010
26.15% done, estimate finish Wed Jul 28 09:05:54 2010
29.05% done, estimate finish Wed Jul 28 09:05:54 2010
31.96% done, estimate finish Wed Jul 28 09:05:54 2010
34.87% done, estimate finish Wed Jul 28 09:05:56 2010
37.77% done, estimate finish Wed Jul 28 09:05:58 2010

```

Рис. 41.13. Процесс создания LiveCD

```

55.19% done, estimate finish Wed Jul 28 09:06:36 2010
58.10% done, estimate finish Wed Jul 28 09:06:35 2010
61.00% done, estimate finish Wed Jul 28 09:06:35 2010
63.91% done, estimate finish Wed Jul 28 09:06:34 2010
66.81% done, estimate finish Wed Jul 28 09:06:34 2010
69.72% done, estimate finish Wed Jul 28 09:06:35 2010
72.62% done, estimate finish Wed Jul 28 09:06:36 2010
75.53% done, estimate finish Wed Jul 28 09:06:39 2010
78.43% done, estimate finish Wed Jul 28 09:06:47 2010
81.34% done, estimate finish Wed Jul 28 09:06:47 2010
84.24% done, estimate finish Wed Jul 28 09:06:46 2010
87.15% done, estimate finish Wed Jul 28 09:06:46 2010
90.05% done, estimate finish Wed Jul 28 09:06:45 2010
92.96% done, estimate finish Wed Jul 28 09:06:44 2010
95.86% done, estimate finish Wed Jul 28 09:06:44 2010
98.77% done, estimate finish Wed Jul 28 09:06:43 2010
Total translation table size: 2048
Total rockridge attributes bytes: 6390
Total directory bytes: 22528
Path table size(bytes): 168
Max brk space used 12000
172125 extents written (336 MB)
Cleaning tmp dirs...
Isohybridizing clonezilla-live-lucid-2010-07-28-07-img.iso... done!
You can burn this iso file onto a CD/DVD and then use it to boot other machines to use Clonezilla: c
lonezilla-live-lucid-2010-07-28-07-img.iso
*****
If you want to use Clonezilla again:
(1) Stay in this console (console 1), enter command line prompt
(2) Run command "exit" or "logout"
*****
When everything is done, remember to use 'poweroff', 'reboot' or follow the menu to do a normal powe
roff/reboot procedure. Otherwise if the boot media you are using is a writable device (such as USB f
lash drive), and it's mounted, poweroff/reboot in abnormal procedure might make it FAIL to boot next
time!
*****
Press "Enter" to continue...

```

Рис. 41.14. LiveCD создан, нажмите клавишу <Enter> для продолжения

41.4. Linux Live

Теперь очередь дошла и до Slackware. Это очень хороший дистрибутив, пусть, может, и не такой удобный как Ubuntu, зато весьма надежный. Для создания LiveCD в Slackware мы воспользуемся уже упомянутым ранее скриптом Linux Live. На сайте этого проекта (<http://www.linux-live.org/>) указано, что для работы с Linux Live рекомендуется дистрибутив Slackware, — именно рекомендуется, а не требуется, т. е., теоретически, Linux Live подойдет и для любого другого дистрибутива).

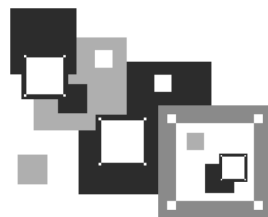
Итак, выполните следующие действия:

1. Соберите (если этого еще не сделано) модули ядра: aufs и squashfs. Все необходимое для сборки (модули aufs, squashfs и lzma) вы найдете на сайте Linux Live.

Нужно отметить, что проект Linux Live исправился — если ранее он ориентировался на версию ядра 2.6.27, когда даже в Slackware уже была версия 2.6.33, то сейчас он предлагает ядро версии 3.6.2, — довольно-таки современную его версию (да, ядро это было выпущено в конце 2012 года, но многие современные версии дистрибутивов еще используют его).

2. Удалите все лишнее — например, лишнюю документацию и лишние программы, — чтобы уменьшить размер дистрибутива.
3. Скачайте скрипты Linux Live с <http://www.linux-live.org/> и распакуйте их в каталог /tmp.
4. Отредактируйте файл .config, если нужно изменить переменные по умолчанию.
5. Запустите файл ./build (находится в каталоге /tmp) с правами root— в результате образуется каталог с данными LiveCD: /tmp/live_data_NNNN, где NNNN — случайное число.
6. Выполните команду `make_iso.sh` для создания ISO-образа или `bootinst.sh` — для создания LiveUSB.

ГЛАВА 42



Шифрование файловой системы

В последнее время наблюдается модная тенденция шифроваться. Шифруют абсолютно все: файлы, папки, создают криптоконтейнеры, шифруют даже Android-смартфоны.

В этой главе мы поговорим о шифровании файловой системы в Linux, и осуществим мы его стандартными средствами — посредством криптографической файловой системы eCryptfs. Различные сторонние решения, которых вполне достаточно (взять тот же TrueCrypt или его форк CipherShed), мы здесь рассматривать не станем — стандартные средства шифрования столь же надежны и ничем не хуже того же TrueCrypt, — ведь они, как и TrueCrypt, используют алгоритм AES, ставший де-факто стандартом шифрования (хотя позволяют воспользоваться и другими алгоритмами шифрования).

ПРОЕКТ TRUECRYPT

Пользуясь моментом и «служебным положением», хочу выразить свое мнение относительно проекта TrueCrypt. Этот проект ни разу не был никем скомпрометирован, и, видимо, это кого-то очень сильно напугало. И проект закрыли. Если бы даже в TrueCrypt и была найдена уязвимость, на что намекает его официальный сайт, это было бы поводом для выпуска новой версии, но никак не для закрытия проекта, да еще и с рекомендацией перейти на проприетарный BitLocker, который разработчики TrueCrypt высмеивали ранее...

42.1. Шифрование папки

Для знакомства с криптографической файловой системой eCryptfs мы сейчас зашифруем мой домашний каталог (/home/den). Причин шифрования данных может быть много, но у меня сейчас причина одна — сугубо академический интерес: попробовать и вам рассказать.

ШИФРОВАНИЕ СРЕДСТВАМИ ДИСТРИБУТИВА

Шифрование файловой системы с помощью средств, включенных в дистрибутив, описано в *главе 2*.

Прежде всего установите утилиты eCryptfs. Поскольку работаю я сейчас в дистрибутиве Debian, то для их установки буду использовать apt-get:

```
sudo apt-get install ecryptfs-utils
```

Перед шифрованием домашнего каталога на всякий случай сделаем его резервную копию — мало ли чего:

```
sudo cp -pfr /home/den /tmp
```

Чтобы зашифровать каталог, нужно его подмонтировать, указав тип файловой системы `ecryptfs`:

```
sudo mount -t ecryptfs /home/den /home/den
```

Вывод будет таким (угловыми скобками здесь выделено то, что нужно сделать вам):

Passphrase: <введите секретную фразу>

Select cipher:

- 1) aes: blocksize = 16; min keysize = 16; max keysize = 32 (not loaded)
- 2) blowfish: blocksize = 16; min keysize = 16; max keysize = 56 (not loaded)
- 3) des3_ede: blocksize = 8; min keysize = 24; max keysize = 24 (not loaded)
- 4) twofish: blocksize = 16; min keysize = 16; max keysize = 32 (not loaded)
- 5) cast6: blocksize = 16; min keysize = 16; max keysize = 32 (not loaded)
- 6) cast5: blocksize = 8; min keysize = 5; max keysize = 16 (not loaded)

Selection [aes]: <просто нажмите Enter (aes по умолчанию)>

Select key bytes:

- 1) 16
- 2) 32
- 3) 24

Selection [16]: <нажмите Enter>

Enable plaintext passthrough (y/n) [n]: <n>

Enable filename encryption (y/n) [n]: <n>

Attempting to mount with the following options:

ecryptfs_unlink_sigs

ecryptfs_key_bytes=16

ecryptfs_cipher=aes

ecryptfs_sig=bd28c38da9fc938b

WARNING: Based on the contents of [/root/.ecryptfs/sig-cache.txt], it looks like you have never mounted with this key before. This could mean that you have typed your passphrase wrong.

Would you like to proceed with the mount (yes/no)? : <yes>

Would you like to append sig [bd28c38da9fc938b] to [/root/.ecryptfs/sig-cache.txt]

in order to avoid this warning in the future (yes/no)? : <yes>

Successfully appended new sig to user sig cache file

Mounted eCryptfs

Мы здесь согласились на использование алгоритма по умолчанию (AES). Если вы считаете, что другой алгоритм лучше, можете выбрать его. Мы также отказались от

шифрования имен файлов (Enable filename encryption)— если что-то случится с зашифрованным каталогом, то разобраться, где и какой файл, будет сложно.

Итак, каталог `/home/den` зашифрован. Восстановим наш бэкап и удалим его (чтобы никто не смог его прочитать):

```
sudo cp -pfr /tmp/den /home/  
sudo rm -fr /tmp/den
```

Осталось самое главное— проверить, а зашифрован ли на самом деле каталог? Попробуем скопировать в него любой файл — например, `/etc/motd` — из незашифрованной файловой системы:

```
cp /etc/motd /home/den
```

Размонтируем зашифрованный каталог:

```
sudo umount /home/den
```

Теперь пробуем прочитать файл `/home/den/motd`:

```
cat /home/den/motd
```

Если вы увидите всякого рода иероглифы и абракадабру— значит, шифрование работает.

42.2. Храним пароль на флешке

Шифрование работает, но каждый день (точнее, после каждой перезагрузки/загрузки системы) вам надоеет вводить секретную фразу, — нужно позаботиться об автоматическом монтировании. Но где будет храниться пароль? На жестком диске? Но тогда нет смысла в самом шифровании — это все равно, что написать пароль на желтой бумажке, приклеенной к монитору.

Впрочем, мы, как обычно, найдем рациональное решение — станем хранить секретную фразу на флешке. Однако на флешке с файловой системой FAT32 секретная фраза будет храниться в незашифрованном виде, поэтому постарайтесь, чтобы флешка не попала к врагу. Двойное шифрование (т. е. и домашнего каталога, и флешки) возможно, но его описание выходит за рамки этой книги.

Первым делом нужно подмонтировать флешку:

```
sudo mkdir /mnt/usb  
sudo mount /dev/sdb1 /mnt/usb
```

Затем загляните в файл `/root/.ecryptfs/sig-cache.txt`— там хранится кэш подписи, он выглядит примерно так: `da51c78bc1fc726d`. Запишите это значение.

Откройте файл `/root/.ecryptfsrc` и добавьте в него следующие строки:

```
key=passphrase:passphrase_passwd_file=/mnt/usb/secret.txt  
ecryptfs_sig=da51c78bc1fc726d  
ecryptfs_cipher=aes
```

```
ecryptfs_key_bytes=16  
ecryptfs_passthrough=n  
ecryptfs_enable_filename_crypto=n
```

Параметр `key` задает имя файла с паролем, второй параметр — подпись из файла `sig-cache.txt`. Остальные параметры задают тип шифрования, размер ключа и устанавливают прочие режимы `ecryptfs`.

Создайте файл `/mnt/usb/secret.txt` и добавьте в него строку:

```
passphrase_passwd=<секретная фраза>
```

Осталось совсем немного — обеспечить автоматическое монтирование флешки и зашифрованной файловой системы. Откройте `/etc/fstab` и добавьте в него строки:

```
/dev/sdb1          /mnt/usb  vfat          ro            0 0  
/home/den          /home/den  encryptfs      defaults      0 0
```

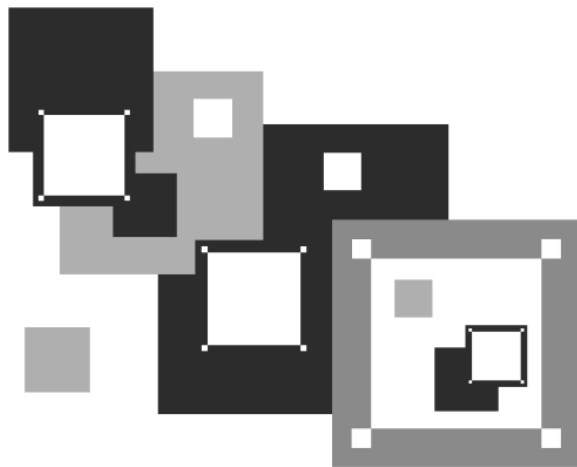
Первая строка монтирует флешку к каталогу `/mnt/usb`, а вторая — зашифрованную файловую систему. Понятно, что флешка должна быть смонтирована до монтирования зашифрованной файловой системы.

Перезагружаемся (`reboot`). В идеале все должно работать нормально — после перезагрузки зашифрованная файловая система подмонтируется автоматически.

Но в моем Debian все пошло не так — флешка не была автоматически подмонтирована, в результате не смонтировалась и `eCryptfs`. «Вылечить» проблему удалось редактированием файла `/etc/rc.local`, в который перед строкой `exit 0` я добавил строку `/bin/mount -a`:

