

Денис Колисниченко



# Linux

## ОТ НОВИЧКА К ПРОФЕССИОНАЛУ



8-е издание



Материалы  
на [www.bhv.ru](http://www.bhv.ru)

- Fedora 33, zRAM
- Варианты загрузки Linux и управление загрузкой
- Работа с файловой системой и устройствами в Linux
- Файловая система Btrfs, UUID накопителей, загрузчик GRUB2
- Настройка сети, Интернета и популярных серверов Apache, ProFTPD, Samba, BIND и др.
- Настройка SSL-сертификата, ускорение веб-сервера с помощью Memcached и Google PageSpeed
- Настройка VPN-соединения, выбор VPN-провайдера, настройка VPN-сервера
- Брандмауэр ufw
- Выбор VPS/VDS-сервера и его настройка
- Программные системы хранения данных с резервированием

Наиболее  
полное  
руководство

В ПОДЛИННИКЕ®

**Денис Колисниченко**

# **Linux**

## **ОТ НОВИЧКА К ПРОФЕССИОНАЛУ**

**8-е издание**

Санкт-Петербург  
«БХВ-Петербург»

2022

УДК 004.451  
ББК 32.973.26-018.2  
К60

**Колисниченко Д. Н.**  
K60      Linux. От новичка к профессионалу. — 8-е изд., перераб. и доп. — СПб.: БХВ-Петербург, 2022. — 688 с.: ил. — (В подлиннике)  
ISBN 978-5-9775-6773-2

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 веб-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.

Восьмом издаании рассмотрены Fedora 33, модуль zRAM, файловая система Btrfs, настройка Apache для работы на нескольких портах, организация поддоменов \*.example.com, выбор и настройка VDS, брандмаузер ufw, лайфхаки для начинающих администраторов.

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

*Для широкого круга пользователей Linux*

УДК 004.451  
ББК 32.973.26-018.2

### Группа подготовки издания:

Руководитель проекта	Евгений Рыбаков
Зав. редакцией	Людмила Гауль
Компьютерная верстка	Ольги Сергиенко
Дизайн серии	Марины Дамбировой
Оформление обложки	Каринды Соловьевой

Подписано в печать 01.11.21.  
Формат 70×100<sup>1/16</sup>. Печать офсетная. Усл. печ. л. 55,47.  
Тираж 1300 экз. Заказ № 1007.  
"БХВ-Петербург", 191036, Санкт-Петербург, Гончарная ул., 20.



Отпечатано в АО «Можайский полиграфический комбинат».  
143200, Россия, г. Можайск, ул. Мира, 93.  
[www.oaoopc.ru](http://www.oaoopc.ru), тел.: (495) 748-04-67, (49638) 20-685

# Оглавление

<b>Предисловие .....</b>	<b>17</b>
Что нового в восьмом издании? .....	17
<b>ЧАСТЬ I. ВЫБОР И УСТАНОВКА ДИСТРИБУТИВА .....</b>	<b>19</b>
<b>Глава 1. Выбор дистрибутива .....</b>	<b>21</b>
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.....	28
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
<b>Глава 2. Особенности установки.....</b>	<b>32</b>
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.....	39
2.4. Изменение таблицы разделов .....	39
2.4.1. Разметка диска в Fedora 30-33 .....	40
2.4.2. Разметка диска в Ubuntu .....	44
2.4.3. Разметка диска в openSUSE .....	45
2.4.4. Шифрование файловой системы .....	48
2.5. Выбор устанавливаемых пакетов программ .....	48
2.6. Выбор графической среды.....	50
2.7. Установка пароля root.....	51
2.8. Создание учетных записей пользователей .....	52
2.9. Порядок установки операционных систем .....	53

2.10. Установка Linux по сети .....	53
2.10.1. Немного о загрузке и установке по сети.....	53
2.10.2. Подготовка загрузочного сервера .....	53
Установка DHCP-сервера .....	54
Настройка TFTP-сервера .....	54
Загрузка установочного образа .....	55
2.10.3. Настройка клиента.....	55
2.11. Проблемы при установке .....	56
2.11.1. Проблема с APIC .....	56
2.11.2. Ошибка: <i>kernel panic: VFS: Unable to mount root fs</i> .....	56
2.11.3. Проблемы с некоторыми LCD-мониторами.....	57
2.11.4. Сообщение <i>Probing EDD</i> и зависание системы .....	57
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 .....	58
2.11.8. Проблема с механизмом Enhanced Disk Device (EDD) .....	58

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

<b>Глава 3. Сразу после установки... .....</b>	<b>61</b>
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
3.3. Изменение параметров графического интерфейса .....	71
3.3.1. Отключение блокировки экрана.....	71
3.3.2. Изменение способа переключения языков ввода .....	74
3.3.3. Изменение фона рабочего стола.....	75
3.4. «Аварийные» комбинации клавиш, использование клавиши <SysRq> .....	76
3.5. Практические приемы работы с консолью.....	77
3.5.1. Автодополнение командной строки и псевдонимы команд .....	77
3.5.2. Графические терминалы .....	78
3.5.3. Перенаправление ввода/вывода.....	78
<b>Глава 4. Файловая система Linux .....</b>	<b>80</b>
4.1. Файловые системы, поддерживаемые Linux .....	80
4.1.1. Выбор файловой системы .....	83
4.1.2. Linux и файловые системы Windows .....	84
4.1.3. Сменные носители .....	85
4.2. Особенности файловых систем Linux .....	85
4.2.1. Имена файлов в Linux .....	85
4.2.2. Файлы и устройства .....	85
4.2.3. Корневая файловая система и монтирование.....	86
4.2.4. Стандартные каталоги Linux.....	87
4.3. Внутреннее строение файловой системы .....	88
4.4. Команды для работы с файлами и каталогами.....	91
4.4.1. Работа с файлами.....	91
4.4.2. Работа с каталогами.....	93

4.5. Использование ссылок. Команда <i>ln</i> .....	95
4.5.1. Жесткие и мягкие ссылки .....	95
4.5.2. Создание ссылок .....	96
4.5.3. Определение ссылок .....	96
4.5.4. Удаление файлов и жесткие ссылки .....	97
4.5.5. Разница между копированием и созданием жесткой ссылки .....	98
4.6. Права доступа и атрибуты файла. Команды <i>chown</i> , <i>chmod</i> и <i>chattr</i> .....	99
4.6.1. Права доступа к файлам и каталогам .....	99
4.6.2. Смена владельца файла .....	101
4.6.3. Специальные права доступа (SUID и SGID) .....	101
4.6.4. Атрибуты файла. Запрет изменения файла .....	102
4.6.5. Команды поиска файлов: <i>find</i> , <i>which</i> и <i>locate</i> .....	103
4.7. Монтирование файловых систем .....	104
4.7.1. Команды <i>mount</i> и <i>umount</i> .....	104
4.7.2. Файлы устройств и монтирование .....	105
Жесткие диски .....	105
Приводы оптических дисков .....	107
Флешки и внешние жесткие диски .....	107
4.7.3. Опции монтирования файловых систем .....	108
4.7.4. Монтирование разделов при загрузке .....	109
4.7.5. Подробно о UUID и файле /etc/fstab .....	111
4.7.6. Монтирование флеш-дисков .....	114
4.8. Настройка журнала файловой системы ext3/ext4 .....	116
4.9. Файловая система ext4 .....	117
4.9.1. Сравнение ext3 и ext4 .....	117
4.9.2. Совместимость с ext3 .....	118
4.9.3. Переход на ext4 .....	118
4.10. Программы для разметки диска .....	119
4.10.1. Стандартная программа <i>fdisk</i> .....	119
4.10.2. Графическая программа для разметки диска <i>GParted</i> .....	122
4.11. Таблица разделов GPT .....	122
4.12. Несколько слов о CD/DVD-дисках .....	123
4.13. Scalpel — инструмент для восстановления файлов .....	124
4.14. Новшества Fedora 33: zRAM и Btrfs .....	126
<b>Глава 5. Командный интерпретатор bash .....</b>	<b>128</b>
5.1. bash: основные сведения .....	128
5.2. Автоматизация задач с помощью bash .....	129
5.3. Привет, мир! .....	130
5.4. Использование переменных в собственных сценариях .....	131
5.5. Передача параметров сценарию .....	132
5.6. Массивы .....	132
5.7. Циклы .....	133
5.8. Условные операторы .....	134
5.9. Мониторинг и перезапуск сервисов Apache и MySQL с помощью bash .....	135
<b>Глава 6. Пользователи и группы .....</b>	<b>138</b>
6.1. Многопользовательская система .....	138
6.2. Пользователь root .....	139
6.2.1. Полномочия пользователя root .....	139

6.2.2. Временное получение полномочий root .....	140
Команда <i>sudo</i> .....	140
Команда <i>su</i> .....	141
Команды <i>gksudo/gksu</i> и <i>kdesudo/kdesu</i> .....	141
Проблемы с <i>sudo</i> в Ubuntu и Kubuntu.....	142
Ввод серии команд <i>sudo</i> .....	142
6.2.3. Переход к традиционной учетной записи root .....	143
Преимущества и недостатки <i>sudo</i> .....	143
Традиционная учетная запись root в Ubuntu .....	145
6.3. Создание, удаление и модификация пользователей и групп стандартными средствами .....	145
6.3.1. Отдельные пользователи.....	145
6.3.2. Группы пользователей .....	148
6.4. Управление пользователями и группами с помощью графических конфигураторов .....	148
6.4.1. Конфигураторы в Fedora и Ubuntu .....	149
6.4.2. Графический конфигуратор в openSUSE.....	153
Еще о правах root и командах <i>su</i> и <i>sudo</i> применительно к openSUSE.....	156
Конфигуратор Центр безопасности openSUSE.....	157
6.5. Квотирование .....	159
<b>Глава 7. Пакеты и управление пакетами.....</b>	<b>162</b>
7.1. Способы установки программного обеспечения в Linux .....	162
7.2. Репозитории пакетов .....	164
7.3. Программы для управления пакетами .....	165
7.4. Программа rpm (все Red Hat-совместимые дистрибутивы).....	166
7.5. Программа <i>igrpmi</i> .....	167
7.5.1. Установка пакетов .....	167
7.5.2. Обновление и удаление пакетов .....	168
7.5.3. Поиск пакета. Получение информации о пакете.....	168
7.6. Программа yum .....	168
7.6.1. Использование yum .....	168
7.6.2. Управление источниками пакетов .....	171
7.6.3. Установка пакетов через прокси-сервер.....	172
7.6.4. Плагины для yum .....	173
7.7. Менеджер пакетов dnf .....	173
7.8. Программы dpkg и apt-get: установка пакетов в Debian/Ubuntu .....	175
7.8.1. Программа dpkg .....	175
7.8.2. Программа apt .....	177
7.8.3. Установка RPM-пакетов в Debian/Ubuntu .....	178
7.8.4. Подключение репозитория Medibuntu .....	179
7.8.5. Графические менеджеры в Debian/Ubuntu .....	179
7.8.6. Волшебная команда <i>update</i> .....	181
7.9. Установка пакетов в Slackware .....	181
7.9.1. Управление пакетами .....	183
Программа установки пакетов <i>installpkg</i> .....	184
Программа удаления пакетов <i>removepkg</i> .....	185
Программа обновления пакетов <i>upgradepkg</i> .....	185
7.9.2. Нет нужного пакета: вам поможет программа <i>rpm2tgz</i> .....	185
7.9.3. Программа <i>slackpkg</i> : установка пакетов из Интернета .....	186

7.10. Установка программ в openSUSE .....	187
7.10.1. Менеджер пакетов zypper .....	187
7.10.2. Графический менеджер пакетов openSUSE .....	190
7.11. Снапы .....	191
7.11.1. Введение в снапы .....	191
7.11.2. Работа со снапами .....	192
<b>ЧАСТЬ III. НАСТРОЙКА СЕТИ И ИНТЕРНЕТА .....</b>	<b>195</b>
<b>Глава 8. Настройка локальной сети .....</b>	<b>197</b>
8.1. Локальная сеть с использованием технологии Gigabit Ethernet .....	197
8.2. Файлы конфигурации сети в Linux .....	200
8.3. Об именах сетевых интерфейсов .....	201
8.4. Настройка сети с помощью конфигуратора nm-connection-editor .....	204
8.5. Конфигуратор netconfig в Slackware .....	208
8.6. Утилиты для диагностики соединения .....	208
8.7. Для фанатов, или настройка сети вручную .....	212
8.7.1. Конфигурационные файлы Fedora/CentOS .....	213
8.7.2. Конфигурационные файлы openSUSE .....	215
8.7.3. Конфигурационные файлы старых версий Debian/Ubuntu .....	216
8.7.4. Команда <i>hostnamectl</i> .....	217
8.7.5. Команда <i>mii-tool</i> .....	218
8.8. Еще несколько слов о настройке сети .....	219
<b>Глава 9. Настройка соединения Wi-Fi .....</b>	<b>220</b>
9.1. Настройка беспроводного соединения с помощью NetworkManager .....	220
9.2. Что делать, если сети нет в списке? .....	225
9.3. Точка доступа Wi-Fi на смартфоне .....	226
<b>Глава 10. Настройка VPN-соединения .....</b>	<b>228</b>
10.1. Вкратце о выборе VPN-сервера и тарифного плана .....	228
10.2. Настройка VPN-подключения .....	230
<b>Глава 11. Объединение интернет-каналов .....</b>	<b>233</b>
11.1. Цели и средства решения задачи .....	233
11.2. Простой способ со статической маршрутизацией .....	234
11.3. Сложный способ с гибкой настройкой отказоустойчивости .....	236
<b>ЧАСТЬ IV. LINUX ДОМА И В ОФИСЕ .....</b>	<b>241</b>
<b>Глава 12. Поддержка форматов мультимедиа .....</b>	<b>243</b>
12.1. Что такое кодеки и почему их нет в Linux? .....	243
12.2. Настройка дистрибутива Fedora 32-33 .....	244
12.3. Установка кодеков в openSUSE .....	244
12.4. Установка кодеков в Ubuntu 20.10 .....	248
12.5. Домашний медиацентр на основе openELEC .....	249
12.5.1. Выбор дистрибутива .....	249
12.5.2. Установка дистрибутива .....	250
12.5.3. Настройка и использование .....	253
12.5.4. Удаленный доступ .....	259

12.5.5. А где же консоль?	259
12.5.6. Ложки дегтя	260
<b>Глава 13. Графическая подсистема</b>	<b>261</b>
13.1. Настройка X.Org в современных дистрибутивах	261
13.2. Конфигурационный файл X.Org	262
13.3. Синтаксис файла xorg.conf	264
13.4. Установка проприетарных драйверов NVIDIA в Fedora 21–29	270
13.5. Команда xrandr	274
<b>Глава 14. Офисные пакеты</b>	<b>277</b>
14.1. Выбор офисного пакета	277
14.1.1. LibreOffice	277
14.1.2. Calligra Suite	279
14.1.3. WPS Office (Kingsoft Office)	280
14.2. Кроссплатформенная совместимость	281
14.3. Вкратце об OpenOffice.org	282
<b>Глава 15. Графический редактор GIMP</b>	<b>283</b>
15.1. Начало работы	283
15.2. Обработка фотографий	285
15.2.1. Изменение размера (масштабирование)	285
15.2.2. Вращение	287
15.2.3. Кадрирование (обрезка)	288
15.2.4. Инструмент <i>Размытие-Резкость</i>	288
15.3. Работа в GIMP с помощью скриптов	291
15.4. Windows-версия GIMP	291
<b>Глава 16. Обзор текстовых редакторов кода</b>	<b>293</b>
16.1. Текстовые редакторы vi, nano, pico, ee, mcedit	293
16.2. Современные редакторы кода	298
16.2.1. Atom	298
16.2.2. Sublime Text 3	299
16.2.3. Brackets от Adobe	299
<b>Глава 17. Популярные программы для работы с Интернетом</b>	<b>301</b>
17.1. Браузер Firefox	301
17.2. Браузер Chromium	302
17.3. Почтовый клиент	303
17.4. Skype	304
17.5. FTP-клиенты	305
17.6. P2P-клиенты	308
<b>Глава 18. Виртуальная машина VirtualBox</b>	<b>310</b>
18.1. Зачем нужна виртуальная машина?	310
18.2. Установка эмулятора VirtualBox	311
18.3. Создание новой виртуальной машины	312
18.4. Изменение параметров виртуальной машины	316
18.4.1. Общие параметры	316
18.4.2. Раздел <i>Система</i>	317

18.4.3. Виртуальные жесткие диски.....	317
18.4.4. А нужен ли звук? .....	319
18.4.5. Параметры сети .....	319
18.4.6. Последовательные порты.....	321
18.5. Запуск виртуальной машины и установка гостевой операционной системы .....	322
<b>Глава 19. Эмулятор Wine: запуск Windows-игр в Linux .....</b>	<b>323</b>
19.1. Эмуляторы, эмуляторы... .....	323
19.2. Установка Wine .....	324
19.3. Настройка Wine и прозрачного запуска Windows-приложений .....	325
19.4. Использование Wine.....	327
<b>ЧАСТЬ V. СИСТЕМНЫЕ ТРИОКИ, ИЛИ LINUX ИЗНУТРИ .....</b>	<b>329</b>
<b>Глава 20. Ядро.....</b>	<b>331</b>
20.1. Процесс загрузки ядра .....	331
20.2. Параметры ядра .....	334
20.3. Компиляция ядра в дистрибутиве Ubuntu.....	339
20.3.1. Установка дополнительных пакетов .....	339
20.3.2. Загрузка исходных текстов ядра.....	339
20.3.3. Настройка ядра .....	341
20.3.4. Компиляция ядра .....	343
20.4. RT-ядро.....	348
20.5. Особенности компиляции ядра в других дистрибутивах Linux.....	349
<b>Глава 21. Загрузчики Linux.....</b>	<b>350</b>
21.1. Основные загрузчики .....	350
21.2. Конфигурационные файлы GRUB и GRUB2 .....	351
21.2.1. Конфигурационный файл GRUB.....	351
21.2.2. Конфигурационный файл GRUB2.....	353
21.3. Команды установки загрузчиков .....	356
21.4. Установка собственного фона загрузчиков GRUB и GRUB2.....	357
21.5. Постоянные имена устройств .....	358
21.6. Восстановление загрузчика GRUB/GRUB2 .....	358
21.7. Загрузка с ISO-образов.....	359
21.8. Установка пароля загрузчика .....	360
21.8.1. Загрузчик GRUB .....	360
21.8.2. Загрузчик GRUB2 .....	362
<b>Глава 22. Системы инициализации.....</b>	<b>365</b>
22.1. Начальная загрузка Linux .....	365
22.2. Система инициализации init.....	367
22.2.1. Команда <i>init</i> .....	369
22.2.2. Команда <i>service</i> .....	369
22.2.3. Редакторы уровней запуска .....	370
22.2.4. Параллельная загрузка сервисов, или как сделать старый init быстрее.....	370
22.3. Система инициализации systemd .....	371
22.3.1. Идеальная система инициализации .....	371
22.3.2. <i>systemd</i> — основные понятия .....	372

22.3.3. Основные особенности systemd.....	374
22.3.4. Сравнение init, upstart и systemd .....	374
22.3.5. Немного практики .....	376
22.3.6. Команды системного администратора.....	380
22.4. Система инициализации Slackware .....	381
<b>Глава 23. Процессы.....</b>	<b>383</b>
23.1. Аварийное завершение процесса .....	383
23.2. Программа top: кто больше всех расходует процессорное время? .....	385
23.3. Изменение приоритета процесса .....	387
23.4. Запуск NodeJs-приложений в фоновом режиме.....	387
<b>Глава 24. Псевдофайловые системы sysfs и proc .....</b>	<b>389</b>
24.1. Виртуальная файловая система sysfs .....	389
24.2. Виртуальная файловая система proc .....	390
24.2.1. Информационные файлы .....	390
24.2.2. Файлы, позволяющие изменять параметры ядра.....	391
24.2.3. Файлы, изменяющие параметры сети.....	392
24.2.4. Файлы, изменяющие параметры виртуальной памяти.....	392
24.2.5. Файлы, позволяющие изменить параметры файловых систем.....	393
24.3. Сохранение произведенных изменений.....	393
<b>Глава 25. Команды Linux, о которых нужно знать каждому линуксоиду .....</b>	<b>394</b>
25.1. Общие команды .....	394
25.1.1. Команда arch — вывод архитектуры компьютера .....	394
25.1.2. Команда clear — очистка экрана.....	394
25.1.3. Команда date .....	394
25.1.4. Команда echo.....	395
25.1.5. Команда exit — выход из системы.....	395
25.1.6. Команда man — вывод справки .....	395
25.1.7. Команда passwd — изменение пароля .....	395
25.1.8. Команда startx — запуск графического интерфейса X.Org .....	395
25.1.9. Команда uptime — информация о работе системы .....	396
25.1.10. Команда users — информация о пользователях.....	396
25.1.11. Команды w, who и whoami — информация о пользователях .....	396
25.1.12. Команда xf86config — настройка графической подсистемы .....	397
25.2. Команды для работы с текстом .....	397
25.2.1. Команды diff и cmp — сравнение файлов .....	397
25.2.2. Команды grep и egrep — текстовый фильтр .....	398
25.2.3. Команды more и less — постраничный вывод .....	399
25.2.4. Команды head и tail — вывод начала и хвоста файла .....	399
25.2.5. Команда wc — подсчет слов в файле.....	400
25.2.6. Команды vi, nano, pico, ee, mcedit — текстовые редакторы .....	400
25.2.7. Язык gawk — мощное средство обработки текста .....	400
25.3. Команды для работы с Интернетом .....	400
25.3.1. Команда ftp — стандартный FTP-клиент.....	400
25.3.2. Команда lynx — текстовый браузер .....	401
25.3.3. Команда mail — чтение почты и отправка сообщений .....	402

25.4. Команды системного администратора.....	402
25.4.1. Команды <i>free</i> и <i>df</i> — информация о системных ресурсах .....	402
25.4.2. Команда <i>md5sum</i> — вычисление контрольного кода MD5.....	402
25.4.3. Команды <i>ssh</i> и <i>telnet</i> — удаленный вход в систему.....	403
<b>Глава 26. Конфигурационные файлы Linux .....</b>	<b>404</b>
26.1. Каталог /etc.....	404
26.2. Каталог /etc/NetworkManager.....	405
26.3. Каталог /etc/abrt .....	406
26.4. Каталог /etc/alsa.....	406
26.5. Каталоги /etc/audit и /etc/audisp .....	406
26.6. Каталог /etc/avahi — файлы конфигурации демона Avahi .....	406
26.7. Файлы конфигурации планировщиков задач .....	407
26.8. Каталог /etc/cups .....	407
26.9. Файл /etc/fonts/fonts.conf .....	409
26.10. Каталог /etc/gdm (или /etc/gdm3) .....	410
26.11. Файлы конфигурации популярных сетевых служб .....	410
26.12. Каталог /etc/logrotate.d.....	410
26.13. Каталог /etc/mail.....	412
26.14. Каталог /etc/ntp.....	412
26.15. Каталог /etc/openldap .....	412
26.16. Каталог /etc/openvpn .....	412
26.17. Каталоги /etc/pam.d и /etc/security .....	412
26.18. Каталог /etc/ppp.....	412
26.19. Каталог /etc/rc.d.....	413
26.20. Каталог /etc/sane.d.....	413
26.21. Каталог /etc/selinux .....	413
26.22. Каталог /etc/skel .....	413
26.23. Каталог /etc/sysconfig.....	414
26.24. Каталог /etc/X11 .....	415
26.25. Конфигурационные файлы yum/dnf .....	415
26.26. Основные конфигурационные файлы сети.....	415
26.27. Остальные конфигурационные файлы каталога /etc.....	415
<b>Глава 27. Протоколирование системы .....</b>	<b>417</b>
27.1. Протоколирование по-новому: journalctl.....	418
27.1.1. Установка времени .....	418
27.1.2. Просмотр и фильтрация логов.....	419
Текущая и предыдущие загрузки .....	419
Фильтр по дате .....	420
Фильтр по сервису.....	421
Фильтр по пути.....	421
Фильтр по процессу или пользователю.....	421
Просмотр сообщений ядра .....	421
Фильтр по уровню ошибки.....	421
27.1.3. Журналы в реальном времени .....	422
27.1.4. Централизованное хранение логов.....	422
27.2. Демоны syslogd и rsyslogd.....	422

<b>ЧАСТЬ VI. LINUX НА СЕРВЕРЕ .....</b>	<b>427</b>
<b>Глава 28. Обеспечение безопасности сервера.....</b>	<b>429</b>
28.1. Защита от «восстановления пароля root».....	429
28.1.1. Параметр ядра <i>single</i> .....	429
28.1.2. Пароль загрузчиков GRUB/GRUB2 .....	431
28.1.3. Осторожно: LiveCD .....	431
28.2. Защита от перезагрузки.....	431
28.3. Отключение учетной записи root: нестандартный метод.....	433
28.4. Отключение учетной записи root средствами KDM и GDM.....	435
28.5. Системы управления доступом .....	436
<b>Глава 29. Модули аутентификации PAM.....</b>	<b>437</b>
29.1. Каталог /etc/pam.d .....	437
29.2. Дополнительные файлы конфигурации.....	438
29.2.1. Содержимое каталога /etc/security.....	438
29.2.2. Файл <i>access.conf</i> : ограничение доступа к системе.....	439
29.2.3. Файл <i>limits.conf</i> : ограничение на используемые системные ресурсы .....	440
29.2.4. Файл <i>time.conf</i> : регистрация только в рабочее время .....	441
29.3. Список PAM-модулей .....	442
29.4. Борьба с простыми паролями .....	443
<b>Глава 30. Оптимизация системы. Автоматизация выполнения задач .....</b>	<b>445</b>
30.1. Оптимизация подкачки .....	445
30.2. Создание файла подкачки .....	446
30.3. Настройка планировщика ввода/вывода.....	447
30.4. Двухканальный режим памяти .....	448
30.5. Автоматизация выполнения задач.....	448
30.5.1. Планировщик <i>crond</i> .....	448
30.5.2. Планировщик <i>anacron</i> .....	450
30.5.3. Разовое выполнение команд — демон <i>atd</i> .....	451
<b>Глава 31. Маршрутизация. Настройка брандмауэра.....</b>	<b>452</b>
31.1. Таблица маршрутизации ядра. Установка маршрута по умолчанию .....	453
31.2. Изменение таблицы маршрутизации. Команда <i>route</i> .....	456
31.3. Включение IPv4-переадресации, или превращение компьютера в шлюз.....	459
31.4. Настройка брандмауэра.....	460
31.4.1. Цепочки и правила.....	461
31.4.2. Брандмауэр <i>iptables</i> .....	463
31.4.3. Шлюз своими руками .....	467
<b>Глава 32. Безопасный удаленный доступ. OpenSSH.....</b>	<b>473</b>
32.1. Протокол SSH .....	473
32.2. Использование SSH-клиента .....	474
32.3. Настройка SSH-сервера.....	474
<b>Глава 33. Веб-сервер. Связка Apache + PHP + MySQL.....</b>	<b>479</b>
33.1. Самый популярный веб-сервер .....	479
33.2. Установка веб-сервера и интерпретатора PHP. Выбор версии.....	479
33.3. Тестирование настроек.....	483

33.4. Файл конфигурации веб-сервера.....	485
33.4.1. Базовая настройка.....	485
33.4.2. Самые полезные директивы файла конфигурации .....	486
33.4.3. Директивы <i>Directory</i> , <i>Limit</i> , <i>Location</i> , <i>Files</i> .....	487
33.4.4. Работа сервера на нескольких портах.....	490
33.4.5. Динамические поддомены .....	491
33.5. Управление запуском сервера Apache .....	492
33.6. Оптимизация Apache .....	492
33.7. Пользовательские каталоги.....	494
33.8. Установка сервера баз данных MySQL.....	495
33.8.1. Установка сервера .....	495
33.8.2. Изменение пароля root и добавление пользователей.....	495
33.8.3. Запуск и останов сервера .....	498
33.8.4. Программа phpMyAdmin.....	498
33.9. Обеспечение безопасности сайта от вирусов .....	500
33.9.1. Как вирусы попадают на сайт? .....	500
33.9.2. Установка прав доступа .....	501
33.9.3. Антивирус ClamAV .....	502
33.9.4. Сценарий scanner .....	503
33.10. SSL-сертификат для сайта.....	504
33.10.1. Выбор SSL-сертификата .....	504
Основные типы сертификатов.....	504
Какой тип сертификата выбрать? .....	505
Особенности SSL-сертификатов разных типов .....	505
Где купить SSL-сертификат?.....	508
33.10.2. Конвертирование сертификатов.....	509
33.10.3. Сертификат Let's Encrypt.....	510
Установка клиента Let's Encrypt.....	510
Создаем каталог webroot-path/.well-known/acme-challenge/ .....	510
Создаем файл конфигурации.....	511
Запрос сертификата.....	511
Настройка веб-сервера.....	512
Автоматическое обновление сертификата .....	513
33.11. Ускорение веб-сервера: PageSpeed и Memcached.....	514
33.11.1. Установка PageSpeed .....	514
33.11.2. Установка Memcached .....	515
33.12. Протоколирование POST-запросов.....	516
<b>Глава 34. FTP-сервер .....</b>	<b>517</b>
34.1. Установка FTP-сервера .....	517
34.2. Конфигурационный файл.....	518
34.3. Настройка FTP-сервера.....	522
34.4. Оптимизация FTP-сервера .....	524
34.5. Программы <i>ftpwho</i> и <i>ftpcount</i> .....	526
34.6. Несколько слов о защите FTP.....	527
<b>Глава 35. DNS-сервер.....</b>	<b>528</b>
35.1. Еще раз о том, что такое DNS .....	528
35.2. Кэширующий сервер DNS .....	529

35.3. Полноценный DNS-сервер .....	534
35.4. Вторичный DNS-сервер .....	539
35.5. Обновление базы данных корневых серверов.....	539
<b>Глава 36. Прокси-сервер: Squid и squidGuard.....</b>	<b>542</b>
36.1. Зачем нужен прокси-сервер в локальной сети? .....	542
36.2. Базовая настройка Squid.....	542
36.3. Практические примеры .....	544
36.3.1. Управление доступом.....	544
36.3.2. Создание «черного» списка адресов .....	545
36.3.3. Отказ от баннеров.....	545
36.4. Управление прокси-сервером squid .....	545
36.5. Настройка клиентов.....	546
36.6. Прозрачный прокси-сервер.....	546
36.7. squidGuard — ваше дополнительное «оружие».....	547
<b>Глава 37. Почтовый сервер.....</b>	<b>551</b>
37.1. Выбор почтового сервера .....	551
37.2. Настройка МТА Exim.....	553
37.3. Настройка аутентификации SMTP .....	554
37.4. Настройка демона SASL .....	555
<b>Глава 38. Сервис Samba .....</b>	<b>556</b>
38.1. Установка Samba.....	556
38.2. Базовая настройка Samba .....	556
38.3. Настройка общих ресурсов .....	558
38.4. Просмотр ресурсов Windows-сети .....	559
38.5. Оптимизация Samba .....	559
38.6. Samba и Active Directory .....	561
38.7. Samba в качестве контроллера домена.....	564
<b>Глава 39. Поддержка RAID .....</b>	<b>568</b>
39.1. Аппаратные RAID-массивы .....	568
39.2. Программные RAID-массивы.....	571
39.3. Создание программных массивов .....	572
39.4. RAID-массив только для данных.....	573
39.5. Сбой и его имитация.....	574
<b>Глава 40. Программные системы хранения данных.....</b>	<b>576</b>
40.1. Аппаратные хранилища с резервированием .....	576
40.2. Программные хранилища с резервированием .....	578
40.3. Распределенная система хранения данных Серф .....	580
40.3.1. Система Серф: дополнительная информация .....	581
<b>Глава 41. Средства резервного копирования. Создание образа системы на LiveUSB .....</b>	<b>582</b>
41.1. Необходимость в «живой» резервной копии.....	582
41.2. Средства клонирования Linux .....	583
41.3. Clonezilla.....	584
41.4. Linux Live .....	591

<b>Глава 42. Шифрование файловой системы.....</b>	<b>593</b>
42.1. Шифрование папки.....	593
42.2. Храним пароль на флешке .....	595
<b>ЧАСТЬ VII. ВИРТУАЛЬНЫЕ СЕРВЕРЫ .....</b>	<b>597</b>
<b>Глава 43. А нужен ли физический сервер?.....</b>	<b>599</b>
43.1. Физический или виртуальный? .....	599
43.1.1. Стоимость физического сервера .....	599
43.1.2. Необходимость в аппаратном сервере.....	600
43.1.3. Про VPS, VDS и спекулянтов.....	601
43.1.4. Стоимость VDS.....	603
43.1.5. Физический сервер или VDS? .....	604
43.1.6. Стоимость владения физическим сервером .....	605
43.1.7. Выводы .....	606
43.2. Виртуальный тест-драйв .....	606
43.2.1. «Джино».....	607
О ценах .....	607
Создание сервера.....	608
Тестирование .....	609
Выводы.....	613
43.2.2. «Спринтхост» .....	613
О ценах .....	613
Создание сервера.....	614
Тестирование .....	615
Выводы.....	617
43.2.3. «Макхост».....	618
О ценах .....	618
Создание сервера.....	618
Тестирование .....	619
Выводы.....	621
43.2.4. «UltraVDS» .....	621
О ценах .....	621
Создание сервера.....	621
Тестирование .....	624
Выводы.....	625
43.2.5. Облачный сервис «1cloud».....	626
О ценах .....	626
Тестирование .....	627
Выводы.....	629
43.3. Сравнительная таблица .....	630
43.4. Сразу после покупки виртуального Linux-сервера. Шесть шагов к безопасности сервера.....	631
43.4.1. Меняем пароль пользователя root .....	631
43.4.2. Создаем обычного пользователя .....	631
43.4.3. Установка удобного редактора.....	631
43.4.4. Превращаем обычного пользователя в администратора.....	632
43.4.5. Запрещаем вход как root по SSH .....	633

43.4.6. Настройка брандмауэра .....	633
Базовая настройка .....	633
Создание правил для сервисов .....	635
Разрешаем IP-адреса .....	636
Запрещаем IP-адреса и службы .....	636
Удаление/сброс правил .....	636
<b>Глава 44. Сервер виртуализации OpenVZ .....</b>	<b>637</b>
44.1. Способы виртуализации .....	637
44.2. Установка OpenVZ .....	639
44.3. Создание и настройка виртуального контейнера .....	641
44.4. Запуск виртуальной машины .....	642
<b>Глава 45. Знакомство с Virtuozzo Linux.....</b>	<b>644</b>
45.1. Что такое Virtuozzo? .....	644
45.2. Как это работает? .....	644
45.3. Системные требования и ограничения .....	645
45.4. Установка Virtuozzo .....	646
45.5. Выбор шаблона .....	649
45.6. Создание и настройка контейнера .....	650
45.7. Управление ресурсами контейнера .....	651
45.8. Управление контейнерами .....	653
45.9. Запуск команд и вход в гостевую операционную систему .....	654
45.10. Настройка сети .....	655
45.11. Делаем работу с Virtuozzo удобнее .....	658
<b>Глава 46. Сервер виртуальной частной сети.....</b>	<b>659</b>
46.1. Настройка собственного VPN-сервера .....	659
46.2. Установка OpenVPN .....	660
46.3. Настройка центра сертификации .....	660
46.4. Создание сертификата и ключей для сервера .....	661
46.5. Создание сертификата и ключей для клиента .....	662
46.6. Настройка сервера OpenVPN .....	662
46.7. Инфраструктура настройки клиентов .....	664
46.8. Настройка клиентов .....	666
<b>Глава 47. Виртуальные диски на виртуальном сервере .....</b>	<b>668</b>
47.1. Добавление еще одного виртуального диска .....	668
47.2. Расширение существующего диска .....	671
<b>Приложение. Описание файлового архива .....</b>	<b>675</b>
<b>Предметный указатель .....</b>	<b>677</b>

# 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», начинающий<sup>1</sup> 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 восьмом 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 удивишь. Вот лишь некоторые основные обновления материала, произведенные для восьмого издания книги:

- в версии 33 дистрибутива Fedora по умолчанию стала использоваться файловая система Btrfs, что послужило причиной изменений в главе 4. Также в ней описано другое нововведение Fedora 33 — zRAM;
- в главу 5 добавлены практические примеры bash-сценариев, а именно — сценарии мониторинга и автоматического перезапуска сервисов Apache и MySQL, что позволяет обойтись без установки стороннего ПО (вроде monit);
- изменениям также подверглась вся третья часть книги — по большей части изменения эти косметические, однако в главе 10 вместо канувшего в Лету SecurityKISS рассмотрен другой VPN-сервис;

---

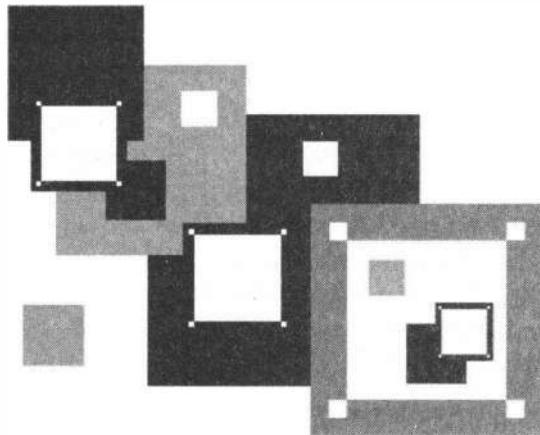
<sup>1</sup> Обращаясь здесь к начинающему пользователю, автор отнюдь не имеет в виду сугубого новичка, впервые подсаживающегося к компьютеру... Напротив, книга ориентирована на вполне уверенного современного пользователя Windows или Mac, по тем или иным причинам заинтересовавшегося работой в Linux.

- в главах 22 и 23 представлены интересные «лайфхаки» для начинающих системных администраторов: в главе 22 — запуск Python HTTPServer как системного сервиса, а в главе 23 — запуск NodeJS-приложения в фоновом режиме;
- традиционно для последних изданий этой книги подверглась изменениям глава 33. На этот раз, помимо настройки Apache, огромное внимание уделяется и настройке PHP. В частности, показано, как установить PHP версии, которая отличается от той, что установлена в дистрибутиве по умолчанию. Там также рассказывается, как настроить Apache для работы на нескольких портах одновременно и как организовать динамические поддомены \*.example.com, чтобы не пришлось переконфигурировать сервер при добавлении очередного поддомена. Поскольку в связке Apache и PHP часто используется MySQL, то настройка MySQL-сервера и создание базы данных и пользователя также рассмотрены в главе 33. Нужно отметить, что в версии MySQL 8 эти процессы отличаются от тех же процессов для версии MySQL 5.x;
- глава 43 в восьмом издании стала практической. Кроме рекомендаций по выбору VDS, в ней описаны действия, которые нужно выполнить сразу после его приобретения. Также в этой главе рассмотрен процесс работы с брандмауэром ufw.

Более подробно описывать здесь все произведенные для восьмого издания книги изменения смысла нет — скажу только, что была актуализирована большая часть всего материала.

Файловый архив с информацией, расширяющей и дополняющей материал «бумажной» книги, можно скачать с FTP-сервера издательства «БХВ» по ссылке: <ftp://ftp.bhv.ru/9785977567732.zip>, а также со страницы книги на сайте <https://bhv.ru/>. Подробная информация об этом архиве приведена в *приложении*.

Приятного чтения!

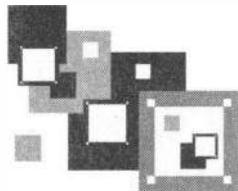


## ЧАСТЬ I

# Выбор и установка дистрибутива

*Первая часть книги*, 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. Раньше 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. Впрочем, есть и некоторые нюансы. Если в предыдущем издании книги я отмечал, что мне не очень нравится Fedora, поскольку при начальной загрузке этого дистрибутива с его установочного носителя нельзя было выбрать русский язык (рис. 1.1), то к этому списку сейчас добавился еще и дистрибутив Ubuntu 20.10 — вместо привычного меню загрузчика мы видим меню загрузчика по умолчанию (рис. 1.2) так же без возможности выбора русского языка. Все это мелочи, но именно из всех таких мелочей и складывается общее впечатление о дистрибутиве.

Но если Ubuntu перед запуском собственно процесса установки все же позволяет выбрать русский язык (1.3), то Fedora отображает подобное окно по-прежнему без списка выбора языка (рис. 1.4).

Это были плохие новости. А хорошие заключаются в том, что если не считать неприятных моментов с локализацией загрузочного носителя, весь интерфейс в новых дистрибутивах переведен более качественно и остается все меньше непереведенных надписей.

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

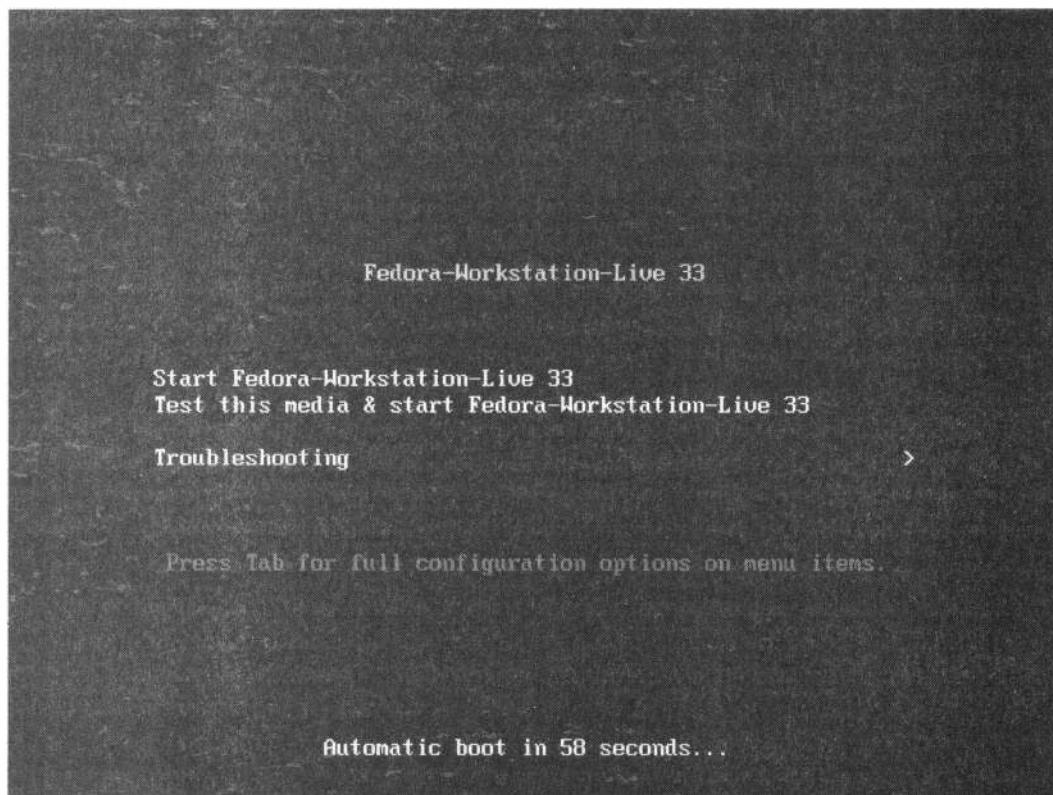


Рис. 1.1. Меню загрузчика при установке Fedora 33

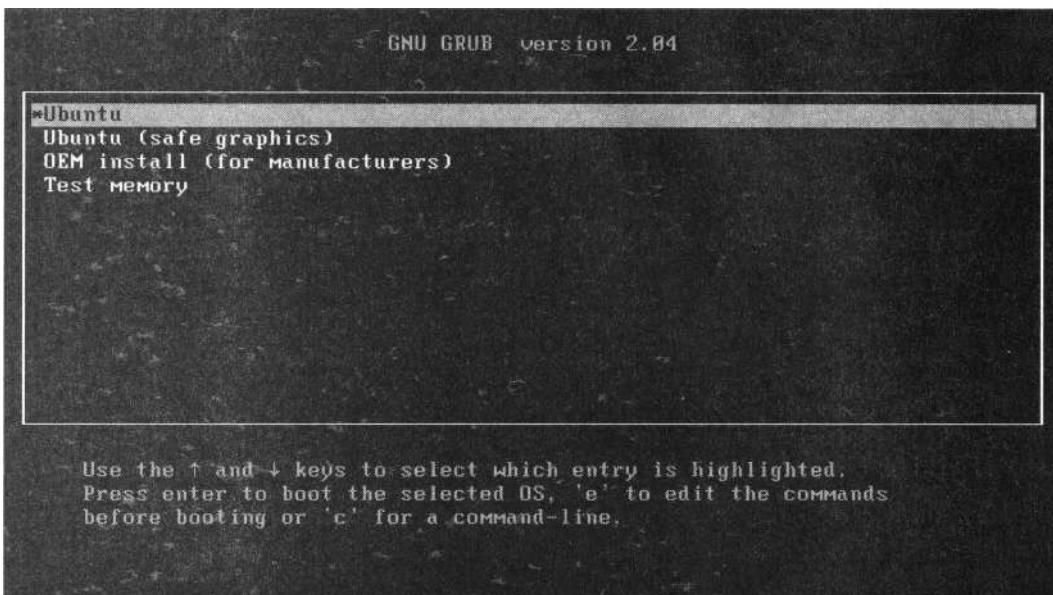


Рис. 1.2. Меню загрузчика при установке Ubuntu 20.10



Рис. 1.3. Сразу после загрузки с LiveUSB Ubuntu 20.10



Рис. 1.4. Сразу после загрузки с LiveUSB Fedora 33

## 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 назвали 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 «мертвые») 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.

Если вас интересует, каким дистрибутивом пользуюсь я, — это Ubuntu. В том числе и на серверах — все мои виртуальные серверы работают под управлением Ubuntu разных версий. Версия выбирается в зависимости от необходимой версии программного обеспечения, входящего в состав дистрибутива, — на сервере последние версии ПО нужны далеко не всегда.

## 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 версия 7.1 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 десять лет 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](http://fedoraproject.org)) — вполне приличный дистрибутив. Да, в нем есть определенные недоработки, но их не больше, чем в других. Здесь мы рассматриваем одну из самых последних на момент написания этих строк (октябрь 2020 года) версий Fedora — 33-ю.

По сравнению с предыдущими версиями новшеств в версии 33 не так уж и много:

- рабочий стол GNOME 3.38;
- в качестве файловой системы по умолчанию используется файловая система Btrfs;
- в консоли текстовый редактор vi заменен на текстовый редактор nano. Это нововведение вызывает двойственное чувство: с одной стороны — наконец-то додумались! С другой — не до такой уж степени это важная «фича», чтобы упоминать о ней в списке «Что нового?»;
- для подкачки используется zRAM (см. далее);
- сервис systemd-resolved включен по умолчанию;
- есть и другие незначительные изменения.

- Обычно в каждом следующем выпуске Fedora появляются достаточно новые и экспериментальные решения. В рассматриваемом здесь выпуске — это zRAM. zRAM увеличивает производительность системы, используя для подкачки страниц вместо жесткого диска сжатую область в оперативной памяти, пока не возникнет необходимости все же задействовать файл подкачки на жестком диске. Скорость обмена с оперативной памятью выше, чем с жестким диском, — соответственно zRAM позволяет Linux производить большее число операций подкачки в единицу времени, особенно на старых компьютерах с малым объемом оперативной памяти.

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

- **Server** (Сервер) — все самое необходимое для построения сервера, при этом графический интерфейс отсутствует (его можно установить отдельно, но зачем он серверу?), по умолчанию используется файловая система XFS;
- **Workstation** (Рабочая станция) — идеален для офисных/домашних компьютеров. По умолчанию устанавливается графический интерфейс и неплохой набор программ;
- **IoT (Internet of Things)** — начиная с выпуска 33, проект Fedora IoT, позиционирующийся в качестве комплексного решения для Интернета вещей, получил статус официальной редакции дистрибутива и доступен для загрузки с официального сайта.

### 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 дистрибутивом, — 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](http://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.

В последней (9.1) версии ALT Linux, вышедшей 29 июля 2020 года, появилась возможность установки сервера видеоконференций Jitsi Meet, сокращен размер минимальной установки и обновлена пакетная база.

## 1.2.5. Debian

Debian ([www.debian.org](http://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](http://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](http://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 — 20.10). Существуют 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 пяти лет. Это означает, что для установленной в 2018 году текущей на тот момент LTS-версии (18.04 LTS) окончание поддержки наступит только в 2023 году. График цикла релизов доступен по адресу <https://ubuntu.com/about/release-cycle>.

Следует учесть, что LTS-дистрибутивы больше имеет смысл 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](http://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. 28 лет назад. Тем не менее дистрибутив не заброшен, 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!). Но как бы там ни было, Slackware будет рассмотрен в нашей книге, чтобы после ее прочтения вы смогли работать и с ним.

### 1.2.8. openSUSE

openSUSE ([www.opensuse.org](http://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 не 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?

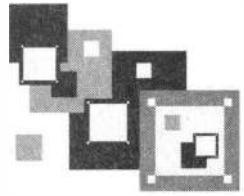
Очень часто читатели задают именно этот вопрос. И немудрено, ведь Linux — это не только настольная система, и весьма часто приходится на базе Linux настраивать сервер. Но какой дистрибутив для этого выбрать?

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, веб-сервера в нем 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 22–33 (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. Впрочем, на текущий момент Ubuntu является достаточно стабильным дистрибутивом, и если вы к нему привыкли (например, как я), его можно с успехом использовать и на серверах. За последние четыре года все возникшие на моем сервере проблемы никак не были связаны с самим дистрибутивом и могли возникнуть при использовании любого дистрибутива Linux.

Остается только посоветовать: чтобы выбрать лучший дистрибутив, нужно попробовать хотя бы 3–4 разных дистрибутива и выбрать лучший для себя.



## 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 весьма 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.

Ради эксперимента я попытался установить современные дистрибутивы в виртуальную машину с одним гигабайтом оперативной памяти. Инсталлятор последней версии openSUSE — 15.2 Tumbleweed<sup>1</sup> — после добавления сетевых репозиториев переключился в консольный режим и «порадовал» меня ошибкой. Когда же объем памяти был увеличен до двух гигабайт, установка продолжилась нормально. Инсталлятор Ubuntu 19.04 также не захотел на одном гигабайте нормально работать — постоянно зависал после запуска. Что же касается Ubuntu 20.10, то на двух гигабайтах она запустилась и работала, но для полноценной работы пришлось добавить еще один гигабайт (поскольку тестирование проводилось на виртуальной машине, проблем с добавлением еще одного гигабайта не было). На практике это означает,

---

<sup>1</sup> Разработчики openSUSE изменили схему нумерации версий, поэтому не удивляйтесь, что в предыдущем издании книги рассматривалась версия 42.3, а здесь — 15.2. Это самая последняя на момент подготовки книги версия openSUSE.

что для полноценной работы с Ubuntu 20.10 ваш компьютер должен быть оснащен как минимум 4 Гбайт ОЗУ.

Рекомендуемые системные требования для openSUSE сейчас выглядят так: минимум 2 Гбайт ОЗУ, двухъядерный процессор с частотой 2 ГГц или выше, 40 Гбайт на жестком диске. Замечания к выпуску можно прочитать по адресу: [https://doc.opensuse.org/release-notes/x86\\_64/openSUSE/Leap/15.1/](https://doc.opensuse.org/release-notes/x86_64/openSUSE/Leap/15.1/).

Конечно, все современные компьютеры оснащены как минимум четырьмя гигабайтами ОЗУ, и сообщение, показанное на рис. 2.1, скорее всего, вы никогда не получите. Но Linux всегда славилась небольшими требованиями к оперативной памяти, а сейчас же, как можно видеть, и ей уже нужен минимум 2 Гбайт... К слову, 1 Гбайт ОЗУ — это как раз минимальные требования для Windows 10 (и я запускал Windows 10 в виртуальной машине с одним гигабайтом оперативной памяти, хотя работать в Windows 10 при таком объеме памяти не слишком комфортно), так что по системным требованиям Linux уже опередила Windows.

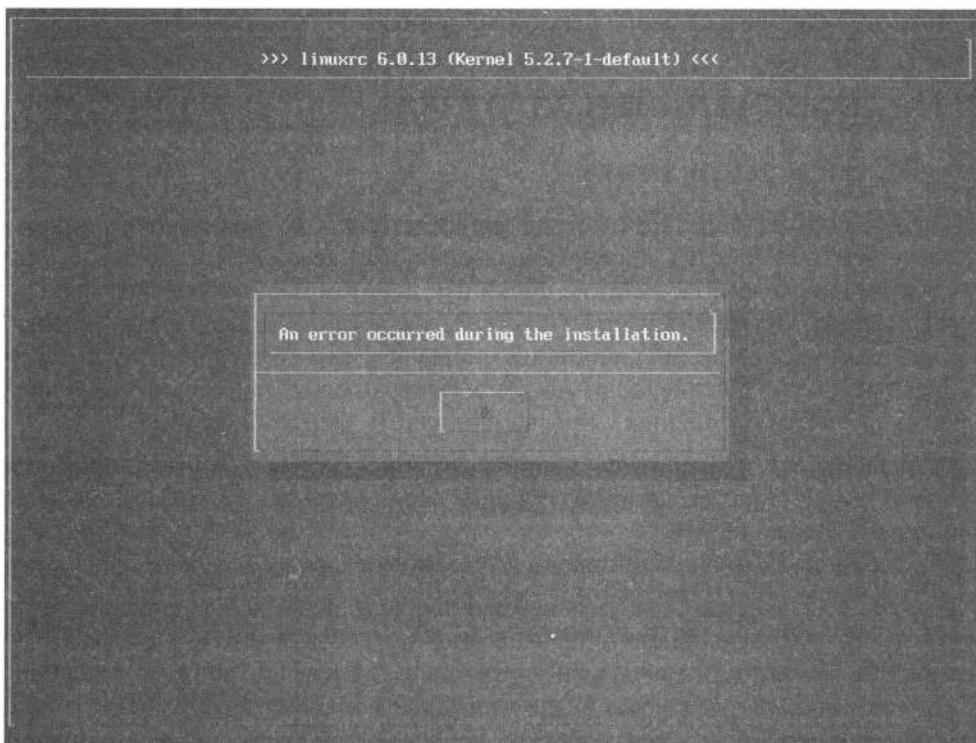


Рис. 2.1. openSUSE 15.1: ошибочка вышла

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 два гигабайта (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 два гигабайта 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 15.2 сразу после установки заняла почти 6,4 Гбайт (версия с KDE), Ubuntu 20.10 — до 7,8 Гбайт (с установкой обновлений), а Fedora 33 — 6,9 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. 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 других пользователей. 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 `portcmcia`. 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  $640\times480$  пикселов, и это позволяет произвести установку на самые «древние» компьютеры или такие, видеокарта которых не полностью совместима с Linux (редко, но и так бывает).

В различных дистрибутивах редактирование параметров ядра, естественно, осуществляется по-разному. Так, в Fedora 33 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).

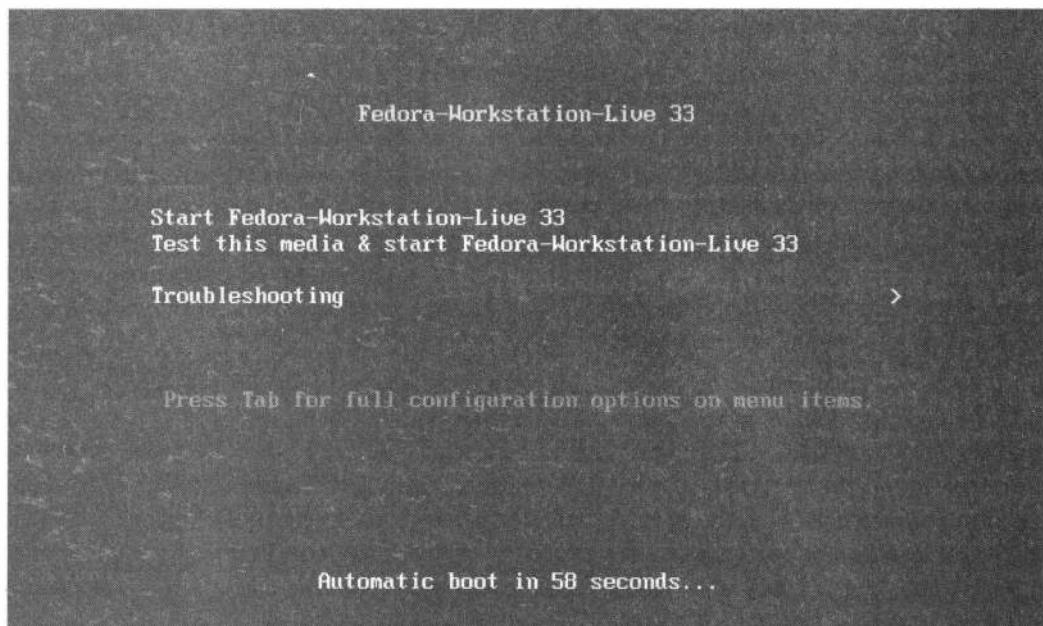
### **ЗАПУСК ИНСТАЛЛЯЦИИ FEDORA LIVE**

Если вы скачали дистрибутив Fedora 33 Live, то для запуска инсталляции надо сначала выбрать опцию **Start Fedora-Workstation-Live 30** (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. На сайте по адресу: [https://docs.fedoraproject.org/en-US/Fedora/html/Installation\\_Guide/chap-anaconda-boot-options.html](https://docs.fedoraproject.org/en-US/Fedora/html/Installation_Guide/chap-anaconda-boot-options.html) (или <https://bit.ly/2THUIQR>) вы можете ознакомиться с параметрами программы установки Fedora.

При установке openSUSE для редактирования параметров ядра следует выбрать нужный вариант установки (рис. 2.3), нажать клавишу `<F5>` и добавить параметры



**Рис. 2.2.** Fedora 33: редактирование параметров ядра

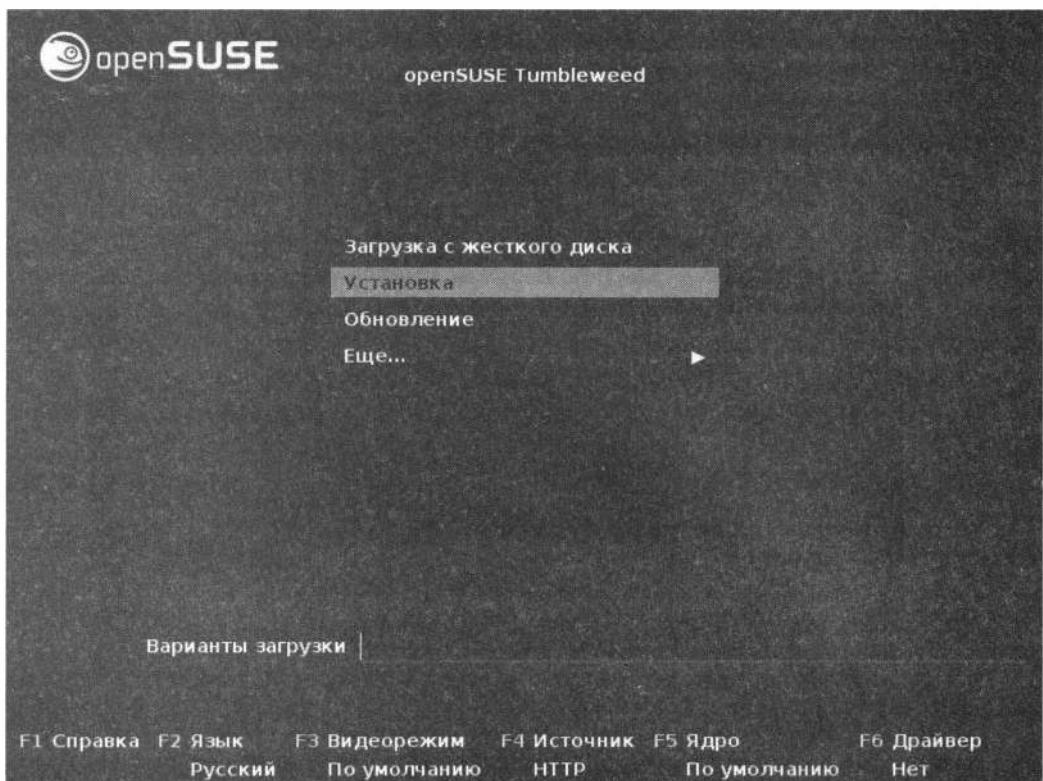


Рис. 2.3. openSUSE Tumbleweed 15.1: начальное меню установки

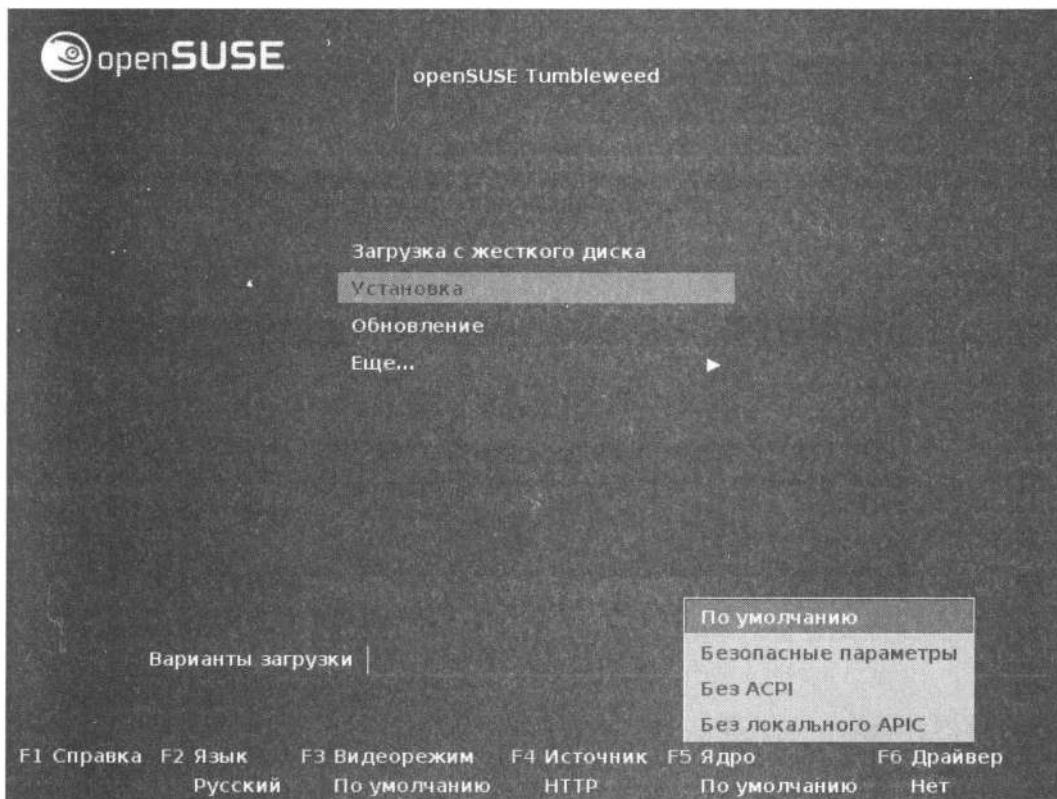


Рис. 2.4. openSUSE Tumbleweed 15.1: редактирование параметров ядра при установке

загрузки в поле **Варианты загрузки**, которое находится в нижней части экрана (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 (да и у Ubuntu) она, к сожалению, отсутствует.

Как уже отмечалось в главе 1, с последними версиями Ubuntu не все так просто. Раньше (до версии 20.10) при установке Ubuntu сначала появлялась графическая заставка — вам надо было нажать здесь любую клавишу, и на экран выводилось меню выбора языка. После выбора языка появлялось загрузочное меню на выбранном пользователем языке. Для выбора параметров ядра следовало нажать клавишу <F6>. Словом, это был привычный для Ubuntu экран загрузчика. В версии 20.10 все немного иначе. Экран загрузчика теперь стал текстовым и классическим — никакой роскоши вроде выбора языка или разрешения экрана (рис. 2.5). Здесь нужно выбрать пункт **Ubuntu**, поскольку параметры ядра мы будем изменять для него, и нажать клавишу <e>, в результате чего у вас появится возможность редактировать параметры ядра. Когда все будет готово (рис. 2.6), нажмите комбинацию клавиш

<Ctrl+X> или <F10> для загрузки с новыми параметрами ядра (подробнее о параметрах ядра вы сможете прочитать в главе 20).

### **ВСПОМОГАТЕЛЬНЫЕ ВИДЕОФАЙЛЫ**

В папке *Видео* сопровождающего книгу файлового архива (см. *приложение*) содержатся видеофайлы, демонстрирующие процессы передачи параметров ядра, входа как root без пароля, а также процесс установки на компьютер дистрибутива Ubuntu 20.10.

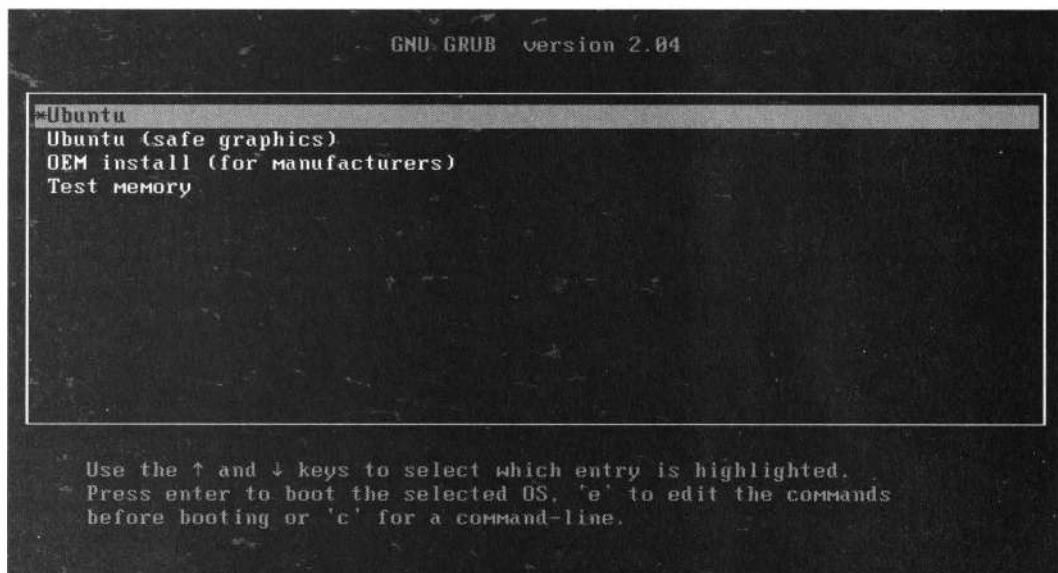


Рис. 2.5. Ubuntu 20.10: главный экран загрузчика

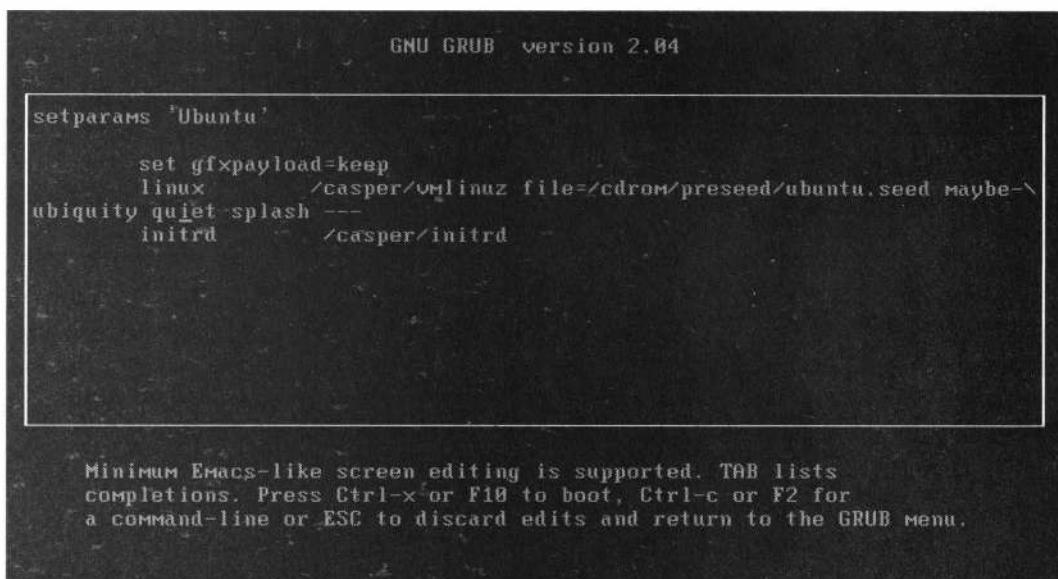


Рис. 2.6. Ubuntu 20.10: редактирование параметров ядра при установке

## 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 удается 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.

Для проверки носителя Fedora нужно выбрать в загрузочном меню опцию **Test this media & start Fedora-Workstation-Live 33** (см. рис. 2.2), а в Ubuntu — опцию **Test memory** (см. рис. 2.5). В openSUSE следует выбрать команду **Еще** (см. рис. 2.4), а затем — **Проверка носителей дистрибутива**.

## 2.4. Изменение таблицы разделов

Система Linux не может быть установлена в Windows-разделы типа 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. Поэтому 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, и она может лишь совсем удалить этот раздел. Поэтому перед установкой Linux на компьютер, где уже установлена другая операционная система, я рекомендую сделать резервную копию всех важных данных и использовать для разметки или переразметки диска 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. первый блок раздела должен располагаться до 1024-го цилиндра). Это не проблема операционной системы, а требование старого загрузчика Linux. В некоторых случаях проблему удается 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.

### Загрузчик LILO

Описание давным-давно устаревшего загрузчика LILO (раздел главы 21 из 2-го издания книги) вы найдете в папке *Дополнения* сопровождающего книгу файлового архива (см. приложение).

Перед установкой Linux следует также произвести *дефрагментацию* того Windows-раздела, который вы собирались 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 30-33

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).

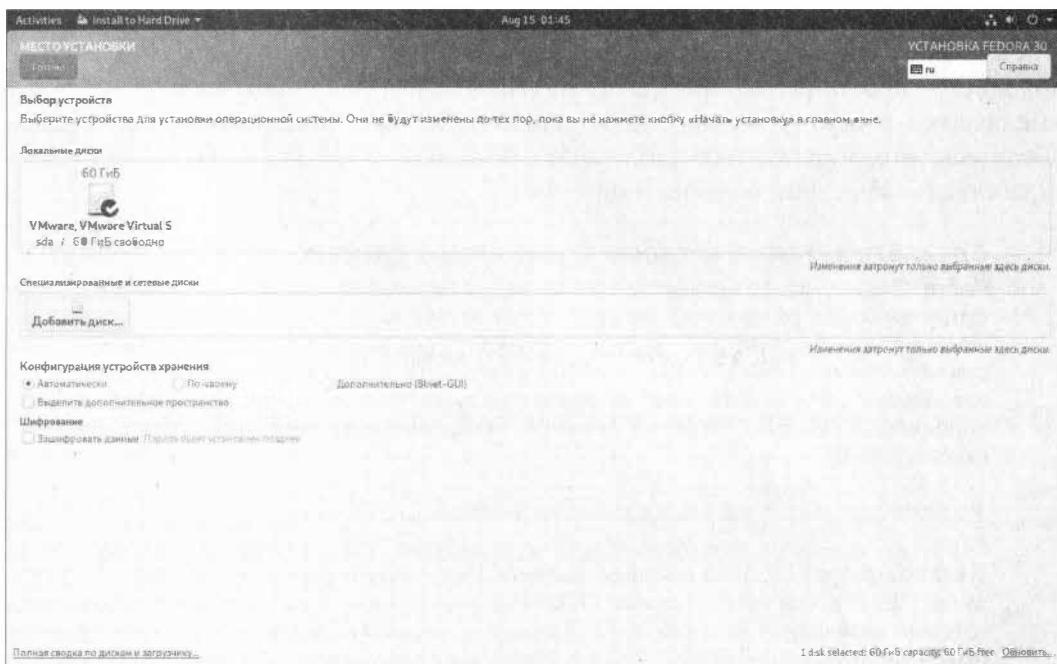


Рис. 2.7. Fedora 30: выбор типа разметки

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).

### ПРИМЕЧАНИЕ

Пусть вас не смущает, что скриншоты приводятся из 30-й версии Fedora, — в версии 33 по этой части ничего не изменилось, в чем вы можете убедиться, посмотрев видеофайл процесса установки Fedora 33, содержащийся в папке **Видео сопровождающего** книги файлового архива.

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).

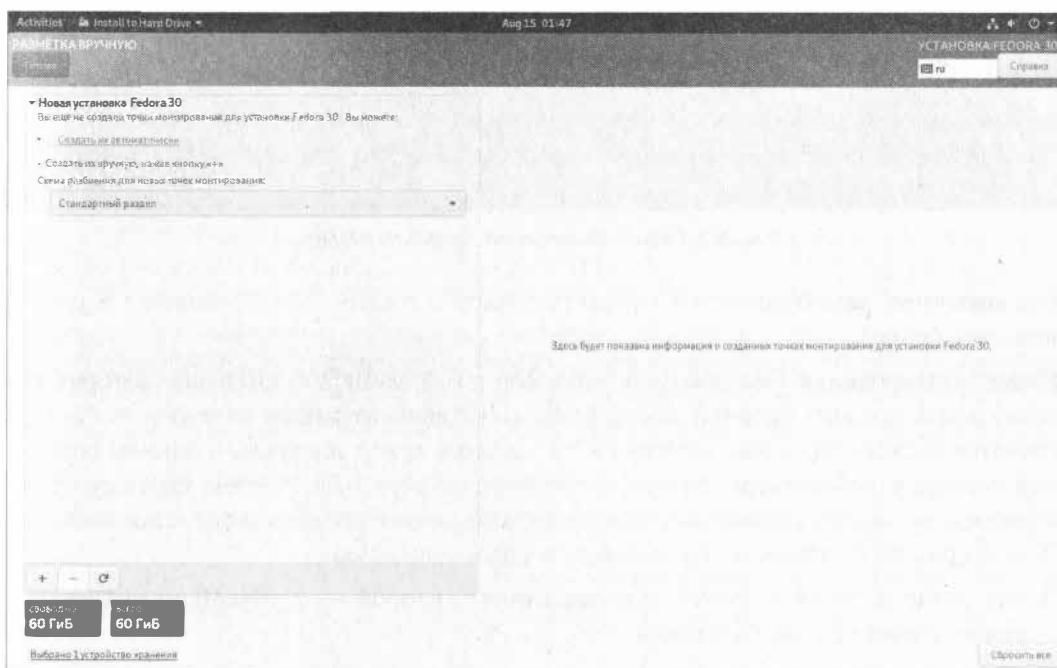


Рис. 2.8. Fedora 30: выберите **Стандартный раздел**

Затем для создания раздела нажмите кнопку **+** (для удаления раздела, соответственно, служит кнопка **-**), после чего в открывшемся окне (рис. 2.9) выберите *точку монтирования* и введите размер раздела. При вводе размера можно к численной величине добавить символы G или ГиБ<sup>1</sup> (в современных версиях Fedora единицы объема можно вводить на русском языке), чтобы указать, что размер задается в гигабайтах.

<sup>1</sup> В предыдущих версиях Fedora можно было вводить привычные ГБ, но сейчас в качестве единицы измерения используется ГиБ — именно в таком написании. Очень непривычно...

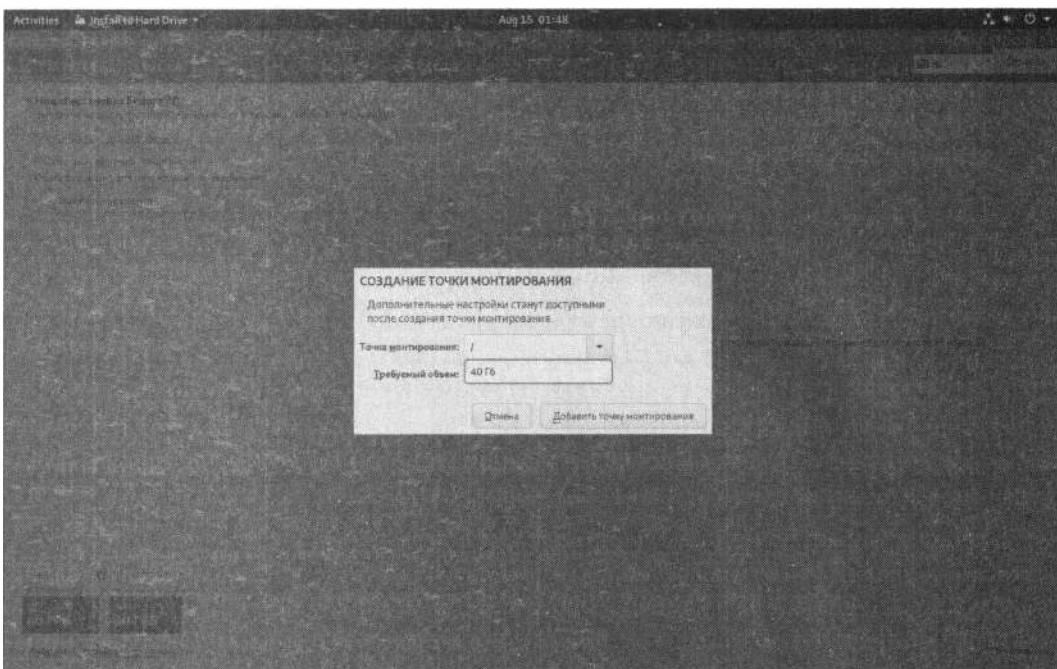


Рис. 2.9. Fedora 30: создание основного раздела

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: веб-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).

Основной недостаток программы разметки диска в Fedora — она не умеет уменьшать размер существующих на диске 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

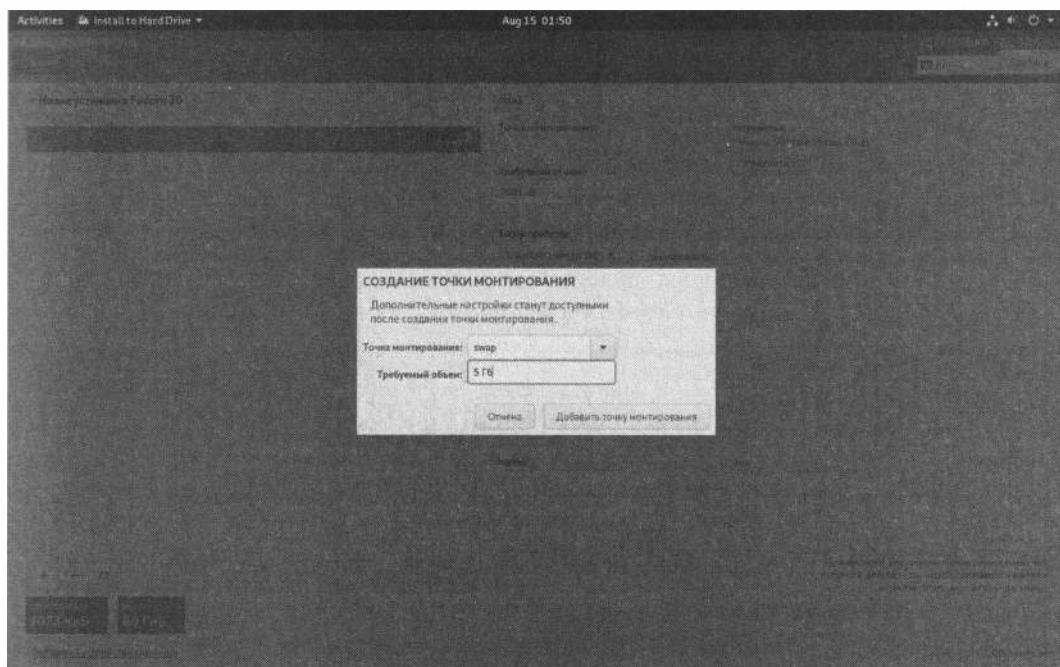


Рис. 2.10. Fedora 30: создание раздела подкачки

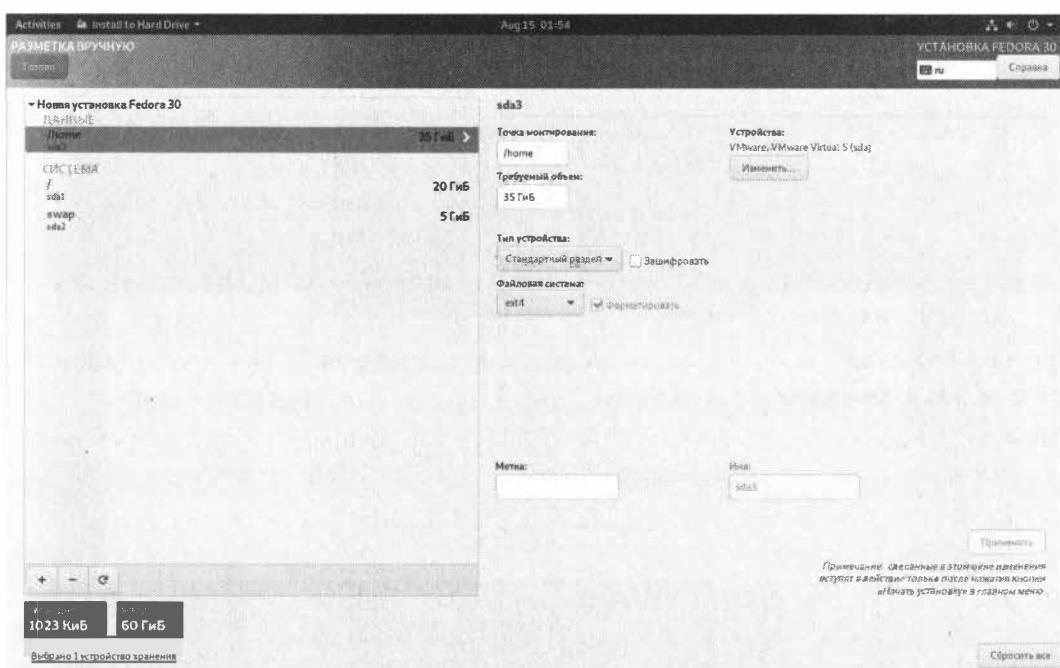


Рис. 2.11. Fedora 30: разметка диска завершена

один раздел, будет весьма проблематичной, — потребуется или задействовать сторонние программы разметки диска (например, уже упомянутую ранее AOMEI Partition Assistant — хотя бы для изменения размера существующего раздела Windows), или переустанавливать Windows с переразметкой диска. 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.