```
/bin/mount -a  
exit 0
```

Теперь вы можете комфортно использовать `eCryptfs`.

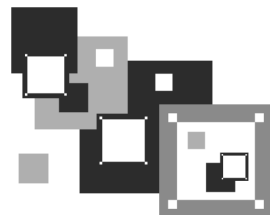


ЧАСТЬ VII

Виртуальные серверы

В настоящее время все чаще серверы становятся виртуальными. Дело в том, что «железо» настолько ушло вперед, что опережает во много раз запросы среднестатистического пользователя. Это и служит причиной развития рынка виртуальных серверов: провайдер покупает один мощный физический сервер, на котором с использованием средств виртуализации создаются несколько виртуальных серверов, предоставляемых в аренду пользователям. Количество виртуальных серверов зависит от их конфигурации и, конечно же, от конфигурации «железного» сервера. Виртуальные серверы стали настолько привычным явлением, что вся седьмая часть этой книги посвящена им.

ГЛАВА 43



А нужен ли физический сервер?

Наверное, каждый крупный проект или просто каждая крупная организация рано или поздно сталкивается с выбором: покупать физический или арендовать виртуальный сервер? Давайте попробуем разобраться, что лучше и экономически более оправданно. Заодно мы протестируем некоторых провайдеров, предоставляющих в аренду виртуальные серверы.

43.1. Физический или виртуальный?

43.1.1. Стоимость физического сервера

Прежде всего нужно присмотреть физический сервер, который будет соответствовать вашим ожиданиям, — чтобы перед глазами была какая-то сумма, и стало понятно, какой вариант экономически более целесообразен именно в вашем случае.

Прежде всего следует определиться, что мы считаем *сервером*. Если просто компьютер в обычном тауэр-корпусе, который будет пылиться в дальнем углу офиса, — это одно. По сути, и на любой ноутбук можно установить MS SQL Server и сделать его сервером баз данных. Вот только как быстро такая база «упадет» при реальной нагрузке даже в 5-10 пользователей при одновременной работе, например, в 1С?

Если вы себе представляете сервер именно так: отдельный компьютер, скажем, с 16-ю гигабайтами оперативки и одним терабайтником, тогда дальше эту главу можно не читать и не тратить свое время. Отправляйтесь лучше в любой онлайн-магазин и покупайте эту рабочую станцию — сервером такой продукт назвать нельзя.

В моем же представлении сервер — это машина с серверным процессором Xeon, регистровой памятью с ECC (англ. Error-Correcting Code Memory, память с коррекцией ошибок) и аппаратным дисковым массивом.

Корпус при размещении внутри офиса и при отсутствии серверных стоек значения не имеет, но я бы в перспективе присматривал корпуса в формате 1U/2U — рано или поздно вы поймете, что сервер лучше хранить в дата-центре.

На рис. 43.1 представлен сервер HP ProLiant DL180 Gen9:

- восьмиядерный процессор Intel Xeon E5-2620 v4 (2,1- 3 ГГц);
- регистровая (Registered) память с ECC, 16 Гбайт;
- RAID-контроллер Smart Array P440/2G 12Gb;
- форм-фактор корпуса 2U.

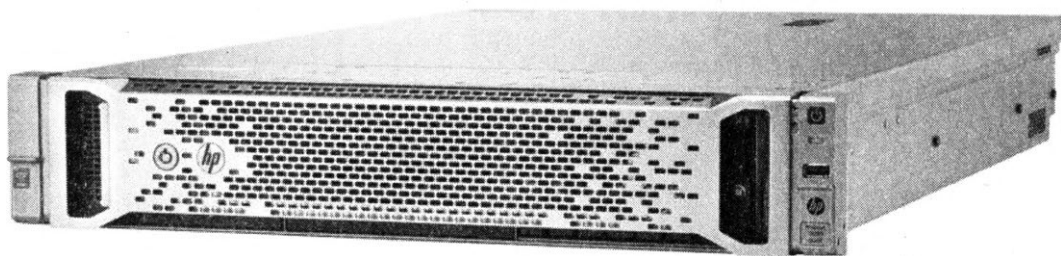


Рис. 43.1. Сервер HP ProLiant DL180 Gen9

Вот такой «комп» имеет право называться сервером. Но подобный аппарат, в зависимости от конфигурации, обойдется по данным Яндекс.Маркета в сумму порядка 162 250 рублей (примерно \$2800). Есть и более дешевые варианты этого сервера, но там или нет жесткого диска вообще, и/или объем памяти составляет 8 Гбайт, а не 16. В рассмотренную конфигурацию входит один модуль RDIMM памяти DDR объемом 16 Гбайт и два жестких диска по 300 Гбайт SAS. Это хороший вариант — как по цене, так и по конфигурации. Такой себе «среднячок» — есть варианты дороже, есть и дешевле. Можно купить китайский Patriot — он даже с лицензией на MS Server 2012 будет стоить дешевле. Но лучше на такие варианты не смотреть, особенно если вам важна стабильность, а не сам факт покупки сервера.

43.1.2. Необходимость в аппаратном сервере

Теперь, когда мы определились со стоимостью «железа», давайте подумаем, а нужен ли нам «физический» сервер вообще? Очень часто человек мучается над выбором сервера, а на самом деле он ему вообще не нужен. Аргументы «у соседа есть», «мне посоветовали» лучше сразу отбросить. Вот в каких случаях нужен собственный аппаратный сервер:

- *сайт с очень высокой посещаемостью* — когда обычный хостинг уже не выдерживает, и сайт часто отключают за превышение лимитов процессорного времени и/или трафика;
- *крупный интернет-проект* — серьезный интернет-магазин, социальная сеть, игровой сервер и т. д.;
- *портал с объемным контентом* — фото-банк вроде depositphotos.com, сайт с большим количеством музыки/видео;
- *необходимость совместной/удаленной работы* с каким-то ресурсоемким приложением — например, с 1С.

Да, во всех этих случаях сервер нужен (заметьте — я пока не говорю, какой). В остальных случаях, например, когда у вас относительно небольшой сайт и всего лишь один бухгалтер, на компьютер которого можно установить 1С, сервер не нужен вовсе. Можно арендовать обычный хостинг для размещения сайта и платить за это сущие копейки — что-то около 240 рублей в месяц за 6 Гбайт дискового пространства. Этого хватит для размещения даже нескольких сайтов. И если вы предполагали под такие нужды приобрести собственный сервер, то лишь стоимость «железа» окупится за более, чем 676 месяцев, или за 56 лет аренды. А при таком долгосрочном размещении вам еще и сделают существенную скидку ©.

Если же ваш проект подходит под приведенные ранее критерии, тогда сервер нужен. Осталось решить только, какой — ведь в большинстве случаев можно обойтись или виртуальным выделенным сервером (VDS), или виртуальным частным сервером (VPS), что значительно дешевле.

43.1.3. Про VPS, VDS и спекулянтов

И вот вы решились арендовать виртуальный сервер, но как его правильно выбрать? и что значат все эти непонятные аббревиатуры: VPS, VDS... Давайте рассмотрим, чем отличается VPS от VDS, и разберемся с некоторыми спекуляциями на разнице в одной букве.

VPS — это Virtual Private Server, т. е. *виртуальный частный сервер*, а VDS — Virtual Dedicated Server, т. е. *виртуальный выделенный сервер*. Бытует мнение, что это одно и то же. Отчасти, так оно и есть, это как бы различные варианты названия одной и той же сущности: как автомобиль и машина. Слова разные, но имеется в виду одно устройство.

Однако на практике частенько под VPS подразумевают виртуальные серверы, основанные на технологии OpenVZ (см. главу 44) или на Virtuozzo (см. главу 45), представляющей собой развитие технологии OpenVZ. Совсем другое дело VDS — это уже аппаратная виртуализация и реализуется она средствами гипервизоров (VMware, KVM, Hyper-V и некоторых других). В настоящее время фактически промышленным стандартом является технология KVM (Kernel-based Virtual Machine). Она же, в отличие от прочих, и единственная бесплатная. Следовательно, стоимость виртуальных серверов, построенных по технологии KVM, будет ниже по сравнению с коммерческими решениями на VMware или Hyper-V.

Далее мы разберемся, в чем разница между OpenVZ и KVM, а пока вам просто нужно знать, что KVM лучше.

Некоторые не очень честные компании спекулируют на том, что якобы между VPS и VDS нет разницы, и продают OpenVZ-серверы, как VDS. А некоторые вообще просто пишут в своих прайс-листах «виртуальный сервер». Именно поэтому важно перед покупкой сервера уточнить технологию виртуализации. Как правило, никто, специально обманывать вас не станет, просто иногда, особенно при использовании более дешевой OpenVZ-виртуализации, об этом умалчивают — до того момента, пока вы напрямую не спросите об этом.

Второй момент, который стоит уточнить перед покупкой сервера — тип накопителя, который используется на физическом сервере. Здесь возможны следующие варианты: SSD, SAS, SATA, а также некоторые гибридные решения вроде многоуровневого хранения (tiered storage) либо простого SSD-кэширования при помощи одного из стандартных средств Linux: dmcache, bcache, flashcache. Но обычно, чтобы не запутывать пользователя, предлагают три только что упомянутых варианта:

- самый дешевый из них — SATA. Такой жесткий диск будет мало чем отличаться от того, что установлен в вашем компьютере. Скорость чтения/записи у него порядка 100 Мбайт/с или, может, немногим больше — до 150 Мбайт/с;
- преимущество дисков SAS вовсе не в скорости, а в надежности. Конечно, SAS-диски за счет более высоких оборотов немного быстрее дисков SATA, но особой прыти и от них ожидать не приходится;
- SSD — единственный приемлемый для современного сервера вариант, но и стоит он дороже.

А в чем же состоит разница между OpenVZ и KVM? При использовании технологии OpenVZ (ранее уже отмечалось, что продукты, основанные на этой технологии, часто продаются как VPS) все виртуальные машины создаются на базе одного ядра операционной системы, каждая машина представляет собой сервер с программным окружением, однако без права изменения ядра и самой операционной системы.

Преимущества этой технологии — ее дешевизна. Виртуальные серверы, основанные на технологии OpenVZ, стоят дешевле. Также эта технология очень выгодна для самих компаний, предоставляющих в аренду виртуальные серверы, поскольку позволяет «оверселлить» ресурсы — т. е. продавать одни и те же ресурсы несколько раз разным пользователям.

Рассмотрим недостатки OpenVZ:

- *оверселлинг* — ресурсы оперативки и ядра выделяются без привязки к конкретной машине. Например, вы и ваш сосед арендуете у одного провайдера VPS-серверы одинаковой конфигурации — скажем, по 4 Гбайт оперативной памяти. Вот только вы используете самостоятельно написанный движок интернет-магазина, потребление памяти которого минимально (на уровне пусть 500 Мбайт на весь сервер), а ваш сосед установил монструозную Magento, которой своих 4 Гбайт оказалось мало, и она узурпировала еще и ваши оставшиеся 3,5 Гбайт. Даже если такого не произошло, то неиспользуемую оперативку провайдер может продать еще раз — выделить под другие серверы. Получается, что вы будете платить за ресурсы, которые вами не используются;
- *зависимость от соседей* — из предыдущего пункта следует еще одна проблема. Избыточная нагрузка на одну машину может привести к проблемам в работе соседних VPS. Например, соседский виртуальный сервер нагрузил процессор, а ваш сайт будет из-за этого тормозить. А нагрузить процессор очень легко — достаточно установить какую-то прожорливую CMS вроде той же Magento, и постоянный перерасход процессорного времени и оперативки вам гарантирован;

- *ограниченность настройки* — часть настроек OpenVZ-сервера изменить невозможно. Например, у вас не будет возможности управлять сетевыми интерфейсами, следовательно, тот же VPN-сервер развернуть уже не получится. Что-либо «сотворить» с ядром операционной системы тоже не удастся. Хорошо хоть, что это нужно далеко не всем.

Справедливости ради нужно отметить, что часть проблем OpenVZ может быть решена с помощью тонкой настройки ограничений (с использованием `cgroups`), но как именно выполнена настройка ограничений у того или иного хостера, вы не узнаете.

Совсем другое дело — технология KVM. Как уже было отмечено, это уже аппаратная виртуализация, реализуемая средствами гипервизора. Преимуществ — масса:

- *полноценный root-доступ* — по сути, KVM-сервер мало чем отличается от физического сервера, — пожалуй, только тем, что физически не существует. Вам подвластны ядро, правила маршрутизации, порты, фильтры и т. д.;
- *нет оверселлинга* — вы честно получаете те ресурсы, за которые платите. Если вы заказали сервер с 4 Гбайт оперативной памяти, то вашими гигабайтами никто из соседей воспользоваться не сможет, — можете в этом быть уверены. Даже если вы задействуете всего лишь 1 Гбайт, остальные три просто останутся свободными, точнее, система сможет использовать их под кэш файловой системы, что увеличит скорость работы виртуального сервера;
- *надежность и стабильность*, сравнимая с физическим оборудованием и даже выше, — ведь вы в любой момент можете одним нажатием кнопки сделать клон виртуальной машины и использовать его для моментального восстановления своего сервера.

Недостаток здесь только один — стоимость. KVM-серверы стоят несколько дороже серверов, созданных на базе технологии OpenVZ. Но в последнее время цены на виртуальные серверы взяли курс на снижение, и сейчас виртуальный KVM-сервер ненамного дороже хорошего хостинга.

43.1.4. Стоимость VDS

И вот настал момент истины. Давайте посчитаем, что выгоднее: физический сервер или VDS. Напомню, физический сервер (только «железо», без стоимости операционной системы и размещения в дата-центре) стоит, как мы ранее выяснили, порядка 162 тыс. рублей, в комплект входят два жестких диска по 300 Гбайт, из которых рабочим будет только один, а второй станет использоваться в качестве «зеркала». Собственно, от размера дискового пространства мы и будем отталкиваться — чем больше места на диске, тем дороже VDS.

Сколько настоящее время необходимо пространства для вашего проекта? Именно сейчас, а не через год или два, — в отличие от физического сервера, где вы покупаете 300 Гбайт сразу, в случае с VDS вы сможете докупать ресурсы постепенно — по мере того, как в них будет возникать надобность.

Давайте ориентироваться на немецкую компанию Hetzner. Выделенный сервер версии EX40-SSD обойдется в 58 евро (примерно 4060 рублей) в месяц. Конфигурация этого сервера следующая:

- процессор Intel Core i7-4770;
- 32 Гбайт оперативной памяти DDR3;
- два SSD-диска по 240 Гбайт (SATA 6 Гбит/с);
- один Gb/s-порт с гарантированной пропускной способностью 1 Гбит/с и 30 Гбайт трафика;
- полный root-доступ;
- 100 Гбайт дополнительного пространства для резервных копий.

При этом тех 162 тысяч хватит примерно на 40 месяцев аренды. Можно найти варианты и существенно дешевле. Например, предлагаются также серверы с 2-4 Гбайт оперативной памяти и 32 Гбайт дискового пространства (совсем слабая конфигурация, но вдруг именно она лучше всего соответствует вашим запросам) — стоимость аренды такого сервера составит примерно 24 тыс. рублей в год (!). Другими словами, стоимости физического сервера хватит на оплату 6,5 лет аренды этого виртуального сервера (VDS). И прошу заметить: вам не придется вносить все средства сразу.

На мой взгляд, физическое оборудование целесообразно покупать, если стоимость годовой аренды VDS нужной конфигурации превышает или примерно равна стоимости физического оборудования. Так, аренда VDS с 256 Гбайт дискового пространства и 8 Гбайт оперативки обойдется в год примерно в 126 867,6 рубля (средняя цена на российском рынке), а аренда такого же VDS, но с 16 Гбайт оперативки, в год будет стоить 143 786,8 рубля. Эти цифры уже вплотную приближаются к стоимости физического сервера, и в таком случае есть смысл задуматься о покупке физического оборудования. Действительно, если сравнивать с самой дорогой конфигурацией VDS (16 Гбайт RAM, 256 Гбайт HDD), то физический сервер окупит себя уже через 13 месяцев. Конечно, реальный срок окупаемости окажется чуть дольше, но об этом мы поговорим позже.

Для сроков окупаемости 24-36 месяцев нужно учитывать еще и гарантийный срок. На упомянутый ранее физический сервер HP он составляет 36 месяцев. А это означает, что спустя три года этот сервер, возможно, потребует дополнительных «вливаний» — может выйти из строя тот же жесткий диск или блок питания.

43.1.5. Физический сервер vs VDS

У каждого из решений — если забыть о стоимости — есть свои преимущества. К преимуществам физического сервера можно отнести производительность и большее дисковое пространство.

Что бы ни говорили, а производительность виртуального сервера хоть и будет высокой, но окажется ниже, чем у физического сервера. Это факт. К тому же, даже если сравнивать наш физический сервер с самой дорогой конфигурацией VDS, то

у вас будет 44 (300 минус 256) Гбайт дополнительного дискового пространства — что весьма ощутимо.

Преимуществом физического сервера является и то, что он «физический», и его можно пощупать. Попробуйте в бухгалтерии объяснить, за что им придется платить 143 786,8 рубля в год? А так им можно предъявить «ящик», именуемый сервером.

Зато к преимуществам VDS можно смело отнести простоту обслуживания. Вам не придется беспокоиться, что жесткий диск выйдет из строя, — если это и случится, то это проблемы провайдера, а вы сможете восстановить свою виртуальную машину из бэкапа. О клонировании сервера нажатием одной кнопки я уже молчу — виртуальная машина на то и виртуальная. Не нужно ни о чем заботиться: ни о перепадах напряжения, ни об отключении электричества, и т. д. Все это — проблемы провайдера, которые он будет решать сам.

Модернизировать VDS тоже очень просто — заказали в панели администратора дополнительные ресурсы, и сразу после оплаты они стали доступны. Так, за 1 Мбайт оперативки придется доплатить 0,18 руб./мес., а за 1 Мбайт места на диске — 0,02 руб./мес. Не нужно останавливать сервер, переносить данные на другой жесткий диск или просто конфигурировать другой жесткий диск. Все очень просто.

VDS вы получаете сразу после оплаты — купили и сразу можете использовать. С физическим сервером все сложнее — после оплаты его нужно доставить, на что может уйти несколько дней. Затем его надо будет настроить: установить операционную систему, отладить сервисы... VDS же сразу готов к использованию — залили свой контент, настроили DNS, и ваш сайт уже работает.

К тому же в стоимость VDS уже входит один IP-адрес, гарантированный интернет-канал, а также бесперебойное питание. Если первые две «плюшки» (IP-адрес и Интернет) сейчас особо не проблема, то о бесперебойном питании нужно поговорить отдельно.

43.1.6. Стоимость владения физическим сервером

Сервер мало купить. Нужно еще платить за его существование. Как минимум, надо обеспечить:

- резервный интернет-канал;
- резервное питание (ИБП стоят дорого, а возможно, придется устанавливать еще и дизель-генераторы);
- систему кондиционирования для поддержания температуры, оптимальной для работы сервера.

С системой кондиционирования все просто, с резервным интернет-каналом все не так просто, но решаемо. А вот обеспечить резервное питание сложнее, и если при его отсутствии отключат электричество, вы останетесь без сервера, а репутации вашего ресурса будет нанесен огромный удар.

По тем или иным причинам организовать резервную линию получается не всегда, а ИБП достаточной для работы сервера мощности будет стоить дороже самого сер-

вера, — посмотрите любопытства ради на ИБП APC серии Symmetra MW. Сразу говорю: покупать вам их не захочется. Выход один — дизель-генераторы. И если у вас собственное частное здание, то такой вариант возможен. Но если вы арендуете офис в бизнес-центре или квартиру, то вряд ли соседи будут рады генератору. Да и по пожарным нормам установка такого генератора там запрещена.

В ситуации, когда вам нужен именно мощный физический сервер, арендовать VDS вы не хотите, а организовать резервное питание невозможно, есть вариант воспользоваться *услугой размещения сервера* (collocation). При этом сервер физически помещается в дата-центре провайдера, где обеспечивается резервирование интернет-канала и питания, а также поддерживается оптимальная температура.

Стоят услуги по размещению сервера относительно недорого. Само размещение сервера обойдется примерно в 2500 руб./мес. (это усредненная стоимость). При выборе услуги размещения нужно обращать внимание на предоставляемую пропускную способность — от этого зависит стоимость размещения. Часто полоса ограничивается на уровне 10 Мбит/с, а если требуется большая полоса, то придется доплачивать.

Таким образом, существование сервера обойдется как минимум в 3000 р. в месяц. Этот момент нужно учитывать при подсчете рентабельности покупки физического оборудования. Ведь за эти деньги (и даже дешевле) можно арендовать VDS с 2 Гбайт ОЗУ и 32 Гбайт дискового пространства!

Все это я к тому, что из стоимости аренды VDS можно смело вычитать цену размещения сервера — ведь в случае с физическим сервером все равно бы пришлось платить эти деньги.

43.1.7. Выводы

Арендовать VDS в большинстве случаев не только проще, но и выгоднее. Использование же физических серверов целесообразно только, если планируемая нагрузка столь высока, что с ней не справится виртуальный сервер (если планируется, что будут задействованы все 8 ядер физического процессора), и сразу необходимо все дисковое пространство. Во всех остальных случаях выгоднее арендовать VDS. Именно поэтому вторая часть этой главы посвящена выбору виртуального сервера.

43.2. Виртуальный тест-драйв

На отечественном рынке присутствует множество провайдеров, предоставляющих услуги по аренде виртуальных (VPS/VDS) серверов. Всех возможных провайдеров протестировать, ясное дело, не получится, поэтому ограничимся следующими:

- Джино (<https://jino.ru/>);
- Спринтхост (<http://sprintbox.ru>);
- Макхост (<https://mchost.ru>);
- UltraVDS (<https://ultravds.com/>);
- 1cloud.ru.

Это не реклама провайдеров, поэтому тестирование будет объективным. Во время тестирования мы оценим пропускную способность, скорость работы дисковой подсистемы, по возможности выполним нагрузочное тестирование, а также разберемся с ценами. Здесь надо уточнить, что эта глава готовилась в июле 2017 года, и когда книга попадет вам в руки, цена услуг может измениться. Однако приведенные в главе цены нам понадобятся, чтобы сопоставимо сравнить стоимость услуг различных провайдеров и выявить, какие провайдеры дороже, а какие — дешевле.

43.2.1. Джино

О ценах

Компания «Джино» честно предлагает VPS-сервер и не скрывает этого. Стоимость серверов очень демократична — конфигурация **Альфа** предлагается весьма дешево — всего за 99 рублей в месяц (рис. 43.2). При этом пользователь получит гибридный диск (SSD+HDD) объемом 5 Гбайт, 512 Мбайт оперативки и всего одно ядро на 500 МГц. Конфигурация скудненькая, и по-хорошему ее можно использовать только в качестве тест-драйва. Для более реальных задач подойдет конфигурация **Бета**, которая стоит уже 500 рублей в месяц. За эти деньги вам предлагается 2 ядра по 2000 МГц, 2 Гбайт оперативки и 20 Гбайт SSD+HDD. Вот это уже то, что нужно.

Тарифы			
Альфа Достаточно, чтобы попробовать	Бета Подойдет для большинства задач	Бета Плюс + Только для участников программы «Джино Плюс +»	Гамма Для самых требовательных
CPU 1 ядро × 500 МГц	CPU 2 ядра × 2000 МГц	CPU 2 ядра × 2000 МГц	CPU 4 ядра × 2000 МГц
RAM 512 МБ	RAM 2048 МБ	RAM 3072 МБ	RAM 4096 МБ
Диск 5 ГБ SSD+HDD	Диск 20 ГБ SSD+HDD	Диск 30 ГБ SSD+HDD	Диск 60 ГБ SSD+HDD
Выделенные IP —	Выделенные IP 89 ₽/мес.	Выделенные IP 89 ₽/мес.	Выделенные IP 89 ₽/мес.
Контрольная панель бесплатно	Контрольная панель бесплатно	Контрольная панель бесплатно	Контрольная панель бесплатно
99 ₽ в месяц	499 ₽ в месяц	699 ₽ в месяц	1499 ₽ в месяц
Создать сервер			

Рис. 43.2. Джино: стоимость VPS

Если сравнить эту цену с предложением германского провайдера **hetzner.com**, то конфигурация CX10 обойдется в 4,60 евро в месяц, а это примерно 325 рублей. Но там есть всего одно виртуальное ядро, 1 Гбайт оперативки и 25 Гбайт SSD (на 5 Гбайт больше). Выходит даже немного дешевле. Но нужно учитывать, что у **hetzner.com** трафик платный — бесплатные только первые 2 Тбайт, да и конфи-

гурация немного слабее. Если же нужна конфигурация с двумя ядрами и двумя гигабайтами ОЗУ, то она обойдется уже 8,14 евро, или 569,8 рубля (в пакет входит 5 Тбайт трафика, что на сегодняшний день вполне достаточно).

Также обратите внимание на выделенный IP. У Джино он приобретается отдельно за 89 руб./мес., а у **hetzner.com** уже входит в тарифный план. Получается, что по факту конфигурация Бета стоит немного дороже: 589 рублей против 569,8 рублей у **hetzner.com**. VPS от Джино можно порекомендовать тем, кто планирует общий расход трафика более 5 Тбайт, — тогда предложение от Джино окажется выгоднее.

Создание сервера

Создать сервер у Джино просто, а сам процесс занимает считанные минуты, — нужно выбрать конфигурацию и нажать кнопку Создать сервер. Для теста я создал сервер конфигурации Бета, работающий под управлением CentOS 7.

Панель управления сервером Джино показана на рис. 43.3. Здесь можно выключить, включить и перезагрузить сервер, просмотреть его характеристики, изменить тариф, просмотреть статистику использования ресурсов. Ссылка Открыть консоль в разделе Доступ к серверу позволяет открыть консоль root (рис. 43.4).

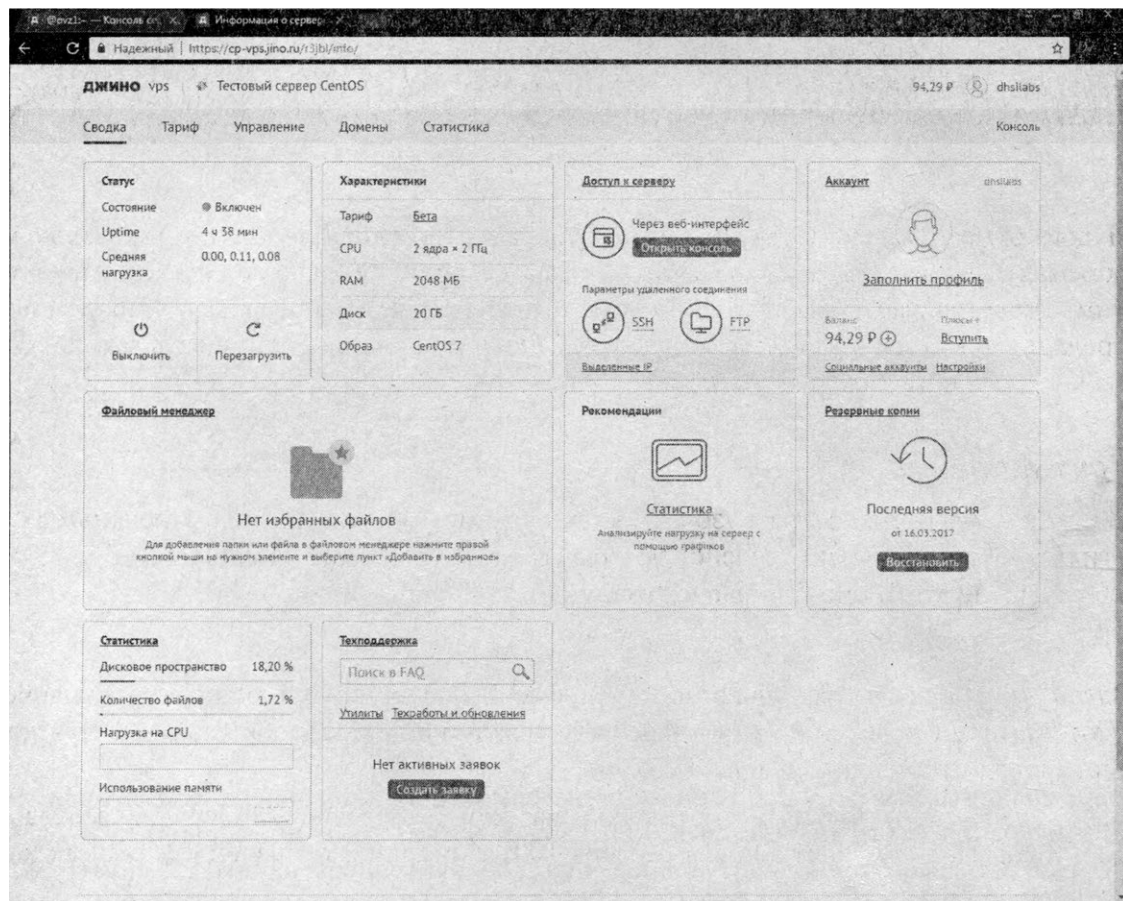


Рис. 43.3. Джино: панель управления

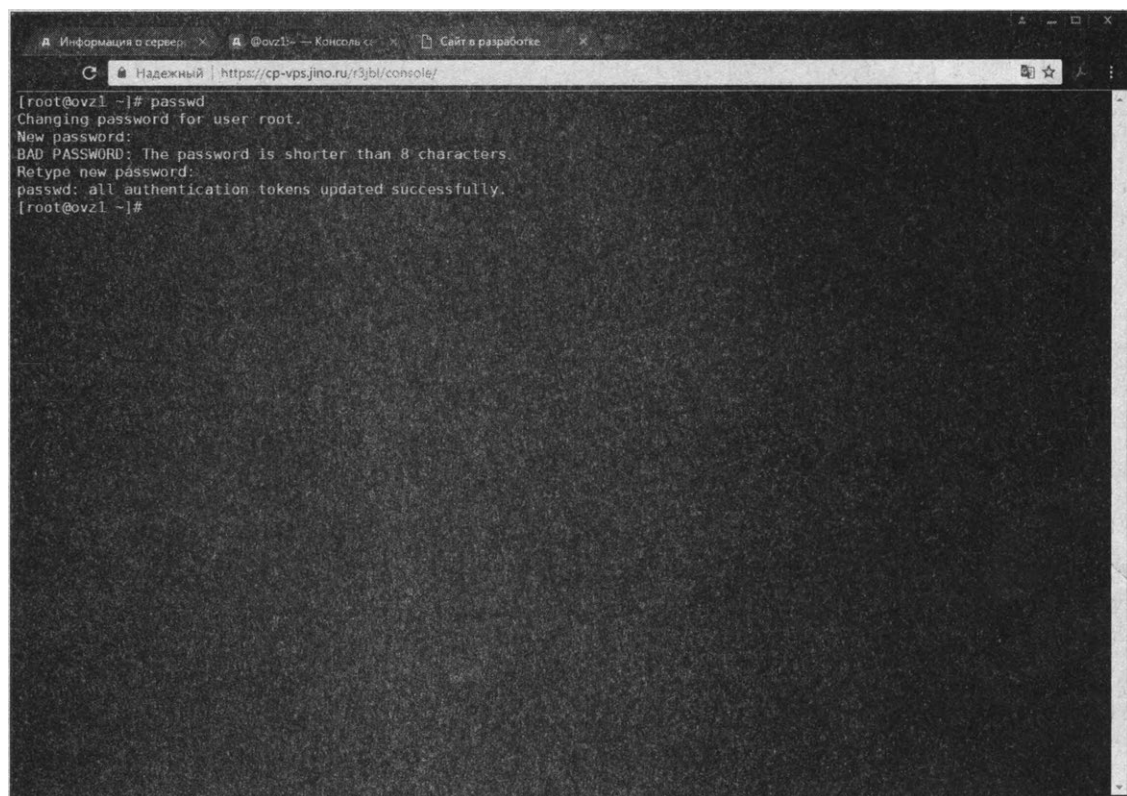


Рис. 43.4. Джино: консоль root

Нужно отметить, что Web-консоль работает в целом нормально, но при больших объемах вывода может подвисать, — тогда ее приходится перезапускать, но при этом теряется часть вывода. В целом это не проблема, поскольку Web-консоль предлагается только как экстренная мера. Обычно нужно использовать ssh, клиенты которого есть даже для Android.

Тестирование

Проведем простенькое нагрузочное тестирование. Можно было бы использовать утилиту `ab`, но это неинтересно, поэтому я установил `siege` командой `yum install siege`. После этого запустил `siege` командой:

```
siege -c 255 -r 10 -d 1 <имя моего VPS>
```

Здесь опция `-c` задает количество одновременных соединений, опция `-r` — количество повторений, а опция `-d` — задержку между попытками обращения. Результат тестирования приведен на рис. 43.5.

Было сделано 5100 хитов, потрачено на это безобразие 75 секунд, передано 567 Мбайт данных, доступность сервера 100%, ни одной проваленной транзакции. Усложним задачу:

```
siege -c 500 -r 10 -d 1 <имя моего VPS> > res.txt
```

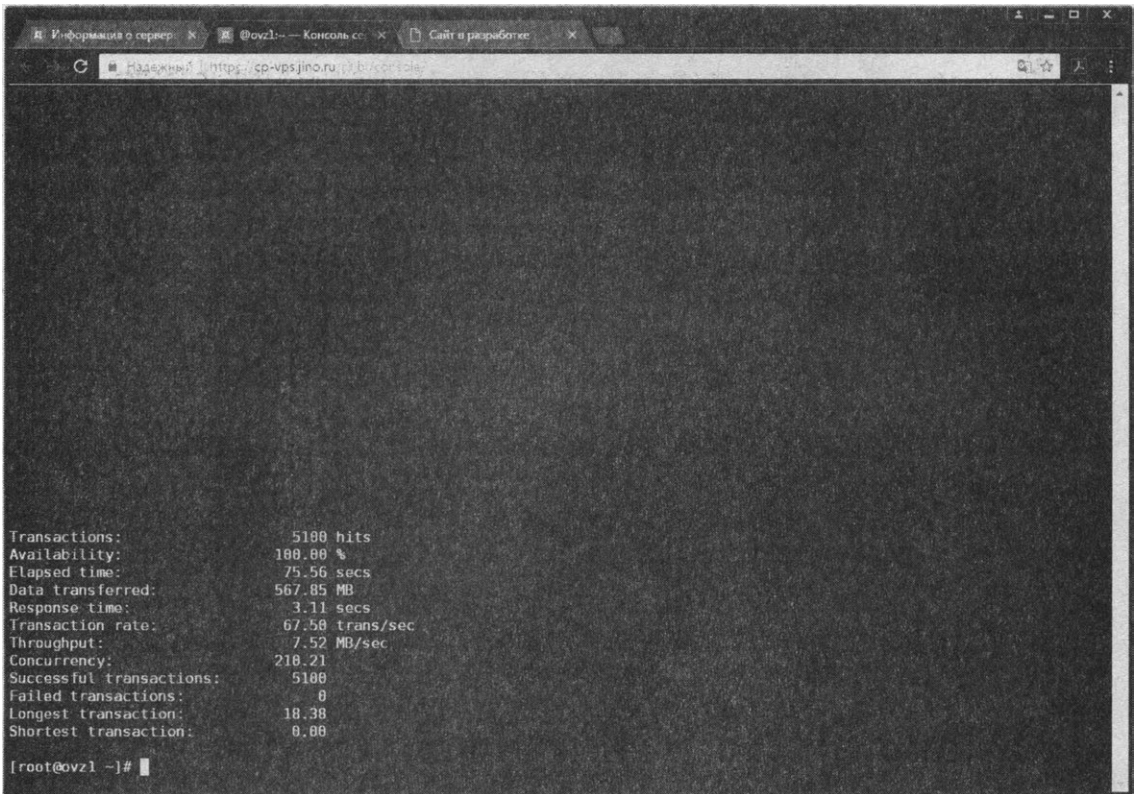


Рис. 43.5. Джинго: результат нагрузочного тестирования

Здесь я увеличиваю количество одновременных соединений до 500, задаю 10 повторений (`-r 10`) и результаты вывожу в файл `res.txt`, чтобы не захламлять вывод на консоль, — на консоли теперь будут выводиться только сообщения об ошибках. А таковые появились (рис. 43.6): при 500 одновременных пользователях доступность сервера составила уже 98,2%, а проваленных транзакций (`failed`) было уже 177.

На мой взгляд, результаты неплохие. Да, при 500 одновременных пользователях появилась ошибка тайм-аута соединения. Но если у вас такой популярный сайт, что к нему обращаются более 500 пользователей, то нужно выбирать и более дорогой тариф. Однако не следует забывать, что сервер тестировался без какой-либо CMS, т. е. `siege` обращался к Web-серверу, возвращающему тестовую страничку. Если была бы установлена CMS, то результаты были бы хуже, — все зависит от «прожорливости» CMS.

Теперь протестируем дисковую подсистему. Нам обещали «скорость SSD по цене HDD». Давайте проверим, насколько быстр гибридный накопитель. Сейчас я пишу эти строки на стареньком ноутбуке, в который установлен свежий SSD, показывающий в `Crystal Disk Mark` 384 Мбайт/с в режиме последовательной записи. В Linux `Crystal Disk Mark` нет, но зато есть старый добрый `dd`. Попробуем создать файл размером 2 Гбайт, а `dd` сообщит нам скорость этой операции:

```
dd if=/dev/zero of=temp bs=1M count=2048
```

```

[error] socket: -1914046720 connection timed out.: Connection timed out
[error] socket: 20702976 connection timed out.: Connection timed out
[error] socket: -1683269888 connection timed out.: Connection timed out
[error] socket: -1798658304 connection timed out.: Connection timed out
[error] socket: -1725229312 connection timed out.: Connection timed out
[error] socket: -293992704 connection timed out.: Connection timed out
[error] socket: -199584000 connection timed out.: Connection timed out
[error] socket: -1746209024 connection timed out.: Connection timed out
[error] socket: -661137664 connection timed out.: Connection timed out
[error] socket: -1767188736 connection timed out.: Connection timed out
[error] socket: -1605224704 connection timed out.: Connection timed out
[error] socket: -1567881472 connection timed out.: Connection timed out
[error] socket: -1573755136 connection timed out.: Connection timed out
[error] socket: 786462464 connection timed out.: Connection timed out
[error] socket: 1824958208 connection timed out.: Connection timed out
[error] socket: -1956006144 connection timed out.: Connection timed out
[error] socket: -1861597440 connection timed out.: Connection timed out
[error] socket: -493299968 connection timed out.: Connection timed out
[error] socket: -1851107584 connection timed out.: Connection timed out
[error] socket: -1557391616 connection timed out.: Connection timed out
[error] socket: -1599351040 connection timed out.: Connection timed out
[error] socket: 734013184 connection timed out.: Connection timed out
[error] socket: -812611840 connection timed out.: Connection timed out
[error] socket: -1668163840 connection timed out.: Connection timed out

Transactions:          9646 hits
Availability:          90.20 %
Elapsed time:          132.02 secs
Data transferred:      1074.02 MB
Response time:         6.01 secs
Transaction rate:      73.06 trans/sec
Throughput:            8.14 MB/sec
Concurrency:          438.95
Successful transactions: 9646
Failed transactions:    177
Longest transaction:    15.18
Shortest transaction:    0.00

[root@ovz1 ~]#

```

Рис. 43.6. Джино: 500 одновременных пользователей

Результаты меня порадовали. Но, правда, не с первого раза. Изначально тест показал 10,5 Мбайт/с. Об этом я написал в службу поддержки, и «баг» был исправлен, — скорость получилась на уровне 805 Мбайт/с (рис. 43.7).

ПРИМЕЧАНИЕ

Если бы я использовал не `/dev/zero`, а «забивал» диск случайными данными (`/dev/random`), то результаты, возможно, были бы хуже, но не намного.

Напоследок посмотрим на пропускную способность, для оценки которой я воспользуюсь консольной версией `speedtest.net`:

```
$ wget -O - https://raw.githubusercontent.com/sivel/speedtest-cli/master/speedtest.py | python
```

Результаты к ближайшему серверу (Ростелеком, Москва) показаны на рис. 43.8. Мне понравилось, что канал синхронный. Пусть и нет сотен Мбит/с, зато пропускная способность одинаковая в обе стороны. У некоторых провайдеров можно встретить ситуацию, когда **download** (входящий трафик) оказывается порядка 800 Мбит/с, а **upload** (исходящий трафик) еле дотягивает до 100 Мбит/с.

Выводы

Что мне понравилось:

- доступные цены на VPS. Нет платы за установку сервера;
- простота установки и управления VPS;

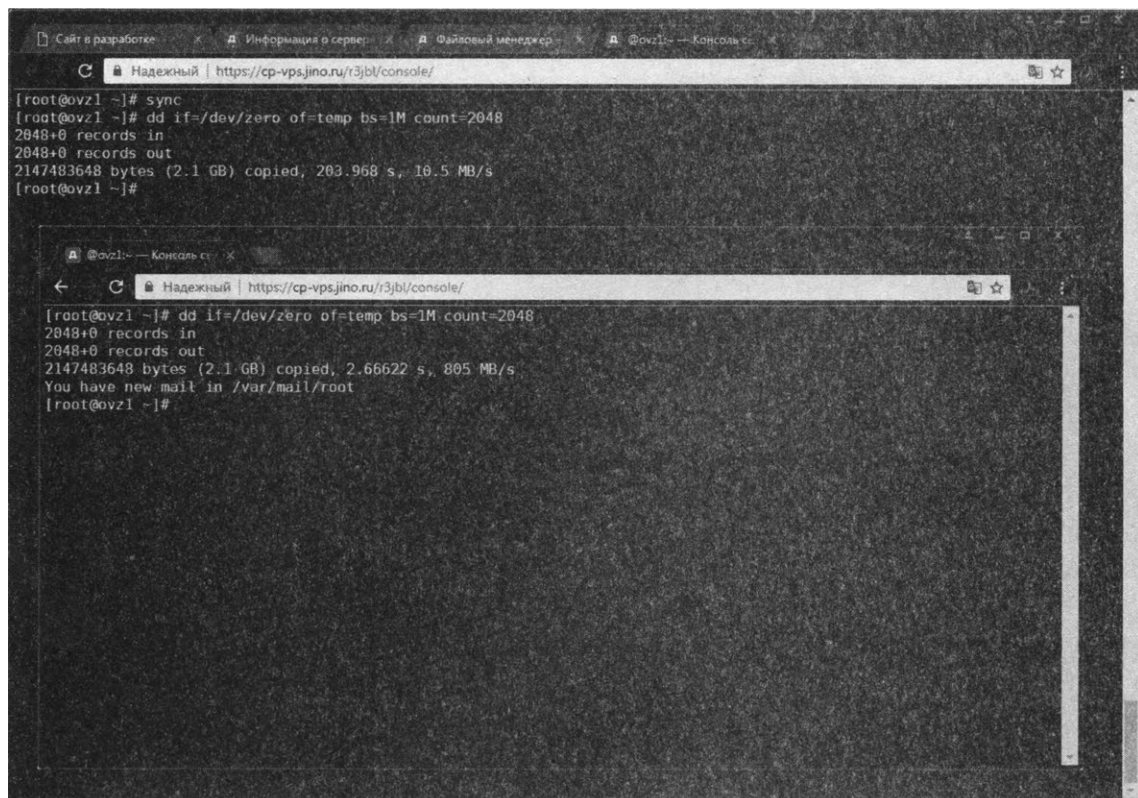


Рис. 43.7. Джино: замер производительности диска командой `dd` — до (вверху) и после (внизу) обращения в службу поддержки

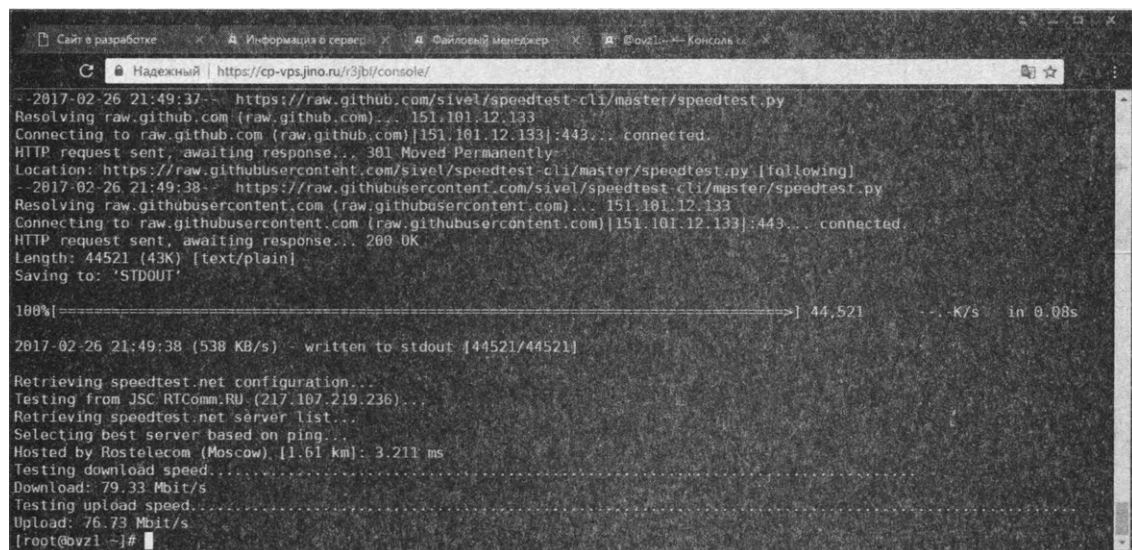


Рис. 43.8. Джино: результаты тестирования пропускной способности

- бесплатная панель собственной разработки;
- неограниченный трафик;
- посуточная тарификация;
- синхронный канал, правда, с пропускной способностью всего порядка 79 Мбит/с;
- сервер выдержал нагрузочное тестирование;
- консоль, работающая в браузере;
- возможность выбора ОС (CentOS, Debian, Ubuntu);
- высокая производительность дисковой подсистемы.

Что не понравилось:

- в тариф не включен выделенный IP-адрес, за который нужно доплачивать 89 руб./месяц;
- «баг» со скоростью работы SSD. Его хоть при мне и исправили, но все же рекомендую при покупке VPS произвести аналогичное тестирование скорости;
- пропускная способность в 79 Мбит/с — маловато.

43.2.2. Спринтхост

О ценах

Как и в предыдущем случае, начнем с ценовой политики, поскольку стоимость услуги — это первое, на что смотрят при выборе виртуального сервера. Конфигураций всего две, так что выбор невелик: или за 400 рублей в месяц, или за 999. При этом вы получите всего одно ядро, а разница только в накопителе и памяти: 32 Гбайт SSD и 2 Гбайт ОЗУ или 48 Гбайт SSD и 4 Гбайт ОЗУ. Если потребуется второе ядро, придется доплачивать 200 рублей.

На первый взгляд кажется, что предлагаемые варианты дороже, чем у Джино, но это не так. Цены можно считать дешевыми, учитывая следующие обстоятельства:

- это VDS, а не VPS, т. е. аппаратная виртуализация KVM;
- выделенный IP-адрес без всяких доплат;
- трафик не ограничен;
- пропускная полоса для входящего трафика — 100 Мбит/с, для исходящего — не ограничена;
- 32 Гбайт настоящего SSD, а не гибридного, что сулит высокую производительность (и это действительно так, что и будет показано далее).

Если сравнивать с Джино, то выйдет дороже всего на 11 рублей — ведь за IP-адрес у Джино нужно доплачивать. Выходит, VDS (а не VPS) окажется всего на 11 рублей в месяц дороже, чем VPS! Плюс к этому дискового пространства будет больше на 12 Гбайт. Так что предложение от Спринтхост может быть весьма заманчивым.

Создание сервера

Время создания сервера VDS исчисляется секундами. Пока я делал и сохранял скриншот, сервер уже был создан (рис. 43.9). В панели администратора сообщается конфигурация сервера, находятся кнопки управления сервером. Можно остановить, перезапустить сервер, создать бэкап и даже удалить сервер, если он больше не нужен.

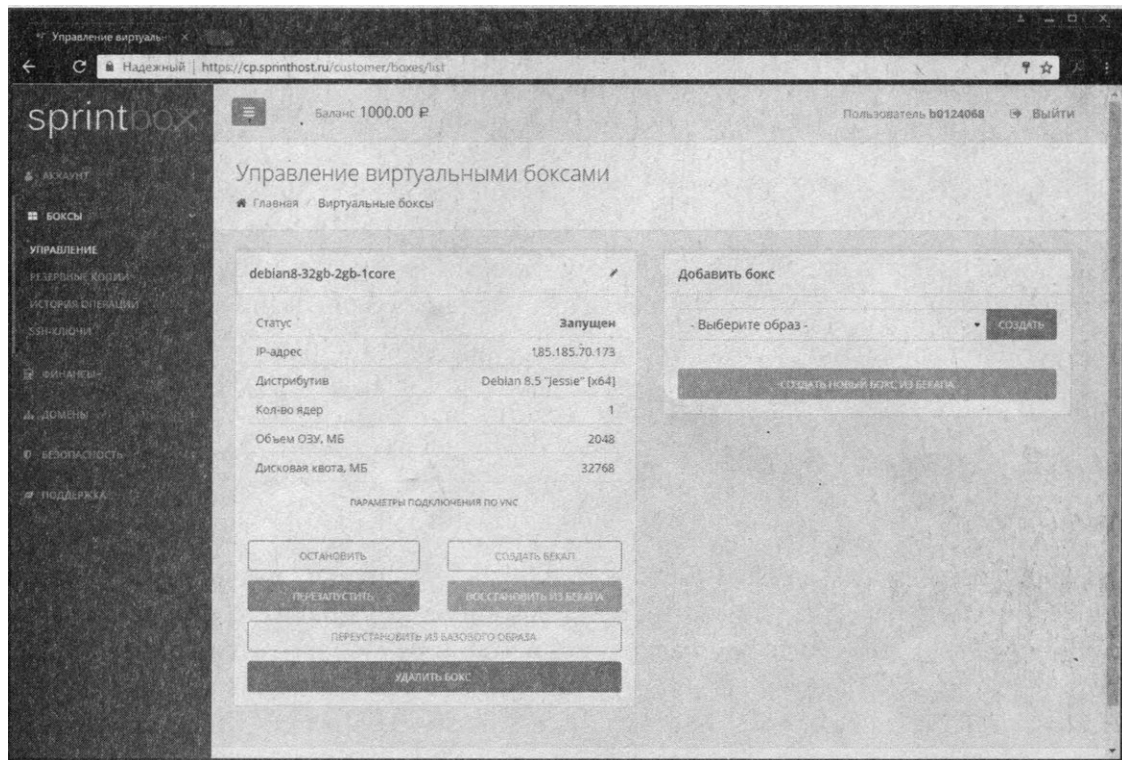


Рис. 43.9. Спринтхост: VDS готов к работе

Пароль root отправляется на адрес электронной почты, указанный при регистрации. Войти на свой сервер можно как по VNC, так и по ssh. Мне лично по душе больше второй вариант, поэтому вхожу по ssh. При первом входе VDS просит сразу поменять пароль root. Web-консоли не предусмотрено, но, как отмечалось ранее, с ssh сейчас никаких проблем нет, и наличия компьютера под рукой не обязательно, — чтобы управлять сервером, хватит современного Android-смартфона.

Поскольку сервер абсолютно «голый», пришлось доустановить некоторый софт, а именно apache2, php5, siege и еще кое-чего по мелочам.

ВНИМАНИЕ!

Услуга VDS предоставляется в концепции «без администрирования», другими словами, никакой красивой панели администратора с графиками нагрузки не будет, но ее никто не запрещает установить самостоятельно (можно использовать ту же бесплатную Vesta CP). Это не совсем чтобы недостаток, но об этом нужно знать.

Тестирование

С файлом `index.html` по умолчанию сервер справился без проблем, как с 500 (рис. 43.10), так даже и с 1000 одновременных соединений (рис. 43.11).

Рис. 43.10. Спринтхост: результаты нагрузочного тестирования для 255 и 500 одновременных пользователей

При 1000 соединений доступность составила 99,96%, поскольку оказалось четыре failed-транзакции. Некоторые другие виртуальные серверы не выдерживают и 500 одновременных обращений к страничке по умолчанию.

Как вы уже догадались, следующий этап теста посвящен тестированию диска. Команда будет той же самой, чтобы читатели могли сравнить полученные результаты с результатами других провайдеров:

```
dd if=/dev/zero of=temp bs=1M count=2048
```

Получилось 286 Мбайт/с (рис. 43.12). С одной стороны, это уровень SSD-диска. С другой — нам обещали, что скорость будет выше, чем у гибридных решений, — видимо, намекая на Джино, где при их гибридном решении я получил 805 Мбайт/с.

Наконец, протестируем пропускную способность (рис. 43.13)— speedtest показал практически синхронный канал: 101 Мбит/с **download**, 111 Мбит/с **upload** — хорошие показатели.


```

SprintHost.tlp - root@185.185.70.173:22 - Bitvise xterm - mc [root@box-3258]~
Elapsed time:          10.22 secs
Data transferred:      14.50 MB
Response time:         0.09 secs
Transaction rate:      489.24 trans/sec
Throughput:            1.42 MB/sec
Concurrency:           45.44
Successful transactions: 5000
Failed transactions:    0
Longest transaction:   1.16
Shortest transaction:  0.00

root@box-3258:~# siege -c 1000 -r 10 -d 1 http://185.185.70.173 > res.txt
** SIEGE 3.0.8
** Preparing 1000 concurrent users for battle.
The server is now under siege...[error] socket: read error Connection reset by peer sock.c:479: Conn
ection reset by peer
[alert] socket: -822724864 select timed out: Connection timed out
[alert] socket: -1617934592 select timed out: Connection timed out
[alert] socket: -1066113280 select timed out: Connection timed out
done.

Transactions:          9996 hits
Availability:          99.96 %
Elapsed time:          39.53 secs
Data transferred:      28.99 MB
Response time:         0.52 secs
Transaction rate:      252.87 trans/sec
Throughput:            0.73 MB/sec
Concurrency:           132.59
Successful transactions: 9996
Failed transactions:    4
Longest transaction:   21.74
Shortest transaction:  0.00

root@box-3258:~#

```

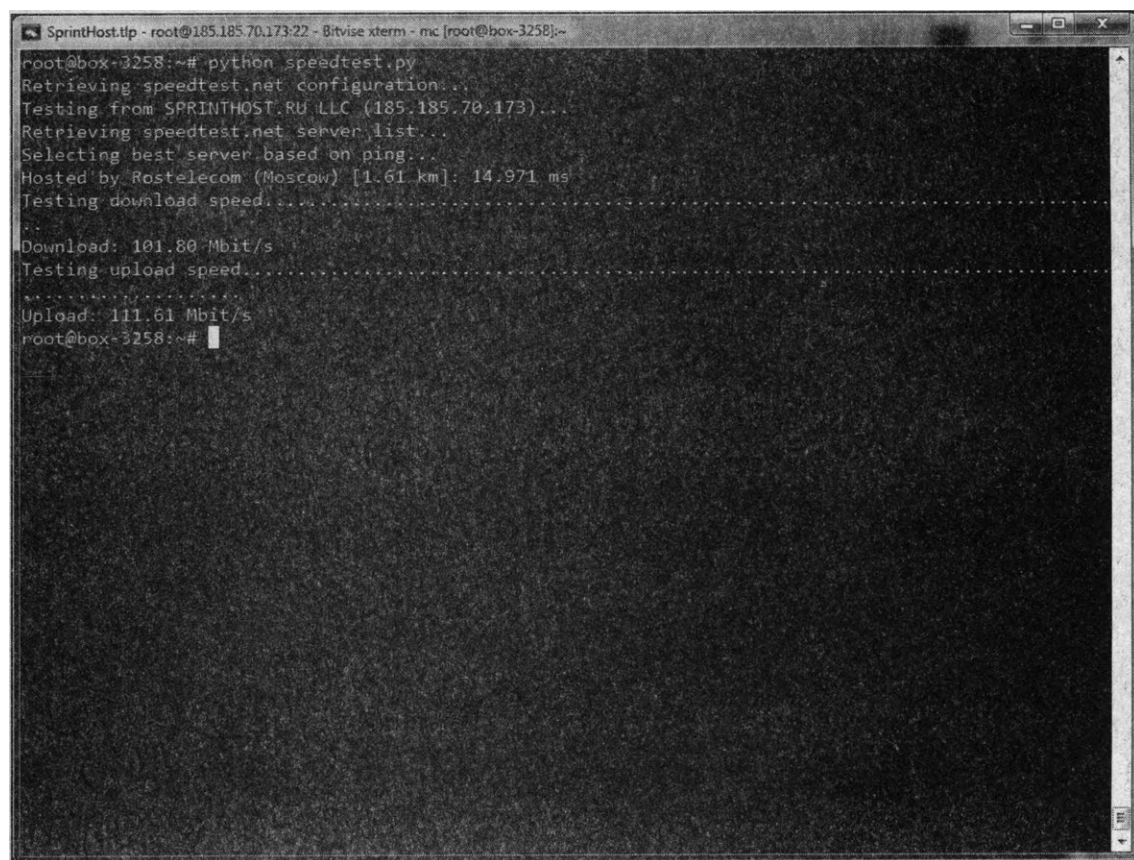
Рис. 43.11. Спринтхост: результаты нагрузочного тестирования — 1000 пользователей

```

SprintHost.tlp - root@185.185.70.173:22 - Bitvise xterm - mc [root@box-3258]~
root@box-3258:~# dd if=/dev/zero of=temp bs=1M count=2048
2048+0 records in
2048+0 records out
2147483648 bytes (2.1 GB) copied, 7.49583 s, 286 MB/s
root@box-3258:~#

```

Рис. 43.12. Спринтхост: скорость записи

A screenshot of a terminal window titled 'SprintHost.tlp - root@185.185.70.173:22 - Bitwise xterm - mc [root@box-3258]:~'. The terminal shows the execution of a speedtest script. The output includes: 'Retrieving speedtest.net configuration...', 'Testing from SPRINTHOST.RU LLC (185.185.70.173)...', 'Retrieving speedtest.net server list...', 'Selecting best server based on ping...', 'Hosted by Rostelecom (Moscow) [1.61 km]: 14.971 ms', 'Testing download speed...', 'Download: 101.80 Mbit/s', 'Testing upload speed...', 'Upload: 111.61 Mbit/s', and the prompt 'root@box-3258~#'.

```
SprintHost.tlp - root@185.185.70.173:22 - Bitwise xterm - mc [root@box-3258]:~
root@box-3258~# python speedtest.py
Retrieving speedtest.net configuration...
Testing from SPRINTHOST.RU LLC (185.185.70.173)...
Retrieving speedtest.net server list...
Selecting best server based on ping...
Hosted by Rostelecom (Moscow) [1.61 km]: 14.971 ms
Testing download speed.....
Download: 101.80 Mbit/s
Testing upload speed.....
Upload: 111.61 Mbit/s
root@box-3258~#
```

Рис. 43.13. Спринтхост: пропускная способность

Выводы

Преимущества Спринтхост:

- ☐ VDS;
- ☐ простота управления;
- ☐ хорошая производительность дисковой подсистемы;
- ☐ приемлемая пропускная способность;
- ☐ выдержал нагрузочное тестирование;
- ☐ доступная цена;
- ☐ наличие VNC-доступа к серверу.

А вот недостатков как таковых нет. Такой себе середнячок, где все хорошо: и скорость записи на диск, и пропускная способность, и цена, которая действительно невысока, учитывая, что это все-таки VDS.

43.2.3. Макхост

О ценах

Компания «Макхост» предлагает VPS-хостинг, тарифы на который приведены на рис. 43.14. Самая дешевая конфигурация обойдется в 879 рублей в месяц— вы получите 2 Гбайт памяти, 2 ядра процессора и 10 Гбайт на SSD-диске. В тариф входит неограниченный трафик и 1 выделенный IP-адрес. С такой конфигурацией вполне можно жить, вот только места на диске хотелось бы больше.

VZ-1	VZ-2	VZ-3	VZ-4
10 ГБ на SSD	35 ГБ на SSD	100 ГБ на SSD	200 ГБ на SSD
2 ядра CPU	2 ядра CPU	4 ядра CPU	8 ядер CPU
2 ГБ RAM DDR4	4 ГБ RAM DDR4	8 ГБ RAM DDR4	16 ГБ RAM DDR4
Неограниченный трафик	Неограниченный трафик	Неограниченный трафик	Неограниченный трафик
1 выделенный IP	1 выделенный IP	1 выделенный IP	1 выделенный IP
Год Месяц	Год Месяц	Год Месяц	Год Месяц
879 Р в месяц при оплате на год	1583 Р в месяц при оплате на год	2903 Р в месяц при оплате на год	5543 Р в месяц при оплате на год
Экономия 1439 Р	Экономия 2591 Р	Экономия 4751 Р	Экономия 9071 Р
Заказать	Заказать	Заказать	Заказать

Рис. 43.14. Макхост: тарифы

Дело в том, что на обычном shared-хостинге вы получаете чистое дисковое пространство, на котором сможете хранить файлы своего сайта, почту и базы данных. А покупая сервер на 10 Гбайт на VPS/VDS-хостинге, не забывайте, что 1-2 Гбайт будет занимать программное обеспечение: операционная система, установленные сервисы и т. д. Вот поэтому и хотелось бы больше места на диске, но следующая конфигурация обойдется уже в 1583 рубля в месяц. Дорого, и при этом вы получаете VPS, а не VDS.

Впрочем, на Макхосте часто проводят различные акции, есть партнерская программа (40% скидки с первого заказа и 20% — со всех последующих заказов приведенных клиентов) — все это позволяет экономить, но для экономии нужно что-либо еще делать. А у других провайдеров вы сразу получаете услуги дешевле.

Создание сервера

Пользователю предоставляются две панели управления. Первая — это общая панель управления аккаунтом (<https://cp.mchost.ru/>), где можно оплачивать услуги, просматривать статистику VPS, выполнять перезагрузку VPS, управлять доменами и т. д.

Вторая панель управления позволяет управлять непосредственно самим VPS. Здесь на выбор предоставляется или бесплатная Vesta, или привычная всем ISPmanager. Мне довелось работать с обеими панелями управления, и Vesta мне нравится даже больше — так что пусть вас не смущает слово «бесплатная».

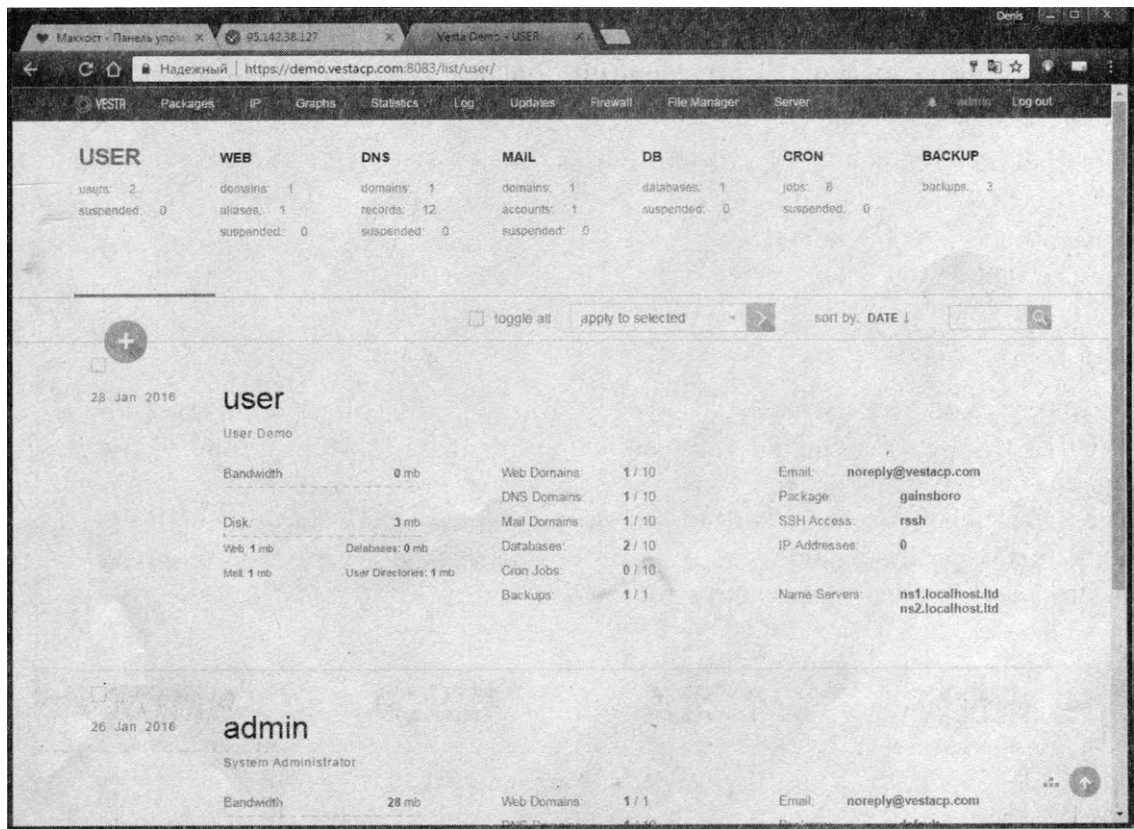


Рис. 43.15. Панель управления Vesta

Тестирование

Поскольку работать с разными провайдерами мне приходится по роду деятельности, тесты могут немного различаться. Дело в том, что в свое время, когда я работал с Макхостом, я не сделал тест на нагрузочное тестирование, поэтому в этом разделе будут приведены результаты только тестов на запись и пропускной способности.

Как обычно, записываем файл, объемом 2 Гбайт, состоящий из нулей. Получаю результат... 107 Мбайт/с (рис. 43.16). Это уровень обычного SATA-диска, но никак не SSD, как нам обещает провайдер.

Как оказалось, провайдер устанавливает ограничение для клиентских VPS на использование дискового ввода/вывода: 1000 IOPS или до 200 Мбайт/с. Но тогда интересно — зачем говорить о том, что используются SSD-диски, и ограничивать скорость на уровне обычного HDD? Это все равно, как на спорткар установить

```
[den@v188134 /]$ sync; dd if=/dev/zero of=~/.temp bs=1M count=2048; sync
2048+0 записей считано
2048+0 записей написано
скопировано 2147483648 байт (2,1 GB), 20,1382 с, 107 MB/c
[den@v188134 /]$
```

Рис. 43.16. Макхост: скорость записи

ограничитель на уровне 100 км/ч. То есть вы платите за SSD, а получаете дисковую подсистему, работающую на уровне HDD. Такого маркетингового хода я не ожидал...

Теперь проверим пропускную способность, воспользовавшись консольным сценарием speedtest.py:

```
$ wget -O - https://raw.githubusercontent.com/sivel/speedtest-cli/master/speedtest.py | python
```

Результаты теста представлены на рис. 43.17. Мы получили отличный **download** в 869,15 Мбит/с и весьма посредственный **upload** в 90,11 Мбит/с (при взаимодействии с сервером Rostelecom в Москве). Результаты как бы и не плохие, но не очень радуйте: для сервера важен upload (исходящий трафик), а не download (входящий), т. к. сервер обычно отдает контент, а не получает его. Тем не менее, будем считать пропускную способность достаточной.

```
[root@v188134 ~]# wget -O - https://raw.githubusercontent.com/sivel/speedtest-cli/master/speedtest.py | python
--2017-01-29 10:15:30-- https://raw.githubusercontent.com/sivel/speedtest-cli/master/speedtest.py
Распознаётся raw.githubusercontent.com... 151.101.36.133
Устанавливается соединение с raw.githubusercontent.com|151.101.36.133|:443... соединение установлено.
Запрос HTTP послан, ожидается ответ... 301 Moved Permanently
Адрес: https://raw.githubusercontent.com/sivel/speedtest-cli/master/speedtest.py. [переход]
--2017-01-29 10:15:31-- https://raw.githubusercontent.com/sivel/speedtest-cli/master/speedtest.py
Распознаётся raw.githubusercontent.com... 151.101.36.133
Устанавливается соединение с raw.githubusercontent.com|151.101.36.133|:443... соединение установлено.
Запрос HTTP послан, ожидается ответ... 200 OK
Длина: 44521 (43K) [text/plain]
Saving to: «STDOUT»

100%[=====>] 44.521 --K/s в 0,08s

2017-01-29 10:15:32 (526 KB/s) - written to stdout [44521/44521]

Retrieving speedtest.net configuration...
Testing from McHost (95.142.38.127)...
Retrieving speedtest.net server list...
Selecting best server based on ping...
Hosted by Rostelecom (Moscow) [1.61 km]: 30.417 ms
Testing download speed.....
Download: 869.15 Mbit/s
Testing upload speed.....
Upload: 90.11 Mbit/s
root@v188134 ~]#
```

Рис. 43.17. Макхост: пропускная способность

Выводы

Преимущества Макхост:

- регулярные акции, бонусы, партнерская программа;
- возможность купить SSL-сертификат от Comodo за 1000 рублей в год (если заказывать сертификат самостоятельно, то он обойдется в 99 евро в год);
- предустановленная панель управления Vesta;
- услуга по чистке сайта от вирусов и устранению уязвимостей;
- высокий download, умеренный upload.

Недостатки:

- высокая стоимость VPS-сервера;
- низкая скорость SSD-диска — на уровне диска HDD;
- образы с предустановленной панелью управления присутствуют для версий дистрибутивов ниже CentOS 7, Ubuntu 14 и 16, Debian 8. Если нужно использовать дистрибутивы новее и требуется панель управления, придется устанавливать ее самостоятельно.

43.2.4. UltraVDS

О ценах

Компания «UltraVDS» предоставляет VDS, работающий под управлением Windows. Не всегда бывает нужен VDS-сервер, работающий под управлением Linux, а провайдеров, предоставляющих в аренду Window-серверы не так уж и много. UltraVDS — один из таких провайдеров.

Средняя конфигурация обойдется в 1120 рублей в месяц: 2 ядра (Xeon E5 2,2 ГГц), 2 Гбайт оперативной памяти, 40 Гбайт SSD, один IP-адрес, канал 200 Мбит/с, лицензия Windows Server 2003-2016. Такой конфигурации будет достаточно как для корпоративного Web-портала, так и для 1С. Желая сэкономить можно предложить базовую конфигурацию (1 ядро, 1 Гбайт ОЗУ, 20 Гбайт HDD) за 360 рублей в месяц. Не забывайте, что в стоимость входит лицензия Windows Server, поэтому такую цену можно считать приемлемой. Тем более, что при оплате за год цена средней конфигурации снижается до 896 рублей в месяц (рис. 43.18).

Здесь есть и конфигуратор — вы можете создать свою собственную конфигурацию. Например, если вам нужно 2 ядра, 2 Гбайт оперативной памяти, но SSD-диск на 20 Гбайт, то нет смысла покупать среднюю конфигурацию и переплачивать (рис. 43.19).

Создание сервера

Создание сервера занимает примерно 5 минут, может, немного меньше. На рис. 43.20 показано, как выглядит панель управления сервером. Кстати, используемая конфигурация (2 ядра, 4 Гбайт ОЗУ, 60 Гбайт SSD, один IP-адрес) мне обошлась в 1780 рублей.



Рис. 43.18. UltraVDS: тарифы

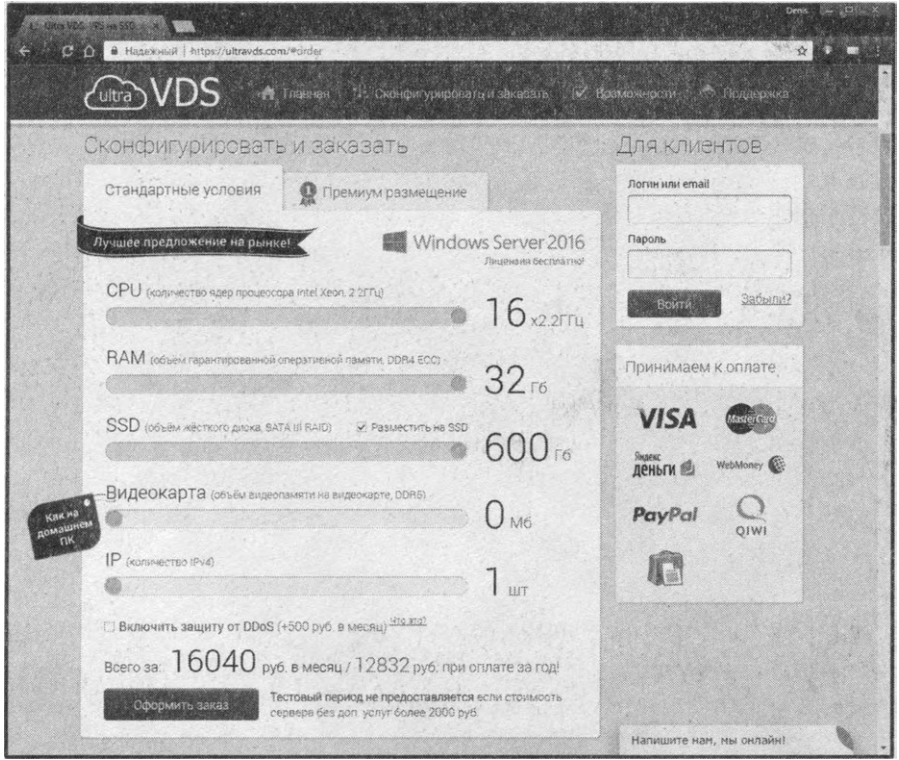


Рис. 43.19. UltraVDS: конфигуратор

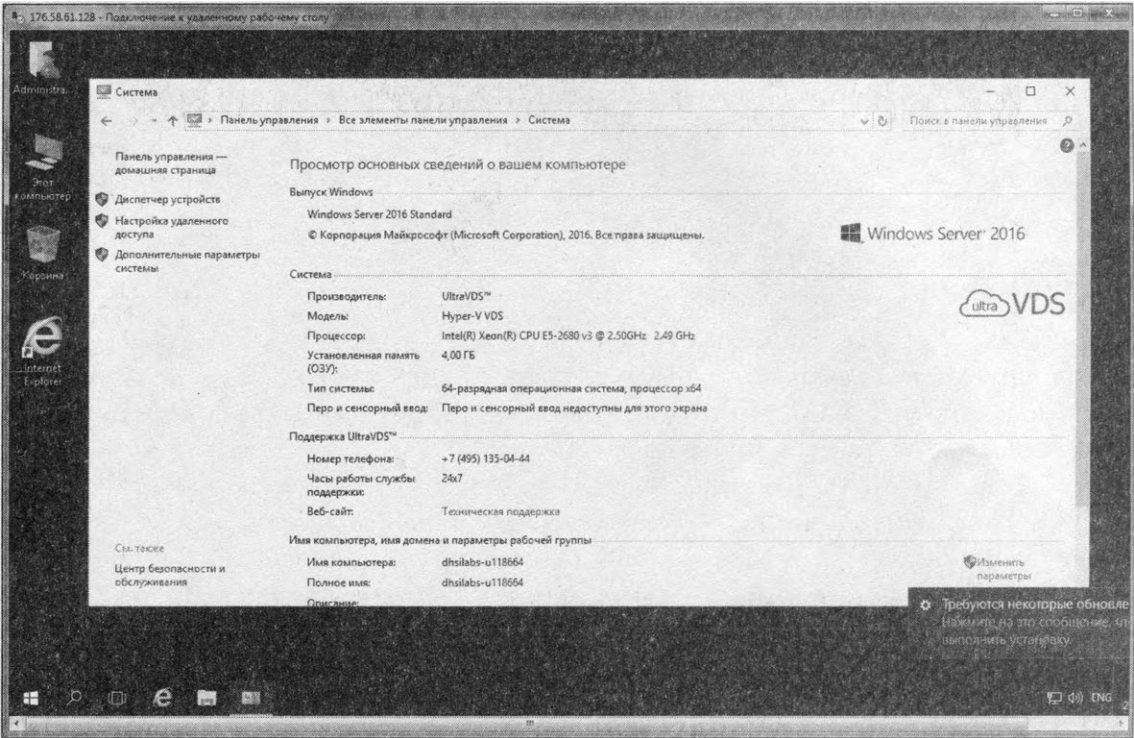


Рис. 43.20. UltraVDS: панель управления

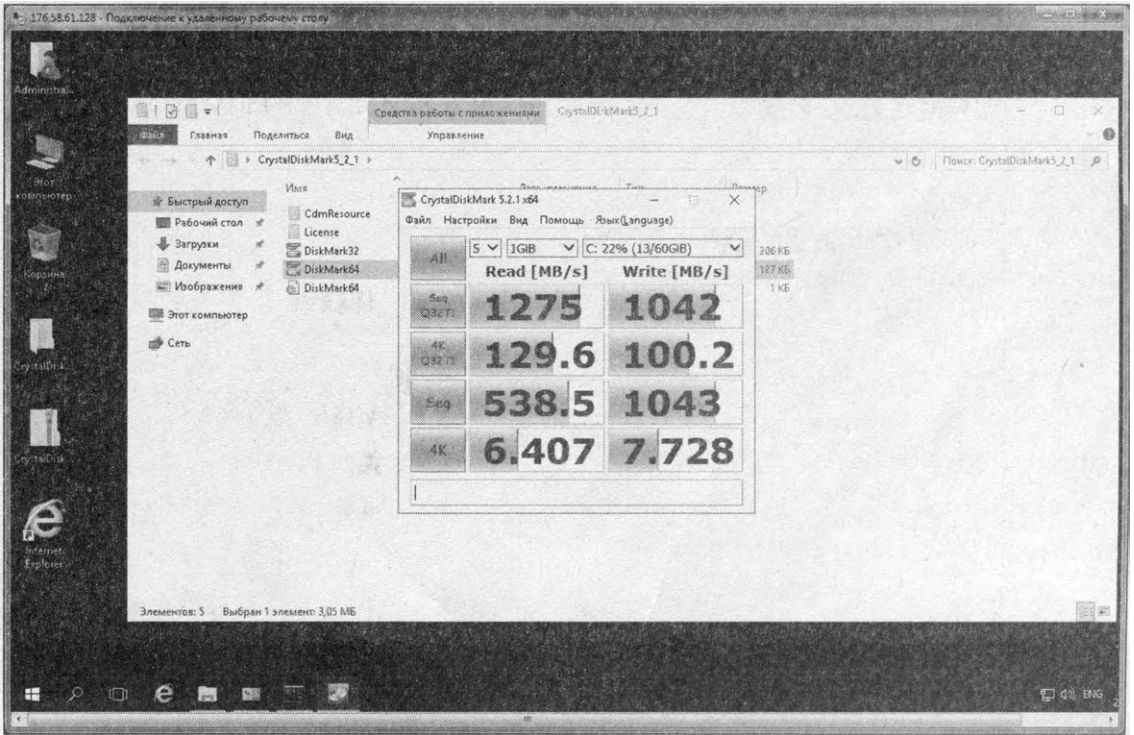


Рис. 43.21. UltraVDS: подключение по RDP

Провайдер предоставляет трехдневный бесплатный тестовый период. Однако он доступен не для всех конфигураций — стоимость конфигурации не должна превышать 2000 рублей в месяц. Так что попробовать VDS можно совершенно бесплатно и потом уже решить, стоит ли его оплачивать дальше или нет.

Подключение осуществляется по RDP (англ. Remote Desktop Protocol, протокол удаленного рабочего стола) — это же Windows (рис. 43.21).

Тестирование

Для чего вы будете использовать Windows-сервер — только вам известно. Поэтому нагрузочного тестирования этого Web-сервера я не производил. Зато произвел тестирование SSD-диска утилитой Crystal DiskMark. Результаты SSD порадовали — безо всяких компромиссов и оговорок (см. рис. 43.21). Например, тот же Kingston UV400 (емкость 480 Гбайт) на одном из моих компьютеров показал только 502 Мбайт/с в тесте на последовательное чтение и 484 Мбайт/с в тесте на последовательную запись. А здесь 1275 и 1043 Мбайт/с соответственно.

На сайте UltraVDS сообщается, что пропускная полоса к серверу составляет более 200 Мбит/с. Для тестирования пропускной способности воспользуемся тем же speedtest.net, а его результаты поместим в табл. 43.1.

Таблица 43.1. UltraVDS: результаты тестирования пропускной способности

Хост	Download, Мбит/с	Upload, Мбит/с	Пинг, мс
CLN, Moscow http://beta.speedtest.net/result/5998827299	599,73	102,79	3
Rostelecom, Krasnodar http://beta.speedtest.net/result/5998838629	258,45	46,71	31
MegaFon, Moscow http://beta.speedtest.net/result/5998841560	325,34	58,44	20
Rostelecom, Saint Petersburg http://beta.speedtest.net/result/5998847918	418,90	69,92	12
DEAC, Amsterdam http://beta.speedtest.net/result/5998855369	143,75	50,53	66
Orange, Paris http://beta.speedtest.net/result/5998852439	202,57	59,96	54
Rostelecom, Yaroslavl http://beta.speedtest.net/result/5998859940	447,91	96,48	11

Думаю, картина понятна (рис. 43.22). Но мне все же хотелось видеть более симметричный канал, пусть даже и с меньшей скоростью. К **download** претензий нет, а вот

upload хотелось бы видеть выше. С другой стороны, у вас есть тестовый период, на протяжении которого вы можете найти время, чтобы подключиться к VDS и посмотреть на пропускную способность до своего офиса.

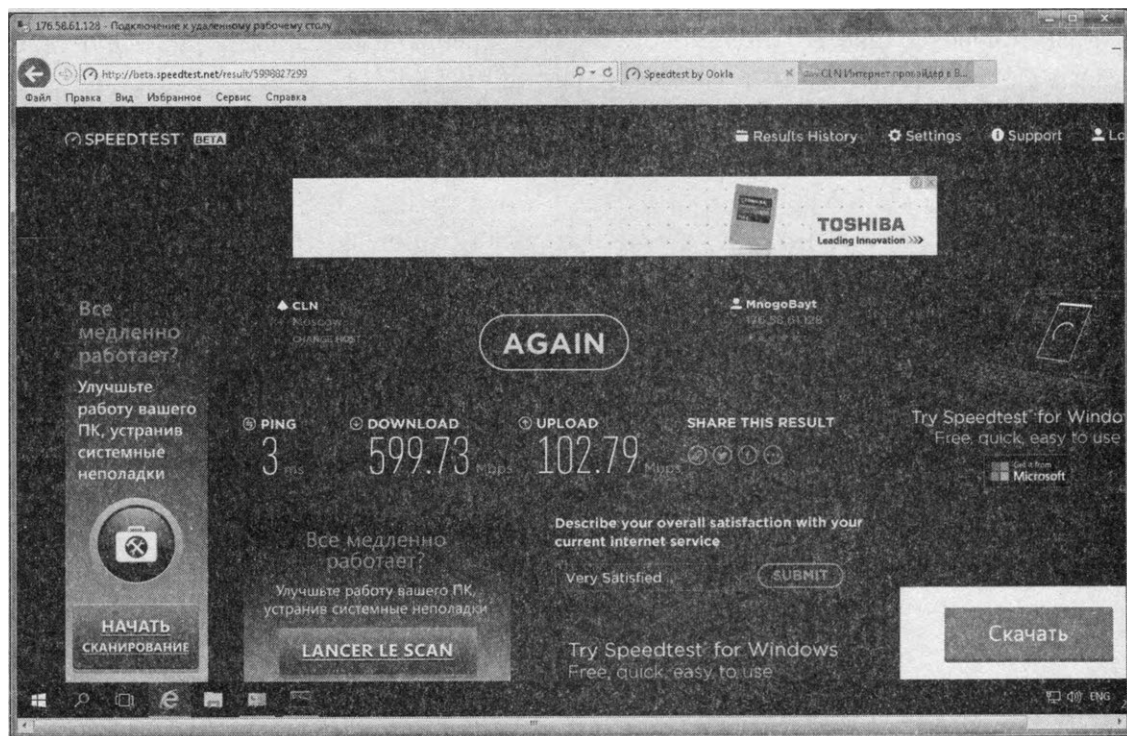


Рис. 43.22. UltraVDS: тестирование пропускной способности (CLN, Moscow)

Выводы

Настало время сделать выводы. К преимуществам UltraVDS можно отнести:

- доступную стоимость VDS;
- высокую скорость интернет-соединения, отличный «пинг» по Москве;
- высокую производительность SSD;
- полную автоматизацию процесса создания серверов, понятную панель управления;
- трехдневный тестовый период.

Есть, конечно, и ложка дегтя:

- в некоторых случаях получаем отличный download, но заметно более низкий upload, хотя полученные 100 Мбит/с являются стандартной скоростью услуги VDS на рынке;
- в личном кабинете указаны среднесуточные лимиты на нагрузку CPU (40%) и HDD/SSD (зависит от типа выбранного диска и его размера), для SSD 60 Гбайт — это 2000 IOPS), однако в ходе проведения нагрузочных тестов действия этих лимитов не выявлены.

43-2.5. 1cloud

О ценах

У 1 cloud есть удобный конфигуратор, позволяющий подобрать сервер на любые деньги. Конфигурация тестируемого в обзоре сервера приведена на рис. 43.23. Это двухъядерный процессор, 5 Гбайт оперативки, 10 Гбайт SSD-диска, операционная система Ubuntu 16.04, базовая производительность. Такая конфигурация обойдется вам в 1995 рублей в месяц.

Инфраструктура ▾ Отзывы SLA API ▾

Д > Услуги > VPS/VDS

Удаленный VPS/VDS сервер в аренду за 2 мин!

Центр обработки данных

SDN, Россия, Санкт-Петербург ▾

Операционная система

Ubuntu 16.04 x64 ▾

Панель управления хостингом

Не устанавливать ▾

Производительность оборудования

Базовая ☒ Высокая

Базовый пул построен на базе серверов Dell PowerEdge R810 (2 GHz), HP BL460 (3 GHz)

Резервное копирование

Не нужно ▾

Дополнительная информация ▾

Стоимость час сутки месяц

1995 Р

с налогами

CPU Core: 2

RAM: 5 Гб

Storage: SAS SSD 10 Гб

При единовременной оплате:

от 5000 Р бонус к сумме **10 %**

от 10000 Р бонус к сумме **20 %**

Ваш E-mail:

Заказать

Рис. 43.23. 1cloud: стоимость сервера

При базовой производительности сервер будет помещен в пул, построенный на базе серверов Dell PowerEdge R810 (2 GHz) и HP BL460 (3 GHz). При высокой производительности будут использоваться серверы Cisco B200 (3 GHz) и Dell PowerEdge R810 (2 GHz), но и стоимость будет выше — примерно на 800 рублей/мес.

После того, как вы нажмете кнопку **Заказать** и подтвердите электронную почту, сервер будет готов примерно через 2-3 минуты. В панели управления услугой (рис. 43.24) можно просмотреть, работает ли сервер, включить, выключить, перезагрузить его и т. д. Здесь же приводятся IP-адрес и пароль пользователя root.

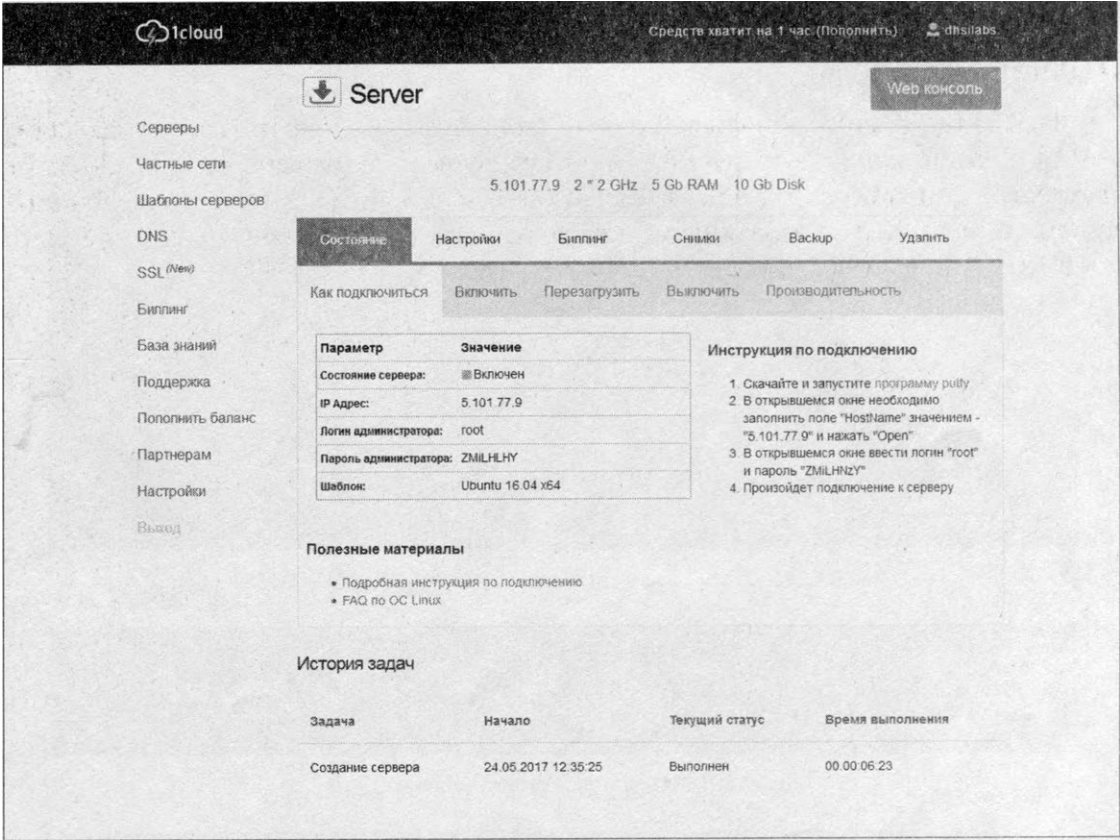


Рис. 43.24. 1cloud: панель управления услугой

Тестирование

Тест строился по привычной схеме: пропускная способность, производительность SSD и нагрузочное тестирование. Так вот, получив довольно посредственные результаты пропускной способности (рис. 43.25), я уже было начал думать, что рекомендовать этот продукт читателям не смогу. Честно говоря, пропускной способности в 20 Мбит/с (хорошо, хоть канал синхронный) будет маловато при серьезных нагрузках, которые может выдерживать сервер. Оказалось, что если пропускной способности мало, то ее можно докупить,— один дополнительный Мбит/с за 10 рублей в месяц. То есть, дополнительные 10 Мбит/с обойдутся в 100 рублей/мес.

Производительность SSD оказалась на высоте, а не как у Макхост, где платишь как бы за SSD, а получаешь скорость HDD, и при этом вам доказывают, что это нормально. Здесь же все без уловок: 502 Мбайт/с — более чем достойный результат (рис. 43.26).

А вот результаты нагрузочного тестирования удивили. Тестирование производилось с настройками по умолчанию и без установки каких-либо приложений, поэтому после установки реального приложения результаты могут быть хуже. Но все равно результаты очень и очень неплохие. Первым делом я протестировал 255 и 500 одновременных соединений (как обычно, по 10 повторов) и получил доступ-

```

root@Ubuntu1604x64:~# python speedtest.py
Retrieving speedtest.net configuration...
Testing from II-Grad (5.101.77.9)...
Retrieving speedtest.net server list...
Selecting best server based on ping...
Hosted by Rostelecom (Moscow) [1.61 km]: 11.415 ms
Testing download speed.....
Download: 19.50 Mbit/s
Testing upload speed.....
Upload: 20.68 Mbit/s
root@Ubuntu1604x64:~# python speedtest.py
Retrieving speedtest.net configuration...
Testing from II-Grad (5.101.77.9)...
Retrieving speedtest.net server list...
Selecting best server based on ping...
Hosted by Rostelecom (Moscow) [1.61 km]: 12.971 ms
Testing download speed.....
Download: 19.54 Mbit/s
Testing upload speed.....
Upload: 20.53 Mbit/s
root@Ubuntu1604x64:~#

```

Рис. 43.25. 1cloud: результаты теста пропускной способности

```

root@Ubuntu1604x64:~# dd if=/dev/zero of=temp bs=1M count=2048
2048+0 records in
2048+0 records out
2147483648 bytes (2.1 GB, 2.0 GiB) copied, 4.27717 s, 502 MB/s
root@Ubuntu1604x64:~# █

```

Рис. 43.26. 1cloud: производительность SSD

```

Transaction rate:      495.54 trans/sec
Throughput:           1.51 MB/sec
Concurrency:          18.43
Successful transactions: 5000
Failed transactions:   0
Longest transaction:   1.15
Shortest transaction:  0.00

FILE: /var/log/siege.log
You can disable this annoying message by editing
the .siegerc file in your home directory; change
the directive 'show-logfile' to false.
root@Ubuntu1604x64:~# siege -c 1000 -r 10 -d 1 5.101.77.9
** SIEGE 3.0.8
** Preparing 1000 concurrent users for battle.
The server is now under siege..      done.

Transactions:         10000 hits
Availability:          100.00 %
Elapsed time:          11.61 secs
Data transferred:      30.36 MB
Response time:          0.16 secs
Transaction rate:      861.33 trans/sec
Throughput:            2.62 MB/sec
Concurrency:           133.66
Successful transactions: 10000
Failed transactions:    0
Longest transaction:    3.83
Shortest transaction:   0.00

FILE: /var/log/siege.log
You can disable this annoying message by editing
the .siegerc file in your home directory; change
the directive 'show-logfile' to false.
root@Ubuntu1604x64:~# █

```

Рис. 43.27. 1cloud: результаты нагрузочного тестирования (500 и 1000 одновременных соединений)

ность 100%. Затем я попробовал 1000 одновременных соединений (рис. 43.27), и снова доступность составила 100%!

Раз так, то, набравшись наглости, я попросил siege сгенерировать 2000 одновременных соединений с 10 повторами. На этот раз все прошло не так гладко, и доступность составила всего 83,57% (рис. 43.28).

Что ж, понизив количество подключений до 1500, я получил снова доступность 100%. Выходит, что сервер от lcloud.ru оказался весьма выносливым и уверенно выдерживает нагрузку в 1500 одновременных подключений.