С другой стороны, все большую популярность приобретает концепция одной операционной системы, когда Linux является единственной ОС на компьютере, — тогда в общем-то все равно, насколько удобна программа разметки.

## 2.4.2. Разметка диска в Ubuntu

Программа разметки диска в Ubuntu мне понравилась гораздо больше, чем программа разметки 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 в настольном варианте (для рабочих станций) устанавливается файловая система Btrfs (до версии 33 — ext4), в серверном — XFS, а пользователю предоставляется возможность лишь указать размер раздела и точку монтирования. После создания раздела вы можете в Fedora сменить тип файловой сис-

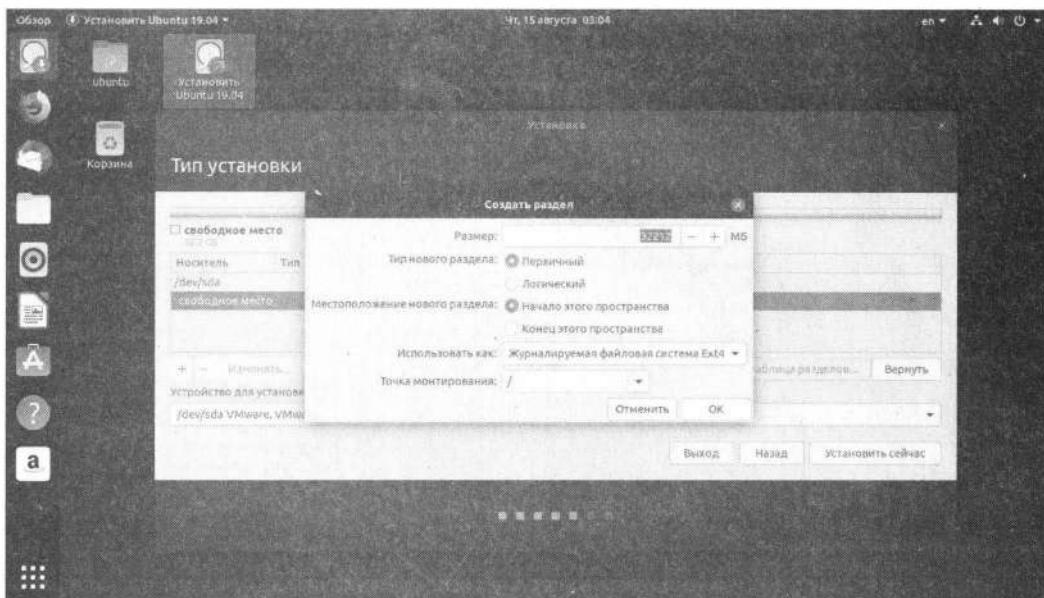


Рис. 2.12. Ubuntu 19.04/20.10: создание нового раздела

темы, но для этого нужно будет производить лишние действия: выбирать раздел, менять тип файловой системы, нажимать кнопку **Применить**.

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).

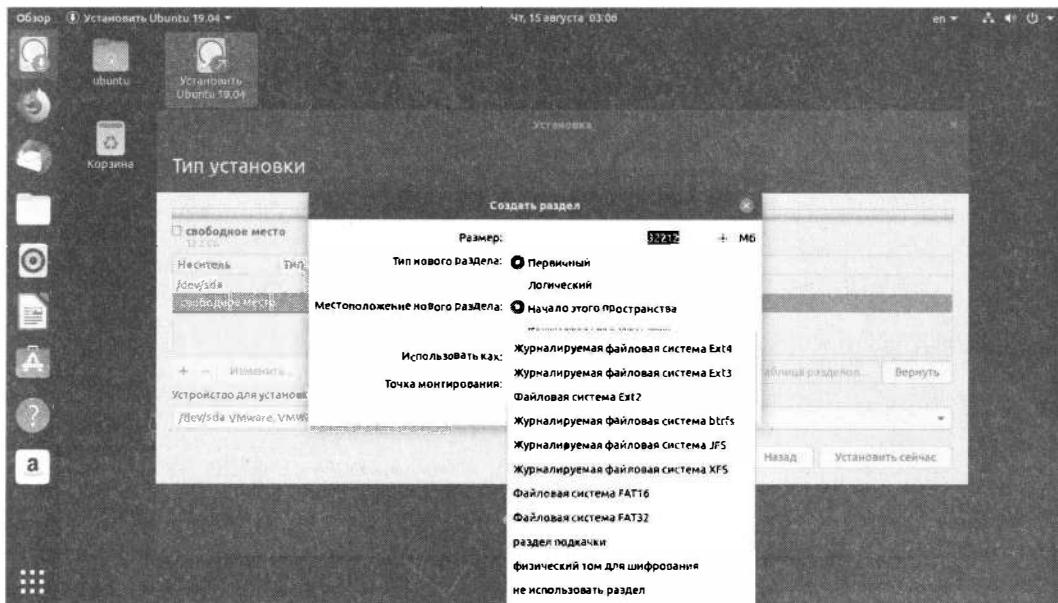


Рис. 2.13. Ubuntu 19.04/20.10: выбор файловой системы

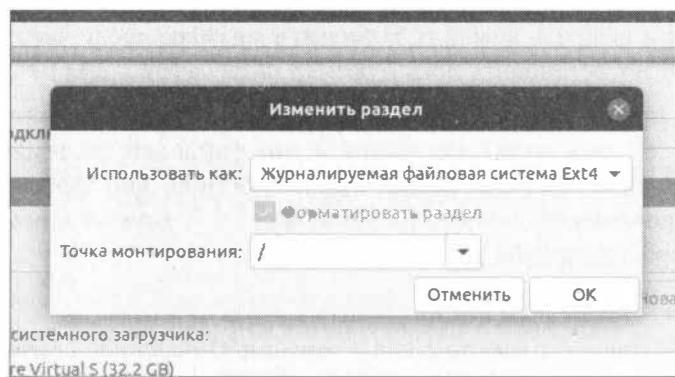


Рис. 2.14. Ubuntu 19.04/20.10: изменение раздела

### 2.4.3. Разметка диска в openSUSE

openSUSE — один из немногих дистрибутивов, программа разметки которого позволяет *изменить размер* существующих разделов. Для этого, когда программа установки предложит вам свою разметку диска (рис. 2.15), следует нажать кнопку

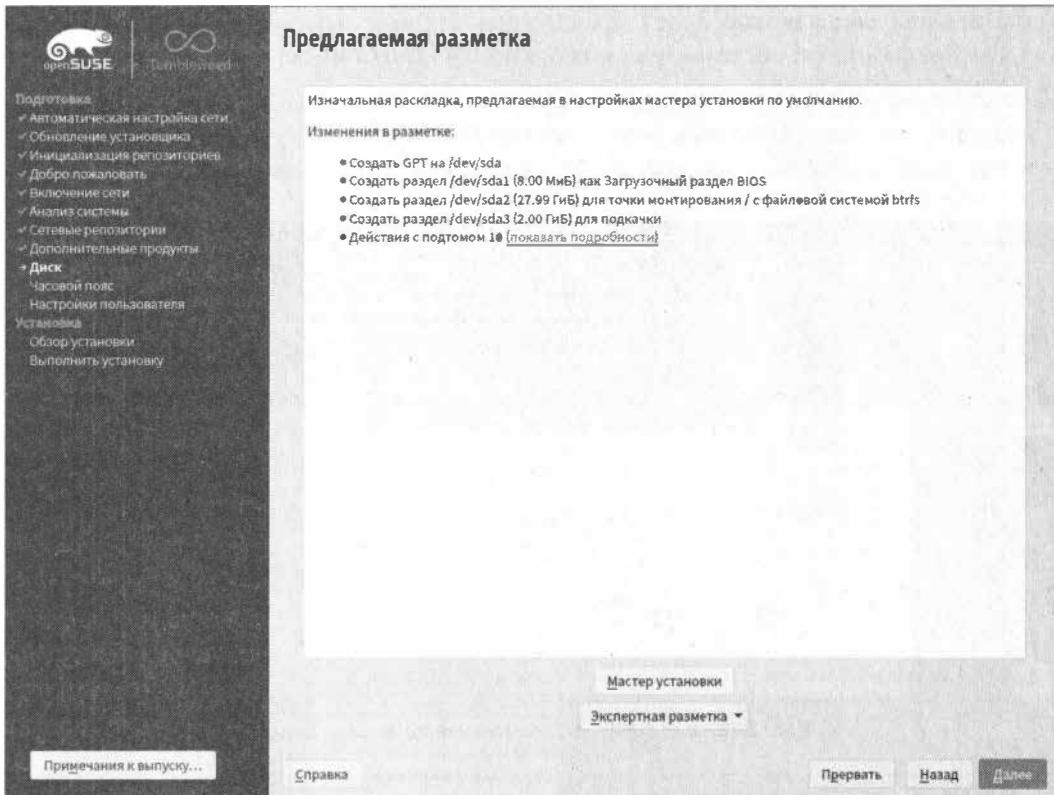


Рис. 2.15. openSUSE 15.1: нажмите кнопку Экспертная разметка

**Экспертная разметка**, в открывшемся окне щелкнуть правой кнопкой мыши на нужном разделе 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 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.

#### ПРОГРАММЫ РАЗМЕТКИ ДИСКА

В разд. 4.10 описана разметка диска с помощью стандартной программы `fdisk` и приведена информация о редакторе разделов `GParted`.

#### ПРЕЗЕНТАЦИЯ УСТАНОВКИ ДИСТРИБУТИВА SLACKWARE

В папке *Дополнения* сопровождающего книгу файлового архива (см. *приложение*) содержится файл презентации *Slackware14.ppt*, наглядно демонстрирующий процесс разметки диска с помощью программы `fdisk` и установки на компьютер дистрибутива Slackware 14.

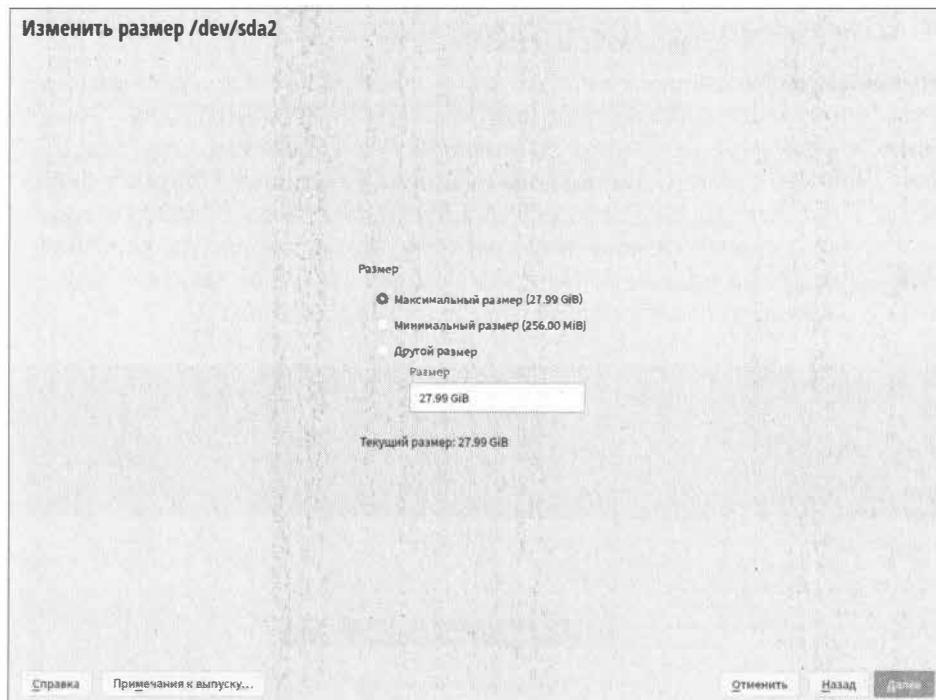


Рис. 2.16. openSUSE 15.1: изменение размера раздела

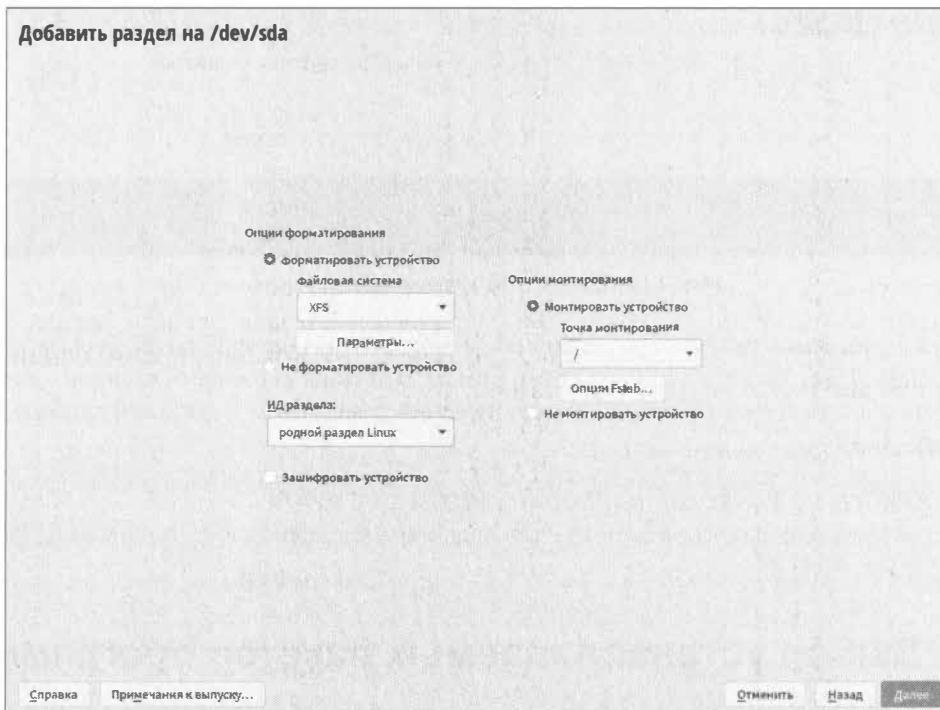


Рис. 2.17. openSUSE 15.1: возможности программы разметки

## 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 параметр **Зашифровать устройство** (см. рис. 2.17) или в Ubuntu 19.04 — параметр **Зашифровать новую установку Ubuntu в целях безопасности**. В Ubuntu 20.10 для появления этого заветного параметра нужно раскрыть список **Дополнительные возможности**. Возможность шифрования станет доступной при использовании LVM или ZFS (рис. 2.18). Как можно видеть, в зависимости от дистрибутива этот параметр называется по-разному.

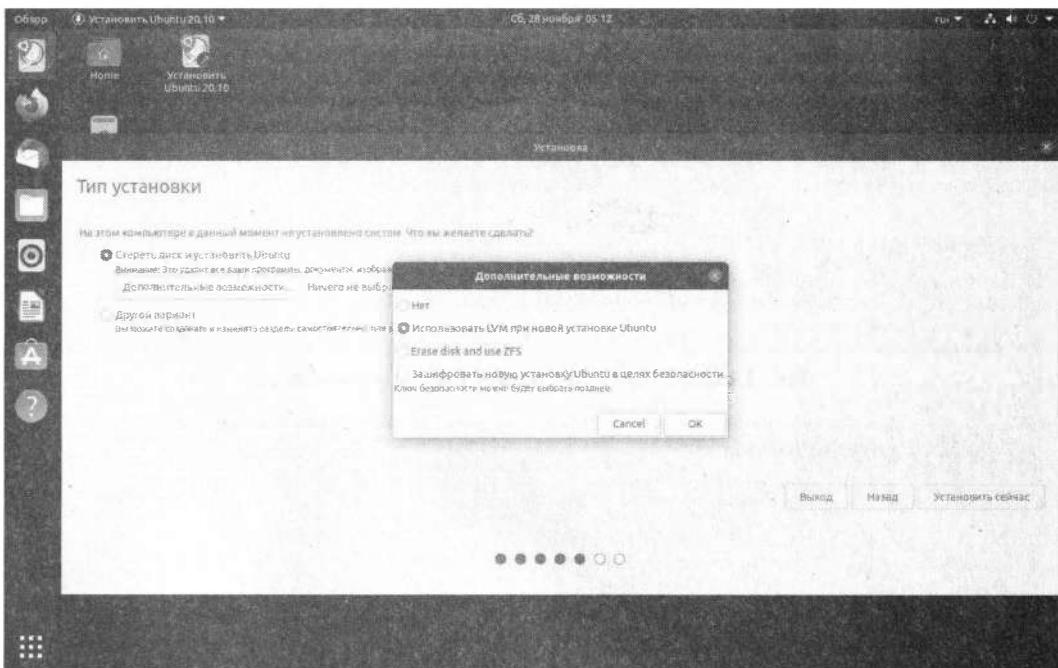


Рис. 2.18. Ubuntu 20.10: включение шифрования

Но нужно ли вам это? Если вы агент 007 — бесспорно, шифрование весьма полезная опция. А вот во всех остальных ситуациях при попытке восстановления данных в случае 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.

## 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 выбирать 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 тов<sup>1</sup> — 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 ошибается и отведет под раздел с системой, допустим, всего 8 Гбайт, а системе потребуется 10... Однако, пока не начнешь установку, об этом не узнаешь. К чему такая простота — не понятно...

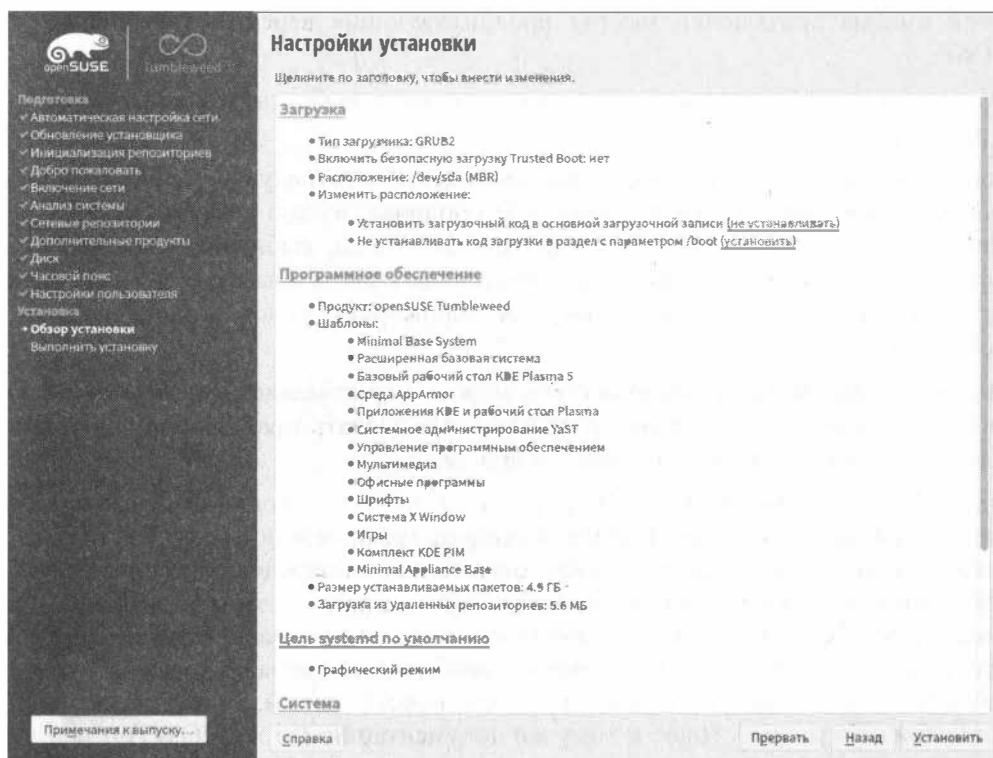


Рис. 2.19. openSUSE 15.1: выбор приложений

<sup>1</sup> В Ubuntu есть возможность выбора типа установки: обычный и минимальный. В первом случае будет установлен набор пакетов по умолчанию — все необходимые для работы системы программы, во втором — только браузер и минимально необходимый набор программ.

Может быть, если устанавливаешь рабочую станцию или домашний компьютер, такое решение себя оправдывает. Но когда планируешь создать 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.

## 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 GNOME (начиная с версии 18.04, разработчики вновь вернулись к GNOME, отказавшись от оболочки Unity, продвигаемой ими на протяжении многих предшествующих версий), в Fedora — тоже GNOME.

И только openSUSE позволяет пользователю самому выбрать графическую оболочку (рис. 2.20).

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 маломощных компьютеров можно порекомендовать оболочки LXDE или Xfce — даже если инсталлятор не позволяет сделать такой выбор при установке системы, вы можете доустановить их 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 достаточно сложно и дорого. Плюс к тому же документацию (те же HOWTO) при установке одной среды можно написать только лишь для нее. Ведь HOWTO, допустим, по настройке кодеков для GNOME, для KDE должен будет содержать совсем иную 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 пра-

вильно 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-alternativ-rabochemu-stolu-unity.html>.

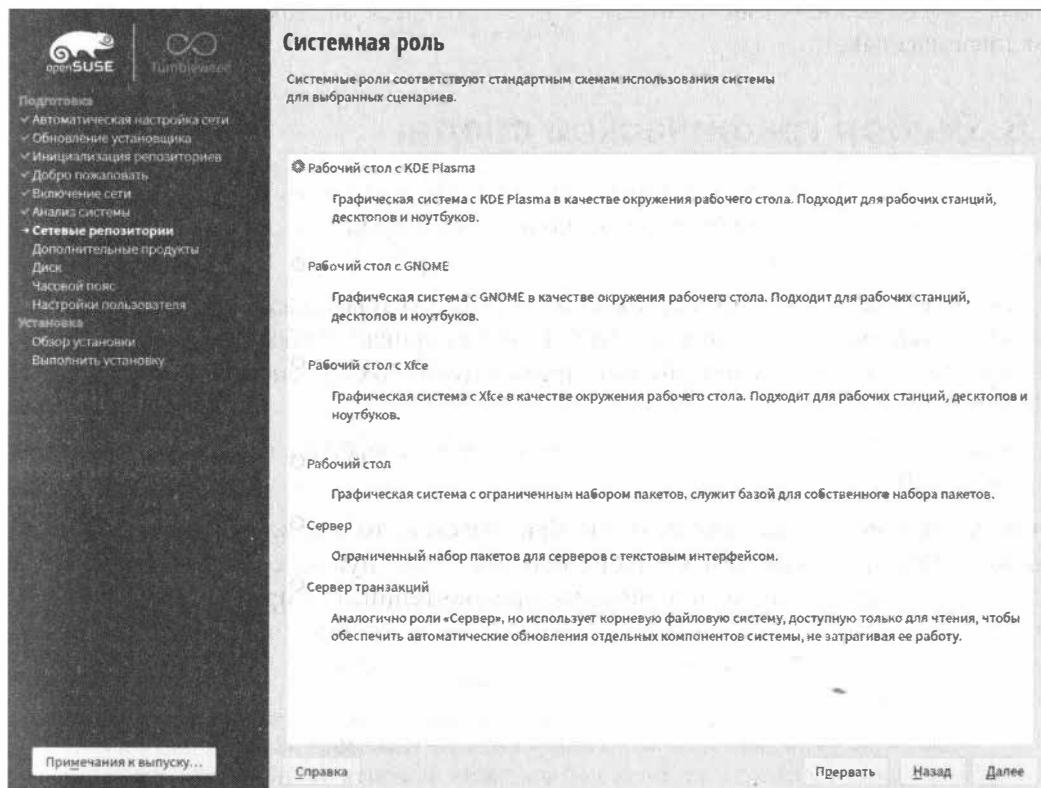


Рис. 2.20. openSUSE: выбор графической среды при установке системы

## 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-таки удобно, но 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).

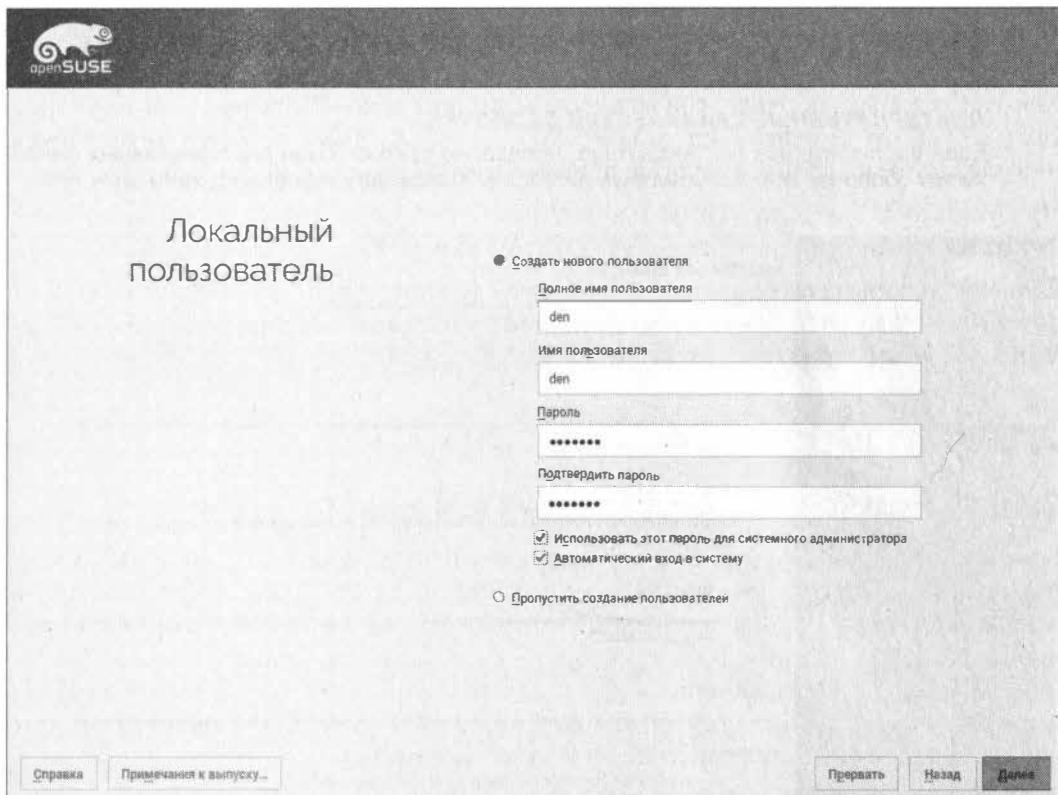


Рис. 2.21. openSUSE 15.1: создание пользователя и задание пароля root

### **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 этих данных? 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 пользователя `root`, поэтому вы будете использовать именно эту созданную при установке учетную запись обычного пользователя.

## 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 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. Но 10 компьютеров подряд — это уже более пяти 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 описывать создание полноценного PXE-сервера мы не станем, но кое-какие полезные сведения приведем.

### 2.10.2. Подготовка загрузочного сервера

Настройку загрузочного сервера мы покажем на примере Ubuntu. Поскольку установка по сети весьма специфическая операция и она нужна далеко не всем пользователям, то я невижу особой необходимости описывать установку PXE-сервера

в разных дистрибутивах, ведь в другом дистрибутиве можно все сделать «по образу и подобию» рассмотренного здесь.

## Установка DHCP-сервера

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

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

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

```
host pxeinstall {  
    hardware ethernet 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/dhcpcd 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 = -l /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 zxf 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 перепрограммировать 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 весьма часто, и 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 + timer doesn't work! Boot with apic=debug  
and send a report. 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 по-

верхностью 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, передайте 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 наличии 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 normally.

### 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

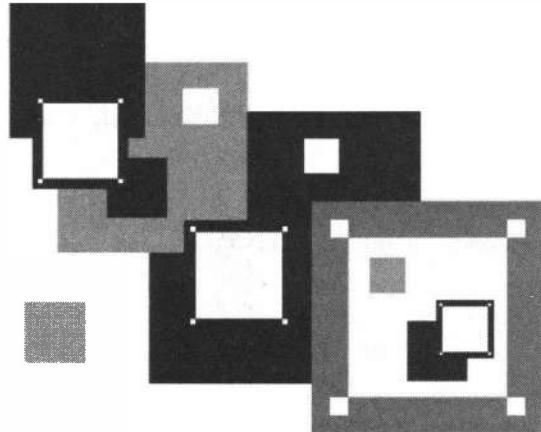
На компьютере с процессором AMD64 ядро переходило в режим паники, и установить Linux было невозможно. При этом на экране красовалось следующее сообщение:

```
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.



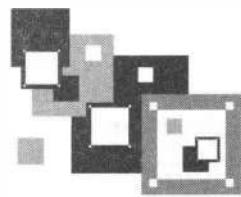
## ЧАСТЬ II

# 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).

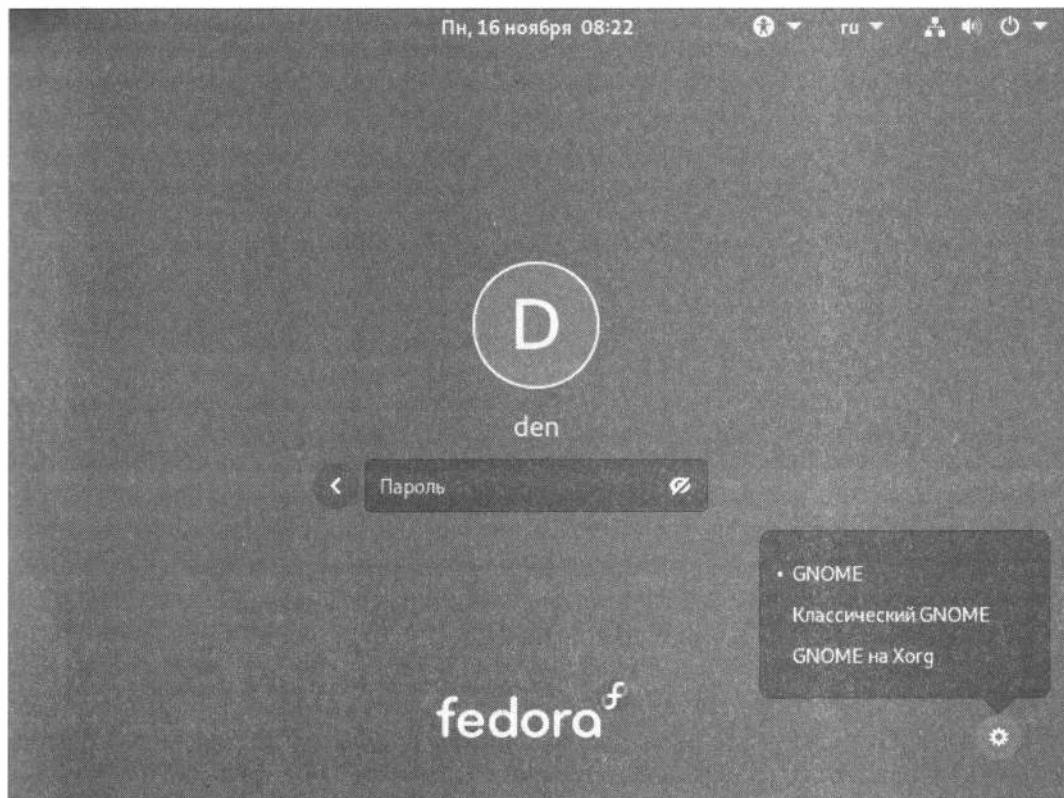


Рис. 3.1. Fedora 33: графический вход в систему

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. В современных версиях openSUSE нужный список расположен в нижнем левом углу экрана.

### **ДИСТРИБУТИВ SLACKWARE — ИСКЛЮЧЕНИЕ**

Однако из всех правил могут быть исключения. Пример тому дистрибутив Slackware — в нем сначала придется выполнить вход в консоли (рис. 3.2), а потом для запуска графического интерфейса ввести команду `startx`.

Забегая немного вперед, отмечу, что **консольный режим** (см. разд. 3.5), несмотря на свой столь устрашающий вид, оказывается весьма полезен в практической работе с Linux, в чем мы впоследствии не раз убедимся.

```
Polling for DHCP server on interface eth0:
dhcpcd: MAC address = 00:0c:29:6f:40:83
Starting Internet super-server daemon: /usr/sbin/inetd
Starting OpenSSH SSH daemon: /usr/sbin/sshd
Starting ACPI daemon: /usr/sbin/acpid
Starting system message bus: /usr/bin/dbus-uuidgen --ensure ; /usr/bin/dbus-dae
mon --system
Starting HAL daemon: /usr/sbin/hald --daemon=yes
ALSA warning: No mixer settings found in /etc/asound.state.
    Sound may be muted. Use 'alsamixer' to unmute your sound card,
    and then 'alsactl store' to save the default ALSA mixer settings
    to be loaded at boot.
Loading OSS compatibility modules for ALSA.
Loading /usr/share/kbd/keymaps/i386/qwerty/us.map.gz
Starting gpm: /usr/sbin/gpm -m /dev/mouse -t ps2

Welcome to Linux 2.6.21.5-smp (tty1) ~
dhsilabs login: root _____ имя пользователя
Password: _____ пароль при вводе не отображается
Linux 2.6.21.5-smp.
Last login: Mon Mar  3 13:21:39 +0300 2008 on tty1.
You have mail.
root@dhsilabs:~#
```

Рис. 3.2. Slackware: регистрация в консоли

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.

**Таблица 3.1. Клавиши переключения между консолями и графическим режимом**

Комбинация клавиш	Предназначение
<Ctrl>+<Alt>+<Fn> (n — от 1 до 6)	Переключение из графического режима в консоль с номером n
<Alt>+<Fn> (n — от 1 до 6)	Переключение между консолями
<Alt>+<F7>	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 позволялось 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 выполне-

ния 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. 3.3).

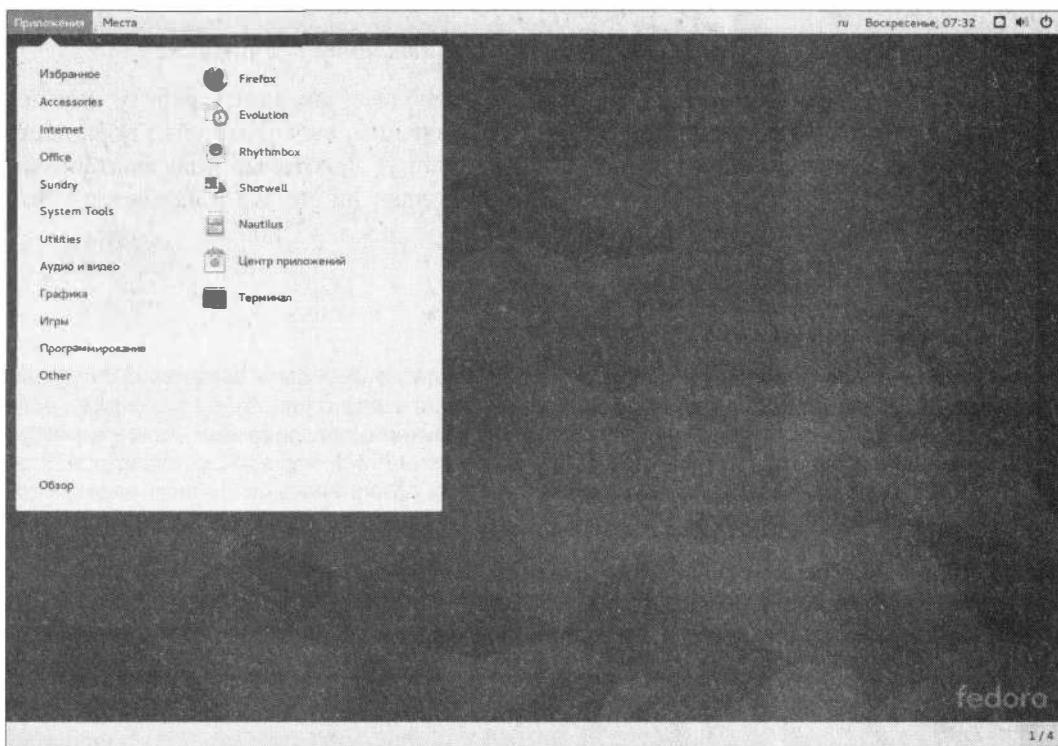


Рис. 3.3. Классический GNOME (скриншот из какой-то старой версии Fedora)

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 GNOME — Unity, которая, начиная с 11-й версии Ubuntu, была принята в этой сис-

теме по умолчанию. Как можно видеть, в верхней части экрана Unity находится меню приложений, а всевозможные кнопки быстрого доступа к тем или иным возможностям расположены на ленте слева (рис. 3.4).

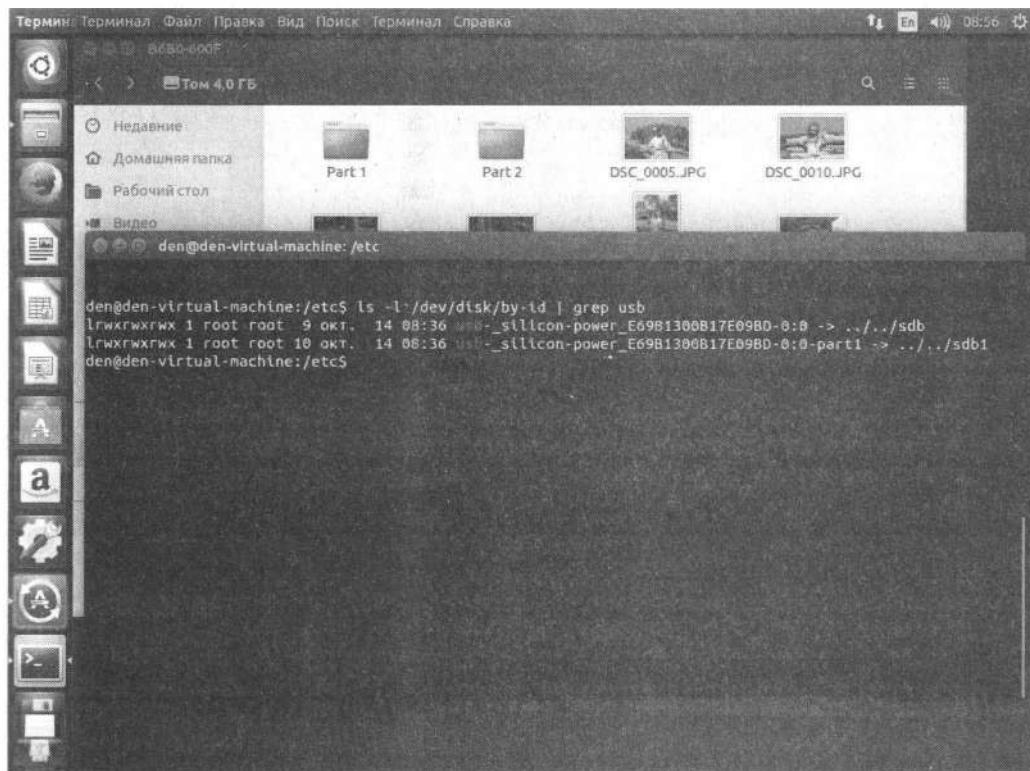
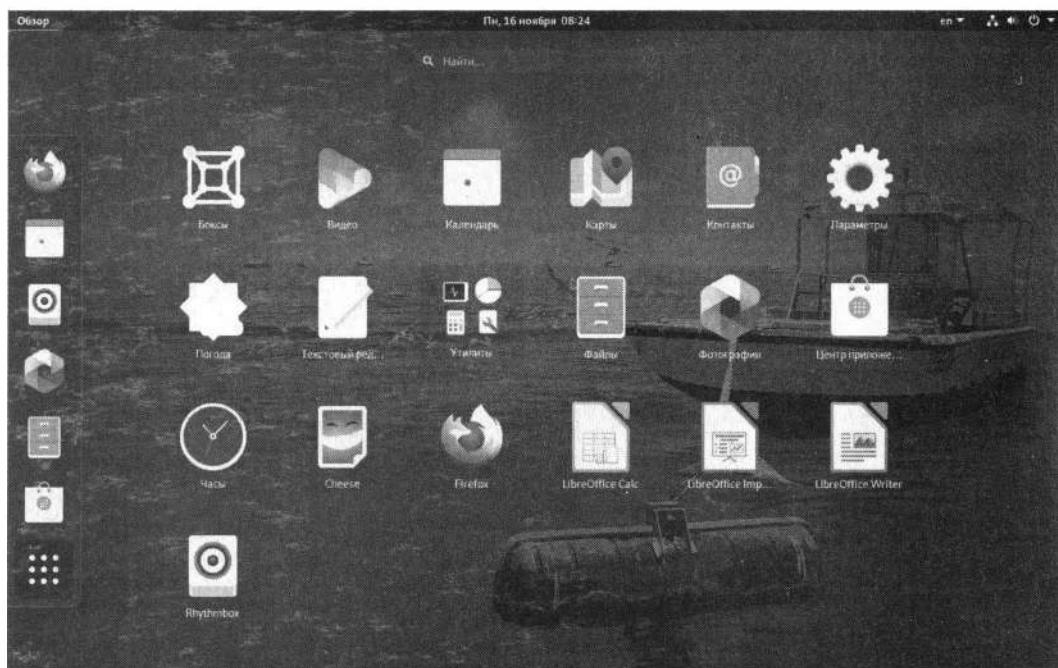


Рис. 3.4. Ubuntu 17.04: оболочка Unity

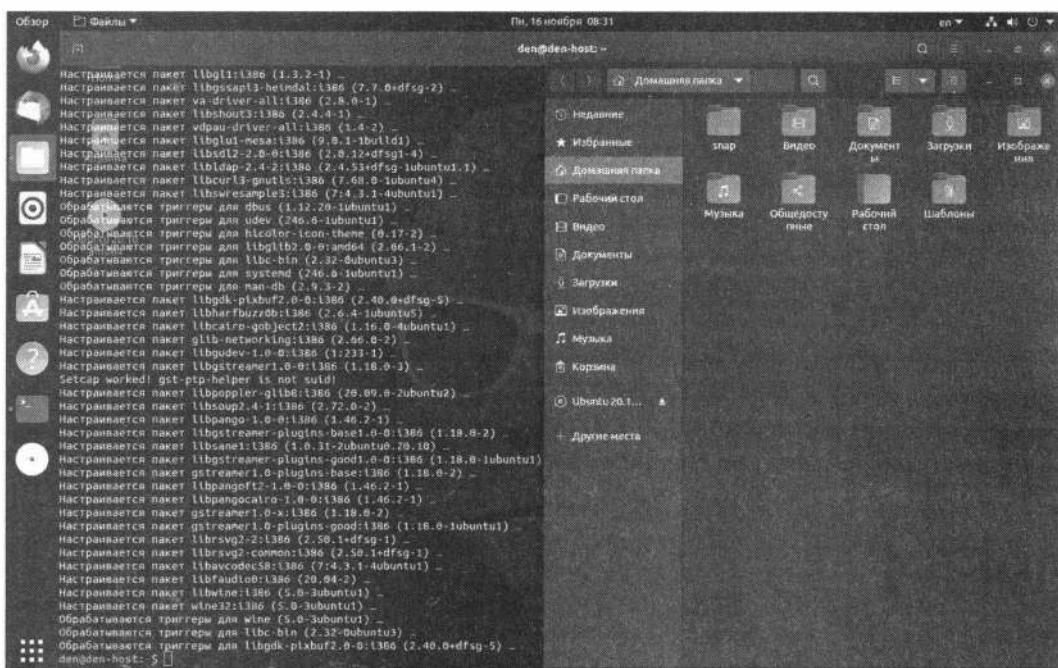
Оболочка Unity многим пришлась по вкусу, и, начиная с версии 3.8, графическая среда GNOME предоставляет теперь на выбор два интерфейса: и классический, и современный,— не правда ли, современный интерфейс GNOME в Fedora 33 (рис. 3.5) и Unity в Ubuntu (см. рис. 3.4) похожи как две капли воды?

Вот только в 2017 году разработчики Ubuntu отказались от собственной оболочки Unity в пользу среды GNOME. Таково было решение инвесторов Canonical, которые прекратили финансирование разработки Unity 8. Поэтому в Ubuntu, начиная с версии 17.10, в качестве графического интерфейса используется среда GNOME (рис. 3.6).

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. А что касается красоты, то современные версии GNOME, особенно в Ubuntu, выглядят ничем не хуже KDE.



**Рис. 3.5. Fedora 33: современный интерфейс GNOME**



**Рис. 3.6.** Ubuntu 20.10: среда GNOME 3.38

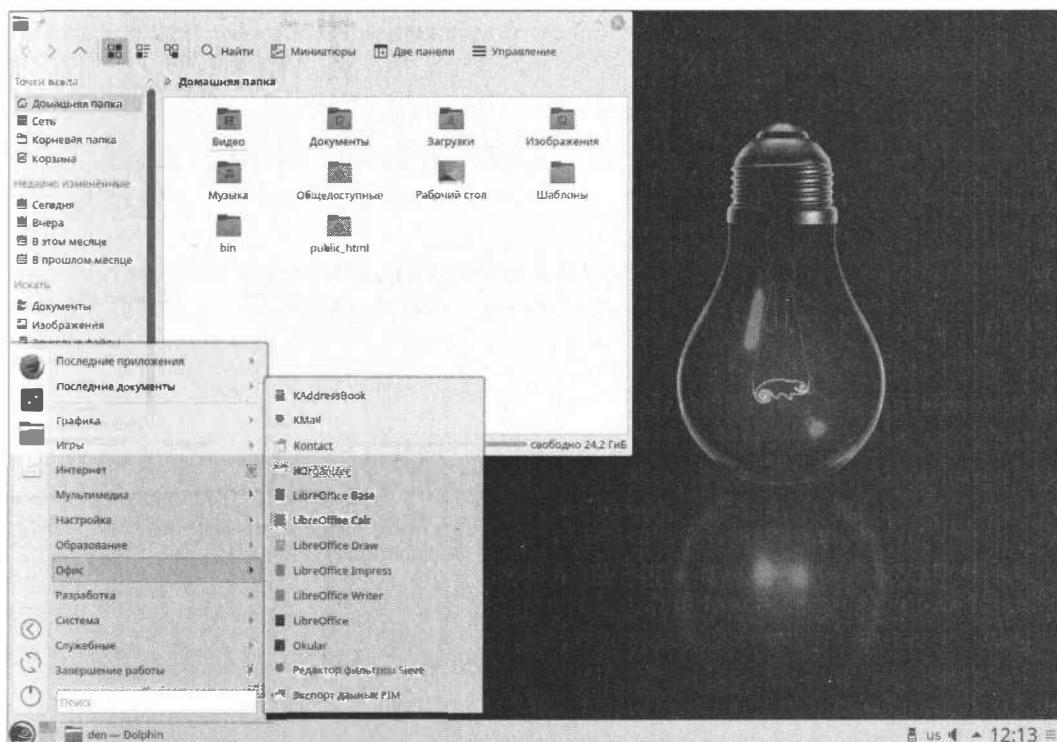


Рис. 3.7. openSUSE: рабочий стол KDE 5

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.

### 3.2.2. Установка альтернативного графического интерфейса

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

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

#### **Выполнение команд 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, 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 на нем 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 (рис. 3.10) вы найдете значок изменения языка ввода , а также значки изменения уровня громкости , завершения работы  и др. 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),

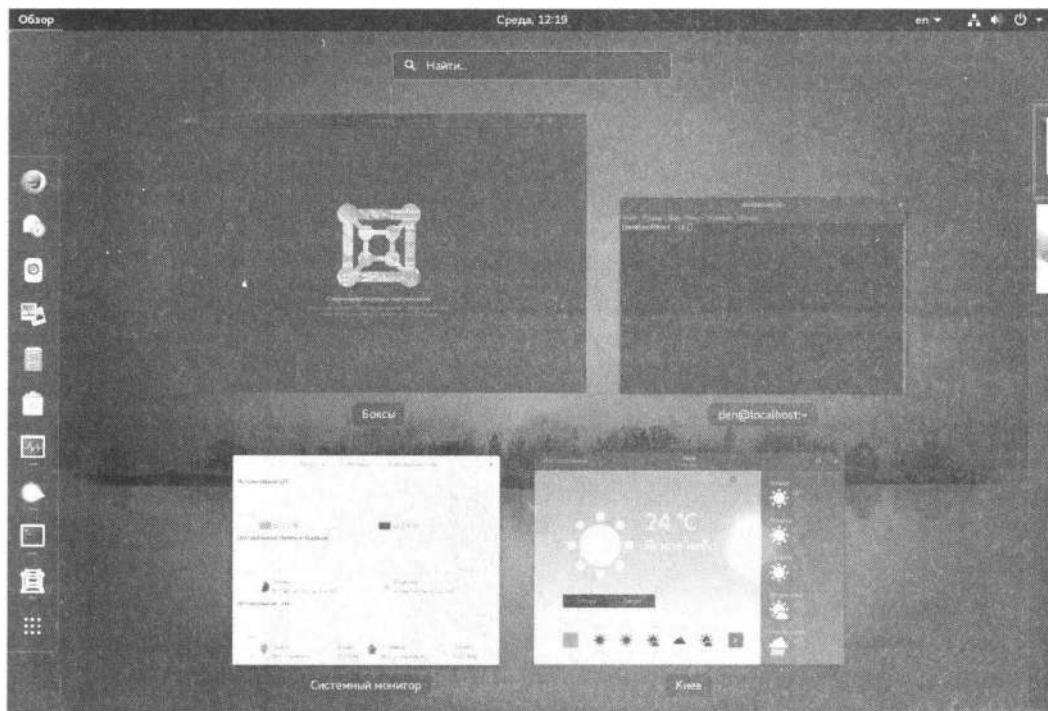


Рис. 3.8. Интерфейс GNOME



Рис. 3.9. Интерфейс GNOME: список установленных приложений

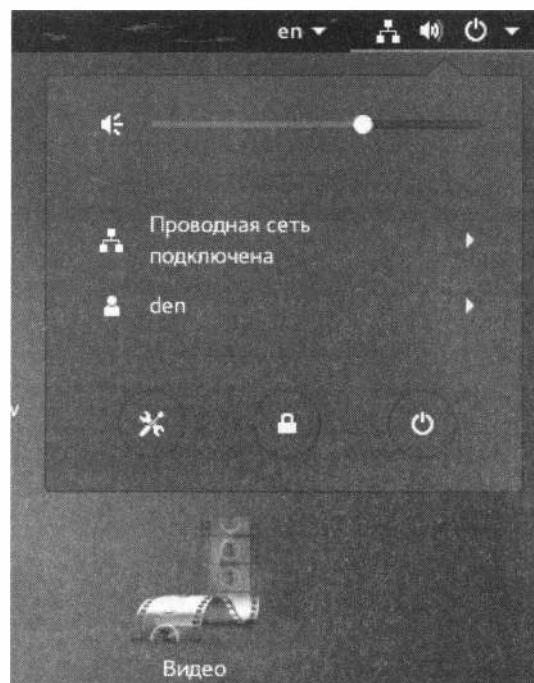


Рис. 3.10. Интерфейс GNOME: изменение громкости, завершение работы и многое другое

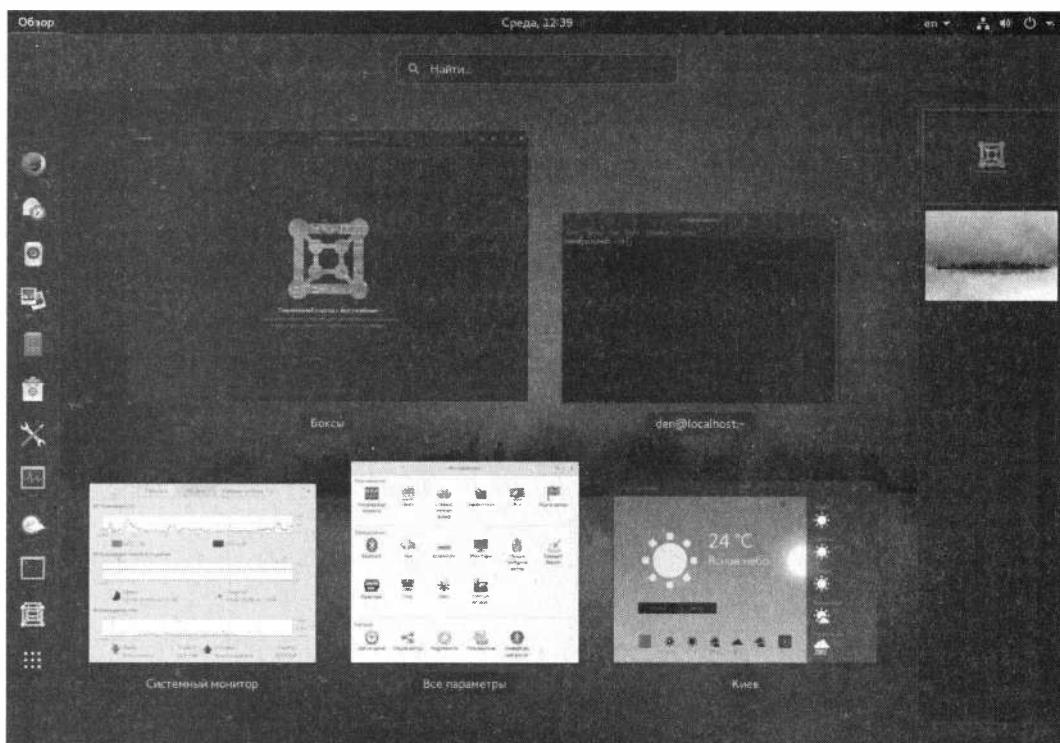


Рис. 3.11. Интерфейс GNOME: панель рабочих столов (справа)

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 заполнении текущего рабочего стола. Так, при запуске системы у вас будет единственный рабочий стол. Но как только вы откроете хотя бы одно приложение, автоматически создастся второй рабочий стол — пустой, и вы сможете переключиться на него с помощью панели рабочих столов. Как только на этом рабочем столе вы запустите хотя бы одно приложение, автоматически будет создан третий рабочий стол, и т. д. Учитывая, что каждый рабочий стол занимает системные ресурсы и в первую очередь оперативную и видеопамять, не перестарайтесь!

## 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 нижняя на левой панели), далее выберите приложение **Параметры** — с изображением шестеренки (рис. 3.12). В открывшемся окне перейдите в раздел **Конфиденциальность** (рис. 3.13) и выключите параметр **Автоматическая блокировка экрана**.

Однако спешу вас разочаровать: да, вам не придется более вводить пароль после каждой блокировки экрана, но тем не менее вы по-прежнему будете, отойдя на минутку от компьютера и вернувшись, 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. 3.14), выполните в терминале команду:

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

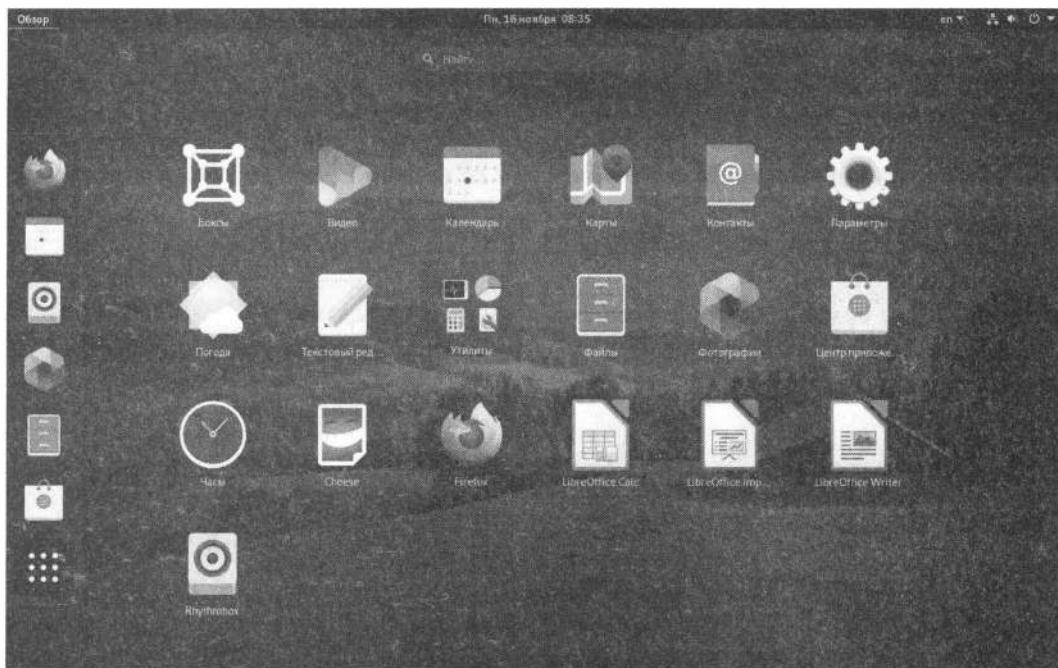


Рис. 3.12. Fedora 33: доступ к параметрам системы

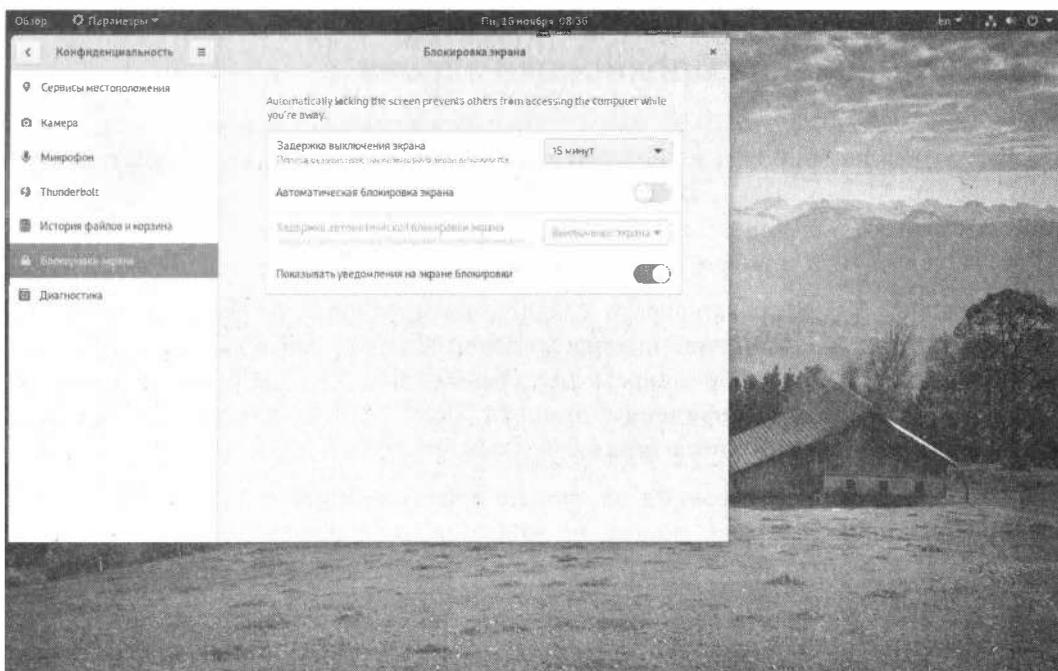


Рис. 3.13. Fedora 33: параметры конфиденциальности

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. 3.15).

den@fedora:~

Результат транзакции

Установка 1 Пакет

Объем загрузки: 676 К  
Объем изменений: 2.8 М  
Продолжить? [д/н]: у  
Загрузка пакетов:  
dconf-editor-3.38.0-1.fc33.x86\_64.rpm 1.8 MB/s | 676 KB 00:00

Общий размер 567 kB/s | 676 kB 00:01

Проверка транзакции  
Проверка транзакции успешно завершена.  
Идет проверка транзакции  
Тест транзакции проведен успешно  
Выполнение транзакции  
Подготовка : 1/1  
Установка : dconf-editor-3.38.0-1.fc33.x86\_64 1/1  
Запуск скриплета: dconf-editor-3.38.0-1.fc33.x86\_64 1/1  
Проверка : dconf-editor-3.38.0-1.fc33.x86\_64 1/1

Установлен:  
dconf-editor-3.38.0-1.fc33.x86\_64

Выполнено!  
[den@localhost ~]\$

Рис. 3.14. Fedora 33: установка `dconf-editor` — редактора конфигурации GNOME

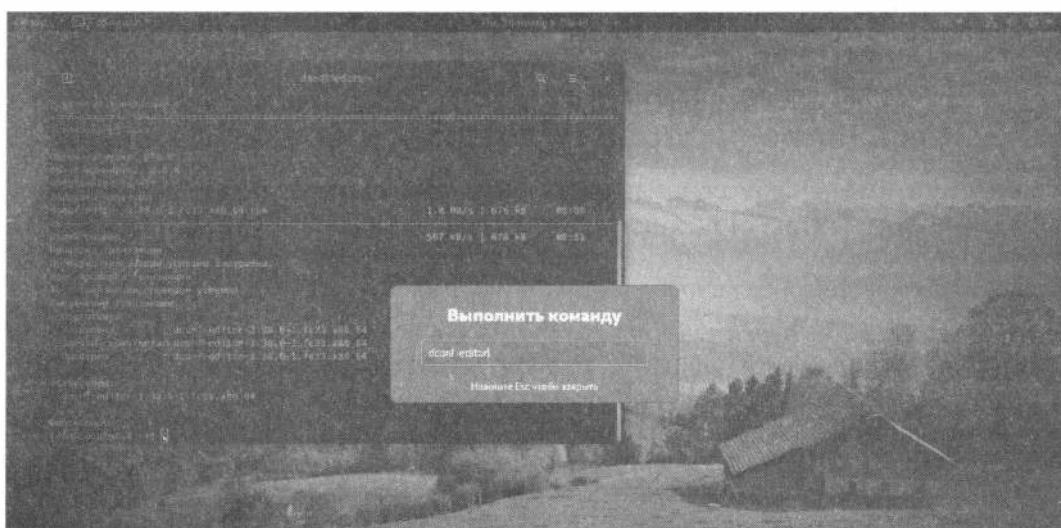


Рис. 3.15. Fedora 33: запуск `dconf-editor`

Осталось только найти опцию **disable-lock-screen** и перевести ее переключатель в положение «включено» (рис. 3.16).

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

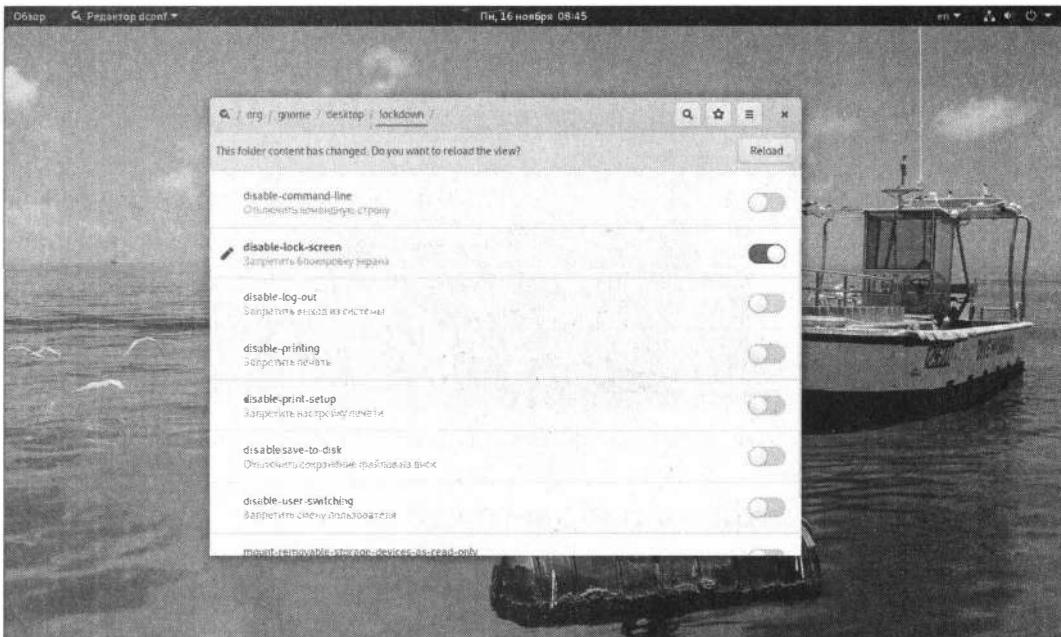


Рис. 3.16. Fedora 33: полное отключение блокировки экрана

В Ubuntu блокировка экрана отключается аналогично (ранее процедура ее отключения была несколько иной — ведь использовалась Unity, теперь же, когда Ubuntu перешла на GNOME, порядок действий такой же).

### 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 **Комбинации клавиш**, щелкнуть на комбинации клавиш, которую вы хотите изменить, а затем на панели **Установить комбинацию клавиш** задать требуемую комбинацию клавиш для переключения источника ввода (рис. 3.17). Ранее комбинация клавиш устанавливалась путем выбора из списка, сейчас же надо просто нажать нужную вам комбинацию.

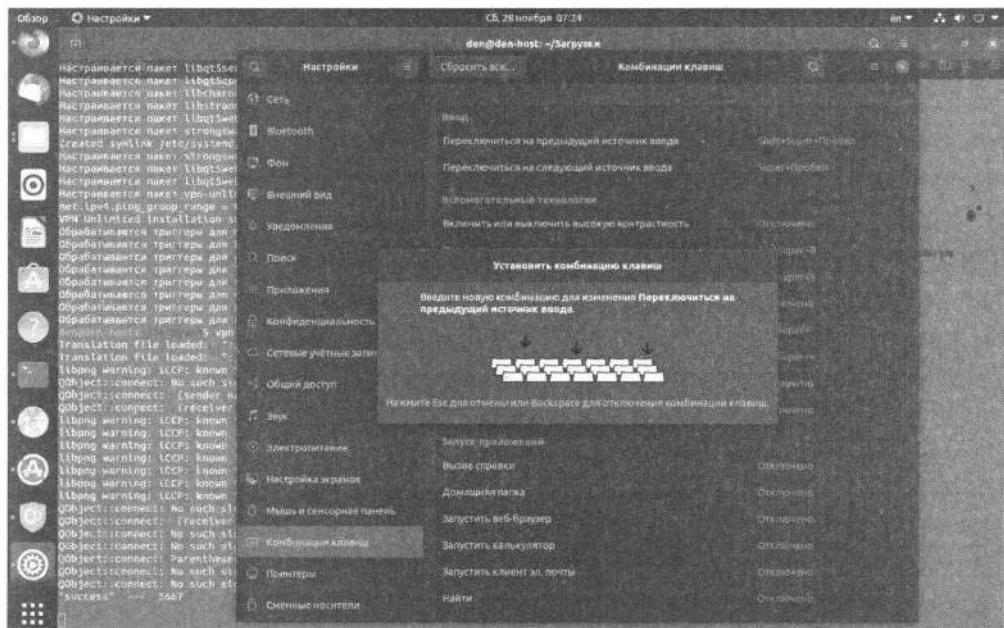


Рис. 3.17. Ubuntu 20.10: изменение способа переключения языков ввода

### 3.3.3. Изменение фона рабочего стола

Изменение *фона рабочего стола* производится в разделе **Фон** (рис. 3.18). Здесь можно изменить как фон рабочего стола, так и фон экрана блокировки.

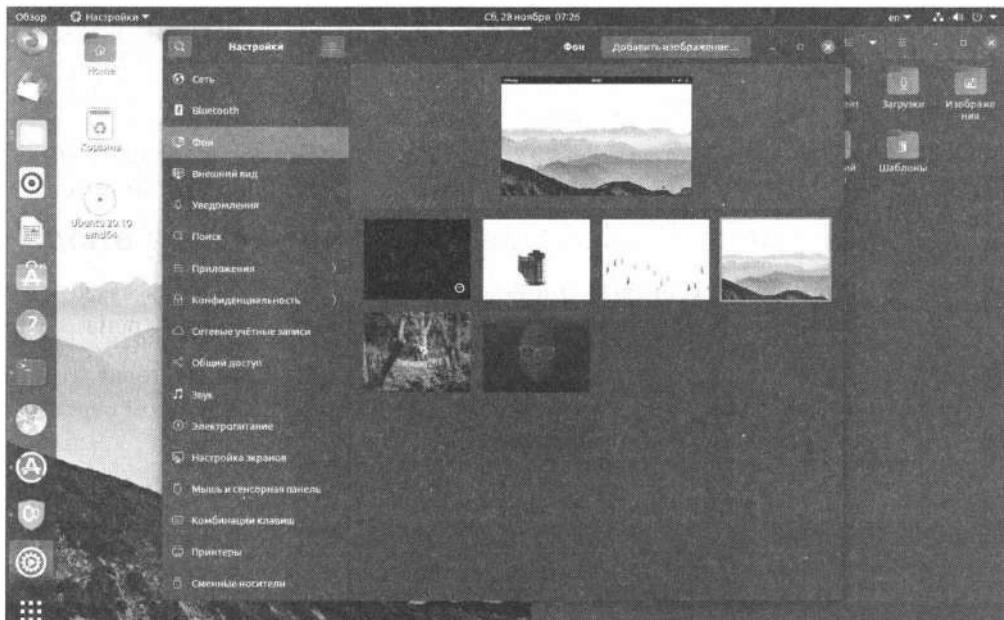


Рис. 3.18. Ubuntu: изменение фона рабочего стола

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

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

- <Ctrl>+<Alt>+<Del> — 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 Unmounting... OK*;
- <Alt>+<SysRq>+<B> — практически эквивалентна 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 весьма 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 мы 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` (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 `man`-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-либо 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. 3.14*), 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
```

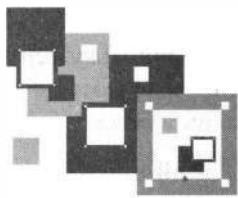
Команда `rm` запросит подтверждение удаления (нужно было бы нажать клавишу `<Y>`), но за нас это сделает команда `echo`.

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, Btrfs, 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 (использовался 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 его 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 стablyно.

□ **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 про-

изводительность (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 Server*, тогда как в *Fedora 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 (начиная с версии 13.2) — вместо проверенной годами ext4. А, начиная с версии 33, к ним присоединились и разработчики Fedora. Ключевые особенности Btrfs: сжатие данных, оптимизированный для твердотельных дисков (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.

Узнать тип файловой системы того или иного раздела/устройства можно с помощью команды *file*, например:

```
sudo file -s /dev/sda1
```

Вывод этой команды будет примерно таким:

```
/dev/sda1: Linux rev 1.0 ext4 filesystem data, UUID=3762b167-9de0-4124-b7c1-46d4a2fc2019 (extends) (64bit) (large files) (huge files)
```

Как видно из приведенного вывода, устройство `/dev/sda1` использует файловую систему `ext4`.

Команда `df -T` пригодится, если вы не знаете точно, как называется ваше устройство, или вам лень вводить его имя:

```
root@hosting:/srv/www/htdocs# df -T
Filesystem     Type      1K-blocks   Used   Available   Use%   Mounted on
Смонтировано в
/dev/sda1       ext4      30829600  21994740  7434548    75%   /
udev           devtmpfs  16460416        4  16460412    1%   /dev
tmpfs          tmpfs     3294256   3088  3291168    1%   /run
none           tmpfs     5120        0   5120      0%   /run/lock
none           tmpfs     16471276        0  16471276    0%   /run/shm
none           tmpfs     102400        0  102400    0%   /run/user
/dev/sdb         ext4      41153856  35731076  3309244   92%   /srv
/dev/sdc1       ext4      82438800  62819492  15408624   81%   /var
/dev/sdd2       reiserfs  45087324  6462812  38624512   15%   /media/reiser-hdd
```

Эта команда выводит информацию о смонтированных файловых системах, в том числе указывает тип каждой файловой системы. Показанные в выводе команды файловые системы `devtmpfs` и `tmpfs` — не совсем файловые системы в привычном понимании этого термина. Они представляют собой временные хранилища, размещаемые в оперативной памяти, а не на физическом накопителе, то есть это так называемые *RAM-диски* (диски в оперативной памяти).

#### 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;
- ◻ `Btrfs` также должна неплохо подойти для современных серверов — она оптимизирована под `SSD`, поддерживает снапшоты и сжатие. Вот только последняя ее характеристика заставляет задуматься о производительности. Впрочем, об этом мы еще поговорим в разд. 4.14;
- ◻ 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 Windows — 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 разделения 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 и 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 «увели» 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 (`ttyS0` соответствует COM1, `ttyS1` — 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. Откройте 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.

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 /`.

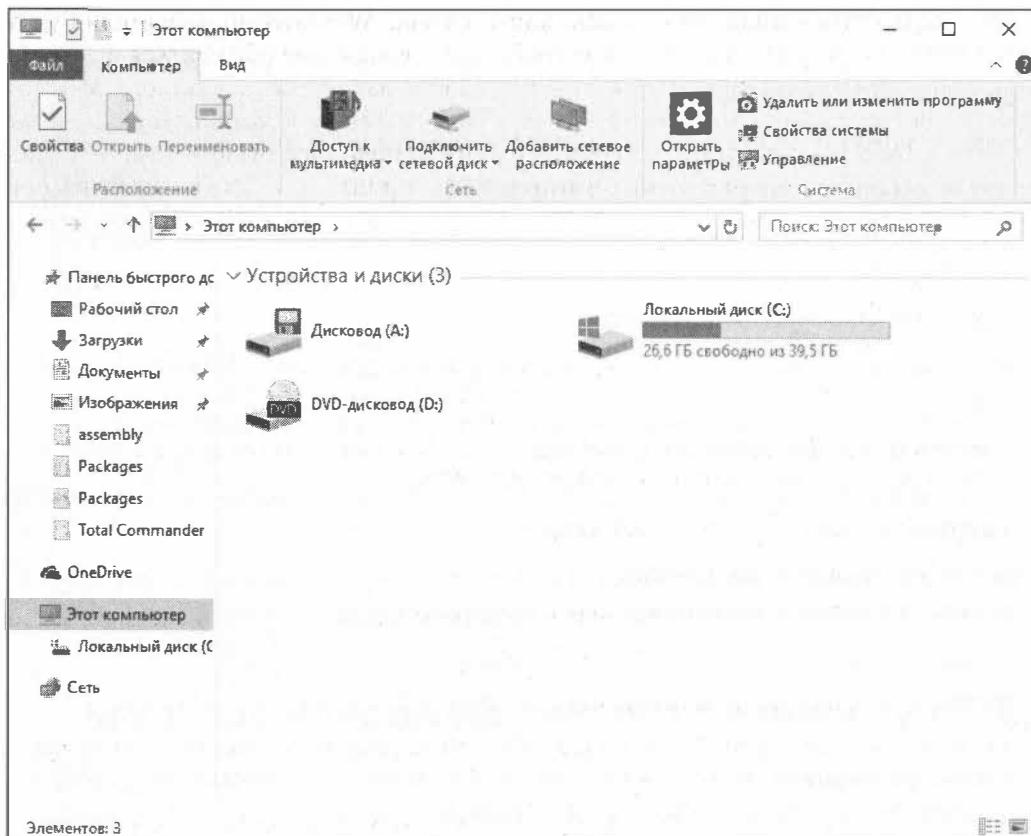


Рис. 4.1. Проводник в Windows 10

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 ей имя файла, а на какой файловой системе расположен этот файл — дело третье.

Рассмотрим схему архитектуры файловой системы (рис. 4.2): верхних два ее 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, —



Рис. 4.2. Архитектура файловой системы

например `open()`, `read()`, `write()`, `close()`, — можно немного повысить производительность приложения — ведь вы минуете `glibc`, которая все равно использует те же системные вызовы.

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 (напомню, 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 &lt;файл&gt;</code>	Создает пустой файл
<code>cat &lt;файл&gt;</code>	Просмотр текстового файла
<code>tac &lt;файл&gt;</code>	Вывод содержимого текстового файла в обратном порядке, т. е. сначала выводится последняя строка, потом предпоследняя и т. д.
<code>cp &lt;файл1&gt; &lt;файл2&gt;</code>	Копирует файл <code>&lt;файл1&gt;</code> в файл <code>&lt;файл2&gt;</code> . Если <code>&lt;файл2&gt;</code> существует, программа попросит разрешение на его перезапись
<code>mv &lt;файл1&gt; &lt;файл2&gt;</code>	Перемещает файл <code>&lt;файл1&gt;</code> в файл <code>&lt;файл2&gt;</code> . Этую же команду можно использовать и для переименования файла
<code>rm &lt;файл&gt;</code>	Удаляет файл
<code>locate &lt;файл&gt;</code>	Производит быстрый поиск файла
<code>which &lt;программа&gt;</code>	Выводит каталог, в котором находится программа, если она вообще установлена. Поиск производится в каталогах, указанных в переменной окружения PATH (это путь поиска программ)
<code>less &lt;файл&gt;</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 клавишу `<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 `s14`, `sqt`, `srg` 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 &lt;каталог&gt;</code>	Создание каталога
<code>cd &lt;каталог&gt;</code>	Изменение каталога
<code>ls &lt;каталог&gt;</code>	Вывод содержимого каталога
<code>rmdir &lt;каталог&gt;</code>	Удаление пустого каталога
<code>rm -r &lt;каталог&gt;</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
```

```
mkdir 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 символ «тильда» (`~`) — так обозначается *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 здесь 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` служит 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 *In*

### 4.5.1. Жесткие и мягкие ссылки

Файлы и каталоги физически хранятся на носителе 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:

```
In: 'link': hard link not allowed for directory
In: '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 отличие 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 den 6 июл 15 08:46 file
-rw-rw-r-- 2 den den 6 июл 15 08:46 hardlink
1rwk1wxrwx 1 den den 4 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 -i`, которая показывает индексные дескрипторы файлов:

```
ls -i
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 файлов и жесткие ссылки

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

В предыдущем разделе приводился вывод команды `ls` с опцией `-i`. Посмотрите также на вывод команды `ls -l` — вторая колонка показывает количество жестких ссылок на `inode`. По сути, в этом контексте нет особой разницы между ссылкой и файлом. Вы можете удалить файл `file`:

```
rm file
ls -l
итого 4
-rw-rw-r-- 1 den den 6 июл 15 08:46 hardlink
1rwk1wxrwx 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'? у
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 символьическая ссылка symlink, последняя превращается в «битую» ссылку — в выводе команды ls она отмечается красным цветом на черном фоне. Таким образом, символьическая 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 вы при необходимости можете 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---- 1 den group 300 Apr 11 11:11 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-подобных систем 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 перевести число 110 из двоичной системы 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 владельца — 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 `c` 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 `u` 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, выполнив команду: `man chattr`.

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

Для поиска файлов в Linux служит команда `find`. Это весьма 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. Команда также может выполнять операции над найденными файлами. Например, вы можете найти временные файлы и сразу удалить их.

Все опции команды `find` мы изучать не станем — это вы можете сделать самостоятельно с помощью команды `man find`. Здесь же мы приведем лишь несколько примеров ее использования:

- 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;

- найти все файлы (не каталоги — это определяет параметр `-type f`) и установить для них права 644 (поиск начинается с текущего каталога .):

```
find . -type f -exec chmod 0644 {} \;
```

Для поиска файлов, 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 команда `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 носитель, 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 ком-

пакт-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 накопители SCSI/SATA — `/dev/sdx` (где в обоих случаях `x` — буква).

После внедрения менеджера устройств `udev`<sup>1</sup> 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 современ-

---

<sup>1</sup> `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`.

ные 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/sda1` — 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*).

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 `r` и нажать клавишу `<Enter>` — вы увидите таблицу разделов (рис. 4.6). Для выхода из программы введите `q` и нажмите клавишу `<Enter>`.

На рис. 4.6 показана 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

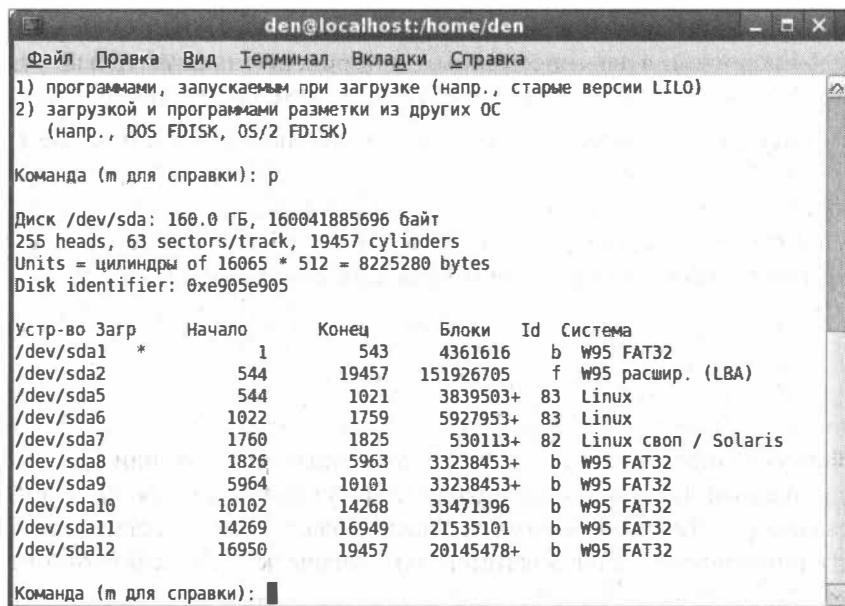


Рис. 4.6. Таблица разделов жесткого диска

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 (поддерживаются Windows 9x, ME, XP);
- `ntfs` — NT File System (поддерживаются 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`, то при попытке указания файловой системы NTFS будет выведено сообщение об ошибке. В таком случае вы можете скачать его с сайта [www.ntfs-3g.org](http://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 (дисковода, CD/DVD-привода, 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.

##### 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`).

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 root)
nouser	Файловую систему может монтировать только пользователь root. Используется по умолчанию
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>

Рассмотрим небольшой пример:

```
/dev/sdc /mnt/cdrom auto umask=0,user,noauto,ro,exec 0 0
/dev/sda1 /mnt/win_c vfat umask=0,utf8 0 0
```

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

- 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. Далее идет достаточно 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 исполняемые файлы. По-

нятно, 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 `1f049af9-2bdd-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 разбираясь 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 перемычки

```
den@localhost:~$ ls -l /dev/disk/by-uuid
итого 0
lrwxrwxrwx 1 root root 10 Фев 12 15:25 1c3b8bd3-c26f-449e-9eba-8f254fefef814 -> ./../sda1
lrwxrwxrwx 1 root root 10 Фев 12 15:25 3106fa17-65ef-42e9-a1d4-2313daee96b5 -> ./../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$ sudo blkid
/dev/sda1: UUID="0cfcfdfc-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`

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) 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 -1F /dev/disk/by-label
```

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

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

**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>

#### 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 флашечки), построенные с использованием флеш-памяти, практически вытеснили обычные дискеты — они очень компактны и позволяют хранить весьма значительные объемы информации. Сегодня никого не удивишь небольшим брелоком, вмещающим 16–32 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 ожидал — во всяком случае, они выглядят настоящими спринтерами на фоне обычных дисков.

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 в 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 носитель данных — 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 флашечки 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

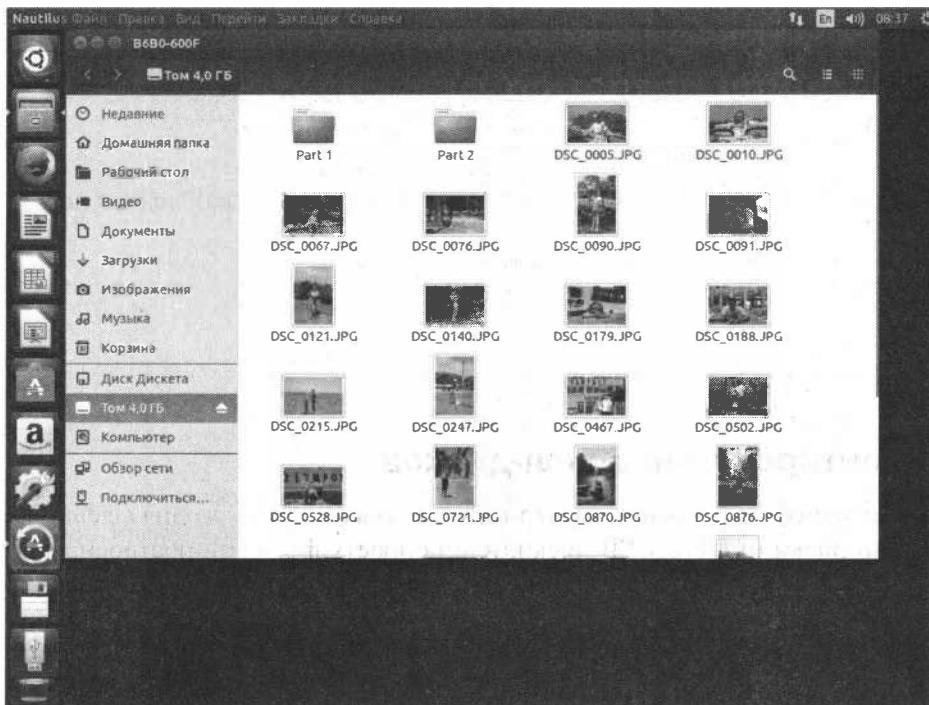


Рис. 4.8. Ubuntu: содержимое подключенной флешки с файлами

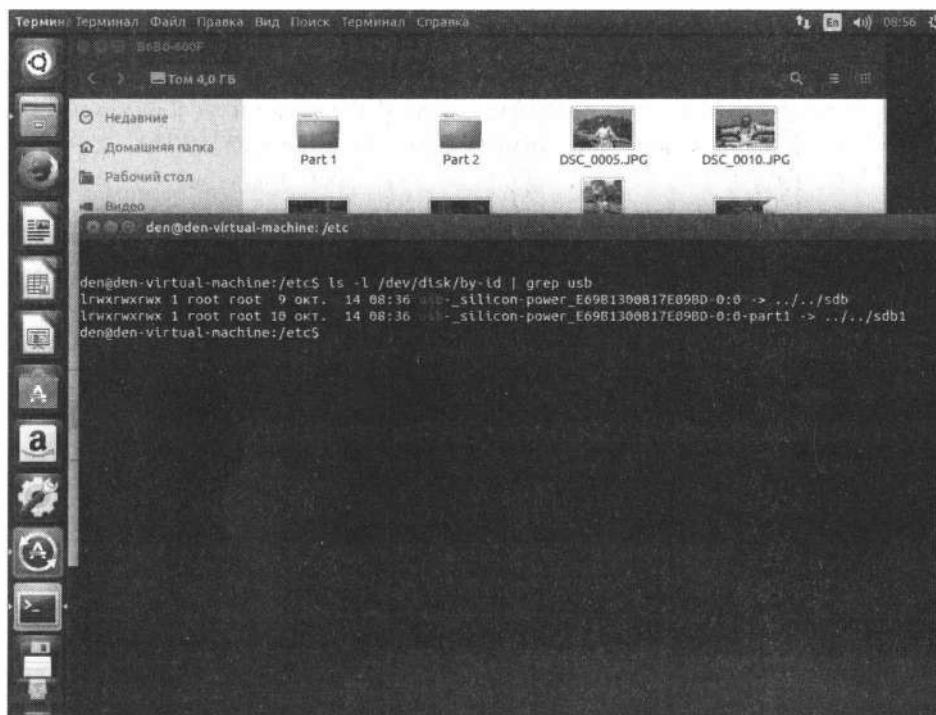


Рис. 4.9. Ubuntu: USB-диск найден

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
```

## 4.8. Настройка журнала файловой системы ext3/ext4

Журнилируемая 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 эксабайт). Максимальный размер файла — более 2 Тбайт. Ресурс Phoronix ([www.phoronix.com](http://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 по производительности 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	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. В случае с ext4 максимальный размер тома составляет 1 эксабайт (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
Экстенты	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. Для улучшения выделения ресурсов и более эффективной организации данных в ext4 были введены <b>экстенты</b> . Экстент — это способ представления непрерывной последовательности блоков памяти. Для эффективного представления маленьких файлов в экстентах применяется уровневый подход, а для больших файлов используются деревья экстентов. 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

**Таблица 4.6 (окончание)**

Особенность	Комментарий
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
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 до сих пор используете 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. Программы для разметки диска

### 4.10.1. Стандартная программа fdisk

Проще всего для разметки диска воспользоваться стандартной программой `fdisk`, которая имеется во всех дистрибутивах Linux.

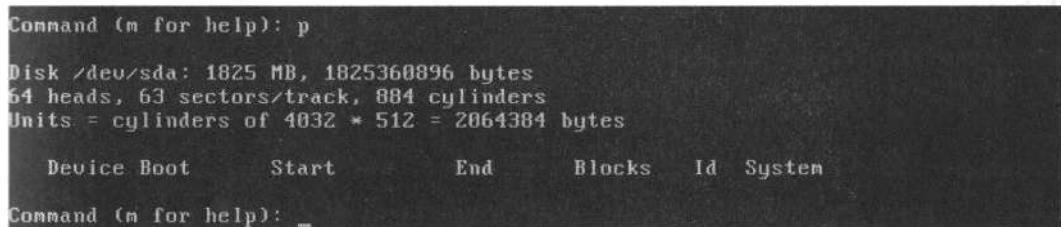
Введите команду (можно использовать короткие имена):

```
= 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 `?`, 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>M`. После числа должна идти именно буква `M`, иначе размер будет воспринят в байтах, а нам нужны мегабайты, — например, если вы хотите создать раздел размером 10 Гбайт, то введите `+10240M`.

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  Venix 80286    84  OS/2 hidden C:  c6  DRDOS/sec (FAT-
5  Extended       41  PPC PReP Boot   85  Linux extended c7  Syrinx
6  FAT16          42  SFS             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 < 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  BSD/OS        fd  Linux raid auto
18 AST SmartSleep  65  Novell Netware b8  BSD/OS 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

## 4.10.2. Графическая программа для разметки диска GParted

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.

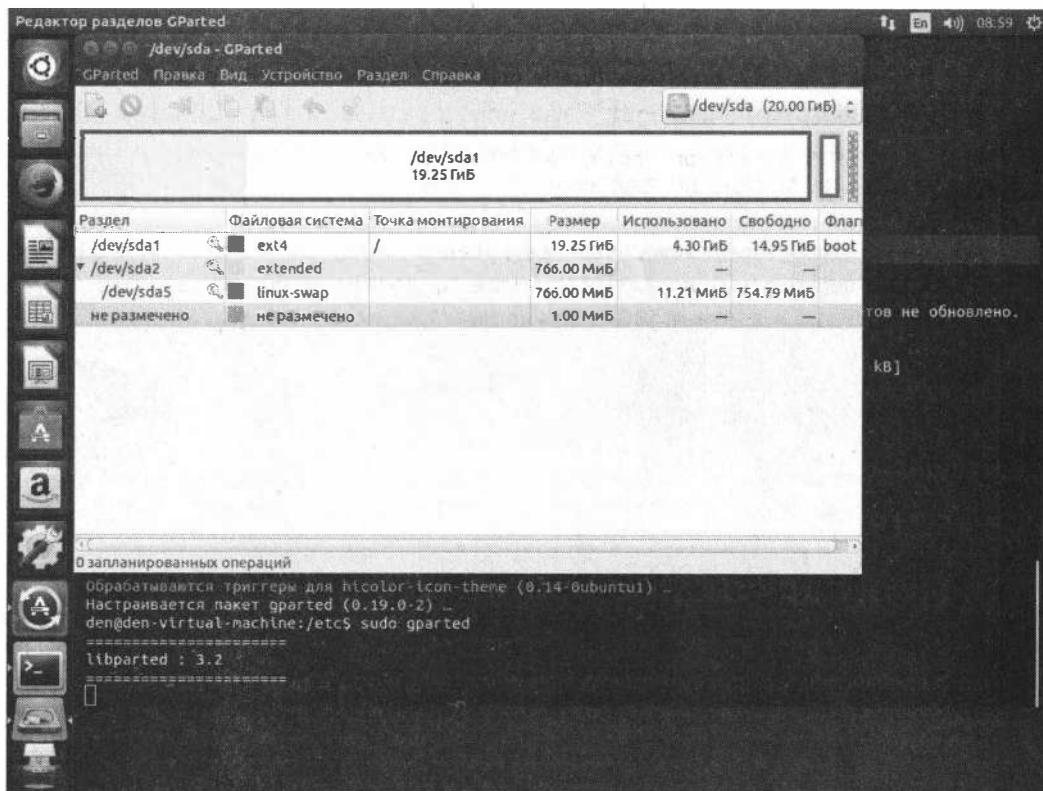


Рис. 4.16. Ubuntu: программа GParted

## 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 предло-

жен компанией Intel на смену BIOS. В EFI таблица GPT решает те же задачи, что в BIOS решает 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 вместо устаревшей системы CHS (Цилиндр-Головка-Сектор), которая применялась в MBR, использована современная система адресации логических блоков (LBA). 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  $9,4 \times 10^{21}$  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>.

## **4.12. Несколько слов о CD/DVD-дисках**

Постепенно CD/DVD-диски уходят в небытие — новые модели ноутбуков даже не оснащаются приводами для их чтения/записи. Это вполне понятно — емкость распространенных вариантов DVD-дисков не превышает 4,5 Гбайт, они не самые надежные и не самые быстрые.

Лет 10 назад DVD-диски с успехом использовались не только для записи фильмов, но и как носители для резервных копий. Но объемы информации растут, и емкости

обычного DVD-диска уже не хватает для хранения всей необходимой информации. Поэтому сейчас можно выделить два основных носителя «бэкапов»: внешние жесткие диски и облачные хранилища. На предприятиях также часто применяют сетевые хранилища (NAS)<sup>1</sup>, а при очень больших объемах информации — ленточные стримеры. Да, поскольку стоимость хранения информации на стримерах очень низка и если записанные данные не используются для оперативного доступа, то стримеры даже в 2019 году — вполне подходящий вариант.

Некоторое время DVD-диски «оставались на плаву» благодаря несовершенным аудиосистемам автомобилей, не поддерживающим флешки. Сейчас же редко можно встретить современный автомобиль без поддержки USB. Так что необходимость в оптических дисках отпала совсем.

Если же вам до сих пор нужно записывать CD/DVD-диски, вы можете воспользоваться программами K3B и Brasero, работа с которыми описана в главе 16 предыдущего издания этой книги. Интерфейс программ K3B и Brasero достаточно прост, но, если у вас возникнут сложности, материал этой главы (PDF-файл Лазерные диски и программы для их прожига) вы сможете найти в папке Дополнения сопровождающего книгу файлового архива (см. *приложение*) или по адресу: [https://dkws.org.ua/novice/pdf/ch\\_16.pdf](https://dkws.org.ua/novice/pdf/ch_16.pdf).

Программы для записи CD/DVD в консоли описаны в материале «*Особые операции при работе с файловой системой*», который также доступен в папке Дополнения сопровождающего книгу файлового архива (см. *приложение*) или по адресу: <https://dkws.org.ua/novice/pdf/fs.pdf>.

## 4.13. Scalpel — инструмент для восстановления файлов

Иногда понимаешь, что одна из последних команд `rm` была лишней. Конечно, лучшее средство восстановления случайно удаленных файлов — это бэкапы. Но бывает так, что резервная копия была создана раньше, чем файл появился в системе, или же с момента создания бэкапа в файлы было внесено много изменений, — например, утром менеджер загрузил фотографии, а администратор не разобрался и удалил их. Тогда вариант из резервной копии окажется неактуальным: бэкап как бы есть, но по факту — его нет. На помощь приходит восстановление удаленных файлов — и чем раньше вы приступите к этому процессу, тем выше вероятность успеха.

Для восстановления удаленных файлов существует много разных инструментов. Выбор инструмента зависит от типа файловой системы и типа самого удаленного файла. Так, при использовании файловой системы ext3/ext4 неплохие результаты показывает утилита `extundelete`. Если нужно восстановить изображение (JPEG/PNG/GIF и др.), лучшим выбором станет `foremost`. Универсальный же солдат — утилита `scalpel`.

---

<sup>1</sup> Если у вас современный роутер с поддержкой USB, то любой внешний жесткий диск можно превратить в NAS, подключив его к маршрутизатору. Можете не благодарить за сэкономленные деньги ☺.

Утилита служит для восстановления удаленных файлов с использованием базы данных заголовков. Перед запуском scalpel в ее конфигурационном файле нужно указать тип и заголовки файлов, которые необходимо восстановить. В этом конфигурационном файле по умолчанию уже прописаны заголовки самых популярных типов файлов — админу остается только раскомментировать соответствующие строки. Конечно, для восстановления экзотических форматов есть возможность указать и собственные заголовки.

Посмотрим, как работает scalpel на практике. Установим утилиту:

```
sudo apt install scalpel
```

Открываем конфигурационный файл `/etc/scalpel/scalpel.conf` или `/etc/scalpel.conf` и видим, что в нем уже прописаны различные типы файлов. Перед запуском утилиты раскомментируем строки, соответствующие типам восстанавливаемых файлов. Например, для восстановления GIF- и JPEG-файлов нужно привести соответствующую секцию конфигурационного файла к следующему виду:

```
# GIF and JPG files (very common)
gif      y      5000000      \x47\x49\x46\x38\x37\x61      \x00\x3b
gif      y      5000000      \x47\x49\x46\x38\x39\x61      \x00\x3b
jpg     y     200000000      \xff\xd8\xff\xe0\x00\x10      \xff\xd9
```

Теперь запустим scalpel:

```
sudo scalpel /dev/sda1 -o recover
```

Здесь `/dev/sda1` — это имя устройства, на котором производится поиск удаленных файлов, а опция `-o` задает название папки, в которую будут помещены восстановленные файлы. Нужно отметить, что вместо имени устройства можно указать образ этого устройства. Суть в следующем: если файловая система активно используется, то чем больше времени прошло с момента удаления до попытки восстановления, тем меньше шансы. Поэтому можно «заморозить» время, создав командой `dd` сразу после обнаружения ошибочного удаления файлов образ устройства, и пытаться восстановить файлы уже с него. Так можно сократить вероятность того, что область, где был записан файл, будет перезаписана другим файлом, — тогда восстановить ничего не получится.

Вывод утилиты:

```
Scalpel version 1.60
Written by Golden G. Richard III, based on Foremost 0.69.
Opening target "/dev/sda1"
Image file pass 1/2.
/dev/sda1: 9.1% |*****          | 9.9 GB 39:16 ETA
```

Осталось только дождаться восстановления.

## 4.14. Новшества Fedora 33: zRAM и Btrfs

Как было отмечено ранее, в Fedora 33 появилось сразу два нововведения, относящихся к файловой системе. Во-первых, файловую систему ext4 в ней заменила Btrfs, во-вторых, вместо разделов подкачки на жестком диске теперь используется zRAM.

Больше всего опасений вызывает производительность Btrfs. Поскольку эта система поддерживает сжатие, то можно ожидать, что она будет работать медленнее, чем ext4. Ресурсом Delightly Linux было проведено сравнительное тестирование производительности двух этих систем (рис. 4.17).

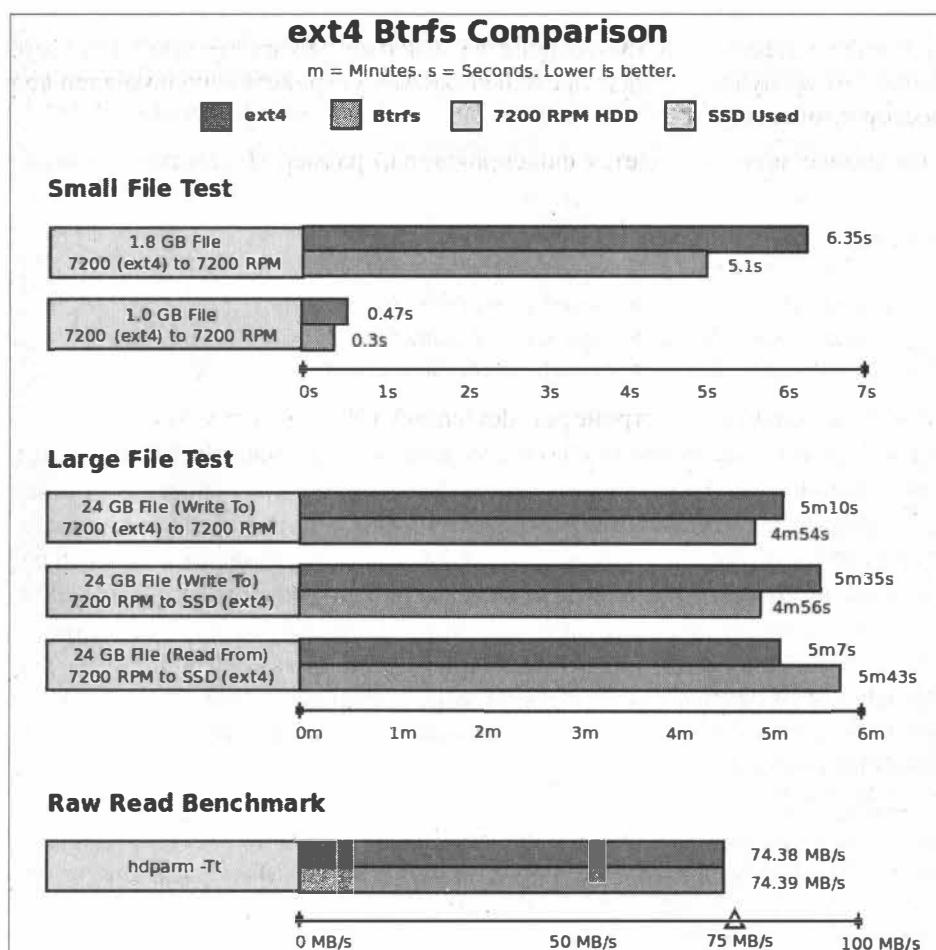


Рис. 4.17. Сравнение производительности Btrfs и ext4 (источник: <https://delightfullylinux.wordpress.com/2015/03/12/which-is-faster-btrfs-or-ext4/>)

Результаты теста показывают, что ext4... медленнее, чем Btrfs. Особенно разница заметна при работе с небольшими файлами. Причем под небольшими подразумеваются размеры 1–1,8 Гбайт. С ростом размера файла разница становится менее

заметной. И только при чтении файла размером 24 Гбайт ext4 показывает большую производительность — файл считывается на 36 секунд быстрее. А при прямом (Raw) доступе производительность файловых систем одинакова. Получается, что в большинстве случаев Btrfs быстрее ext4, и волноваться по поводу производительности не стоит.

Теперь о zRAM. С виртуальными дисками RAM, организуемыми в быстродействующей оперативной памяти, думаю, знакомы все читатели этой книги, поэтому я не привожу здесь пространного введения в тему RAM-дисков и пояснений, зачем они нужны. А zRAM — это RAM-диск со сжатием. Другими словами, он еще и экономит место в памяти. Посмотрим, как его использовать:

```
modprobe zram num_devices=4
```

Эта команда создает 4 zRAM-устройства. Сжатие каждого устройства — однопоточное, поэтому нужно создавать столько zRAM-устройств, сколько ядер содержит процессор компьютера.

При настройке модуля задается фиксированный размер НЕ сжатых данных в байтах:

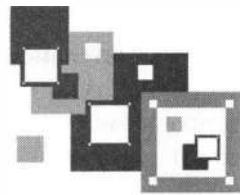
```
SIZE=1024  
echo $((SIZE*1024*1024)) > /sys/block/zram0/disksize  
echo $((SIZE*1024*1024)) > /sys/block/zram1/disksize  
echo $((SIZE*1024*1024)) > /sys/block/zram2/disksize  
echo $((SIZE*1024*1024)) > /sys/block/zram3/disksize
```

В итоге будет создано 4 устройства `/dev/zramN` ( $N$  — номер устройства) заданного размера (1 Гбайт). Остается только создать на этих устройствах разделы подкачки и подключить их:

```
mkswap /dev/zram0  
mkswap /dev/zram1  
mkswap /dev/zram2  
mkswap /dev/zram3
```

```
swapon /dev/zram0 -p 10  
swapon /dev/zram1 -p 10  
swapon /dev/zram2 -p 10  
swapon /dev/zram3 -p 10
```

Далее от пользователя уже ничего не зависит — ему нужно просто работать. А ядро само будет определять, какие данные записывать на RAM-диски и как их использовать.



## ГЛАВА 5

# Командный интерпретатор bash

### 5.1. bash: основные сведения

bash — это наиболее часто использующаяся 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, tcsh<sup>1</sup> и пр. Все командные оболочки, установленные в системе, прописаны в файле /etc/shells. В листинге 5.1 представлен файл /etc/shells дистрибутива Fedora 30 с установками по умолчанию. В этом дистрибутиве, по сравнению, скажем, с Fedora 16, набор оболочек существенно сокращен и оставлены только sh и bash<sup>2</sup>. По сути, в Fedora 30 существует одна командная оболочка, потому что sh и bash — это практически одно и то же. Включенная в листинг 5.1 оболочка tmux представляет собой *терминальный мультиплексор*, позволяющий работать с несколькими сессиями в одном окне. С одной стороны, такое решение интересно тем, что дает возможность создать в консольном режиме как бы несколько «окон». С другой стороны, не всем это нужно, и многие по привычке предпочитают или переключаться на другую консоль (комбинацией клавиш <Alt>+<Fn>), или открывать новое окно терминала (при работе по ssh). Если вас заинтересовал мультиплексор tmux, вы можете подробнее познакомиться с ним в статье по адресу: <https://habr.com/ru/post/327630/>.

Листинг 5.1. Файл /etc/shells дистрибутива Fedora 30

```
/bin/sh  
/bin/bash
```

<sup>1</sup> В папку Дополнения сопровождающего книгу файлового архива включен материал «Автоматизация задач с помощью tcsh», посвященный этому интерпретатору.

<sup>2</sup> Ранее в этом списке присутствовала еще и программа plogin. Однако plogin — это не оболочка, а утилита, отображающая сообщение, что учетная запись недоступна, — в случае обращения к отключенной или системной учетной записи.

```
/usr/bin/sh  
/usr/bin/bash  
/usr/bin/tmux  
/bin/tmux
```

С точки зрения пользователя, 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 заключается 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 оболочки `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 внешний носитель — флешку или внешний USB-диск, хотя с точки зрения системы особой разницы нет. 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 внешний носитель! И все, что для этого потребуется, — это проверить перед запуском сценария наличие на этом носителе достаточного количества свободного места.

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 администраторы 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 обязательен при использовании переменной;
- при объявлении переменной не должно быть пробелов до и после знака =.

Значение для переменной указывать вручную не обязательно — его можно прочитать 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"
        ;;
    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 напишем 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
    команда_1
elif условие_2 then
    команда_2
...
elif условие_N then
    команда_N
else
    команда_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) команда_1 ;;
...
значение_N) команда_N ;;
*) команда_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.

## 5.9. Мониторинг и перезапуск сервисов Apache и MySQL с помощью bash

Интерпретатор `bash` вполне можно использовать для решения практических задач по администрированию системы, и далее будет показано несколько примеров. Начнем с мониторинга веб-сервера `Apache`. Иногда случается, что сервис «падает» и происходит это, как правило, в самое неудобное время: или ночью, когда администратор спит, или когда он уехал в отпуск. Пока причина падения сайта не выяснена (а это может быть все что угодно, в том числе и неправильная настройка самого сервера), нам необходимо иметь средство мониторинга и перезапуска сервиса. Сторонние ресурсы вроде Яндекс.Метрики хороши, но они не позволяют перезапустить сервер, а только сообщают, что «сайт упал, потому что он устал». В листинге 5.4 приведен простейший скрипт, который обеспечит перезапуск сервера `Apache` в случае его отказа.

### Листинг 5.4. Перезапуск сервера Apache

```
#!/bin/bash
if curl -s --head --request GET http://site.name | grep "200 OK" > /dev/null;
then
echo "Site is UP"
else
echo "site is DOWN, restarting Apache"
/usr/sbin/service apache2 restart
fi
```

Работает он просто — с помощью команды `curl` получается главная страница сайта. Если она не может быть получена, тогда сайт перезапускается с помощью

команды `service`. Почему не `systemctl`? Да потому, что в новых системах старая команда `service` все еще поддерживается, а вот в старых поддержки команды `systemctl` нет. Так что команда `service` используется здесь из соображений обратной совместимости с не очень «свежими» системами.

Аналогичным образом мы можем проверить, работает ли MySQL (листинг 5.5), поскольку часто причиной отказа сайта является отказ именно базы данных, а не веб-сервера.

#### Листинг 5.5. Перезапуск MySQL

```
#!/bin/bash
RESTARTM="service mysql restart"
MYSQLD="mysqld"
$PGREP ${MYSQLD}
if [ $? -ne 0 ]; then
$RESTARTM
echo "MySQL restarted" > /var/log/srvmon.log
fi
```

Самая распространенная причина падения сайта — банально закончилось место на диске. Apache пишет весьма подробные логи, база данных постоянно растет, и т. д. Так что даже если вы сами активно не занимаетесь сайтом, может случиться, что он сам по себе заполнит имеющееся место на диске, веб-сервер больше не сможет записать строку в журнал и прекратит свою работу. Спасти может скрипт мониторинга доступного места на диске (листинг 5.6).

#### Листинг 5.6. Мониторинг свободного места на диске

```
#!/bin/bash
# Задаем переменную, где вычисляем свободное место на диске /dev/vda2
# (в мегабайтах)
freespace=`df -m | grep "/dev/vda2" | awk '{print $4}'`
# Если свободного места меньше 3000 Mb, то отправляем письмо на e-mail.
if [ $freespace -lt 3000 ];
then
echo "Warning!!! Out of space on server. Freespace - \"$freespace\"Mb" | mail -s
"FreeSpase" admin@example.com
fi
```

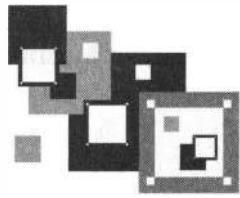
Здесь вам следует изменить имя устройства (`/dev/vda2`) на реальное, а также указать e-mail, на который будет отправляться уведомление.

Все эти три сценария нужно сделать исполнямыми, и команды их вызова добавить в расписание cron. Периодичность запуска первых двух скриптов — раз в 5 минут, третий можно запускать раз в час. Это означает, что максимальный простой сайта в случае отказа сервисов Apache/MySQL составит 5 минут (конечно, не из-за закон-

чившегося места на диске — здесь понадобится вмешательство администратора, благо третий скрипт заблаговременно предупредит его). А дисковое пространство на сервере не заканчивается внезапно, поэтому скрипт проверки достаточно запускать раз в час. Можно увеличить порог срабатывания (вместо 3000 задать 5000 Мбайт) — тогда администратор получит уведомление раньше.

Существует уже готовое решение — `monit`, но мы только что реализовали основные функции мониторинга работоспособности сервера, потратив на это не более 10 минут. А вы потратите больше времени только на чтение документации `monit`.

Дополнительно к этим трем скриптам все-таки неплохо использовать еще и сторонний сервис мониторинга вроде Яндекс.Метрики, чтобы получать уведомления о недоступности сайта по SMS.



## 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 на этом компьютере с системой, вы не помешаете 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. И 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 этого пользователя и попросить его не мешать.

По-настоящему многопользовательский режим возможен в серверных версиях Windows с использованием протокола RDP (Remote Desktop Protocol, протокол удаленного рабочего стола), но это не всегда удобно. Во-первых, RDP-подключение не всегда разрешается корпоративными политиками безопасности. А во-

вторых, часто для работы по RDP приходится 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 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;

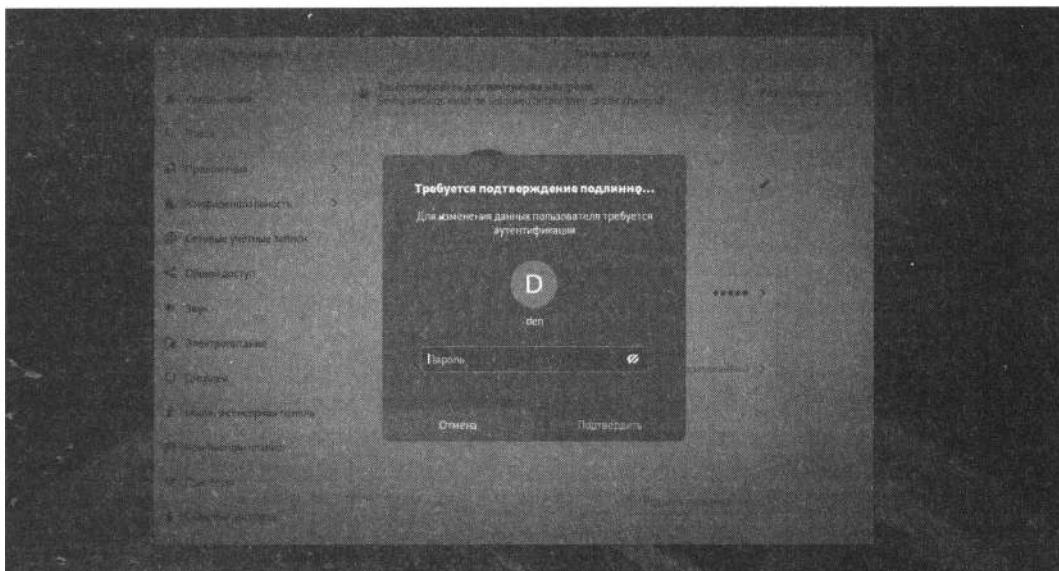


Рис. 6.1. Fedora 33: требование ввести пароль

- 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;
- 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/gksu и kdesudo/kdesu

Если вы в терминале хотите запустить графическую программу с правами root (например, gedit), желательно использовать команды sudo и su с префиксом gk: gksudo и gksu — для GNOME или kdesudo и 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 20.10: 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* — 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. Ubuntu: оболочка root

#### **ПРИМЕЧАНИЕ**

Оболочку с правами root можно также запустить командой `sudo bash`. Тогда все команды, введенные в этой оболочке, будут выполнены с правами root.

### **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 два пароля (т. е. свой 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 такого 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 управления пользователями NSS (Name Service Switch). 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.

### **Вход как root без пароля**

В папке *Видео* сопровождающего книгу файлового архива (см. [приложение](#)) содержится видеофайл, демонстрирующий процессы передачи параметров ядра и входа как root без пароля.

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

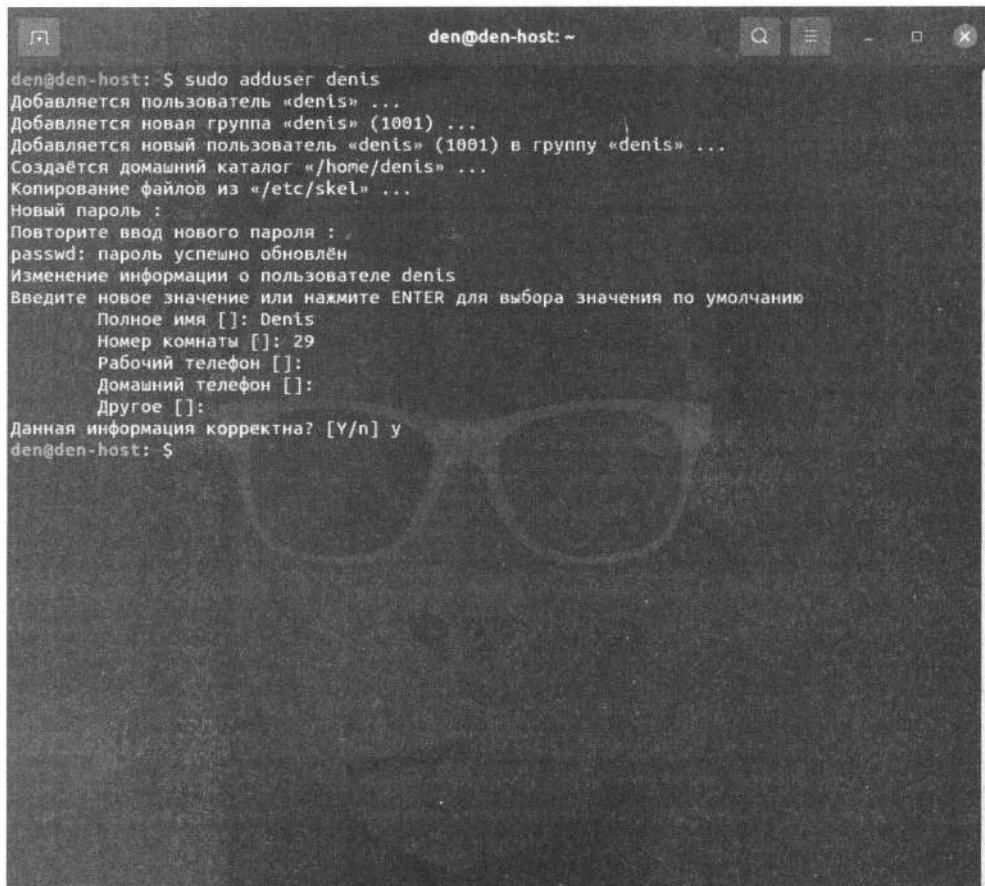
### 6.3.1. Отдельные пользователи

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).



```
den@den-host: $ sudo adduser denis
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» ...
Новый пароль :
Повторите ввод нового пароля :
passwd: пароль успешно обновлён
Изменение информации о пользователе denis
Ведите новое значение или нажмите ENTER для выбора значения по умолчанию
    Полное имя []:
    Номер комнаты []:
    Рабочий телефон []:
    Домашний телефон []:
    Другое []:
Данная информация корректна? [Y/n] y
den@den-host: $
```

Рис. 6.4. Ubuntu: добавление нового пользователя

### 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*.

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 модификации 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 недежный — 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 «дженртльменский 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. Пусть над каким-либо проектом у вас должны совместно работать три разных пользователя — их достаточно объединить в одну группу, и тогда они получат доступ к домашним каталогам друг друга (по умолчанию пользователи не имеет доступа к домашним каталогам других пользователей, поскольку считается, что 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 Fedora 33, на втором — Ubuntu 20.10. Вот только запускаются конфигураторы по-разному:

- в Fedora нужно открыть окно **Параметры**, перейти в раздел **Подробности** и щелкнуть на кнопке **Пользователи**;
- в Ubuntu нужно открыть приложение **Настройки** (она имеет вид шестеренки), перейти в раздел **Сведения о системе** и запустить конфигуратор **Пользователи**.

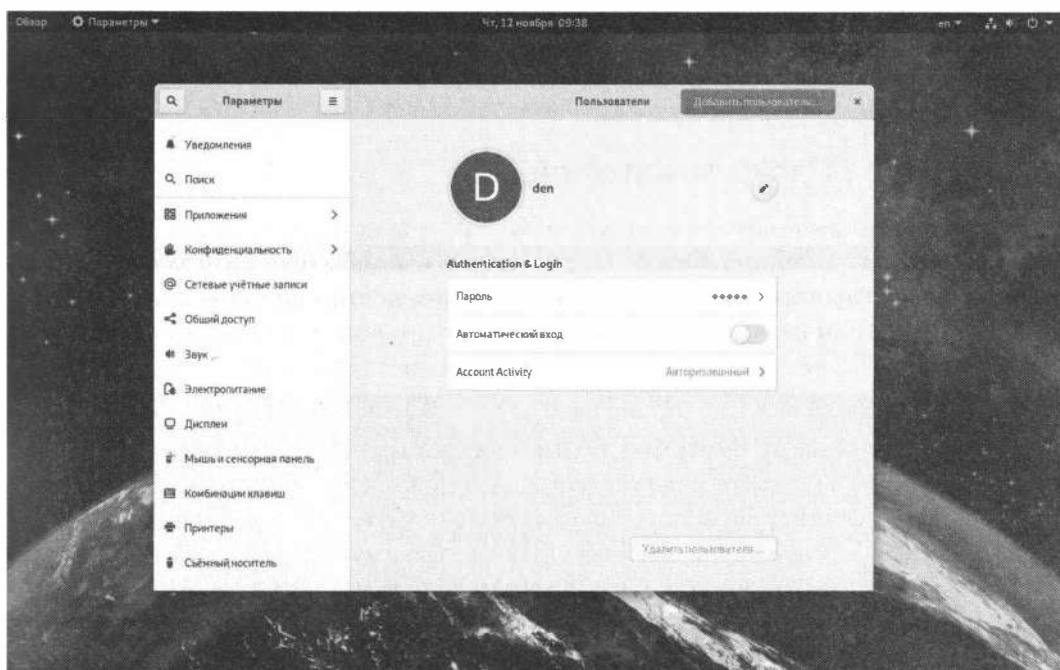


Рис. 6.5. Fedora 33: конфигуратор управления пользователями

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).

### 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 на нем щелкнуть. Например, для изменения изображения пользователя щелкните на пользователе и выберите подходящее для него изображение (рис. 6.8).

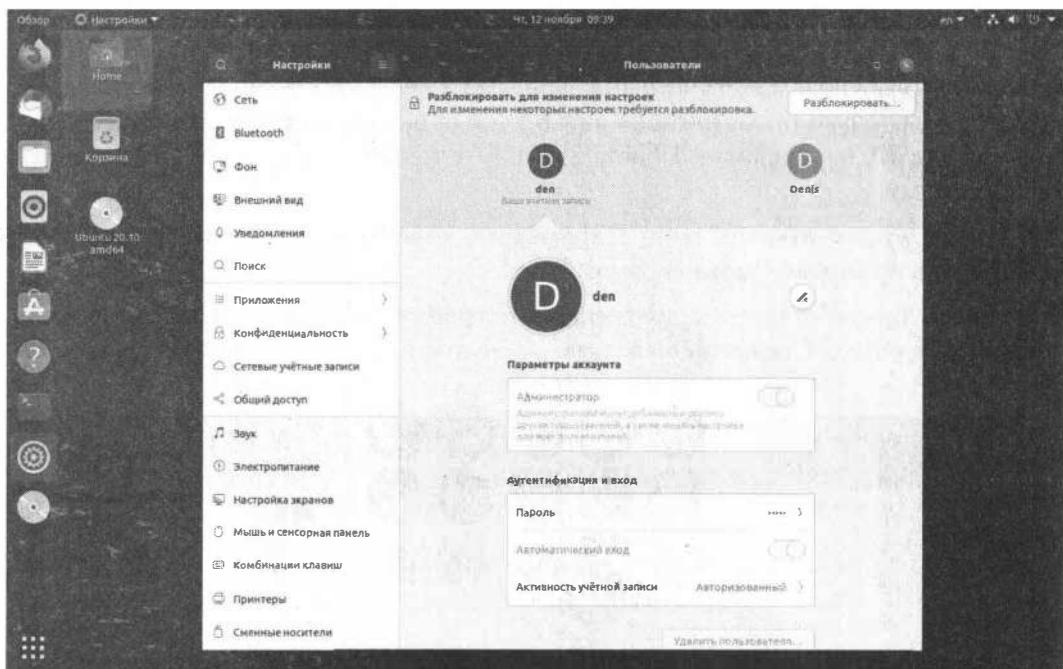


Рис. 6.6. Ubuntu 20.10: конфигуратор управления пользователями

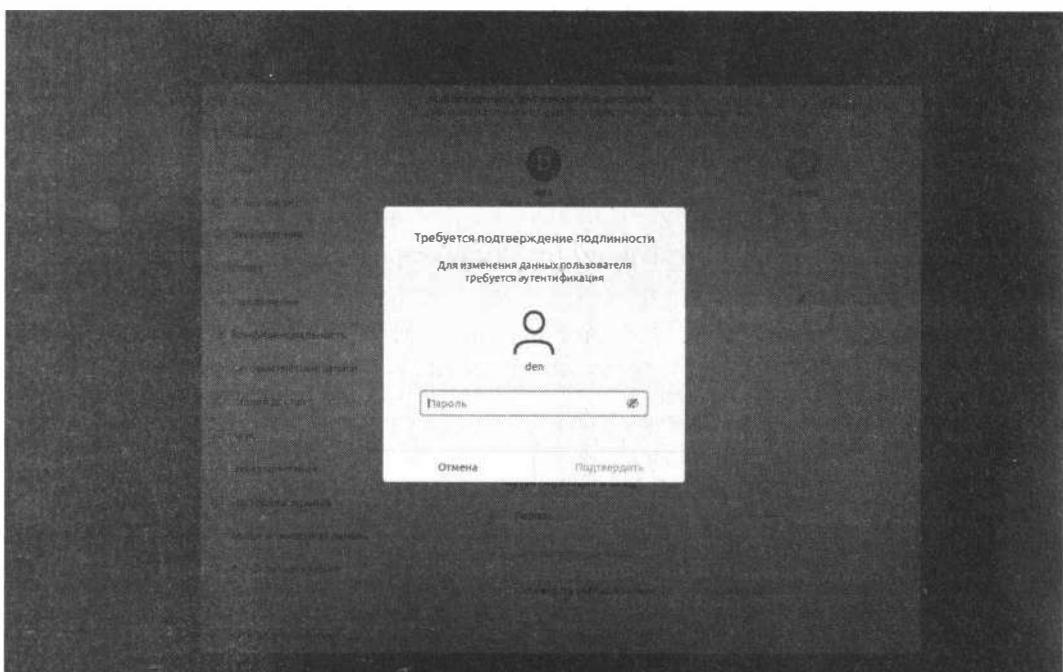


Рис. 6.7. Ubuntu 20.10: разблокирование конфигуратора

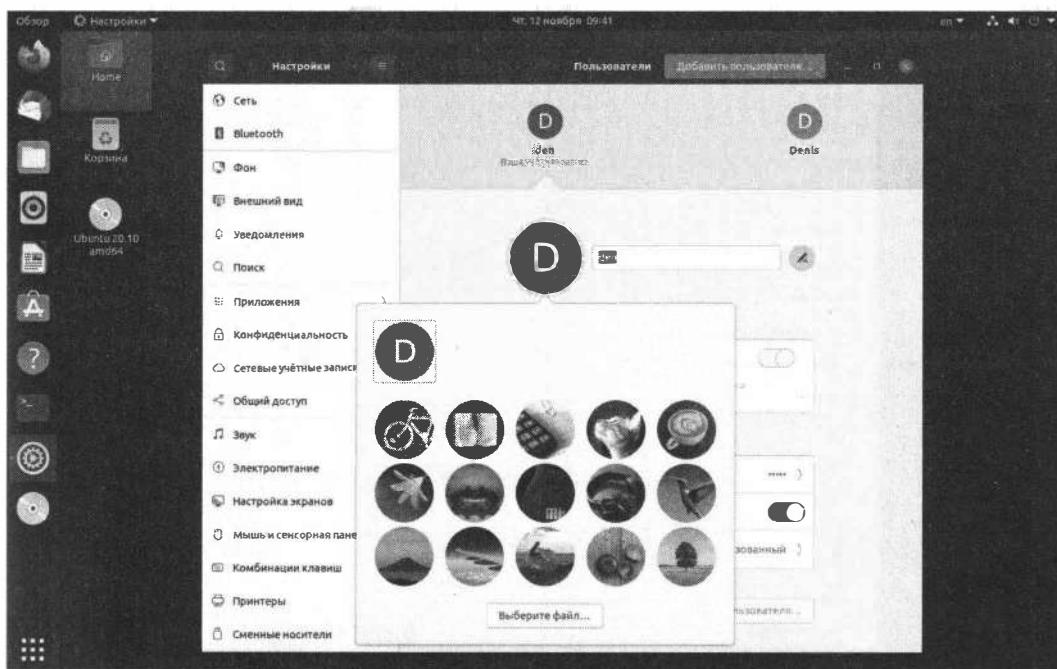


Рис. 6.8. Ubuntu 20.10: выбор картинки пользователя

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. п.);
- **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 учетной записи изменить нельзя. При создании же новой учетной записи выбрать для нее тип разрешается.

Для добавления пользователя нажмите в окне конфигуратора кнопку **Добавить пользователя**. 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.

Для удаления пользователя служит имеющаяся в правом нижнем углу конфигуратора кнопка **Удалить пользователя**. При этом конфигуратор спросит, что сделать с файлами пользователя (с его домашним каталогом): удалить или сохранить на диске (рис. 6.10).

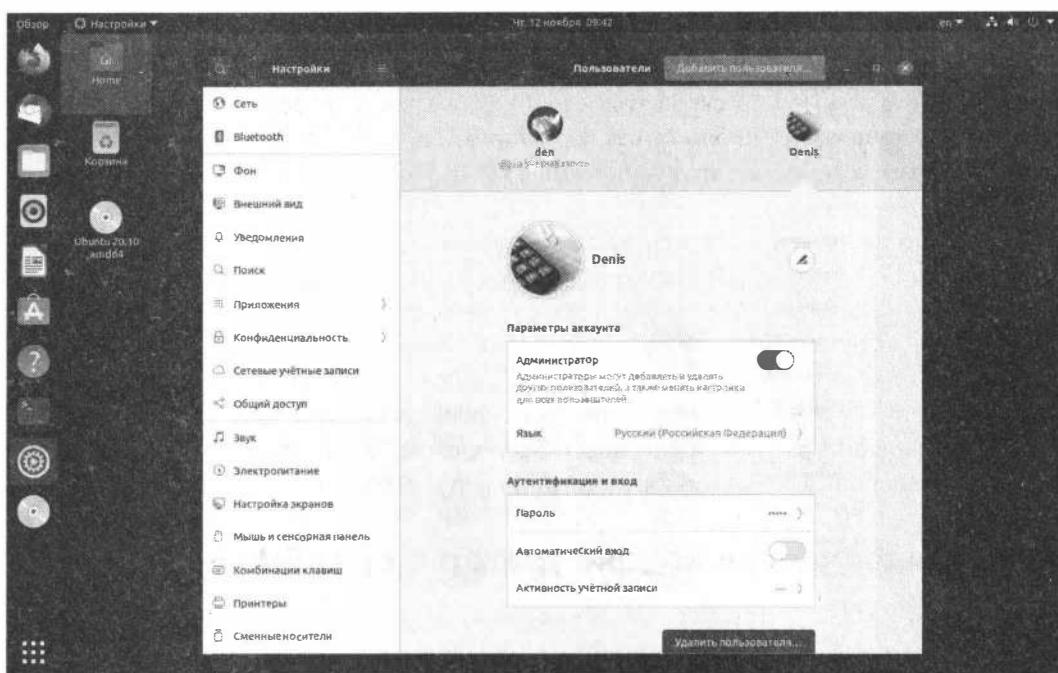


Рис. 6.9. Ubuntu 20.10: выбор типа учетной записи при создании новой учетной записи

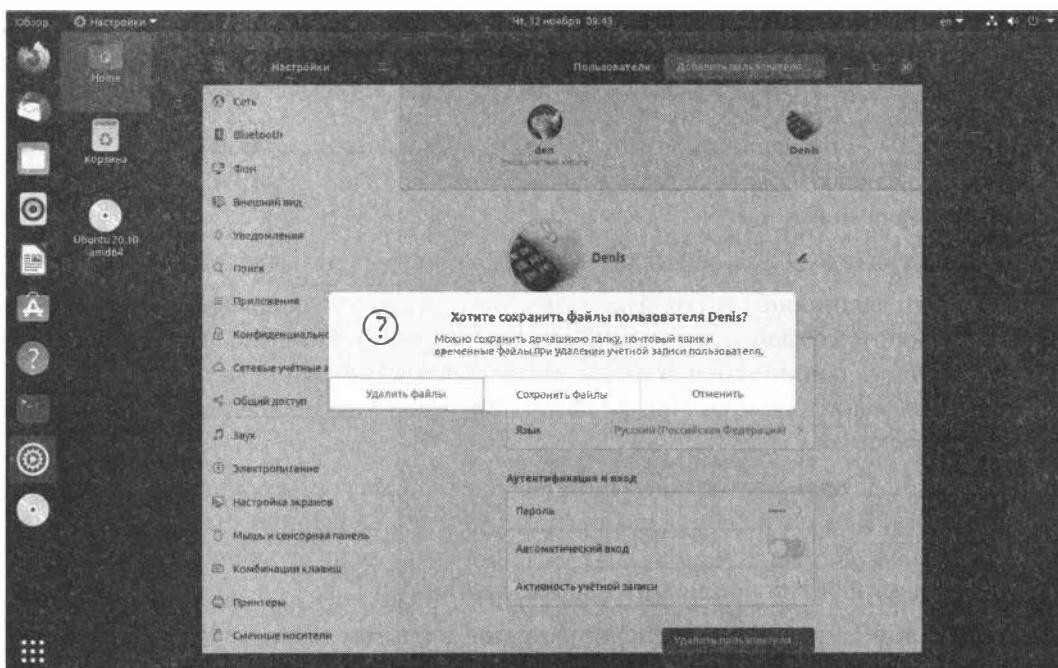


Рис. 6.10. Ubuntu 20.10: удаление пользователя

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 полезно 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

Графический конфигуратор в openSUSE называется **YaST2 - users @ install** (рис. 6.11). Для его запуска используйте меню GNOME/KDE — просто начните вводить начальные буквы названия в строке поиска.

Использовать этот 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 на вкладке **Подробности** (рис. 6.12) у вас есть возможность выбрать, к каким группам должен принадлежать этот пользователь (в списке **Дополнительные группы**). Если пользователю не нужен доступ к Интернету, не следует помечать его принадлежность к группе **dialout**.

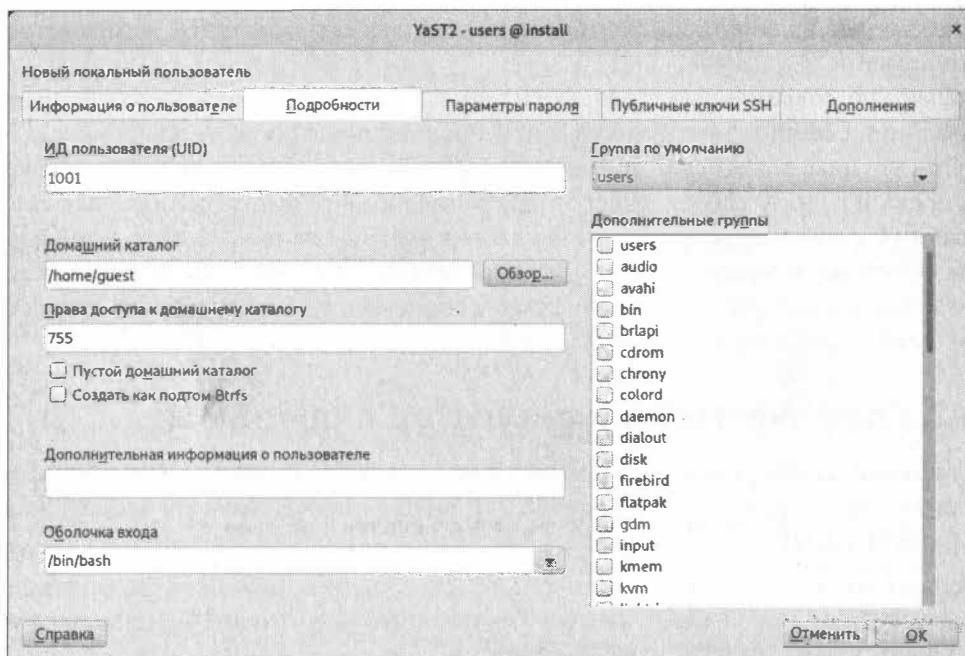


Рис. 6.12. openSUSE: создание нового пользователя

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 по нажатию кнопки **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 флагок **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** (см. рис. 6.11), выбрать нужного пользователя, нажать кнопку **Редактировать**, затем перейти на вкладку **Подробности** (см. рис. 6.12) и уже там отключить группы, членом которых не должен быть пользователь. Да, неудобно, но другого способа нет.

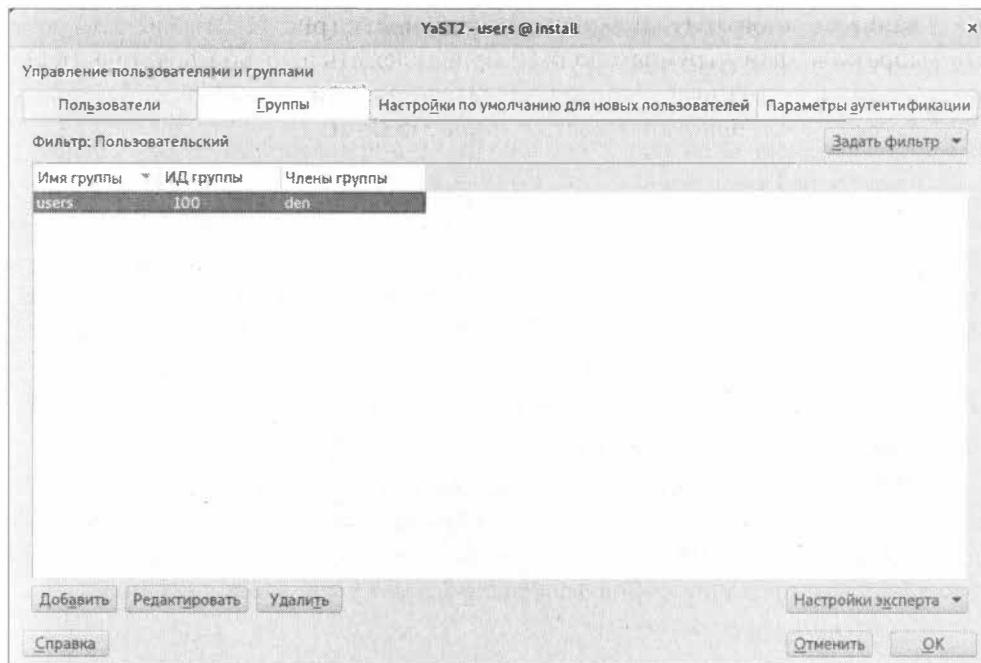


Рис. 6.13. openSUSE: окно Управление пользователями и группами, вкладка Группы

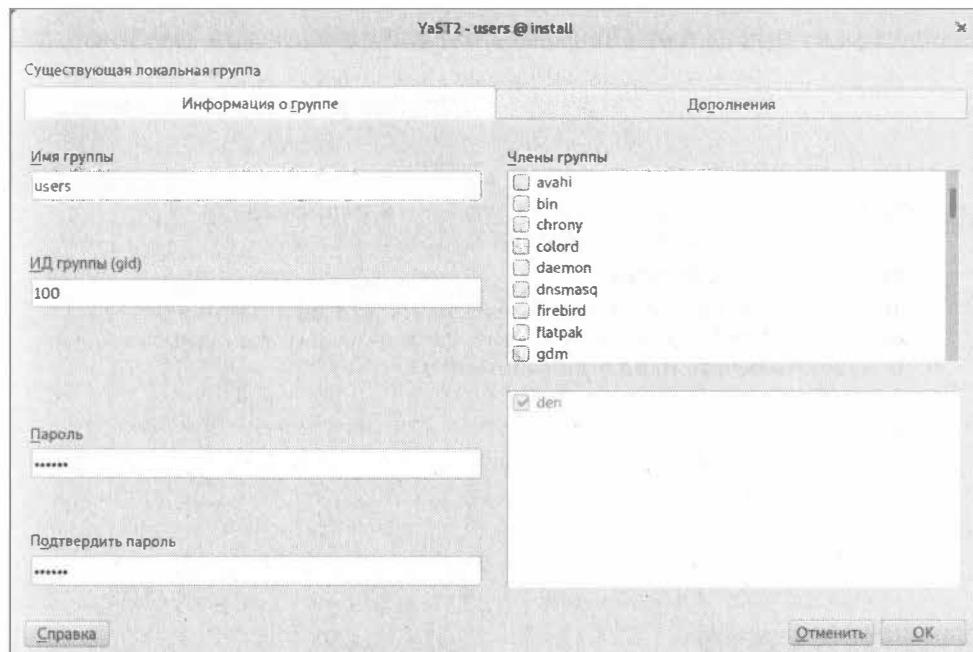


Рис. 6.14. openSUSE: изменение группы

## 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 среды GNOME (**Обзор | Показать приложения | Терминал**) 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 каж-

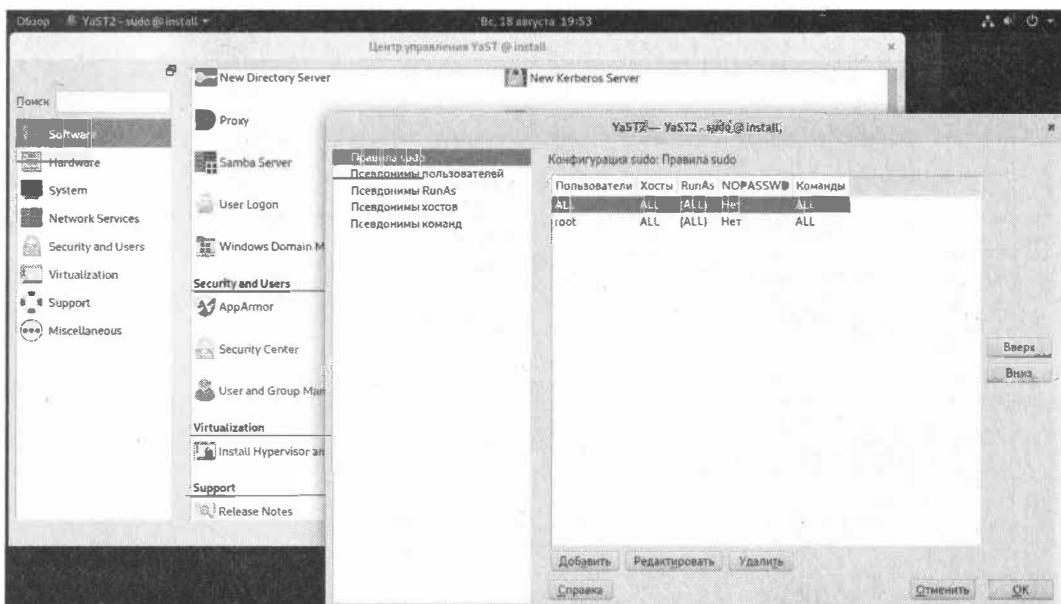


Рис. 6.15. openSUSE: использовать команду *sudo* могут все пользователи этой системы

дый пользователь, а только указанные в файле `/etc/sudoers` (файл редактируется не вручную, а с помощью конфигуратора YaST2 - `sudo @ install`). 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
```

## Конфигуратор Центр безопасности openSUSE

В группе **Security and Users** (Безопасность и пользователи) конфигуратора YaST2 имеется конфигуратор **Security Center** (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, зло-

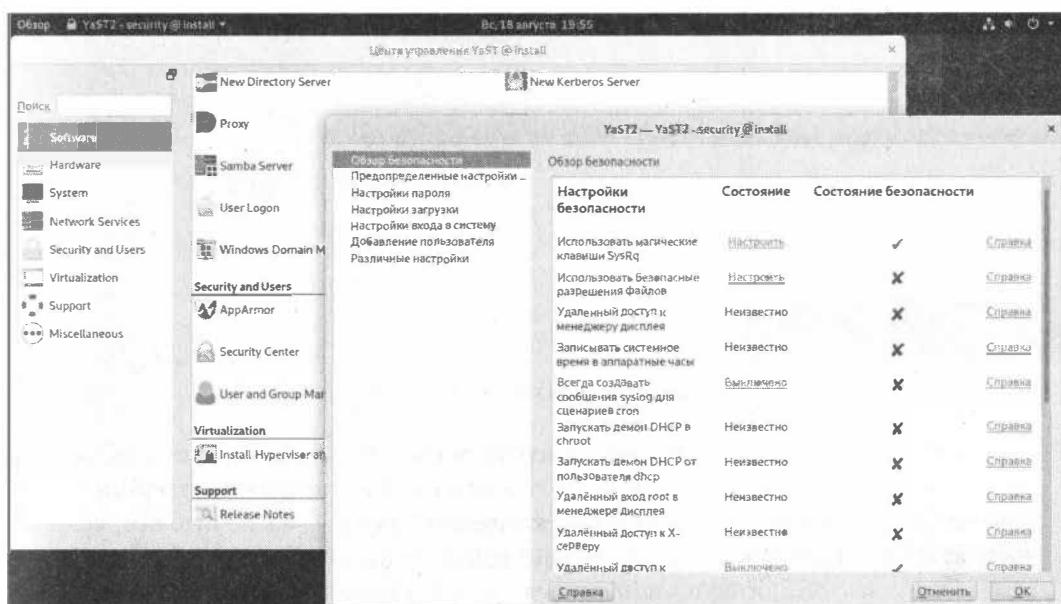


Рис. 6.16. openSUSE: Центр безопасности

умышленник 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.
- 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.
- В разделе **Настройки загрузки** вы можете установить реакцию на нажатие комбинации клавиш <Ctrl>+<Alt>+<Del>. Выключить реакцию на нажатие этой комбинации клавиш целесообразно на сервере, чтобы никто случайно его не перезагрузил.

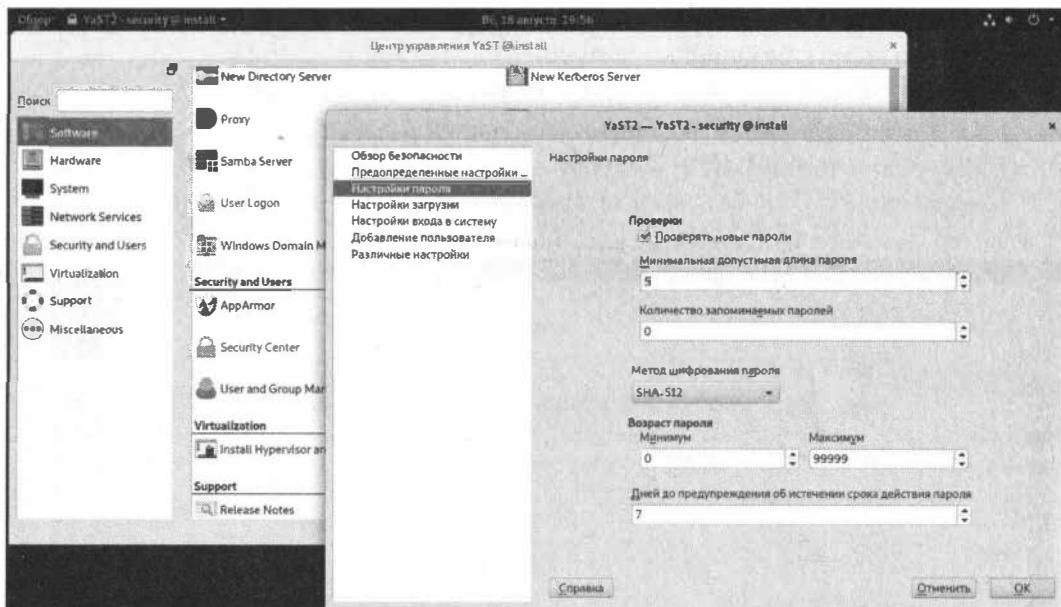


Рис. 6.17. openSUSE: параметры пароля

- 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 здесь не обойтись. Когда вы используете 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 -vagut
```

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 subtracted.
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 слишком 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"/> seconds — секунды; | <input type="checkbox"/> days — дни;      |
| <input type="checkbox"/> minutes — минуты;  | <input type="checkbox"/> weeks — недели;  |
| <input type="checkbox"/> hours — часы;      | <input type="checkbox"/> months — месяцы. |

Например:

- `24hours` — 24 часа;
- `2days` — 2 дня;
- `1weeks` — 1 неделя.

Включим квотирование для наших файловых систем:

```
# quotaon файловая_система
```

Например:

```
# quotaon /
```

После этого перезагрузим систему:

```
= reboot
```

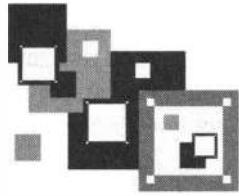
При загрузке вы увидите сообщение: **Turning on user and group quotas for local filesystems** (Включаем квоты пользователей и групп для локальных файловых систем) — это означает, что механизм квотирования работает правильно.

Для просмотра квот служит команда `repquota`, например:

```
* repquota /home
```

Наверняка так устанавливать квоты для каждого пользователя в отдельности, особенно если их много, вам покажется неудобным. 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: <https://www.debian.org/distrib/packages> и <https://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 прочтите его и попытайтесь устранить причину ошибки — например, установите недостающую библиотеку. Ясно, что в случае возникновения 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 предварительно прочитать файл *README*, который обычно присутствует в архиве.

Пакет установки программы, как правило, состоит из набора файлов — например, исполняемого 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 требоваться 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, уста-

новка будет прервана — пока вы не установите все необходимое. Впрочем, имеется 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 (MTA, 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 любом другом носителе данных, 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 дождаться 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 протестирували (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, веб-сервер. А через некоторое время оказалось, что в этой версии веб-сервера имеется «дыра», поэтому разработчики вскоре выпустили новую ее версию, эту «дыру» закрывающую. Пользователь же, установивший исходную программу веб-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
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
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 на Red Hat-совместимую версию Linux пакет, который *не входит* в состав ее дистрибутива (например, загруженный из Интернета), вам следует использовать программу rpm.

### **ГРАФИЧЕСКИЙ МЕНЕДЖЕР ПАКЕТОВ RPMDRAKE**

Для установки пакетов, которые *входят* в состав дистрибутива, намного удобнее использовать графический менеджер пакетов rpmdrake.

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:

<https://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 `urpmi`.

### 7.5.1. Установка пакетов

Для установки пакета служит команда:

```
# urpmi <имя пакета>
```

Так, чтобы установить пакет `mc` (файловый менеджер *Midnight Commander*), следует ввести команду:

```
# urpmi mc
```

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:

```
# urpmi <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 уим

Команда	Описание
yum install пакет	Установить пакет из репозитория (также устанавливаются пакеты, необходимые для работы устанавливаемого пакета, т. е. разрешаются зависимости)
yum remove пакет	Удалить пакет, а также все пакеты, которые зависят от него
yum update	Проверить наличие обновлений всех пакетов. Если обновления есть, то они будут установлены
yum update пакет	Проверить обновления конкретного пакета. Если есть свежая версия, то она будет установлена
yum check-update	Только проверить наличие обновлений (обновления не устанавливаются)
yum check-update пакет	Проверить наличие обновлений конкретного пакета (обновления не устанавливаются)
yum info пакет	Вывести информацию о пакете
yum list	Вывести список всех пакетов: как установленных, так и доступных для установки из репозиториев
yum list a*	Вывести список всех пакетов, которые начинаются на букву «а»
yum search строка	Найти все пакеты, в описаниях которых есть указанная строка
yum groupinstall "группа"	Установить все пакеты из указанной группы
yum grouplist	Вывести список групп пакетов

При установке пакетов с помощью уим не следует далеко отходить от компьютера — часто нужные пакеты находятся не на локальных источниках, а на серверах в Интернете, поэтому уим 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):

**Name** (Название) — имя пакета;

**Эпоха** (Epoch) — как бы подверсия пакета, поле Эпоха используется, когда требуется уменьшить версию или релиз пакета по сравнению с имеющимся в репозитории;

- Version** (Версия) — версия пакета;
- Выпуск** (Release) — релиз пакета (можете считать это подверсией пакета);
- Architecture** (Архитектура) — архитектура компьютера;
- Size** (Объем) — размер занимаемого места на диске;
- Источник** (Source) — название файла пакета (RPM-файла);
- Repository** (Репозиторий) — хранилище, в котором сейчас находится пакет;
- From repository** (Из репозитория) — хранилище, из которого был установлен пакет (только для установленных пакетов);
- Summary** (Аннотация) — общая информация о пакете;
- URL** (Ссылка) — веб-страница разработчика программы;
- Лицензия** (License) — лицензия, по которой распространяется программа;
- Description** (Описание) — описание пакета.

### ПРИМЕЧАНИЕ

К сожалению, разработчики дистрибутивов даже в 2019 году уделяют мало внимания деталям. Это касается не только Fedora, у которой всегда были проблемы с локализацией, но и других дистрибутивов — той же Ubuntu. Вы только посмотрите на рис. 7.1 — хоть что-то, да не переведено... Даже в более старых версиях Fedora с локализацией консольных программ ситуация была лучше.

```
[den@localhost ~]$ sudo yum info mc
den@fedora:~
```

[den@localhost ~]\$ sudo yum info mc

Мы полагаем, что ваш системный администратор изложил вам основы безопасности. Как правило, всё сводится к трем следующим правилам:

№1) Уважайте частную жизнь других.  
№2) Думайте, прежде что-то вводить.  
№3) С большой властью приходит большая ответственность.

[sudo] пароль для den:

Fedora Modular 33 - x86_64 - Updates	15 kB/s   21 kB 00:01
Fedora 33 - x86_64 - Updates	16 kB/s   18 kB 00:01

Имеющиеся пакеты

Имя	:	mc
Эпоха	:	1
Версия	:	4.8.25
Выпуск	:	3.fc33
Архитектура	:	x86_64
Размер	:	2.6 М
Источник	:	mc-4.8.25-3.fc33.src.rpm
Репозиторий	:	fedora
Краткое описание	:	User-friendly text console file manager and visual shell
URL	:	<a href="http://www.midnight-commander.org/">http://www.midnight-commander.org/</a>
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. Fedora 33: вывод информации о пакете

Для вывода информации обо всех пакетах можно использовать команду `yum list`, но пакетов слишком много, и такой вывод может оказаться весьма 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 `GNOOME`), `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.fedoraproject.org/pub/fedora/linux/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 `baseurl`, а параметр `mirrorlist` определяет список зеркал — копий репозитория, которые будут использоваться, если URL источника, указанный в `baseurl`, окажется недоступен. Параметр `enabled`, установленный в 1, указывает на то, что этот источник активен, и утилита использует его при установке пакетов. Следующий параметр: `gpgcheck` — обязывает утилиту 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 веб-серверах репозиториев, 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://download1.rpmfusion.org/free/fedora/
rpmfusion-free-release-stable.noarch.rpm
http://download1.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 использовать не будет.

### 7.6.3. Установка пакетов через прокси-сервер

По умолчанию утилита 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_username=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
```

## 7.7. Менеджер пакетов dnf

Менеджер пакетов dnf пришел в дистрибутив 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. В современных версиях Fedora команда yum представляет собой ссылку на команду dnf.

По сравнению с yum новый менеджер более быстрый, обладает низким потреблением памяти и эффективнее управляет зависимостями.

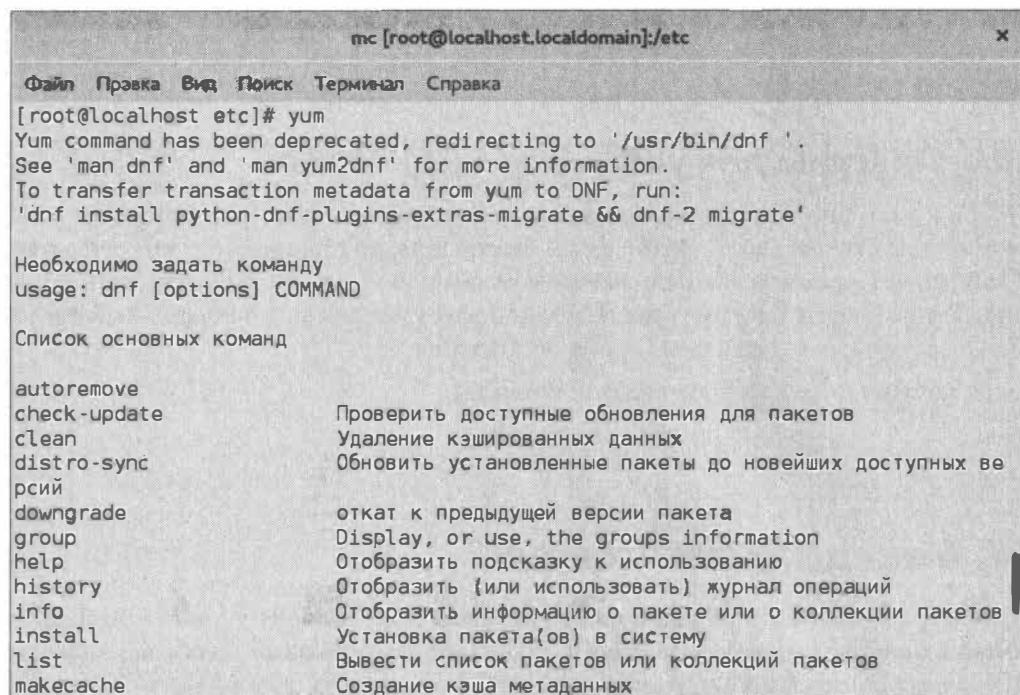
Основные команды менеджера dnf (`install`, `remove`, `upgrade`, `info`, `search` и т. д.) такие же, что и в менеджере пакетов yum, поэтому при переходе с yum на dnf вы не ощутите никакого дискомфорта. Основным конфигурационным файлом менеджера является файл `/etc/dnf/dnf.conf`. Его синтаксис полностью 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 (по состоянию на ноябрь 2020 года и в 33-й версии Fedora — все еще поддерживается). Более того, даже не выводится сообщение о том, что yum — это ссылка на dnf, как это было в более старых версиях Fedora (рис. 7.2).

Вот, нравится мне в Fedora 22–33, 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 22–33 (и в других дистрибутивах на базе GNOME) можно задействовать Центр приложений (команда `gnome-software`), показанный на рис. 7.4.