```
[error] descriptor table full sock.c:119: Too many open files
[error] descriptor table full sock.c:119: Too many open files
[error] descriptor table full sock.c:119: Too many open files
[error] descriptor table full sock.c:119: Too many open files
[error] descriptor table full sock.c:119: Too many open files
[error] descriptor table full sock.c:119: Too many open files
[error] descriptor table full sock.c:119: Too many open files
[error] descriptor table full sock.c:119: Too many open files
[error] descriptor table full sock.c:119: Too many open files
[error] descriptor table full sock.c:119: Too many open files
[error] descriptor table full sock.c:119: Too many open files
[error] descriptor table full sock.c:119: Too many open files
done.
siege aborted due to excessive socket failure; you
can change the failure threshold in $HOME/.siegerc

Transactions:          5544 hits
Availability:          83.57 %
Elapsed time:           4.88 secs
Data transferred:      16.84 MB
Response time:          0.50 secs
Transaction rate:      1136.07 trans/sec
Throughput:             3.45 MB/sec
Concurrency:           562.69
Successful transactions: 5544
Failed transactions:    1090
Longest transaction:    3.39
Shortest transaction:    0.00

FILE: /var/log/siege.log
You can disable this annoying message by editing
the .siegerc file in your home directory; change
the directive 'show-logfile' to false.
root@ubuntu1604x64:~#
```

Рис. 43.28. 1cloud: 2000 одновременных подключений

Выводы

Преимущества VDS от 1 cloud:

- высокая доступность сервера при высоких нагрузках;
- динамическая балансировка — если физический хост с вашим сервером окажется перегруженным, он будет автоматически перенесен без простоя на другой хост;
- наличие разнообразных образов с операционными системами Windows, Linux, FreeBSD — обычно провайдеры предоставляют либо Linux-серверы, либо Windows-серверы;

- гибкость в процессе использования сервера — его конфигурацию можно изменить в любой момент. Подобный подход позволяет не только улучшить конфигурацию при необходимости, но и экономить деньги, — если видите, что заказанная конфигурация не используется на 100%, можно временно ее ухудшить, чтобы платить меньше;
- миграция дисков с SAS на SSD и обратно без простоя;
- большое количество дополнительных возможностей: создание бесплатных (!) частных сетей, в которых можно объединить свои виртуальные серверы на скорости 1 Гбит/с, создание снапшотов, собственных шаблонов и т. д.;
- возможность бесплатного тестирования.

Недостатки:

- высокая стоимость;
- низкая пропускная способность по умолчанию, а за дополнительную пропускную способность нужно доплачивать.

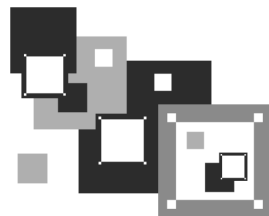
43.3. Заключение

Надеюсь, приведенная в этой главе информация поможет вам выбрать виртуальный сервер. Сравнение тестируемых провайдеров (табл. 43.2) показывает, что победителем по соотношению цена/качества является Спринтхост, — всего за 600 рублей в месяц он предоставляет полноценный VDS с неплохими характеристиками.

Таблица 43.2. Сводная информация

	Джино	Спринтхост	Макхост	UltraVDS	1cloud
тtx	2 ядра, 2 Гбайт ОЗУ и 20 Гбайт SSD+HDD	2 ядра, 2 Гбайт ОЗУ, 32 Гбайт SSD	2 ядра, 2 Гбайт ОЗУ, 10 Гбайт SSD	2 ядра, 2 Гбайт ОЗУ, 40 Гбайт SSD	2 ядра, 2 Гбайт ОЗУ, 20 Гбайт SSD
Тип	VPS	VDS	VPS	VDS	VDS
Трафик	∞	∞	∞	∞	∞
Пропускная способ- ность, Мбит/с	синхронный, 79	101 download, 111 upload	869 download, 90 upload	599 download, 102 upload	синхронный, 20
Скорость записи, Мбайт/с	805	286	107	1043	502
IP-адрес	89 рублей в месяц	+	+	+	+
Цена, руб./месяц	500	600	879	1120	950

ГЛАВА 44



Сервер виртуализации OpenVZ

44.1. Способы виртуализации

Самый простой способ создать виртуальную машину — воспользоваться такими продуктами, как VMware и VirtualBox (см. главу 18). Первый продукт является коммерческим даже для Linux, а второй — абсолютно свободный и ничем не уступает VMware. Выбор какого-либо из них — дело предпочтений пользователя. Каждый выбирает то, к чему он привык (или что может себе позволить).

Эти способы виртуализации мы рассматривать не станем — использовать VirtualBox (как и VMware) очень просто, и создать виртуальную машину с их помощью может даже ребенок. Совсем другое дело — виртуализация на уровне ядра. Здесь все гораздо сложнее в настройке, нет удобного графического интерфейса, но если вам нужна не одна виртуальная машина, а целая ферма виртуальных серверов, то гораздо эффективнее задействовать именно такую виртуализацию. К тому же, используя технологию виртуализации на уровне ядра, вы получаете очень удобные (хоть и не графические) средства управления всеми виртуальными серверами.

Технология виртуализации уровня ядра позволяет на одном физическом сервере запускать изолированные копии различных операционных систем, называемых *виртуальными окружениями* (virtual environments, VE) или *виртуальными контейнерами*.

Типичное применение систем виртуализации на уровне ядра — хостинговые компании. Современные серверы настолько производительные, что в большинстве случаев можно разделить их ресурсы (в первую очередь речь идет о процессорном времени и оперативной памяти) на несколько виртуальных серверов. Соответственно, потом эти серверы можно продать клиентам компании. Сейчас оперативной памятью в 32 Гбайт и 8-ядерным процессором никого не удивишь. Но в большинстве случаев для небольшого сайта вполне достаточно от 512 Мбайт до 1 Гбайт оперативной памяти. Получается, что вы можете оставить некоторый объем ресурсов для операционной системы, а остальное разделить между виртуальными Web-серверами и продать их потребителям. Вот здесь мы и разберемся, как это сделать (разделить, а не продать!).

Технологий виртуализации на уровне ядра Linux существуют две: OpenVZ (и ее развитие Virtuozzo) и KVM (Kernel-based Virtual Machine). Одни администраторы

утверждают, что лучше KVM, другие — OpenVZ. На мой же взгляд, у каждой системы есть свои преимущества и недостатки, а выбор той или иной из них зависит от поставленной задачи (см. главу 43).

К преимуществам OpenVZ можно отнести высокую производительность и низкую стоимость виртуальных серверов (далее вы поймете, почему речь идет о стоимости), а также эффективную систему распределения ресурсов процессора и памяти между виртуальными серверами, — каждый сервер потребляет ровно столько ресурсов, сколько ему нужно, а оставшиеся ресурсы могут быть использованы другими серверами.

Такая система распределения ресурсов очень выгодна хостинговой компании. Например, ваш сервер оснащен 10 Гбайт оперативной памяти, которую вы желаете разделить между виртуальными серверами. Пусть каждому серверу вы планируете выделить по 1 Гбайт — соответственно, вы можете создать максимум 10 серверов. Однако на практике выяснилось, что максимальное потребление памяти каждым из серверов не превышает 700 Мбайт, а обычно составляет 512 Мбайт. Следовательно, примерно 300 Мбайт с каждого сервера вы можете снова продать, — т. е. создать еще 3 дополнительных сервера и получать с них прибыль. Такая система выгодна как хостеру, так и клиенту, — ведь виртуальные серверы, построенные на технологии OpenVZ, обычно дешевле, чем те, которые построены на базе KVM. И никакого обмана — клиенту нужно сообщить, какая система виртуализации используется, а он уже должен сам решить, что ему важнее: финансы или жесткое выделение ресурсов для сервера.

Конечно, у технологии OpenVZ есть и недостатки — она использует на хост-машине одно модифицированное ядро. Это означает, что ядро разделяется между всеми контейнерами OpenVZ, и поэтому можно запустить только контейнеры, содержащие Linux, — нет никакого способа запустить в OpenVZ другую операционную систему, скажем, Windows или FreeBSD.

Второй недостаток OpenVZ также связан с ядром. Ее модули не входят в состав официального ядра (с kernel.org) — следовательно, вам понадобится специальное ядро OpenVZ. Само ядро — не проблема, проблема в его версии. Несмотря на то, что уже вышла 4-я версия ядра Linux, ядро OpenVZ имеет версию 2.6. Соответственно, в виртуальную машину можно будет установить только дистрибутивы, поддерживающие эту версию ядра (помните, что виртуальные контейнеры используют то же ядро, что и хост-машина). Впрочем, для Web-сервера версия ядра Linux не столь и важна, да и последняя версия дистрибутива вам тоже вряд ли понадобится.

Но и это еще не все. На сервере с ядром OpenVZ не будут работать сервисы, тесно интегрирующиеся с ядром, такие как IpSec и OpenVPN. Увы, но это так. Вы также должны иметь в виду, что у всех виртуальных контейнеров общая виртуальная память, — OpenVZ использует подкачку, но на уровне всех хост-машин.

Напоследок еще один недостаток, о котором вы должны знать, — кроме общей виртуальной памяти у всех контейнеров общий дисковый кэш, что не очень хорошо сказывается на производительности работы с диском, особенно если планируется, что виртуальные машины должны записывать на диск большие объемы данных. Поэтому сервер баз данных на базе OpenVZ — не самое лучшее решение.

Таким образом, вы должны задуматься об использовании KVM в следующих случаях:

- если нужно использовать в виртуальных контейнерах операционные системы, отличные от Linux, — например, FreeBSD или Windows;
- если в виртуальных контейнерах должны работать системы, построенные на дистрибутивах Linux со «свежими» ядрами;
- если нужно использовать IPSec и OpenVPN;
- если нужна высокая производительность дисковой подсистемы.

Исходите из поставленных задач

Систему виртуализации следует выбирать, исходя из поставленных задач. Например, если нужно запустить в виртуальной машине операционную систему Windows, то придется использовать VMware, VirtualBox или KVM, поскольку OpenVZ не поддерживает запуск Windows.

44.2. Установка OpenVZ

Настройку сервера виртуализации следует начинать с установки операционной системы на хост-машину. Учитывая версию ядра, которая поддерживается OpenVZ, выбор невелик: или CentOS 6.6, или Debian 7.

По адресу <http://openvz.org/Download/template/precreated> имеется список уже созданных виртуальных контейнеров— дистрибутивов, которые могут работать с OpenVZ, и вам остается выбрать один из них. Конечно, из контейнера вы Linux на хост-машину не установите, но зато этот список поможет вам сориентироваться с дистрибутивом, который вы сможете использовать в качестве операционной системы хост-машины. В этом списке есть и довольно «свежие» дистрибутивы: Fedora 22, OpenSUSE 13.2. Однако не забывайте, что перечисленные в списке контейнеры — это предварительно созданные сборки Linux на базе ядра 2.6.

Далее настройку сервера виртуализации мы рассмотрим на примере дистрибутива CentOS.

Итак, для установки OpenVZ и всего необходимого введите команды:

```
# wget -P /etc/yum.repos.d/ http://ftp.openvz.org/openvz.repo
# rpm --import http://ftp.openvz.org/RPM-GPG-Key-OpenVZ
# yum install vzctl vzquota ploop
```

Первая команда подключает репозиторий OpenVZ, вторая — импортирует GPG-ключ, а третья — устанавливает сами пакеты (рис. 44.1).

После установки пакетов перезагрузите систему и выберите при ее загрузке ядро OpenVZ (рис. 44.2). Такая перезагрузка нужна не для запуска каких-либо служб, а для загрузки установленного ядра, так что убедитесь, что при загрузке выбрано именно ядро OpenVZ, и при необходимости отредактируйте файлы конфигурации загрузчика, чтобы увериться, что это ядро загружается по умолчанию.

Пакет	Архитектура Версия	Репозиторий	Размер
Установка:			
ploop	x86_64 1.12.1-1	openvz-utils	54 k
vzctl	x86_64 4.8-1	openvz-utils	141 k
vzquota	x86_64 3.1-1	openvz-utils	93 k
Установка зависимостей:			
e2fsprogs-resize2fs-static	x86_64 1.42.11-1.ovz	openvz-utils	77 k
libcgroup	x86_64 0.40.rc1-15.el6_6	updates	129 k
ploop-lib	x86_64 1.12.1-1	openvz-utils	146 k
vzctl-core	x86_64 4.8-1	openvz-utils	283 k
vzkernel	x86_64 2.6.32-042stab094.8	openvz-kernel-rhel6	30 M
vzstats	noarch 0.5.3-1	openvz-utils	23 k
Результат операции			
Установить 9 пакет(а,ов)			
Объем загрузки: 31 М			
Будет установлено: 126 М			
Продолжить? [y/N]:			

Рис. 44.1. Процесс установки OpenVZ

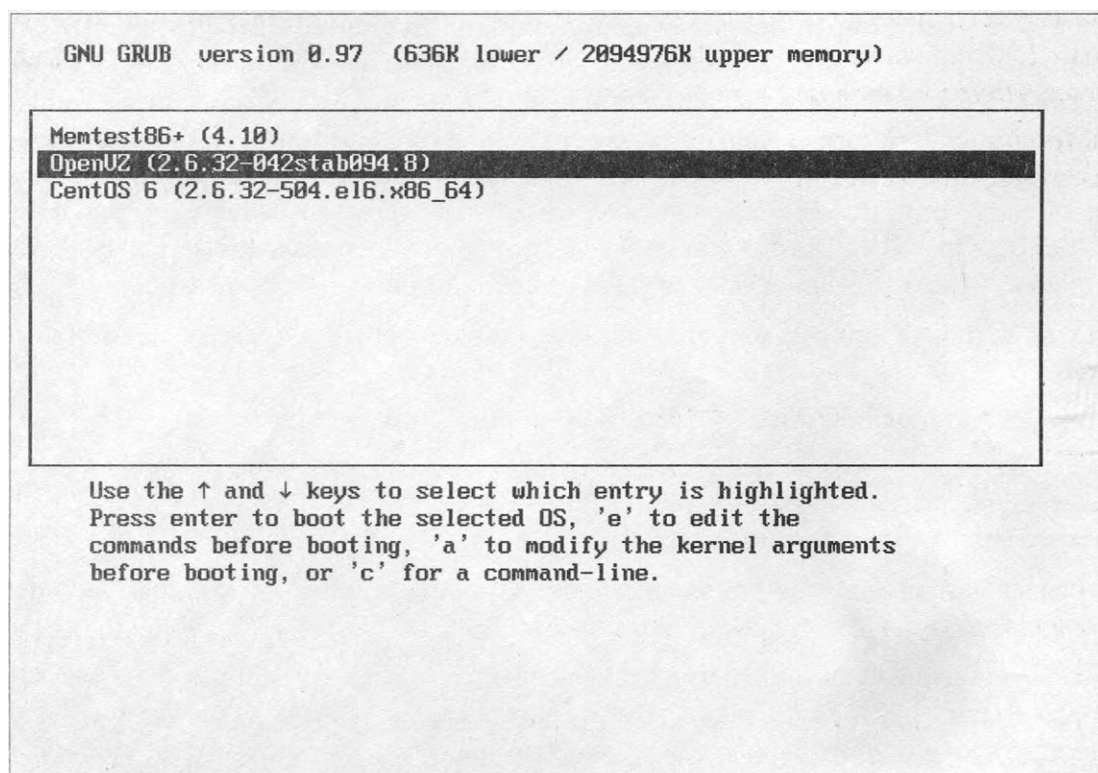


Рис. 44.2. Загрузка сервера виртуализации

44.3. Создание и настройка виртуального контейнера

В отличие от VMware и VirtualBox, где нужно самостоятельно устанавливать в виртуальную машину гостевую операционную систему, в OpenVZ требуется лишь скачать шаблон с уже предустановленной операционной системой. Такие готовые шаблоны доступны по адресу: <http://openvz.org/Download/template/precreated>.

По состоянию на 29 июня 2017 года там находились различные версии CentOS, Debian (в том числе Debian 8), Fedora (доступна даже версия 3), Ubuntu (последняя версия 16.04, а 16.10 доступна в качестве бета-шаблона), а также OpenSUSE 13.2.

Загрузите любой шаблон и, не распаковывая его, поместите в каталог `/vz/templates/cache`. Еще раз подчеркиваю — распаковывать архив шаблона не нужно, просто скачайте его и поместите в указанный каталог.

Для создания контейнера используется команда `vzctl`:

```
# vzctl create 101 --ostemplate suse-13.2-x86_64
```

Первый параметр — это идентификатор виртуальной машины. Мне нравится использовать идентификаторы 101, 102, 103 и т. д. (и чуть позже я поясню, почему это удобно), вы же можете задать любой другой числовой идентификатор, — например, 1, 1001 и т. п. Процесс создания контейнера не быстрый — придется подождать... Когда все будет готово, вы увидите сообщение **Container private area was created**. Команда `vzctl` также создаст файл конфигурации, который будет называться `/etc/vz/conf<идентификатор>.conf`.

Настройка контейнера осуществляется двумя способами: или путем редактирования файла конфигурации, или путем серии команд `vzctl set`, — например:

```
# vzctl set <идентификатор> --onboot yes --save
# vzctl set <идентификатор> --hostname vs1.firma.ru --save
# vzctl set <идентификатор> --ipadd 192.168.5.101 --save
# vzctl set <идентификатор> --nameserver 8.8.8.8
# vzctl set <идентификатор> --cpus 2 --save
# vzctl set <идентификатор> --ram 1024M --save
# vzctl set <идентификатор> --swap 2G --save
# vzctl set <идентификатор> --diskspace 15G --save
```

Рассмотрим эти команды подробнее:

- первая команда здесь указывает, что наш виртуальный сервер должен запускаться при запуске системы;
- вторая — присваивает ему имя компьютера;
- третья — указывает IP-адрес виртуального сервера. В корпоративной сети удобно использовать такую систему идентификации, к какой прибегаю я. Поскольку реальные (физические) машины занимают IP-адреса до 192.168.5.100, а виртуальные — после .100, то сразу становится понятно, какой IP-адрес какому ком-

пьютеру принадлежит: физическому или виртуальному. Конечно, можно было бы создать отдельную подсеть для виртуальных машин, но я решил обойтись тем, что есть;

- следующая команда задает сервер имен для виртуальной машины. Вы можете указать несколько параметров `--nameserver` (до четырех);
- параметр `--cpus` определяет количество процессоров (ядер процессоров). Мною было указано два ядра, но в реальных условиях вам вряд ли захочется предоставлять виртуальной машине более одного ядра (процессора);
- параметр `--ram` задает объем оперативной памяти, а параметр `--swap` — виртуальной. Здесь заданы весьма хорошие для виртуального сервера параметры. Реальные VPS-провайдеры обычно жадничают и редко предоставляют серверы с более чем 512 Мбайт оперативной памяти;
- последний параметр задает объем дискового пространства, доступного виртуальной машине (в данном случае 15 Гбайт).

Как уже было отмечено ранее, можно также настроить контейнер и редактированием файла конфигурации виртуального сервера, но обычно в этом нет необходимости, поскольку весь процесс настройки (который вы будете выполнять всего лишь один раз в жизни каждого виртуального сервера) можно легко выполнить с помощью команды `vzctl`.

44.4. Запуск виртуальной машины

Закончив настройку, можно запустить виртуальную машину. Для этого используется команда `vzctl start`:

```
# vzctl start <идентификатор>
```

Для остановки виртуальной машины служит команда `vzctl stop`, а для ее перезапуска — `vzctl restart`:

```
# vzctl stop <идентификатор>
# vzctl restart <идентификатор>
```

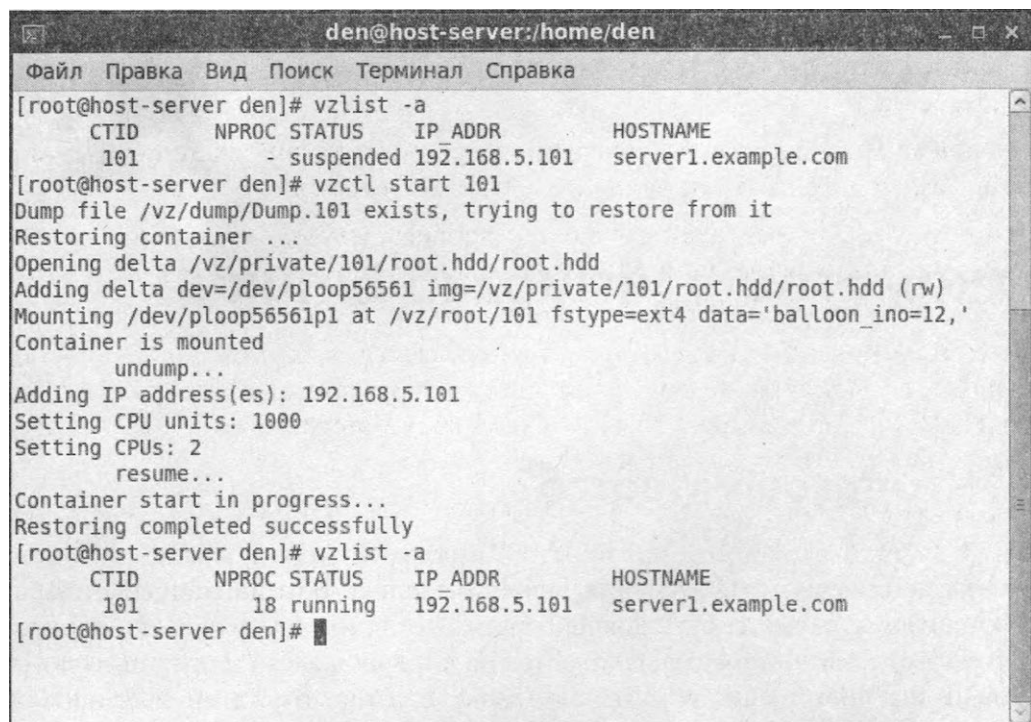
После первого запуска нужно первым делом установить пароль `root`:

```
# vzctl exec <идентификатор> passwd
```

Осталось ввести команду `vzlist -a`, чтобы просмотреть список виртуальных машин и их состояние (рис. 44.3). Здесь видно, что сначала состояние виртуальной машины было `suspended` (это уже не первый запуск машины, до первого запуска в колонке **STATUS** был прочерк), затем я запустил виртуальную машину, и теперь ее состояние `running`. Колонка **NPROC** отображает количество процессов, которые выполняются в виртуальной машине. Назначение колонок **IP_ADDR** и **HOSTNAME**, думаю, понятно и так.

Что делать дальше? Войти в виртуальный сервер по `ssh` и приступить к его настройке так, как если бы это был самый обычный физический сервер:

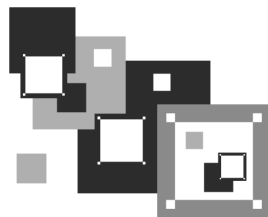
```
# ssh <IP-адрес>
```



```
den@host-server:/home/den
Файл Правка Вид Поиск Терминал Справка
[root@host-server den]# vzlist -a
  CTID      NPROC STATUS   IP_ADDR      HOSTNAME
    101          - suspended 192.168.5.101 server1.example.com
[root@host-server den]# vzctl start 101
Dump file /vz/dump/Dump.101 exists, trying to restore from it
Restoring container ...
Opening delta /vz/private/101/root.hdd/root.hdd
Adding delta dev=/dev/ploop56561 img=/vz/private/101/root.hdd/root.hdd (rw)
Mounting /dev/ploop56561p1 at /vz/root/101 fstype=ext4 data='balloon_ino=12,'
Container is mounted
    undump...
Adding IP address(es): 192.168.5.101
Setting CPU units: 1000
Setting CPUs: 2
    resume...
Container start in progress...
Restoring completed successfully
[root@host-server den]# vzlist -a
  CTID      NPROC STATUS   IP_ADDR      HOSTNAME
    101      18 running 192.168.5.101 server1.example.com
[root@host-server den]#
```

Рис. 44.3. Запуск виртуальной машины и просмотр ее состояния

ГЛАВА 45



Знакомство с Virtuozzo Linux

45.1. Что такое Virtuozzo?

Parallels Virtuozzo Containers или просто Virtuozzo (продукт компании Virtuozzo, Inc., <https://virtuozzo.com/>) — уникальное решение, объединяющее гипервизор KVM и виртуализацию на базе контейнеров. В отличие от других подобных продуктов, решение Virtuozzo устанавливается на «голое» железо и представляет собой отдельный дистрибутив Linux (Virtuozzo Linux), который уже оптимизирован для задач виртуализации и хостинга. Все, что нужно, — это взять и установить его на машину, предназначенную быть сервером виртуализации. При этом не требуется устанавливать или компилировать ядро, бороться со всевозможными глюками, и никто не ограничивает вас возможностями ядра Linux версии 2.6 — Virtuozzo использует ядро 3.10 с долгосрочной технической поддержкой.

45.2. Как это работает?

Дистрибутив Virtuozzo Linux, как уже было отмечено, устанавливается на будущий сервер виртуализации, после чего администратор создает, настраивает и запускает контейнеры, или виртуальные машины, каждая из которых становится виртуальным сервером и может работать под управлением различных дистрибутивов Linux.

После того как виртуальный сервер запущен, уже никто не ограничивает администратора в установке и настройке программного обеспечения. Дистрибутивы Linux, устанавливаемые в виртуальные серверы, являются полноценными, а не урезанными копиями.

Далее все зависит от поставленных задач — например, можно превратить виртуальные серверы в Web-серверы и продавать их (типичное решение для VPS-провайдера).

Схема виртуализации представлена на рис. 45.1, позаимствованном из документации по Virtuozzo. Как можно видеть, здесь присутствует «железо» сервера (**Server Hardware**), выделены уровень виртуализации (**OS Virtualization Layer**) и контейнеры (**Container**).

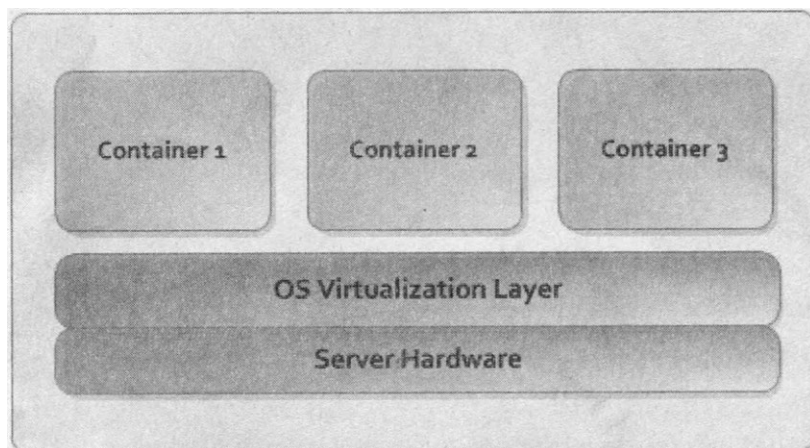


Рис. 45.1. Схема виртуализации

Контейнеры выглядят как независимые серверы под управлением Linux, они не используют для виртуализации эмуляцию аппаратуры, а эффективно разделяют общее ядро и его ресурсы между всеми контейнерами и самим физическим сервером.

Каждый контейнер может использовать ресурсы всего физического сервера, а также эффективно ограничиваться по использованию памяти, процессорного времени, операциям ввода/вывода и сетевому трафику.

Технология контейнерной виртуализации предоставляет наивысшую плотность среди других решений виртуализации. Можно создать и запустить сотни контейнеров на стандартном физическом production-решении. В каждом контейнере может быть только одна операционная система, что упрощает обслуживание и процесс обновления контейнеров.

45.3. Системные требования и ограничения

Системные требования для автономных установок выглядят так:

- ☐ платформа x86-64 с аппаратной поддержкой виртуализации Intel VT-x (с технологией «неограниченного гостя»);
- ☐ минимум 4-ядерный 64-битный процессор;
- ☐ минимум 4 Гбайт оперативной памяти;
- ☐ минимум 64 Гбайт на жестком диске, желательно SSD;
- ☐ сетевой адаптер Ethernet с подключением к сети и с корректным IP-адресом.

Проверить, поддерживает ли ваш Intel-процессор технологию «неограниченного гостя», можно с помощью сценария, расположенного по ссылке:

<https://github.com/qemu/qemu/blob/master/scripts/kvm/vmxcap>.

Запустите его так:

```
python vmxcap.py | grep -i unrest
```

В результате должно быть получено: yes.

Системные требования для размещения серверов в VirtuoZZo Storage Cluster:

- VirtuoZZo 7;
- 1 Гбайт оперативной памяти на каждые 100 Тбайт хранилища;
- 10 Гбайт или более дискового пространства;
- 1 Ethernet-адаптер 1 Гбит/с, статический IP-адрес для каждого адаптера.

Ограничения:

- максимальный объем оперативной памяти (сертифицированный) — 256 Гбайт, теоретический максимум — 64 Тбайт;
- максимальный размер HDD — 16 Тбайт.

45.4. Установка VirtuoZZo

Установка VirtuoZZo производится аналогично установке дистрибутива Fedora инсталлятор Anaconda абсолютно такой же (рис. 45.2).

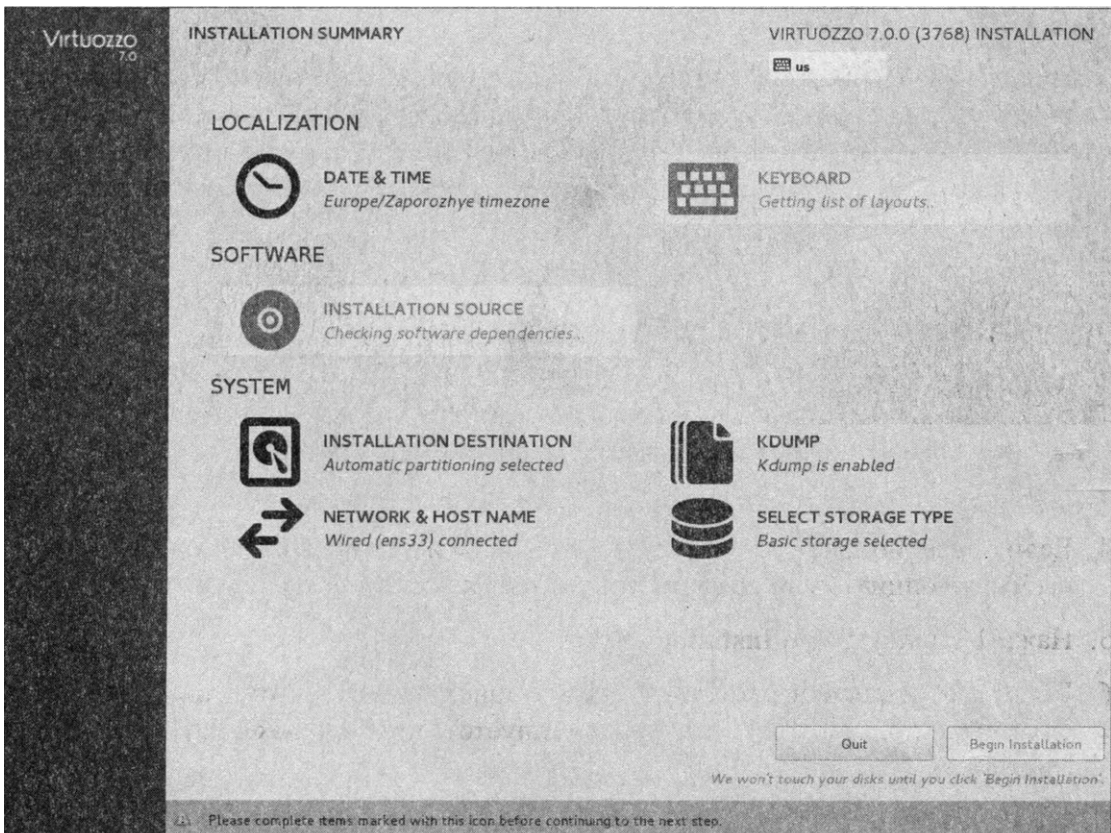


Рис. 45.2. Инсталлятор VirtuoZZo

Для установки Virtuozzo нужно выполнить следующие действия:

1. Загрузиться с инсталляционного диска.
2. Нажать кнопку **Installation destination**.
3. Если производится установка на новый сервер, где нет операционной системы, выберите опцию **Automatically configure a partitioning** и нажмите кнопку **Done** (рис. 45.3).

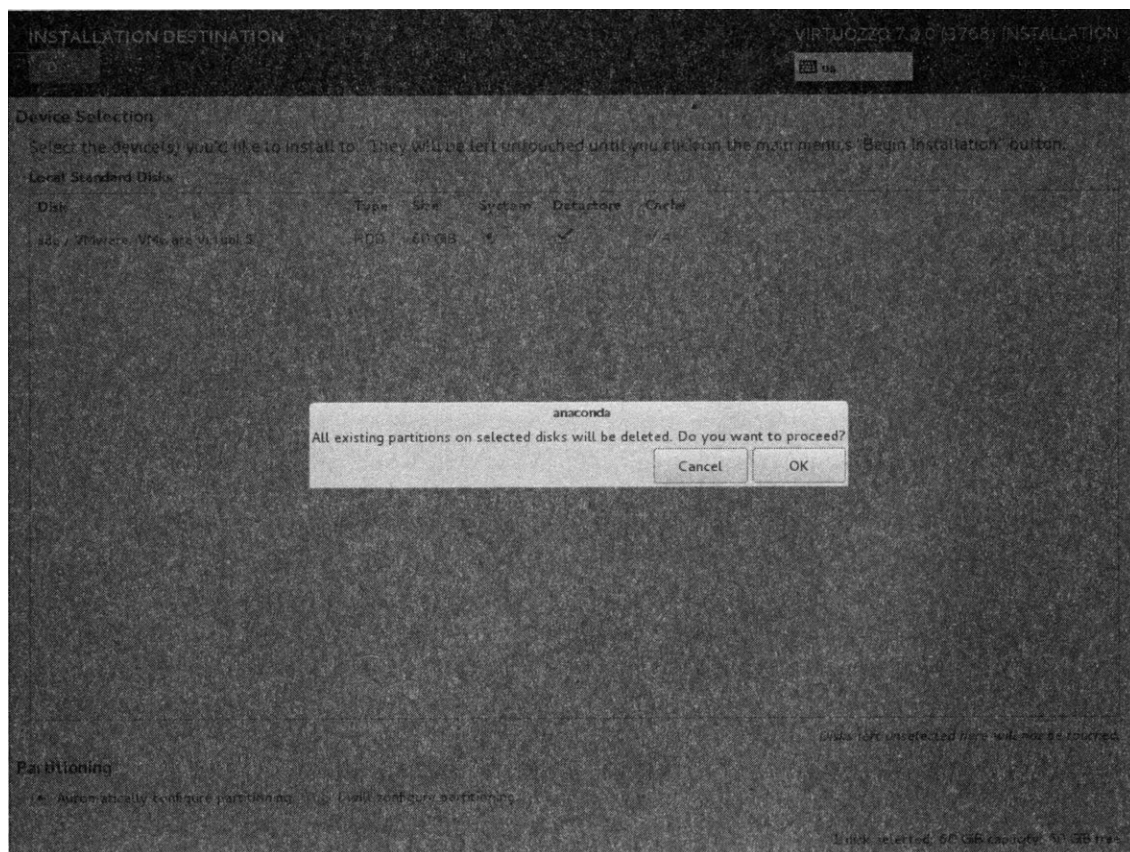


Рис. 45.3. Выбор способа разметки диска

4. Если операционная система уже установлена, и есть желание ее сохранить, тогда выберите опцию **I will configure partitioning** и настройте разделы вручную.
5. Нажать кнопку **Begin Installation** (рис. 45.4).
6. Во время установки системы нужно установить пароль root и создать одного обычного пользователя, что рекомендуется по соображениям безопасности (рис. 45.5).

После перезагрузки появится возможность входа в систему (рис. 45.6).

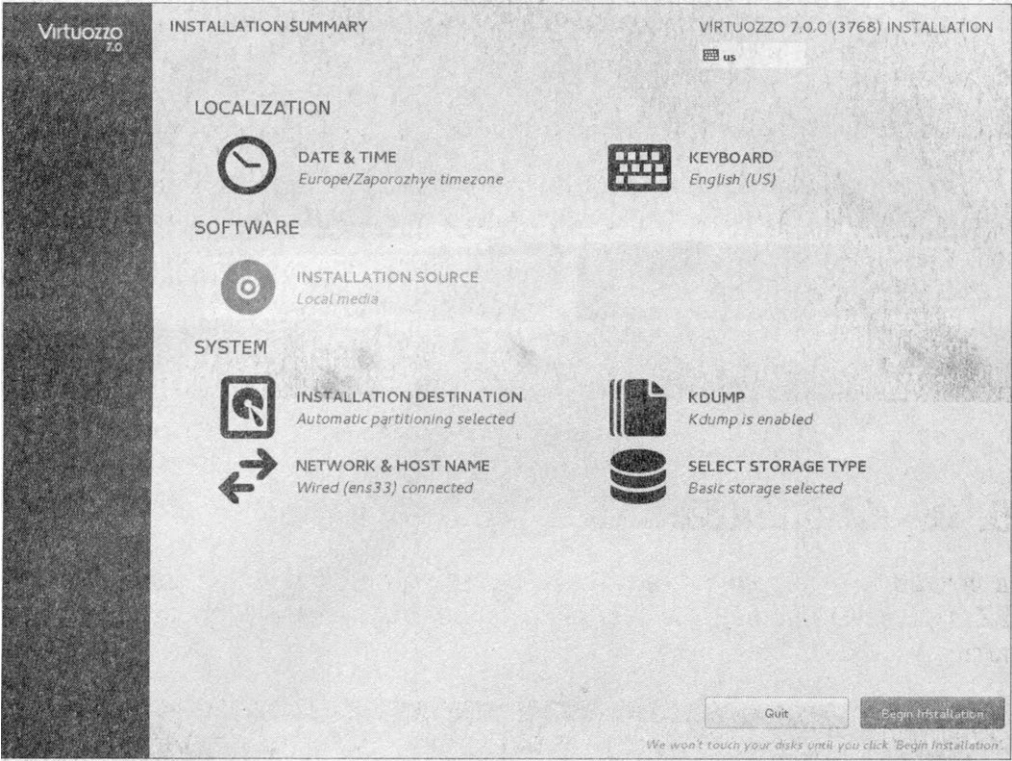


Рис. 45.4. Нажмите кнопку Begin Installation

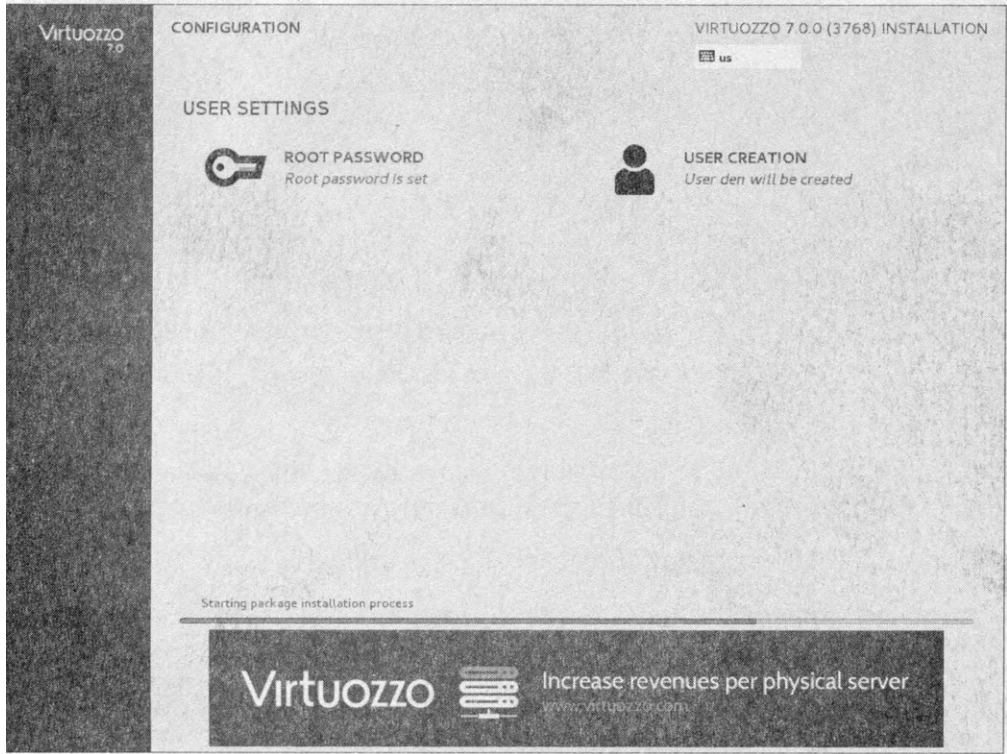


Рис. 45.5. Задание пароля root и создание обычного пользователя

```

Dear          user!

vzkernel: 3.10.0-327.18.2.vz7.15.2

Use the following hostname and IP address to connect to this server:

08:38:06 Thu Sep 15 2016

localhost login:

```

Рис. 45.6. Вход в систему

45.5. Выбор шаблона

Перед созданием контейнера необходимо выбрать шаблон операционной системы (OS EZ Template). Проще говоря, выбрать операционную систему, которая будет работать в контейнере.

```

ubuntu-16.04-x86_64      :ubuntu-16.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64      php      :php for Ubuntu 16.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64      phpmyadmin :phpmyadmin for Ubuntu 16.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64      jre        :jre for Ubuntu 16.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64      cyrus-imap :cyrus-imap for Ubuntu 16.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64      imp        :imp for Ubuntu 16.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64      devel     :devel for Ubuntu 16.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64      spamassassin :spamassassin for Ubuntu 16.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64      mysql      :mysql for Ubuntu 16.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64      phpPgadmin :phpPgadmin for Ubuntu 16.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64      jsdk      :jsdk for Ubuntu 16.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64      mod_perl   :mod_perl for Ubuntu 16.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64      squirrelmail :squirrelmail for Ubuntu 16.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64      mailman    :mailman for Ubuntu 16.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64      webalizer  :webalizer for Ubuntu 16.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64      tomcat     :tomcat for Ubuntu 16.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64      wordpress  :wordpress for Ubuntu 16.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64      proftpd    :proftpd for Ubuntu 16.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64      postgresql :postgresql for Ubuntu 16.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64      :ubuntu-14.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64      php      :php for Ubuntu 14.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64      phpmyadmin :phpmyadmin for Ubuntu 14.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64      jre        :jre for Ubuntu 14.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64      cyrus-imap :cyrus-imap for Ubuntu 14.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64      imp        :imp for Ubuntu 14.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64      devel     :devel for Ubuntu 14.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64      spamassassin :spamassassin for Ubuntu 14.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64      mysql      :mysql for Ubuntu 14.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64      phpPgadmin :phpPgadmin for Ubuntu 14.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64      jsdk      :jsdk for Ubuntu 14.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64      mod_perl   :mod_perl for Ubuntu 14.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64      squirrelmail :squirrelmail for Ubuntu 14.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64      mailman    :mailman for Ubuntu 14.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64      webalizer  :webalizer for Ubuntu 14.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64      tomcat     :tomcat for Ubuntu 14.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64      wordpress  :wordpress for Ubuntu 14.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64      proftpd    :proftpd for Ubuntu 14.04 (for AMD64/Intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64      postgresql :postgresql for Ubuntu 14.04 (for AMD64/Intel EM64T) Virtuozzo Template
centos-7-x86_64          :Centos 7 (for AMD64/Intel EM64T) Virtuozzo Template
centos-7-x86_64          php      :php for Centos 7 (for AMD64/Intel EM64T) Virtuozzo Template
centos-7-x86_64          docker   :docker for Centos 7 (for AMD64/Intel EM64T) Virtuozzo Template
centos-7-x86_64          jre      :jre for Centos 7 (for AMD64/Intel EM64T) Virtuozzo Template
centos-7-x86_64          cyrus-imap :cyrus-imap for Centos 7 (for AMD64/Intel EM64T) Virtuozzo Template
centos-7-x86_64          devel     :devel for Centos 7 (for AMD64/Intel EM64T) Virtuozzo Template
centos-7-x86_64          spamassassin :spamassassin for Centos 7 (for AMD64/Intel EM64T) Virtuozzo Template
centos-7-x86_64          mysql      :mysql for Centos 7 (for AMD64/Intel EM64T) Virtuozzo Template
centos-7-x86_64          jsdk      :jsdk for Centos 7 (for AMD64/Intel EM64T) Virtuozzo Template

```

Рис. 45.7. Список шаблонов

Для просмотра всех шаблонов введите команду:

```
# vzpkg list --with-summary | less
```

Дистрибутивы есть на любой вкус — доступны как RH-совместимые, так и богатый выбор различных дистрибутивов Debian/Ubuntu (рис. 45.7).

Для поиска какого-либо определенного дистрибутива удобнее использовать команду `grep` (рис. 45.8):

```
# vzpkg list --with-summary | grep centos
```



```
root@localhost ~# vzpkg list --with-summary | grep centos
centos-7-x86_64      :centos 7 (for AMD64/Intel EM64T) VirtuoZzo Template
centos-7-x86_64      :php for Centos 7 (for AMD64/Intel EM64T) VirtuoZzo Template
centos-7-x86_64      :docker for Centos 7 (for AMD64/Intel EM64T) VirtuoZzo Template
centos-7-x86_64      :jre for Centos 7 (for AMD64/Intel EM64T) VirtuoZzo Template
centos-7-x86_64      :cyrus-imap for Centos 7 (for AMD64/Intel EM64T) VirtuoZzo Template
centos-7-x86_64      :devel for Centos 7 (for AMD64/Intel EM64T) VirtuoZzo Template
centos-7-x86_64      :spamassassin for Centos 7 (for AMD64/Intel EM64T) VirtuoZzo Template
centos-7-x86_64      :mysql for Centos 7 (for AMD64/Intel EM64T) VirtuoZzo Template
centos-7-x86_64      :jsdk for Centos 7 (for AMD64/Intel EM64T) VirtuoZzo Template
centos-7-x86_64      :mailman for Centos 7 (for AMD64/Intel EM64T) VirtuoZzo Template
centos-7-x86_64      :vzftpd for Centos 7 (for AMD64/Intel EM64T) VirtuoZzo Template
centos-7-x86_64      :mod_ssl for Centos 7 (for AMD64/Intel EM64T) VirtuoZzo Template
centos-7-x86_64      :tomcat for Centos 7 (for AMD64/Intel EM64T) VirtuoZzo Template
centos-7-x86_64      :postgresql for Centos 7 (for AMD64/Intel EM64T) VirtuoZzo Template
centos-6-x86_64      :Centos 6 (for AMD64/Intel EM64T) VirtuoZzo Template
centos-6-x86_64      :php for Centos 6 (for AMD64/Intel EM64T) VirtuoZzo Template
centos-6-x86_64      :jre for Centos 6 (for AMD64/Intel EM64T) VirtuoZzo Template
centos-6-x86_64      :cyrus-imap for Centos 6 (for AMD64/Intel EM64T) VirtuoZzo Template
centos-6-x86_64      :devel for Centos 6 (for AMD64/Intel EM64T) VirtuoZzo Template
centos-6-x86_64      :spamassassin for Centos 6 (for AMD64/Intel EM64T) VirtuoZzo Template
centos-6-x86_64      :mysql for Centos 6 (for AMD64/Intel EM64T) VirtuoZzo Template
centos-6-x86_64      :jsdk for Centos 6 (for AMD64/Intel EM64T) VirtuoZzo Template
centos-6-x86_64      :mod_perl for Centos 6 (for AMD64/Intel EM64T) VirtuoZzo Template
centos-6-x86_64      :mailman for Centos 6 (for AMD64/Intel EM64T) VirtuoZzo Template
centos-6-x86_64      :vzftpd for Centos 6 (for AMD64/Intel EM64T) VirtuoZzo Template
centos-6-x86_64      :mod_ssl for Centos 6 (for AMD64/Intel EM64T) VirtuoZzo Template
centos-6-x86_64      :webalizer for Centos 6 (for AMD64/Intel EM64T) VirtuoZzo Template
centos-6-x86_64      :tomcat for Centos 6 (for AMD64/Intel EM64T) VirtuoZzo Template
centos-6-x86_64      :postgresql for Centos 6 (for AMD64/Intel EM64T) VirtuoZzo Template
root@localhost ~#
```

Рис. 45.8. Отфильтровываем шаблоны

45.6. Создание и настройка контейнера

Создать контейнер на базе шаблона по умолчанию позволяет команда:

```
# prlctl create MyCT --vmtype ct
```

Шаблон по умолчанию указывается в файле `/etc/vz/vz.conf`. Кстати, по умолчанию используется шаблон `centos-7`.

Создать контейнер на базе определенного шаблона можно так (рис. 45.9):

```
# prlctl create MyCT --vmtype ct --ostemplate centos-6-x86_64
```

Все содержимое контейнеров хранится в их приватной области. Чтобы выяснить, где она находится, используется команда `prlctl list`:

```
# prlctl list MyCT -i | grep "Home"
```

Home: `/vz/private/26bc47f6-353f-454b-bc35-b634a88dbbcc`


```

[root@localhost ~]# prctl create MyCT --vmttype ct --ostemplate centos-6-x86_64
Creating the Virtuozzo Container...
Creating cache
Processing metadata for centos-6-x86_64
Creating temporary Container
Creating virtual disk
Running the script pre-cache
Package manager: installing
Running the script post-cache
Running the script post-install
Resizing virtual disk
Packing cache
The Container has been successfully created.
[root@localhost ~]#

```

Рис. 45.9. Процесс создания контейнера

При желании эту область можно перенести на другой жесткий диск— более быстрый или туда, где есть больше свободного пространства.

45.7. Управление ресурсами контейнера

После создания контейнера его конфигурация хранится в файле `/etc/vz/conf/<ID контейнера>.conf`. По умолчанию создается контейнер с 64 Мбайт оперативной памяти, 10 Гбайт дискового пространства, 1000 единиц CPU. Пример конфигурационного файла приведен на рис. 45.10.

```

/etc/vz/conf/6ba9cd71-84fd-4e5e-b5d1-cdd73dfa0ea3.conf
PHYSPPAGES="131072:131072"
SWAPPAGES="131072:131072"
DISKSPACE="10485760:10485760"
DISKINODES="2621440:2621440"
CPUUNITS="1000"
NETFILTER="stateless"
ONBOOT="yes"
AUTOCOMPACT="yes"
RATE="*:1:8"
RATEBOUND="no"
VE_ROOT="/vz/root/$VEID"
VE_PRIVATE="/vz/private/$VEID"
OSTEMPLATE=".centos-6-x86_64"
NAME="MyCT"
TECHNOLOGIES="x86_64 nptl"
DISTRIBUTION="redhat-el6"
OSRELEASE="2.6.32"
VEID="6ba9cd71-84fd-4e5e-b5d1-cdd73dfa0ea3"
UUID="6ba9cd71-84fd-4e5e-b5d1-cdd73dfa0ea3"

```

Рис. 45.10. Конфигурационный файл контейнера

Очень важным является параметр **ONBOOT** — если он включен (значение "yes"), то контейнер будет загружаться при запуске сервера виртуализации.

Единственное, к чему придется привыкнуть, — это неудобные идентификаторы контейнеров. Вывести список доступных контейнеров можно командой:

```
# prlctl list -a
```

Поле **STATUS** в этом списке показывает состояние контейнера или виртуальной машины, поле **IP-ADDR** — IP-адрес контейнера, **T** — это тип объекта, здесь может стоять или **CT** (контейнер), или **VM** (виртуальная машина), **NAME** — это имя контейнера/машины, заданное при создании (в нашем случае **MyCT**). Конечно же, поле **UUID** содержит уникальный идентификатор контейнера/машины.

Рассмотрим несколько примеров управления ресурсами контейнера (подробная информация на этот счет имеется в мануале).

Начнем с изменения производительности процессора. По умолчанию задается 1000 процессорных единиц (CPU Units). При желании можно повысить производительность процессора и задать больше процессорных единиц:

```
# prlctl set MyCT ---cpuunits 2000
```

Процессорные единицы — несколько абстрактное понятие, но Virtuozzo позволяет задавать и конкретные значения. Например, в следующем примере контейнер не может расходовать более 25% от физического процессорного времени:

```
# prlctl set MyCT --cpulimit 25
```

Можно задать частоту процессора контейнера (здесь — 750 МГц):

```
# prlctl set MyCT --cpulimit 750m
```

Или ограничить количество ядер:

```
# prlctl set MyCT --cpus 1
```

Теперь о памяти. Задать размер ОЗУ и свопа можно так:

```
# prlctl set MyCT --memsize 1G --swappages 512M
```

Можно также отредактировать файл конфигурации контейнера (разумеется, при остановленном контейнере):

```
PHYSPAGES="65536:65536"  
SWAPPAGES="65536"
```

Изменить размер виртуального диска позволяет команда **pri disk tool**:

```
prl_disk_tool resize --hdd /vz/vmprivate/d35d28e5-11f7-4b3f-9065-8fef6178bc5b/ \\  
harddisk.hdd --size 80G
```

Перед изменением размера диска следует остановить контейнер/виртуальную машину, а также удалить любые снапшоты, если они были созданы.

Параметры сети задаются так:

```
# prlctl set MyCT --hostname myct.example.com  
# prlctl set MyCT --ipadd 192.168.52.101
```

```
lroot@localhost ~]# prctl set MyCT --cpus 1
set cpus(2): 1

The CT has been successfully configured.
lroot@localhost ~]# prctl set MyCT --memsize 256M
Set the memsize parameter to 256Mb.