**Рис. 7.2. Fedora 26: команда yum все еще поддерживается, хотя является просто псевдонимом для команды dnf**

The terminal window title is 'den@fedora:~'. The session starts 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.
```

Then, the user runs:

```
[den@localhost ~]$ which yum
/usr/bin/yum
[den@localhost ~]$ mc
bash: mc: command not found...
Install package 'mc' to provide command 'mc'? [N/y] y
```

The user responds with 'y'. The package installation process begins:

```
* Waiting in queue...
The following packages have to be installed:
gpm-libs-1.20.7-24.fc33.x86_64 Dynamic library for gpm
mc-1:4.8.25-3.fc33.x86_64 User-friendly text console file manager and visual shell
slang-2.3.2-8.fc33.x86_64 The shared library for the S-Lang extension language
Proceed with changes? [N/y] y
```

The progress of the download and installation is shown:

```
* Waiting in queue...
* Waiting for authentication...
* Waiting in queue...
* Downloading packages...
* Requesting data...
* Testing changes...
* Installing packages...
```

The final message indicates a permission issue:

```
[den@localhost bin]$ read (STDIN_FILENO, pty_buffer...): Неприменимый к данному устройству ioctl (25)
```

**Рис. 7.3. Fedora 33: предложение установить необходимый пакет**

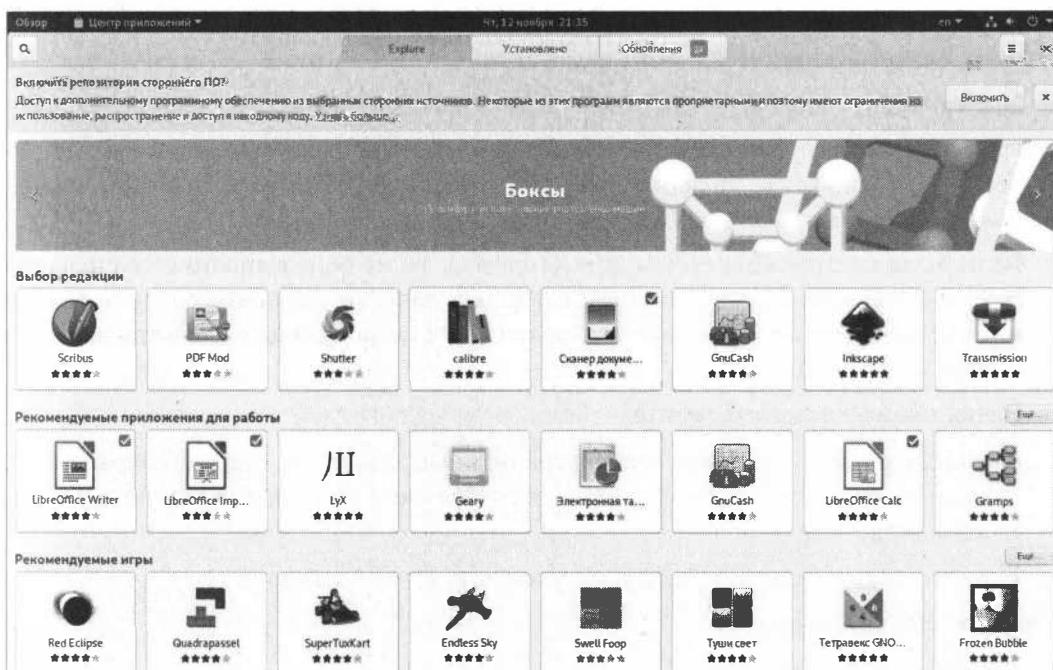


Рис. 7.4. Fedora 33: Центр приложений GNOME

Использовать эту утилиту очень просто, поэтому с ней вы сможете разобраться самостоятельно.

## 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 `prerm` — 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 * и ? — например, образец <code>a*</code> соответствует любому имени пакета, начинающемуся на букву «а». Если образец не задан, выводятся все пакеты
<code>-L &lt;имя_пакета&gt;</code>	Выводит имена файлов из указанного пакета (пакет должен быть установлен)
<code>-p &lt;имя_пакета&gt;</code>	Выводит информацию об установленном пакете
<code>-s &lt;имя_пакета&gt;</code>	Выводит информацию о статусе пакета
<code>--unpack &lt;имя_пакета.deb&gt;</code>	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`, выполните команду: `man dpkg` — страница руководства будет выведена на русском языке.

## 7.8.2. Программа apt

Программа apt 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 APT-GET**

В старых дистрибутивах использовалась команда apt-get, в новых ее название упростили до просто apt, однако команда apt-get все еще используется из соображений обратной совместимости.

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 install package
```

Программа apt просматривает файл /etc/apt/sources.list — в этом файле перечислены источники (репозитории) DEB-пакетов, в качестве которых может выступать как внешний носитель, 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 программы устанавливаемого пакета), а затем устанавливает нужный нам пакет. Все загруженные программой apt 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 репозиторий 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 ре-

позиториев `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 [опции] команды [пакет]
```

Основные команды `apt` представлены в табл. 7.4.

**Таблица 7.4. Основные команды `apt`**

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, то есть всей системы. При этом установка новых пакетов не производится, а загружаются и устанавливаются только новые версии уже установленных пакетов
<code>full-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
<code>list</code>	Показывает список пакетов на основе указанных имен
<code>show</code>	Показывает информацию о пакете
<code>search</code>	Производит поиск по описаниям пакетов
<code>autoremove</code>	Удаляет все неиспользуемые пакеты
<code>reinstall</code>	Переустанавливает пакеты

### 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 нежелательна, 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 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 install synaptic
```

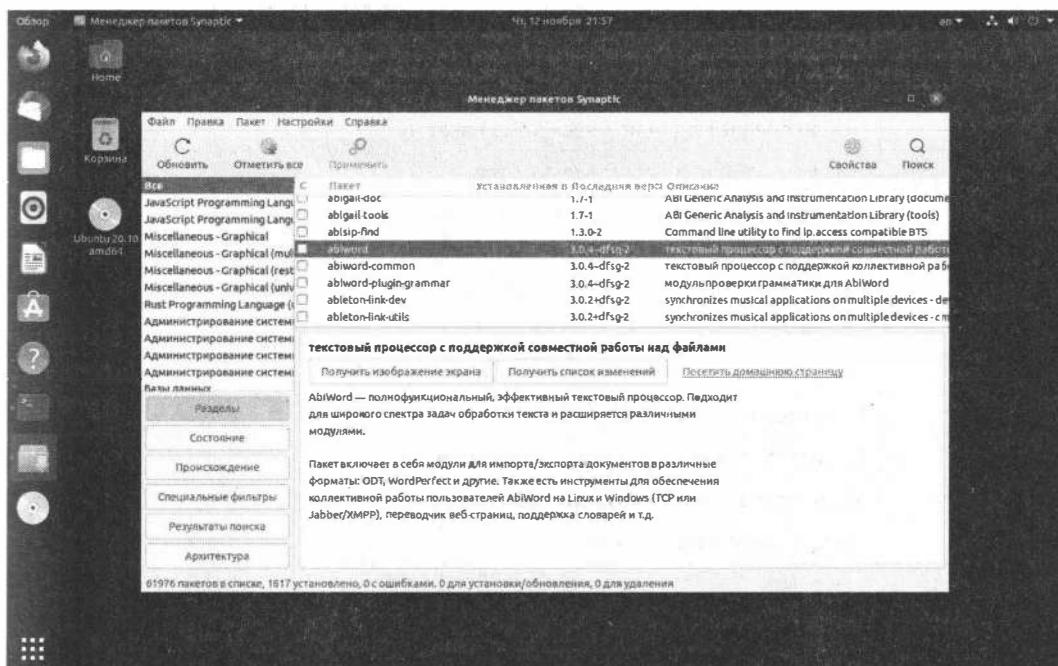


Рис. 7.5. Ubuntu 20.10: менеджер пакетов Synaptic

Кроме Synaptic в Ubuntu, насколько мне известно, имеются еще два графических менеджера пакетов: Muon (рис. 7.6) и Ubuntu Software (рис. 7.7). Получается, что вам здесь доступно как минимум три графических менеджера пакетов: Synaptic, Muon и Ubuntu Software.

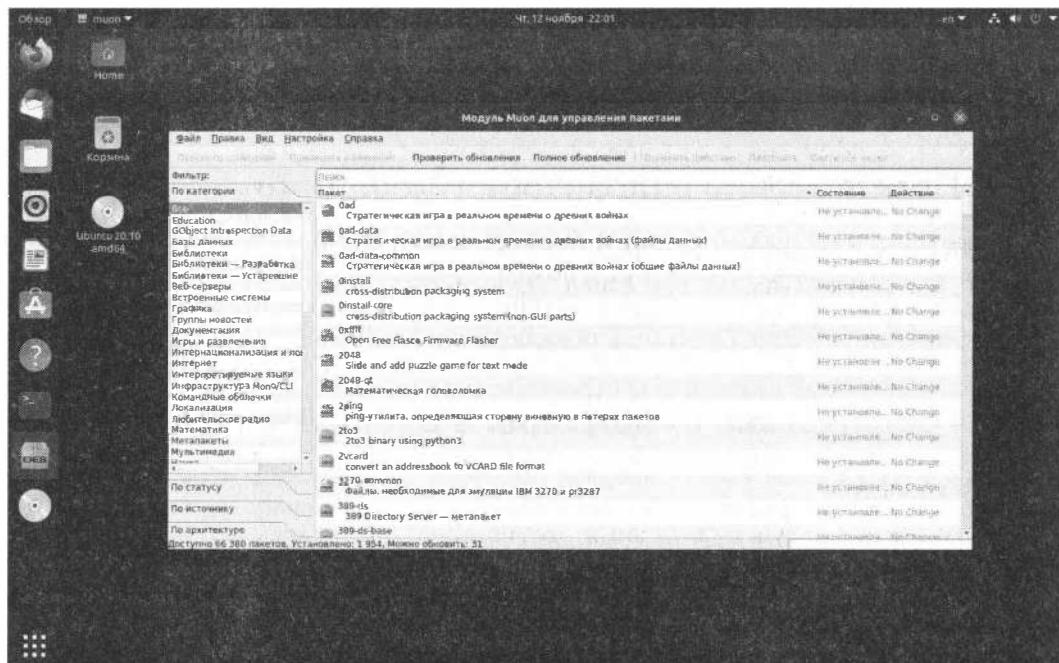


Рис. 7.6. Ubuntu 20.10: менеджер пакетов Муон

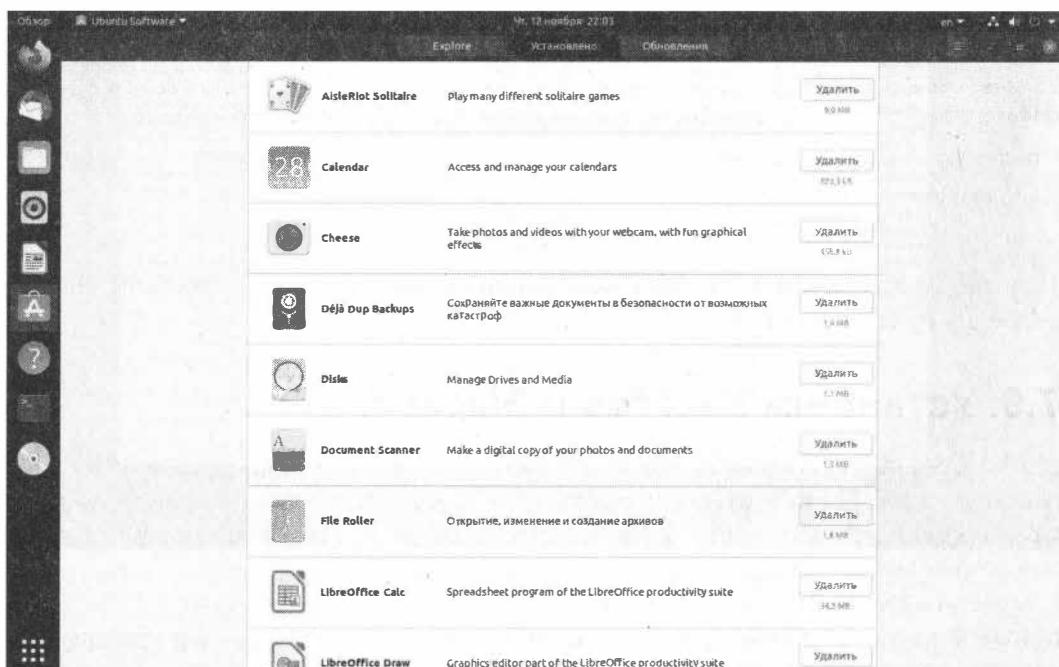


Рис. 7.7. Ubuntu 20.04: менеджер пакетов Ubuntu Software

Почему «как минимум»? Не исключено, что где-то в недрах репозиториев Ubuntu вы найдете и дополнительные менеджеры пакетов. Я не искал, поскольку использую apt или в крайнем случае Synaptic.

Какой из менеджеров пакетов выбрать, если вам не нравится apt? Для самых начинающих пользователей подойдет Менеджер приложений Ubuntu — это тот же gpmenu-software, только в Ubuntu он называется иначе. 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, читатели. Честно говоря, смотреть там не на что (и рис. 7.6 тому подтверждение). Установите Synaptic — он намного удобнее.

На самом же деле Synaptic, Muon и другие подобные программы — просто оболочки для apt, 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:

**E: 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 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 пользователям, привыкшим к Red Hat-совместимым дистрибутивам, трудно привыкнуть к особенностям Slackware. Возможно, «коренным» пользователям Slackware трудно привыкнуть к обращению с RPM-пакетами... Но утверждать не буду, потому что сам начинал свой путь линуксоида с дистрибутива Red Hat.

Однако однажды я не выдержал и установил на свой компьютер Slackware. Цель была одна — разобраться с установкой пакетов. Неужели все так сложно? Как ока-

залось, ничего сложного нет, если вникнуть в особенности Slackware, не известные пользователям Red Hat.

Прежде чем приступить к рассмотрению системы управления пакетами Slackware, 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 назвали 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 оставшихся 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 следует 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 зависит 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 не обязательно;
- *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).

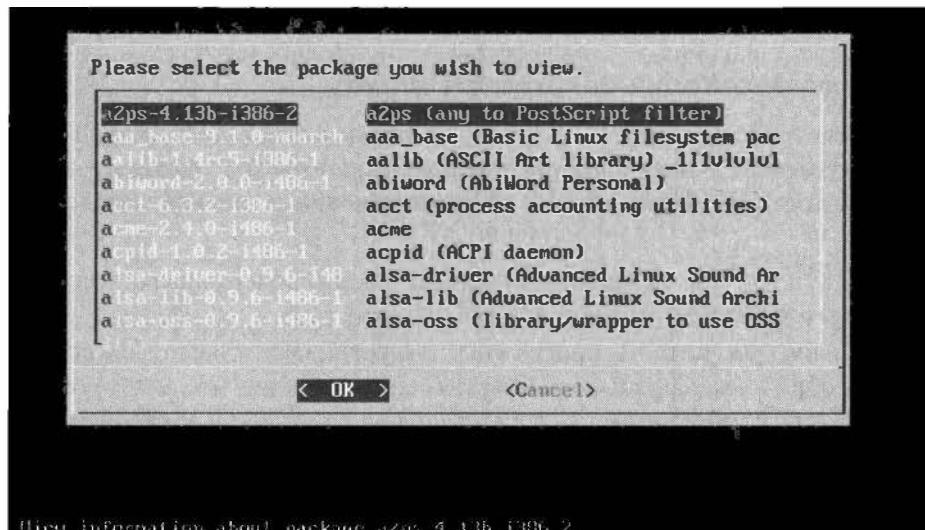


Рис. 7.8. Slackware: программа pkgtool



Рис. 7.9. Slackware:  
программа XPKGTOOL

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.

## Программа установки пакетов `installpkg`

Перед рассмотрением программы `installpkg` определимся со структурой пакета. Как уже было отмечено, пакет с программным обеспечением в Slackware — это обычный TGZ-архив, предназначенный для распаковки в корневой каталог файловой системы.

Вот пример структуры каталогов условного пакета, содержащего всего одну программу — `program`:

```
./  
usr/  
usr/bin/  
usr/bin/program  
usr/man/  
usr/man/man1  
usr/man/man1/program.1.gz  
install/  
install/doinst.sh
```

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 имён (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 `removepkg`

Формат вызова программы `removepkg` такой же, что и в предыдущем случае:

```
= removepkg <опция> <имя_пакета>
```

Опций у `removepkg` 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 для Linux распространяются 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 конфигурационный 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 1).

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:

= slackware upgrade script

## 7.10. Установка программ в openSUSE

#### 7.10.1. Менеджер пакетов zypper

Менеджер пакетов `zypper` дистрибутивов openSUSE 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)**

Параметр `baseurl` 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 `keerppackages`, 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

```
[download.opensuse.org-oss]
name=Основной репозиторий (OSS)
enabled=1
autorefresh=1
baseurl=http://download.opensuse.org/tumbleweed/repo/oss/
path=
type=yast2
keerppackages=0
```

Как видите, параметр `baseurl` 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-<дата>.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 `keerppackages` в 1, то для этого репозитория менеджер пакетов будет сохранять все загруженные пакеты. Если `keerppackages=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) для openSUSE 42.3 Leap.

**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-споме.упр, 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
sl	Выводит список используемых репозиториев
sa URL имя	Добавляет репозиторий (URL — адрес репозитория, а имя — имя, под которым он будет отображаться). Пример: zypper sa http://сервер/pub/linux/suse/SUSE-Linux-Updates
sd URL имя	Удаляет репозиторий. При удалении вы можете указать URL или имя репозитория
install пакеты	Устанавливает пакеты. Пример: zypper install mc Если нужно установить несколько пакетов, то имена пакетов разделяются пробелами
search маска	Ищет пакеты по маске. Маска — это часть имени (или полное имя) пакета. Пример: zypper search mc*
list-updates	Отображает доступные обновления
update пакет	Обновляет пакет. Если пакет не задан, обновляет всю систему
info пакет	Выводит информацию о пакете
remove пакет	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 *grm*.

Программой *zypper* (см. разд. 7.10.1) пользоваться решатся не все — она работает в командной строке. Программу *grm* (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 внешний носитель данных или сервер Интернета, намного удобнее воспользоваться графической оболочкой программы *zypper*: ввели название пакета, отметили его для установки и установили.

Для запуска графической оболочки *zypper* (графического менеджера пакетов) выполните команду основного меню **Компьютер | Центр программ**. Использовать графическую утилиту всегда просто, поэтому, думаю, вы разберетесь с ней без моих комментариев.

## 7.11. Снапы

Относительно недавно в Ubuntu появился новый инструмент — *снапы* (от англ. snap), но что это такое и как их использовать, знают далеко не все пользователи.

### 7.11.1. Введение в снапы

Представим вполне реальную ситуацию. Пользователь устанавливает самую новую на тот момент версию Ubuntu, пусть это будет версия 16.04 — именно в ней впервые появились снапы по умолчанию. По ряду причин пользователь отключает обновления. Такая ситуация не надумана. Мы все знаем, что после обновления системы Ubuntu, к сожалению, не всегда работает корректно и даже не всегда загружается.

Пользователя все устраивает, и он спокойно себе использует дистрибутив некоторое время. Через несколько лет пользователь хочет установить новую версию какого-либо приложения — например, браузера — и обнаруживает, что он не может этого сделать.

Почему? Да потому, что его дистрибутив устарел. Просто-напросто для установки приложения нужны новые версии библиотек, а для их установки нужно обновить уже установленные пакеты. Иногда процесс настолько масштабный, что приходится обновлять дистрибутив. А ведь мы знаем, что пользователь не хочет этого делать, да и опасно это: система может быть разрушена из-за нарушения связей между программами и библиотеками.

Вторая ситуация. Пусть у пользователя самая новая версия Ubuntu (сейчас это 19.04 — чтобы никто не упрекнул нас в надуманности ситуации). Мы знаем, что у традиционных пакетов есть зависимости и конфликты. Иногда нельзя установить то или иное приложение, поскольку его библиотеки конфликтуют с библиотеками, уже установленными в системе.

Обе проблемы настолько обширны, что разработчики Ubuntu изобрели снапы. Все мы знакомы с понятием пакета. Пакет содержит саму программу, а также различные вспомогательные файлы: документацию, ресурсы (картинки, например), файлы локализации, какие-то сценарии и т. п. Но пакет не содержит всего, что нужно для работы этой программы в системе. Например, если программе для работы нужна некая библиотека, то просто в пакете прописывается зависимость: для работы этого пакета нужно установить пакет с этой библиотекой. А при установке программы менеджер пакетов (apt) производит разрешение зависимостей — устанавливает все необходимые для работы этой программы пакеты.

С одной стороны, такой подход позволяет экономить место на диске. Ведь одну и ту же библиотеку не нужно устанавливать несколько раз. С другой стороны, он рождает уже описанные ранее проблемы установки новых пакетов на ранее сконфигурированные системы.

Снап — это лекарство от всей этой головной боли как пользователя, так и разработчика приложения. Снап представляет собой пакет, в который включены не

только программа, но и все необходимые для ее работы библиотеки. Получается, что все, что нужно для работы программы, и содержится в снапе.

Не нужно бояться, что система превратится в свалку, где перемешаны старые и новые версии библиотек. Снапы устанавливаются в отдельную папку в виде защищенного от записи образа. Изменения, которые должен внести снап в файловую систему (например, в каталог `/lib`), будут внесены в виртуальную файловую систему. Таким образом, установка снапа никак не повлияет на работу основной системы. Всё. Все проблемы с совместимостью версий разных приложений решены.

### 7.11.2. Работа со снапами

Первым делом нужно установить пакет `snapd`. В Ubuntu 16.04 и более новых версиях ничего делать не придется, поскольку этот пакет в них уже установлен. В Ubuntu 14.04 нужно ввести команду:

```
sudo apt install snapd
```

В других дистрибутивах команда будет другой — например, в Fedora команда установки выглядит так:

```
sudo dnf install snapd
```

Установив `snapd`, можно попробовать установить первый снап. Но пока мы не знаем, как он называется. Для поиска доступных снапов введите команду:

```
sudo snap find <название>
```

**Например:**

```
snap find hello
```

**Вывод будет примерно таким:**

hello-node-snap	1.0.2	bhdouglass	-	A simple hello world command
hello-mdeslaur	2.10	mdeslaur	-	GNU Hello, the "hello world" snap
hello-snap	0.01	muhammad	-	GNU hello-snap, the "Hello, Snap!" snap
hello	2.10	canonical	-	GNU Hello, the "hello world" snap
hello-world	6.3	canonical	-	The 'hello-world' of snaps
hello-sergiusens	1.0	sergiusens	-	hello world example
hello-gabriell	0.1	gabriell	-	Qt Hello World example
hello-bluet	0.1	bluet	-	Qt Hello World example
so-hello-world	0.2	shadowen	-	the old classic
hello-huge	1.0	noise	-	a really big snap

**В первой колонке приводится название найденного снапа — `hello`. Давайте его и установим:**

```
sudo snap install hello
```

Как видите, команда установки снапа аналогична команде установки пакета, только вместо команды `apt` используется команда `snap`.

Кстати, после установки снапа можно запустить имеющуюся в нем программу:

```
hello  
Hello, world!
```

Просмотреть установленные снапы можно командой `snap list`:

```
snap list  
Name      Version   Rev  Developer  Notes  
hello      2.10      20   canonical  -  
core       16.04.1423 canonical  -
```

Снап `core` — это базовая система снапов, среди всего прочего он как раз и содержит тот самый демон `snapd`, который вы установили ранее.

Как и пакеты, снапы можно обновлять. Например:

```
sudo snap refresh hello
```

Эта команда обновит снап `hello`, если для него доступны обновления. Это очень удобно — вы обновляете не только приложение, но и все необходимые для его работы библиотеки.

Для обновления всех снапов используется другая команда:

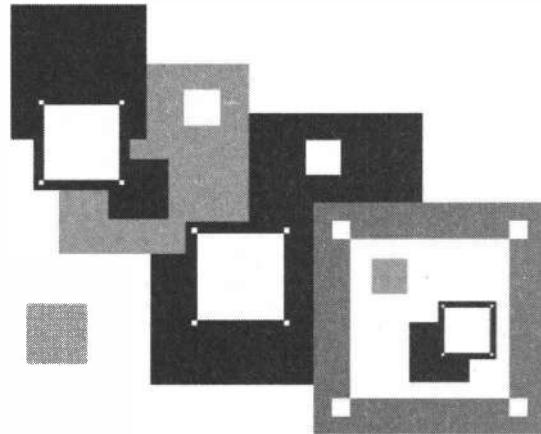
```
sudo snap refresh
```

Теперь вы знаете, для чего используются снапы и что это вообще такое. А вот обновлять систему целиком или использовать снапы — каждый решает сам.

\* \* \*

На этом обзор систем управления пакетами можно считать завершенным. Надеюсь, я ничего не забыл!





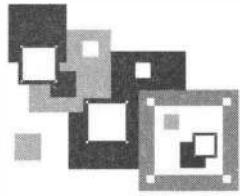
## ЧАСТЬ III

### Настройка сети и Интернета

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.

Соответственно, *третья часть книги* посвящена настройке в Linux локальной сети, интернет-соединения Wi-Fi, настройке VPN-соединения, а также интересной задаче объединения интернет-каналов.





## ГЛАВА 8

# Настройка локальной сети

## 8.1. Локальная сеть с использованием технологии Gigabit Ethernet

Сетевых технологий существует много, но в этой книге мы займемся настройкой локальной сети, построенной по технологии Gigabit Ethernet. Зато рассмотрим мы ее полностью: от обжатия кабеля до конфигурирования сети.

Основные характеристики стандарта Gigabit Ethernet:

- скорость передачи данных: 1000 Мбит/с;
- метод доступа к среде передачи данных: CSMA/CD;
- среда передачи данных: витая пара UTP 5-й или 6-й категории, оптоволоконный кабель;
- максимальное количество компьютеров: 1024;
- максимальное расстояние зависит от используемого стандарта и среды передачи данных. Так, для 1000Base-T (четыре неэкранированные витые пары 5-й категории) максимальное расстояние составляет 100 м.

Прежде всего нужно убедиться, что компьютеры, предназначенные для соединения в сеть, оснащены *сетевыми адаптерами*, поддерживающими технологию Gigabit Ethernet. Как правило, сейчас сетевые адAPTERы интегрированы в материнскую плату, и устанавливать их отдельно необходимости нет. Однако все еще встречаются иногда материнские платы со старыми адаптерами Fast Ethernet. Они будут работать в сети Gigabit Ethernet, но скорость передачи данных окажется в 10 раз ниже (максимальная — 100 Мбит/с). При необходимости адAPTERы, поддерживающие гигабитную сеть, можно приобрести отдельно (рис. 8.1). Стоят они не столь и дорого — от 700 рублей за штуку. Так же невысока и цена на коммутаторы для гигабитной сети. Неплохой неуправляемый коммутатор на 8 портов 10/100/1000BaseT обойдется вам не более чем в 1,5 тыс. рублей.

Установка сетевого адаптера проблем не вызывает — просто вставьте его в свободный разъем шины PCI (все адAPTERы Gigabit Ethernet выполнены в виде плат расширения именно для шины PCI). Существуют также сетевые адAPTERы, подключаемые к компьютеру по USB (рис. 8.2), — стоят они не сильно дороже PCI-

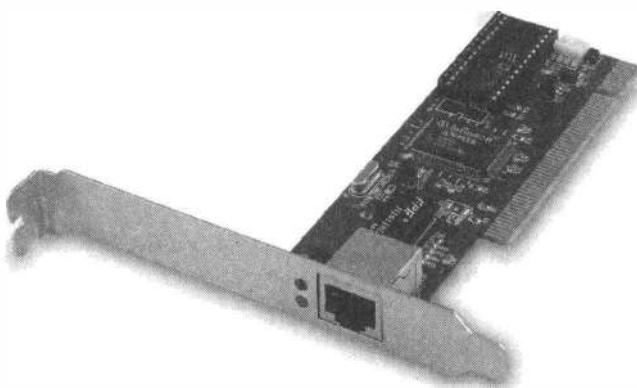


Рис. 8.1. Сетевой PCI-адаптер Gigabit Ethernet

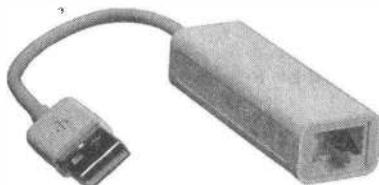


Рис. 8.2. Сетевой адаптер USB

адаптеров, и нет никаких оснований подозревать, что при работе в Linux с ними возникнут какие-либо проблемы.

Ясно, что устанавливать сетевой PCI-адаптер полагается при выключенном компьютере — шина PCI пока еще не поддерживает «горячую замену». Собрав и включив компьютер, подключите к сетевому адаптеру *коннектор* (специальный наконечник) сетевого кабеля.

Сетевые кабели различной длины (от 0,5 до 5 м), оснащенные коннекторами (патчкорды), 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 наконечники-коннекторы (см. далее).

Выполняются эти манипуляции, как правило, администратором сети. Вы сами администрируете свою сеть, но не знаете, как это сделать? Нет проблем, сейчас разберемся. Для создания сети Ethernet вам потребуются следующие устройства:

- сетевые адAPTERы — 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 показан 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;

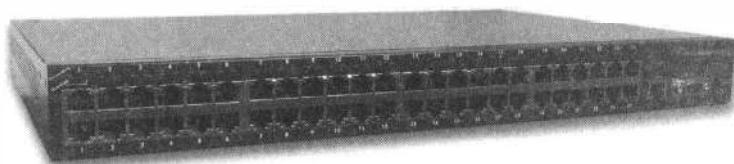


Рис. 8.3. Коммутатор (switch)

- сетевой кабель (витая пара 5-й категории или лучше) — 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 обжатия коннекторов витой пары — хороший инструмент стоит относительно дорого (примерно как коммутатор), а плохой лучше не покупать<sup>1</sup>. Если не хотите выкладываться, одолжите такие щипцы у кого-нибудь на пару дней.

Теперь приступим к самому процессу обжатия. Внутри кабеля идут четыре витые пары проводов (всего восемь), причем у каждого провода своя цветовая маркировка. Суть процесса обжатия 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 станет работать только на скорости 10–100 Мбит/с.

Проверить, правильно ли вы обжали кабель, очень просто — обратите внимание на коммутатор: возле каждого его порта имеются 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

<sup>1</sup> Помню, как-то дешевый инструмент обжимал не все жилы, и некоторые из них приходилось дожимать плоской отверткой. Удовольствие еще то, особенно когда нужно обжать более 30 коннекторов.

переобжать кабель — отрезать плохо обжатые коннекторы и обжать концы кабеля новыми коннекторами заново.

Как видите, в процессе обжатия 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/host.conf	Содержит параметры разрешения доменных имен. Например, директива order hosts,bind 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 multi on 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 этого узла
/etc/hosts.deny	Содержит IP-адреса узлов, которым запрещен доступ к сервисам этого узла
/etc/hostname	В Debian/Ubuntu содержит имя узла
/etc/iftab	Содержит таблицу интерфейсов, т. е. соответствия 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 старых версиях Debian и Ubuntu используется для ручной настройки сетевых интерфейсов (не с помощью NetworkManager).

Таблица 8.2 (окончание)

Файл	Описание
/etc/rc.config	В старых версиях SUSE (не openSUSE) содержит имя компьютера, IP-адрес интерфейса и другую сетевую информацию
/etc/resolv.conf	Задает IP-адреса серверов DNS. Формат файла прост: <i>nameserver IP-адрес</i> 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: <a href="http://www.dkws.org.ua/index.php?page=show&amp;file=a/ubuntu/static-dns-ubuntu9">http://www.dkws.org.ua/index.php?page=show&amp;file=a/ubuntu/static-dns-ubuntu9</a>
/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 символьного имени сервиса (например, <i>pop3</i> ) 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- <i>имя</i>	Параметры конкретного сетевого интерфейса. Например, параметры интерфейса <i>eth0</i> хранятся в файле /etc/sysconfig/network-scripts/ifcfg- <i>eth0</i> (дистрибутив Fedora)
/etc/sysconfig/network/ ifcfg- <i>имя</i>	Параметры конкретного сетевого интерфейса ( <i>имя</i> — 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` вместо привычного имени `eth0` (рис. 8.4), что не есть хорошо... А в некоторых дистрибутивах сетевая карта получила еще менее благозвучное название — `enp3s0`. В общем, в связи с полным переходом новых дистрибутивов на `udev` (см. главу 4) понятная с первого взгляда строгость в назначении имен сетевых адаптеров (когда первая сетевая плата называлась `eth0`, вторая `eth1` и т. д.) исчезла.

```
[den@localhost ~]$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 192.168.84.153 netmask 255.255.255.0 broadcast 192.168.84.255
              inet6 fe80::d903:53b5:41af:b8b2 prefixlen 64 scopeid 0x20<link>
                ether 00:0c:29:8a:e9:f8 txqueuelen 1000 (Ethernet)
                  RX packets 4462 bytes 5917942 (5.6 MiB)
                  RX errors 0 dropped 0 overruns 0 frame 0
                  TX packets 913 bytes 78053 (76.2 KiB)
                  TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

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 1000 (Local Loopback)
                RX packets 4 bytes 240 (240.0 B)
                RX errors 0 dropped 0 overruns 0 frame 0
                TX packets 4 bytes 240 (240.0 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 адAPTERы, встроенные в материнскую плату (`ID_NET_NAME_ONBOARD`), носят теперь название типа `eno1`. 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 коннектора (`ID_NET_NAME_PATH`), будут выглядеть как `enp2s0`. А самые сложные и «страшные» имена у сетевых адAPTERов, 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 использовании DHCP, возможно, его не придется настраивать вовсе).



Рис. 8.5. Передача ядру параметра net.ifnames=0



Рис. 8.6. Fedora 33: сетевым устройствам вернулись старые имена

Второй способ — передать ядру параметр `net.ifnames=0` (рис. 8.5) и вернуть старую схему именования сетевых устройств<sup>1</sup>. Как показано на рис. 8.6, после передачи ядру параметра `net.ifnames=0` сетевая карта стала называться по-старому: `eth0`.

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`<sup>2</sup> или `service network status` (рис. 8.7) отсутствует интерфейс `lo`, значит, сервис `network` остановлен. Такие случаи — редкость, но все 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.

<sup>1</sup> А чтобы изменения сохранились на постоянной основе, нужно отредактировать файл `/etc/default/grub`, а затем ввести команду: `grub2-mkconfig -o /boot/grub2/grub.cfg`.

<sup>2</sup> В Ubuntu для использования команды `ifconfig` нужно установить пакет `net-tools`.

```
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

### 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 маршрутизаторе Wi-Fi, 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 маршрутизатором Wi-Fi. 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 маршрутизатор Wi-Fi.

Вот только на предприятиях 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 (рис. 8.8) — запускается он через команду `sudo` или `gksudo`. Кнопка + (Добавить), расположенная в левом нижнем углу окна конфи-

гуратора, служит для создания нового соединения (в главе 9 будет показано, как создать соединение Wi-Fi), а Ethernet-соединение, как правило, уже создано, поэтому для изменения его параметров нужно нажать там же кнопку с изображением шестеренки (Изменить), в результате чего откроется соответствующее окно.



Рис. 8.8. Ubuntu 20.10: конфигуратор `nm-connection-editor`

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.

Обычно сеть настраивается автоматически, поэтому в поле **Method** (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.

В поле **Серверы DNS** вкладки **Параметры IPv4** 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 в поле **Method** 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 возмож-

ность используется редко. Дополнительную информацию о VLAN можно получить в моих статьях по следующим адресам:

- <http://www.xakeper.ru/magazine/xa/121/122/1.asp>;
- <http://www.dkws.org.ua/index.php?page=show&file=a/ubuntu/network-interfaces>.

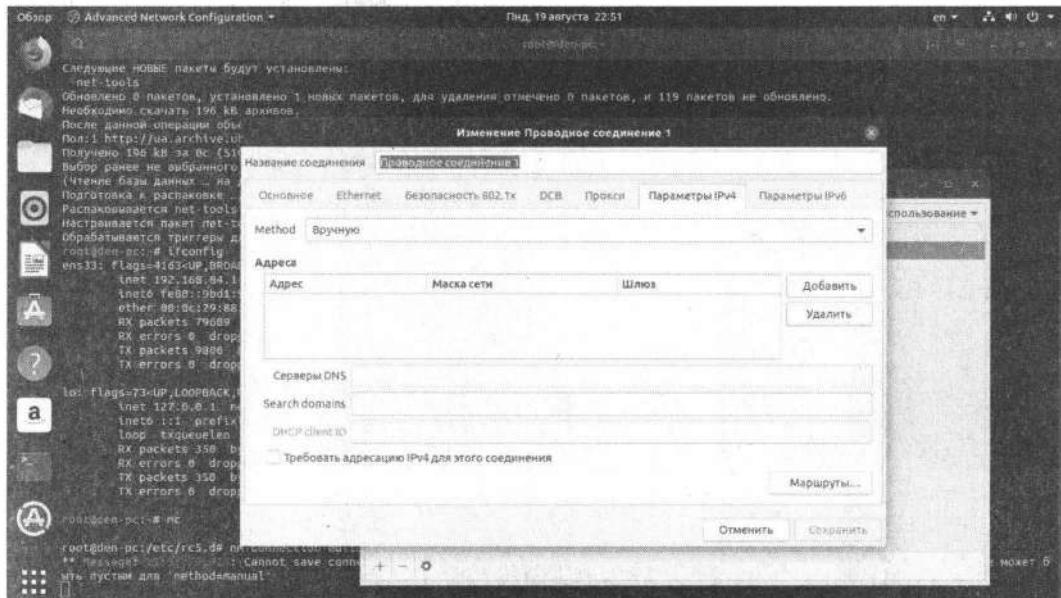


Рис. 8.9. Ubuntu 20.10: конфигуратор nm-connection-editor, вкладка Параметры IPv4

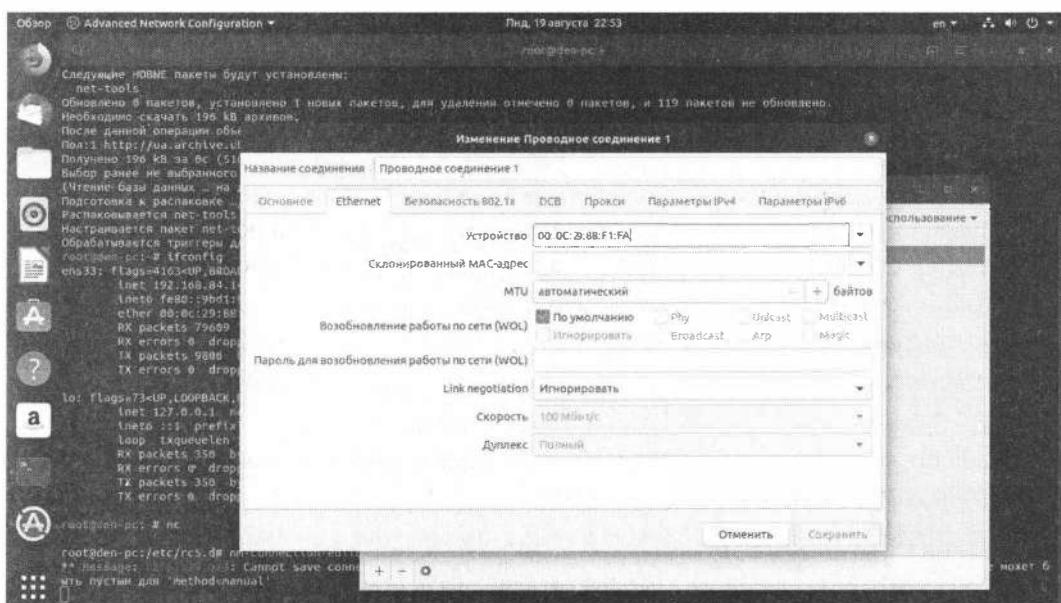


Рис. 8.10. Ubuntu 20.10: конфигуратор nm-connection-editor, вкладка Ethernet

## 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.

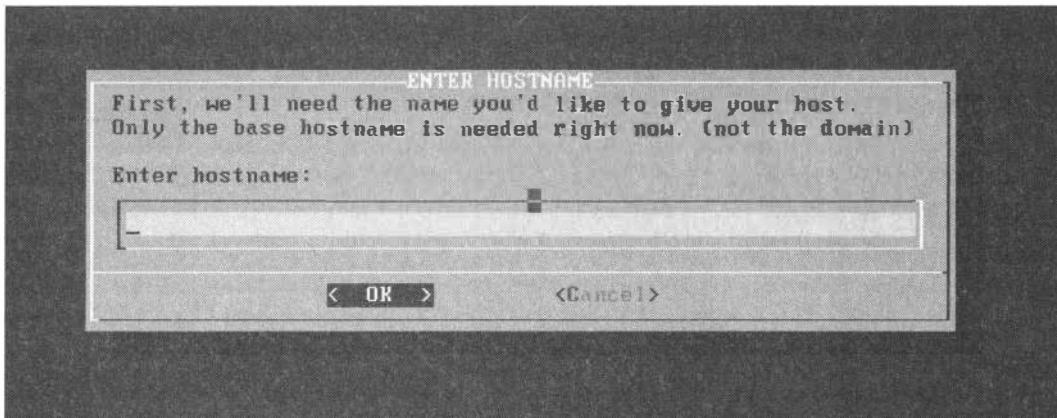


Рис. 8.11. Slackware: конфигуратор netconfig

## 8.6. Утилиты для диагностики соединения

Причинами отказа сети могут стать 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»).

```

user@user-desktop:~$ sudo pon dsl-provider
Password:
Plugin rp-pppoe.so loaded.
user@user-desktop:~$ ifconfig
eth0      Link encap:Ethernet HWaddr 00:00:87:88:BC:96
          inet6 addr: fe80::20d:87ff:fe88:bc96/64 环回子网: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:0xe000

lo       Link encap:Local Loopback
          inet addr:127.0.0.1 Mask:255.0.0.0
          inet6 addr: ::1/128 环回子网: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:148B 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

Итак, если интерфейс не «поднят», нужно просмотреть файл протокола /var/log/messages или /var/log/syslog (все зависит от дистрибутива и его версии) сразу после попытки установки соединения:

```
tail -n 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 14:28:33 user-desktop pppd[5176]: Plugin rp-pppoe.so loaded.
Feb  6 14:28:33 user-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
                                         versio n 2.4.2
Feb  6 14:28:33 user-desktop pppd[5183]: pppd 2.4.4b1 started by root, uid 0
Feb  6 14:28:33 user-desktop pppd[5183]: PPP session is 2838
Feb  6 14:28:33 user-desktop kernel: [17179852.984000] NET: Registered protocol
                                         family 24
Feb  6 14:28:33 user-desktop pppd[5183]: Using interface ppp0
Feb  6 14:28:33 user-desktop pppd[5183]: Connect: ppp0 <--> eth0

```

```

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
Feb 6 14:28:33 user-desktop pppd[5183]: peer from calling number 00:15:F2:60:28
                                         :97 authorized
Feb 6 14:28:33 user-desktop pppd[5183]: local IP address 193.254.218.243
Feb 6 14:28:33 user-desktop pppd[5183]: remote IP address 193.254.218.129
Feb 6 14:28:33 user-desktop pppd[5183]: primary DNS address 193.254.218.1
Feb 6 14:28:33 user-desktop pppd[5183]: secondary DNS address 193.254.218.27

```

Первая строчка — 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 выводятся 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 изменили), 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 устройство в порядке, значит, по-

вреждена 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` (прервать 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=1 ttl=58 time=30.7 ms  
64 bytes from wdt.org.ru (213.186.114.75): icmp_seq=2 ttl=58 time=24.8 ms  
64 bytes from wdt.org.ru (213.186.114.75): icmp_seq=5 ttl=58 time=12.2 ms  
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=7 ttl=58 time=19.3 ms  
64 bytes from wdt.org.ru (213.186.114.75): icmp_seq=9 ttl=58 time=29.0 ms  
...
```

Здесь все нормально. Но иногда ответы от удаленного сервера то приходят, то не приходят. Чтобы узнать, в чем причина (где именно теряются пакеты), нужно выполнить трассировку узла:

```
tracepath dkws.org.ua
```

### Команды трассировки

В некоторых дистрибутивах вместо команды `tracepath` используется команда `traceroute`, а в Windows — `tracert`.

Из вывода команды `tracepath` (рис. 8.13) сразу видно, что с прохождением пакетов до удаленного узла имеются определенные проблемы, т. е. пакеты по пути теряются.

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-1-ge.interfaces.dc.utel.ua`, а после него движение пакетов прекращается.

Если соединение установлено (о чём свидетельствует наличие «поднятого» интерфейса в выводе `ifconfig`), а веб-страницы не открываются, попробуйте пропинговать 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 веб-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

```

Файл Правка Вид Терминал Вкладки Справка
user@user-desktop:~$ traceroute 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-m71-1.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

и выясните причину этого, а еще лучше, уточните IP-адреса серверов DNS и пропишите их в файле `/etc/resolv.conf`. Формат этого файла прост:

`nameserver IP-адрес`

Например:

```
nameserver 193.254.218.1
nameserver 193.254.218.27
```

Всего там можно указать до четырех серверов DNS.

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 утилиту `regedit`. С другой

стороны, редактирование конфигурационных файлов позволяет глубже познать Linux. Если вам интересно, в какие файлы сохраняются сетевые настройки после нажатия кнопки **OK** 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 чтению 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
BROADCAST=192.168.0.255 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='dhcpc'  
IPADDR=''  
MTU=''  
NAME='79c970 [PCnet32 LANCE]'  
NETMASK=''  
NETWORK=''  
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 `ne-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 `no`).

Также можно установить значение `no` 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 старых версий Debian/Ubuntu

Основной конфигурационный файл старых версий 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 по адресу: <https://www.dkws.org.ua/a/ubuntu/network-interfaces.html>, 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, доступной по адресу: <https://www.dkws.org.ua/a/ubuntu/static-dns-ubuntu9.html>.

#### 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 22–30 эта команда сработала без прав `root`, что и показано на рис. 8.14.

Существует также команда `hostname`, но она позволяет лишь изменить имя компьютера до перезагрузки, а после перезагрузки будет восстановлено его старое имя.

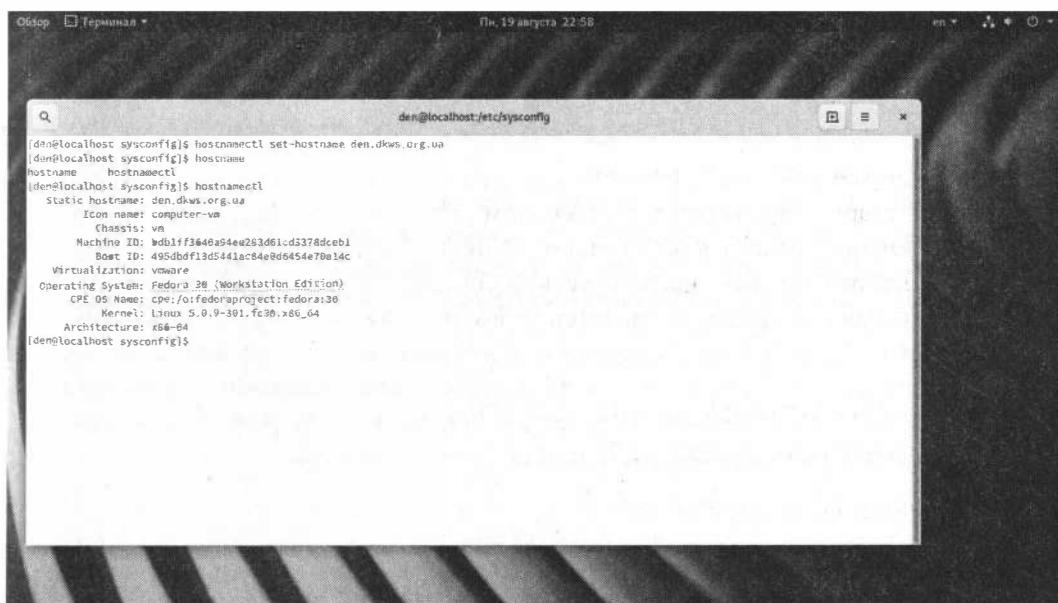


Рис. 8.14. Fedora 33: установка параметров узла

## 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 веб-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-FD 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-FD 100baseTx-HD 10baseT-FD 10baseT-HD
advertising: 100baseTx-FD 100baseTx-HD 10baseT-FD 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:

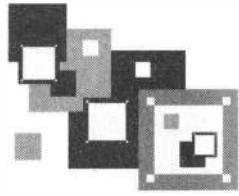
```
# mii-tool -force=10baseT-FD eth0
```

## 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-смартфона. А DSL-провайдеры все чаще и чаще вместо обычных DSL-модемов устанавливают DSL-маршрутизаторы с функциями Wi-Fi, что очень удобно. 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 незадорого купить к нему внешний адаптер Wi-Fi и пользоваться всеми преимуществами скоростного беспроводного Интернета.

### ***Настройка ADSL-доступа к Интернету***

Материал, посвященный настройке ADSL-доступа к Интернету, вы найдете в папке *Дополнения* сопровождающего книгу файлового архива (см. [приложение](#)).

В этой главе мы не станем подробно рассматривать процесс настройки беспроводной сети (об этом читайте в других моих книгах, настройкам сети посвященных) — да и не имеет этот процесс прямого отношения к настройке Linux. К тому же настройки беспроводных маршрутизаторов различаются в зависимости от их производителей 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 адAPTERы так же, как и обычные сетевые адAPTERы. Поэтому все, что вам нужно сделать, — это ока-

заться в зоне действия беспроводной сети и ввести пароль, если доступ к сети запаролен.

Далее мы рассмотрим процесс настройки беспроводного подключения в графической среде GNOME 3.38, т. е. в последних версиях дистрибутивов Ubuntu, Fedora и openSUSE.

Щелкните на значке NetworkManager , и вы увидите меню, позволяющее подключиться к беспроводной сети (рис. 9.1). Раскройте группу **Wi-Fi сеть не подключена** — в ней имеются команды **Выбрать сеть**, **Выключить** и **Параметры Wi-Fi**. Команда **Выключить** позволяет выключить адаптер Wi-Fi устройства. Для его включения нужно использовать кнопку на корпусе ноутбука (или предусмотренную комбинацию клавиш) или же извлечь и снова установить USB-адаптер — при использовании внешнего адаптера Wi-Fi.

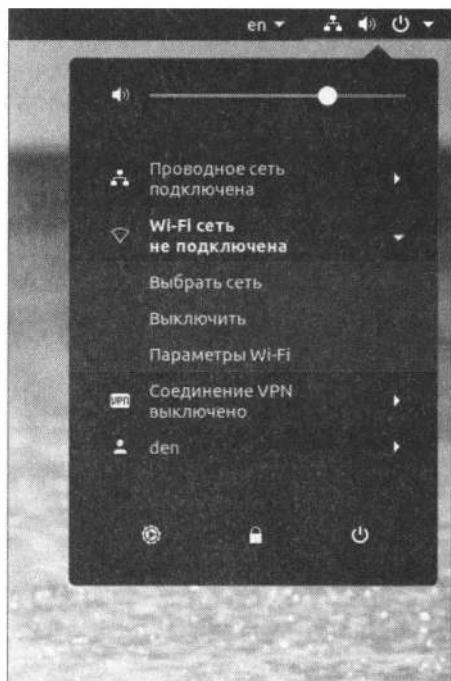


Рис. 9.1. Ubuntu: меню NetworkManager



Рис. 9.2. Ubuntu: выбор беспроводной сети

Выберите команду **Выбрать сеть** — откроется окно с доступными для выбора беспроводными сетями (рис. 9.2). Возле каждой беспроводной сети отображается индикатор уровня сигнала — чем больше он заполнен, тем ближе вы к беспроводному маршрутизатору. Моя сеть называется **dhsilabs**, ее я и выбрал. Посмотрите внимательно на значок индикатора — если возле него имеется маленький символ замка, то сеть закрыта — для доступа к ней нужно будет ввести пароль (рис. 9.3).

И если введенный пароль правильный, в меню NetworkManager вместо группы **Wi-Fi сеть не подключена** вы увидите название сети, к которой подключились (рис. 9.4).

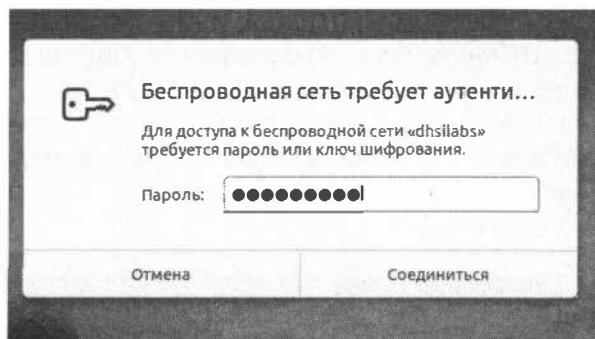


Рис. 9.3. Ubuntu: ввод пароля для доступа к сети

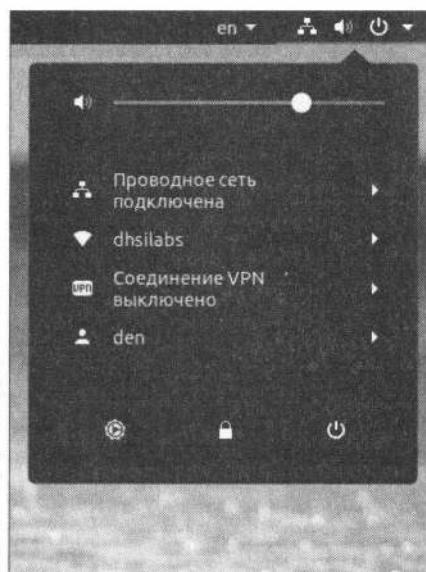


Рис. 9.4. Ubuntu: соединение с беспроводной сетью установлено

Разверните группу параметров с именем беспроводной сети и выберите команду **Параметры Wi-Fi** — появится список беспроводных сетей (рис. 9.5). Активная сеть (та, к которой вы подключены) отмечена в нем «галочкой». Для нее также доступна кнопка с изображением шестеренки — нажмите ее, и откроется окно настройки параметров этой сети (рис. 9.6).

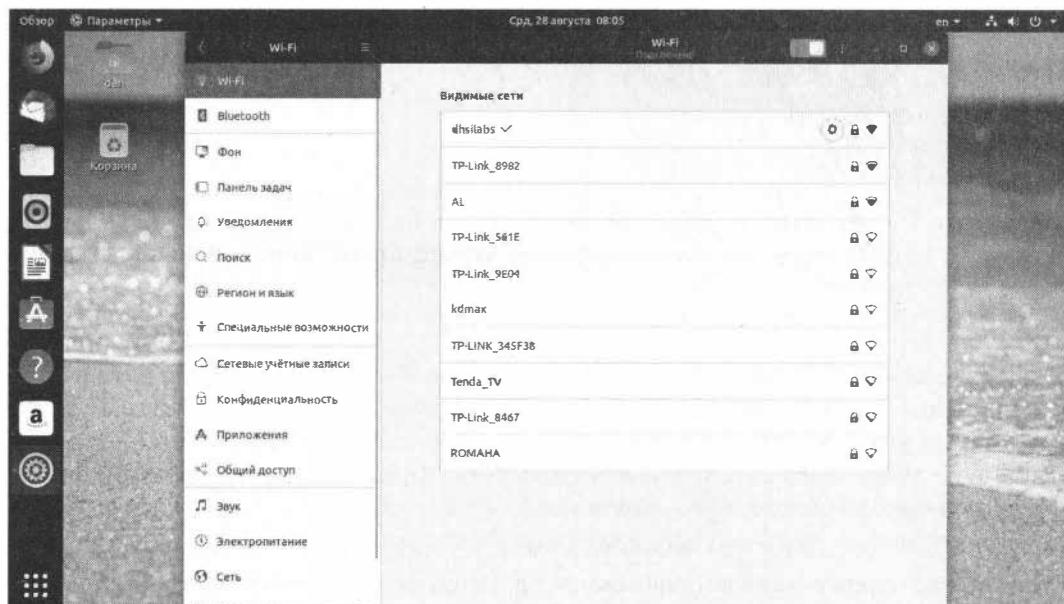


Рис. 9.5. Ubuntu: активная сеть в списке сетей отмечена «галочкой»

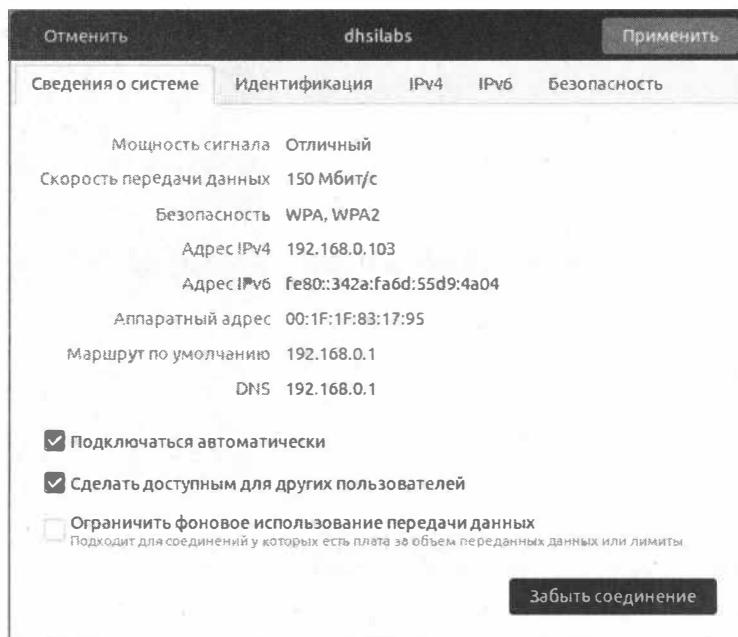


Рис. 9.6. Ubuntu: вкладка Сведения о системе окна параметров активной сети

На вкладке **Сведения о системе** этого окна здесь отображается информация о подключении: мощность сигнала, скорость передачи данных, присвоенный IP-адрес, MAC-адрес (аппаратный адрес) адаптера и т. п.

Будет ли система автоматически подключаться к этой сети, когда вы окажетесь в поле ее действия, зависит от переключателя **Подключаться автоматически**.

Параметр **Сделать доступным для других пользователей** позволяет сделать доступным это соединение другим пользователям системы, чтобы они тоже могли пользоваться этой беспроводной сетью, — полезно, если за компьютером работаете не только вы.

Чтобы система экономила трафик — например, когда вы подключаетесь через сеть Wi-Fi, созданную вашим смартфоном, и ваш мобильный тариф предусматривает плату за трафик, будет полезен параметр **Ограничить фоновое использование передачи данных**.

Пароль для доступа к сети можно изменить на вкладке **Безопасность**, вкладка **Идентификация** (рис. 9.7) позволяет задать имя беспроводной сети (SSID) и изменить MAC-адрес адаптера, если это вам нужно. Вкладка **IPv4** (как и **IPv6**) содержит параметры протокола IP, которые вам в 99% случаев не придется изменять.

Если вы что-либо изменили в параметрах беспроводной сети, не забудьте нажать кнопку **Применить** в верхнем правом углу окна.

Параметры всех ваших соединений хранятся в каталоге `/etc/NetworkManager/system-connections`. В нем вы найдете файлы с параметрами конкретных подключений (название файла соответствует названию подключения). В листинге 9.1 приведен файл параметров соединения для моей беспроводной сети.

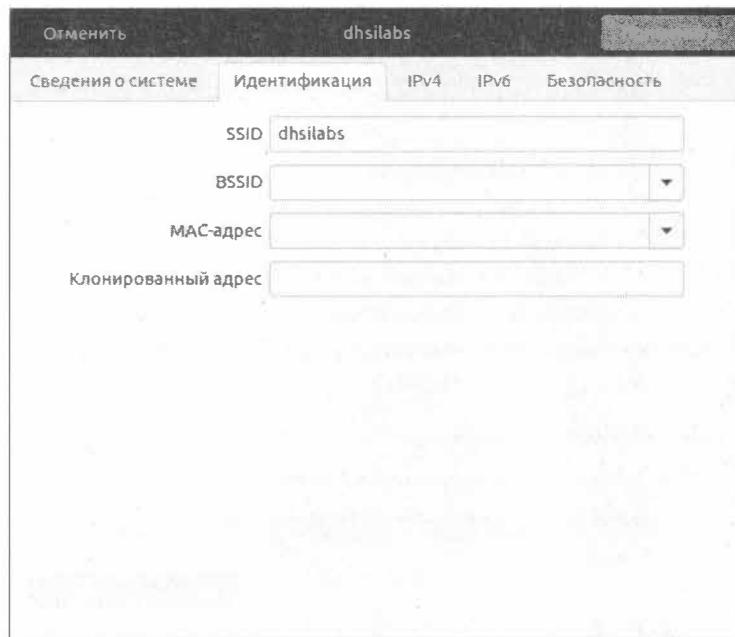


Рис. 9.7. Ubuntu: вкладка Идентификация окна параметров активной сети

#### 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=infrastructure
mac-address=0:1f:1f:83:17:95
security=802-11-wireless-security

[802-11-wireless-security]
key-mgmt=wpa-psk
auth-alg=open
psk=12345678

[ipv4]
method=auto

[ipv6]
method=auto
```

Обратите внимание: пароль доступа к сети задается параметром `psk` (в приведенном примере пароль — это строка 12345678).

## 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 находится совсем 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 со смартфона) 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 имени сети (SSID) выключено, и поэтому ее не видно в списке. Для подключения к такой сети нужно выбрать команду **Подключиться к скрытой сети** из меню окна параметров Wi-Fi (рис. 9.8) и ввести SSID и пароль для доступа к ней.

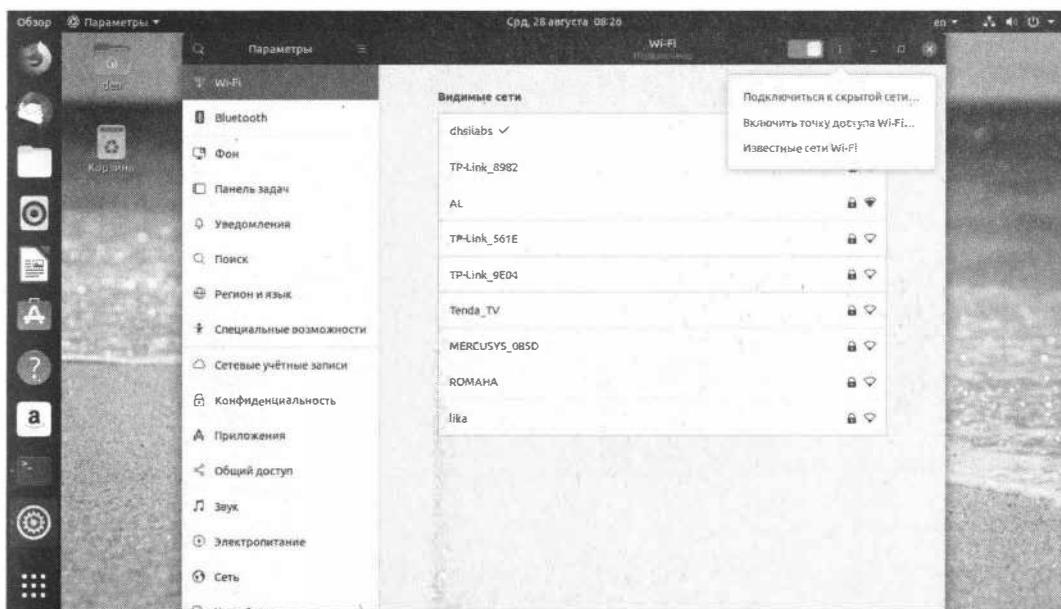


Рис. 9.8. Ubuntu: подключение к скрытой сети

Самый плохой вариант, когда в других ОС подключиться получается, а в Linux — нет. Такие случаи весьма 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 индивидуален 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/4G. Дело в том, что такая настройка сейчас неактуальна. Практически у всех пользователей Linux имеется смартфон на базе Android. А в любом современном Android-смартфоне есть возможность включить точку доступа Wi-Fi. В результате ваше мобильное EDGE/3G/4G-соединение будет «расширяться» 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. 9.9) и выберите опцию **Мобильная точка доступа и modem**.

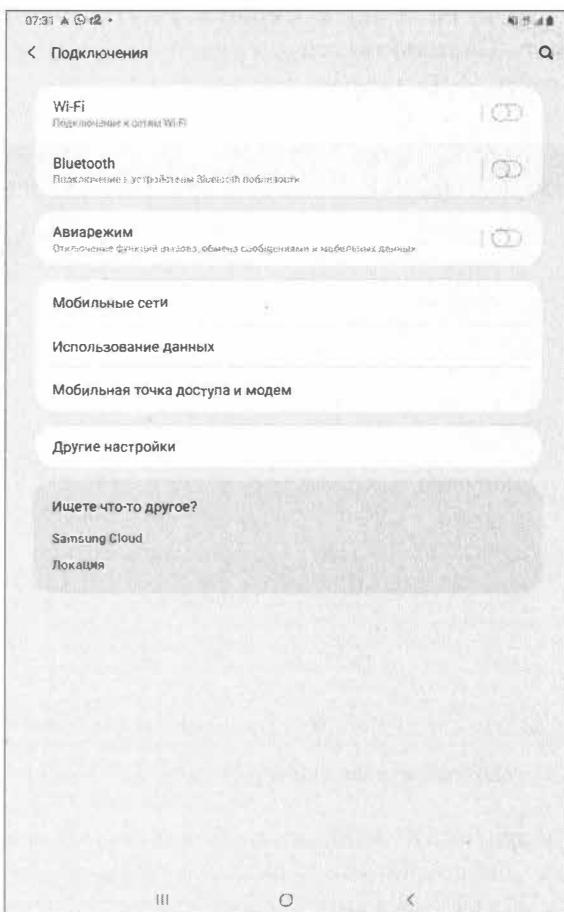


Рис. 9.9. Android 10:  
меню Подключения

В открывшемся окне **Мобильная точка доступа и модем** (рис. 9.10) выберите опцию **Мобильная точка доступа** — появится экран для ввода параметров точки доступа: имени точки (SSID) и пароля (рис. 9.11). Все, что вам остается, — ввести эти параметры и включить функцию **Мобильная точка доступа** с помощью соответствующего переключателя.

В iOS нужно открыть приложение **Настройки**, перейти в раздел **Режим модема**, включить параметр **Разрешать другим** и установить пароль Wi-Fi. После этого появится новая сеть Wi-Fi с именем, указанным в настройках устройства (**Настройки | Основные | Об этом устройстве | Имя**).

Используя введенные вами параметры точки доступа на смартфоне, вы сможете настроить компьютер под управлением Linux на соединение с этой точкой доступа.

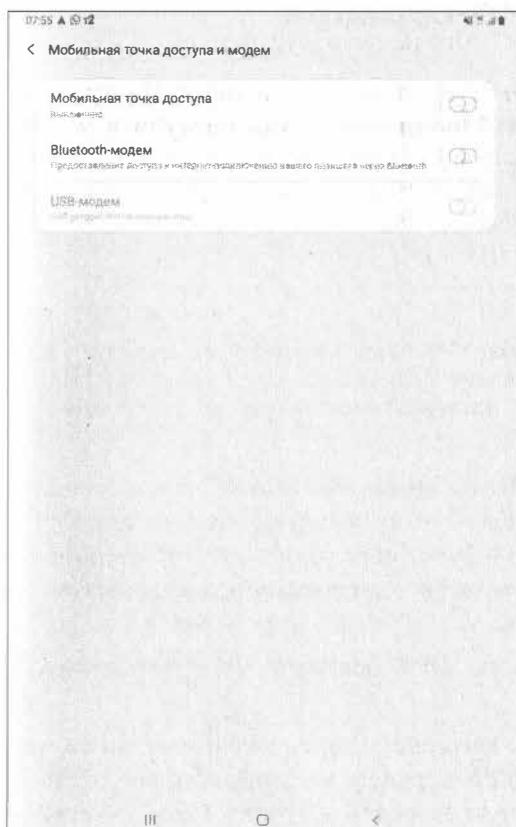


Рис. 9.10. Android 10:  
окно Мобильная точка доступа и модем

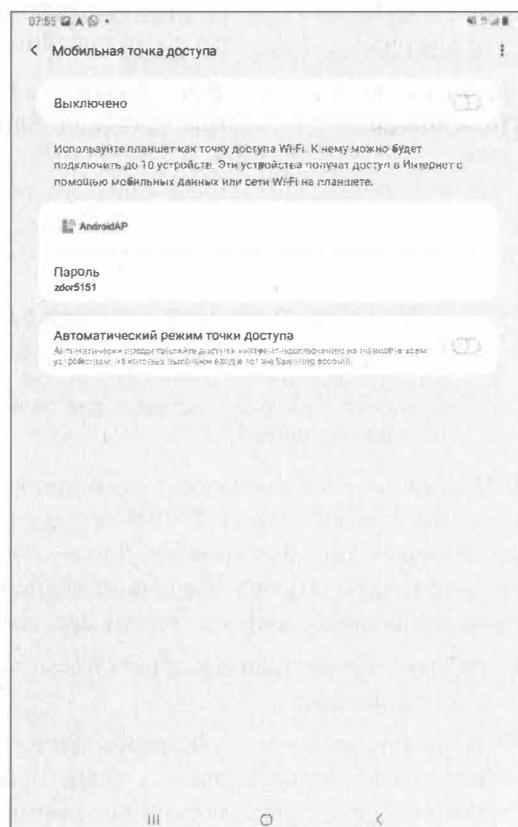
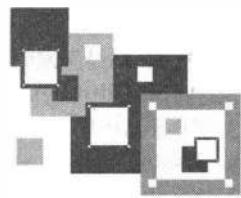


Рис. 9.11. Android 10:  
параметры мобильной точки доступа

# 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 веб-серфинга, просмотра видео онлайн, видеоразговоров в Skype или WhatsApp. 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;
- *ограничения на загрузку файлов* — некоторые бесплатные сервисы имеют ограничения на размер загружаемых файлов. Например, при загрузке больших фай-

лов (несколько десятков мегабайт) через VPN-сервис браузера Орга вы иногда можете получить сообщение Прервано: файл загружен не полностью. Однако достаточно нажать кнопку Возобновить, и загрузка продолжится. Если файл «весит» несколько сотен мегабайт, такое ограничение не очень приятно, но все же позволяет работать, чего не скажешь, если тот или иной бесплатный VPN вообще не позволит загружать большие объемы информации;

- *ограничения на использование портов* — некоторые VPN-сервисы закрывают определенные порты — например, порты отправки почты 25, 465 и т. п. Это означает, что вы не сможете воспользоваться для переписки привычной вам почтовой программой, и почту придется отправлять только с помощью веб-интерфейса. И хорошо, если у вашего почтового ящика есть веб-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 их.

В предыдущем издании этой книги рассматривался VPN-провайдер SecurityKISS. К сожалению, он прекратил деятельность. Поэтому мне пришлось поискать какой-нибудь другой приемлемый вариант. Требования к нему следующие:

- наличие удобного Linux-клиента. Кстати, это одна из причин, по которой в предыдущем издании рассматривался VPN-сервис SecurityKISS;
- наличие бесплатного тарифа или хотя бы продолжительного trial-периода, позволяющего протестировать соединение;
- доступные тарифы.

VPN-провайдеров много, но, как оказалось, далеко не все они предоставляют удобные Linux-клиенты, которые бы работали по принципу: нажал кнопку, и соединение установилось. Некоторые провайдеры (например, SurfShark) предоставляют столь замысловатые консольные клиенты, разбираться с которыми решатся только опытные пользователи. Однако опытные пользователи могут воспользоваться возможностями FreeOpenVPN (несмотря на сказанное ранее) или же направлять весь свой трафик через сеть Tor. В результате они получат бесплатное VPN-соединение и неудобный VPN-клиент. Платить же свои кровные за неудобный клиент не очень хочется.

Муки выбора привели меня к сервису KeepSolid VPN Unlimited (<https://www.vpnunlimitedapp.com/ru/>). Он полностью соответствует приведенным ранее критериям:

- обладает удобным VPN-клиентом;
- предоставляет тестовый период 7 дней;
- стоимость от 5 лолларов США в месяц (при оплате за год) или же 200 долларов одноразово.

Дорого? Максимальный тарифный план у SecurityKISS назывался EMERALD и стоил 90 евро в год. VPN Unlimited обойдется вам в 60 долларов за год, при этом у вас будут неограниченные трафик и скорость, доступ к 500+ серверам в 80+ локациях и поддержка до 10 устройств на аккаунт. Другими словами, за 60 долларов в год с его помощью можно защитить все домашние устройства (и даже устройства небольшой фирмы).

## 10.2. Настройка VPN-подключения

Настройку VPN-подключения мы рассмотрим на примере провайдера KeepSolid VPN Unlimited и в дистрибутиве Ubuntu, как одном из самых популярных для настольных компьютеров.

Первым делом нужно удостовериться, что вы используете нужные DNS-серверы. Как бы там ни было, но вам следует задать 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).

```

root@den-pc:~$ sudo apt install resolvconf
Чтение списков пакетов... готово
Построение дерева зависимостей
Чтение информации о состояниии... готово
Следующие новые пакеты будут установлены:
resolvconf
Обновлено 0 пакетов, установлено 1 новых пакетов, для удаления отмечено 0 пакетов, и 119 пакетов не обновлено.
Нет необходимости скачивать 461 kB в архивах.
После данной операции объем занятого дискового пространства возрастет на 185 kB.
Пакет http://releases.ubuntu.com/ubuntu/disco/universe amd64 resolvconf all 1.79ubuntu13 [46,1 kB]
Получены 48,1 kB из 88 (173 kB/s)
Предварительная загрузка пакетов...
Выбор ранее не выбранного файла resolvconf.
Установка базы данных на данный момент установлено 205886 файлов.
Пакет отбрасывается resolvconf_1.79ubuntu13_all.deb...
Распаковывается resolvconf_1.79ubuntu13...
Настраивается пакет resolvconf (1.79ubuntu13)...
Создан symlink /etc/systemd/system/multi-target.wants/resolvconf.service
nameserver 8.8.8.8
nameserver 8.8.4.4
Created symlink /etc/systemd/system/system-resolved.service.
system-resolved.service is disabled or a static unit.
resolvconf-pull-resolved.service is disabled or a static unit.
resolvconf-pull-resolved.service is disabled or a static unit.
Обрабатывается триггер для systemd (240:ubuntu18.04)
Обрабатывается триггер для pam-db (2.8.5-2)
Обрабатывается триггер для resolvconf (1.79ubuntu13)
root@den-pc:~$ 
```

Рис. 10.1. Ubuntu: установка пакета `resolvconf` и редактирование его конфигурационного файла

### 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 собственно VPN-соединения. Перейдите по ссылке <https://www.vpnunlimitedapp.com/ru/downloads/linux> и скачайте клиент для Debian/Ubuntu.

Но не спешите устанавливать пакет с помощью установщика пакетов или команды `dpkg`. Скачанный пакет требует установки множества других пакетов, поэтому для его установки нужно использовать команду `apt`:

```
sudo apt install ./vpn-unlimited-<версия>.deb
```

После установки пакета VPN Unlimited (рис. 10.2) запустите VPN-клиент командой:

При первом запуске (рис. 10.3) вам будет предложено зарегистрироваться.

После подтверждения указанного при регистрации адреса электронной почты войдите под своей учетной записью и нажмите кнопку **СТАРТ**. Затем перейдите по адресу [myip.ru](http://myip.ru) и убедитесь, что ваш IP-адрес сменился (рис. 10.4).

Бесплатный тестовый период, как уже отмечалось, составляет 7 дней. Далее пользователем должно быть принято решение о покупке услуги.

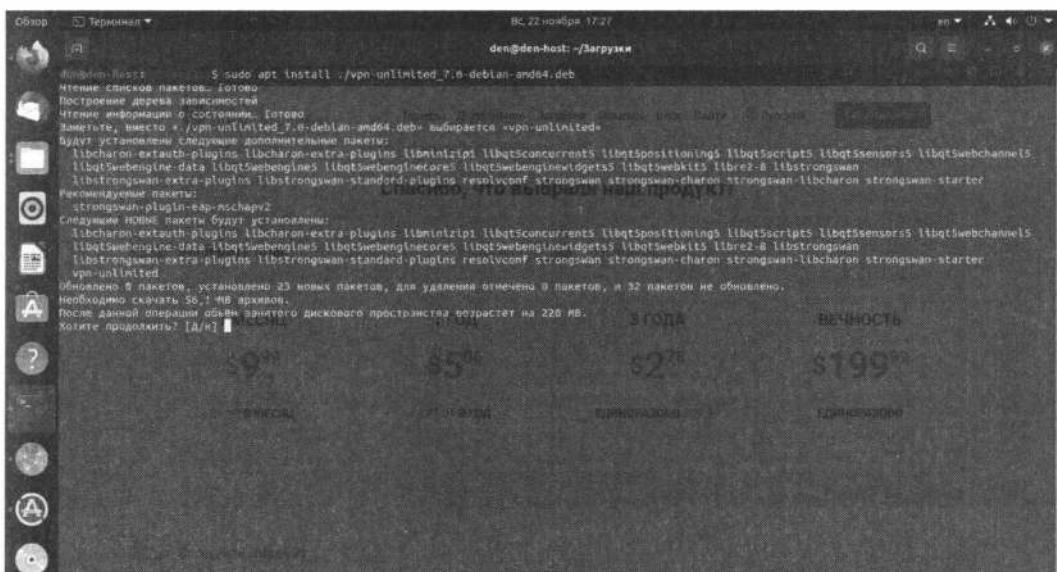


Рис. 10.2. Ubuntu: установка пакета VPN Unlimited

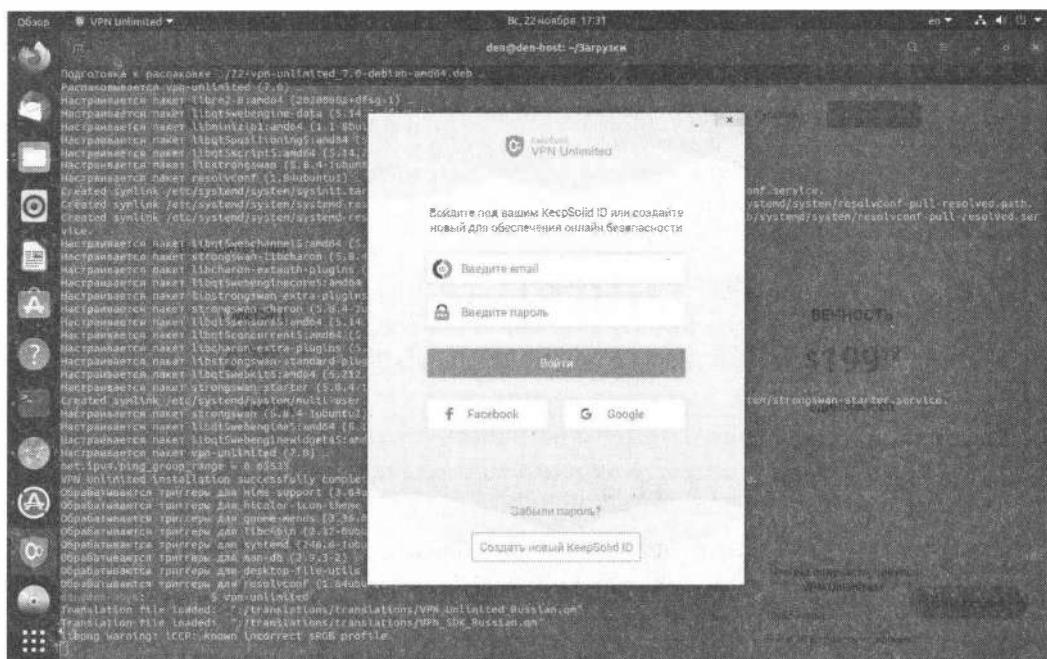


Рис. 10.3. Ubuntu: создайте аккаунт для VPN Unlimited

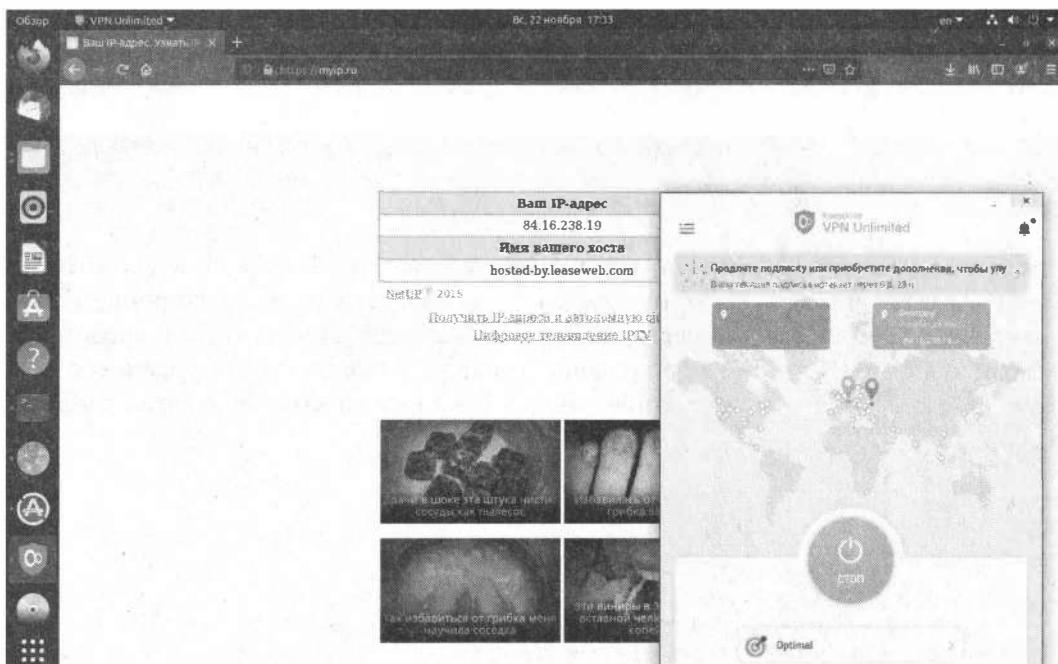
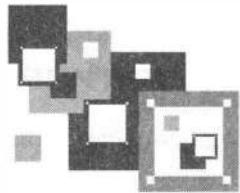


Рис. 10.4. VPN-соединение установлено



## 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.

#### **ПРИМЕЧАНИЕ**

Для дома или небольшого офиса, возможно, оптимальным решением станет покупка маршрутизатора, в который уже встроена возможность балансировки каналов или переключения на резервный канал в случае недоступности основного. В этом случае пусть и возможность конфигурации будет ограничена, но зато вы получите уже готовое решение без необходимости выделения под эти нужды отдельного компьютера.

## 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: ISP1 и ISP2 (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 запишите в него 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 под вашу реальность.

### 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_net dev $i2_eth scope link table $table1
ip route add $il_net dev $il_eth scope link src $il_ip table $table1
ip route add 127.0.0.0/8 dev lo scope link table $table1
ip route add default via $il_gw table $table1

ip rule add prio 51 fwmark 1 table $table1
ip rule add from $il_ip table $table1

ip route add $local_net dev $local_eth scope link table $table2
ip route add $il_net dev $il_eth scope link table $table2
ip route add $i2_net dev $i2_eth scope 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 предыдущем разделе и не предполагает статической маршрутизации, хотя вы можете организовать ее по образу и подобию приведенного там способа, — нужно будет лишь незначительно модифицировать его код.

Прежде всего, как обычно, правим файл `/etc/iproute2/route_tables`. 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 исполняемым: *chmod +x routing*.