The CT has been successfully configured.
lroot@localhost ~]# prctl set MyCT --hostname myct.example.com

The CT has been successfully configured.
lroot@localhost ~]# prctl set MyCT --ipadd 192.168.52.101
Enable automatic reconfiguration for this network adapter.
Configure venet0 (+) type='routed' ips='192.168.52.101/255.255.255.0 '

Configured venet0 (+) type='routed' ips='192.168.52.101/255.255.255.0 '

The CT has been successfully configured.
lroot@localhost ~]# _
```

Рис. 45.11. Конфигурирование контейнера

Первая команда здесь задает имя узла, вторая — его IP-адрес. Процесс настройки контейнера показан на рис. 45.11.

45.8. Управление контейнерами

Что ж, после настройки контейнера самое время его запустить. Для этого используется команда:

```
# prctl start MyCT
```

Запустив контейнер, сразу вводим команду просмотра состояния `prctl list -a` и видим, что наш контейнер запущен (статус **running**) и ему присвоен IP-адрес **192.168.52.101**. Попробуем его «пропинговать». Результат всех этих действий приведен на рис. 45.12. Как видите, контейнер полностью функционирует — он запущен, и к нему идет ping.

Для остановки и перезапуска контейнера служат команды `stop` и `restart` соответственно:

```
# prctl stop MyCT
# prctl restart MyCT
```

Для удаления контейнера его нужно сначала остановить, а потом удалить:

```
# prctl stop MyCT
# prctl delete MyCT
```



```

[root@localhost ~]# prctl start MyCT
Starting the CT...
The CT has been successfully started.
[root@localhost ~]# prctl list -a

```

UUID	STATUS	IP_ADDR	T_NAME
{6ba9cd71-84fd-4e5e-b5d1-cdd73dfa8ea3}	running	192.168.52.101	CT MyCT

```

[root@localhost ~]# ping 192.168.52.101
PING 192.168.52.101 (192.168.52.101) 56(84) bytes of data:
64 bytes from 192.168.52.101: icmp_seq=1 ttl=64 time=0.067 ms
64 bytes from 192.168.52.101: icmp_seq=2 ttl=64 time=0.066 ms
64 bytes from 192.168.52.101: icmp_seq=3 ttl=64 time=0.351 ms
64 bytes from 192.168.52.101: icmp_seq=4 ttl=64 time=0.064 ms
^C
--- 192.168.52.101 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3034ms
rtt min/avg/max/mdev = 0.064/0.137/0.351/0.123 ms
[root@localhost ~]# _

```

Рис. 45.12. Контейнер запущен

45.9. Запуск команд и вход в гостевую операционную систему

Для выполнения произвольных команд используется команда **exec**. Первым делом изменим пароль root (рис. 45.13):

```
# prctl exec MyCT passwd
```

```

[root@localhost ~]# prctl exec MyCT passwd
New password:
Retype new password:
Changing password for user root.
passwd: all authentication tokens updated successfully.
[root@localhost ~]# _

```

Рис. 45.13. Изменение пароля root для гостевой ОС

Теперь попробуем подключиться к гостевой ОС по ssh:

```
# ssh 192.168.52.101
```

Служба sshd на гостевой ОС уже запущена, что упрощает управление гостевой ОС. Вообще, можно вводить команды и через **exec**, но по ssh, думаю, будет удобнее. На рис. 45.14 показаны подключение к гостевой ОС, разметка диска контейнера, а также использование памяти.

```

root@localhost ~# ssh 192.168.52.101
The authenticity of host '192.168.52.101 (192.168.52.101)' can't be established.
RSA key fingerprint is ec:1a:8b:bf:d4:cb:03:ff:a8:d4:c4:91:89:ce:f7:8f.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.52.101' (RSA) to the list of known hosts.
root@192.168.52.101's password:
root@mjct ~# df -H
Filesystem      Size  Used Avail Use% Mounted on
rootfs          11G   715M   9.2G   8% /
/dev/loop37668p1 11G   715M   9.2G   8% /
none            2.0G     0   2.0G   0% /sys/fs/cgroup
none            2.0G   8.2k   2.0G   1% /dev
none            135M     0   135M   0% /dev/shm
root@mjct ~# free
             total        used        free      shared  buffers     cached
Mem:       262144         42664       219480           40           0       26784
-/+ buffers/cache:      15888       246264
Swap:      262144           0       262144
root@mjct ~# _

```

Рис. 45.14. Подключение к контейнеру по ssh

После установки ssh-подключения можно вводить команды без префикса `prictl exec`, что гораздо удобнее.

45.10. Настройка сети

Прежде, чем приступить к настройке серверов, нужно настроить сеть, иначе в виртуальном сервере менеджер пакетов работать не будет, и программное обеспечение установить не получится.

На данный момент мы установили только доменные имена и IP-адреса виртуальных узлов. Но этого мало. Нужно настроить NAT, разрешить доступ к виртуальным серверам извне и настроить DNS.

Начнем с настройки NAT. В Virtuozzo Linux пакет `iptables-services` установлен по умолчанию, а IPv4-форвардинг включен (в файле `cat/proc/sys/net/ipv4/ip_forward` имеется соответствующая единица), поэтому никаких подготовительных действий не нужно, и сразу можно приступить к настройке правил `iptables`.

Для NAT определенного контейнера используется команда:

```
# iptables -t nat -A POSTROUTING -s src_net -o if -j SNAT -to ip_address
```

Здесь вместо **src net** нужно указать IP-адрес подсети контейнера, вместо `if` — интерфейс, а вместо `ip_address` — внешний IP-адрес вашего физического узла. Но можно сделать проще и ввести команду:

```
# iptables -t nat -A POSTROUTING -o ens33 -j MASQUERADE
```

ОТСУТСТВИЕ ТАБЛИЦЫ NAT

Если система выведет сообщение об отсутствии таблицы NAT, не обращайтесь внимания. Видимо, существует некий конфликт между версией `iptables` и ядра... Разбираться с этим я не стал, но самое интересное, что правила работают ☺.

```

Package iptables-services-1.4.21-16.el7.x86_64 already installed and latest version
Nothing to do.
root@localhost ~# cat /proc/sys/net/ipv4/ip_forward
1
root@localhost ~# prctl start first
Starting the CT...
The CT has been successfully started.
root@localhost ~# iptables -t nat -A POSTROUTING -o ens33 -j MASQUERADE
iptables v1.4.21: can't initialize iptables table 'nat': Table does not exist (do you need to insmod?)
Perhaps iptables or your kernel needs to be upgraded.
root@localhost ~# ssh 192.168.52.101
root@192.168.52.101's password:
Last login: Sun Oct 23 12:35:13 2016 from 192.168.52.212
root@first ~# ping mail.ru
ping: unknown host mail.ru
root@first ~# route
Kernel IP routing table
Destination      Gateway         Genmask         Flags Metric Ref    Use Iface
default          *              0.0.0.0         U         0      0        0 venet0
link-local      *              255.255.0.0     U         1002    0        0 venet0
192.168.52.0    *              255.255.255.0   U         0       0        0 venet0
root@first ~# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data:
64 bytes from 8.8.8.8: icmp_seq=1 ttl=127 time=274 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=127 time=52.7 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=127 time=54.3 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=127 time=52.4 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=127 time=52.5 ms
64 bytes from 8.8.8.8: icmp_seq=6 ttl=127 time=52.4 ms
64 bytes from 8.8.8.8: icmp_seq=7 ttl=127 time=52.4 ms
64 bytes from 8.8.8.8: icmp_seq=8 ttl=127 time=52.4 ms
^C
--- 8.8.8.8 ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 753ms
rtt min/avg/max/mdev = 52.442/88.497/274.465/73.316 ms
root@first ~# exit
logout
Connection to 192.168.52.101 closed.
root@localhost ~# prctl stop first
Stopping the CT...
The CT has been successfully stopped.
root@localhost ~# prctl set first --nameserver 8.8.8.8
The CT has been successfully configured.
root@localhost ~# prctl start first
Starting the CT...
The CT has been successfully started.
root@localhost ~#

```

Рис. 45.15. Включение NAT и установка сервера DNS для первого сервера

В этом случае все IP-адреса будут транслироваться SNAT. Именно эту команду я и ввел на рис. 45.15.

На радостях запускаем контейнер и ... обнаруживаем, что пропинговать-то узел по IP мы можем, а вот DNS настроить забыли. Ничего, все это решается командой:

```
# prctl set <контейнер> --nameserver 8.8.8.8
```

Пока мы используем OpenDNS, ведь свой мы еще не настроили. Теперь нужно убедиться, что мы можем пропинговать узел по его имени (рис. 45.16):

```
# ping mail.ru
```

Вот теперь все в порядке, и можно приступить к установке пакетов. Только не забудьте настроить доступ к виртуальным серверам извне, иначе никто не сможет до них достучаться:

```
# iptables -t nat -A PREROUTING -p tcp -d ip_address -dport port_num \
-i ens33 -j DNAT --to-destination ve_address :dst_port_num
```

Здесь `ve_address` — это IP-адрес контейнера, `dst_port` — TCP-порт, `ip_address` — внешний (публичный) IP-адрес вашего узла, а `port_num` — TCP-порт аппаратного узла, который будет использоваться для интернет-соединений к приватным контейнерам.

Обратите внимание, что это правило сделает сервис, который раньше «висел» на порту с заданным номером (`port_num`), недоступным. Также нужно учитывать, что трансляция SNAT, которую мы делали раньше, тоже необходима.

```
[root@localhost ~]# prctl set first --nameserver 8.8.8.8
The CT has been successfully configured.
[root@localhost ~]# prctl start first
Starting the CT...
The CT has been successfully started.
[root@localhost ~]# ssh 192.168.52.101
root@192.168.52.101's password:
Last login: Sun Oct 23 12:46:00 2016 from 192.168.52.212
[root@first ~]# ping mail.ru
PING mail.ru (217.69.139.199) 56(84) bytes of data:
64 bytes from ms.mail.ru (217.69.139.199): icmp_seq=1 ttl=127 time=35.3 ms
64 bytes from ms.mail.ru (217.69.139.199): icmp_seq=2 ttl=127 time=35.5 ms
^C
--- mail.ru ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1577ms
rtt min/avg/max/mdev = 35.365/35.475/35.586/0.218 ms
[root@first ~]#
```

Рис. 45.16. Сеть на виртуальном сервере поднята и работает

Если нужно, чтобы порт 80 был доступен на физическом узле, а доступ к виртуальным серверам осуществлялся через порт 8080, используйте такие правила:

```
# iptables -t nat -A PREROUTING -p tcp -d ip_address --dport 8080 \
-i ethO -j DNAT --to-destination ve_address:80
# iptables -t nat -A POSTROUTING -s ve_address -o ethO -j SNAT --to ip_address
```

Тогда «достучаться» до виртуальных серверов можно будет так:

`http://ip_address:8080/`.

Осталось только сохранить правила iptables:

```
# service iptables save
# service iptables restart
```

Теперь у наших виртуальных серверов есть доступ к Интернету, они могут пропинговать друг друга, и к ним можно обратиться извне. По сути, они мало чем отличаются от обычных интернет-серверов.

Вот, собственно, и все. Виртуальный сервер создан и работает, далее, используя `ssh`, можно приступить к установке программного обеспечения и к его настройке. Дополнительная информация по настройке и управлению контейнерами Virtuozzo имеется в официальном руководстве: <http://docs.virtuozzo.com/master/index.html>.

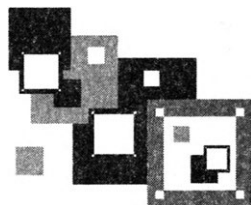
45.11. Делаем работу с Virtuozzo удобнее

Virtuozzo Linux— это обычный дистрибутив, а не какая-то урезанная его версия. Дистрибутив является RH-совместимым, что позволяет устанавливать RPM-пакеты. К счастью, вручную устанавливать ничего не придется, т. к. Virtuozzo Linux содержит довольно богатые репозитории, из которых вы можете установить свои любимые программы. Я, например, установил tc:

```
# yum install tc
```

На этом все. Дополнительную информацию, в том числе и о совместимости с родственной OpenVZ, можно найти в блоге разработчика по адресу: <https://habrahabr.ru/company/virtuozzo/>.

ГЛАВА 46



Настройка собственного VPN-сервера

46.1. Что мы будем настраивать?

Здесь не будет длинного и скучного введения о том, что такое VPN и зачем она нужна (см. об этом в *главе 10*). Думаю, большинство IT-специалистов на этот счет в курсе.

Именно поэтому мы сразу приступим к делу. Настройка VPN-сервера будет производиться на примере Ubuntu 16.04 TLS, но в других дистрибутивах все настраивается аналогично, — будут отличаться разве что команды установки пакетов и, возможно, пути к некоторым файлам конфигурации.

Прежде, чем приступать к настройке, нужно понимать, что именно мы будем настраивать. Виртуальную частную сеть (VPN) можно организовать по-разному, соответственно, настройка VPN-сервера тоже будет различна, — в зависимости от преследуемой цели. Можно, например, объединить несколько филиалов в одну большую VPN. Тогда в каждом филиале будет собственный VPN-сервер. А можно настроить сервер VPN так, чтобы он предоставлял удаленным пользователям — например, тем, кто уехал в командировку, или тем, кто предпочитает работать дома, — безопасный доступ к ресурсам внутренней (корпоративной) сети. Именно такой вид VPN-сервиса мы и создадим, но с одним отличием — вместо корпоративной сети он будет предоставлять доступ к Интернету.

Что нам это даст? Если вам (или пользователям вашей организации) часто приходится путешествовать (не важно, отпуск это или командировка), то у вас постоянно возникает необходимость подключаться к незащищенным сетям Wi-Fi отелей и других публичных мест. Чтобы не допустить утечки данных (в том числе и передаваемых паролей), и создается VPN-сервер. Тогда весь ваш трафик в зашифрованном виде будет проходить в Интернет через вашу VPN. Следовательно, на отрезке от вашего устройства до VPN-сервера никто этот трафик перехватить не сможет. А это означает, что злоумышленник, «работающий» в сети отеля, даже если и перехватит ваш трафик, толку от него он получит немного — ведь трафик будет зашифрован.

На данный момент мы предполагаем, что вы уже выполнили начальную настройку компьютера, работающего под управлением Ubuntu: настроили интернет-соединение, правила брандмауэра и т. д.

46.2. Установка OpenVPN

Прежде всего нужно установить пакет OpenVPN. Обычно он доступен в стандартных репозиториях дистрибутива, так что нет необходимости подключать сторонние репозитории.

Введите команды:

```
sudo apt update
sudo apt install openvpn easy-rsa
```

Первая команда обновляет информацию о пакетах, вторая — устанавливает два пакета. Первый — это сам OpenVPN, а второй (*easy-rsa*) — пакет, позволяющий построить собственный сервер сертификации.

46.3. Настройка центра сертификации

Open VPN использует TLS/SSL, поэтому нам понадобятся сертификаты для шифрования трафика между сервером и клиентом. Чтобы не покупать сертификаты, мы создадим собственный центр сертификации.

Скопируем шаблонный каталог *easy-rsa* в наш домашний каталог с помощью команды *make-cadir*:

```
make-cadir ~/openvpn-ca
cd ~/openvpn-ca
```

Откройте из этого каталога файл *vars*:

```
mcedit vars
```

Вместо редактора *mcedit* можете использовать тот, который вам больше нравится. В конце файла будут описаны переменные, используемые при создании сертификатов. Выглядеть они должны так:

```
export KEY_COUNTRY="US "
export KEY_PROVINCE="CA"
export KEY_CITY="SanFrancisco"
export KEY_ORG="Fort-Funston"
export KEY_EMAIL="me@myhost.mydomain"
export KEY_OU="MyOrganizationalUnit"
```

Установите для них свои значения, например:

```
export KEY_COUNTRY="US"
export KEY_PROVINCE="NY"
export KEY_CITY="Chicago"
export KEY_ORG="My Company"
export KEY_EMAIL="admin@company.com"
export KEY_OU="MyWorkgroup"
```

Также найдите и отредактируйте переменную *KEY_NAME*:

```
export KEY_NAME="server"
```

Для простоты можно задать название "server" или любую другую строку (но запомните, какую именно). Если вы зададите название, отличное от "server", вам придется изменить некоторые команды, в которых встречается это название.

Создав и отредактировав конфигурационные файлы, можно приступить к созданию центра сертификации:

```
cd ~/openvpn-ca
source vars
```

Вывод будет примерно таким:

NOTE: If you run ./clean-all, I will be doing a rm -rf on /home/den/openvpn-ca/keys

После этого введите команды:

```
./clean-all
./build-ca
```

Первая команда выполнит очистку имеющихся ключей, а вторая начнет процесс создания ключа и сертификата корневого центра сертификации. Поскольку все значения уже указаны в файле vars, вам нужно будет только нажимать клавишу <Enter> для подтверждения выбора.

Теперь у нас есть собственный центр сертификации, который мы будем использовать при создании сертификата, ключа и файлов шифрования для сервера.

46.4. Создание сертификата и ключей для сервера

Для создания ключей для сервера введите команду:

```
./build-key-server server
```

Процесс создания ключей очень прост — нажимайте клавишу <Enter> в ответ на предлагаемые значения. Значение challenge password задавать не нужно. В конце процесса надо два раза ввести y — для подписи и для подтверждения создания сертификата:

Certificate is to be certified until Jul 13 12:00:16 2027 GMT (3650 days)

Sign the certificate? [y/n]:y

1 out of 1 certificate requests certified, commit? [y/n]y

Write out database with 1 new entries

Data Base Updated

Осталось досоздать остальные файлы:

```
./build-dh
openvpn --genkey --secret keys/ta.key
```

Первая команда создает ключи протокола Диффи-Хеллмана, вторая — генерирует подпись HMAC. В зависимости от расторопности вашего сервера, эти команды могут работать несколько минут каждая.

46.5. Создание сертификата и ключей для клиента

Следующий шаг— это создание сертификата и пары ключей для клиента. Это можно сделать и на клиенте, а затем подписать полученный ключ центром сертификации сервера, но для простоты мы все будем делать на сервере.

Мы создадим ключ и сертификат только для одного клиента. Если клиентов несколько, вы можете повторять этот процесс до бесконечности, — пока не сгенерируете сертификаты и ключи для каждого клиента.

Команда `build-key` служит для создания файлов без пароля для облегчения автоматических соединений:

```
cd ~/openvpn-ca
source vars
./build-key client1
```

Если нужны файлы, защищенные паролем, используйте команду `build-key-pass`:

```
cd ~/openvpn-ca
source vars
./build-key-pass client1
```

46.6. Настройка сервера OpenVPN

Вот только теперь можно приступить к процессу настройки сервера OpenVPN. В самом начале процесса нужно скопировать сгенерированные ранее файлы из каталога `openvpn-ca/keys` в каталог `/etc/openvpn`:

```
cd ~/openvpn-ca/keys
sudo cp ca.crt ca.key server.crt server.key ta.key dh2048.pem /etc/openvpn
```

Пример файла конфигурации можно взять из файла `/usr/share/doc/openvpn/examples/sample-config-files/server.conf.gz`. Его нужно распаковать в файл `/etc/openvpn/server.conf`.

Распаковав шаблон файла конфигурации, можно приступить к его редактированию. Откройте файл `/etc/openvpn/server.conf` в любимом текстовом редакторе.

Далее приведен фрагмент этого файла. Внимательно читайте комментарии:

```
# Раскомментируйте эту строку
tls-auth ta.key 0 # This file is secret
# Установите key-direction в 0
key-direction 0
# Раскомментируйте эту строку
cipher AES-128-CBC
# Сразу после строки с cipher добавьте следующую строку:
auth SHA256
# Укажите имя пользователя и группы, от имени которых будет запускаться сервер%
user nobody
group nogroup
```

```
# Чтобы VPN-соединение использовалось для всего трафика,  
# нужно "протолкнуть" настройки DNS на машины клиентов.  
# Для этого раскомментируйте следующую строку:  
push "redirect-gateway defl bypass-dhcp"  
# Также добавьте DNS-серверы (используем OpenDNS):  
push "dhcp-option DNS 208.67.222.222"  
push "dhcp-option DNS 208.67.220.220"  
# При необходимости измените порт и протокол:  
port 443  
proto tcp  
# Если при вызове build-key-server вы указали значение, отличное от  
# "server", измените имена файлов сертификата и ключа  
cert server.crt  
key server.key
```

Теперь нужно немного настроить сам сервер, и первым делом разрешить пересылать трафик, если вы этого еще не сделали. Откройте файл `sysctl.conf`:

```
sudo mcedit /etc/sysctl.conf
```

Раскомментируйте строчку:

```
net.ipv4.ip_forward=1
```

Чтобы изменения вступили в силу, введите команду:

```
sudo sysctl -p
```

Осталось только настроить брандмауэр, и можно запускать VPN-сервер. Будем считать, что используется брандмауэр UFW (в современных дистрибутивах он заменил iptables). Вы должны знать имя публичного интерфейса — пусть это будет `ens33` (это для примера, в вашем случае имя публичного интерфейса будет отличаться). Выяснить его можно командой:

```
ip route | grep default
```

Полученное имя нужно добавить в файл `/etc/ufw/before.rules`. В самое начало этого файла нужно добавить строки (также укажите IP-адрес и маску вашей подсети):

```
# START OPENVPN RULES  
# NAT table rules  
*nat  
:POSTROUTING ACCEPT [0:0]  
# Allow traffic from OpenVPN client to eth0  
-A POSTROUTING -s 192.168.0.0/24 -o ens33 -j MASQUERADE  
COMMIT  
# END OPENVPN RULES
```

Вместо `ens33` нужно указать имя вашего публичного интерфейса. Теперь откройте файл `/etc/default/ufw`:

```
nano /etc/default/ufw
```

и найдите в нем директиву `default _forward _policy` :

```
DEFAULT_FORWARD_POLICY="ACCEPT"
```

Откроем порт для OpenVPN:

```
sudo ufw allow 443/tcp
```

или

```
sudo ufw allow 1194/udp
```

Первую команду нужно вводить, если вы применяете протокол TCP, вторую, если используется протокол UDP. Чтобы изменения вступили в силу, брандмауэр нужно перезапустить:

```
sudo ufw disable
```

```
sudo ufw enable
```

Все готово для запуска VPN-сервера. Запустим его командой:

```
sudo systemctl start openvpn@server
```

Проверить состояние сервера можно так:

```
sudo systemctl status openvpn@server
```

Вы должны увидеть что-то вроде этого:

```
openvpn@server.service - OpenVPN connection to server
   Loaded: loaded (/lib/systemd/system/openvpn@.service; disabled; vendor
   preset: enabled)
   Active: active (running) since Tue 2017-07-11 16:59:05 EDT; 25s ago
```

Если все нормально, тогда обеспечим автоматический запуск сервера:

```
sudo systemctl enable openvpn@server
```

46.7. Инфраструктура настройки клиентов

Прежде, чем клиенты смогут подключиться к нашему VPN-серверу, нужно позаботиться об инфраструктуре настройки клиентов. Создадим каталог для хранения файлов:

```
mkdir -p ~/clients/files
chmod 700 ~/clients/files
```

Такие права доступа нужны, поскольку этот каталог будет содержать ключи клиентов.

Далее установим базовую конфигурацию:

```
cd /usr/share/doc/openvpn/examples/sample-config-files/
cp client.conf ~/clients/base.conf
```

Откройте файл `~/clients/base.conf`. В нем нужно сделать несколько изменений:

```
# Укажите IP-адрес сервера и порт (1193 для UDP или 443 для TCP)
remote IP-адрес порт
```

```
# Укажите протокол udp или tcp
proto протокол
# Раскомментируйте директивы
user nobody
group nogroup
# Найдите директивы ca, cert и key. Закомментируйте их
#ca ca.crt
#cert client.crt
#key client.key
# Добавьте параметры cipher и auth так, как они описаны в server.conf
cipher AES-128-CBC
auth SHA256
# Установите key-direction в 1
key-direction 1
```

Теперь создадим сценарий генерации файлов конфигурации (листинг 46.1):

```
cd ~/clients
touch make_config
chmod +x make_config
mcedit make_config
```

Листинг 46.1. Файл `make_config`

```
#!/bin/bash

# First argument: Client identifier

KEY_DIR=~/.openvpn-ca/keys
OUTPUT_DIR=~/.clients/files
BASE_CONFIG=~/.clients/base.conf

cat ${BASE_CONFIG} \
  <(echo -e '<ca>' ) \
  ${KEY_DIR}/ca.crt \
  <(echo -e '</ca>\n<cert>' ) \
  ${KEY_DIR}/${1}.crt \
  <(echo -e '</cert>\n<key>' ) \
  ${KEY_DIR}/${1}.key \
  <(echo -e '</key>\n<tls-auth>' ) \
  ${KEY_DIR}/ta.key \
  <(echo -e '</tls-auth>' ) \
  > ${OUTPUT_DIR}/${1}.ovpn
```

Используя этот сценарий, вы сможете легко генерировать файлы конфигурации клиентов:

```
cd ^/clients
./make_config user1
```

Если все прошло успешно, то в каталоге `~/clients/files` вы найдете файл `user1.ovpn`.

46.8. Настройка клиентов

Теперь нужно передать файлы конфигурации клиентам. Как вы это сделаете — значения не имеет. Желательно, конечно, передавать их по безопасному каналу связи — например, по электронной почте с шифрованием или через sFTP.

В Windows полученный `ovpn`-файл нужно поместить в каталог `C:\Program Files\OpenVPN\config`, предварительно установив клиент OpenVPN для Windows. Загрузить эту программу можно с официальной страницы проекта: <https://openvpn.net/index.php/open-source/downloads.html>.

После запуска OpenVPN он должен автоматически увидеть ваш профиль. Щелкните на значке клиента в панели быстрого запуска правой кнопкой мыши и выберите команду **Подключиться** (рис. 46.1).

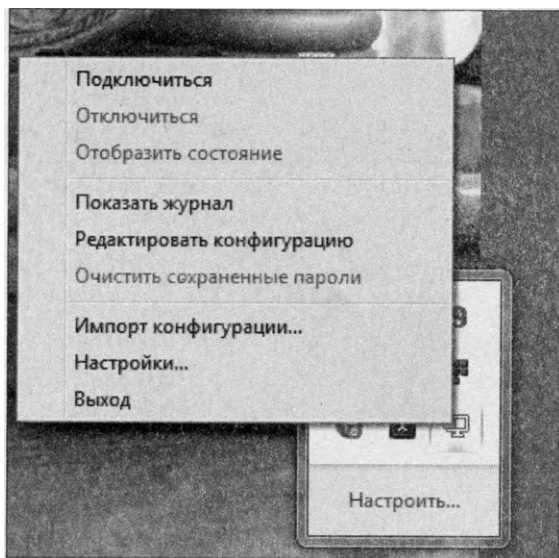


Рис. 46.1. OpenVPN для Windows

В Linux настройка будет отличаться от той, которая была описана в *главе 10*, поскольку все необходимые ключи мы будем хранить в `ovpn`-файле. Первым делом установите пакет `openvpn`:

```
sudo apt-get install openvpn
```

Откройте файл `user1.ovpn`, полученный с сервера. Раскомментируйте следующие строки:

```
script-security 2
up /etc/openvpn/update-resolv-conf
down /etc/openvpn/update-resolv-conf
```

ВНИМАНИЕ!

Если в вашем дистрибутиве нет файла */etc/openvpn/update-resolv-conf*, то делать ничего не нужно!

Если у вас CentOS, то в файле *user1.ovpn* группу с *nogroup* нужно изменить на *nobody*:

```
group nobody
```

Теперь подключитесь к VPN-серверу:

```
sudo openvpn --config user1.ovpn
```

Собственно, на этом настройка сервера и клиентов окончена пользоваться Интернетом стало немного безопаснее!

ПРИЛОЖЕНИЕ

Описание электронного архива

Электронный архив, сопровождающий книгу, можно скачать с FTP-сервера издательства по ссылке **ftp.bhv.ru/9785977539432.zip**, а также со страницы книги на сайте **www.bhv.ru**.

Электронный архив включает две папки:

- в папке **Дополнения** в PDF-файлах с соответствующими названиями приведены следующие материалы:
 - Автоматизация с tcsh (введение в язык TC Shell);
 - Антивирус ClamAV (настройка антивируса в Linux, прозрачная проверка файлов) — глава из 4-го издания;
 - Загрузчик LILO (для старых дистрибутивов) — фрагмент *главы 21* из 2-го издания;
 - Особые операции при работе с файловой системой;
 - Сетевая файловая система (Network File System);
 - Система управления доступом SELinux — *глава 33* из 2-го издания;
 - Электронная подпись в Linux. Утилита GnuPG (использование GPG в Linux);
 - Язык gawk. Обработка текста (введение в язык gawk);
 - Настройка ADSL-доступа к Интернету — материал *главы 9* из 5-го издания.

Процесс установки дистрибутива Slackware 14 показан в файле презентации Slackware14.ppt. Здесь же вы найдете советы по работе с программой fdisk;

- в папке Видео находятся следующие файлы:
 - **передача параметров ядра и вход как root без пароля** — видео показывает, как в современном дистрибутиве (на примере Debian 8) войти под именем пользователя root без пароля и установить новый пароль;
 - **установка Ubuntu 15.10** — простое видео установки Ubuntu на новый компьютер.

Для воспроизведения видеоуроков необходимо установить кодек XVid.

Звукового сопровождения видеоуроков не предусмотрено.

Предметный указатель

3

3DES 494

A

ACL 546
Apache 500
APM 365

B

bash 128
BIOS 385
BlowFish 145, 494
BogoMIPS 354

C

CentOS 477
Ceph 579, 581
Clonezilla 584, 585
cyrus-pop3d 554

D

DEB-пакеты 160
Destination NAT 484
DHCP 53
DirectX 337
DNS 530
DNS-сервер
 ♦ вторичный 541
 ♦ первичный 536
DoS 462
DOS 90
DSL-маршрутизаторы с функциями Wi-Fi 216

E

eCryptfs 595
EDD58
EFI126
Ext2Fsd 85

F

Fast Ethernet 193
Firefox 311

firewall 482
Flash 311
FreeOpenVPN 224
FTP 520
FTP-клиент 520
 ♦ FileZilla 319
FTP-сервер 520
 ♦ ProFTPD 520
 ♦ wu-ftp 520

G

Gigabit Ethernet 193
GIMP 278
GlusterFS 579
GPT 126
GrSecurity 458
GRUB 357
GRUB2 383

H

Hetzner 606

I

IDEA 494
IMAP 553
initrd 386
Intel VT-x 641
iptables 485, 549
IPv4 forwarding 481, 486
i-узел 94

K

KDM 457
KVM 603
 ♦ преимущества 605

L

LIDS 458
Linux Live 584, 585, 594
LiveCD 583-585, 591, 593, 594
LiveUSB 584
logrotate 434
Lustre 579
LXDE50

M

MAC 53
Macromedia Flash Player 311
MBR 94,126, 385
MD5 145
MooseFS 579
MTA 553
MTA Exim 554
MTA QMail 554
MTU 212

N

NAS 577
NAT 482, 651
NCSA HTTPd 1.3 500
NetworkManager 217
Norton Ghost 583
ntfs-3g 112
NX-защита 352

O

OpenLDAP 435,436
Open VPN 656, 658
OpenVZ 601, 603, 633-637, 640, 655
◊ недостатки 604
Opera VPN 224
OS EZ Template 645
OSD 582
OSI478

P

P2P 321
PAM 459
Parallels Cloud Storage 579
Photoshop 278
PID 404
POP 553
POST 34
proc 409
PStorage 580
PXE 52

R

RAID 570, 577
◊ недостатки 578
RAID-массив 570, 571, 573-576
RDIMM 602
Red Hat Ceph Storage 582
Remastersys Backup 584
Remote Administrator 136
rpm 164
RPM-пакеты 160,162
RSA 494
rsyslogd 446

S

SecurityKISS 223,224
◊ настройка 225
SELinux 355,458
sendmail 435, 553
SGID 105
Slackware 62
SMP 353
SMTP 553
SMTP-сервер 556
SOA 538
Source NAT 485
speedtest.net
◊ консольная версия 613
SQL-оператор 513
squidGuard 550
SSID 221
SUID 105
Synaptic 179
sysfs 409
syslog-ng 446
SysRq 81

T

Telnet 494
TFTP 54
TLD 530
TLS/SSL 656
TrueCrypt 595

U

Ubuntu
◊ DNS 536
Unity 49, 71
UNIX 420
UUID 114

V

VDS 603
virtual environments 633
Virtual File System 409
VirtualBox 633, 635, 637
Virtuozzo 603, 640
О установка 643
Virtuozzo Storage Cluster 642
VLAN 203
VMware 476, 633, 635, 637
VPN 655
VPN-доступ 223
VPN-сервер 655
VPS 603

W

Web-сервер Apache 552
Wi-Fi 216
Windows-версия GIMP 288

А

«Аварийные» комбинации клавиш 81
Автодополнение 82
Автоматизация выполнения задач 470
Алгоритм шифрования AES 595
Архитектура
◊ Linux 26
◊ файловой системы 93
Атака на отказ 462
Аутентификация SMTP-AUTH 556

Б

Блок 94
Борьба с простыми паролями 465
Брандмауэр 482
◊ iptables 485, 549
◊ nftables 485
Браузер
◊ Chromium 314, 317
◊ Google Chrome 314

В

Виртуализация на уровне ядра 633
Виртуальная машина 323
◊ VirtualBox 324, 574
◊ VMware 323, 574
Виртуальное окружение 633
Виртуальные файловые системы 409
Виртуальный контейнер 633
Включение ИЧ4-переадресации 482
Восстановление загрузчика GRUB/GRUB2 379
Вращение изображения 282
Вторичный загрузчик 385
Выбор
◊ графической среды 49
◊ дистрибутива 21
◊ пакетов для установки 48
◊ языка установки 36

Г

Главная загрузочная запись 94, 385
Гостевая операционная система 323
Графическая
◊ подсистема X.Org 467
◊ среда 61
Графический
◊ интерфейс 259
 □ GNOME 467
 □ KDE 467
 □ X Window 25
◊ менеджер регистрации 61
◊ редактор GIMP 278
◊ режим 62

Д

Двухканальный режим работы памяти 470
Демон
◊ SASL557
◊ saslauthd 556
Демоны протоколирования 440
Дефрагментация 39
Директива
◊ DefaultRoot 525
◊ Directory 507
◊ Files 509
◊ Limit 508
◊ MaxClients 525
◊ ServerName 506
Директивы файла конфигурации
 proftpd.conf 523
Диск
◊ USB-диск 118
◊ виртуальный 386
◊ гибкий 109
Дистрибутив
◊ ALT Linux 29
◊ CentOS 28
◊ Debian Sarge 29
◊ Fedora 27
◊ Mageia 27
◊ Mandrake 27
◊ Mandriva 27,43
◊ openELEC 247
◊ openSUSE 30
◊ Red Hat 27
◊ Slackware 30
0 Ubuntu 29
Дистрибутивы Linux 25
Длинные имена дисков 114
Домашний каталог 98
Домен 536
Дополнительные файлы конфигурации
 PAM 460

Ж

Журналы 85

З

Загрузчик
◊ ASPLoader 371
◊ GRUB 34, 371
◊ GRUB2 34, 357,371
◊ LILO 34, 371, 385
Запись
◊ CD 289
◊ DVD 290

Защита
 ◇ загрузчика паролем 380
 ◇ от «восстановления пароля root» 455
 ◇ от перезагрузки 454, 455
 Зона 536

И

Изменение
 ◇ размера
 □ изображения 280
 □ существующих разделов 45
 ◇ таблицы
 □ маршрутизации 479
 □ разделов 39
 Имена разделов диска в GRUB 373
 Информационные про-файлы 410

К

Кадрирование изображения 283
 Каталог
 ◇ /etc/cron.daily 471
 ◇ /etc/cron.hourly 471
 ◇ /etc/cron.weekly 471
 ◇ /etc/rc.d 389
 ◇ /etc/rc.d/init.d 389
 ◇ /etc/skel 145
 ◇ /etc/zypp/repos.d 186
 ◇ /var/cache/apt/archives 175
 ◇ домашний 97
 ◇ признак каталога 103
 ◇ родительский 97
 ◇ текущий 97
 Квотирование 157
 Классический интерфейс GNOME 64
 Клиент обмена мгновенными сообщениями
 Pidgin 316
 Кодек 241
 Команда
 ◇ /sbin/grub-install 377
 ◇ /sbin/init 389
 ◇ adduser 143
 ◇ alien 177
 ◇ apt 163
 ◇ apt-get 175
 ◇ arch 414
 ◇ at 473
 ◇ atq 473
 ◇ atrm 473
 ◇ cat 95
 ◇ cd 97
 ◇ chmod+xl30
 ◇ chmod 103
 ◇ chown 105

◇ clear 83, 414
 ◇ cshp 418
 ◇ configure 161
 ◇ convert 377
 ◇ cp 95
 ◇ date 414
 ◇ df 426
 ◇ diff 417
 ◇ dmesg351
 ◇ dpkg 163,173
 ◇ echo 415
 ◇ edquota 158
 ◇ egrep418, 419
 ◇ exit 139,415
 ◇ fdisk 123
 ◇ find 106
 ◇ free 426, 467
 ◇ fsck 113
 ◇ ftp 424
 ◇ grep418
 ◇ groupadd 146
 ◇ grub 381
 ◇ grub-mkconfig 377
 ◇ grub-mkpasswd-pbkdf2 384
 ◇ gzip 377
 ◇ halt 63
 ◇ head 419
 ◇ ifconfig 83, 204
 ◇ kdesu 140
 ◇ kill 404
 ◇ killall 405
 ◇ less 95,419
 ◇ ln 100
 ◇ locate 95, 107
 ◇ logout 63, 83
 ◇ ls 97
 ◇ lynx 425
 ◇ mail 425
 ◇ make 161
 ◇ md5sum 426
 ◇ mii-tool214
 ◇ mkdir 97
 ◇ mkraid 573
 ◇ more 419
 ◇ mv 95
 ◇ netstat 475
 ◇ nice 408
 ◇ nslookup 536
 ◇ passwd 143,415
 ◇ ping 207
 ◇ poweroff 63
 ◇ prctl 646
 ◇ ps 404
 ◇ quotacheck 157
 ◇ quotaon 159
 ◇ raidhotadd 573

- ◇ raidhotremove 573
- ◇ reboot 63
- ◇ repquota 159
- ◇ rm 95, 97
- ◇ rmdir 97
- ◇ mdc-confgen 537
- ◇ route 475, 478
- ◇ rpm 163
- ◇ service 389
- ◇ shutdown 64
- ◇ ssh 426, 495
- ◇ startx62,415
- ◇ su 139
- ◇ sudo 138
- ◇ swapon 469
- ◇ tac 95
- ◇ tail 419
- ◇ telnet 426
- ◇ touch 95
- ◇ tracepath 207
- ◇ traceroute 207
- ◇ umount 108
- ◇ update-grub 377
- ◇ uptime 416
- ◇ userdel 144
- ◇ usermod 144
- ◇ users 416
- ◇ vzpkg 646
- ◇ wc 420
- ◇ which 95, 107
- ◇ who 137
- ◇ who 416
- ◇ xf86config 417
- ◇ yum 163, 166
- Компания TransGaming 338
- Коннектор сетевого кабеля 194
- Консоль 62
- Конфигуратор
 - ◇ bum 390
 - ◇ gproftpd 520
 - ◇ ppoeconf 206
 - ◇ system-config-services 390, 400
 - ◇ YaST 390
- Конфигурационный файл
 - ◇ GRUB 372
 - ◇ GRUB2 374
 - ◇ X.Org 260
- Концепция модулей 392
- Корневая файловая система 90
- Корневой раздел 33
- Коэффициент подкачки 468
- Криптографическая файловая система eCryptfs 595
- Кроссплатформенная совместимость офисных пакетов 276
- Кэширующий сервер DNS 531, 532

Л

Линус Торвальдс 24

М

Маршрутизатор 482
Маршрутизация 474
◇ пакетов 474
Массивы 132
Масштабирование изображения 280
Менеджер пакетов 187, 189
Метод

- ◇ SASL556

- ◇ отключения учетной записи root 455

Н

Настройка

- ◇ X.Org в современных дистрибутивах 259
- ◇ анонимного FTP-сервера 526
- ◇ межсетевого экрана 482
- ◇ неанонимного FTP-сервера 525
- ◇ планировщика ввода/вывода 469
- ◇ ядра 363

О

Обновление базы данных корневых серверов 541
Оболочка Unity 65, 71
Образ жесткого диска 325
Объединение интернет-каналов 231
Оператор

- ◇ case 135
- ◇ if 134

Операционная система

- ◇ AIX87
- ◇ QNX369
- ◇ UNIX 24
- ◇ UNIX-подобная 24

Опции DNS-сервера 533
Основные особенности systemd 393
Отключение учетной записи root 457
Отключить блокировку экрана 73, 77
Офисный пакет

- ◇ Calligra Suite 274
- ◇ Kingsoft Office 275
- ◇ LibreOffice 272
- ◇ OpenOffice.org 277

Очистка дерева исходного кода 366, 369

П

Пакет 160
 ◇ bind 532
 ◇ mysql-admin 512
 ◇ mysql-client 512
 ◇ mysql-server 512
 ◇ raidtools 573
 ◇ resolvconf 225
 ◇ samba 558
 ◇ samba-server 558
 ◇ зависимости 161
 ◇ конфликты 162
 Панель Unity 71
 Параллельный запуск сервисов 391
 Параметры
 ◇ виртуальной машины 330
 ◇ фильтрации пакетов 486
 ◇ ядра 357
 Пароль на вход в BIOS Setup 453
 Первичный загрузчик 385
 Первый дистрибутив Linux 25
 Переключение языков ввода 78
 Перекомпиляция ядра 361
 Переменные 131
 ◇ окружения 131
 ◇ специальные 131
 Перенаправление ввода/вывода 83
 Пиринговый протокол BitTorrent 322
 Планировщик
 ◇ anacron 472
 ◇ atd 473
 ◇ crond 470
 Повышение отказоустойчивости интернет-соединения 231
 Подключение Linux к сети Microsoft 558
 Пользователь root 51
 Порядок установки операционных систем 52
 Почтовый клиент 553
 ◇ Evolution 316
 ◇ KMail 316
 ◇ Mozilla Thunderbird 316
 Почтовый сервер 553
 Правила
 ◇ брандмауэра 490
 ◇ маршрутизации 474
 Пример файла конфигурации X.Org 262
 Проблемы при установке Linux 55
 Проверка установочного DVD 38
 Программа
 ◇ /usr/sbin/grub-mkconfig 374
 ◇ ftpcount 529
 ◇ ftpwho 529
 ◇ GParted 125
 ◇ installpkg 181
 ◇ k3b 297

◇ pkgtool 181
 ◇ removepkg 181
 ◇ mdc 535
 ◇ rpm2tgz 184
 ◇ slackpkg 184
 ◇ smbclient 561
 ◇ Transmission 322
 ◇ upgradepkg 181
 ◇ urpmi 165
 ◇ xpkgtool 181
 ◇ zypper 188
 Программные RAID-массивы 573
 Программы (агенты) передачи почты (MTA) 553
 Программы-«заглушки» 129
 Прозрачный прокси-сервер 549
 Проигрыватель
 ◇ Videos 241
 ◇ VLC 242
 Производительность файловых систем 87
 Прокси-сервер 544
 ◇ прозрачный 548
 ◇ Squid 544
 Протокол
 ◇ Kerberos 563
 ◇ TLS 556
 ◇ отправки почты (SMTP) 553
 Протоколы получения почты (POP, IMAP) 553
 Прототипы 159
 Псевдонимы команд 82, 83
 Псевдофайловая система
 ◇ proc 410
 ◇ sysfs 409
 Псевдофайловые системы 409

Р

Работа с консолью 95
 Раздел подкачки 467, 468
 Разметка диска 40, 43, 46
 Размывание изображения 285
 Распространение Linux 25
 Расширение имени файла 89
 Регион DVD-Video 295
 Редактирование
 ◇ конфигурации GRUB2 377
 ◇ параметров ядра 35, 359
 Редактор
 ◇ joe 424
 ◇ mcedit 424
 ◇ nano 424
 ◇ pico 423
 ◇ vi 420
 Резервное копирование 129
 Репозиторий 162

С

- Сервер 601
 - ◊ физический 602
 - ◊ FTP 520
 - ◊ POP 553
 - ◊ Samba 520
 - ◊ SMTP 553
 - ◊ аутентификации SASL 556
- Серверный дистрибутив 31
- Сервис
 - ◊ Samba 558
 - ◊ systemd-journald.service 440
 - ◊ интернет-телефонии Skype 316
- Сетевой протокол аутентификации Kerberos 558
- Сетевые интерфейсы 478
- Сеть
 - ◊ отказ работы 204
- Система
 - ◊ доменных имен 530
 - ◊ инициализации
 - init 386
 - Slackware 401
 - systemd 387
 - systemd 440, 445
 - upstart 386
- О хранения данных
 - распределенная 581
- Системные требования Linux 32
- Системы управления доступом 458
- Создание
 - ◊ виртуальной машины 325
 - ◊ учетных записей пользователей 52
- Специальный демон WinBind 558
- Список управления доступом (ACL) 546
- Способы
 - ◊ аутентификации 459
 - ◊ виртуализации 633
- Сравнение дистрибутивов 26
- Статическая маршрутизация 231
- Суперблок 94
- Схема разрешения доменного имени 531
- Сценарии (скрипты) GIMP 286
- Сценарий 128
- ◊ .bashrc 129

Т

- Таблица
 - ◊ маршрутизации 475, 477, 478
 - ◊ разделов 40
- Текстовый клиент ftp 319
- Терминал 83
- Терминалы 64, 83
- Технология виртуализации
 - ◊ KVM 633
 - ◊ OpenVZ 633

Точка

- ◊ доступа Wi-Fi на смартфоне 221
- ◊ монтирования 42, 114

У

- Увеличение скорости доступа к Интернету 231
- Уровни RAID-массивов 570
- Установка Linux 32, 52, 57
 - ◊ Linux по сети 52
 - ◊ из сетевых репозиториях 32
 - ◊ кодаков в openSUSE 242
 - ◊ пароля
 - root 51
 - загрузчика 453
 - ◊ программ из исходных кодов 160
 - ◊ проприетарных драйверов NVIDIA 267
 - ◊ разрешения монитора 259
- Утилита journald 440

Ф**Файл**

- ◊ .{ICE,X} authority 140
- ◊ .bash history 129
- ◊ .bash_profile 83
- ◊ /boot/boot.b 385
- ◊ /boot/grub/grub.conf 372
- ◊ /boot/grub/menu.lst 372
- ◊ /boot/map 386
- ◊ /etc/anacrontab 472
- ◊ /etc/apt/sources.list 175
- ◊ /etc/audit/auditd.conf 429
- ◊ /etc/crontab 470
- ◊ /etc/cups/printers.conf 431
- ◊ /etc/default/grub 375
- ◊ /etc/default/ufw 659
- ◊ /etc/dhcp3/dhcpd/dhcpd.conf 53
- ◊ /etc/fstab 112, 157
- ◊ /etc/group 146
- ◊ /etc/hostname 212
- ◊ /etc/HOSTNAME 212
- ◊ /etc/httpd/conf 510
- ◊ /etc/inetd.conf 54
- ◊ /etc/inittab 387
- ◊ /etc/network/interfaces 212, 478
- ◊ /etc/NetworkManager/system-connections 219
- ◊ /etc/openvpn/server.conf 658
- ◊ /etc/passwd 144
- ◊ /etc/proftpd/proftpd.conf 521
- ◊ /etc/resolv.conf 535
- ◊ /etc/resolvconf/resolv.conf.d/base 225
- ◊ /etc/route.conf 477
- ◊ /etc/samba/smb.conf 558
- ◊ /etc/shadow 145

Файл (*npod.*)

- ◇ /etc/shells 128
 - ◇ /etc/sshd_config 495
 - ◇ /etc/sudoers 139
 - ◇ /etc/sysconfig/network 209
 - ◇ /etc/sysconfig/network/config 211
 - ◇ /etc/sysconfig/network/dhcp 212
 - ◇ /etc/sysconfig/network/ifcfg-eth0 211
 - ◇ /etc/sysconfig/network/routes 211, 477
 - ◇ /etc/sysconfig/network-scripts/ifcfg-eth0 209
 - ◇ /etc/sysconfig/static-routes 211
 - ◇ /etc/ufw/before.rules 659
 - ◇ /etc/urpmi/urpmi.conf 165
 - ◇ /etc/yum.conf 169
 - ◇ /etc/zypp/zypp.conf 187
 - ◇ /proc/filesystems 409
 - ◇ /proc/sys/vm/swappiness 468
 - ◇ /var/log/messages 204
 - ◇ apache.conf 505
 - ◇ apache2.conf 505
 - ◇ aquota.user 157
 - ◇ etc/squid/squid.conf 544
 - ◇ fstab 120
 - ◇ httpd.conf 505
 - ◇ httpd2.conf 505
 - ◇ resolv.conf 536
 - ◇ smb.conf 561
 - ◇ xorg.conf 261
 - ◇ подкачки 468
 - ◇ права доступа 103
 - ◇ устройства 90, 109
- Файловая система 92
- ◇ Btrfs 87
 - ◇ ext2 85, 575
 - ◇ ext3 85
 - ◇ ext4 85, 120
 - ◇ JFS 87
 - ◇ NTFS 89
 - ◇ Reiser4 86
 - ◇ ReiserFS 575
 - ◇ ReiserFS 86
 - ◇ Tux2 87
 - ◇ Tux3 87
 - ◇ XFS 86
 - ◇ Xiafs 87
 - ◇ ZFS 87
 - ◇ журналируемая 85, 120
- Файловые системы 86
- Файловый менеджер
- ◇ Dolphin 561
- Файлы
- ◇ конфигурации RAM 459
 - ◇ устройств 90

Фильтрация пакетов 483

- Флеш-память 111
- Фон рабочего стола 78
- Форвард-сервер 534
- Формат
 - ◇ Blue-ray DVD 291
 - ◇ DVD+R/+RW 295
 - ◇ DVD-Audio 294
 - ◇ DVD-R 294
 - ◇ DVD-RAM 295
 - ◇ DVD-ROM 290, 293
 - ◇ DVD-Video 290, 293
 - ◇ DWD-RW 295

Х

- Хост-клавиша виртуальной машины 336
- Хост-машина 635

Ц

- Цепочка правил брандмауэра 483
- Цикл
 - ◇ for 133
 - ◇ while 133

Ч

- Черный список интернет-адресов 547

Ш

- Шифрование файловой системы 47
- Шлюз 482, 488
 - ◇ по умолчанию 474, 477

Э

- Эмулятор 337
 - ◇ Cedega 338
 - ◇ VirtualBox 337
 - ◇ VMware 337
 - ◇ Wine 337
 - ◇ Winex 337
 - ◇ виртуальной машины 324

Я

- Ядро 34, 351
 - ◇ модуль ядра 365
 - ◇ параметры ядра 35
 - ◇ системный вызов 34



Linux

ОТ НОВИЧКА К ПРОФЕССИОНАЛУ



Колисниченко Денис Николаевич, инженер-программист и системный администратор. Имеет богатый опыт эксплуатации и создания локальных сетей — от домашних до уровня предприятия на базе операционной системы Linux. Автор более 70 книг компьютерной тематики, в том числе «Самоучитель Linux», «Самоучитель Microsoft Windows 10», «Планшет и смартфон на базе Android для ваших родителей», «PHP и MySQL. Разработка веб-приложений», «Серверное применение Linux», «Самоучитель системного администратора Linux» и др.

Гарантия эффективной работы в Linux

Книга предназначена для широкого круга пользователей Linux и поможет им самостоятельно настроить и оптимизировать эту операционную систему. Даны ответы на все вопросы, возникающие при работе с Linux: от установки и настройки этой ОС до настройки сервера на базе Linux. Материал книги максимально охватывает все сферы применения Linux от запуска Windows-игр под управлением Linux до настройки собственного Web-сервера. Материал ориентирован на последние версии дистрибутивов Fedora, openSUSE, Slackware, Ubuntu. В шестом издании описаны виртуальные частные сети, виртуальные серверы, настройка VPN-соединения и VPN-сервера, выбор VPN-провайдера, системы виртуализации OpenVZ и Virtuozzo, программные системы хранения данных с резервированием.



Дополнительные главы в PDF-файлах и видеоуроки можно скачать по ссылке <ftp://ftp.bhv.ru/9785977539432.zip>, а также со страницы книги на сайте www.bhv.ru.

БХВ-ПЕТЕРБУРГ

191036, Санкт-Петербург,
Гончарная ул., 20
Тел.: (812) 717-10-50,
339-54-17, 339-54-28
E-mail: mail@bhv.ru
Internet: www.bhv.ru

ISBN 978-5-9775-3943-2



9 785977 539432

В ПОДЛИННИКЕ®