**Листинг 11.4. Сценарий /etc/iproute2/routing**

```
#!/bin/bash
# Импортируем файл конфигурации
. /etc/iproute2/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 $IFO table $TBL1 > /dev/null 2>&1
ip route add $ISP2_NET dev $IF2 table $TBL1 > /dev/null 2>&1
ip route add 127.0.0.0/8 dev lo table $TBL1 > /dev/null 2>&1
ip route add $LOCAL_NET dev $IFO table $TBL2 > /dev/null 2>&1
ip route add $ISP1_NET dev $IF1 table $TBL2 > /dev/null 2>&1
ip route add 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

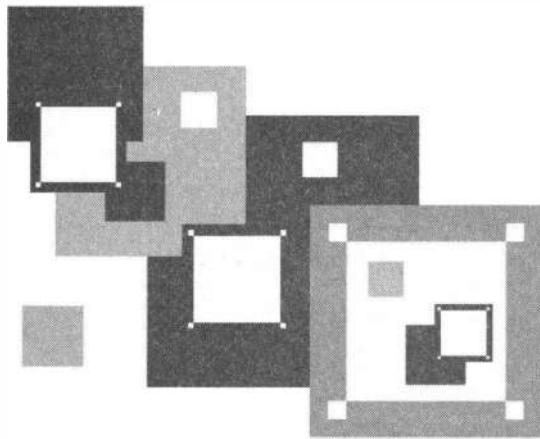
OLDIF1=$NEWIF1
OLDIF2=$NEWIF2
sleep 5
done
```

Теперь разберемся, 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 первый. Если нужно, чтобы больше трафика пошло через первый шлюз, просто измените содержимое переменных `w1` и `w2`, — чем выше «вес», тем главное считается канал.

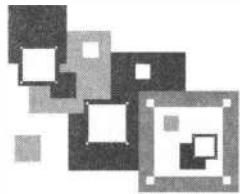


## ЧАСТЬ IV

### **Linux дома и в офисе**

Эта часть книги посвящена домашнему и офисному применению Linux. Первым делом мы узнаем, как добавить в вашу систему поддержку популярных форматов мультимедиа MP3 и DivX, поскольку разработчики современных дистрибутивов по лицензионным соображениям исключили ее из своих продуктов. Потом построим собственный медиацентр. А уже затем поговорим о настройке графической подсистемы, офисных пакетах, программе GIMP, текстовых редакторах кода 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, ей для этого формата нужен кодек (codec, от COder/DECoder) — специальная программа, «знающая» как работать с тем или иным форматом. Кодек 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, чего никто не хочет. 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.

Ради справедливости нужно отметить, что при установке некоторых дистрибутивов можно выбрать опцию установки кодеков. Другими словами, кодеки по-прежнему не входят в состав дистрибутива, но вы, как конечный пользователь, имеете право их загрузить при установке операционной системы.

## 12.2. Настройка дистрибутива Fedora 32-33

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. Введите одну команду, которая установит обе версии репозиториев:

```
sudo dnf install https://mirrors.rpmfusion.org/free/fedora/rpmfusion-free-release-$(rpm -E %fedora).noarch.rpm https://mirrors.rpmfusion.org/nonfree/fedora/rpmfusion-nonfree-release-$(rpm -E %fedora).noarch.rpm
```

Можно просто открыть в браузере ссылку: <https://rpmfusion.org/Configuration> и выбрать пакеты для вашей версии Fedora. Например, для 33-й версии они называются: rpmfusion-free-release-33.noarch.rpm и rpmfusion-nonfree-release-33.noarch.rpm. Пакеты можно скачать и установить командой rpm или dnf.

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 gstreamer-libav gstreamer1-plugins-  
bad-free-extras gstreamer1-plugins-bad-freeworld 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

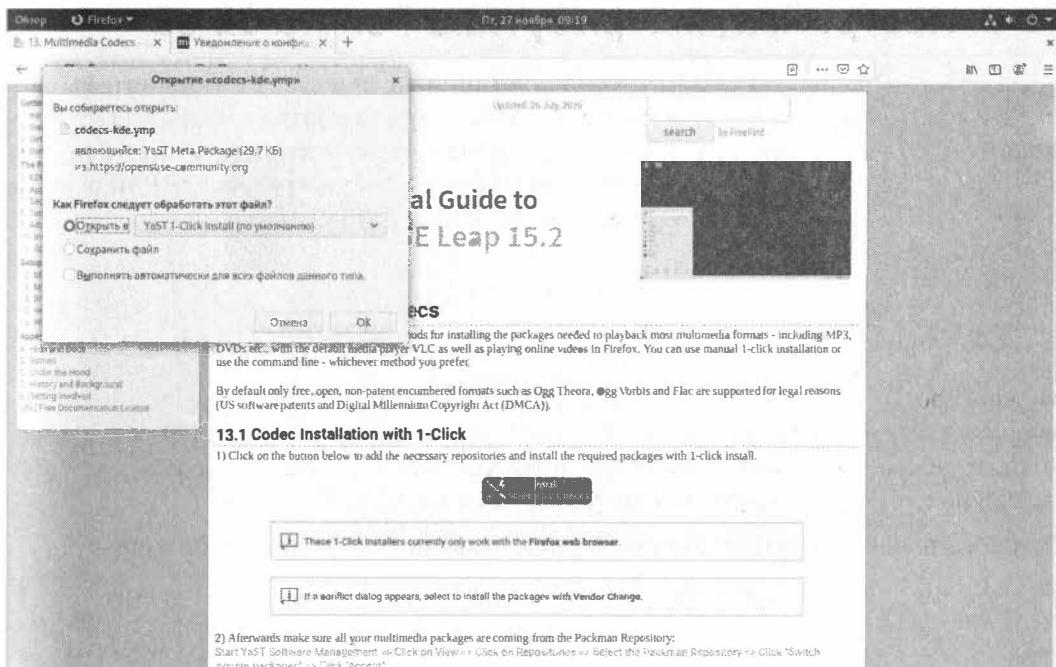
Установить кодеки в openSUSE можно двумя способами: автоматически или вручную, и здесь вам будут наглядно продемонстрированы оба способа.

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).

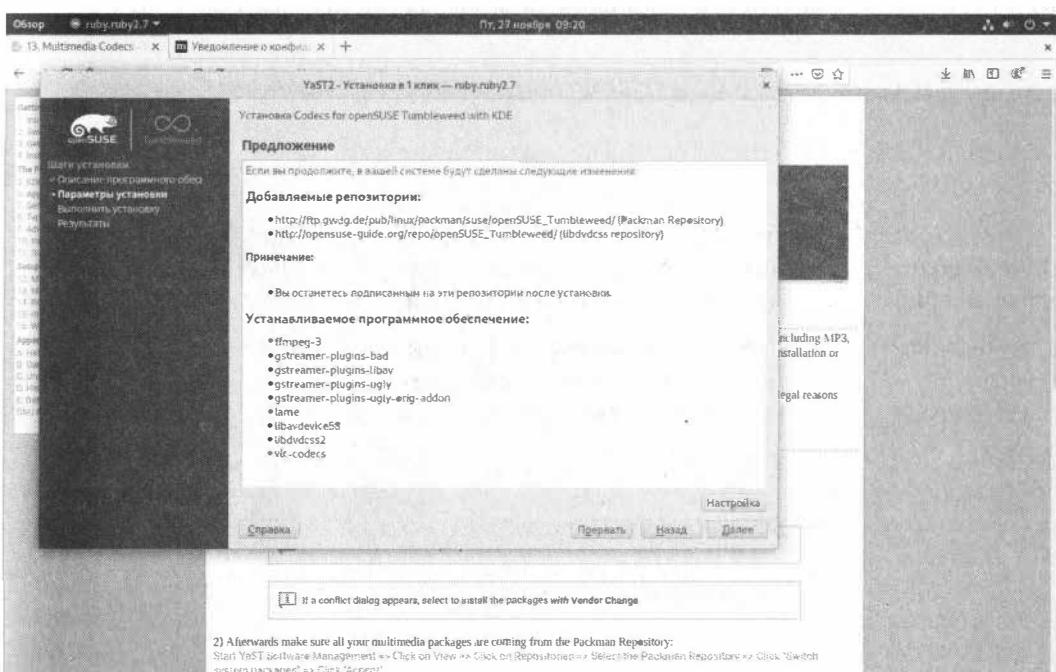
В открывшемся окне **YaST2 - Установка в 1 клик** следуйте инструкциям инсталлятора. Кстати, последний покажет, что он собирается сделать: добавить репозитории Packman и libdvdcss, а также установить девять пакетов (рис. 12.2).

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 **OK — Повторите попытку**. 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).



**Рис. 12.1. openSUSE: открываем загружаемый файл в YaST 1-Click Install**



**Рис. 12.2. openSUSE: добавление репозиториев и установка пакетов**

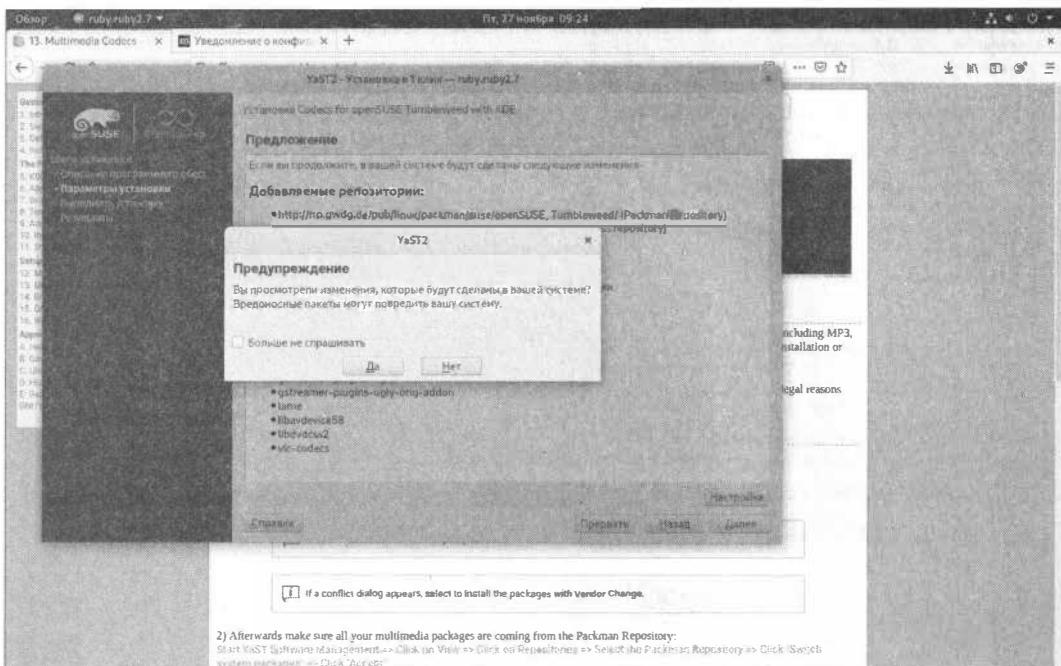


Рис. 12.3. openSUSE: надоедливые вопросы инсталлятора

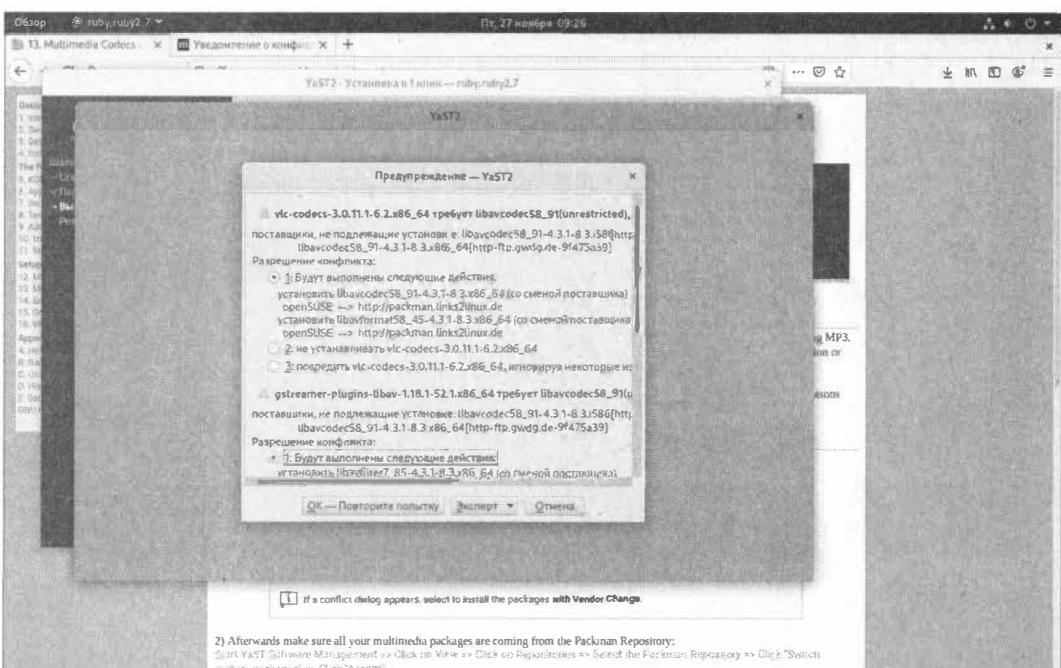


Рис. 12.4. openSUSE: разрешение конфликта пакетов

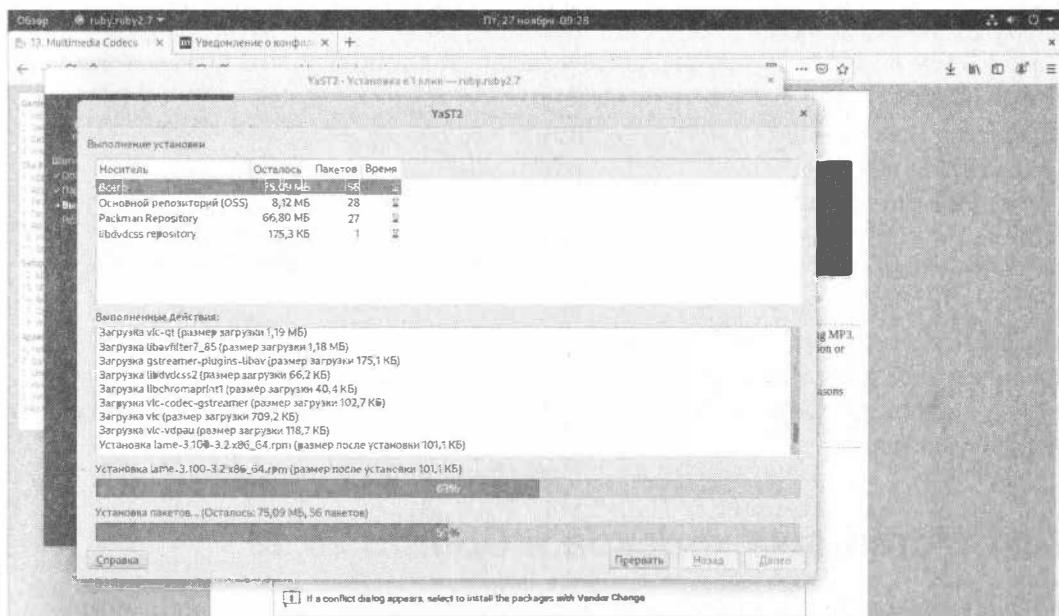
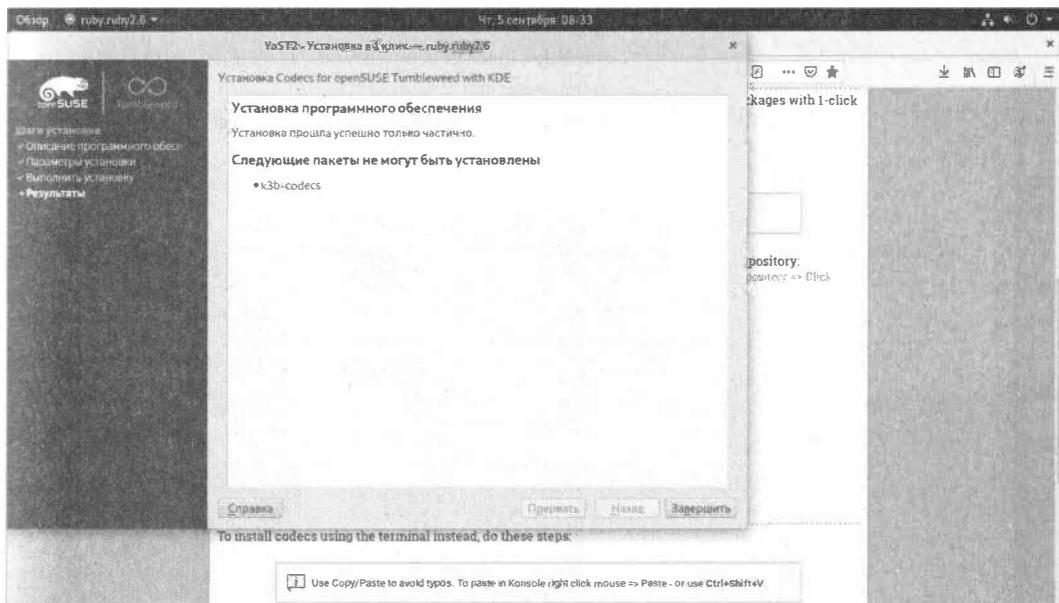


Рис. 12.5. openSUSE: процесс установки

Рис. 12.6. openSUSE: установка завершена  
(приложение k3b не установлено, поэтому и кодеки для него не устанавливались)

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.

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://ftp.gwdg.de/pub/linux/packman/suse/
openSUSE_Tumbleweed/
zypper addrepo -f http://opensuse-guide.org/repo/openSUSE_Tumbleweed/
zypper install ffmpeg-3 lame gstreamer-plugins-bad gstreamer-plugins-ugly
gstreamer-plugins-ugly-orig-addon gstreamer-plugins-libav libavdevice58
libdvdcss2 vlc-codecs
```

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 20.10

В Ubuntu также есть два способа установки: автоматический и ручной. Автоматический сводится к установке пакетов из области Дополнения | Кодеки (рис. 12.7).

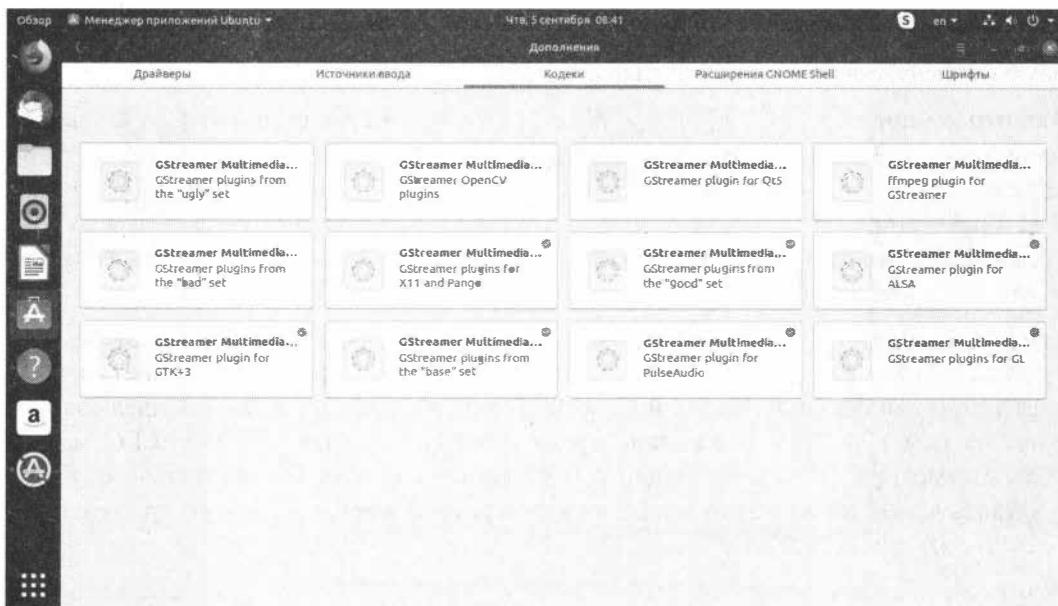


Рис. 12.7. Ubuntu: автоматическая установка кодеков

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. Мне 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 компьютер к телевизору, 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 «весят» 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 Гбайт. 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 по адресу: <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 `create_installstick.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 затягивается, можно завершить его, переформатировать флешку и запустить файл `create_installstick.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 приготовленную флешку. Будьте осторожны и не установите ненароком дистрибутив на жесткий диск! Это не openSUSE или Ubuntu, которые используют для создания Linux-раздела свободное пространство 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.

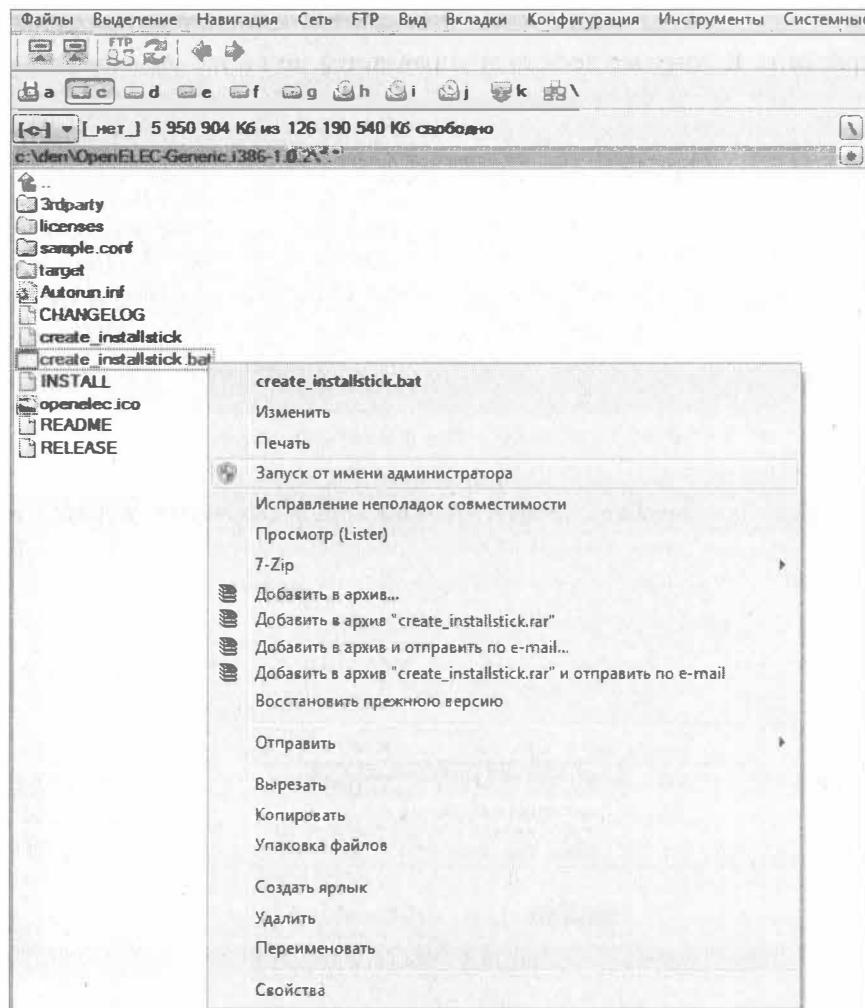


Рис. 12.8. Windows: запуск файла create\_installstick.bat

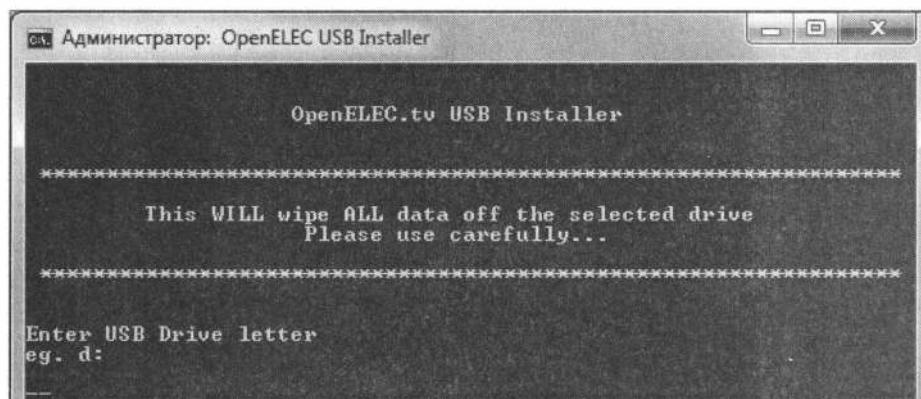


Рис. 12.9. Windows: введите букву накопителя

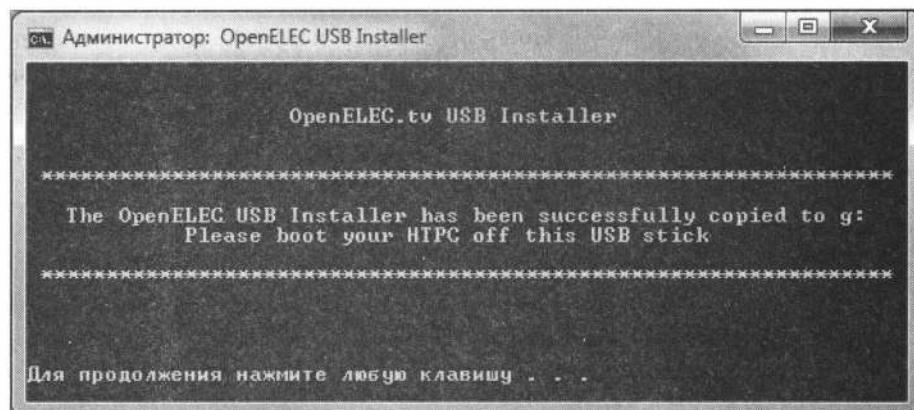


Рис. 12.10. Windows: запись инсталлятора завершена

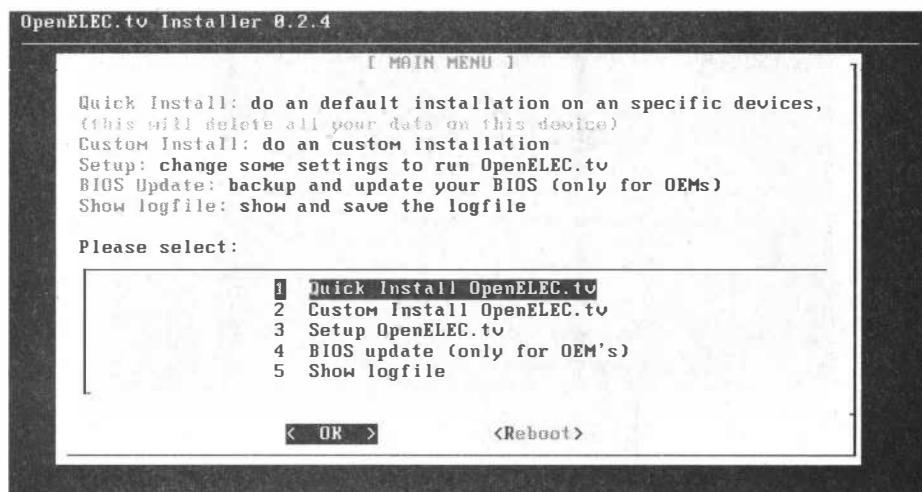


Рис. 12.11. openELEC: меню инсталлятора

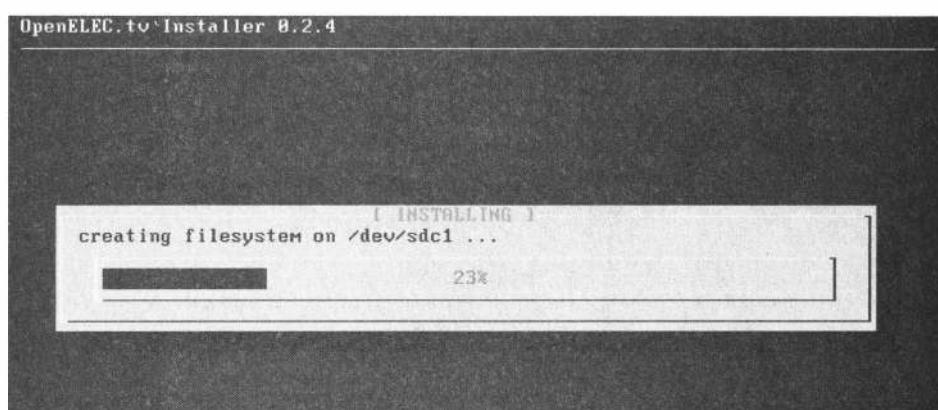


Рис. 12.12. openELEC: установка дистрибутива

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.

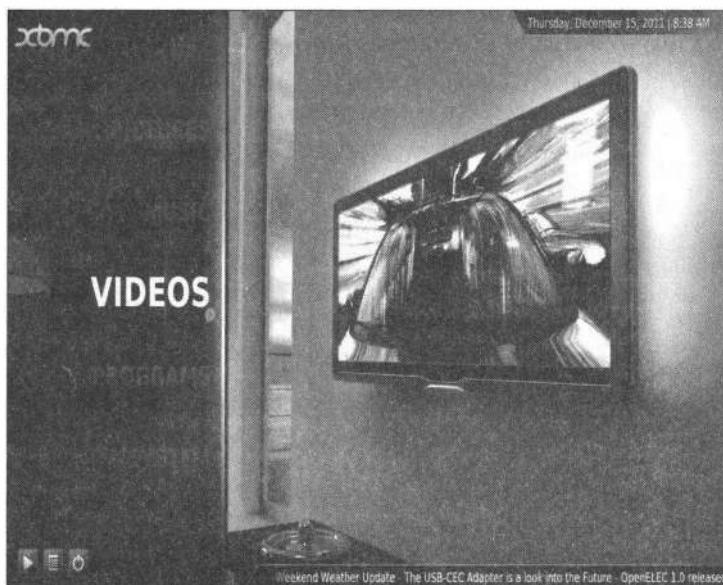


Рис. 12.13. openELEC: интерфейс XBMC

### 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 сети, **насколько я понял**, только Ethernet. Никакой поддержки ни Wi-Fi, ни PPPoE. Пришлось подключать медиацентр 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.

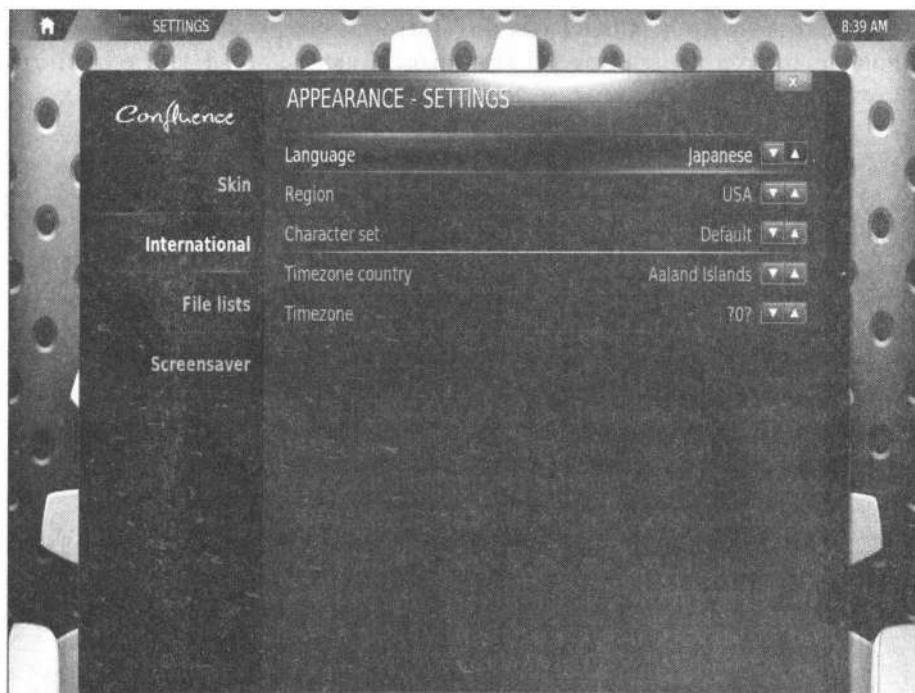


Рис. 12.14. openELEC: изменение языка



Рис. 12.15. openELEC: выбран русский язык

Настройки сети (**Система | Сеть | Доступ в интернет**) весьма скучны — 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.

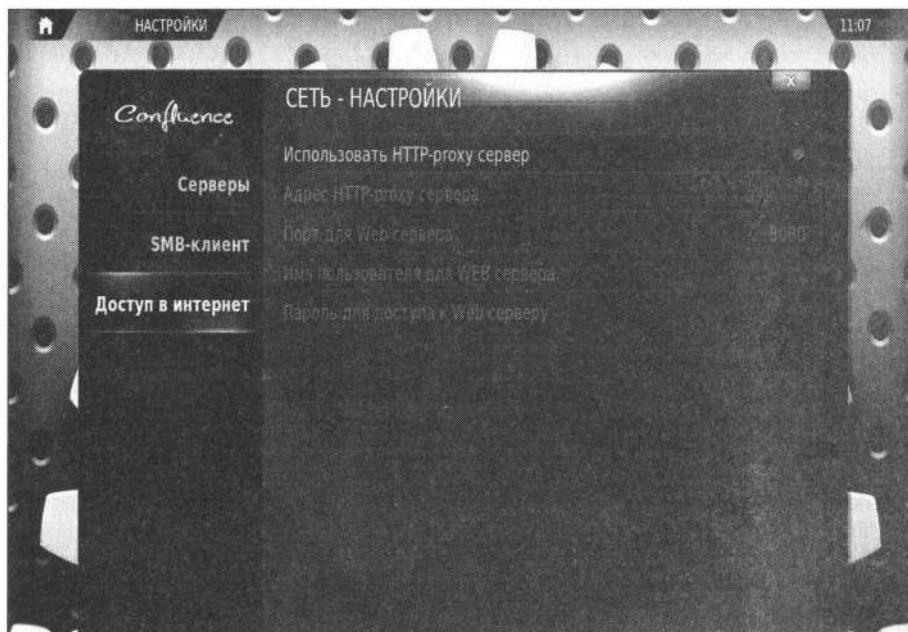


Рис. 12.16. openELEC: параметры сети

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 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

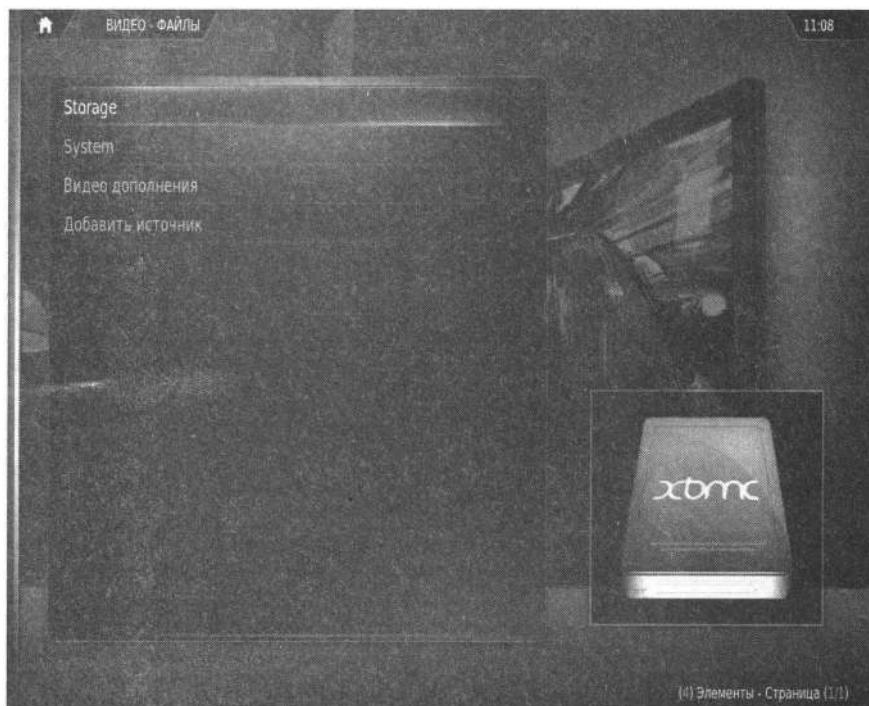


Рис. 12.17. openELEC: выбор видеофайлов

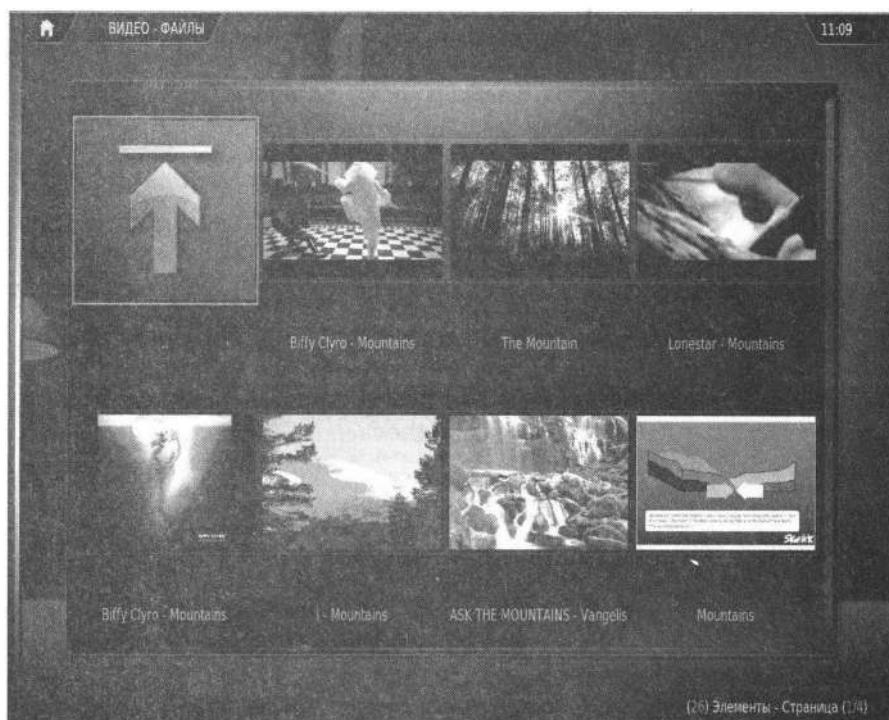


Рис. 12.18. openELEC: результаты поиска на YouTube



Рис. 12.19. openELEC: просмотр видео

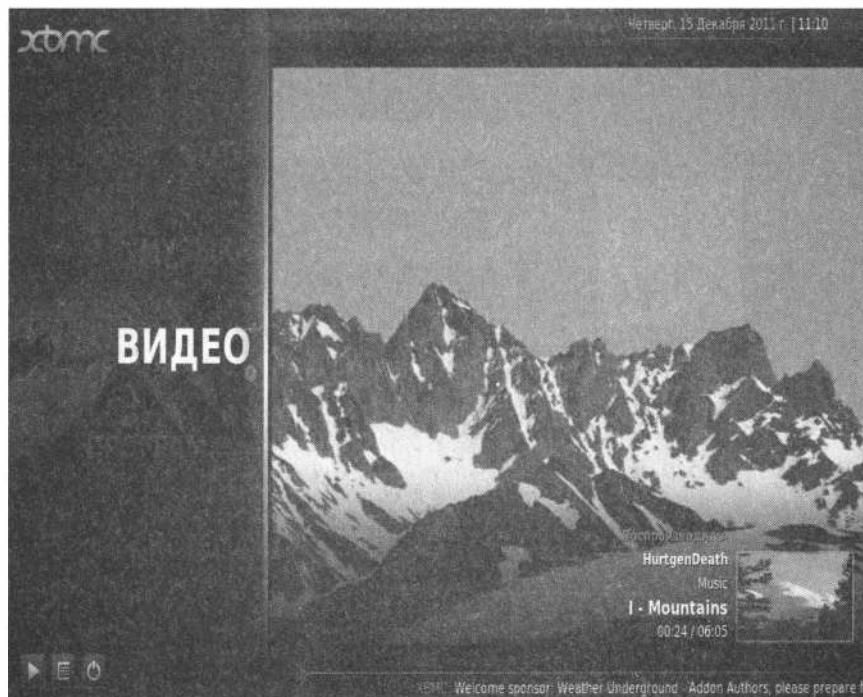


Рис. 12.20. openELEC: воспроизведение в фоновом режиме



Рис. 12.21. openELEC: раздел Программы

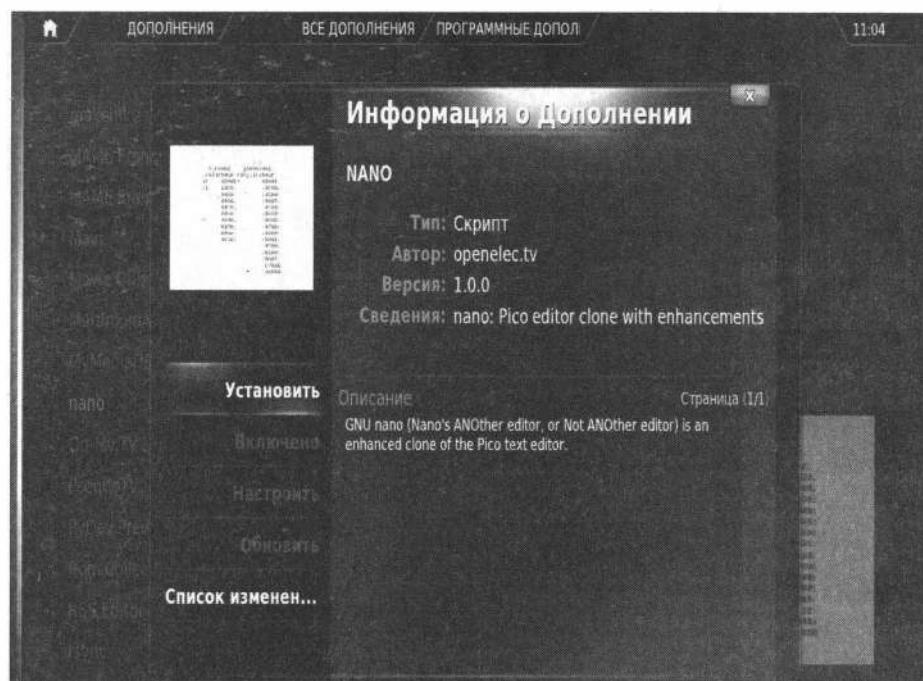


Рис. 12.22. openELEC: установка программы

мало. Стандартных программ вроде офисных приложений вы можете здесь и не искать. Зато есть Торент-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 *nano*), 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 плеялиста (см. рис. 12.20), — вы увидите окошко, позволяющее выключить, перезагрузить или отправить в сон ваш компьютер (рис. 12.23).

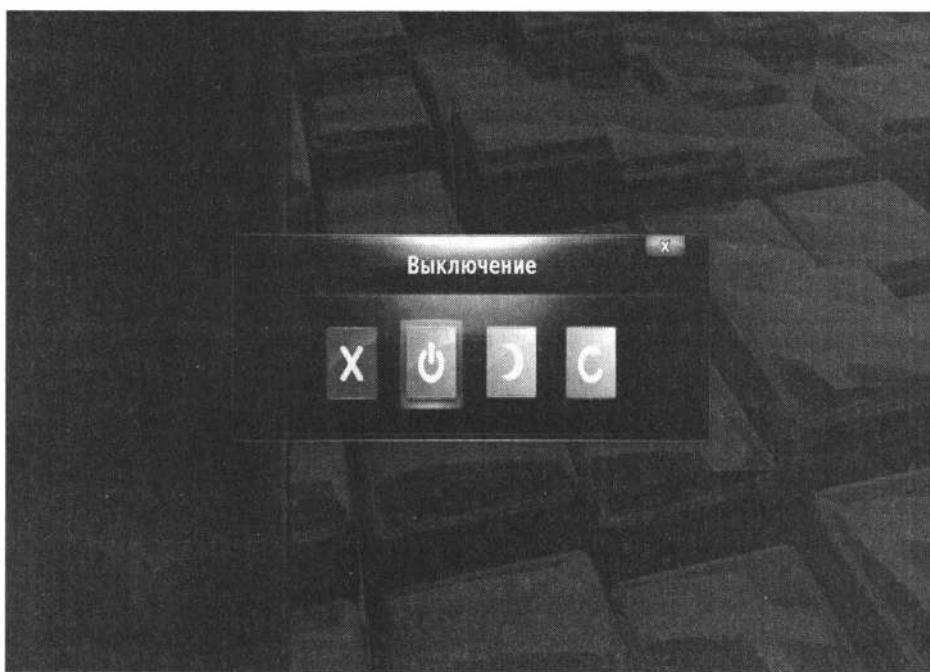


Рис. 12.23. openELEC: завершение работы

#### 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 веб-интерфейс, позволяющий управлять медиацентром.

#### 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:

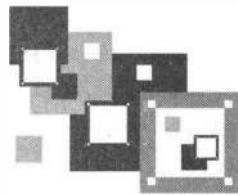
- IP: IP-адрес вашего медиацентра;
- имя пользователя: root;
- пароль: openelec.

### 12.5.6. Ложки дегтя...

Как известно, 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 30, 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 оставляют 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 медиацентра 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 вовсю 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 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 **Устройства | Настройка экранов**, 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.

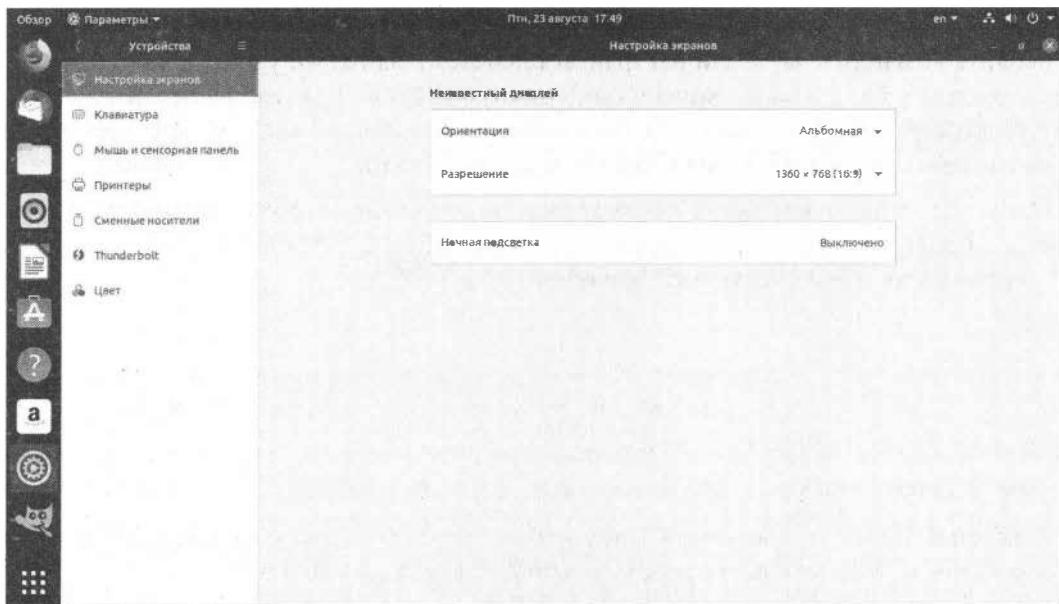


Рис. 13.1. GNOME 3.32: окно Настройка экранов

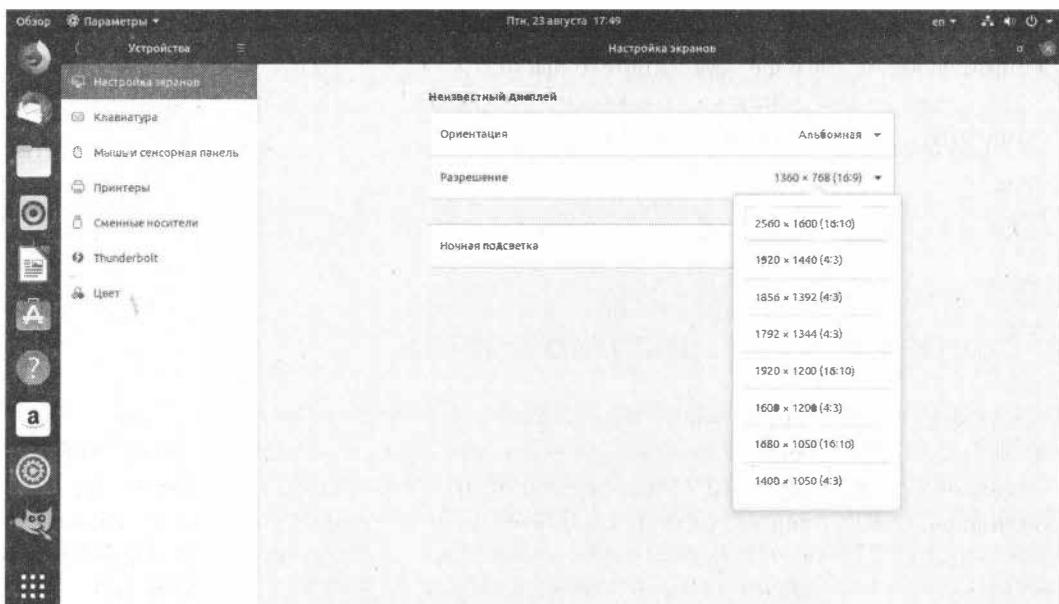


Рис. 13.2. GNOME 3.32: выбор разрешения

## 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 версиях Ubuntu с оболочкой Unity используется менеджер дисплеев `lightdm`, поэтому там нужно завершать работу командой 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` (или `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-video`, `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 повторюсь — 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 настоящий 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 PnP 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:unscaled"
    FontPath      "/usr/share/fonts/local"
    FontPath      "/usr/share/fonts/75dpi:unscaled"
    FontPath      "/usr/share/fonts/100dpi:unscaled"
    FontPath      "/usr/share/fonts/Typel"
```

```
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:unscaled"
FontPath      "/usr/share/fonts/latin2/75dpi:unscaled"
FontPath      "/usr/share/fonts/latin2/100dpi:unscaled"
FontPath      "/usr/share/fonts/latin2/Type1"
FontPath      "/usr/share/fonts/latin7/75dpi:unscaled"
FontPath      "/usr/share/fonts/baekmuk:unscaled"
FontPath      "/usr/share/fonts/japanese:unscaled"
FontPath      "/usr/share/fonts/kwintv"
FontPath      "/usr/share/fonts/truetype"
FontPath      "/usr/share/fonts/uni:unscaled"
FontPath      "/usr/share/fonts/CID"
FontPath      "/usr/share/fonts/ucs/misc:unscaled"
FontPath      "/usr/share/fonts/ucs/75dpi:unscaled"
FontPath      "/usr/share/fonts/ucs/100dpi:unscaled"
FontPath      "/usr/share/fonts/hellas/misc:unscaled"
FontPath      "/usr/share/fonts/hellas/75dpi:unscaled"
FontPath      "/usr/share/fonts/hellas/100dpi:unscaled"
FontPath      "/usr/share/fonts/hellas/Type1"
FontPath      "/usr/share/fonts/misc/sgi:unscaled"
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        "type1"
    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 X-сервер продолжит работу даже в случае неработоспособности мыши (когда мышь поломана или не подключена)
<code>AllowNonLocalModInDev</code>	Разрешает удаленным пользователям изменять параметры клавиатуры и мыши X-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 Compiz Fusion. Поэтому, если вы планируете использовать трехмерный рабочий стол, вам нужно включить этот флаг. В openSUSE этого делать не стоит, поскольку вместо AIGLX в SUSE используется собственная разработка — Xgl
<code>DontZap</code>	Запрещает комбинацию клавиш <code>&lt;Ctrl&gt;+&lt;Alt&gt;+&lt;Backspace&gt;</code> , служащую для аварийной остановки X-сервера

Таблица 13.1 (окончание)

Флаг	Описание
StandbyTime	Время простоя (в минутах), после которого X-сервер выключит монитор. Монитор должен поддерживать DPMS (Display Power Management Signaling) — систему управления энергопотреблением дисплеев
NoPm	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 рассматриваемом 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 (Display Power Management Signaling, систему управления энергосбережением монитора);
  - `HorizSync`, `VertRefresh` — допустимая частота горизонтальной и вертикальной разверток соответственно (в кГц). Обычно значение лучше устанавливать конфигуратором, поскольку без подробного руководства по вашему монитору здесь не обойтись;
  - `Identifier` — уникальное имя монитора. В файле конфигурации вы можете описать несколько мониторов, а потом в секции `Screen` указать идентификатор монитора, используемого в настоящий момент;
  - `UseModes` — задает массив режимов монитора, описываемый секцией `Modes`;

- `ModelName, 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 тот или иной режим (вообще-то эта строка может содержать все, что угодно);
  - 1 — частота подачи пикселов 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 конфигуратора, поскольку в его базе данных есть описание режимов практически для всех мониторов.
- Секция `Screen` предназначена для описания экрана, главным образом — для связи видеокарты и монитора. В этой секции представлены используемые в настоящий 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 (клавиатуру 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 конфигу-

ратора) 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 21–29

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.

### **ВНИМАНИЕ!**

На момент подготовки этого издания на официальном сайте NVIDIA не было драйвера для Fedora 30. Возможно он появится, когда книга уже выйдет из печати.

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 закачать по ссылке: [ftp://download.nvidia.com/XFree86/Linux-x86\\_64/352.63/README/supportedchips.html](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 воспользуйтесь средством подбора на официальном сайте (рис. 13.3).

Обычно закачиваемый 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
```

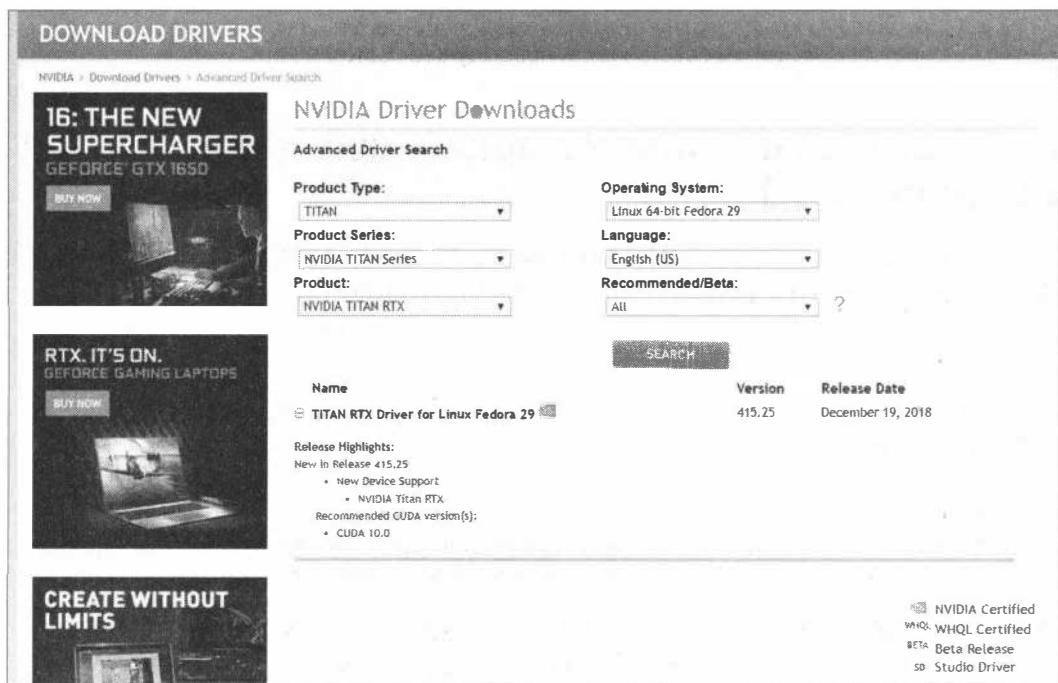


Рис. 13.3. Средство подбора драйверов NVIDIA

Теперь обновим менеджер пакетов — напомню, 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 файл 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 (систему принудительного контроля доступа) — 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).

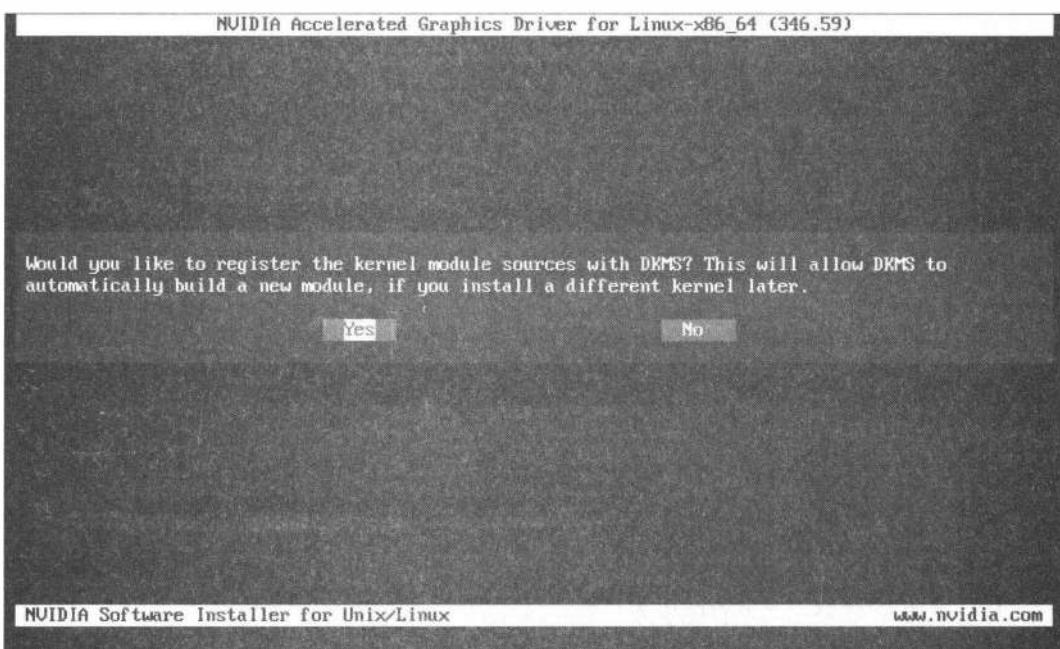
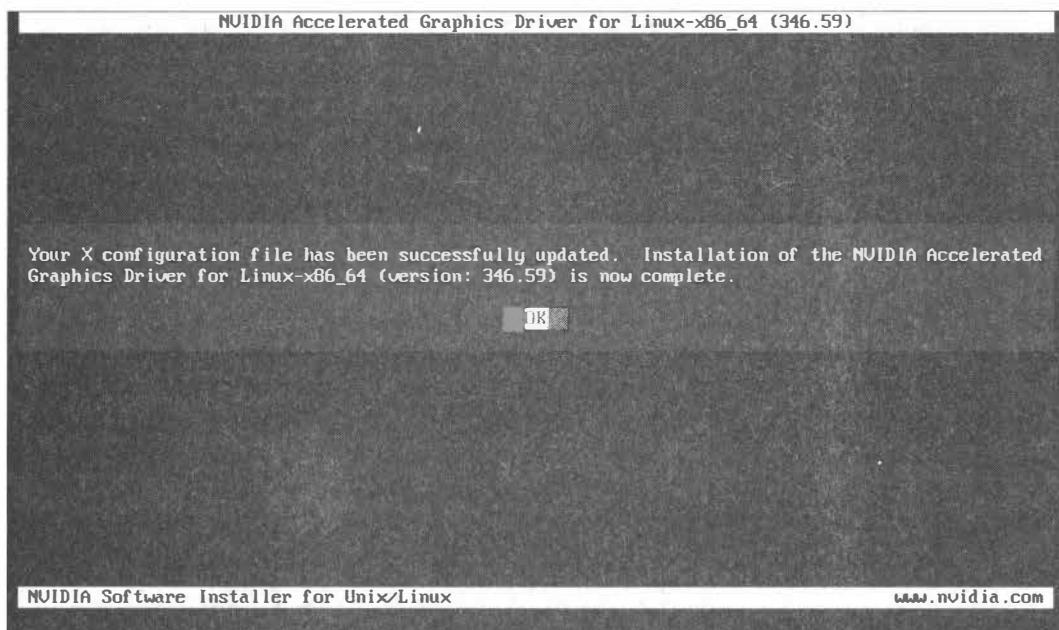
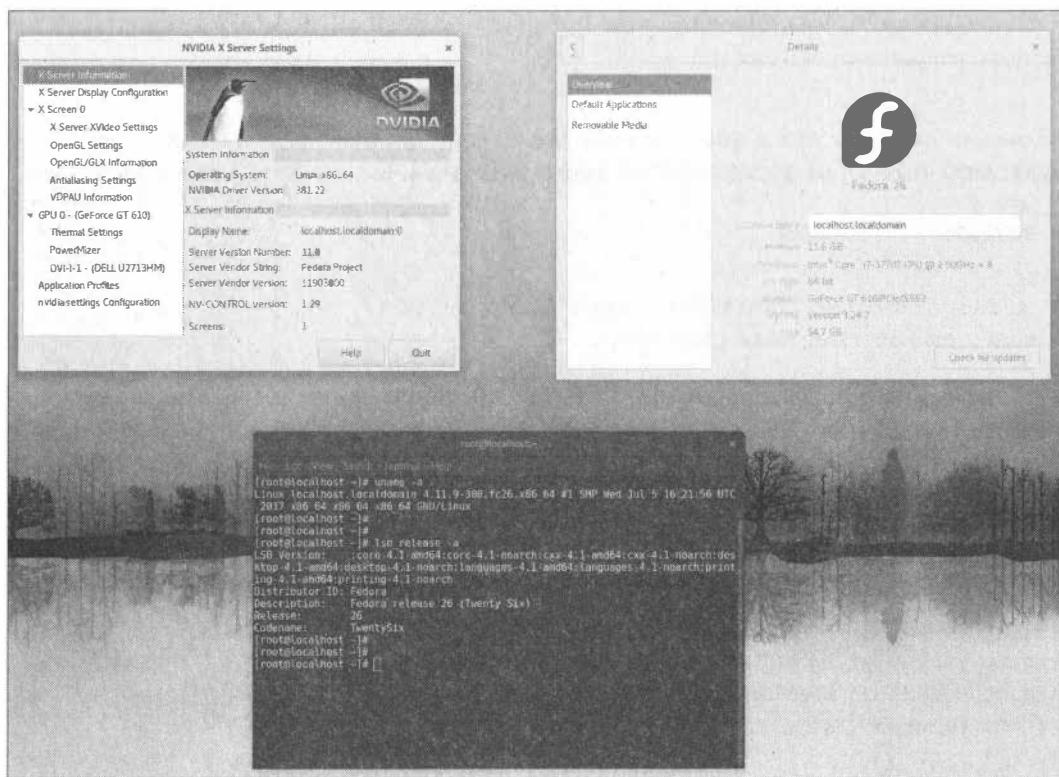


Рис. 13.4. Нажмите Yes



**Рис. 13.5. Драйвер установлен**



**Рис. 13.6.** Fedora 26: драйверы NVIDIA установлены

Восстанавливаем запуск графического интерфейса:

```
sudo systemctl set-default graphical.target
reboot
```

После перезагрузки надо включить видеоускорение (для этого у вас должна быть 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.

## 13.5. Команда *xrandr*

Причиной включения в главу этого раздела стало непонятное поведение менеджера входа в систему lightdm — при перезагрузке системы разрешение экрана сбрасывалось до 800×600, соответственно, после входа в систему разрешение таким и оставалось.

Тратить время, чтобы докопаться до истины, не хотелось, поэтому я попытался изменить разрешение командой *xrandr*:

```
> xrandr
Screen 0: minimum 320 x 200, current 800 x 600, maximum 8192 x 8192
XWAYLAND0 connected 800x600+0+0 (normal left inverted right x axis y axis) 0mm
x 0mm
    800x600      59.86*+
```

```
> xrandr --output XWAYLAND0 -mode 1360x768x32
xrandr: cannot find mode 1360x768x32
```

Разберемся, что здесь происходит. У нас есть один монитор (Screen 0), который называется XWAYLAND0. Он поддерживает разрешения от 320 × 200 до умопомрачительных 8192 × 8192. Но в настоящий момент используется разрешение 800×600 и оно же является единственным настроенным. Другие разрешения (например, 1360×768) установить нельзя, поскольку они не настроены.

Чтобы установить новый режим работы монитора, его сначала нужно добавить:

```
xrandr --newmode "1360x768_60.00" 84.75 1360 1432 1568 1776 768 771 781 798
-hsync +vsync
xrandr --addmode XWAYLAND0 1360x768_60.00
xrandr --output XWAYLAND0 --mode 1360x768_60.00
```

Первая строка создает режим, описывая его. Вторая — добавляет к монитору XWAYLAND0. Третья — устанавливает его для монитора.

Как правильно составить первую строку? Для этого введите команду cvt:

```
cvt 1360 768 60
```

Как вы уже догадались, ей нужно передать разрешение экрана и частоту обновления. Процесс настройки монитора приведен на рис. 13.7, а на рис. 13.8 показано, что монитор успешно работает при заданном разрешении (добавлен режим и выбран по умолчанию).

```
Обзор Терминал Пт, 23 августа 18:26 den@den:~  
[den@den ~]$ cvt 1360 768 60  
# 1360x768 59.80 Hz (CVT) hsync: 47.72 kHz; pclk: 84.75 MHz  
Modeline "1360x768_60.00" 84.75 1360 1432 1568 1776 768 771 781 798 -hsync +vsync  
[den@den ~]$ xrandr --newmode "1360x768_60.00" 84.75 1360 1432 1568 1776 768 771 781 798 -hsync +vsync  
[den@den ~]$ xrandr --addmode XWAYLAND0 "1360x768_60.00"  
[den@den ~]$ xrandr --output XWAYLAND0 --mode "1360x768_60.00"
```

Рис. 13.7. Процесс настройки монитора

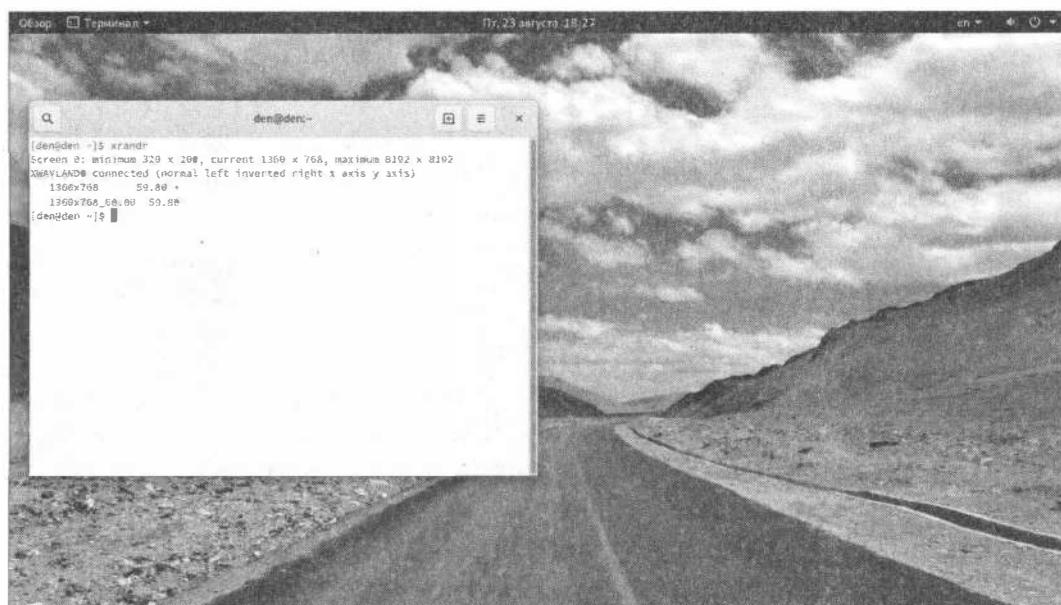


Рис. 13.8. Монитор работает

Когда вы добьетесь корректной работы монитора, нужно сохранить изменения. Для этого создайте сценарий:

```
sudo mcedit /usr/bin/xrandr.sh
```

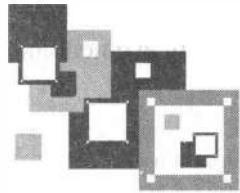
В него добавьте те самые три команды xrandr. После этого сохраните файл и сделайте его исполняемым:

```
sudo chmod +x /usr/bin/xrandr.sh
```

Осталось открыть файл конфигурации lightdm (`/usr/share/lightdm/lightdm.conf.d/50-ubuntu.conf`) и добавить строку, задающую сценарий инициализации дисплея:

```
display-setup-script=/usr/bin/xrandr.sh
```

Теперь этот сценарий будет запускаться до входа в систему, устанавливая должное разрешение монитора.



## 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 (2007–2019).

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 несложный 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 значительную русификацию пакета LibreOffice, с ним без проблем смогут разобраться даже самые начинающие пользователи.

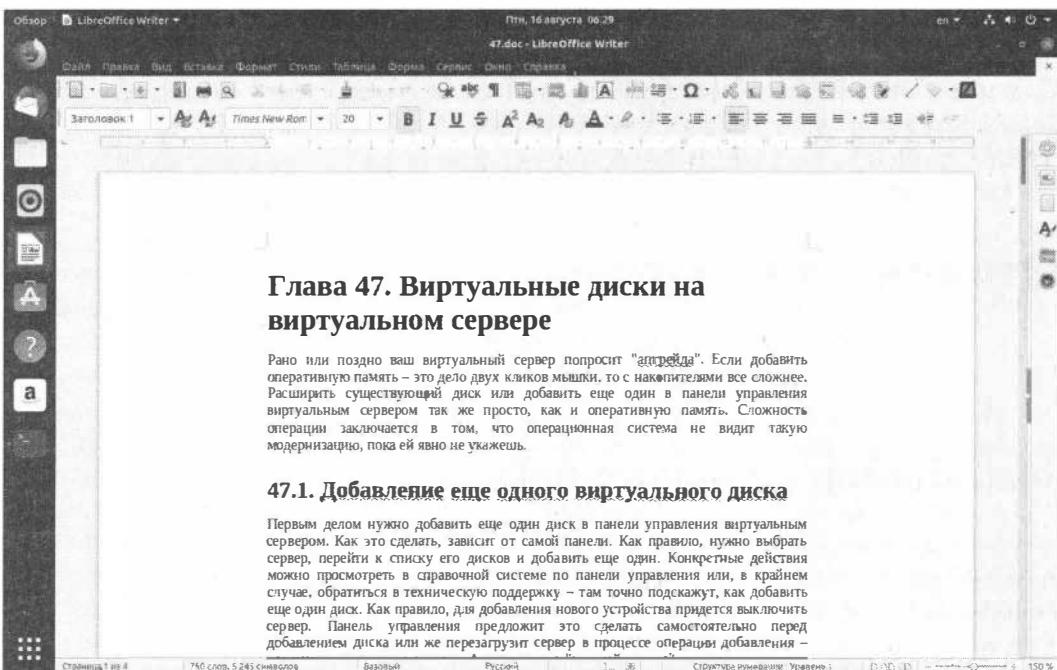


Рис. 14.1. Программа LibreOffice Writer — аналог MS Word

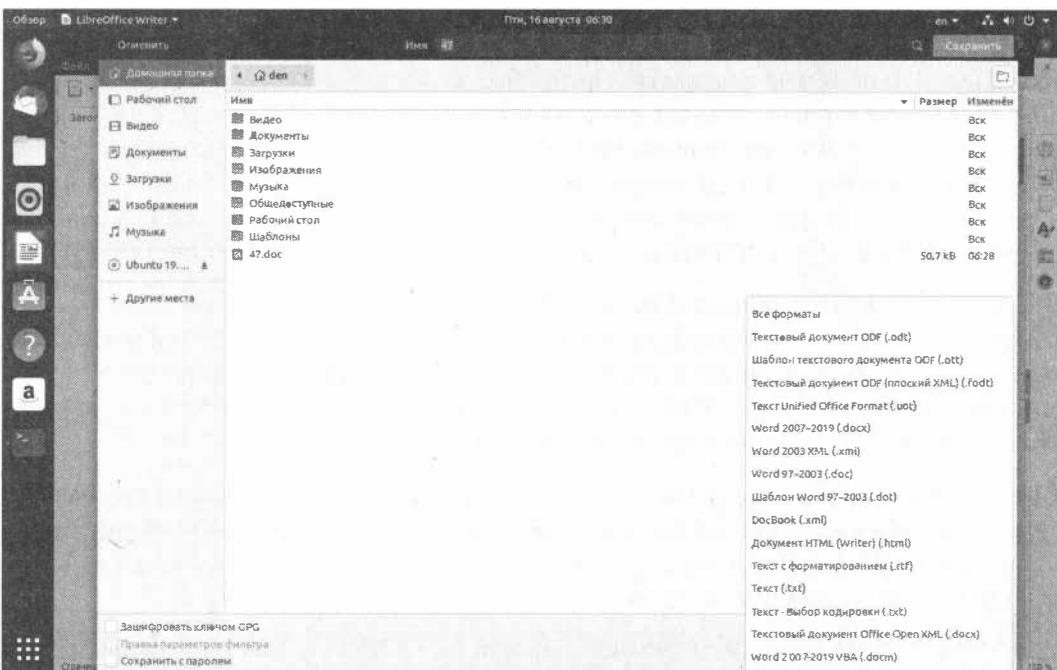


Рис. 14.2. Программа LibreOffice Writer: поддерживаемые форматы

## 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, весьма 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.

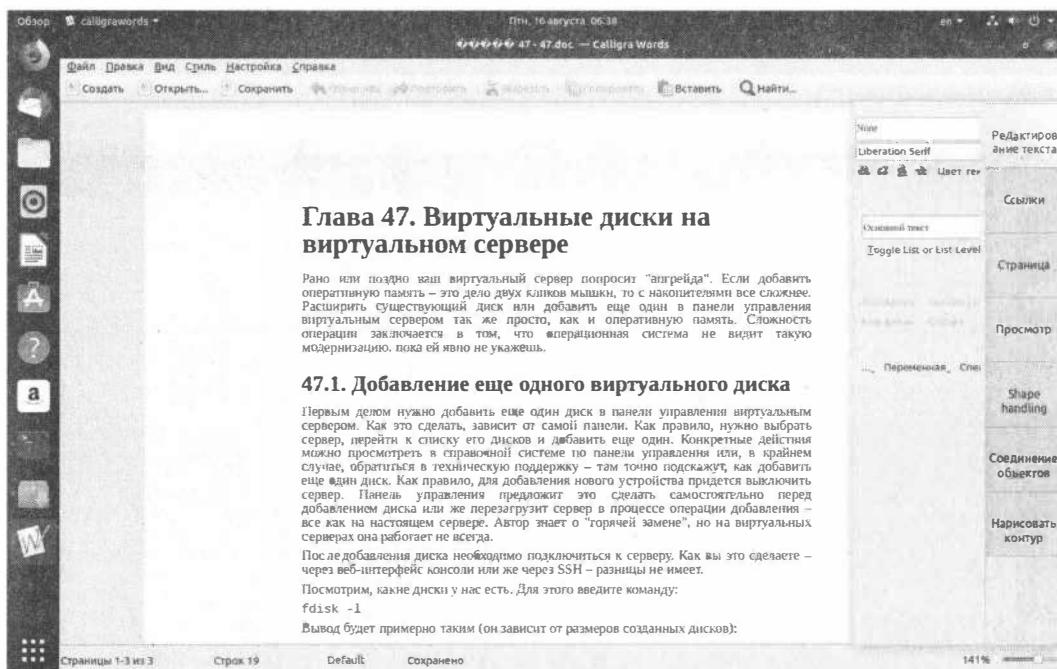


Рис. 14.3. Программа Calligra Words — аналог MS Word

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 достаточно 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. WPS Office (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 WPS Office можно по адресу: <http://wps-community.org/downloads>.

В состав пакета Kingsoft Office входят текстовый процессор, электронная таблица и программа для создания презентаций, поэтому его называют также WPS Office. Здесь WPS — аббревиатура входящих в состав офисного пакета компонентов: Writer, Presentation и Spreadsheet.

Внешне WPS 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.

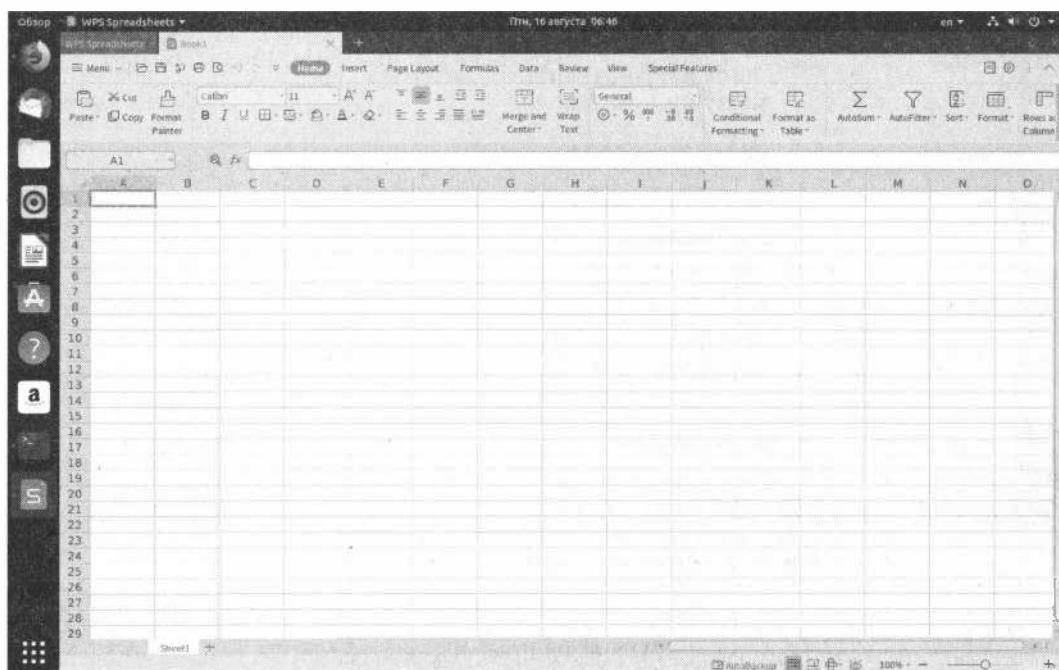


Рис. 14.4. Программа Kingsoft Office Spreadsheet — аналог MS Excel

Программы пакета Kingsoft Office весьма удобны в работе, а их интерфейс наиболее дружественный 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 macOS 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.

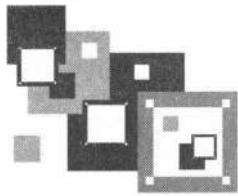
Непосредственно OpenOffice.org появился как Open-Source версия (версия с открытым исходным кодом) пакета StarOffice в 2002 году. Пакет LibreOffice отделился от OpenOffice.org в 2010 году. По сути, это тот же OpenOffice, но с небольшими изменениями.

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 Ubuntu 19), 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 macOS.

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 несложно 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 офисных пакетов из 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](http://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 в GIMP мы здесь и рассмотрим. Кстати, в фотостудиях их выполнение не столь уж и дешево. Например, печать стандартной фотографии открытого 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 с ней рекомендую свою книгу «GIMP 2 — бесплатный аналог Photoshop для Windows/Linux/Mac OS, 2-е изд.»<sup>1</sup>.

### 15.1. Начало работы

Ранее при первом запуске GIMP запрашивал, сколько оперативной памяти можно выделить под нужды приложения. В версии GIMP 2.10.x, которая входит в состав всех современных дистрибутивов Linux, размер кэша установлен по умолчанию в 1581 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 | Системные ресурсы**.

#### ТЕМЫ ОФОРМЛЕНИЯ GIMP

Посмотрите на рис. 15.1. На нем GIMP представлен с темой оформления по умолчанию **Dark** (Темная). Для улучшения качества иллюстраций мы воспользуемся светлой

<sup>1</sup> См. <http://bhv.ru/books/book.php?id=186881>.

темой оформления, поэтому далее на всех иллюстрациях будет показан GIMP2 с темой оформления Light. Сменить тему оформления можно в меню Правка | Параметры | Интерфейс | Тема.

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).

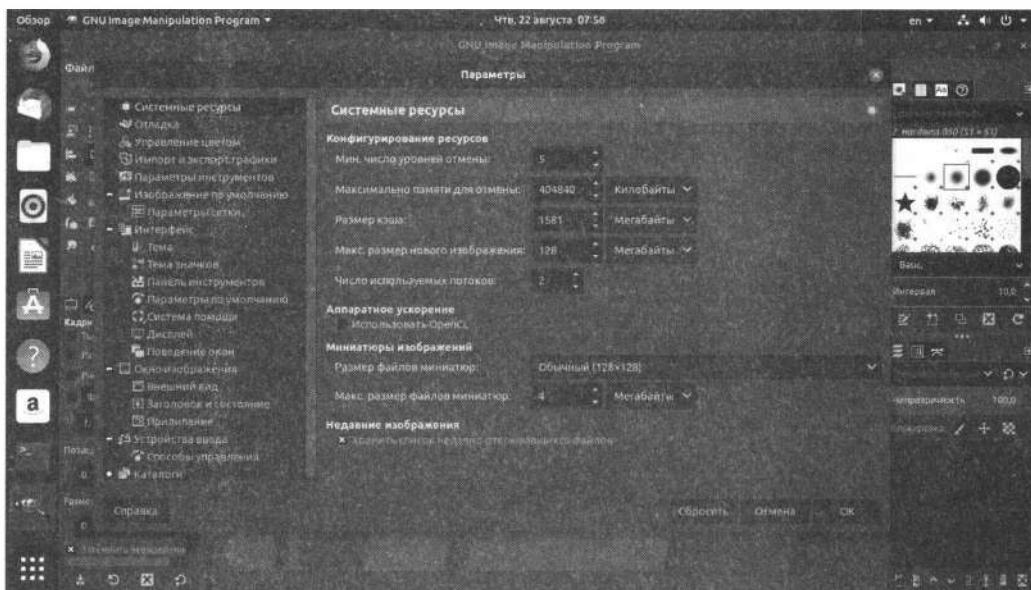


Рис. 15.1. Ubuntu: настройка кэша GIMP

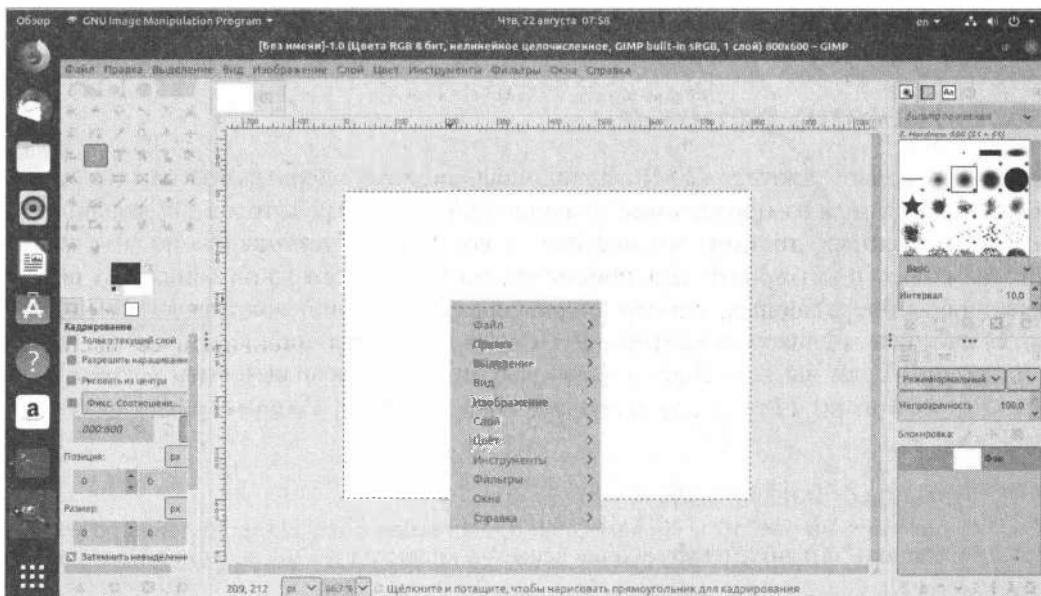


Рис. 15.2. Ubuntu: GIMP в работе

## 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).

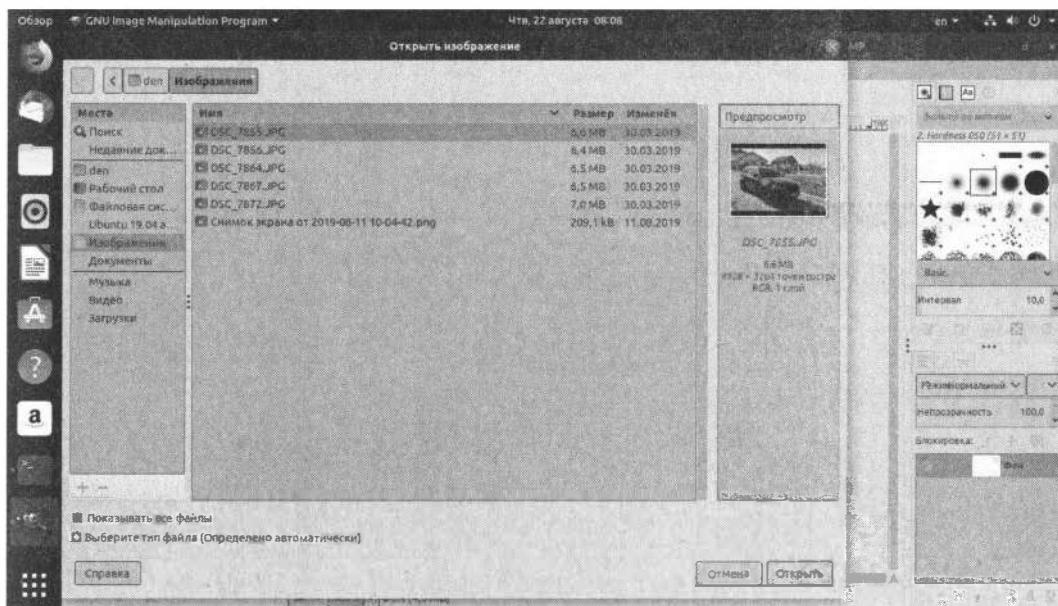


Рис. 15.3. Ubuntu: открытие фотографии в GIMP

### 15.2.1. Изменение размера (масштабирование)

Давайте попробуем для начала изменить размер изображения. Это весьма важная операция. Предположим, вы снимаете цифровым фотоаппаратом с матрицей в 6 мегапикселов, 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 пикселях (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.

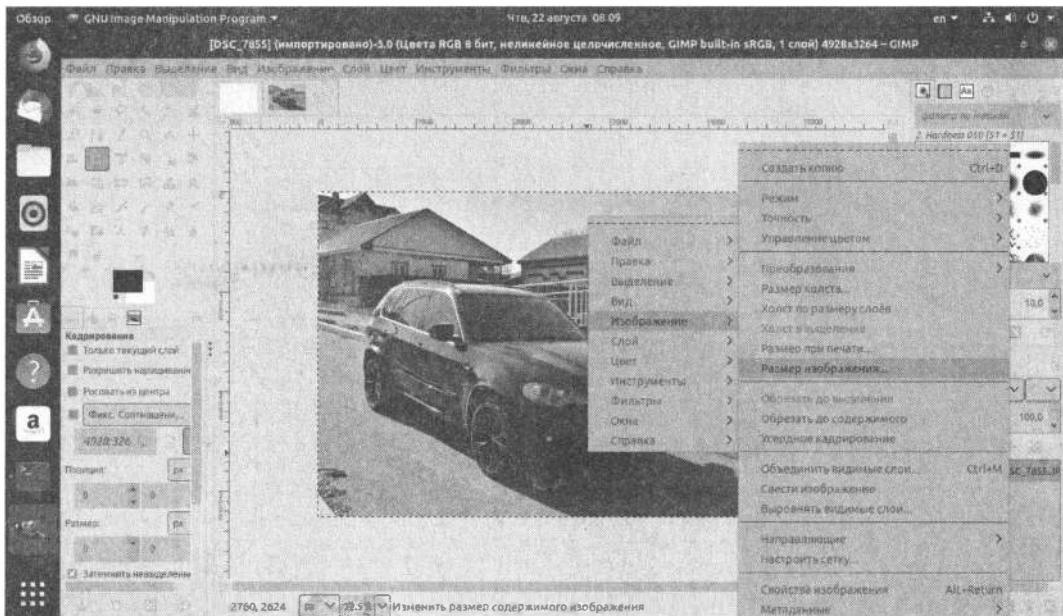


Рис. 15.4. Ubuntu: выбор в GIMP команды изменения размера изображения

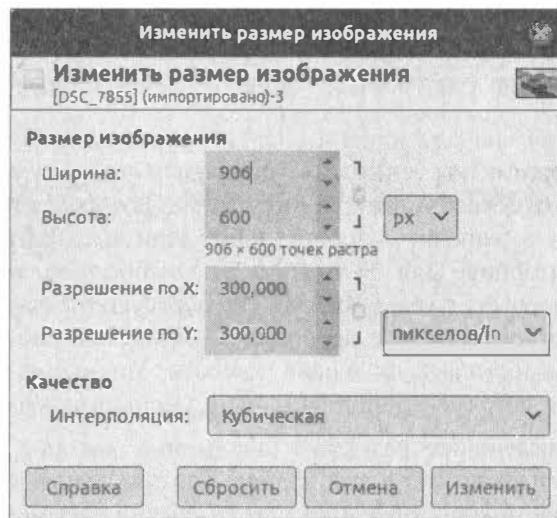


Рис. 15.5. Ubuntu: окно масштабирования GIMP

## 15.2.2. Вращение

Теперь попробуем вращать изображение. Для этого предусмотрено меню **Инструменты | Преобразование | Зеркало**, позволяющее отражать картинку по вертикали и горизонтали, а также вращать ее на 90 и 180 градусов (рис. 15.6).

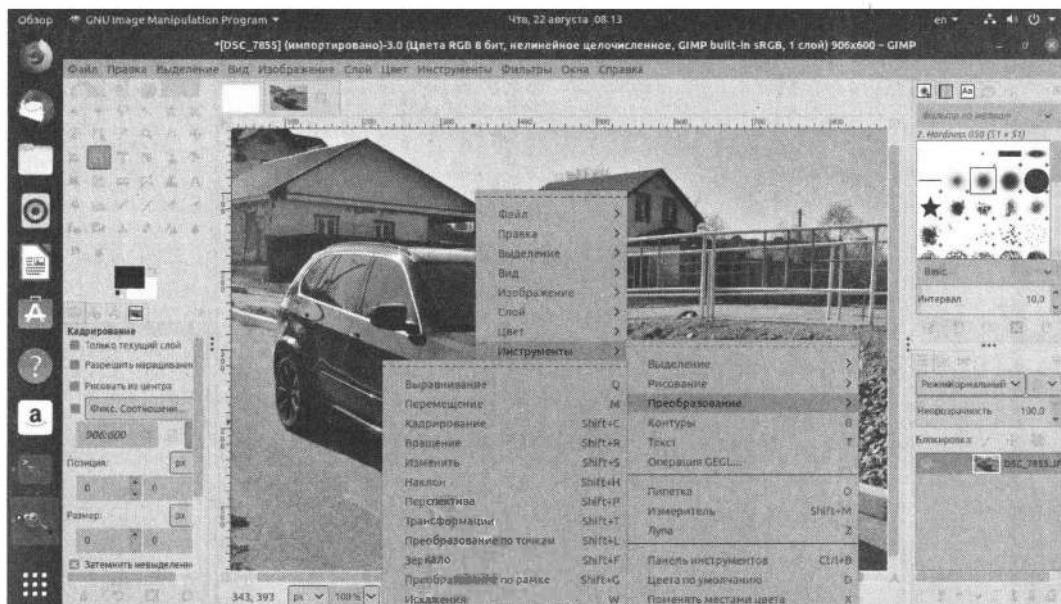


Рис. 15.6. Ubuntu: меню Преобразование GIMP

Если этих изменений вам покажется недостаточно, и вы захотите задать собственные параметры вращения (его угол и центр), воспользуйтесь инструментом **Вращение** (рис. 15.7), вызвать который можно, нажав клавиши **<Shift>+<R>**.

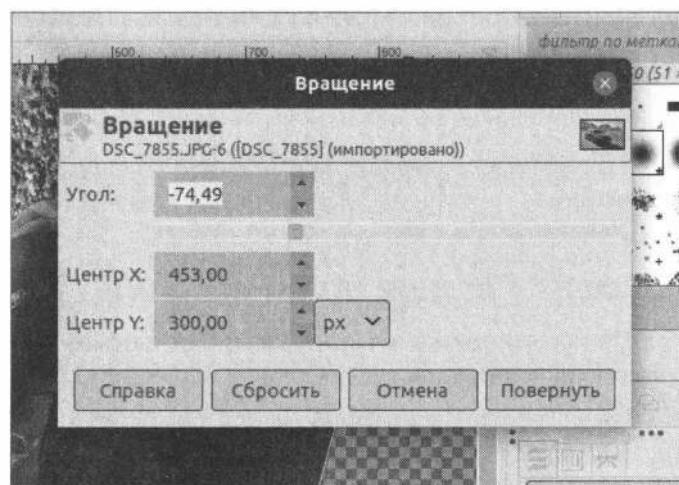


Рис. 15.7. Ubuntu: инструмент Вращение GIMP

### 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

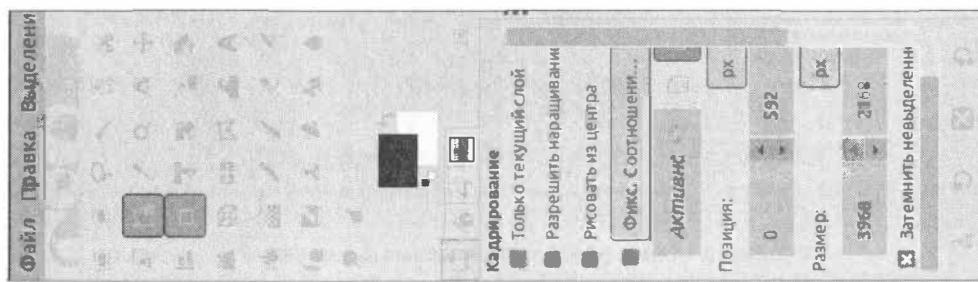
Для кадрирования нажмите комбинацию клавиш **<Shift>+<C>** — указатель мыши примет форму скальпеля. Выделите им прямоугольную область (рис. 15.8) и установите на панели инструментов (рис. 15.9) дополнительные параметры кадрирования, если в них будет необходимость. Чтобы обрезать выделенную часть изображения, нажмите клавишу **<Enter>** или щелкните на выделении 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.

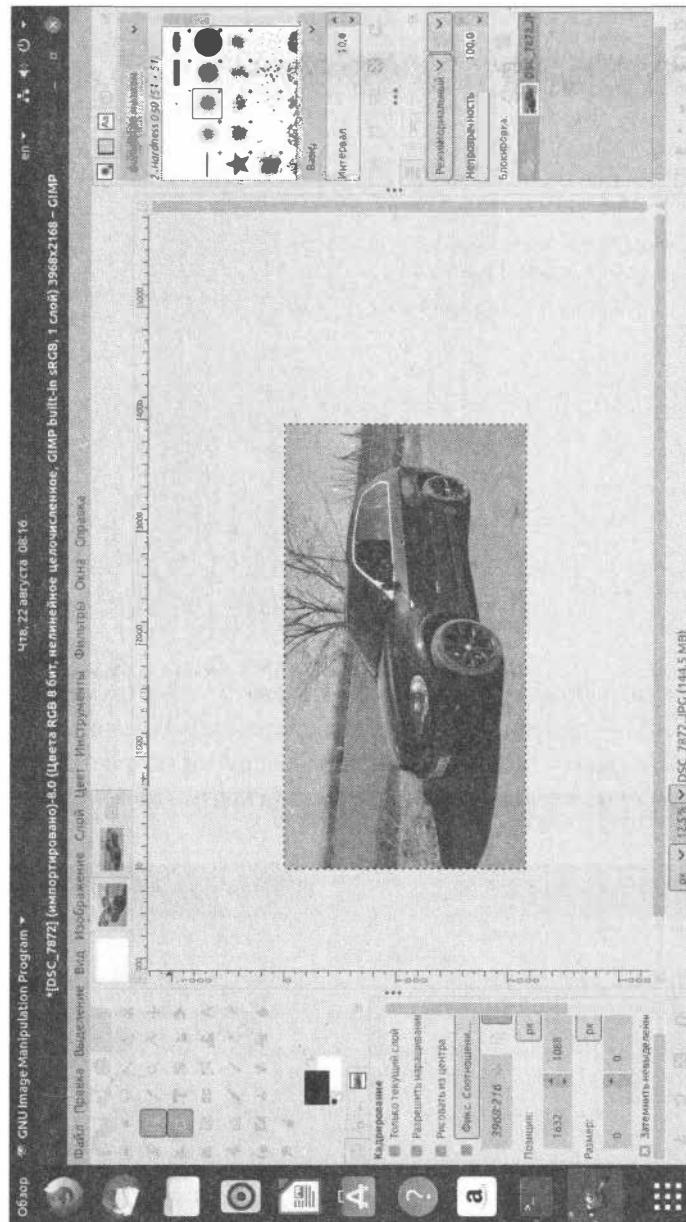
### 15.2.4. Инструмент Размытие-Резкость

Иногда нужно «размыть» некоторые участки картинки или же, наоборот, придать некоторым участкам больше резкости.

Но чаще все-таки используется размытие — для скрытия некоторых участков фотографии, которые совсем необязательно видеть посторонним. Например, весьма часто можно встретить объявления о продаже автомобилей с фотографиями, на которых «размыт» государственный номер.



**Рис. 15.9.** Ubuntu:  
параметры



**Рис. 15.10.** Ubuntu: результат кадрирования в окне GIMP



**Рис. 15.11.** Ubuntu:  
параметры инструмента GIMP  
**Размытие-Резкость**

Для размытия можно использовать инструмент **Размытие-Резкость** (рис. 15.11) — активируйте его, выберите кисть (обычно используется круглая кисть), установите режим (резкость или размытие) и скорость. Теперь вам остается только «размыть» участок изображения. Результат размытия представлен на рис. 15.12.



**Рис. 15.12.** Ubuntu: результат размытия в окне GIMP

### ВИДЕОУРОК ПО GIMP

На страничке <http://dkws.org.ua/novice/> вы найдете небольшой пятиминутный видео-урок по GIMP, в котором продемонстрированы описанные здесь основные операции.

## 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 хватит, 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? Перезагружаться в 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/>.

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.

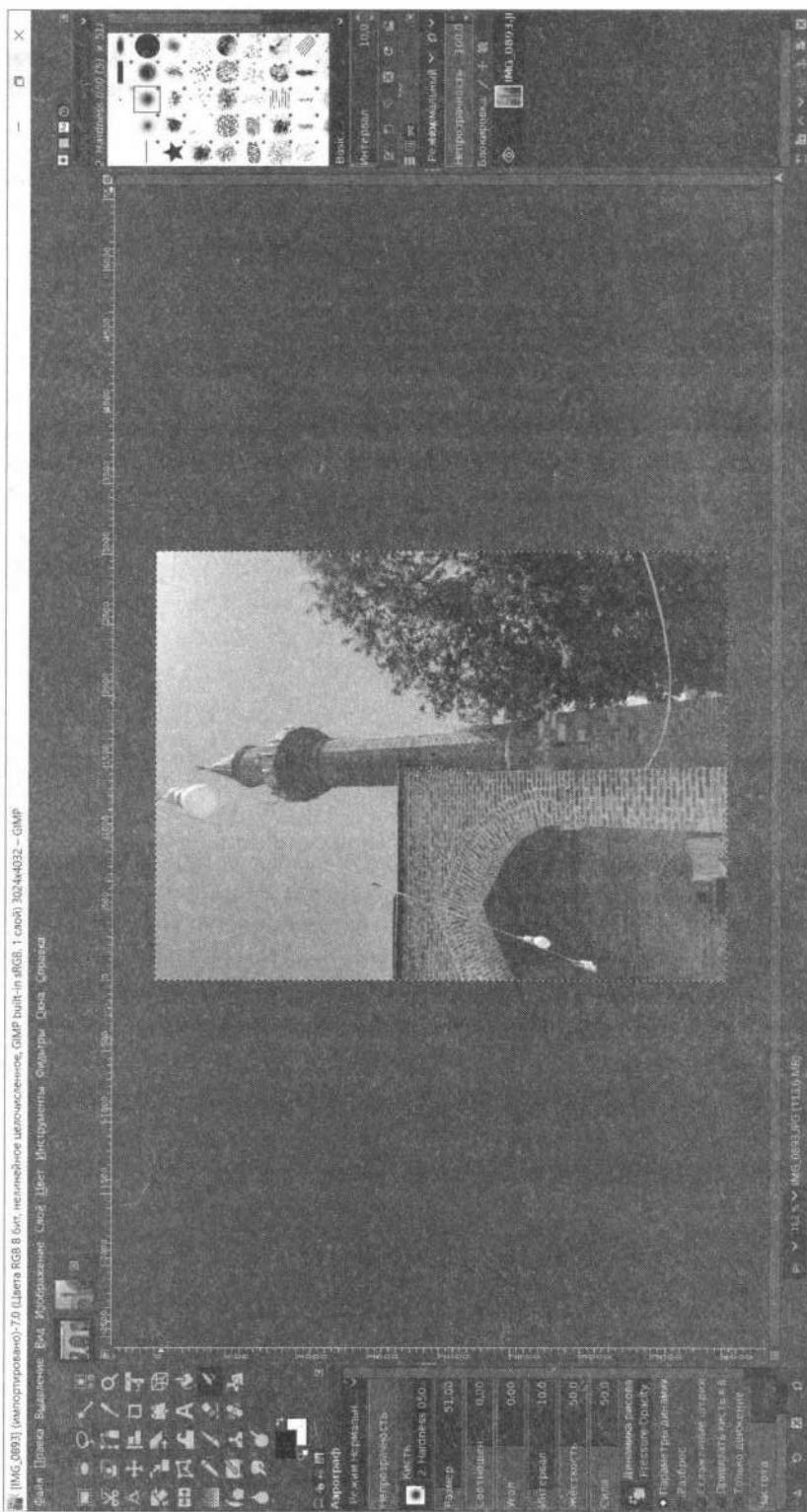
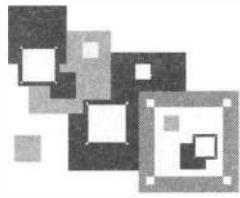


Рис. 15.13. Windows-версия GIMP



## ГЛАВА 16

# Обзор текстовых редакторов кода

Представленные в этой главе текстовые редакторы помогут начинающим разработчикам, переходящим с других платформ на Linux. Для Linux разработано много разных редакторов, и здесь мы рассмотрим две их группы: классические, но несколько архаичные редакторы, пришедшие к нам из эпохи становления Linux, и три современных редактора, обладающих развитым пользовательским интерфейсом.

### 16.1. Текстовые редакторы *vi*, *nano*, *pico*, *ee*, *mcedit*

Со времен первых версий UNIX в современные системы перекочевал текстовый редактор *vi*. То, что ему более сорока 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 такого 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 *vi* до сих пор используется по умолчанию, 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. 16.1 представлен редактор *vi*, в который загружен файл */etc/passwd*.

Редактор *vi* 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.);

```

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
hplip:x:103:7:HPLIP system user,,,:/var/run/hplip:/bin/false
avahi-autoipd:x:104:110:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/bin/false
"/etc/passwd" [readonly] 33 lines, 1617 characters

```

Рис. 16.1. Редактор vi

режим просмотра — используется только для просмотра файла (если надумаете использовать этот режим, вспомните про команду less).

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. 16.1 приведены основные команды редактора vi.

Таблица 16.1. Основные команды редактора vi

| Команда   | Описание                              |
|-----------|---------------------------------------|
| :q!       | Выход без сохранения                  |
| :w        | Сохранить изменения                   |
| :w <файл> | Сохранить изменения под именем <файл> |

Таблица 16.1 (окончание)

| Команда | Описание                                                 |
|---------|----------------------------------------------------------|
| :wq     | Сохранить и выйти                                        |
| :q      | Выход, если нет изменений                                |
| 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 окна редактора, 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 клавиши *<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 различием — 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. 16.2). Раньше он назывался pico и входил в состав почтового клиента pine.

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 (рис. 16.3), который запускается в mc по нажатию клавиши <F4>. Но 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>
```

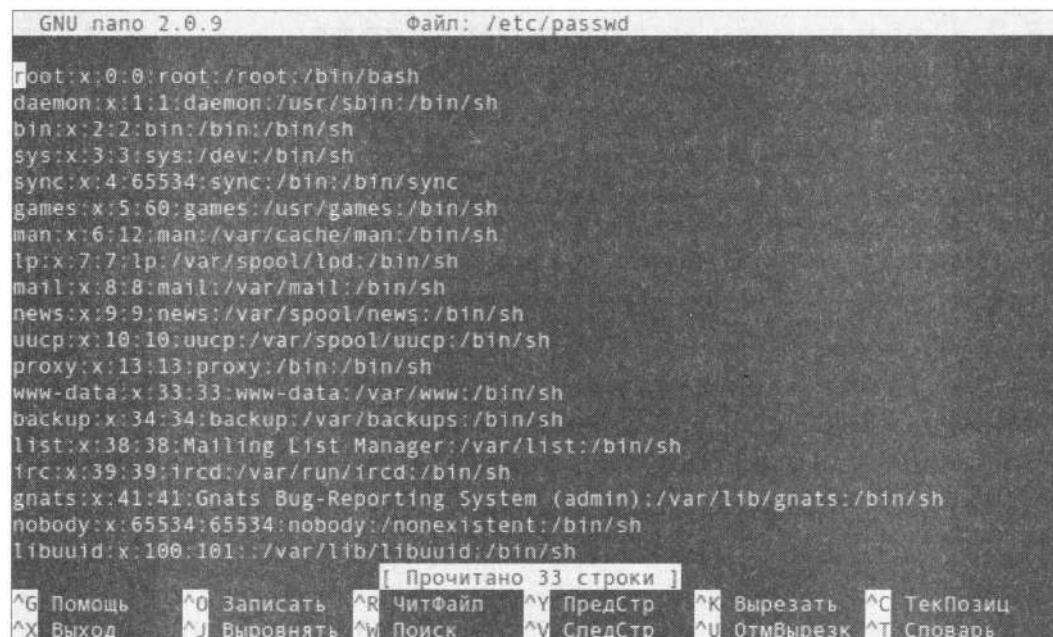


Рис. 16.2. Редактор nano

```
/etc/passwd      [---]  0 L:[ 1+ 0   1/ 34] *(0    /1617b)= r 114 0x72
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
hplip:x:103:7:HPLIP system user...:/var/run/hplip:/bin/false
```

Рис. 16.3. Редактор mcedit

## 16.2. Современные редакторы кода

### 16.2.1. Atom

Atom (рис. 16.4) — редактор с открытым исходным кодом, который подойдет абсолютно всем разработчикам: от новичка до эксперта. Основное преимущество этого редактора в его гибкости — Atom можно настроить для любых потребностей. В настоящее время Atom является лучшим текстовым редактором для программиста — во всяком случае, на мой взгляд.

Преимущества редактора:

- отличное автодополнение, позволяющее писать код быстрее;
- встроенный менеджер пакетов, позволяющий инсталлировать новые пакеты, расширяющие функционал редактора;
- наличие встроенного браузера файловой системы;
- кроссплатформенность — редактор также доступен в Windows и macOS.

Скачать DEB- или RPM-пакет с редактором можно с официального сайта по адресу: [atom.io](http://atom.io).

#### **ПРИМЕЧАНИЕ**

Редактор Atom использует модную нынче темную тему оформления. Окна с таким оформлением, увы, не всегда корректно отображаются при печати, поэтому цвета на рис. 16.4 были инвертированы.

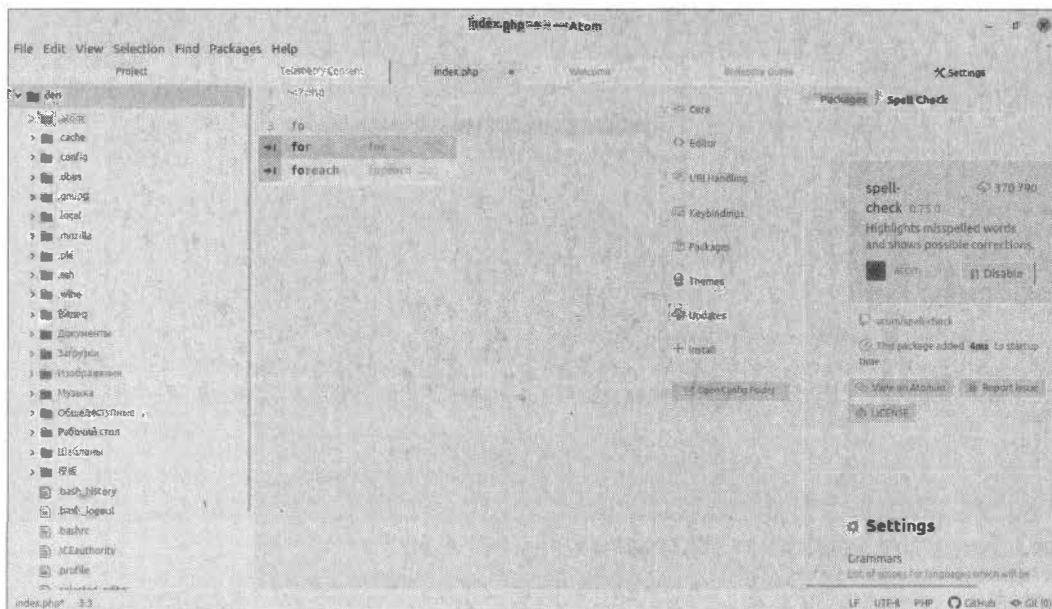


Рис. 16.4. Редактор Atom: автодополнение в действии

## 16.2.2. Sublime Text 3

Sublime Text (рис. 16.5) — один из самых распространенных редакторов кода в сообществе Linux. Это весьма простой и быстрый редактор кода. При желании его можно использовать в качестве обычного текстового редактора.

Редактор поддерживает много языков программирования, в нем работают как автодополнение, так и подсветка синтаксиса.

Преимущества редактора:

- очень простой интерфейс;
- неплохое автодополнение;
- поддержка множества языков программирования.

Но и у него есть недостатки, а именно сложность интегрирования дополнительного функционала. Если для расширения функционала редактора Atom требуется всего лишь установить нужный пакет, то здесь придется постараться.

Инструкции по установке редактора приведены по ссылке: [https://www.sublimetext.com/docs/3/linux\\_repositories.html](https://www.sublimetext.com/docs/3/linux_repositories.html).



Рис. 16.5. Редактор Sublime Text

## 16.2.3. Brackets от Adobe

Текстовый редактор Brackets (рис. 16.6) — разработка компании Adobe. В отличие от других продуктов Adobe, этот редактор полностью бесплатный. Brackets обладает рядом особенностей, которые могут быть расширены дополнительными плаги-

нами. В Adobe работали над Brackets с тем, чтобы создать лучший современный редактор кода для Linux.

Проще всего установить редактор в виде снапа:

```
sudo apt install snapd
sudo snap install brackets --classic
```

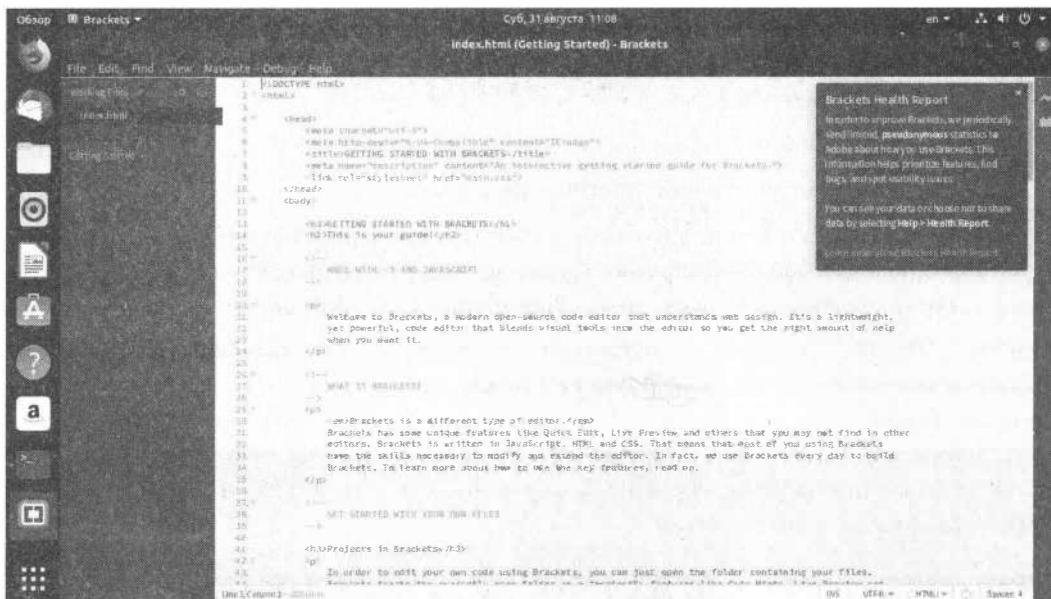


Рис. 16.6. Текстовый редактор Brackets от Adobe

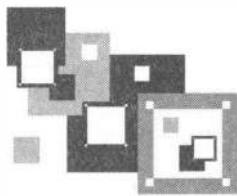
### Преимущества редактора Brackets:

- Live-просмотр** — вы можете сразу отслеживать результат изменения кода;
- открытый исходный код;**
- мощные визуальные инструменты;**
- доступность полезных расширений.**

\* \* \*

Какой редактор выбрать? Известно, что многообразие выбора порождает свою проблему — проблему выбора. Попробуйте поработать с каждым редактором, а потом решите, какой из них лучше для вас.

# 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 о его усовершенствовании.

Как известно, браузер Firefox (рис. 17.1) 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 все еще присутствуют, хотя разработчики и стараются отказаться от этой технологии.

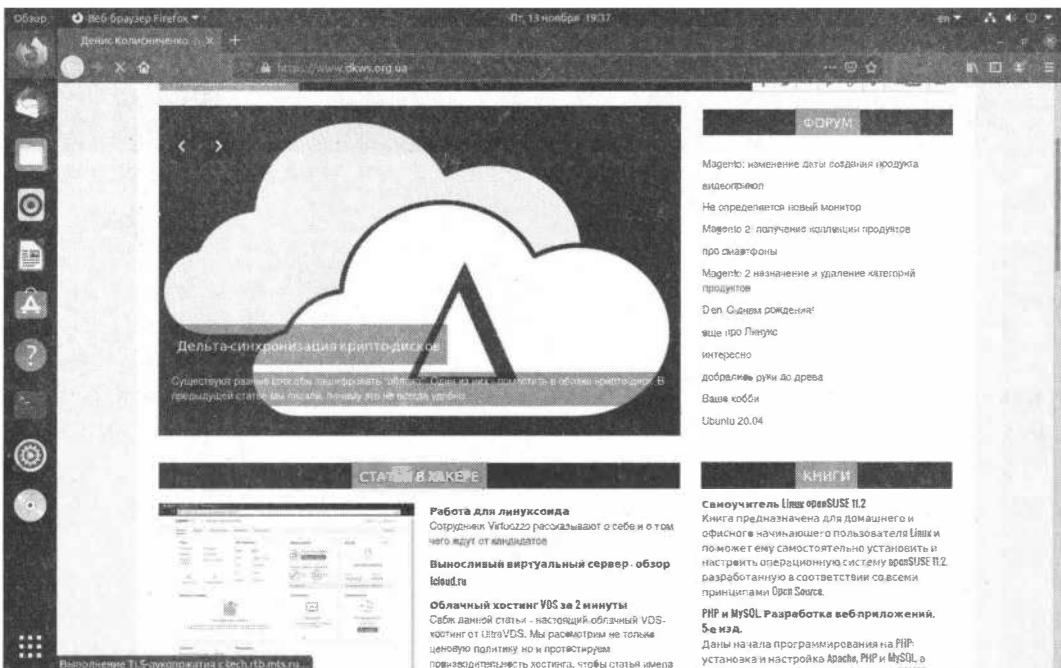


Рис. 17.1. Браузер Firefox в Ubuntu 20.10

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 вовсе 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. Однако в 2020 году было решено отказаться от Flash, поскольку этот плагин был признан небезопасным. В Ubuntu 20.10 нет пакета `flashplugin-installer`, обеспечивающего установку Flash-плагина. Если вы все еще хотите посещать сайты, которые до сих пор используют Flash, вам нужно использовать Ubuntu 20.04 — это последняя версия Ubuntu, которая обеспечивала поддержку Flash.

## 17.2. Браузер Chromium

Браузер Google Chrome становится все популярнее и популярнее. 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. 17.2) — 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 репозиториев большинства современных дистрибутивов, поэтому его не составит труда установить стандартными средствами (рис. 17.3).

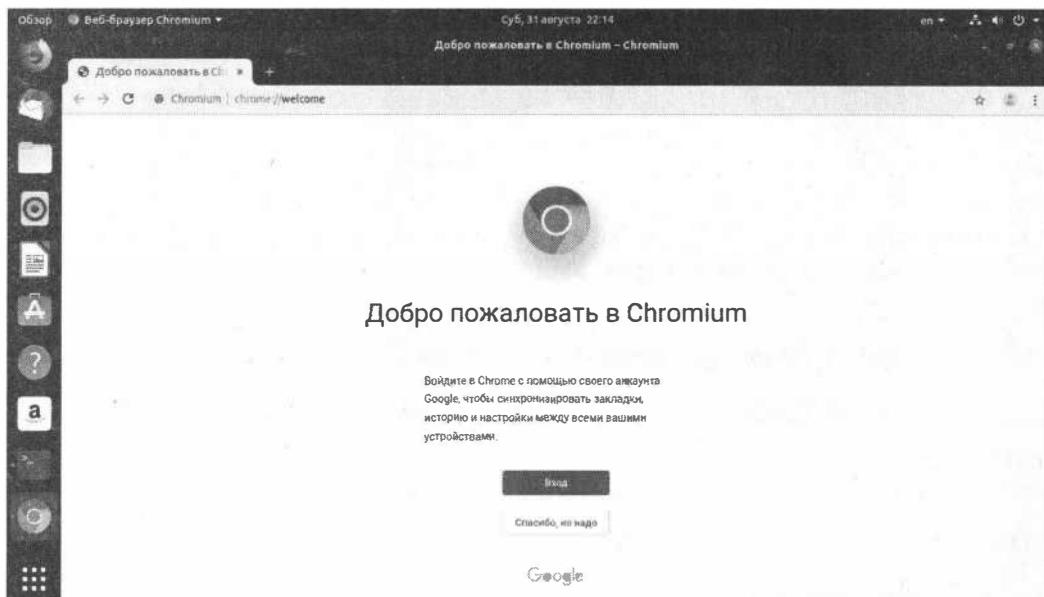


Рис. 17.2. Ubuntu: браузер Chromium

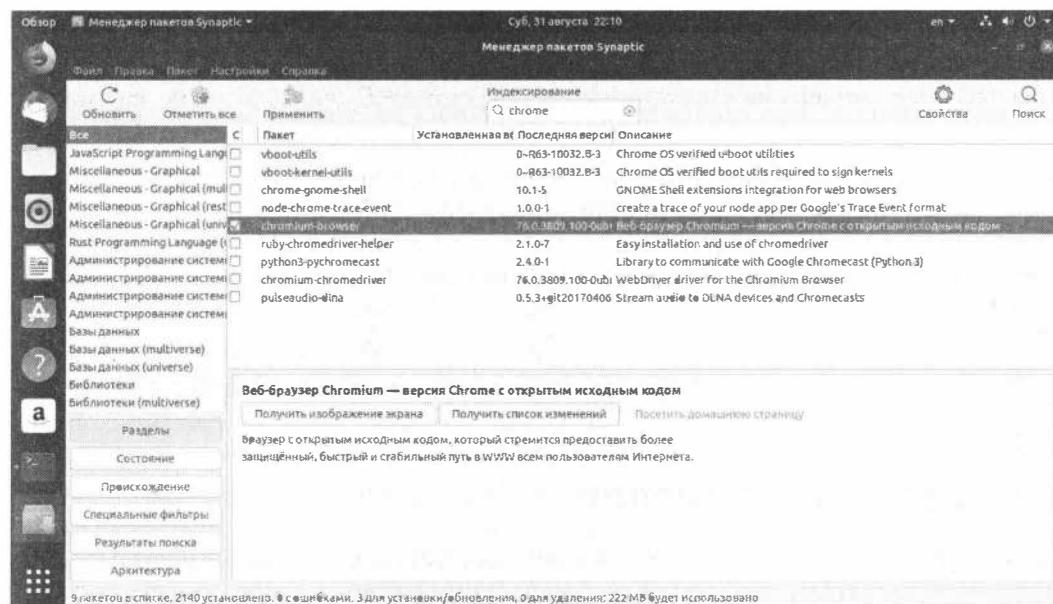


Рис. 17.3. Ubuntu: установка браузера Chromium с помощью менеджера пакетов Synaptic

## 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, су-

мевший 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 перекочевали 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 сайт [www.skype.com](https://www.skype.com) и скачать версию для своего дистрибутива (рис. 17.4).

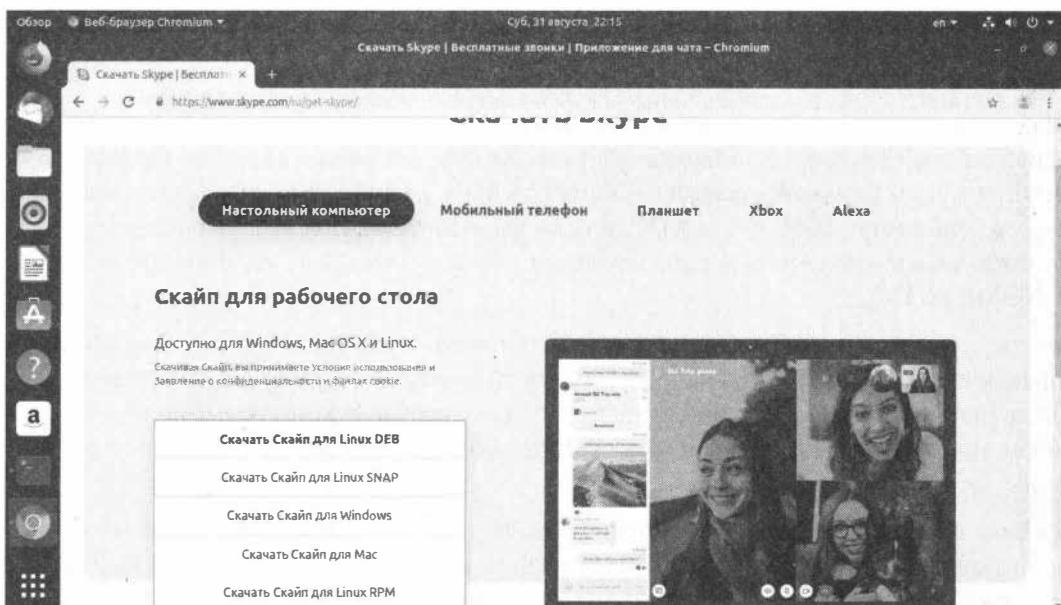


Рис. 17.4. Ubuntu: загрузка Skype

Далее все просто: перейдите в папку загрузок и щелкните на пакете двойным щелчком — откроется окно, в котором нужно нажать кнопку **Установить** (рис. 17.5). Установив пакет, можете запустить Skype (рис. 17.6). Вот оно и свершилось — Microsoft добралась до Linux ☺.

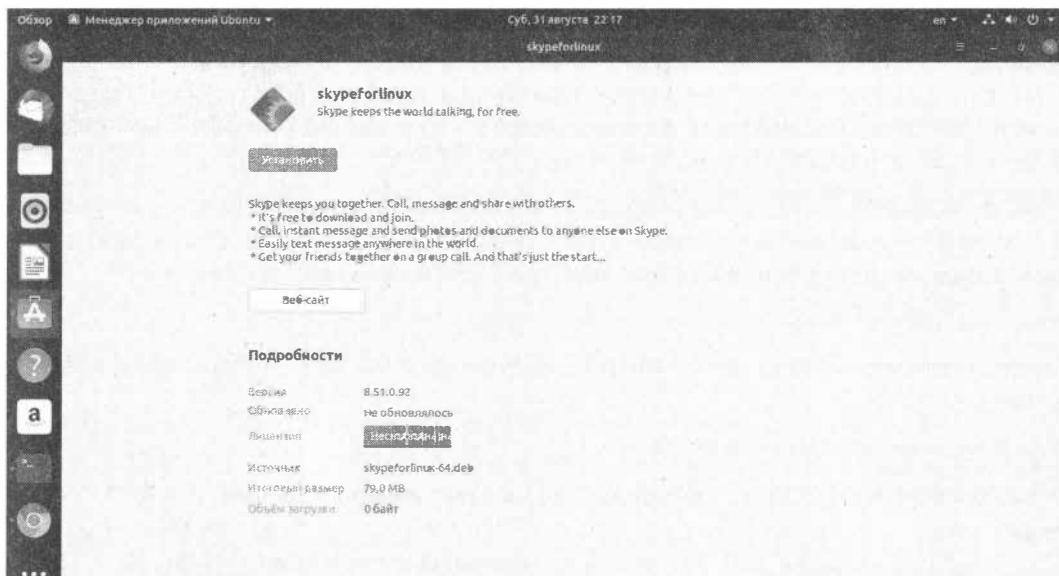


Рис. 17.5. Ubuntu: установка Skype

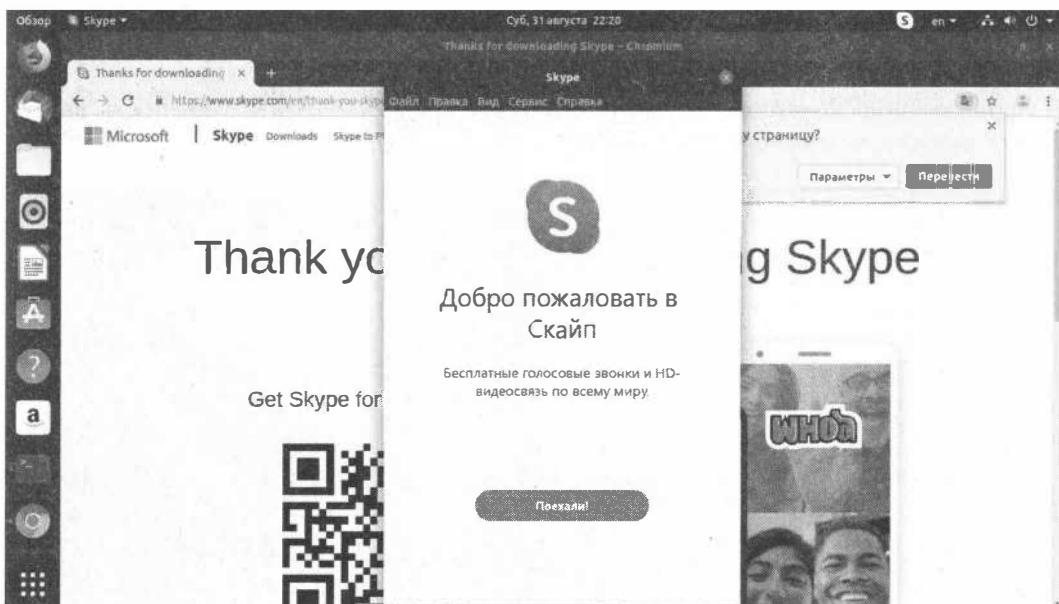


Рис. 17.6. Ubuntu: Skype запущен

## 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 недотягивают 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 26–33`, 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. 17.7).

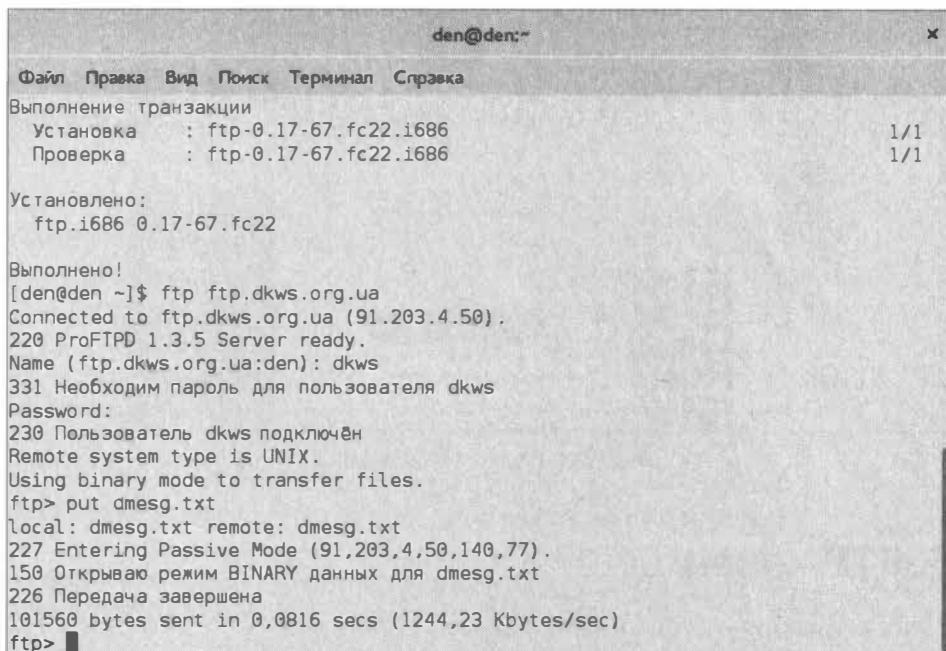


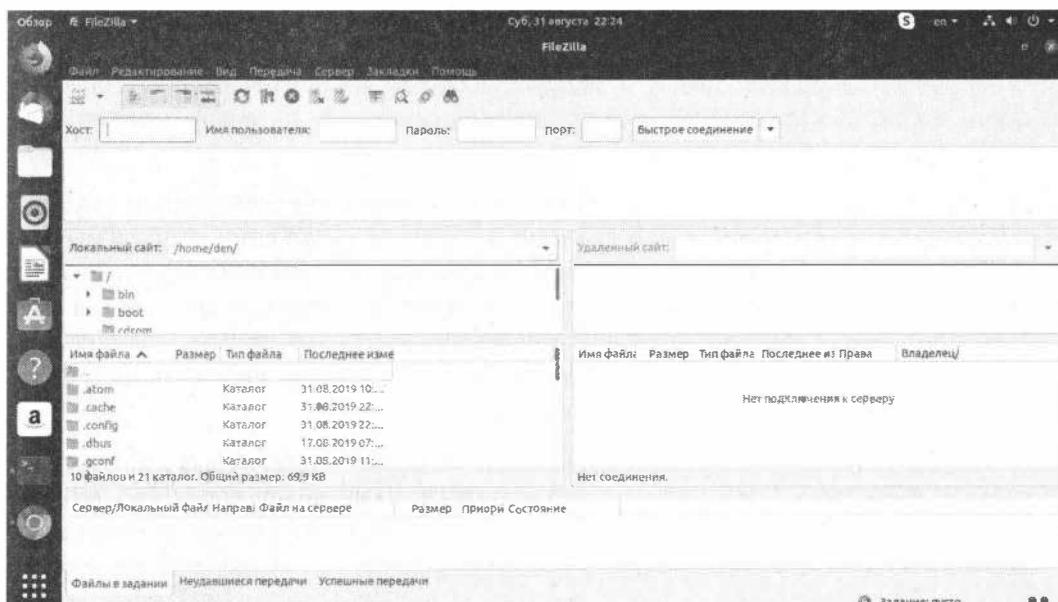
Рис. 17.7. Fedora: команда `ftp`

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.

**Таблица 17.1. Некоторые команды FTP-клиента**

| Команда             | Описание                                                                                                   |
|---------------------|------------------------------------------------------------------------------------------------------------|
| <code>ls</code>     | Вывод содержимого каталога                                                                                 |
| <code>get</code>    | Загрузить файл с сервера                                                                                   |
| <code>put</code>    | Загрузить файл на сервер                                                                                   |
| <code>mget</code>   | Получить несколько файлов с сервера. Допускается использование масок файлов — например: <code>*.grm</code> |
| <code>mput</code>   | Загрузить несколько файлов на сервер                                                                       |
| <code>cd</code>     | Изменить каталог                                                                                           |
| <code>mkdir</code>  | Создать каталог                                                                                            |
| <code>rmdir</code>  | Удалить пустой каталог                                                                                     |
| <code>delete</code> | Удалить файл                                                                                               |

Из графических FTP-клиентов самым лучшим, на мой взгляд, FTP-клиентом является FileZilla, версии которого существуют как для Linux (рис. 17.8), так и для Windows. Linux-версия FileZilla входит в состав многих современных дистрибутивов — для ее установки нужно установить пакет `filezilla`.



**Рис. 17.8. Ubuntu: FTP-клиент FileZilla**

## 17.6. P2P-клиенты

В январе 1999 года Шон Фэннинг, восемнадцатилетний студент одного из американских вузов, написал программу, позволяющую обмениваться MP3-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 совершил настоящий прорыв в компьютерных технологиях.

Что же сделал Шон Фэннинг? Им была создана так называемая *пиинговая сеть* (Peer-to-peer, peer2peer, P2P). Не нужно путать ее с обычной одноранговой сетью, которая существовала с самого начала развития сетевых технологий. Но у пиинговой 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 (разумеется, только к им избранным), и все остальные участники пиинговой сети смогут скачать эти файлы. Ясно, что этот пользователь должен находиться в сети, иначе файлы скачать будет невозможно.

Существуют две модели пиинговых сетей: централизованные и децентрализованные. Первая пиинговая 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 пиинговая сеть будет разрушена. Поэтому следующий виток в развитии пиинговых 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 пиинговые сети, представляющие собой нечто среднее между централизованной и децентрализованной моделями.

В пиинговых 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 пиинговых 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 пиинговым протоколом является BitTorrent. Для работы с ним во всех дистрибутивах Linux используется программа Transmission (рис. 17.9), которая устанавливается по умолчанию. Именно с помощью этой программы можно качать с торрентов музыку, фильмы и видео.

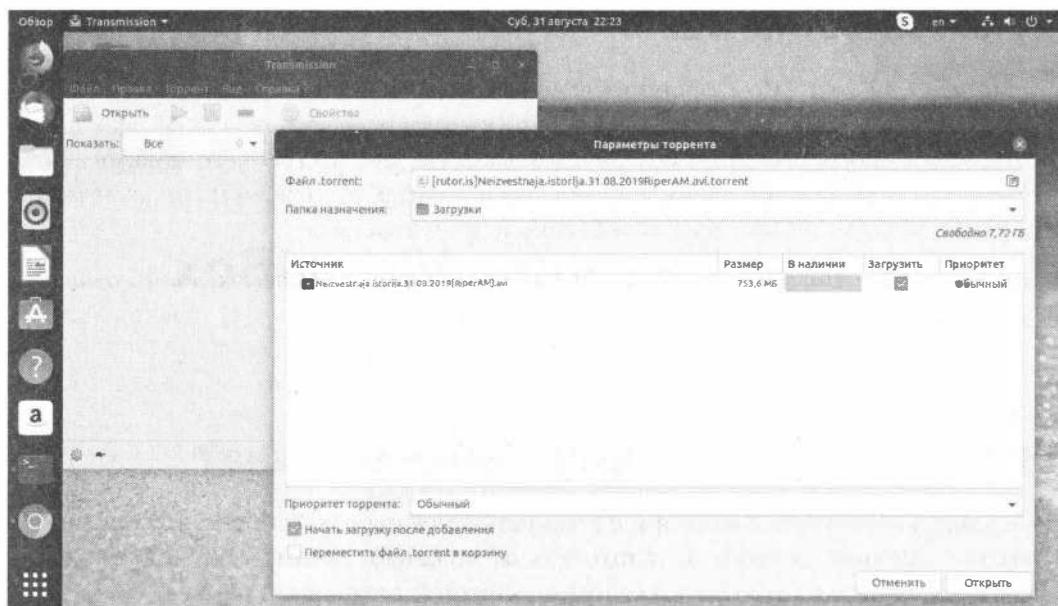
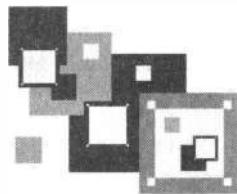


Рис. 17.9. Ubuntu: программа Transmission

## ГЛАВА 18



# 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.

Кому нужна виртуальная машина? По большому счету — в основном разработчикам программного обеспечения: 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 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 виртуаль-

ной 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 4 гигабайта оперативной памяти: 2 — останется для основной операционной системы (современные дистрибутивы Linux откровенно медленно работают на системах с меньшим объемом памяти), а 2 — будет отдано виртуальной машине. Опять-таки многое зависит еще и от гостевой ОС. Одно дело запускать в виртуальной машине Linux без графического интерфейса (вроде Fedora Server), 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 должны 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 44 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. Для этого в терминале введите команду:

```
sudo usermod -a -G vboxusers <имя_пользователя>
```

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).

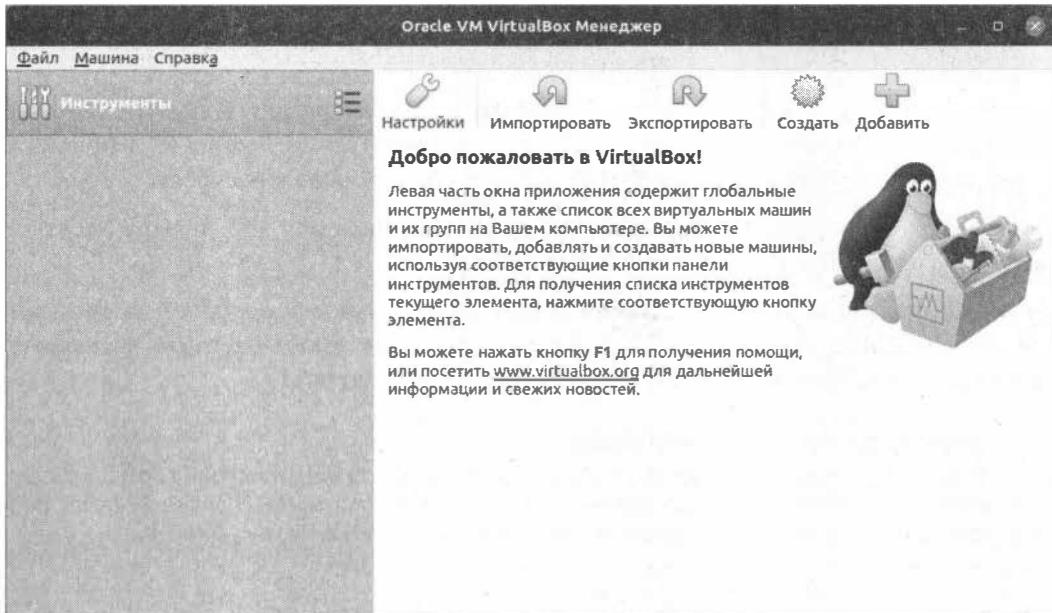


Рис. 18.1. Ubuntu: менеджер виртуальных машин VirtualBox

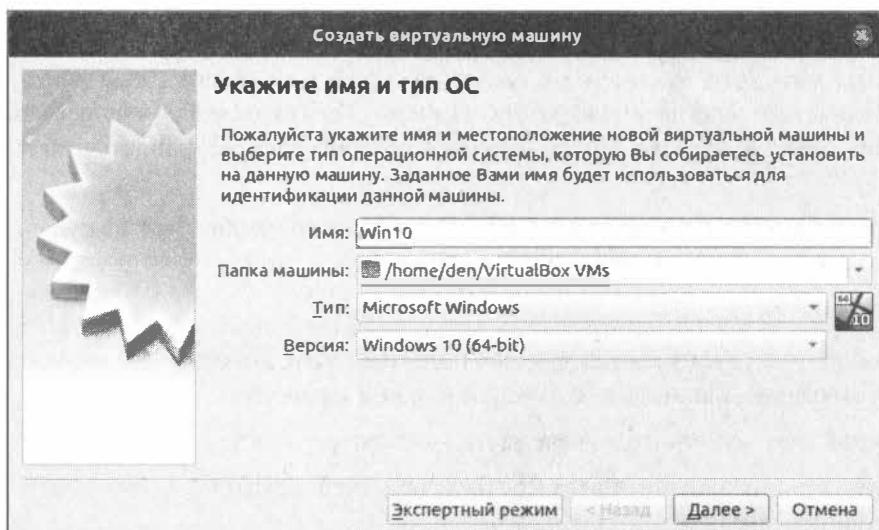


Рис. 18.2. Менеджер виртуальных машин VirtualBox: выбор гостевой ОС

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 2 гигабайта. Я выбрал **Windows 10 (64-bit)**, а для нее минимальный размер ОЗУ равен 2 гигабайта, т. е. меньше — нельзя. Но поскольку у моего компьютера ОЗУ имеет объем 6 Гбайт, то я смело выделил половину — 3 гигабайта — под виртуальную машину (рис. 18.3).

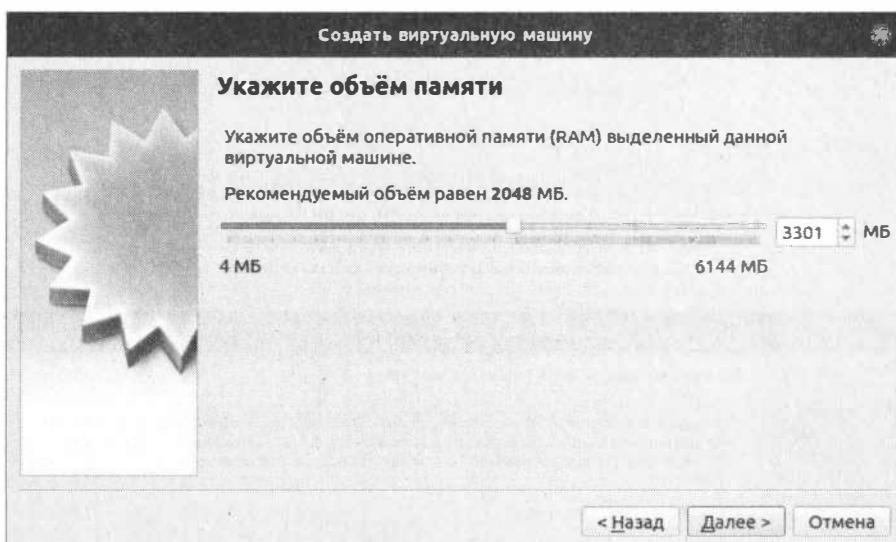


Рис. 18.3. Менеджер виртуальных машин VirtualBox:  
установка размера оперативной памяти для гостевой операционной системы

#### ПРИМЕЧАНИЕ

В списке гостевых только 32-разрядные версии операционных систем? Перейдите в BIOS SETUP и включите виртуализацию в расширенных (**Advanced**) настройках. Обычно нужный параметр называется **Intel VT**. Например, в ASUS BIOS нужно перейти в раздел **Advanced**, а затем — в **CPU Configuration**, где включить параметр **Intel Virtualization Technology**.

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) и нажмите кнопку **Создать** (Create). В дальнейшем для использования уже существующего файла образа жесткого диска можно выбрать его из списка по варианту **Использовать существующий виртуальный жесткий диск** (Use existing hard disk).

Следующий шаг — выбор типа файла, определяющего формат 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)**.

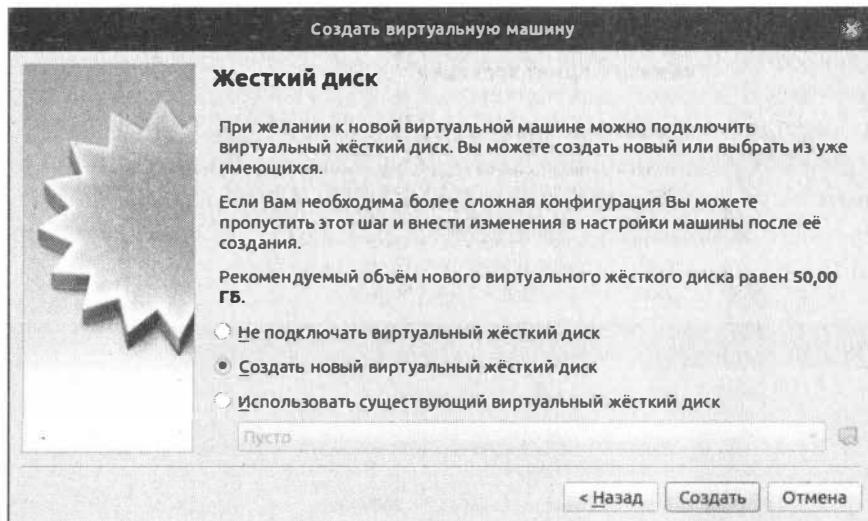


Рис. 18.4. Менеджер виртуальных машин VirtualBox:  
создание нового виртуального жесткого диска

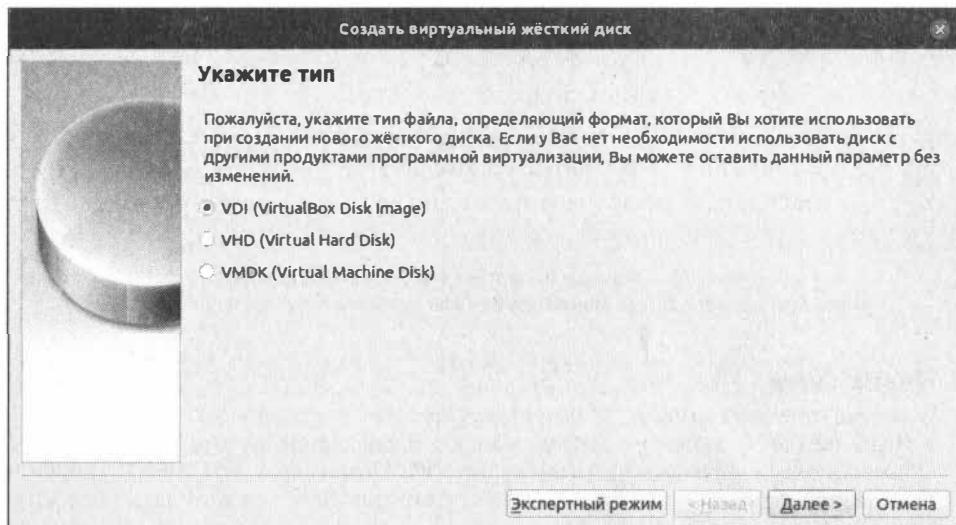


Рис. 18.5. Менеджер виртуальных машин VirtualBox: выбор формата файла виртуального диска

Нажав кнопку **Вперед** (Next), мы попадем в окно (рис. 18.6), где нужно определить формат хранения данных на виртуальном жестком диске, — то есть способ задания размера файла этого диска:

- Динамический виртуальный жесткий диск** (Dynamically allocated) — размер файла будет увеличиваться или уменьшаться по мере необходимости, но не превысит заданного вами предела;
- Фиксированный виртуальный жесткий диск** (Fixed size) — размер будет четко зафиксирован вне зависимости от того, сколько места реально занимает гостевая операционная система.

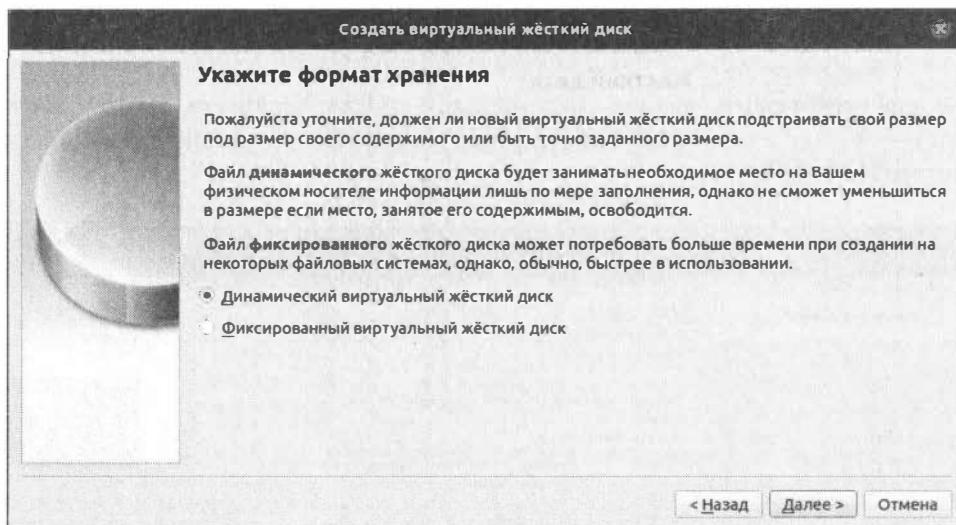


Рис. 18.6. Менеджер виртуальных машин VirtualBox:  
выбор формата хранения данных на виртуальном жестком диске

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 **Далее**.

Сразу после выбора формата хранения данных (метода резервирования места на диске) нужно установить размер создаваемого виртуального диска (рис. 18.7).

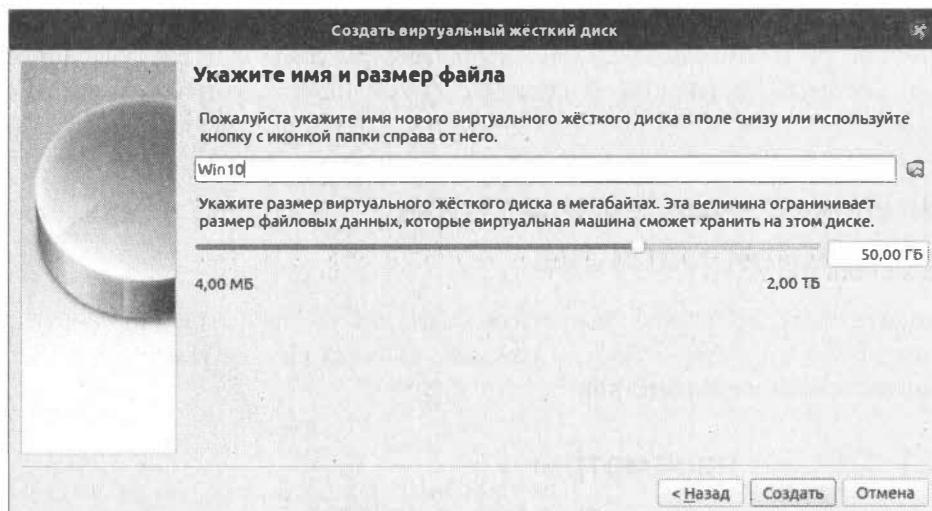


Рис. 18.7. Менеджер виртуальных машин VirtualBox: задание размера виртуального диска

В моем случае эмулятор рекомендует установить размер 50 Гбайт. Рекомендуемый размер диска зависит от выбранной гостевой операционной системы.

Установив требуемый размер виртуального диска, нажмите кнопку **Создать** (Create) — VirtualBox создаст виртуальную машину, и вы увидите основное окно эмулятора (рис. 18.8).

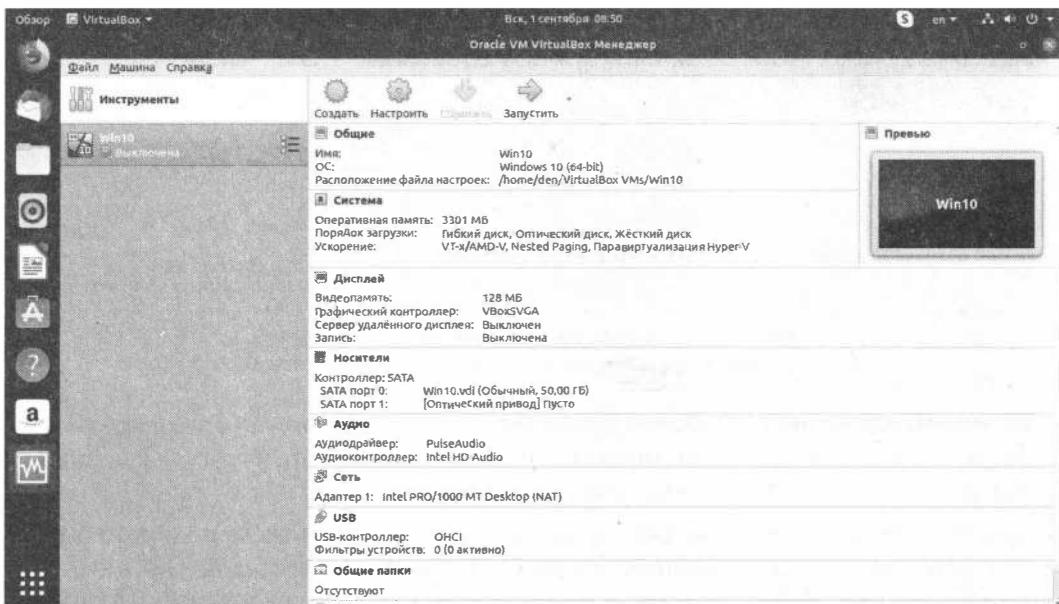


Рис. 18.8. Ubuntu: созданная виртуальная машина в окне VirtualBox

В предыдущих версиях VirtualBox 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** (рис. 18.9) вы можете изменить общие параметры ВМ: название, тип гостевой ОС, версию гостевой ОС, а на вкладке

**Дополнительно** — задать различные дополнительные параметры: например, определить папку для снимков с экрана, включить общий буфер обмена и т. д.

#### ПРИМЕЧАНИЕ

Если VirtualBox обнаружит неправильные настройки, то вы не сможете сохранить конфигурацию виртуальной машины, пока не исправите их.

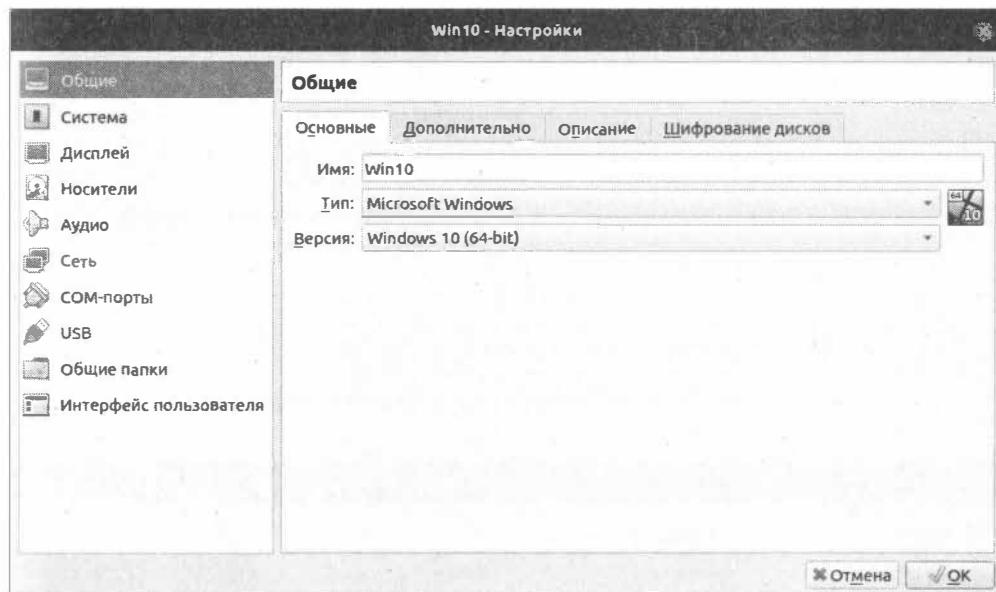


Рис. 18.9. Менеджер виртуальных машин VirtualBox: общие параметры виртуальной машины

#### 18.4.2. Раздел Система

В разделе **Система** на вкладке **Материнская плата** (рис. 18.10) можно установить объем оперативной памяти, выбрать порядок загрузки и установить другие параметры — для работы некоторых ОС приходится, например, включать EFI. 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 добавить образы дисков, а в новой — можно даже выбрать контроллер (IDE или SATA), к которому будет «подключен» виртуальный носитель. По умолчанию здесь к SATA-контроллеру подключен всего один образ — тот, который вы организовали при создании ВМ. Обратите внимание: здесь же выводится реальный размер, занимаемый образом, и его максимально возможный размер.

Также ранее в VirtualBox существовал раздел **CD/DVD-ROM**, в котором можно было установить параметры виртуального привода DVD. Сейчас эти параметры

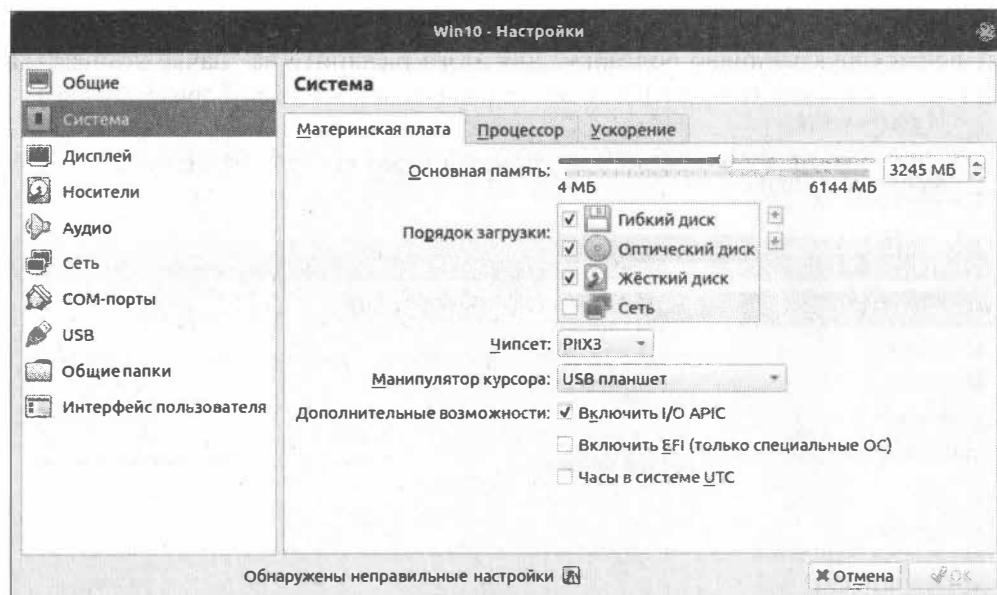


Рис. 18.10. Менеджер виртуальных машин VirtualBox: раздел Система

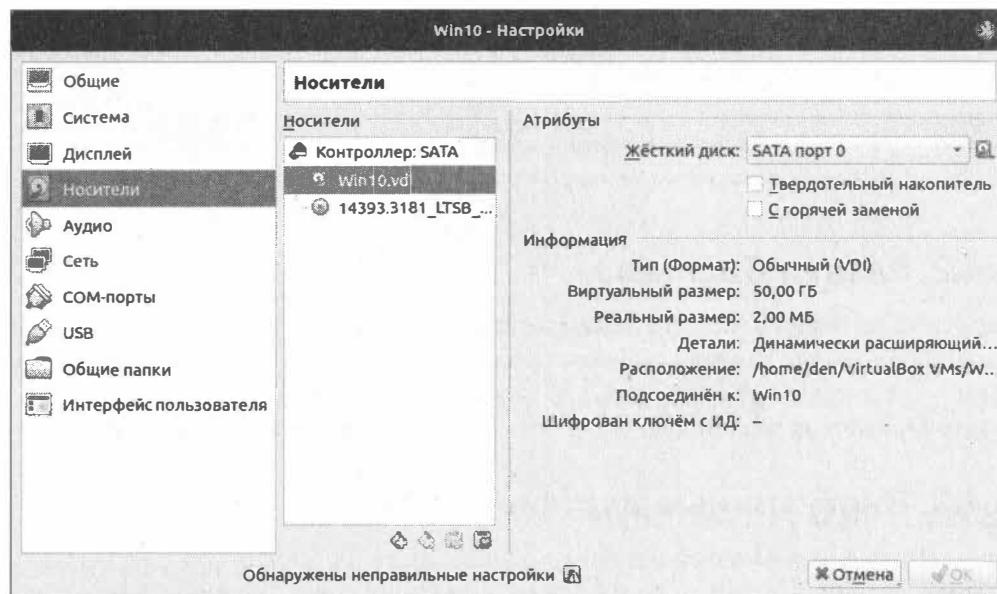


Рис. 18.11. Менеджер виртуальных машин VirtualBox: параметры виртуальных жестких дисков

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 на значке компакт-диска в дереве носителей (рис. 18.12) и 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.

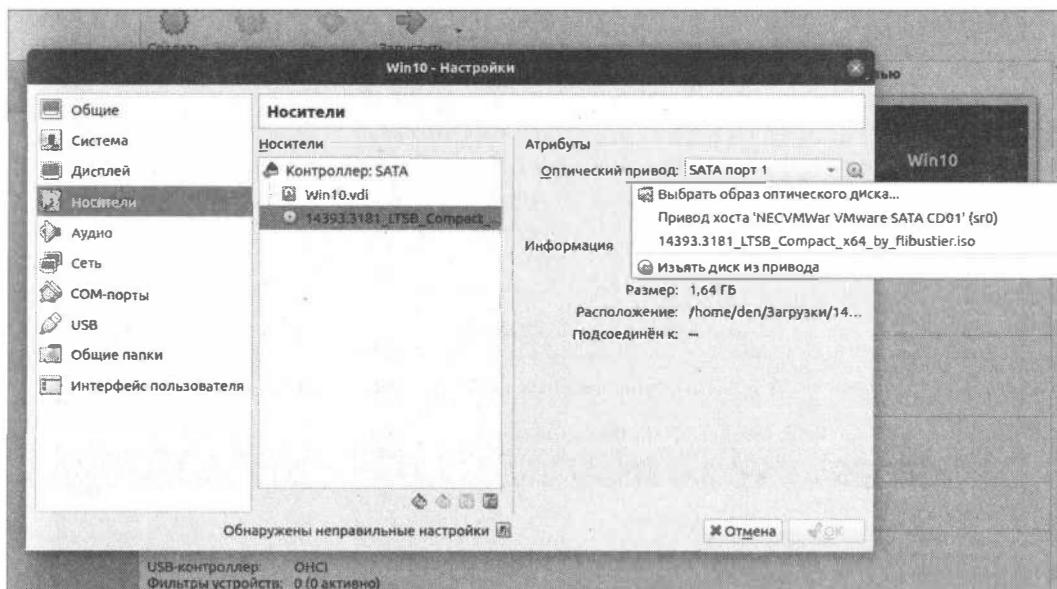


Рис. 18.12. Менеджер виртуальных машин VirtualBox: выбор образа оптического диска

#### 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 PulseAudio, поэтому в параметрах ВМ нужно выбрать именно ее.

#### 18.4.5. Параметры сети

В разделе **Сеть** (рис. 18.14) вы можете определить, как будет гостевая операционная система взаимодействовать по сети с основной.

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 обычный компьютер и станет виден остальным компьютерам сети. Недостаток этого способа — необходимость

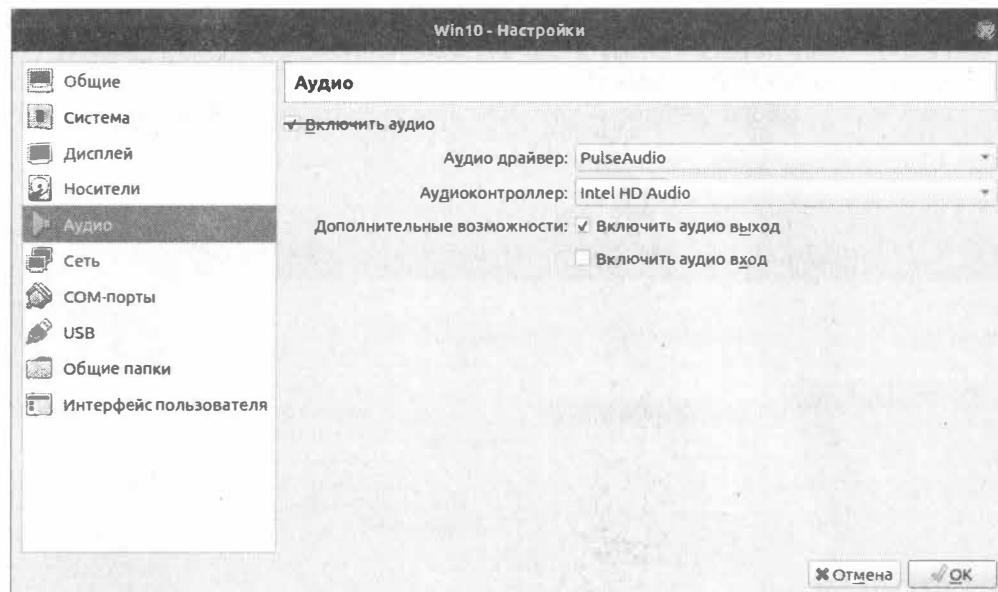


Рис. 18.13. Менеджер виртуальных машин VirtualBox: параметры звука



Рис. 18.14. Менеджер виртуальных машин VirtualBox: параметры сети

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 **СОМ-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 **OK** для возврата в основное окно VirtualBox.

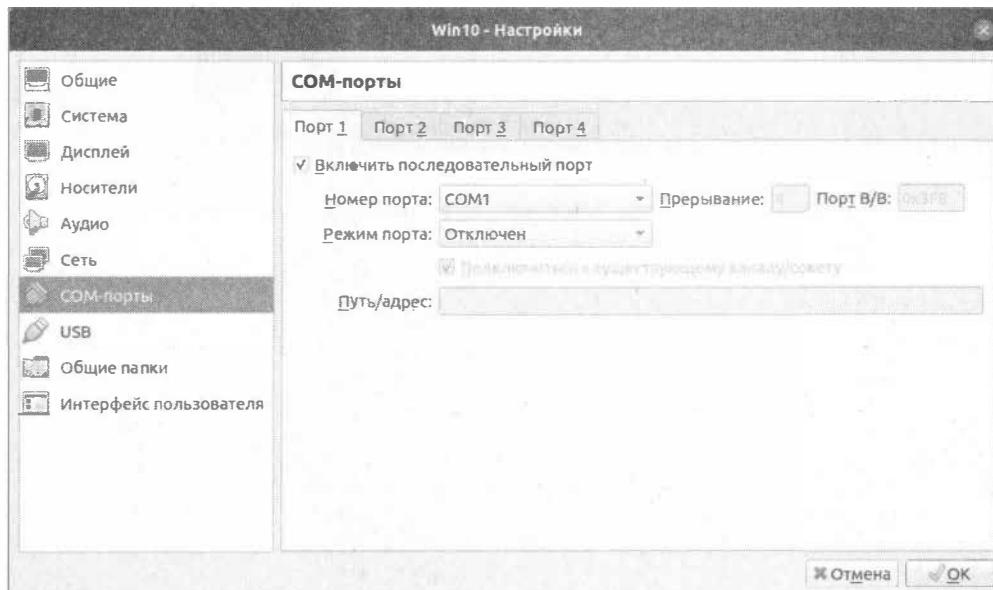


Рис. 18.15. Менеджер виртуальных машин VirtualBox: использование последовательных портов

## 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.

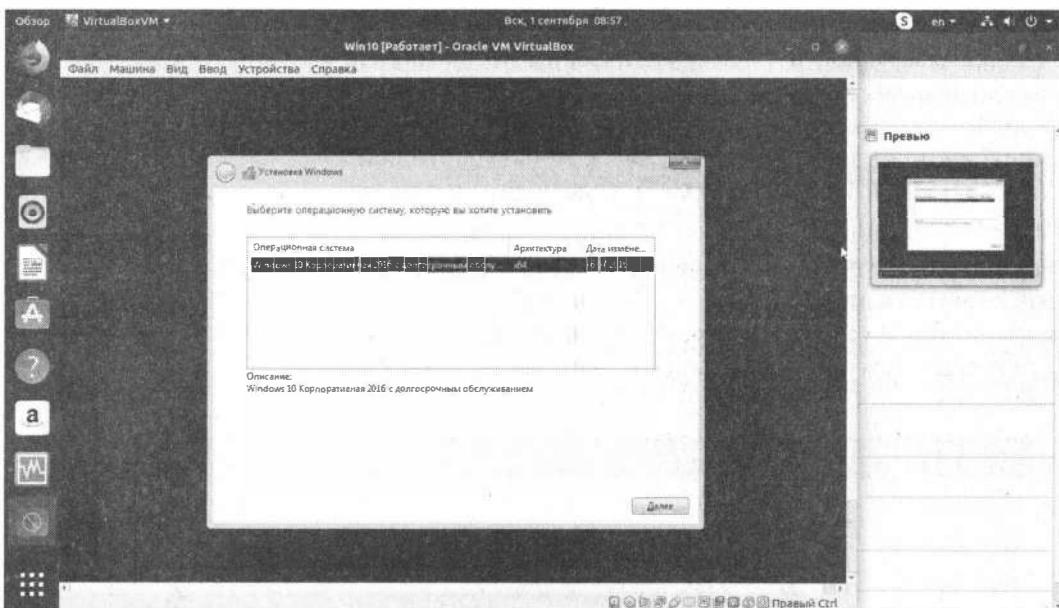
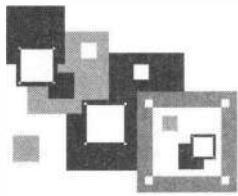


Рис. 18.16. Ubuntu: запуск виртуальной машины в VirtualBox



## 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. 26 лет назад! Развивался он сначала медленно (тогда просто не было острой необходимости в запуске Windows-приложений из-под Linux), 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.

## 19.2. Установка Wine

Установите Wine — или через Synaptic, или с помощью apt (рис. 19.1) — как вам удобнее.

## **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.



**Рис. 19.1.** Ubuntu 20.10: установка Wine

К сожалению, в Ubuntu 20.10 не были созданы какие-либо значки, упрощающие запуск эмулятора. Если в версии Ubuntu 17.04 (рис. 19.02) на панель **Приложения** попадало хоть что-то, то в версиях 19.04/20.10 управлять эмулятором придется только через консоль.

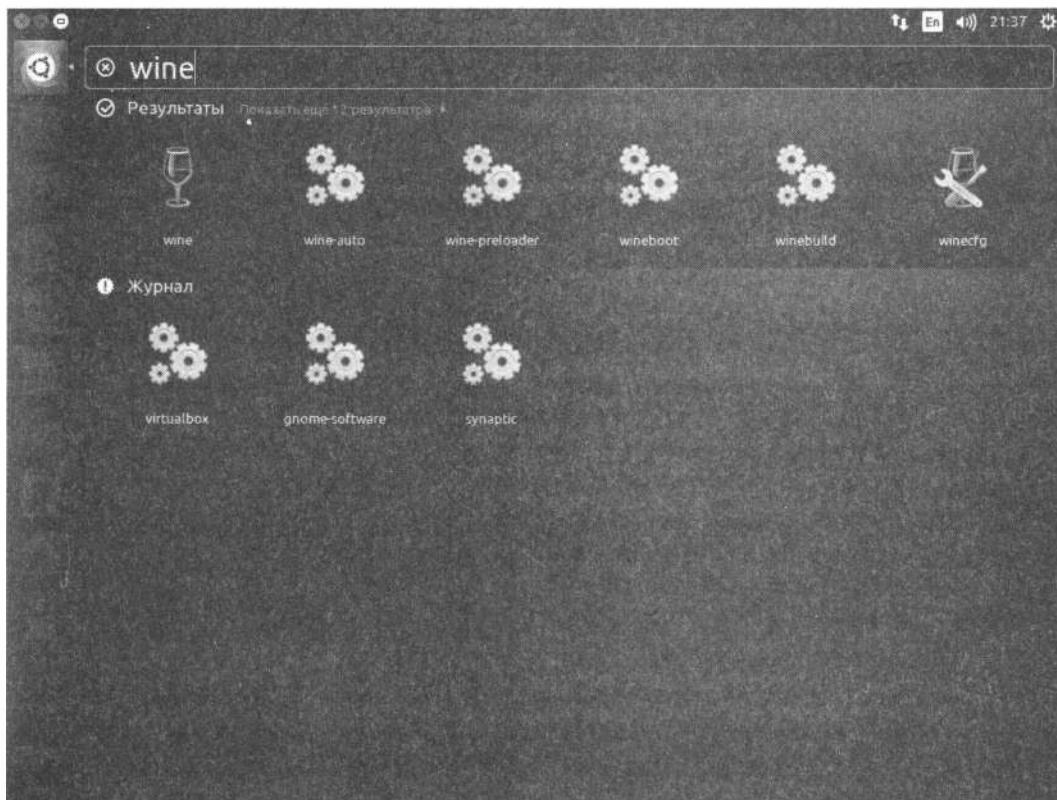


Рис. 19.2. Ubuntu 17.04: значки группы Wine

### 19.3. Настройка Wine и прозрачного запуска Windows-приложений

Перед тем как приступить к установке Windows-приложений, Wine необходимо настроить. Откройте терминал и запустите программу winecfg. В открывшемся окне (рис. 19.3) на вкладке **Приложения** вы можете выбрать версию Windows. К настоящему моменту Wine уже поддерживает самую новую версию — Windows 10.

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.

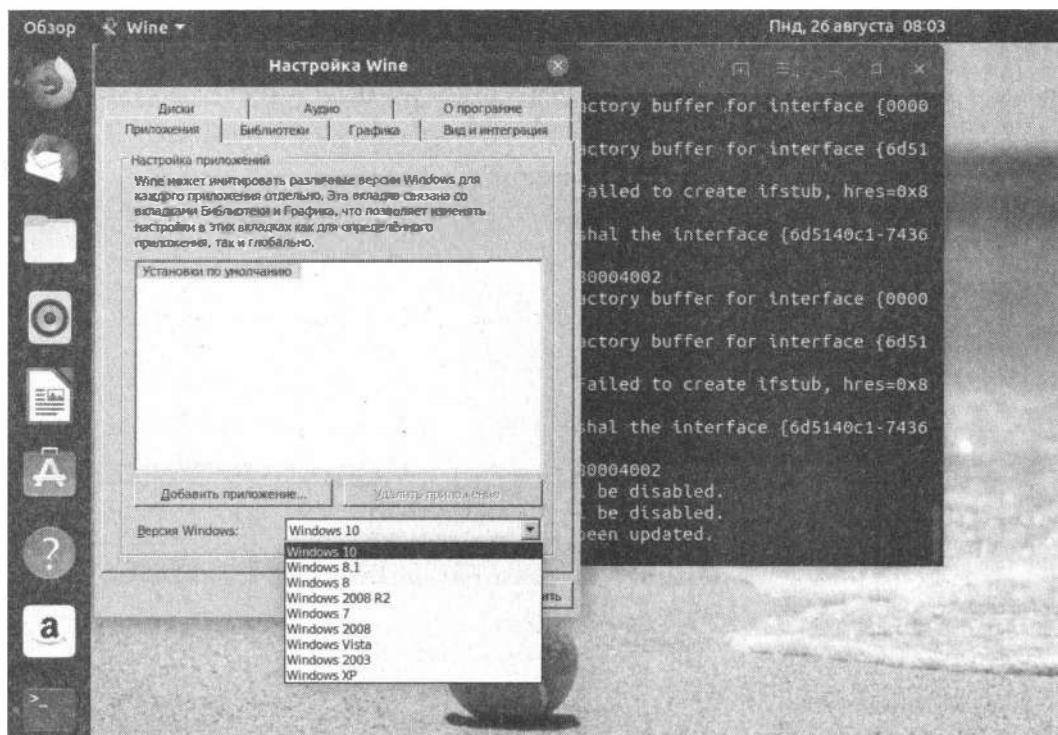


Рис. 19.3. Ubuntu: выбор версии Windows в Wine

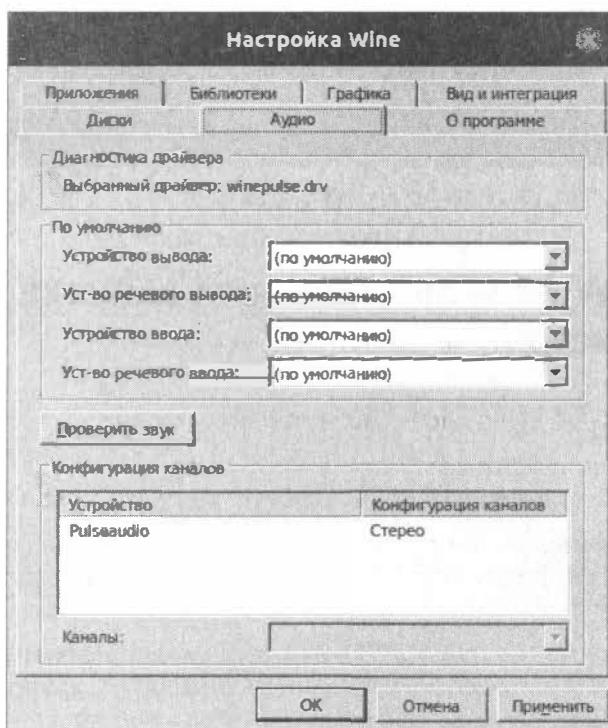


Рис. 19.4. Ubuntu: проверка звука в Wine

## 19.4. Использование Wine

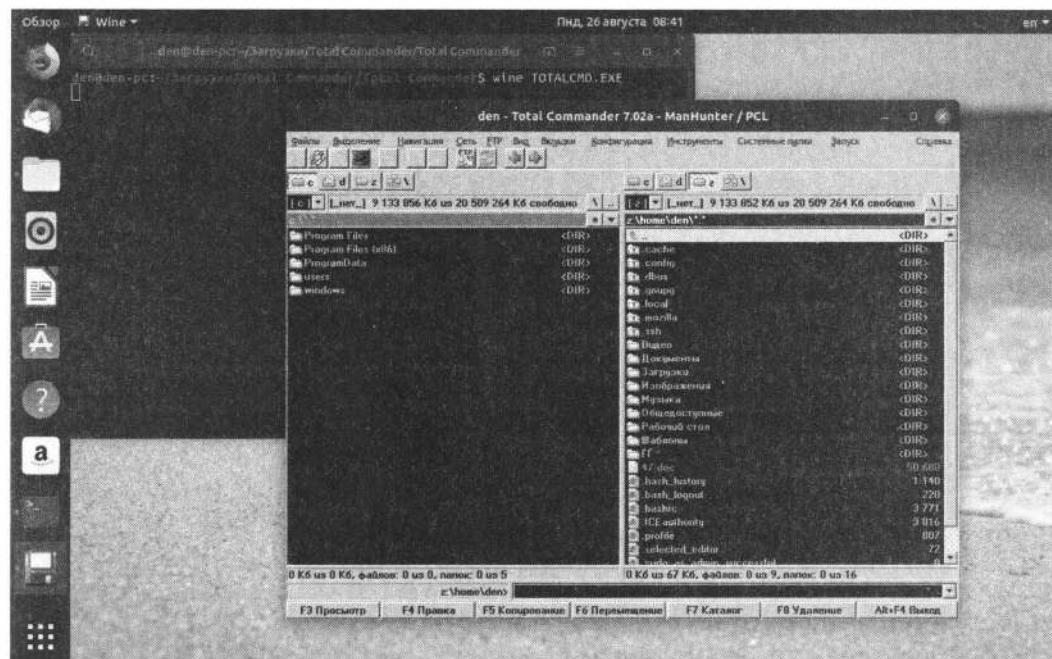
В предыдущих версиях Ubuntu можно было связать Wine с EXE-файлами, сейчас же Wine почему-то не отображается даже в списке доступных приложений. Но нам это и не нужно, поскольку есть терминал и команда wine.

Итак, для запуска Windows-приложения нужно ввести команду:

```
wine <путь к приложению/приложение.exe>
```

Например, для запуска Total Commander (рис. 19.5) я сначала перешел в каталог, в который распаковал архив с программой, а затем ввел команду:

```
wine TOTALCMD.EXE
```



**Рис. 19.5.** Ubuntu: приложение Total Commander запущено в Linux

При этом Total Commander — отнюдь не бесполезное приложение в Linux. Оно позволяет просматривать не только виртуальный диск C: (созданный эмулятором), но и файловую систему Linux, которая в приложении обозначается как диск Z:, что и показано на рис. 19.5.

Для упрощения запуска программ можно использовать сценарии оболочки. Например, для запуска Total Commander я создал следующий сценарий:

```
#!/bin/bash
cd ~/Загрузки/Total Commander
wine TOTALCMD.exe
```

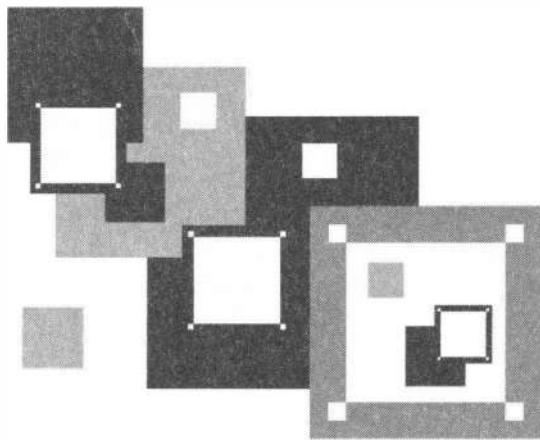
Далее сценарий нужно сделать исполняемым и скопировать в каталог /usr/bin:

```
chmod +x totalcmd  
sudo cp totalcmd /usr/bin
```

После всего этого запустить наш Total Commander можно будет одной командой:

```
totalcmd
```

Точно так же вы можете теперь запускать в Wine и свои любимые игрушки.



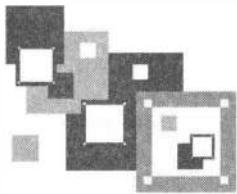
## ЧАСТЬ V

# 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 это 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 появлением на экране этих сообщений. 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 Ubuntu и версия ядра 5.0.0-25-generic (команда dmesg была запущена в Ubuntu 19.04):

```
[ 0.000000] Linux version 5.0.0-25-generic (buildd@lgw01-amd64-008) (gcc
version 8.3.0 (Ubuntu 8.3.0-6ubuntu1)) #26-Ubuntu SMP Thu Aug 1 12:04:58 UTC
2019 (Ubuntu 5.0.0-25.26-generic 5.0.18)
```

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 0x000000000009f800-0x000000000009ffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000000ca000-0x0000000000cbfff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000000dc000-0x0000000000fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x00000000000100000-0x0000000003feffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000003fef0000-0x0000000003fefefff] ACPI data
[ 0.000000] BIOS-e820: [mem 0x0000000003feff000-0x0000000003fefffff] ACPI NVS
[ 0.000000] BIOS-e820: [mem 0x0000000003ff00000-0x0000000003fffffff] usable
[ 0.000000] BIOS-e820: [mem 0x00000000e0000000-0x00000000efffffff] reserved
[ 0.000000] BIOS-e820: [mem 0x00000000fec00000-0x00000000fec0ffff] reserved
[ 0.000000] BIOS-e820: [mem 0x00000000fee00000-0x00000000fee0ffff] reserved
[ 0.000000] BIOS-e820: [mem 0x00000000ffffe0000-0x00000000ffffffff] 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, «тロjanских коней» и других вредоносных программ. Весьма 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 ☺:

```
[ 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-FFFFF write-protect
[ 0.000000] MTRR variable ranges enabled:
[ 0.000000] 0 base 00C0000000 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/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:

```
[ 0.000000] found SMP MP-table at [c00ff780] ff780
```

скажет нам, что найдена таблица SMP (Symmetrical MultiProcessing) MP (MultiProcessing). Это означает, что у нас несколько процессоров или хотя бы несколько ядер.

Далее ядро сообщает объем оперативной памяти (6 Гбайт):

```
[ 0.747465] Memory: 5672044K/5930420K available (14339K kernel code, 2335K rwd
```

data, 4304K rodata, 2576K init, 5204K bss, 258376K reserved, 0K cma-reserved)

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] Command line: BOOT_IMAGE=/boot/vmlinuz-5.0.0-25-generic
root=UUID=3762b167-9de0-4597-b194-3af32fb73ec4 ro quiet splash
```

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 с помощью UUID. Помните, в главе 4 мы говорили о способах адресации разделов? Чтобы идентифицировать раздел, можно указать его короткое имя — вроде `/dev/sda1`, длинное имя или метку. Ubuntu сейчас использует UUID (универсальный идентификатор, или длинное имя).

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 диагностические сообщения, а не графический индикатор загрузки, тогда уберите параметры `quiet splash` — 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`.

Также при загрузке выводится информация и о процессоре. Например, для Intel Core i5 она выглядит так:

```
[ 0.863688] smtboot: CPU0: Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz (family:
0x6, model: 0x8e, stepping: 0x9)
```

Даже пятая версия ядра вычисляет так называемые BogoMIPS:

```
[    0.794548] Calibrating delay loop (skipped) preset value.. 5424.00 BogoMIPS  
(lpj=10848008)
```

### **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 весьма точное определение BogoMIPS: «сколько миллионов раз в секунду процессор может ничего не делать». О производительности процессора по BogoMIPS можно судить лишь косвенно. 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 (главу одного из предыдущих изданий книги) вы найдете в папке *Дополнения* сопровождающего книгу файлового архива (см. *приложение*).

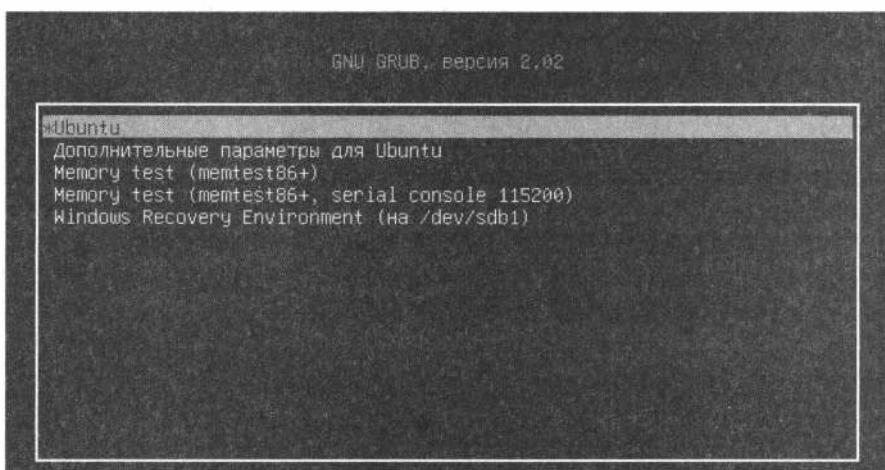
Далее я позволю себе еще существенное сократить представление вывода ядра, поскольку полный вывод занимает целые 34 страницы, и я не думаю, что вам будет интересна каждая его строчка. При желании вы можете ознакомиться с выводом ядра самостоятельно — полезная информация (объем оперативной памяти, информация о процессоре и накопителе) понятна и без моих комментариев, а бесполезная для обычного пользователя — типа информации о распределении памяти — мало кому нужна.

## **20.2. Параметры ядра**

Параметры ядра позволяют управлять его поведением. Как уже говорилось ранее (см. главу 2), 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.

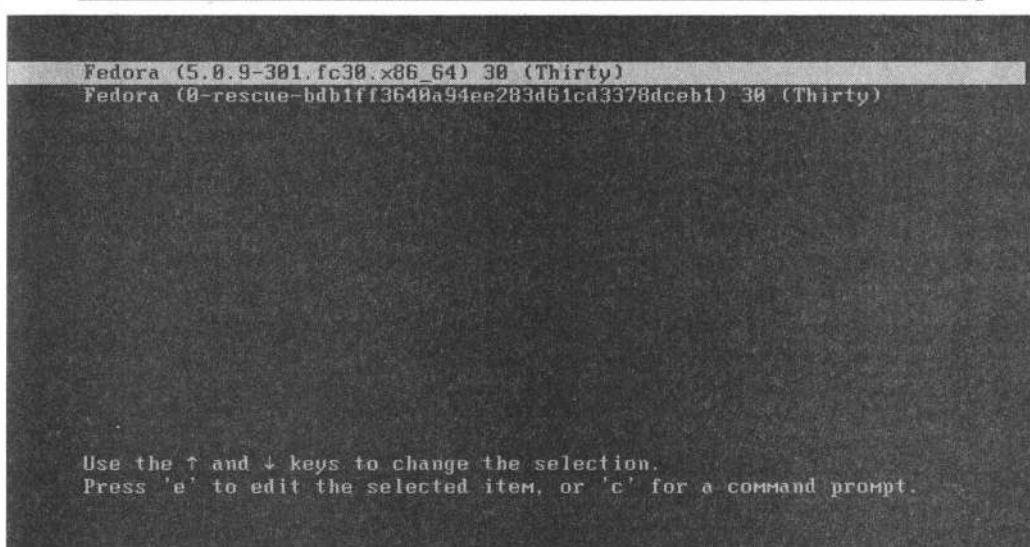
Если вы используете загрузчик GRUB/GRUB2, то передать параметры ядру можно так: сначала надо выбрать образ (метку), а затем нажать клавишу `<e>` — появится поле, в котором вы можете отредактировать параметры ядра, указанные в файлах конфигурации загрузчика.

Внешний вид меню загрузчика GRUB/GRUB2 определяется его конфигурацией: на рис. 20.1, *а* показано, как выглядит меню загрузчика GRUB2 в Ubuntu 19.04, а на рис. 20.1, *б* — в Fedora 30. Как видите, в разных дистрибутивах меню может выглядеть по-разному.



Используйте клавиши ↑ и ↓ для перемещения по пунктам.  
Нажмите «enter» для загрузки выбранной ОС, «e» для  
редактирования команды загрузки или «c» для получения  
командной строки.

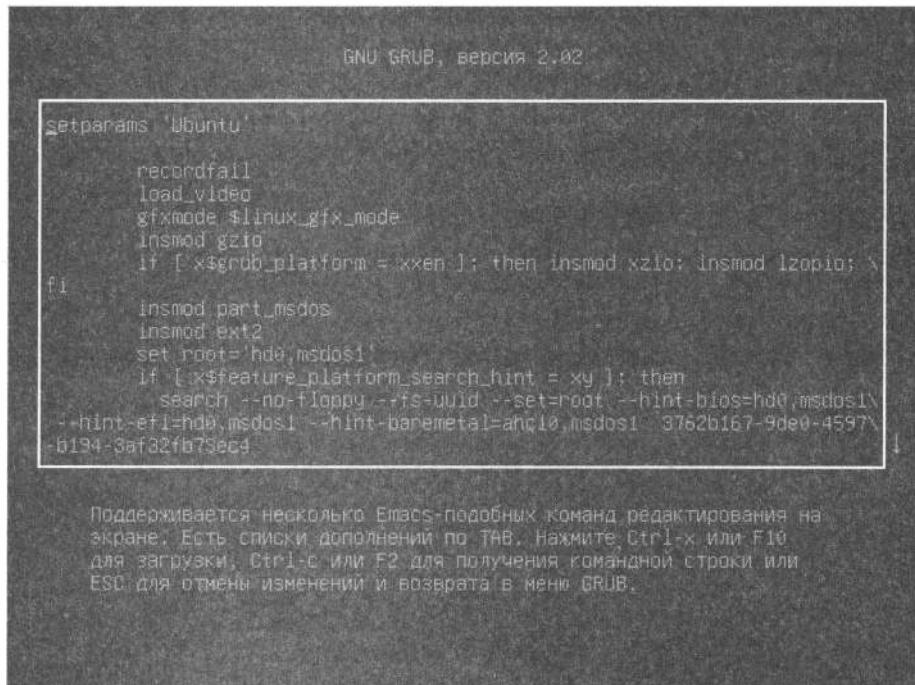
а



б

Рис. 20.1. Меню загрузчика GRUB2: а — в Ubuntu 19.04; б — в Fedora 30

На рис. 20.2, *а* показан процесс редактирования параметров ядра в Ubuntu, а на рис. 20.2, *б* — в Fedora 30. Параметры ядра указываются здесь после служебного слова **linux** — вы можете добавить свои собственные параметры или же отредактировать имеющиеся.



GNU GRUB, версия 2.02

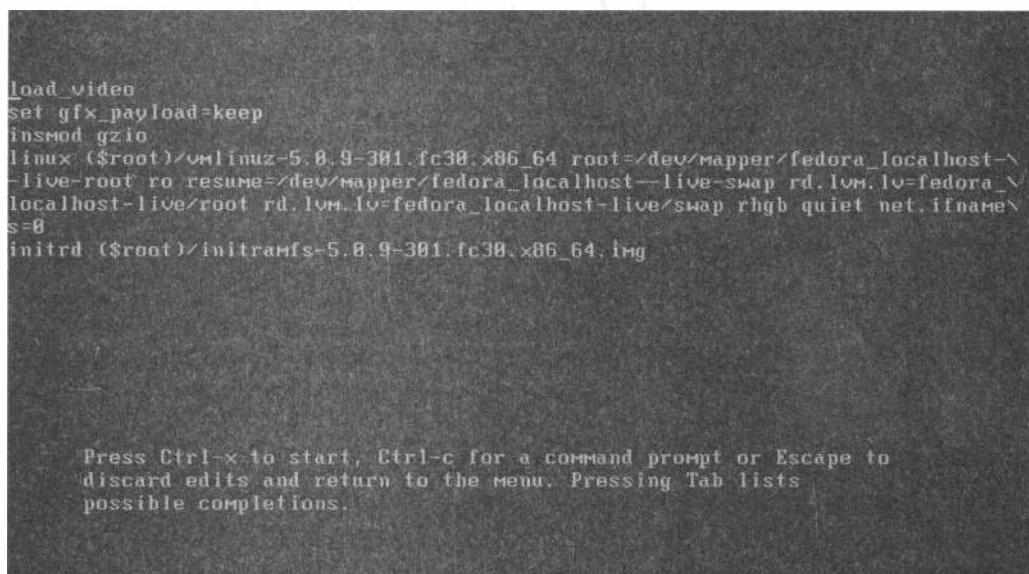
```

setparams 'Ubuntu'
recordfail
load_video
gfxmode $linux_gfx_mode
insmod gzio
if [ $x grub_platform = xxen ]; then insmod xzio; insmod lzopio; fi
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 \
--hint-efi=hd0,msdos1 --hint-baremetal=ahci0,msdos1 3762b167-9de0-4597 \
-8194-3af32fb79ec4
fi

```

Поддерживается несколько Emacs-подобных команд редактирования на экране. Есть списки дополнений по TAB. Нажмите Ctrl-x или F10 для загрузки, Ctrl-c или F2 для получения командной строки или ESC для отмены изменений и возврата в меню GRUB.

а



```

load_video
set gfx_payload=keep
insmod gzio
linux ($root)/vmlinuz-5.8.9-301.fc38.x86_64 root=/dev/mapper/fedora_localhost \
live-root ro resume=/dev/mapper/fedora_localhost-live-swap rd.lvm.lv=fedora \
localhost-live/root rd.lvm.lv=fedora_localhost-live-swap rhgb quiet net.ifname \
s=0
initrd ($root)/initramfs-5.8.9-301.fc38.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.

б

Рис. 20.2. Редактирование параметров ядра: *а* — в Ubuntu 19.04; *б* — в Fedora 30

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                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>root=устройство</code> | Позволяет указать корневую файловую систему.<br>Например: <code>root=/dev/sda5</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <code>rootdelay=N</code>     | Ждать N секунд перед монтированием корневой файловой системы                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <code>rootflags=флаги</code> | Задает флаги корневой файловой системы (см. главу 4)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <code>rootfstype=тип</code>  | Задает тип корневой файловой системы. Полезен, если не удалось определить автоматически                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <code>rootwait</code>        | Ожидание корневой файловой системы. Ядро будет ждать, пока не появится устройство с корневой файловой системой. Полезно, когда корневая файловая система расположена на съемном носителе — например, на флешке                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <code>ro</code>              | Монтирует корневую файловую систему в режиме «только чтение». Используется по умолчанию. После проверки файловой системы программой <code>fsck</code> корневая файловая система перемонтируется в режим <code>rw</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <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>            | 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:<br><br><code>mem=768M</code><br><br>После числа обязательно должна следовать буква <code>M</code> , иначе ядро «подумает», что объем оперативной памяти 768 байтов.<br>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-х гигабайт, а потом обнаружить, что ваш дистрибутив видит только первые 4 гигабайта. В этом случае вам нужно перекомпилировать ядро с поддержкой PAE (Physical Address Extension) или же установить ядро, изначально поддерживающее PAE |
| <code>init=</code>           | Позволяет задать программу инициализации. По умолчанию используется программа <code>/sbin/init</code> , но вы можете задать другую                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |

Таблица 20.1 (окончание)

| Параметр             | Описание                                                                                                                                                                                                                                                                                                                                        |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| reboot=              | Позволяет задать тип перезагрузки компьютера. Возможные значения: <code>cold</code> и <code>warm</code> , т. е. «холодная» или «горячая» перезагрузка                                                                                                                                                                                           |
| single               | Однопользовательский режим для администрирования системы — например, в случае отказа                                                                                                                                                                                                                                                            |
| nodmraid             | Отключает программные RAID-массивы, организованные на уровне BIOS                                                                                                                                                                                                                                                                               |
| noapic               | Полезен, если вы при загрузке увидите сообщение:<br><code>kernel panic - not syncing: IO-APIC + timer doesn't work!</code><br>Подробнее об этом параметре вы можете прочитать по адресу:<br><a href="http://www.dkws.org.ua/phpbb2/viewtopic.php?topic=2973&amp;forum=5">http://www.dkws.org.ua/phpbb2/viewtopic.php?topic=2973&amp;forum=5</a> |
| portcmcia            | Отключает PCMCIA-карты (для ноутбуков). Полезен, если вы подозреваете, что у вас проблемы с PCMCIA-картой                                                                                                                                                                                                                                       |
| nodma                | Отключается DMA (Direct Memory Access, прямой доступ к памяти) для всех IDE-устройств                                                                                                                                                                                                                                                           |
| noapm                | Отключает APM (Advanced Power Management) — расширенное управление питанием                                                                                                                                                                                                                                                                     |
| nousb                | Отключает поддержку USB                                                                                                                                                                                                                                                                                                                         |
| noscsi               | Отключает поддержку SCSI                                                                                                                                                                                                                                                                                                                        |
| pci=noacpi           | Не использовать ACPI для управления PCI-прерываниями                                                                                                                                                                                                                                                                                            |
| apci=off             | Полностью отключает ACPI (Advanced Configuration and Power Interface). Полезен на некоторых ноутбуках, когда не удается установить (а потом загрузить) Linux                                                                                                                                                                                    |
| edd=off              | Отключает EDD (Enhanced Disk Drive). Если при загрузке Linux вы видите сообщение <code>Probing EDD</code> , 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 главе 30                                                                                                                                                                                                                                                              |
| vga=режим            | Позволяет задать VGA-режим. Подробнее см. файл Documentation/svga.txt. Можно также задать значение <code>ask</code> , чтобы ядро спросило, какой режим нужно использовать:<br><code>vga=ask</code>                                                                                                                                              |
| quiet                | «Тихий» режим, отключает большинство сообщений ядра                                                                                                                                                                                                                                                                                             |
| splash               | При загрузке системы отображается графическая заставка                                                                                                                                                                                                                                                                                          |

### ДОПОЛНИТЕЛЬНЫЕ ПАРАМЕТРЫ ЯДРА

Повторюсь: 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 достаточно 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](http://www.kernel.org).

Далее в этом разделе будет рассмотрена сборка ядра на примере дистрибутива Ubuntu 19.04. Но прежде чем продолжить, хочу отметить, что для компиляции ядра вам понадобится весьма много дискового пространства. Во-первых, нужно установить дополнительное программное обеспечение. 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 install git build-essential ncurses-base ncurses-dev fakeroot  
kernel-package xz-utils libncurses-dev flex bison libssl-dev
```

### 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/v5.x/linux-5.2.9.tar.gz
```

На момент подготовки книги версия ядра 5.2.14 была последней, хотя, возможно, уже появилась и новая версия. Просто откройте браузер и просмотрите содержимое каталога <https://cdn.kernel.org/pub/linux/kernel/v5.x/>. Впрочем, если вы сделаете, как здесь предлагается, то все равно установите более новую версию ядра, чем 5.0.0, которую предлагает Ubuntu 19.04 по умолчанию.

После загрузки архива нужно его распаковать:

```
$ tar xvf linux-5.2.9.tar.xz  
$ cd linux-5.2.9
```

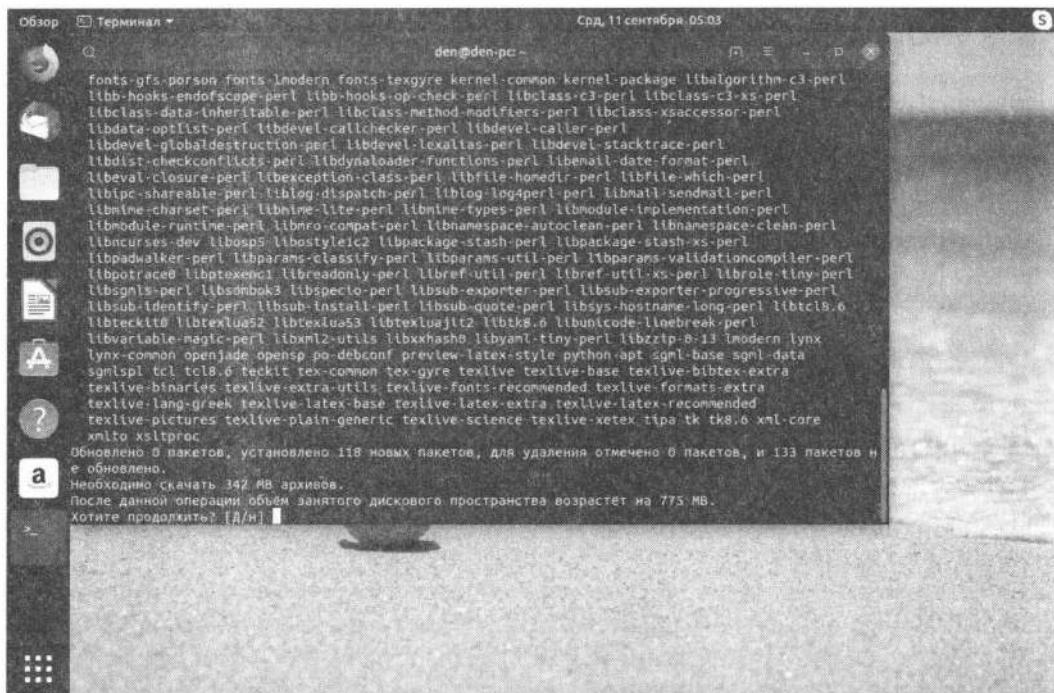


Рис. 20.3. Ubuntu 19.04: установка дополнительного программного обеспечения

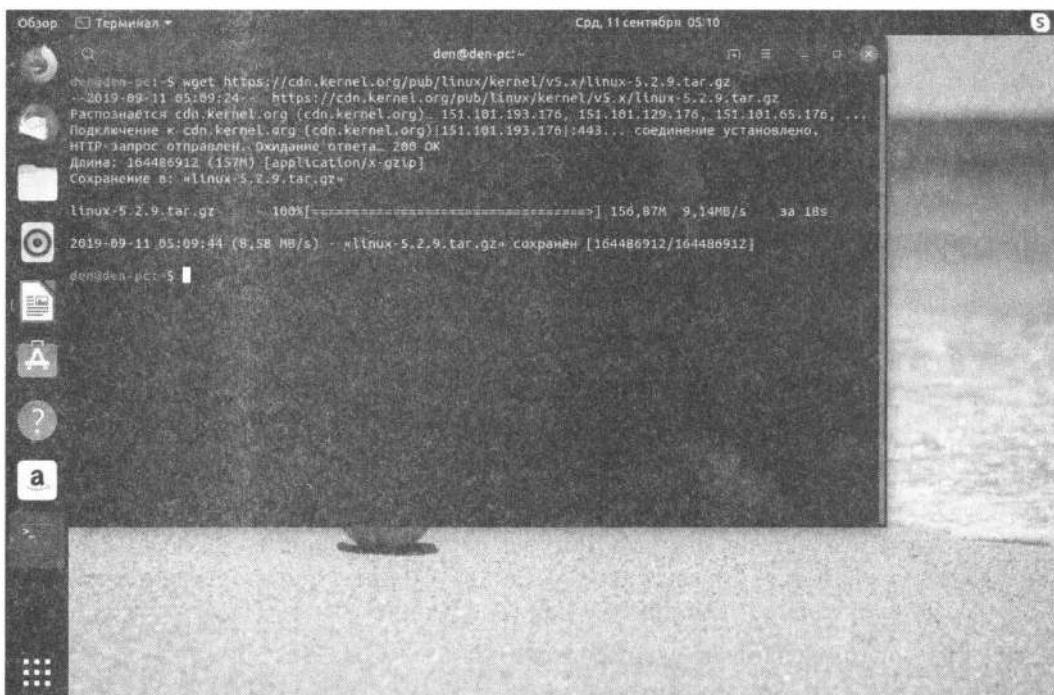


Рис. 20.4. Ubuntu 19.04: загрузка исходников ядра

### 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-5.2.9`, 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 конфигуратора, но, на мой взгляд, `make menuconfig` — наиболее удобный (рис. 20.5).

Меню конфигуратора 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 конфигуратора ядра представлены в табл. 20.2.

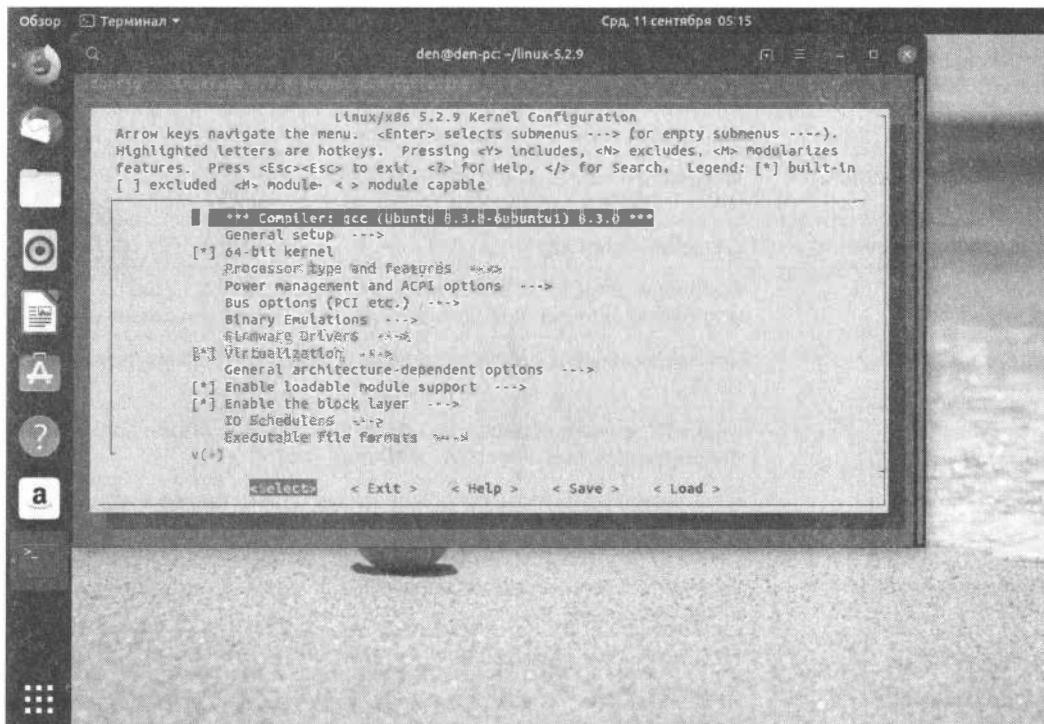


Рис. 20.5. Ubuntu 19.04: конфигуратор `make menuconfig`

**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 swap-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.<br><br>Если же вы хотите построить малообслуживаемый 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                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| 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 исполняемых 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 — M. Это означает, что опция будет включена в ядро как *модуль*. После сборки ядра и модулей все опции, скомпилированные в режиме M, 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 конфигуратора он спросит вас, хотите ли вы сохранить изменения в конфигурации ядра (рис. 20.6)? Конечно, хотим!

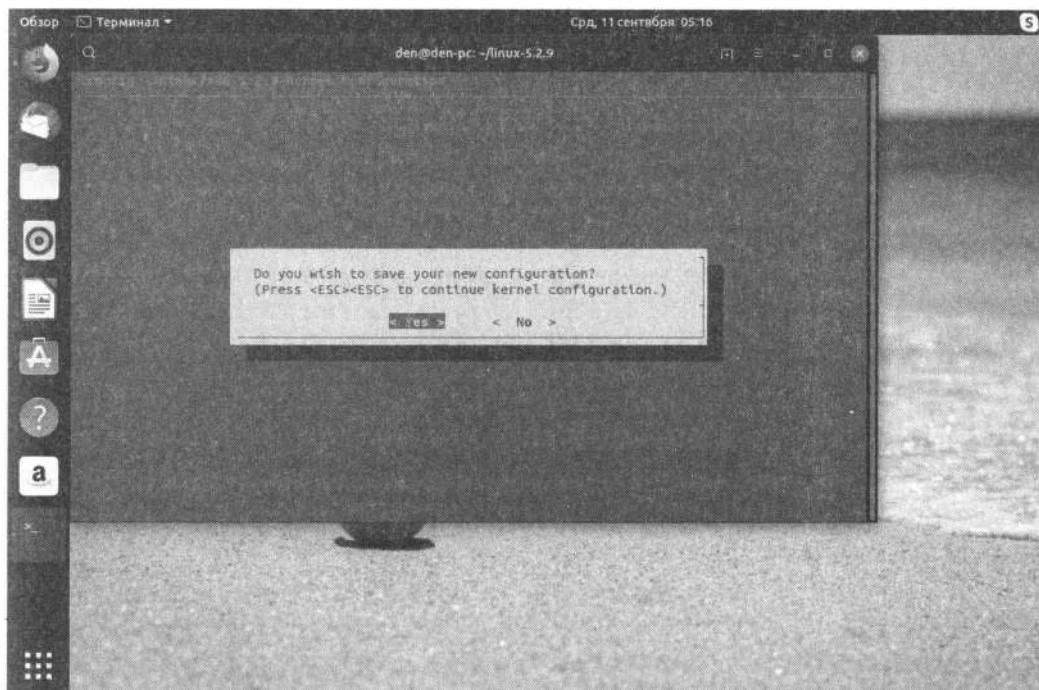


Рис. 20.6. Ubuntu 19.04: сохранить изменения в конфигурации ядра?

#### 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 (рис. 20.8) и сбросить параметры `kernel-package`, а затем собрать ядро, используя команду `fakeroot`, что позволит откомпилировать ядро от имени обычного пользователя, а не `root` (рис. 20.9):

```
den@den-pc:~/linux-5.2.9
```

Настраивается пакет bison (2:3.3.2.dfsg-1)  
update-alternatives: используется /usr/bin/bison.yacc для предоставления /usr/bin/yacc ( yacc ) в автоматическом режиме  
Настраивается пакет libfl-dev:amd64 (2:6.4-0~Z)  
Обрабатывается триггер для libc-bin (2:29~Ubuntu2)  
Обрабатываются триггеры для man-db (2:8.5-2)  
Обрабатываются триггеры для install-info (6.5.0.dfsg.1-4build1)  
den@den-pc:~/linux-5.2.9\$ make menuconfig  
LEX scripts/kconfig/lexer.lex.c  
YACC scripts/kconfig/parser.tab.h  
HOSTCC scripts/kconfig/lexer.lex.o  
YACC scripts/kconfig/parser.tab.c  
HOSTCC scripts/kconfig/parser.tab.o  
HOSTCC scripts/kconfig/preprocess.o  
HOSTCC scripts/kconfig/symbol.o  
HOSTLD scripts/kconfig/nconfig  
scripts/kconfig/mconfig Kconfig  
.config:118:warning: symbol value `m' invalid for NF\_CT\_PROTO\_GRE  
.config:192:warning: symbol value `m' invalid for NET\_DEVLINK  
.config:785:warning: symbol value `m' invalid for ASHMEM  
.config:8724:warning: symbol value `m' invalid for ANDROID\_BINDER\_IPC  
.config:8725:warning: symbol value `m' invalid for ANDROID\_BINDERSFS  
configuration written to .config  
\*\*\* End of the configuration.  
\*\*\* Execute 'make' to start the build or try 'make help'.  
den@den-pc:~/linux-5.2.9\$

Рис. 20.7. Ubuntu 19.04: конфигурация сохранена

```
den@den-pc:~/linux-5.2.9
```

scripts/kconfig/mconfig Kconfig  
.config:118:warning: symbol value `m' invalid for NF\_CT\_PROTO\_GRE  
.config:192:warning: symbol value `m' invalid for NET\_DEVLINK  
.config:785:warning: symbol value `m' invalid for ASHMEM  
.config:8724:warning: symbol value `m' invalid for ANDROID\_BINDER\_IPC  
.config:8725:warning: symbol value `m' invalid for ANDROID\_BINDERSFS  
configuration written to .config  
\*\*\* End of the configuration.  
\*\*\* Execute 'make' to start the build or try 'make help'.  
den@den-pc:~/linux-5.2.9\$ make-kpkg clean  
exec make kpkg\_version=13.018+nnu1 -f /usr/share/kernel-package/ruleset/minimal.mk clean  
===== making target minimal\_clean [new prereqs: ] =====  
This is kernel package version 13.018+nnu1.  
test ! -f .config || cp -pf .config config.previous  
test ! -e stamp-building || rm -f stamp-building  
test ! -f Makefile ||  
 make ARCH=x86\_64 distclean  
make[1]: выход из каталога «/home/den/linux-5.2.9»  
 CLEAN scripts/basic  
 CLEAN scripts/kconfig  
 CLEAN include/config include/generated  
 CLEAN .config .config.old  
make[1]: выход из каталога «/home/den/linux-5.2.9»  
test ! -f config.previous || mv -f config.previous .config  
rm -f modules/nodversions.h modules/ksyms.ver scripts/cramfs scripts/cramfs/nkcramps  
den@den-pc:~/linux-5.2.9\$

Рис. 20.8. Ubuntu 19.04: команда make-kpkg clean в действии



Рис. 20.9. Ubuntu 19.04: компиляция ядра

```
$ make-kpkg clean
$ fakeroot make-kpkg --initrd --revision=1.0.Bagira 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 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.

тель, 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 (Intel Core i5) машине с 6-ю гигабайтами оперативной памяти компиляция ядра 5.2.9 заняла около четырех часов. Если машина слабее, то процедура эта, соответственно, займет больше времени. Так что, запасайтесь терпением. В любом случае, у вас есть как минимум пару часов 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 весьма скромными по размеру (тут все зависит от выбранных при настройке ядра опций): пакет с ядром (`linux-image`) — 55 Мбайт, а пакет с заголовками (`linux-headers`) — 10 Мбайт.

Сначала нужно установить пакет `linux-headers`, а затем — `linux-image` (рис. 20.10):

```
$ cd ~
$ sudo dpkg -i linux-headers-5.2.9_1.0.Bagira_amd64.deb
$ sudo dpkg -i linux-image-5.2.9_1.0.Bagira.amd64.deb
```

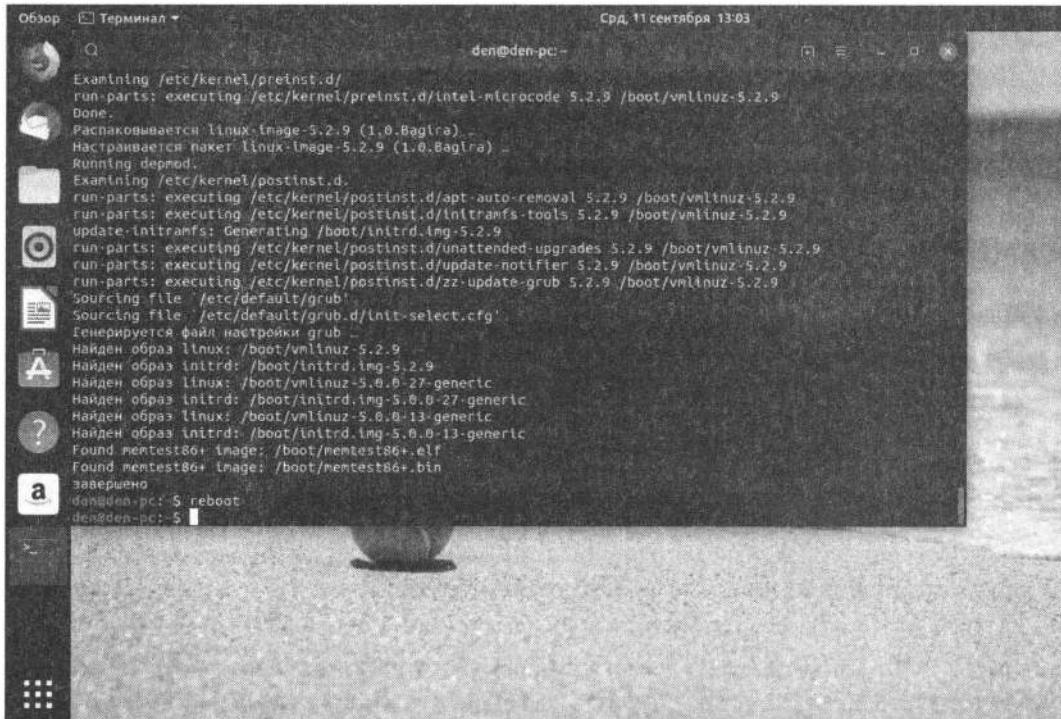


Рис. 20.10. Ubuntu 19.04: установка пакета с ядром

После чего перезагрузить компьютер:

```
$ reboot
```

Чтобы убедиться, что загрузилось именно скомпилированное ядро, введите команду:

```
$ uname -a
```

В выводе команды должна присутствовать примерно такая строка (рис. 20.11):

```
Linux den-pc 5.2.9 #1 SMP Wed Sep 11 07:40:44 UTC 2019 x86_64 x86_64 x86_64  
GNU/Linux
```

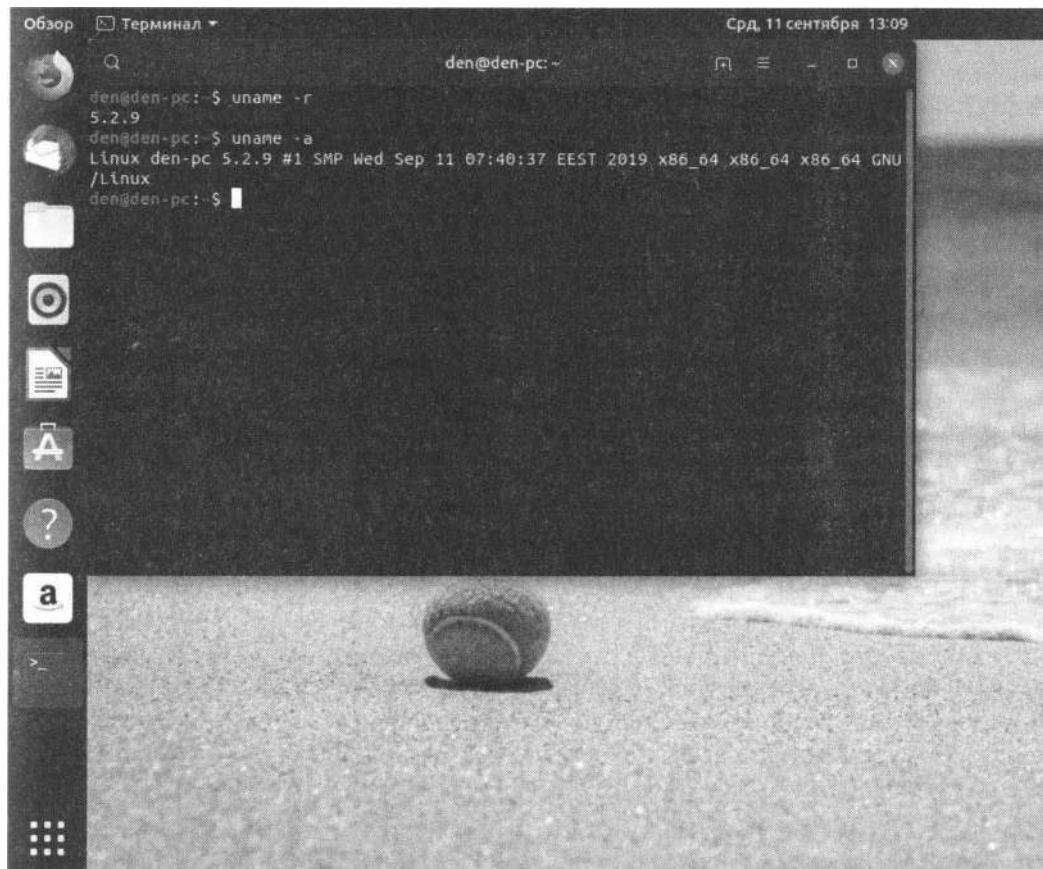


Рис. 20.11. Ubuntu 19.04: вывод команды `uname`

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 «весыма много». На самом деле, если до компиляции ядра объем занятого на диске пространства составлял 7,2 Гбайт, то после компиляции эта величина возросла до 28 Гбайт, — получается, что 20,8 Гбайт понадобилось на компиляцию ядра, и из них только 0,9 Гбайт — на дополнительные пакеты.

Соответственно, файловая система `/home` должна содержать примерно 19,9 Гбайт доступного пространства:

28 Гбайт (после компиляции) – 7,2 Гбайт (до компиляции) – 0,9 Гбайт (пакеты, установленные в корневой каталог) = (примерно) 19,9 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-5.2.9` 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 9,3 Гбайт, но никак не 28.

## 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](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 `kernel-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:

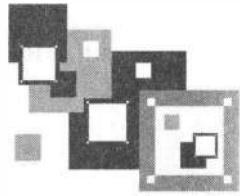
```
# genkernel initrd
```

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: [https://fedoraproject.org/wiki/Building\\_a\\_custom\\_kernel/ru](https://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://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 материал (раздел главы 21 из 2-го издания книги) вы найдете в папке *Дополнения* сопровождающего книгу файлового 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 2019 году можно установить обычный GRUB, если очень хочется, — он присутствует в составе популярных дистрибутивов (необхо-

димый для его загрузки пакет называется `grub-legacy`), хотя по умолчанию в них используется GRUB2.

## 21.2. Конфигурационные файлы GRUB и GRUB2

### 21.2.1. Конфигурационный файл GRUB

Конфигурационным файлом GRUB в старых версиях дистрибутивов являлся файл `/boot/grub/menu.lst`, сейчас же — файл `/boot/grub/grub.conf` (кстати, имеющийся в новых версиях файл `menu.lst` — это просто 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  
fallback=1  
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). Тем не менее, 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*`.

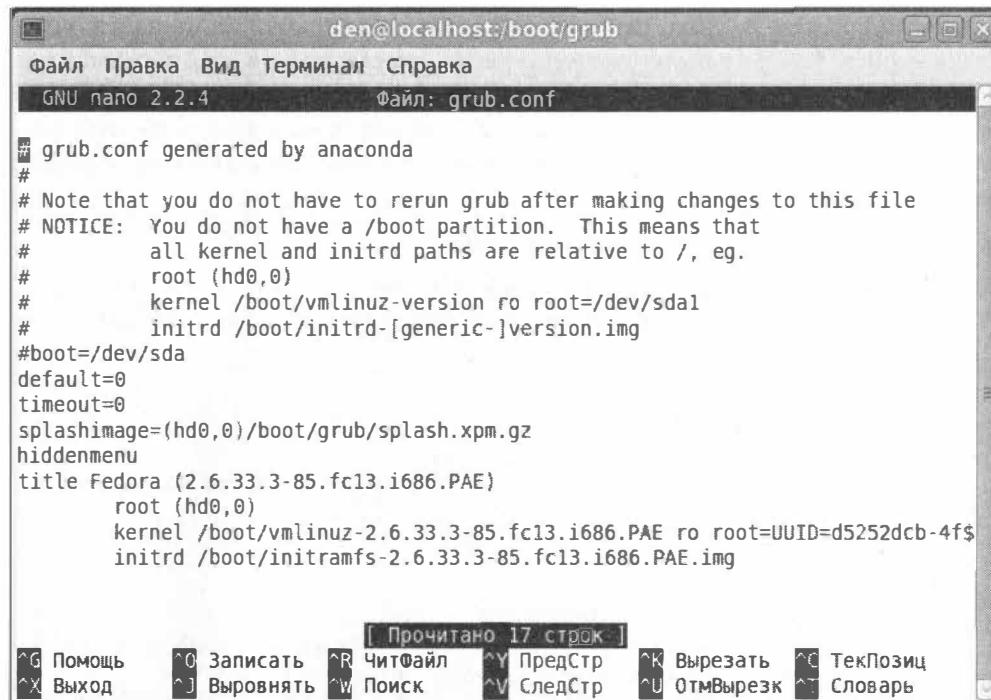
При работе с GRUB вам поначалу будет трудно разобраться с именами разделов диска. GRUB вместо привычных `/dev/hd*` (или `/dev/sd*` — для SCSI-дисков) использует свои собственные имена. Перевести имя `/dev/hd*` в имя в формате GRUB просто. Во-первых, опускается `/dev/`. Во-вторых, устройства отчитываются не с буквы «`a`», как в Linux, а с нуля. Разделы на дисках отчитываются не с единицы, а тоже с нуля, причем номер раздела указывается через запятую. Потом все имя берется в скобки. Соответственно, раздел `/dev/hda1` в GRUB будет выглядеть как `(hd0,0)`, а раздел `/dev/hdb2` — как `(hd1,1)`. 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 `splashimage`. 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`.



```

den@localhost:/boot/grub
Файл Правка Вид Терминал Справка
GNU nano 2.2.4          Файл: grub.conf

# grub.conf generated by anaconda
#
# Note that you do not have to rerun grub after making changes to this file
# NOTICE: You do not have a /boot partition. This means that
#          all kernel and initrd paths are relative to /, eg.
#          root (hd0,0)
#          kernel /boot/vmlinuz-version ro root=/dev/sda1
#          initrd /boot/initrd-[generic-]version.img
#boot=/dev/sda
default=0
timeout=0
splashimage=(hd0,0)/boot/grub/splash.xpm.gz
hiddenmenu
title Fedora (2.6.33.3-85.fc13.i686.PAE)
    root (hd0,0)
    kernel /boot/vmlinuz-2.6.33.3-85.fc13.i686.PAE ro root=UUID=d5252dcb-4f$initrd /boot/initramfs-2.6.33.3-85.fc13.i686.PAE.img

[ Прочитано 17 строк ]
^G Помощь   ^O Записать   ^R Читать файл   ^Y ПредСтр   ^K Вырезать   ^C ТекПозиц
^X Выход   ^P ВыровняТЬ   ^W Поиск   ^V СледСтр   ^U ОтмВырезк   ^T Словарь

```

Рис. 21.1. Fedora: редактирование файла grub.conf

На рис. 21.1 представлен 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 весьма 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 этот 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/grub2-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-0cfcdxfc-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 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 --hint-
efi=hd0,msdos1 --hint-baremetal=ahci0,msdos1 0cfcdxfc-d3e4-4755-
a0ea-5d44470dee4f
    else
        search --no-floppy --fs-uuid --set=root 0cfcdxfc-d3e4-4755-a0ea-
5d44470dee4f
    fi
    linux /boot/vmlinuz-3.19.0-15-generic root=UUID=0cfcdxfc-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.

В более новых дистрибутивах название элемента меню не задается статически, а генерируется сценарием. Поэтому вы можете увидеть такую строку:

```
echo "menuentry '$(echo \"$os\" | grub_quote)' ..."
```

Здесь видно, что для формирования заголовка записи меню используется переменная `$os`, которая формируется ранее в сценарии.

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_DISTRO='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

# 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/grub2-mkconfig` и передающий ей параметр — имя выходного файла (по умолчанию `/boot/grub/grub.cfg`). Ничто не мешает вам вызвать утилиту `grub-mkconfig/grub2-mkconfig`<sup>1</sup> вручную, но использовать команду `update-grub` более удобно.

#### **ВНИМАНИЕ!**

В современных дистрибутивах команда `update-grub` отсутствует. Вместо нее нужно использовать утилиту `grub2-mkconfig`.

#### **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 чём 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 утилиту `grub2-mkconfig` или команду `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` утилой `grub2-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 утилиты GRUB2-MKCONFIG**

По умолчанию утилита `grub2-mkconfig` генерирует конфигурационный файл на консоль, поэтому вызывать ее нужно так:

```
sudo grub2-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
```

---

<sup>1</sup> Сначала утилита `grub2-mkconfig` называлась `grub-mkconfig`, но, видимо, чтобы подчеркнуть ее принадлежность именно к GRUB2, она была переименована.

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 640×480 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.xpm.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 grub2-mkconfig > /boot/grub/grub.cfg
```

## 21.5. Постоянныe имена устройств

Как было отмечено ранее (см. главу 4), 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/vmlinuz509 root=/dev/sdal vga=0x318 ro
```

Эта запись указывает имя ядра: `/boot/vmlinuz509`. 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 (в зависимости от версии GRUB):

```
/sbin/grub-install /dev/sda  
/sbin/grub2-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

Ведите 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):

`Password: secret_password`

закодирует его и выведет на экран шифр введенного вами пароля — например (рис. 21.2):

`Encrypted: $1$71wCI$XYwZ5oYZ.NEYn3GUwtqaQ1`

Итак, вы получили зашифрованный пароль. Перепишите этот шифр (а еще лучше выделите его, выполните команду меню **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`

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`

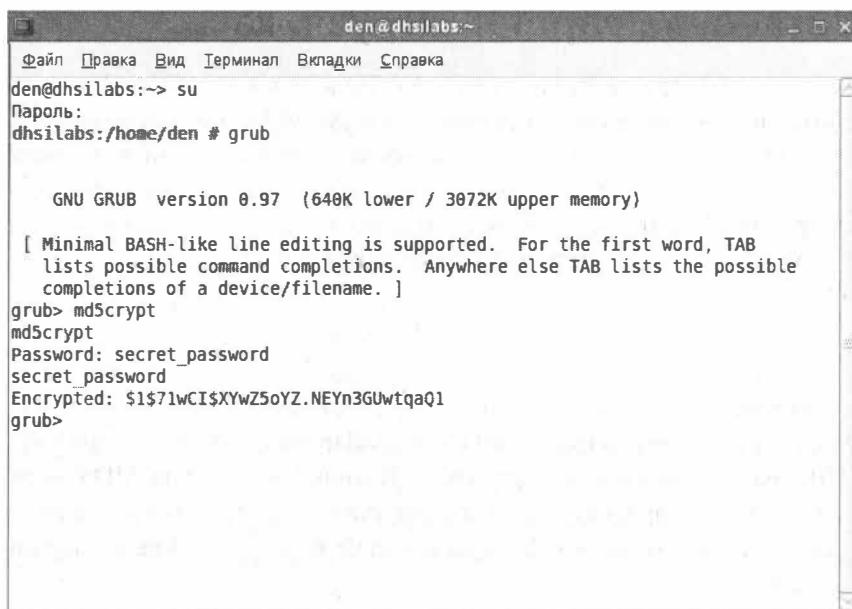


Рис. 21.2. openSUSE: шифрование пароля GRUB

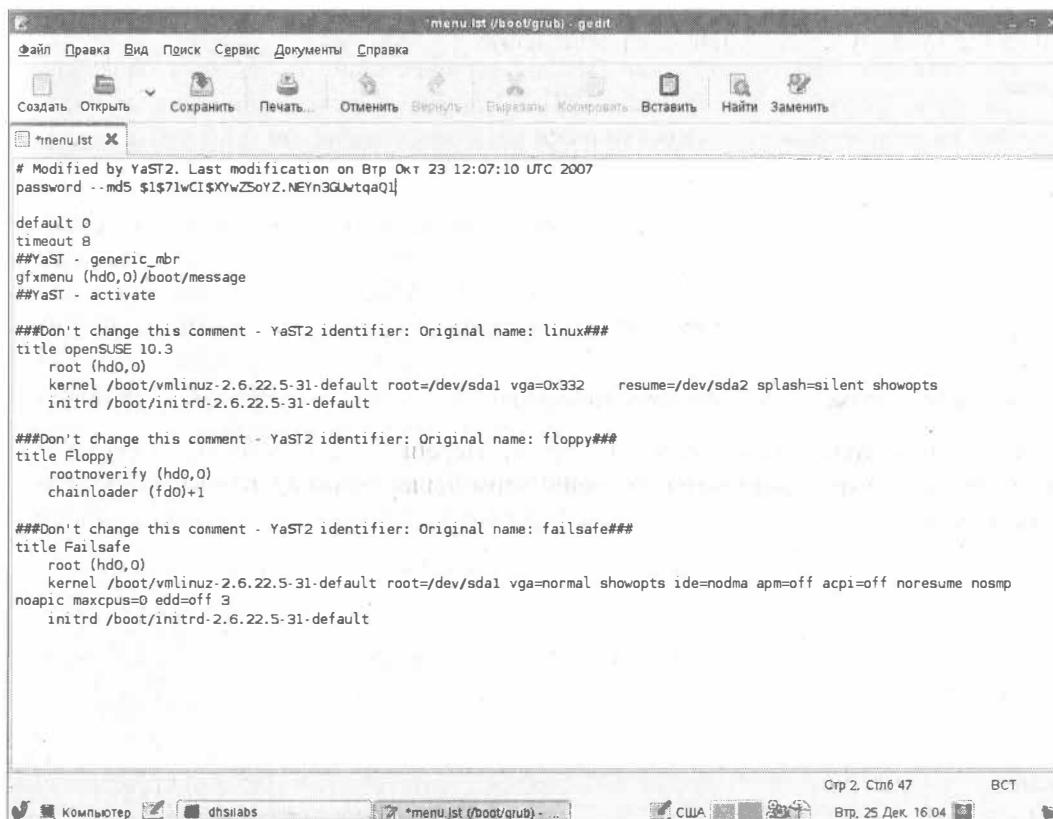


Рис. 21.3. openSUSE: редактирование файла /boot/grub/menu.lst

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 (в примере на рис. 21.2 — это: *\$1\$71wCI\$XYwZ5oYZ.NEYn3GUwtqaQ1*).

Таким образом мы задали пароль, с помощью которого можно редактировать загрузочное меню GRUB. И пока не будет указан заданный пароль, GRUB не разрешит его редактировать.

## 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 grub2-mkconfig > /boot/grub/grub.cfg
```

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 лишь загрузить Linux с параметрами по умолчанию.

Можно даже задать условие, что метку Windows будет загружать только пользователь `user2`:

```
menuentry "Windows" --users user2 {  
    set root=(sd0,2)  
    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
```

Вот пример такого шифра:

```
grub.pbkdf2.sha512.10000.9290F727ED06C38BA4549EF7DE25CF5642659211B7FC076F2D28FE  
FD71784BB8D8F6FB244A8CC5C06240631B97008565A120764C0EE9C2CB0073994D79080136.887C  
FF169EA8335235D8004242AA7D6187A41E3187DF0CE14E256D85ED97A97357AAA8FF0A3871AB9EE  
FF458392F462F495487387F685B7472FC6C29E293F0A0
```

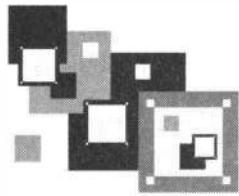
Весь этот хэш нужно скопировать в конфигурационный файл GRUB2:

```
password_pbkdf2 den  
grub.pbkdf2.sha512.10000.9290F727ED06C38BA4549EF7DE25CF5642659211B7FC076F2D28FE  
FD71784BB8D8F6FB244A8CC5C06240631B97008565A120764C0EE9C2CB0073994D79080136.887C  
FF169EA8335235D8004242AA7D6187A41E3187DF0CE14E256D85ED97A97357AAA8FF0A3871AB9EE  
FF458392F462F495487387F685B7472FC6C29E293F0A0
```

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: `stage1` 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 — `*stage1_5`, — помогающих загрузчику найти `stage2` и выполняя других 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 такая 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: <https://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 в Google Play), 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`<sup>1</sup> — 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
```

<sup>1</sup> Впервые `systemd` появилась в `Debian 8`.

```
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:12345:powerokwait:/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 уровнем по умолчанию он 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, S10network 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: K01service. 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:

```
# rcaapache start
```

### 22.2.3. Редакторы уровней запуска

Редактировать уровни запуска можно вручную, а можно и с помощью программ-конфигураторов<sup>1</sup>:

- 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 версией 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 весьма 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 — весьма нерасторопный и запускается медленно, а сервис Г зависит от сервиса В. Получается, что сервис Г не сможет быть корректно запущен, покуда 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, парал-

---

<sup>1</sup> Следует понимать, что речь идет о старых версиях дистрибутивах. В новых эти конфигураторы недоступны, поскольку в них используется система инициализации systemd, а не init.

лько 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 настоящий 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 веб- или FTP-серверу, нет необходимости их запускать, что сэкономит не только лишние секунды при запуске системы, но и снизит нагрузку на процессор во время работы системы. Да и память сбережет.

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 запус-

титься быстрее базового сервиса A. 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 macOS — там она называется launchd. Но поскольку не все знакомы с macOS, 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 `nscd.service` управляет сервисом (демоном) `nscd`.

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 новая система инициализации. А вот в Fedora 15<sup>1</sup> 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 проснеться.

<sup>1</sup> openSUSE 12.1 и Fedora 15 — первые версии дистрибутивов openSUSE и Fedora с `systemd`.

### 22.3.3. Основные особенности systemd

Приведем 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;
- синтаксис файлов конфигурации systemd очень похож на синтаксис файлов .desktop и поэтому будет знаком многим Linux-пользователям;
- 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 новая<sup>1</sup>;
- 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 или 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://0pointer.de/blog/projects/why.html>. Информации там весьма много, поэтому самое основное я вычленил и представил в табл. 22.1.

Таблица 22.1. Сравнение систем инициализации

| Возможность                                     | Init | Upstart | Systemd |
|-------------------------------------------------|------|---------|---------|
| <b>Сервисы</b>                                  |      |         |         |
| Совместимость с SysV                            | +    | +       | +       |
| Управление SysV-сервисами как родными сервисами | +    | -       | +       |
| Управление сервисами с помощью /dev/initctl     | +    | -       | +       |
| Контролируемый останов сервисов                 | -    | -       | +       |

<sup>1</sup> Даже в самых новых версиях дистрибутивов — например, в той же Ubuntu 20.10, 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 |
|-------------------------------------------------------------------------|------|---------|---------|
| Перезапуск сервисов с сериализацией состояния                           | +    | -       | +       |
| Отключение сервисов без редактирования файлов                           | +    | -       | +       |
| Отправка сигналов сервисам                                              | -    | -       | +       |
| Перезапуск сервисов при их крахе без потери соединения                  | -    | -       | +       |
| Поддержка сервисов типа «instantiated»                                  | -    | +       | +       |
| Показывает все процессы, принадлежащие сервису                          | -    | -       | +       |
| Идентификация процессов сервиса                                         | -    | -       | +       |
| Активация сервисов на основе сокетов                                    | -    | -       | +       |
| <b>Система</b>                                                          |      |         |         |
| 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 автомонтирование         | -    | -       | +       |
| 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                   | -    | -       | +       |
| <b>· Ядро и протоколирование</b>                                        |      |         |         |
| 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                       | -    | -       | +       |

Таблица 22.1 (окончание)

| Возможность                                                                             | Init | Upstart | Systemd |
|-----------------------------------------------------------------------------------------|------|---------|---------|
| <b>Безопасность</b>                                                                     |      |         |         |
| Интеграция с Linux Control Groups (cgroups)                                             | -    | -       | +       |
| Генерация событий аудита для запускаемых сервисов                                       | -    | -       | +       |
| Поддержка PAM                                                                           | -    | -       | +       |
| 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                                             | -    | -       | +       |
| <b>Сеть</b>                                                                             |      |         |         |
| 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 33 (самой последней версии дистрибутива, появившейся на свет 27 октября 2020 года). 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.

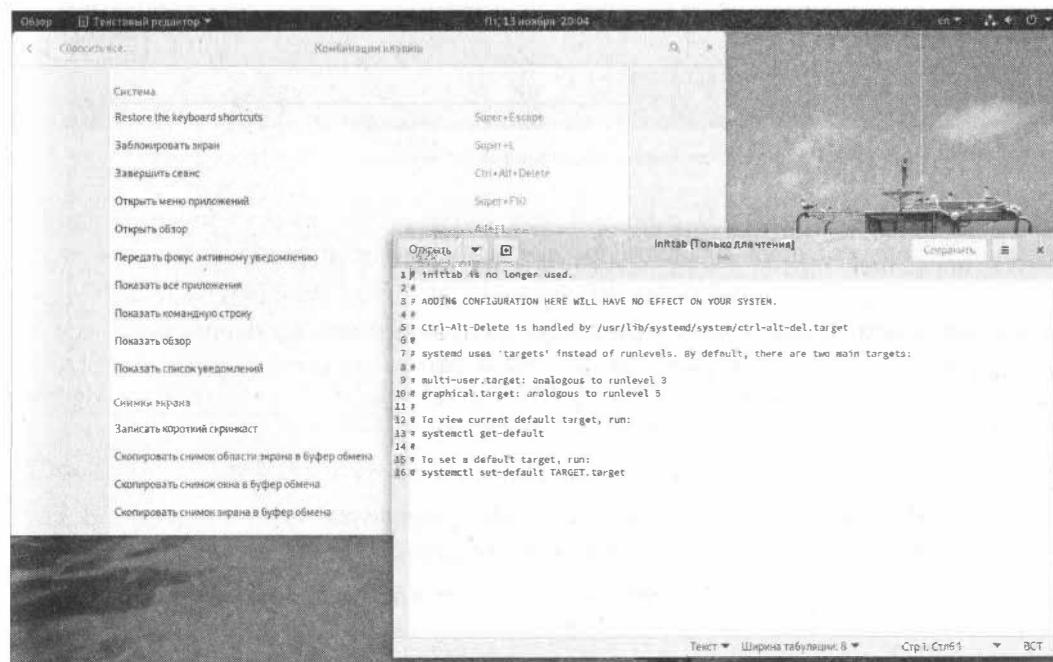


Рис. 22.1. Fedora 33: реакция в GNOME 3 на комбинацию клавиш `<Ctrl>+<Alt>+<Delete>`

Откройте окно **Параметры**, перейдите в раздел **Устройства | Клавиатура**, откройте группу **Система** (см. 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 на действии и нажмите клавишу <Пробел>.

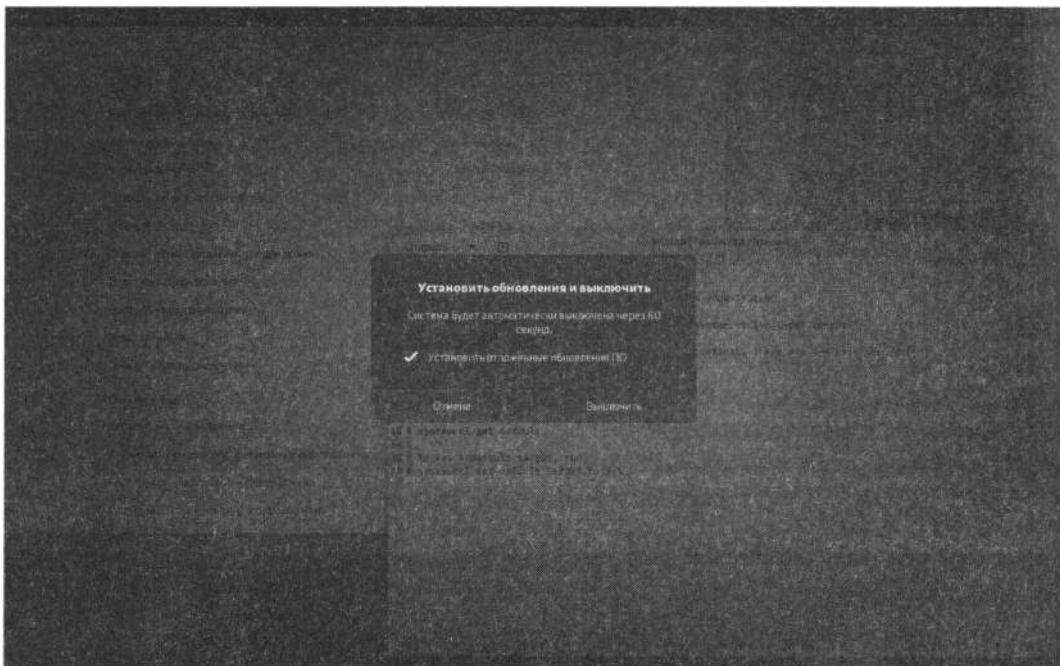


Рис. 22.2. Fedora 33: система отреагировала на нажатие <Ctrl>+<Alt>+<Delete>

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 другие 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.

Попробуем создать файл, описывающий сетевой сервис. В язык программирования Python встроен собственный небольшой веб-сервер, предназначенный для запуска веб-приложений, написанных на Python. Обычно он запускается командой:

```
/usr/bin/python -m SimpleHTTPServer 8000
```

где 8000 — номер порта. Проблема в том, что, если вы закроете окно консоли, в которой вы вводили эту команду, работа веб-сервера также будет прекращена, и веб-приложение перестанет работать. Но сейчас мы создадим сервис, обеспечивающий бесперебойную работу сервиса. Перейдите в каталог `/etc/systemd/system` и создайте файл `simplehttp.service`. Содержимое этого файла представлено в листинге 22.3.

#### Листинг 22.3. Файл simplehttp.service

```
[Unit]
Description=Wave2 HTTP Service
After=network.target

[Service]
Type=simple
User=root
WorkingDirectory=/var/www/wave2/dist
ExecStart=/usr/bin/python -m SimpleHTTPServer 8000
Restart=on-abort

[Install]
WantedBy=multi-user.target
```

Здесь все просто. Вам нужно только указать путь к папке с приложением — в нашем случае это: `/var/www/wave2/dist` и номер порта (8000), который будет использоваться веб-сервером. Запустить сервис можно, как обычно, командой:

```
sudo systemctl start simplehttp
```

Далее предлагаю вам продолжить знакомство с системой самостоятельно. Там все весьма несложно, даже проще, чем вы думаете. Откройте, к примеру, файл `graphical.target` из `/lib/systemd/system` (листинг 22.4).

#### Листинг 22.4. 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>                                                                                 |
| <code>system s status</code>     | <code>systemctl status s.service</code>                                         | Статус сервиса <code>s</code>                                                                                     |
| <code>ls /etc/rc.d/init.d</code> | <code>ls /lib/systemd/system/*.service<br/>/etc/systemd/system/*.service</code> | Получение списка служб                                                                                            |
| <code>chkconfig s on</code><br>⋮ | <code>systemctl enable s.service</code>                                         | Включает запуск сервиса <code>s</code> после перезагрузки (сервис <code>s</code> будет загружаться автоматически) |
| <code>chkconfig s off</code>     | <code>systemctl disable s.service</code>                                        | Отключает автоматический запуск сервиса <code>s</code>                                                            |
| <code>chkconfig s</code>         | <code>systemctl is-enabled s.servive</code>                                     | Проверяет, запускается ли <code>s</code> автоматически                                                            |
| <code>chkconfig s --list</code>  | <code>ls /etc/systemd/system/<br/>*.wants/s.service</code>                      | Выводит список целей (уровней запуска), на которых сервис будет запускаться автоматически                         |

## 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 заинтересовалась 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 его все же 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 существовать, 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:

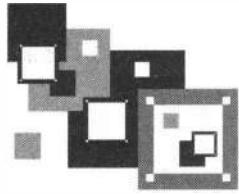
```
IPADDR[0] = "192.168.1.1"
NETMASK[0] = "255.255.255.0"
USE_DHCP[0] = ""
DHCP_HOSTNAME[0] = ""
```

Сценарий `/etc/rc.d/rc.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 22.5.

#### Листинг 22.5. 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`. Хоть это и маловероятно (не помню, чтобы он когда-нибудь зависал), но для примера сойдет. 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 -U 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 mc

Теперь, когда мы знаем PID нашего процесса, мы можем его «убить»:

```
# kill 2818
```

Перейдите на первую консоль после выполнения этой команды — mc на ней уже не будет. Если выполнить команду ps -a, то в списке процессов mc 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. При этом 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`.

| PID  | USER | PR | NI  | VIRT   | RES  | SHR  | S | %CPU  | %MEM  | TIME+   | COMMAND                     |
|------|------|----|-----|--------|------|------|---|-------|-------|---------|-----------------------------|
| 227  | root | 20 | 0   | 0      | 0    | 0    | R | 0.662 | 0.000 | 0:01.89 | <code>kworker/1:1</code>    |
| 2840 | root | 20 | 0   | 14000  | 2552 | 1940 | R | 0.662 | 0.252 | 0:00.05 | <code>top</code>            |
| 1    | root | 20 | 0   | 118268 | 5572 | 3376 | S | 0.000 | 0.550 | 0:05.17 | <code>systemd</code>        |
| 2    | root | 20 | 0   | 0      | 0    | 0    | S | 0.000 | 0.000 | 0:00.04 | <code>kthreadd</code>       |
| 3    | root | 20 | 0   | 0      | 0    | 0    | S | 0.000 | 0.000 | 0:02.72 | <code>ksoftirqd/0</code>    |
| 5    | root | 0  | -20 | 0      | 0    | 0    | S | 0.000 | 0.000 | 0:00.00 | <code>kworker/u128:0</code> |
| 6    | root | 20 | 0   | 0      | 0    | 0    | S | 0.000 | 0.000 | 0:01.58 | <code>kworker/u128:0</code> |
| 7    | root | -2 | 0   | 0      | 0    | 0    | S | 0.000 | 0.000 | 0:00.00 | <code>rcu_bh</code>         |
| 8    | root | -2 | 0   | 0      | 0    | 0    | S | 0.000 | 0.000 | 0:00.00 | <code>rcu_bh</code>         |
| 9    | root | -2 | 0   | 0      | 0    | 0    | S | 0.000 | 0.000 | 0:00.00 | <code>rcu_bh</code>         |
| 10   | root | 20 | 0   | 0      | 0    | 0    | S | 0.000 | 0.000 | 0:05.36 | <code>rcu_preempt</code>    |
| 11   | root | 20 | 0   | 0      | 0    | 0    | S | 0.000 | 0.000 | 0:03.13 | <code>rcu_bh</code>         |
| 12   | root | 20 | 0   | 0      | 0    | 0    | R | 0.000 | 0.000 | 0:01.70 | <code>rcu_bh</code>         |
| 13   | root | 20 | 0   | 0      | 0    | 0    | S | 0.000 | 0.000 | 0:00.00 | <code>rcu_bh</code>         |
| 14   | root | 20 | 0   | 0      | 0    | 0    | S | 0.000 | 0.000 | 0:00.00 | <code>rcu_bh</code>         |
| 15   | root | 20 | 0   | 0      | 0    | 0    | S | 0.000 | 0.000 | 0:00.00 | <code>rcu_bh</code>         |
| 16   | root | 20 | 0   | 0      | 0    | 0    | S | 0.000 | 0.000 | 0:00.00 | <code>rcu_bh</code>         |
| 17   | root | 20 | 0   | 0      | 0    | 0    | S | 0.000 | 0.000 | 0:00.00 | <code>rcu_bh</code>         |
| 18   | root | 20 | 0   | 0      | 0    | 0    | S | 0.000 | 0.000 | 0:00.00 | <code>rcu_bh</code>         |
| 19   | root | 20 | 0   | 0      | 0    | 0    | S | 0.000 | 0.000 | 0:00.00 | <code>rcu_bh</code>         |
| 20   | root | 20 | 0   | 0      | 0    | 0    | S | 0.000 | 0.000 | 0:00.00 | <code>rcu_bh</code>         |
| 21   | root | 20 | 0   | 0      | 0    | 0    | S | 0.000 | 0.000 | 0:00.00 | <code>rcu_bh</code>         |
| 22   | root | 20 | 0   | 0      | 0    | 0    | S | 0.000 | 0.000 | 0:00.00 | <code>rcu_bh</code>         |
| 23   | root | 20 | 0   | 0      | 0    | 0    | S | 0.000 | 0.000 | 0:00.00 | <code>rcu_bh</code>         |
| 24   | root | 20 | 0   | 0      | 0    | 0    | S | 0.000 | 0.000 | 0:00.00 | <code>rcu_bh</code>         |
| 25   | root | 20 | 0   | 0      | 0    | 0    | S | 0.000 | 0.000 | 0:00.00 | <code>rcu_bh</code>         |
| 26   | root | 20 | 0   | 0      | 0    | 0    | S | 0.000 | 0.000 | 0:00.00 | <code>rcu_bh</code>         |
| 27   | root | 20 | 0   | 0      | 0    | 0    | S | 0.000 | 0.000 | 0:00.00 | <code>rcu_bh</code>         |
| 28   | root | 20 | 0   | 0      | 0    | 0    | S | 0.000 | 0.000 | 0:00.00 | <code>rcu_bh</code>         |
| 29   | root | 20 | 0   | 0      | 0    | 0    | S | 0.000 | 0.000 | 0:00.00 | <code>rcu_bh</code>         |

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                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>PID</b>     | Идентификатор процесса                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>USER</b>    | Имя пользователя, запустившего процесс                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>PR</b>      | Приоритет процесса                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <b>NI</b>      | Показатель nice (см. разд. 23.3)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| <b>VIRT</b>    | Виртуальная память, использованная процессом (в Кбайт)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>RES</b>     | Размер процесса, не перемещенный в область подкачки (в Кбайт). Этот размер равен размерам сегментов кода и данных, т. е. <code>RES = CODE + DATA</code>                                                                                                                                                                                                                                                                                                                                                                                               |
| <b>S</b>       | Состояние процесса: <ul style="list-style-type: none"> <li>• R — выполняется;</li> <li>• S — «спит» (режим ожидания), в этом состоянии процесс выгружен из оперативной памяти в область подкачки;</li> <li>• D — «непрерываемый сон» (uninterruptible sleep), из такого состояния процесс может вывести только прямой сигнал от оборудования;</li> <li>• T — процесс в состоянии трассировки или остановлен;</li> <li>• Z (зомби) — специальное состояние процесса, когда сам процесс уже завершен, но его структура еще осталась в памяти</li> </ul> |
| <b>%CPU</b>    | Занимаемое процессом процессорное время                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <b>%MEM</b>    | Использование памяти процессом                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <b>TIME+</b>   | Процессорное время, израсходованное с момента запуска процесса                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <b>COMMAND</b> | Команда, которая использовалась для запуска процесса (обычно имя исполняемого файла процесса)                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

### ПРОГРАММА HTOP

Кроме стандартной программы `top` вы можете использовать программу `htop`, которая немного нагляднее. Она не устанавливается по умолчанию, поэтому для ее установки нужно установить пакет `htop` командой: `sudo apt install htop`.

## 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
```

## 23.4. Запуск NodeJs-приложений в фоновом режиме

Как правило, NodeJs-приложения запускаются командой:

```
npm start
```

При этом приложение запускается и работает на порту, заданном в настройках. В эру настоящих физических серверов, когда у администратора был полноценный доступ к консоли сервера (клавиатуре и монитору), проблем не возникало. Сегодня же, как правило, используются виртуальные серверы, к которым администраторы подключаются удаленно по SSH. Так вот, по SSH вы тоже можете так же запустить NodeJS-приложение, но как только вы закроете окно сеанса, это приложение будет завершено и прекратит работать на порту, указанном в настройках. Другими словами, ваш сайт перестанет работать, как только вы закроете окно SSH-консоли. В эру физических серверов такое поведение никого не напрягало: вы могли запустить прт на отдельной консоли, затем — нажав комбинацию клавиш `<Alt>+<Fn>` (где `n` — номер консоли) — переключиться на другую консоль и продолжить работу, как обычно. Сервер всегда включен, выключать ничего никогда не было нужно. Но в случае с виртуальным сервером все иначе — администратор не может выключить свой личный компьютер, поскольку это приведет к завершению прт и отказу сайта. Ситуация очень неприятная, поэтому нужно искать способ запуска прт в фоновом режиме. Причем команда `prtm start&` в случае с прт не работает.

Специально для таких случаев был разработан PM2 (Process Manager 2). Он позволяет поддерживать работу ваших NodeJs-приложений в режиме 24/7.

Установить его можно так:

```
sudo npm install pm2 -g
```

После этого перейдите в папку с приложением и введите команду:

```
pm2 --name <Имя> start npm -- start
```

Здесь <Имя> — произвольная строка, задающее название процесса приложения в списке процессов PM2. Другими словами, в этой строке вы можете указать что угодно, и она будет идентифицировать приложение в списке процессов.

Все, после этого можно закрывать консоль — приложение будет работать.

Для вывода списка процессов используется команда:

```
pm2 ps
```

Для удаления процесса нужно задать его ID (выводится в списке процессов слева от имени процесса, как показано на рис. 23.4). Команда будет такой:

```
pm2 delete <ID>
```

Дополнительная информация о pm2 доступна в документации по адресу: <https://pm2.keymetrics.io/docs/usage/pm2-doc-single-page/>.

The screenshot shows a terminal window with the following content:

```
root@vm658224:/var/www/flex
Runtime Edition

PM2 is a Production Process Manager for Node.js applications
with a built-in Load Balancer.

Start and Daemonize any application:
$ pm2 start app.js

Load Balance 4 instances of api.js:
$ pm2 start api.js -i 4

Monitor in production:
$ pm2 monitor

Make pm2 auto-boot at server restart:
$ pm2 startup

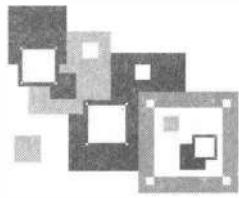
To go further checkout:
http://pm2.io/

[PM2] Spawning PM2 daemon with pm2_home=/root/.pm2
[PM2] PM2 Successfully daemonized
[PM2] Starting /usr/bin/npm in fork_mode (1 instance)
[PM2] Done.

id      name        mode    status   cpu     memory
0       Flex        fork    0       online  0%     25.4mb

root@vm658224:/var/www/flex#
```

Рис. 23.4. Приложение Flex запущено в фоновом режиме



## 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 (есть 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 этойшине.

## 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>+<Del> 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>+<Del> равнозначно нажатию кнопки <code>Reset</code> . Сами понимаете, значение 1 устанавливать не рекомендуется                                                                                                                                                                                                                                     |
| /proc/sys/kernel/domainname   | Здесь находится имя домена — например, <code>dkws.org.ua</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| /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 <code>kernel panic</code> 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 1 7. 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                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|---------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| /proc/sys/net/core/message_burst      | 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 |
| /proc/sys/net/core/message_cost       | Чем выше значение в этом файле, тем больше сообщений будет проигнорировано в перерыв, заданный файлом message_burst                                                                                                                                                                                                                                                                                                                                                                                                             |
| /proc/sys/net/core/netdev_max_backlog | Задает максимальное число пакетов в очереди. По умолчанию — 300. Используется, если сетевой интерфейс передает пакеты быстрее, чем система может их обработать                                                                                                                                                                                                                                                                                                                                                                  |
| /proc/sys/net/core/optmem_max         | Задает 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 суперблоков;

### 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).

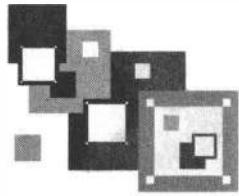
- в файле *super-pr* находится количество открытых суперблоков в текущий момент. Этот файл нельзя записывать, его можно только читать.

## 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 0311191707
```

Первая команда выводит дату, а вторая — устанавливает дату (при условии, что команда запущена от имени root) 3 ноября (0311) 2019 года (19) 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` покажет 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     2019-07-23 12:59
den          pts/1     2019-07-23 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 *cstr* — 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 `cmp` вообще ничего не выводит, а вот если файлы различаются, 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` приведены 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:<br>0 — файлы одинаковые;<br>1 — файлы отличаются;<br>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 n строк после строки, в которой есть искомая строка                           |
| <code>-B n</code>      | Вывод n 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`.

Примеры использования:

```
head -n 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**, **pico**, **ee**, **mcedit** — текстовые редакторы

В предыдущих изданиях книги в этом разделе присутствовало развернутое описание классических текстовых редакторов **vi**, **nano**, **pico**, **ee** и **mcedit**. Однако с включением в книгу главы 16, содержащей описание современных текстовых редакторов кода, мне представилось более уместным перенести в нее материал этого раздела, чтобы у читателей сформировалось более цельное представление о рассматриваемой в ней теме.

### 25.2.7. Язык **gawk** — мощное средство обработки текста

Если есть необходимость в мощном средстве обработки текста, рассмотрите использование языка **gawk**. Это небольшой язык, позволяющий создавать сценарии, ориентированные на обработку тестовых файлов. Подробно о языке **gawk** рассказано в материале «*Автоматизация обработки текста средствами gawk*», расположенном в папке Дополнение сопровождающего книгу файлового архива (см. *приложение*).

## 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. 25.4). 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    rhelp    type
cd        idle     ntrans   rename   user
cdup     image    open     reset    umask
chmod    lcd      passive  restart  verbose
clear    ls       private  runique?
close    macdef  prompt   protect  safe
cprotect mdelete
ftp> ■
```

Рис. 25.4. Список команд FTP-клиента

Кроме `ftp`, для Linux разработаны и другие текстовые FTP-клиенты:

- NcFTP — <http://www.ncftp.com>;
- lukeftp — <ftp://ftp.netbsd.org/pub/NetBSD/misc/lukeftp/>;
- 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. 25.5 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. 25.5. 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

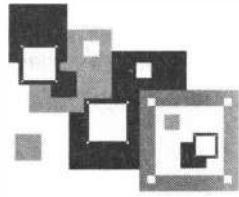
передаче был поврежден (или это другая версия пакета, которая, возможно, была подложена злоумышленником с целью ввода вражеского кода в вашу систему).

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 соответствующие страницы руководства (*man*).



## ГЛАВА 26

# Конфигурационные файлы Linux

### 26.1. Каталог /etc

Все пользователи Windows наверняка слышали о «святая 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 дистрибутивов Ubuntu 19.04 и Fedora 30. 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 веб-сервер Apache, поэтому в нем имеется каталог `/etc/apache2`, содержащий файлы конфигурации этого

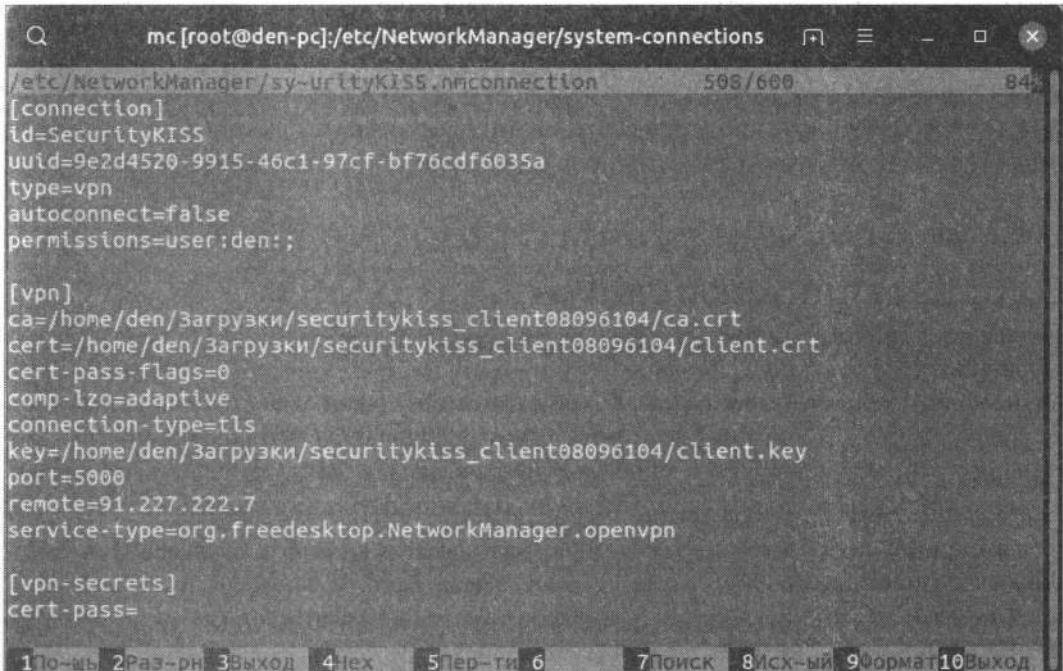
веб-сервера. В вашей системе веб-сервер может быть не установлен, поэтому такого каталога у вас не будет, зато будут свои конфигурационные каталоги и файлы, которых нет у меня.

## 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 (незашифрованном) виде...

На рис. 26.1 приведен файл конфигурации моего VPN-соединения с сервером SecurityKISS. Как видите, его формат очень прост.



```
mc [root@den-pc]:/etc/NetworkManager/system-connections 593/600 84%
```

```
[connection]
id=SecurityKISS
uuid=9e2d4520-9915-46c1-97cf-bf76cdf6035a
type=vpn
autoconnect=false
permissions=user:den:;

[vpn]
ca=/home/den/Загрузки/securitykiss_client08096104/ca.crt
cert=/home/den/Загрузки/securitykiss_client08096104/client.crt
cert-pass=flags=0
comp-lzo=adaptive
connection-type=tls
key=/home/den/Загрузки/securitykiss_client08096104/client.key
port=5900
remote=91.227.222.7
service-type=org.freedesktop.NetworkManager.openvpn

[vpn-secrets]
cert-pass=
```

Рис. 26.1. Ubuntu: файл конфигурации VPN-соединения, созданный с помощью NetworkManager

## 26.3. Каталог /etc/abrt

Этот каталог содержит файлы конфигурации утилиты abrt (Automatic bug detection and reporting tool), служащей 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 станете вызывать abrt вручную. Кроме того, в Ubuntu abrt вовсе не используется, поэтому ее каталога вы там не найдете.

## 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 этом случае применение конфигураторов оправданно, поскольку изучение таких конфигурационных файлов нецелесообразно с точки зрения расходования времени.

## 26.5. Каталоги /etc/audit и /etc/audisp

Эти каталоги содержат конфигурационные файлы демона аудита: auditd и его диспетчера событий — audisp (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. Как и в случае с abrt, демон auditd и диспетчер событий audisp «вырезаны» из Ubuntu, и этих каталогов вы в ней не найдете.

## 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 Windows-файла /etc/hosts, но для Avahi).

Демон Avahi на практике используется весьма 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. Файлы конфигурации планировщиков задач

В 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
```

## 26.8. 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 26.1.

### Листинг 26.1. 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 (State Idle), т. е. ничего не печатается, к тому же принтер является общим (Shared Yes).

Файл конфигурации `/etc/cups/cupsd.conf` определяет настройки сервера печати. Пример этого файла вместе с комментариями представлен в листинге 26.2.

#### Листинг 26.2. 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

# Ограничеваем 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>

    <Limit All>
        Order deny,allow
    </Limit>
</Policy>
```

## 26.9. 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.

## 26.10. Каталог /etc/gdm (или /etc/gdm3)

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. В дистрибутивах, где используется менеджер `lightdm`, будет присутствовать каталог `/etc/lightdm`. Именно такой каталог был до недавнего в Ubuntu, пока она не отказалась от собственной оболочки Unity и не перешла на GNOME.

## 26.11. Файлы конфигурации популярных сетевых служб

Файлы конфигурации популярных сетевых служб, таких как веб-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
<code>/etc/ssh/sshd_config</code>	SSH-сервер	32
<code>/etc/httpd/conf/httpd.conf</code> – Fedora	Веб-сервер Apache	33
<code>/etc/apache2/apache2.conf</code> – Ubuntu	Веб-сервер Apache	33
<code>/etc/proftpd/proftpd.conf</code>	FTP-сервер ProFTPD	34
<code>/etc/bind/named.conf</code>	DNS-сервер	35
<code>/etc/squid/squid.conf</code>	Прокси-сервер SQUID	36
<code>/etc(exports</code>	NFS (Network File System)*	*
<code>/etc/samba/smb.conf</code>	Samba: доступ к Windows-сети	38

\* Сетевая файловая система.

Материал с описанием сетевой файловой системы (Network File System) вы найдете в папке Дополнения сопровождающего книгу файлового архива (см. [приложение](#)).

## 26.12. 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 26.3).

#### Листинг 26.3. 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  
}
```

## 26.13. 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.

## 26.14. Каталог /etc/ntp

Этот каталог содержит файлы конфигурации сервера времени. Сервер времени подробно рассматривается в моей книге «Серверное применение Linux»<sup>1</sup>. А вообще, файлы из этого каталога редактируются весьма редко. Все, что требуется, — это в файле /etc/ntp/ntp servers прописать серверы времени, откуда нужно получать точное время.

## 26.15. 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>.

## 26.16. 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. 26.14. Виртуальным частным сетям также посвящены главы 10 и 46 настоящего издания.

## 26.17. 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.

## 26.18. Каталог /etc/ppp

Этот каталог содержит файлы конфигурации демона pppd, отвечающего за установку PPP-соединений (в том числе и PPPoE-соединений):

---

<sup>1</sup> Колисниченко Д. Н. Серверное применение Linux. — 3-е изд. — СПб.: БХВ-Петербург, 2011, <http://bhv.ru/books/book.php?id=188723>.

- `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.

## 26.19. Каталог `/etc/rc.d`

Каталог `/etc/rc.d` содержит сценарии инициализации системы. Подробно этот каталог рассматривается в главе 22.

## 26.20. 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](http://www.dkws.org.ua/novice/pdf/printer_scanner.pdf).

## 26.21. Каталог `/etc/selinux`

Это конфигурационный каталог системы управления доступом SELinux. В этом издании система SELinux не рассматривается.

Описание системы управления доступом SELinux (главу 33 из 2-го издания книги) вы найдете в папке *Дополнения* сопровождающего книгу файлового архива (см. *приложение*).

## 26.22. 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`.

## 26.23. Каталог /etc/sysconfig

Каталог `/etc/sysconfig` содержит конфигурационные файлы системы. В нем очень много файлов, и все их мы рассматривать не станем. Каждый файл тщательно за-комментирован, 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`;
- ❑ `fstboot` — 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.

## 26.24. Каталог /etc/X11

В этом каталоге содержатся настройки системы X.Org. Основной конфигурационный файл этой системы `xorg.conf` подробно описан в главе 13, поэтому не вижу смысла рассматривать его еще и здесь.

## 26.25. 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 репозиториев рассматривается в главе 7 — если вы ее пропустили, то самое время прочитать. Поскольку в дистрибутиве Fedora, 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 30) конфигурация репозиториев хранится в файле `/etc/yum.repos.d`, хотя уже несколько лет как используется менеджер `dnf`.

## 26.26. 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.

## 26.27. Остальные конфигурационные файлы каталога /etc

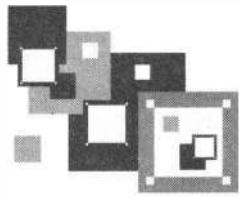
В табл. 26.2 представлены подробно не рассмотренные пока файлы конфигурации из каталога `/etc`. Еще раз замечу, что файлы тщательно прокомментированы, поэтому вам даже не придется читать по ним какую-либо дополнительную документацию.

**Таблица 26.2. Некоторые конфигурационные файлы**

Файл	Описание
<code>bashrc</code>	Содержит описание некоторых функций командного интерпретатора <code>bash</code> . Обычно этот файл никогда не редактируется
<code>filesystems</code>	В файле находится список поддерживаемых ядром файловых систем. Сугубо информационный файл, вам не нужно его редактировать
<code>fstab</code>	Описывает файловые системы, монтируемые при запуске операционной системы. Подробно описан в главе 4
<code>group</code>	Содержит информацию о группах пользователей (см. главу 6)
<code>issue</code>	Текст сообщения, выводимого перед локальной регистрацией пользователя в системе

**Таблица 26.2 (окончание)**

Файл	Описание
issue.net	Текст сообщения, выводимого перед удаленной регистрацией пользователя в системе
man_db.conf	Файл конфигурации справочной системы man
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 в нем были, перенесены в каталог /etc/modprobe.d. Также загляните в каталог modules-load.d — 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. главу 6)
protocols	Содержит список поддерживаемых протоколов
shells	Содержит список командных интерпретаторов, установленных в системе
sudoers	Определяет, кому можно использовать команду sudo (см. главу 6)
sysctl.conf	Системная конфигурация ядра



## ГЛАВА 27

# Протоколирование системы

В любой UNIX-подобной 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 `journalctl`.

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 `journalctl` — привычные файлы вроде `/var/log/messages` более недоступны. Конечно, вы можете параллельно установить демон `rsyslogd`, и он будет прекрасно работать в паре с `journalctl`. Однако у `journalctl` гораздо больше возможностей — например, с помощью опции `-b` можно просмотреть логи текущей или предыдущей загрузки. Некоторые возможности утилиты `journalctl` рассмотрены в этой книге, а с остальными вы сможете ознакомиться в справочной системе;

### УТИЛИТА JOURNALCTL

Полное описание утилиты `journalctl` 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.

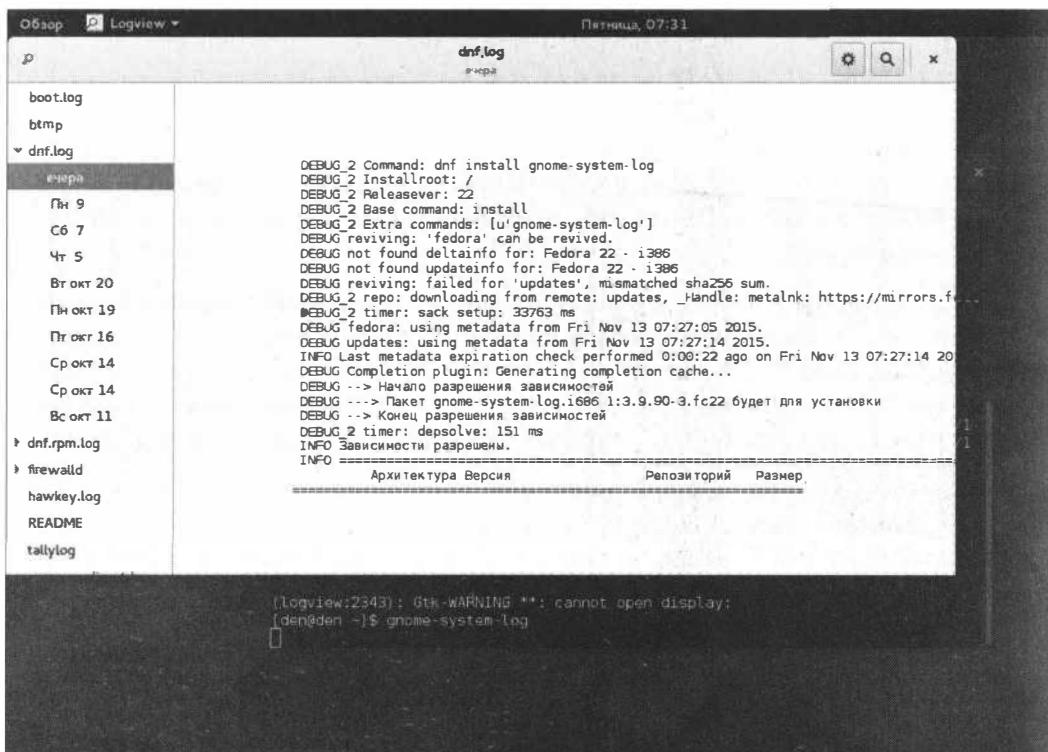


Рис. 27.1. Fedora: утилита `gnome-system-log`

## 27.1. Протоколирование по-новому: `journalctl`

### 27.1.1. Установка времени

Первое, что нужно сделать, — это установить правильный часовой пояс. Основной недостаток `syslogd/rtsyslogd` 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 2019-10-11 21:20:24 EEST, end at Пт 2019-11-15 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 cpuset  
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 0x000000000009f800  
окт 08 21:20:24 localhost.localdomain kernel: BIOS-e820: [mem 0x00000000000ca000  
окт 08 21:20:24 localhost.localdomain kernel: BIOS-e820: [mem 0x00000000000dc000  
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 0x0000000002feff0000  
окт 08 21:20:24 localhost.localdomain kernel: BIOS-e820: [mem 0x0000000002ff00000  
окт 08 21:20:24 localhost.localdomain kernel: BIOS-e820: [mem 0x00000000e0000000  
окт 08 21:20:24 localhost.localdomain kernel: BIOS-e820: [mem 0x00000000fec00000  
окт 08 21:20:24 localhost.localdomain kernel: BIOS-e820: [mem 0x00000000fee00000  
окт 08 21:20:24 localhost.localdomain kernel: BIOS-e820: [mem 0x00000000ffffe0000  
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:~$ journalctl --list-boots
-13 479175409c394b9eba9836d9e7911240 Sat 2019-08-17 10:13:59 EEST-Sat 2019-08-17 07:42:15 EEST
-12 109f90848e2a41c68398fdbcd0993331 Sun 2019-08-18 07:00:08 EEST-Sun 2019-08-18 07:26:52 EEST
-11 8349616f2b7c48859b14f23108d40828 Mon 2019-08-19 22:13:30 EEST-Mon 2019-08-19 22:27:36 EEST
-10 5c7e29382bf44e599d2323021b338bd2 Mon 2019-08-19 22:28:24 EEST-Mon 2019-08-19 22:38:31 EEST
-9 495dbdf13d5441ac84e0d6454e70a14c Mon 2019-08-19 22:38:49 EEST-Mon 2019-08-19 22:59:53 EEST
-8 2ff50603fc59b428a8e34ee2ba2ef9626 Tue 2019-08-20 07:02:40 EEST-Tue 2019-08-20 07:28:26 EEST
-7 7eb788bcc6974b09b3aea12df0d9da20 Wed 2019-08-21 08:07:29 EEST-Wed 2019-08-21 08:17:20 EEST
-6 beaaa13c86b54bba892cd06937a5b7b4 Thu 2019-08-22 07:19:28 EEST-Thu 2019-08-22 08:47:35 EEST
-5 b376f658bcfa4fa6be396d06e4bf4849 Fri 2019-08-23 10:24:54 EEST-Tue 2019-08-27 07:14:37 EEST
-4 3ea0b885916e461bb2e4630074827e3c Tue 2019-08-27 07:17:27 EEST-Tue 2019-08-27 09:26:15 EEST
-3 c359aab5eaa4420a9aadce7107562f41 Wed 2019-08-28 08:38:33 EEST-Sun 2019-09-01 20:01:02 EEST
-2 dd5903c982394913b016e28790b9ff33 Wed 2019-09-11 04:53:30 EEST-Wed 2019-09-11 04:53:40 EEST
-1 bfbcc0c2c949f446786e72f2028fd1728 Tue 2019-10-29 20:39:27 EET-Tue 2019-10-29 21:15:41 EET
0 6413d169f31e40a7807a6ba776a681f9 Sun 2019-11-03 17:18:53 EET-Sun 2019-11-03 17:21:33 EET
-
```

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 11.10.19 7:00 введите команду:

```
$ journalctl --since "2019-10-11 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 (то есть, 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 -k или --dmesg:

```
$ journalctl -k
```

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 -k -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 весьма 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;
- user — 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.

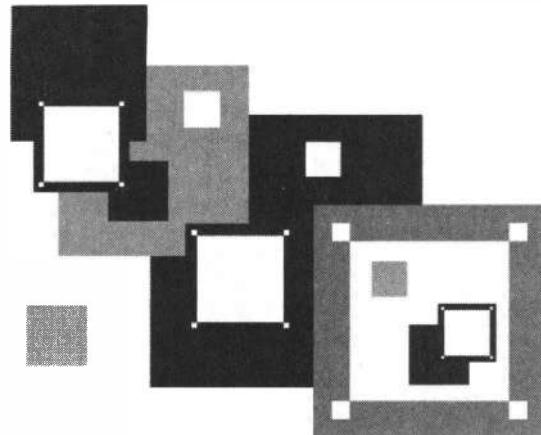
Теперь перейдем ко второму параметру файла конфигурации — к параметру `действие`. В большинстве случаев `действие` — это имя файла журнала, в который нужно записать сообщение селектора. Если перед именем файла стоит дефис (-), то после каждой записи в журнал демон не станет выполнять синхронизацию файла, 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)**





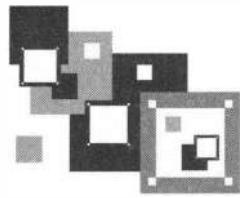


## ЧАСТЬ VI

### 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 при загрузке в однопользовательском режиме запрашивала пароль. Но мы не станем 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-

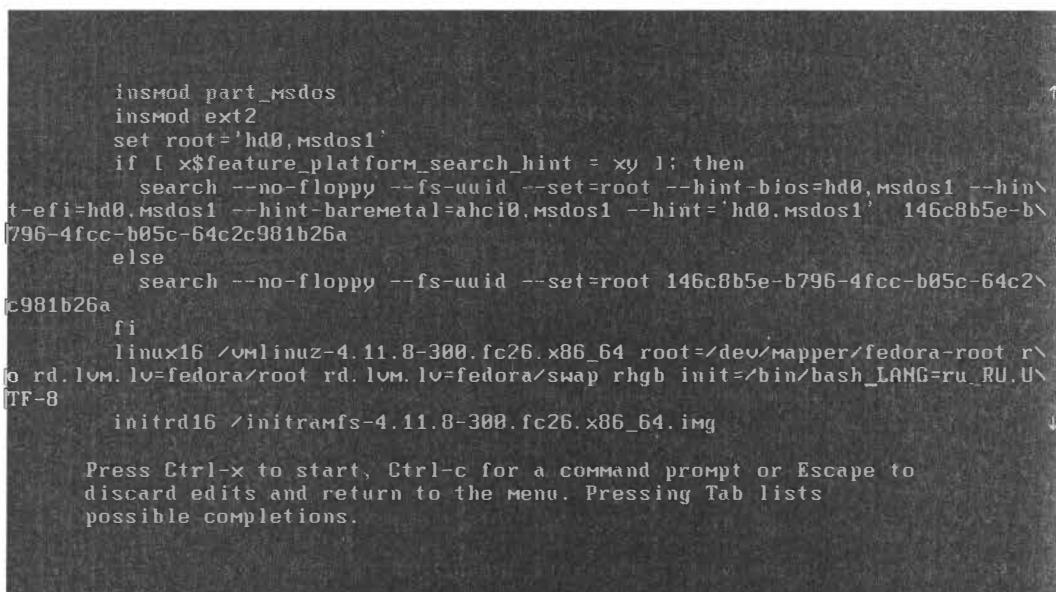


Рис. 28.1. Fedora 33: передаем параметр ядра



Рис. 28.2. Fedora 33: 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 файлового архива* (см. *приложение*) содержиться 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 33 можно указать параметр ядра `init=/bin/bash` (рис. 28.1), и система будет загружена с правами `root` (рис. 28.2).

## **28.1.2. Пароль загрузчиков GRUB/GRUB2**

Защитить систему лучше с помощью загрузчиков GRUB/GRUB2. 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 загрузчик запросит пароль.

О том, как установить пароль загрузчиков GRUB и GRUB2, 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 полу-

чит физический доступ к серверу, то ничего ему не помешает вытащить из розетки кабель питания. Вот и весь секрет...

Но мы можем блокировать хотя бы программную перезагрузку, осуществляющую с помощью клавиатурной комбинации **<Ctrl>+<Alt>+<Del>**.

Для этого в старых дистрибутивах, основанных на системе инициализации init, откройте файл */etc/inittab* и найдите в нем строчку:

```
ca::ctrlaltdel:/sbin/shutdown -r -t 4 now
```

Эту строчку 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>+<Del>** 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>+<Del>** 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"
```

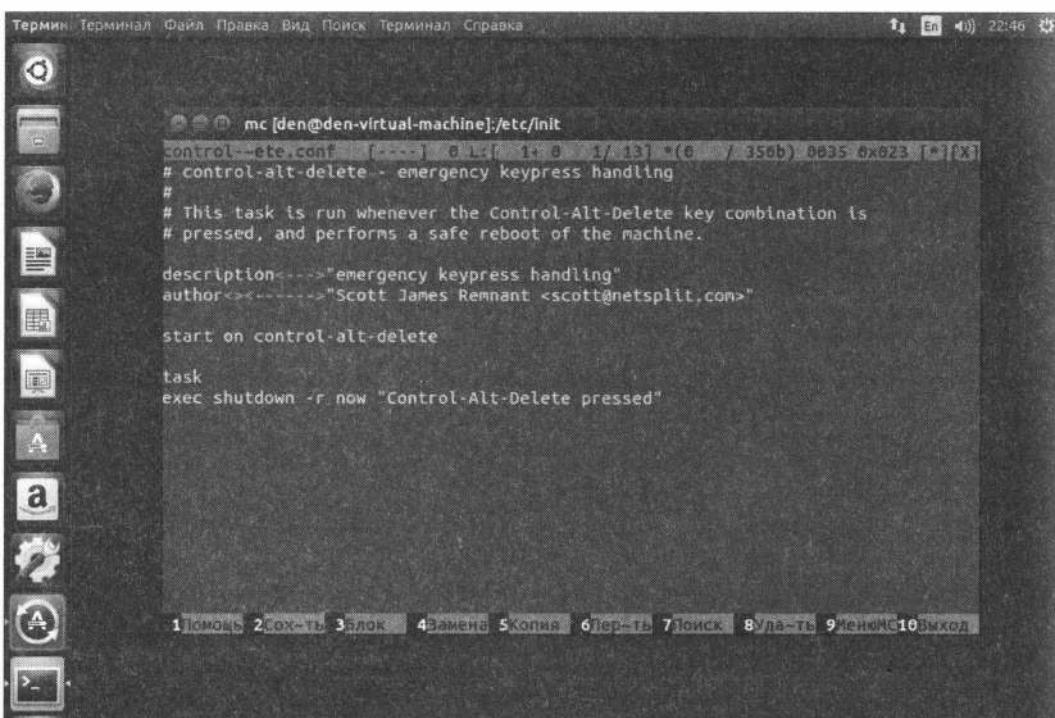


Рис. 28.3. Ubuntu: редактирование файла */etc/init/control-alt-delete.conf*

и закомментируйте ее с помощью символа #:

```
#exec shutdown -r now "Control-Alt-Delete pressed"
```

В дистрибутивах, основанных на системе инициализации systemd, — например, в последних версиях Fedora, реакция на нажатие комбинации клавиш <Ctrl>+<Alt>+<Del> 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`.

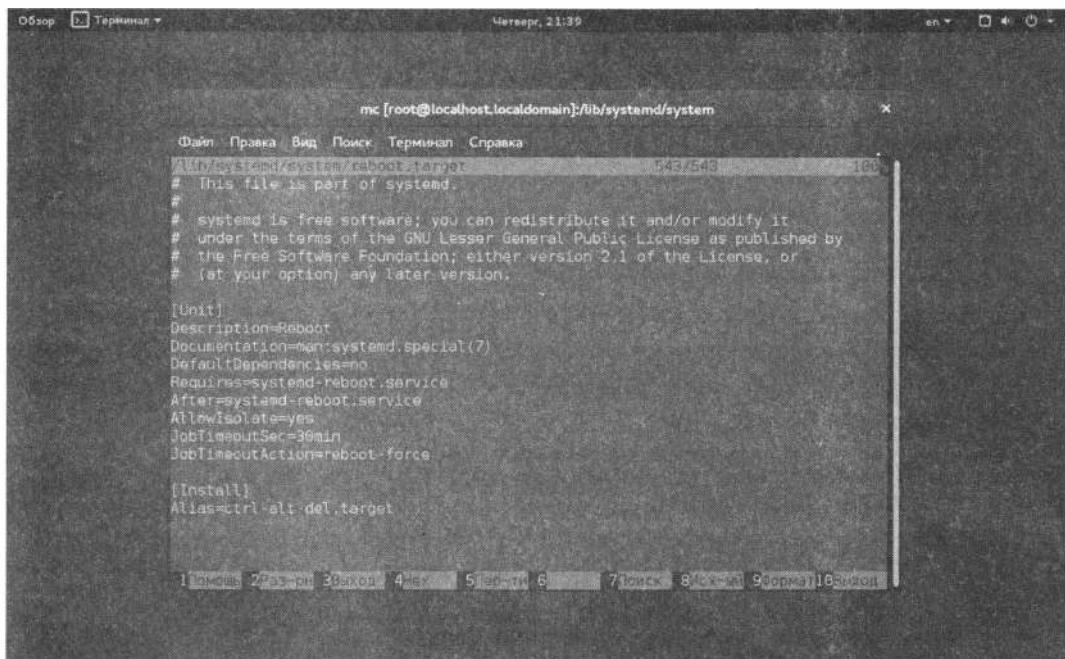


Рис. 28.4. Fedora: редактирование файла `/lib/systemd/system/reboot.target`

В разд. 22.3.5 также приведен способ изменить реакцию на нажатие комбинации клавиш <Ctrl>+<Alt>+<Del> — на уровне GNOME 3 (см. рис. 22.1 и 22.2).

## 28.3. Отключение учетной записи root: нестандартный метод

Весьма 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 «доброжела-

тель», 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>+<Del> 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. 5.1?). 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/sda1 /
```

Конечно, фрагмент `/dev/sda1` нужно заменить в этой команде на имя раздела, на который установлена Linux. Вуаля! 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

Достаточно часто «доброжелатели» знают, что такое `root`, но не знают ни одной UNIX-команды — следовательно, не смогут причинить системе особого вреда в консоли, но могут наворотить 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 KDE 3.5) 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.

Ежели вы выбрали графическую среду GNOME, то в роли графического менеджера в ней выступает GDM (G Display Manager). С его помощью 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
```

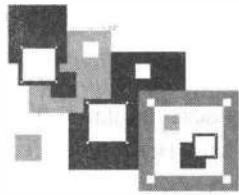
Подробно о конфигурационном файле `custom.conf` можно прочитать по адресу: <http://projects.gnome.org/gdm/docs/2.14/configuration.html?pagewanted=all>.

## 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 всерьез заботитесь о безопасности сервера, настоятельно рекомендую установить и настроить одну из *систем управления доступом* (LIDS, GrSecurity, SELinux, TomoYO). К сожалению, рассмотрение систем управления доступом выходит за рамки этой книги, но в Интернете вы найдете всю необходимую информацию.

### **СИСТЕМА УПРАВЛЕНИЯ ДОСТУПОМ SELINUX**

Описание системы управления доступом SELinux (главу 33 из 2-го издания книги) вы найдете в папке *Дополнения* сопровождающего книгу файлового 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 достаточно 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 для входа в систему вводит имя пользователя и пароль. 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, то есть 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
<b>Тип модуля</b>	
auth	Модуль аутентификации. Проверяет наличие пользователя в системе, запрашивает его имя, разрешает или запрещает доступ в ту или иную группу, независимо от содержимого файла <i>/etc/groups</i> , 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.
<b>Тип контроля</b>	
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 required, 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 *ram\_access*;
- console.perms* — определяет права, получаемые привилегированными пользователями консоли во время входа в систему и возвращаемые при выходе. Эти настройки обслуживает модуль *ram\_console*;

- ❑ `group.conf` — позволяет указать, какой группе будет принадлежать служба, запущенная определенным пользователем в определенное время с определенного терминала. Эти настройки обслуживаются модулями `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).

Весьма 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, и 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.)
#:wbscaro wsbsecr 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 задействования 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);
- cputime — 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
<code>pam_cracklib</code>	<code>password</code>	Проверяет пароль на стойкость. К полезным параметрам модуля можно отнести следующие: <code>minlen=N</code> (минимальная длина пароля), <code>dcredit=N</code> (минимальное количество цифр), <code>lcredit=N</code> (количество строчных букв), <code>ocredit</code> (количество прописных букв и других символов). Подробно этот модуль рассмотрен в разд. 29.5
<code>pam_deny</code>	Любой	Всегда закрывает доступ
<code>pam_env</code>	<code>auth</code>	Контролирует сохранность переменных среды
<code>pam_ftp</code>	<code>auth</code>	Предназначен для организации анонимного доступа по FTP. Получив имя пользователя <code>anonymous</code> , ждет что-то, напоминающее e-mail, в качестве пароля
<code>pam_group</code>	<code>auth</code>	Предназначен для обслуживания файла <code>groups.conf</code>
<code>pam_lastlog</code>	<code>auth</code>	Сообщает о времени и месте входа в систему. Обновляет файл <code>/var/log/wtmp</code>
<code>pam_limits</code>	<code>session</code>	Обслуживает файл <code>limits.conf</code>
<code>pam_listfile</code>	<code>auth</code>	Предназначен для организации доступа на основе конфигурационных файлов-списков — например, <code>/etc/ftpaccess</code>
<code>pam_mail</code>	<code>auth</code>	Сообщает о наличии почты, если таковая имеется
<code>pam_nologin</code>	<code>auth</code>	Стандартная реакция на наличие файла <code>/etc/nologin</code> . Если он присутствует, в систему может зайти только <code>root</code> , а остальным будет выдано на экран содержимое этого файла
<code>pam_permit</code>	Любой	Всегда означает успешную аутентификацию. Использование этого модуля опасно
<code>pam_pwdb</code>	Любой	Замещает модули серии <code>pam_unix</code> . Использует интерфейс библиотеки <code>libpwdb</code> (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 hosts.equiv и .rhosts, используемых для аутентификации таких служб, как rlogin и rsh
pam_root_ok	auth	Используется в случае, когда администратору необходимо получать доступ к сервису без введения пароля
pam_securetty	auth	Обслуживает файл /etc/securetty. Пользователь root может зайти с консоли, указанной в этом файле
pam_time	account	Этот модуль обслуживает файл time.conf (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 root может только пользователь группы wheel (в Linux она называется root). Можно отметить параметр group, 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 устанавливают так:

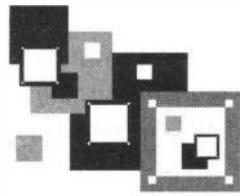
```
password required /lib/security/pam_cracklib.so параметры
```

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, swap), 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 настоящую «обжору», съедающую десятки мегабайт памяти. Сама X.Org тоже не особенно требовательна к памяти, чего не скажешь о графических интерфейсах GNOME и KDE, — 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 мегабайт оперативной памяти у вас свободно. Обратите внимание и на «остаток» области подкачки. Если и там, и там осталось всего несколько мегабайт памяти, значит, вам пора покупать 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 2 Гбайт, это не сравняется с одним настоящим модулем памяти на 1 Гбайт. 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 программами 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 медленнее, 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 поскупились 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).

### Несколько файлов подкачки

При желании можно создать несколько файлов подкачки. Но не создавайте файлы подкачки слишком больших размеров — они попросту будут без пользы занимать драгоценное место на диске (в случае с SSD-накопителями — действительно драгоценное).

## 30.3. Настройка планировщика ввода/вывода

Производительность многозадачной системы в целом сильно зависит от правильно-го 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 сможет правильно «угадывать», то производительность системы существенно увеличится. Ясно, что работа этого алгоритма очень зависит от логики программы;
- во втором случае (значение `cfq`) ядро станет 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=значение
```

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 материнская плата вашего компьютера 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 взять одинаковые модули одного и того же производителя — тогда вам гарантирована полная совместимость.

К слову, если вам нужно 8 Гбайт оперативной памяти, то, с учетом сказанного, 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 это вариант не лучший. Представьте, что ваш рабочий день будет начинаться с команды запуска программы обновления антивируса, а каждые 30 минут вам придется запускать 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 -n 19 run-parts --report /etc/cron.hourly
02 4 * * * root nice -n 19 run-parts --report /etc/cron.daily
22 4 * * 0 root nice -n 19 run-parts --report /etc/cron.weekly
42 4 1 * * root nice -n 19 run-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 `update_av 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, и 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

Теперь понятно, что для реализации нашего расписания 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 выполнятьсь 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 `cron`, дальнейшее его развитие. Главное преимущество `anacron` заключается в том, что он, в отличие от `cron`, учитывает время, когда компьютер был выключен. Планировщик `cron` родом из UNIX, а эта операционная система устанавливалась только на серверах, которые всегда включены. Предположим, что вам нужно каждый понедельник в 7 часов утра рассыпать некоторую информацию вашим сотрудникам. Вы настроили `cron` 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, `cron` не выполнит задание по отправке сообщений, а ваши сотрудники не получат 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 получат 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 `cron` 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

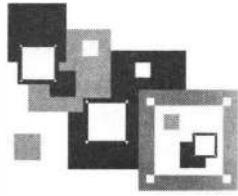
Иногда нужно просто выполнить определенные команды в определенное время (однократно), поэтому редактировать для этого таблицу crontab не совсем уместно. Такую задачу можно решить более рационально. Убедитесь, что у вас установлен и запущен демон atd. После этого введите команду:

```
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 AM/PM-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.

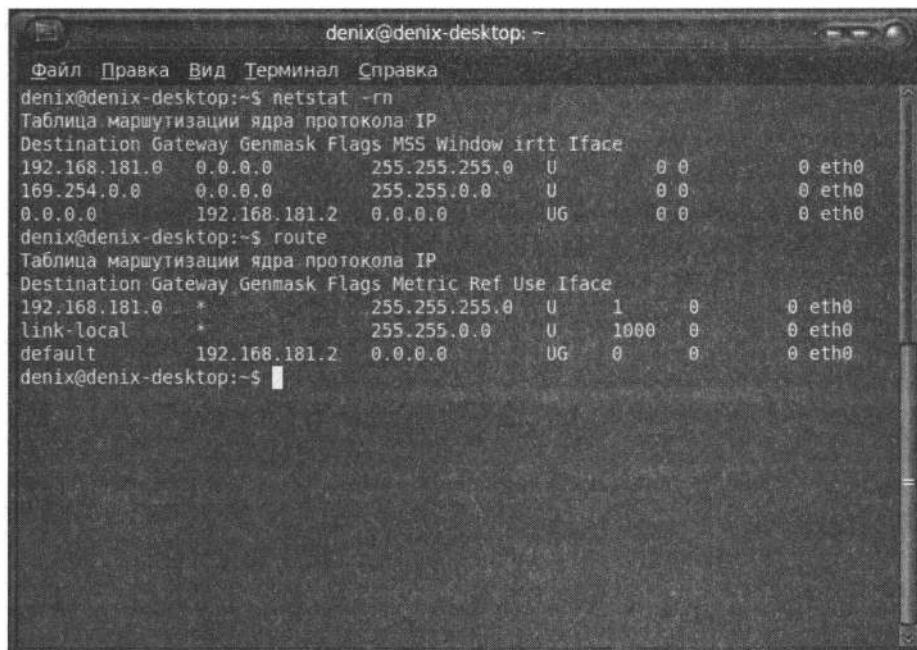
Данные о маршрутах хранятся в таблице маршрутизации ядра Linux. Каждая запись этой таблицы содержит несколько параметров: адрес сети назначения, сетевую маску и т. д. Если пакет не удалось отправить ни по одному маршруту (в том числе и по стандартному), отправителю пакета передается ICMP-сообщение «сеть 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 применял `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
```

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 этих 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
<b>Destination</b>	Адрес сети назначения
<b>Gateway</b>	Шлюз по умолчанию
<b>Genmask</b>	Маска сети назначения
<b>Flags</b>	Поле <b>Flags</b> содержит флаги маршрута: <ul style="list-style-type: none"> <li>• U — маршрут активен;</li> <li>• H — маршрут относится не к сети, а к хосту;</li> <li>• G — эта 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);</li> <li>• D — динамический маршрут, установлен демоном маршрутизации;</li> <li>• M — маршрут, модифицированный демоном маршрутизации;</li> <li>• C — запись кэширована;</li> <li>• ! — запрещенный маршрут</li> </ul>
<b>Metric</b>	Метрика маршрута, т. е. расстояние к цели в хопах (переходах). Один хоп (переход) означает один маршрутизатор
<b>Ref</b>	Количество ссылок на маршрут. Не учитывается ядром Linux, но в других операционных системах, например в FreeBSD, вы можете столкнуться с этим полем
<b>Use</b>	Содержит количество пакетов, прошедших по этому маршруту
<b>Iface</b>	Используемый интерфейс
<b>MSS</b>	Максимальный размер сегмента (Maximum Segment Size) для TCP-соединений по этому маршруту
<b>Window</b>	Размер окна по умолчанию для TCP-соединений по этому маршруту
<b>irtt</b>	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 <b>irtt</b> — это начальное время <b>rtt</b> . 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 <b>irtt</b> можно увеличить командой <b>route</b> . По умолчанию его значение — 0

Добавить маршрут в таблицу маршрутизации можно статически (с помощью команды `route`), динамически или комбинированно (так, 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, рассматриваемыми 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/sysconfig/network/routes`

```
#  
# /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/c.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`), присваиваетсяся 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 час-

тичное 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 такой задан в таблице маршрутизации. Как уже отмечалось ранее, если шлюза по умолчанию нет, отправителю пакета посыпается 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 не обязательно — 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-подобных 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 С, 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 gate1
```

Добавляет маршрут по умолчанию. Все пакеты будут отправлены компьютеру с именем gate1. 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. 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-*interface*, где *interface* — 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/Networking\\_Guide/sec-Configuring\\_Static\\_Routes\\_in\\_ifcfg\\_files.html](https://docs.fedoraproject.org/en-US/Fedora/22/html/Networking_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 IPv4-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 PROC**

Подробно о настройке системы с помощью псевдофайловой системы proc рассказано в главе 24.

## **31.4. Настройка брандмауэра**

*Брандмаэр* (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 брандмаэр анализирует его и затем принимает решение: принять пакет (ACCEPT) или избавиться от него (DROP). Брандмаэр может выполнять и более сложные действия, но часто при его настройке ограничиваются именно этими двумя.

Прежде чем брандмаэр 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. Брандмаузру 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**

Кроме брандмауэра `iptables`, вы можете использовать брандмауз `UFW`, который рассмотрен в главе 43. Он проще в использовании, но не такой гибкий, как `iptables`.

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 приведена 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 веб-сервер), то пакет сначала обрабатывается правилами цепочки `INPUT` таблиц `mangle` и `filter`. Если пакет не был отброшен, он передается приложе-

нию (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.

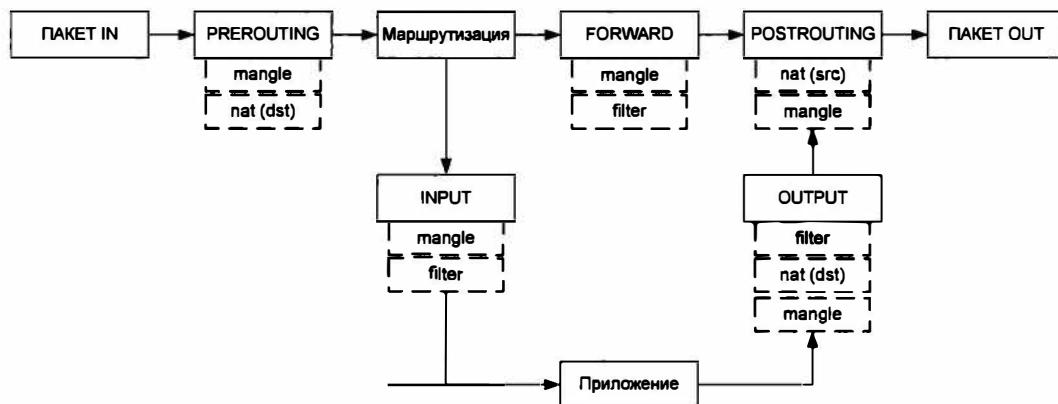


Рис. 31.2. Схема обработки пакета

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 OUT) и отправляется в сеть.

### Несколько слов о брандмауэре nftables

Прежде чем мы перейдем к изучению брандмауэра iptables, нужно сказать пару слов об 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 20.10 — в ней старый добрый iptables, а когда вводишь команду nft (основная команда nftables), система предлагает сначала установить пакет nftables (рис. 31.3). Почему разработчики дистрибутивов не спешат переходить на nft — не знаю. Ранее я полагал, что переход на nft состоится вместе с переходом на более новую версию ядра, но ядро уже 5.0, а воз и ныне там...

## 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 (см. разд. 31.2).

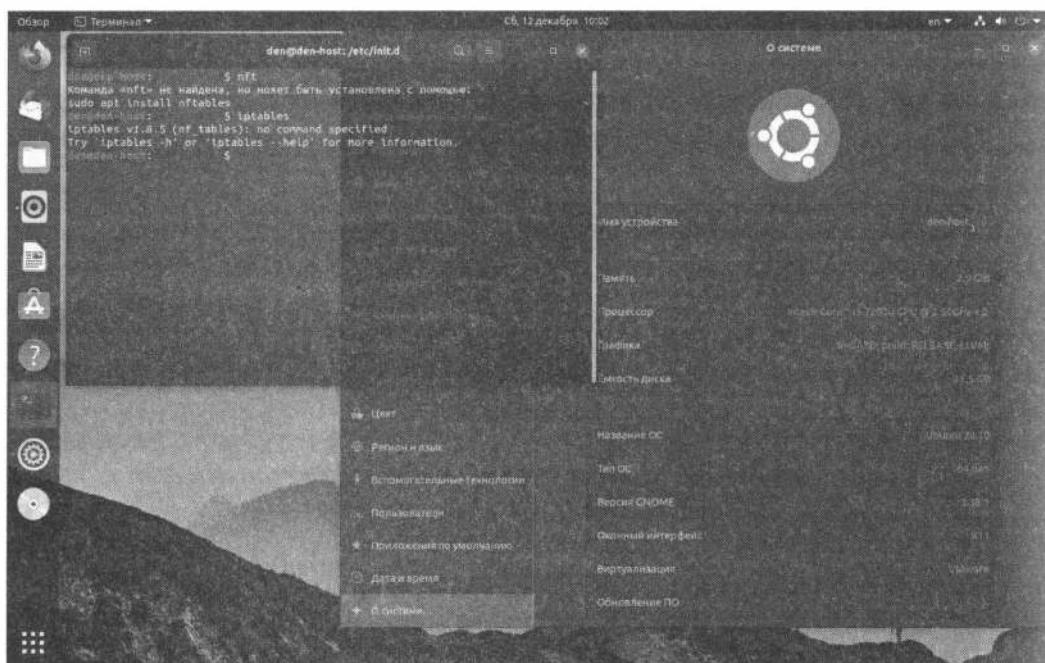


Рис. 31.3. Версия Ubuntu 20.10: nft — нет, а iptables — есть...

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 неустановленное 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 ( <code>den.dkws.org.ua</code> ), так и его IP-адрес ( <code>192.156.1.1</code> ) и даже набор адресов ( <code>192.168.1.0/255.255.255.0</code> ). Пример: <code>iptables -A FORWARD --source 192.168.1.11 ...</code>
--destination	Задает назначение (адрес получателя) пакета. Синтаксис такой же, как и у <code>--source</code>
-protocol (или -p)	Задает протокол. Чаще всего работают с <code>tcp</code> , <code>icmp</code> или <code>udp</code> , но можно указать любой протокол, определенный в файле <code>/etc/protocols</code> . Также можно указать <code>all</code> , что означает все протоколы. Примеры: <code>iptables -A FORWARD -protocol tcp ...</code> <code>iptables -A FORWARD -p tcp ...</code>
--source-port (или --sport)	Определяет порт отправителя. Эта опция может использоваться только вместе с параметром <code>-p</code> . Например: <code>iptables -A FORWARD -p tcp -source-port 23 ...</code>
--destination-port (или --dport)	Задает порт-назначение. Опция возможна только с параметром <code>-p</code> . Синтаксис такой же, как и в случае с <code>-source-port</code>
-state	Позволяет отфильтровать пакеты по состоянию. Параметр <code>-state</code> доступен только при загрузке модуля <code>state</code> с помощью другого параметра <code>-m state</code> . Состояния пакета: <ul style="list-style-type: none"> <li>• <code>NEW</code> — новое соединение (еще неустановленное);</li> <li>• <code>ESTABLISHED</code> — установленное соединение;</li> <li>• <code>RELATED</code> — пакеты, которые не принадлежат соединению, но связаны с ним;</li> <li>• <code>INVALID</code> — неопознанные пакеты.</li> </ul> Пример: <code>iptables -A FORWARD -m state -state RELATED,INVALID</code>

Таблица 31.3 (окончание)

Параметр	Описание
-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 (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
REJECT	<p>Уничтожает пакет и сообщает об этом отправителю с помощью ICMP-сообщения. Параметр <i>-reject-with</i> позволяет уточнить тип ICMP-сообщения:</p> <ul style="list-style-type: none"> <li>• <i>icmp-host-unreachable</i> — узел недоступен;</li> <li>• <i>icmp-net-unreachable</i> — сеть недоступна;</li> <li>• <i>icmp-port-unreachable</i> — порт недоступен;</li> <li>• <i>icmp-proto-unreachable</i> — протокол недоступен.</li> </ul> <p>По умолчанию это действие отправляет сообщение о недоступности порта. При этом, используя сообщение <i>icmp-host-unreachable</i>, можно сбить с толку атакующего вас злоумышленника. Предположим, что вы просто решили отбрасывать неугодные вам пакеты (действие <i>DROP</i>). Но злоумышленник будет посыпать и посыпать вам эти пакеты, чтобы брандмауэр только и делал, что занимался бы фильтрацией и удалением этих пакетов (один из видов атак на отказ). А если вы ответите сообщением <i>icmp-host-unreachable</i>, 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 <i>RETURN</i>
SNAT	Выполняет подмену IP-адреса отправителя (Source NAT). Используется в цепочках POSTROUTING и OUTPUT таблицы nat

Таблица 31.4 (окончание)

Действие	Описание
DNAT	Выполняет подмену адреса получателя (Destination NAT). Используется только в цепочке POSTROUTING таблицы nat
MASQUERADE	Похож на SNAT, 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 POSTROUTING таблицы nat

### 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.

Будем считать, что ваше подключение к Интернету организовано через DSL-соединение — так вам будет проще разобраться, где и какой интерфейс. Ведь при подключении по DSL используется протокол PPP, поэтому имя интерфейса для него — ppp0, а при подключении по Ethernet название внешнего интерфейса окажется таким же, как и внутреннего, — ens\* (разница только в цифрах после ens), и поэтому новичок может просто запутаться.

Так что нет разницы, по какому сетевому интерфейсу осуществляется ваше подключение к Интернету, — будет отличаться только имя интерфейса, а вся прочая настройка выполняется точно так же.

Вполне может быть, что у вас иная конфигурация сети. Например, ваш компьютер может иметь два сетевых интерфейса: ens33 (eth0) и ens38 (eth1). Первый «смотрит» в локальную сеть, а второй — подключен к Интернету. Тогда и правила вы должны формировать исходя из того, что соединение с Интернетом происходит по интерфейсу eth1. Далее мы будем использовать сетевые имена eth0 и eth1 — так привычнее и проще<sup>1</sup>.

При DSL-соединении у вас 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

<sup>1</sup> Переключить современную версию Linux на привычные имена сетевых интерфейсов можно с помощью передачи ядру параметра: net.ifnames=0.

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, включите IP-переадресацию. 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 весь этот раздел 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 на-

писания 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 вызывать 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 непrivилегированных 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
# биллинговую 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 $UPORTS -j ACCEPT
$IPT -A OUTPUT -p tcp -m tcp -o $INET --dport 53 --sport $UPORTS -j ACCEPT
$IPT -A INPUT -p udp -m udp -i $INET --dport $UPORTS --sport 53 -j ACCEPT
$IPT -A INPUT -p tcp -m tcp -i $INET --dport $UPORTS --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 $UPORTS -j ACCEPT
$IPT -A INPUT -p tcp -m tcp -i $INET --dport $UPORTS --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 $UPORTS -j ACCEPT
$IPT -A INPUT -p tcp -m tcp -i $INET --dport $UPORTS --sport 21 -j ACCEPT ! --
syn

# SSH-клиент (порт 22)
$IPT -A OUTPUT -p tcp -m tcp -o $INET --dport 22 --sport $UPORTS -j ACCEPT
$IPT -A INPUT -p tcp -m tcp -i $INET --dport $UPORTS --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 $UPORTS -j ACCEPT
$IPT -A INPUT -p tcp -m tcp -i $INET --dport $UPORTS --sport 25 -j ACCEPT ! --syn

# HTTP/HTTPS-клиент (порты 80, 443)
$IPT -A OUTPUT -p tcp -m tcp -m multiport -o $INET --sport $UPORTS -j ACCEPT --
dports 80,443
$IPT -A INPUT -p tcp -m tcp -m multiport -i $INET --dport $UPORTS -j ACCEPT --
sports 80,443 ! --syn

# POP-клиент (порт 110)
$IPT -A OUTPUT -p tcp -m tcp -o $INET --dport 110 --sport $UPORTS -j ACCEPT
$IPT -A INPUT -p tcp -m tcp -i $INET --dport $UPORTS --sport 110 -j ACCEPT ! --
syn

# Разрешаем прохождение DHCP-запросов через iptables
# Необходимо, если IP-адрес динамический
$IPT -A OUTPUT -p udp -m udp -o $INET --dport 67 --sport 68 -j ACCEPT
$IPT -A INPUT -p udp -m udp -i $INET --dport 68 --sport 67 -j ACCEPT
```

Вот практически и всё... Конечно, приведенное здесь описание iptables нельзя назвать полным. Но для полного описания iptables пришлось бы создать отдельную книгу под названием «Брандмауэр в Linux».

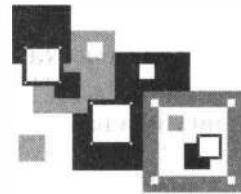
#### **ПОЛНОЕ РУКОВОДСТВО ПО IPTABLES**

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, следо 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](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 впрямую 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 и 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	Задает список шифров, разделенных 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 удаленном компьютере. Опцию использовать не обязательно, поскольку удаленный компьютер и так запросит имя пользователя и пароль
-pпорт	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. Полезна 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 ее

## 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/sshd\_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 $  
  
# Задает 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 задает 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  
# Key_regenerationInterval 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 no
# Разрешить TCP-форвардинг
# AllowTcpForwarding yes

# Использовать порты шлюза
# GatewayPorts no

# 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 no) 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).

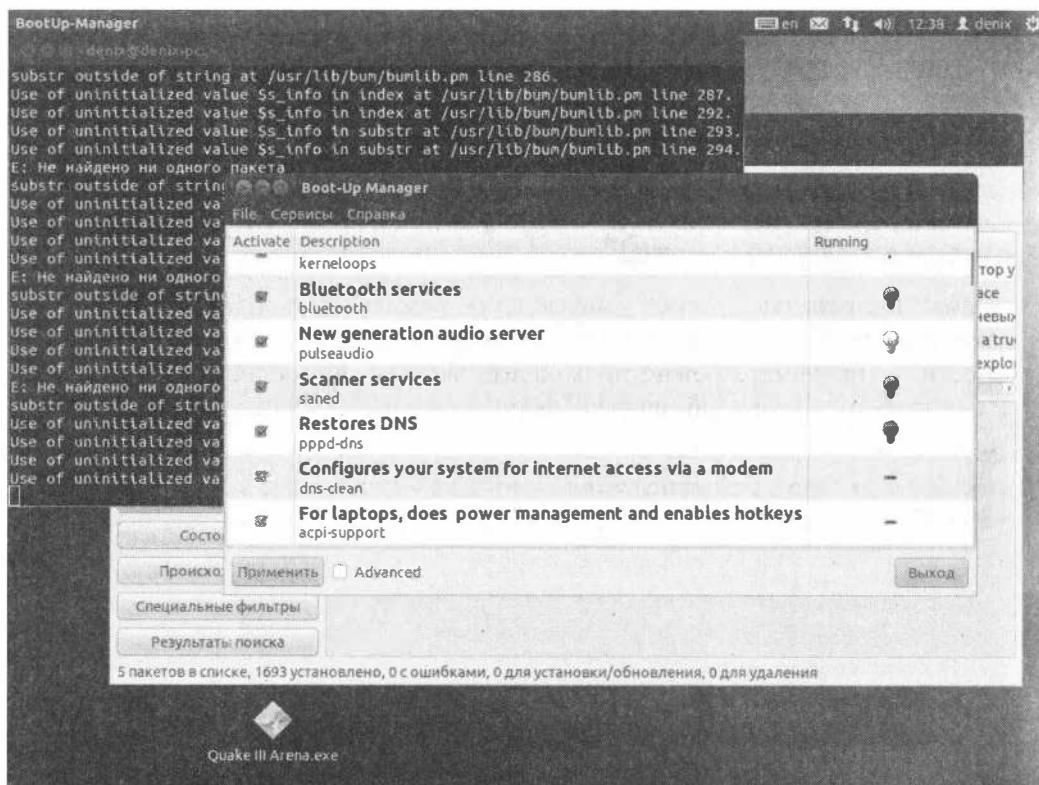
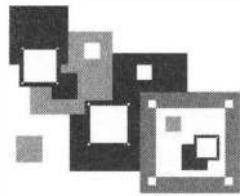


Рис. 32.1. Ubuntu: конфигуратор служб bum

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

# Веб-сервер. Связка Apache + PHP + MySQL

### 33.1. Самый популярный веб-сервер

Apache — это веб-сервер с открытым исходным кодом. История его развития началась в 1995 году — тогда Apache был всего лишь «заплаткой», устраняющей ошибки популярного в то время веб-сервера NCSA HTTPd 1.3. Считается, что отсюда произошло и название Apache (a patchy — заплатка). Сейчас Apache — самый популярный веб-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 веб-ориентированного языка программирования.

Но есть у Apache и недостатки — без этого никак, у любой медали всегда есть оборотная сторона. Основной недостаток — отсутствие удобного графического интерфейса администратора. Да, настройка Apache осуществляется путем редактирования его конфигурационного файла. В Интернете можно найти простые конфигураторы Apache, но их возможностей явно не хватает для настройки всех функций этого веб-сервера.

### 33.2. Установка веб-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. В большинстве современных дистрибутивов нужный нам пакет называется apache2, в некоторых устаревших он может называться httpd. Для установки Apache введите команду:

```
sudo apt install apache2
```

```
[sudo] пароль для den:
Сум:1 http://ua.archive.ubuntu.com/ubuntu groovy InRelease
Пол:2 http://ua.archive.ubuntu.com/ubuntu groovy-updates InRelease [188 kB]
Сум:3 http://security.ubuntu.com/ubuntu groovy-security InRelease
Сум:4 http://ua.archive.ubuntu.com/ubuntu groovy-backports InRelease
Пол:5 http://ua.archive.ubuntu.com/ubuntu groovy-updates/main amd64 DEP-11 Metadata [3.424 kB]
Пол:6 http://ua.archive.ubuntu.com/ubuntu groovy-updates/universe amd64 DEP-11 Metadata [1.804 kB]
Получено 113 kB за 1с (143 kB/s)
Чтение списков пакетов... Готово
Построение дерева зависимостей
Чтение информации о состоянии... Готово
Может быть обновлено 38 пакетов. Запустите «apt list --upgradable» для их показа.
den@den-host: $ sudo apt install apache2
Чтение списков пакетов... Готово
Построение дерева зависимостей
Чтение информации о состоянии... Готово
Будут установлены следующие дополнительные пакеты:
 apache2-bin apache2-data apache2-utils libapr1 libaprutil1 libaprutil1-dbd-sqlite3 libaprutil1-ldap
 liblua5.2-0
Предлагаемые пакеты:
 apache2-doc apache2-suexec-pristine | apache2-suexec-custom
Следующие НОВЫЕ пакеты будут установлены:
 apache2 apache2-bin apache2-data apache2-utils libapr1 libaprutil1 libaprutil1-dbd-sqlite3 libaprutil1-ldap
 liblua5.2-0
Обновлено 0 пакетов, установлено 9 новых пакетов, для удаления отмечено 0 пакетов, и 38 пакетов не обновлено.
Необходимо скачать 1.843 kB архивов.
После данной операции объем занятого дискового пространства возрастет на 8.012 kB.
Хотите продолжить? [Д/Н]
```

Рис. 33.1. Ubuntu: дополнительные пакеты для установки Apache

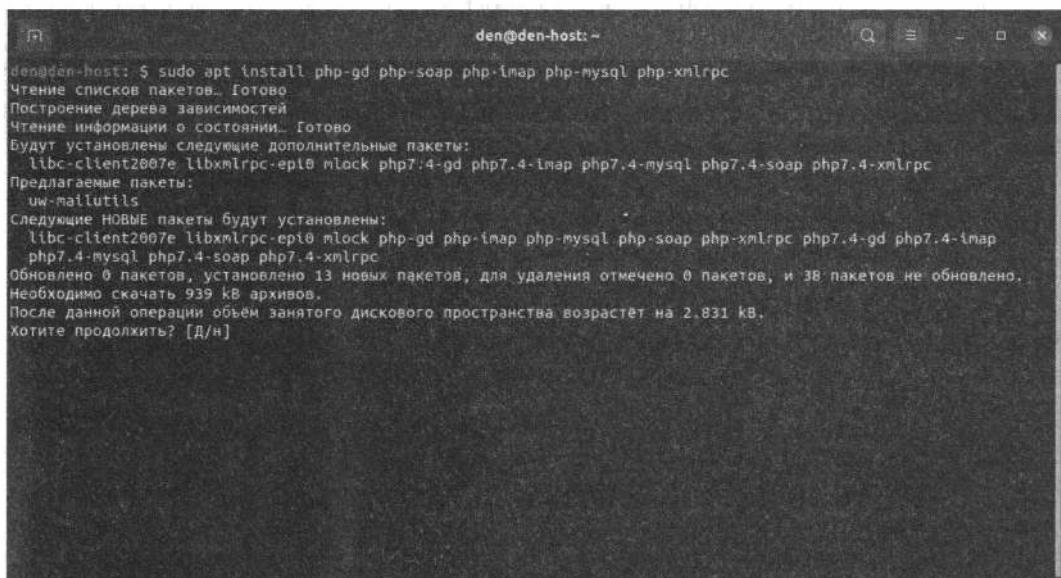
Менеджер пакетов сообщит вам, что нужно установить дополнительные пакеты (рис. 33.1).

Чтобы сразу «убить двух зайцев», выберите еще и пакет libapache2-mod-php — он устанавливает PHP7 и добавляет его поддержку в Apache. Менеджер снова предложит вам установить дополнительные пакеты, но теперь для PHP (рис. 33.2).

```
den@den-host: $ sudo apt install libapache2-mod-php
Чтение списков пакетов... Готово
Построение дерева зависимостей
Чтение информации о состоянии... Готово
Будут установлены следующие дополнительные пакеты:
 libapache2-mod-php7.4 php-common php7.4-cli php7.4-common php7.4-json php7.4-opcache php7.4-readline
Предлагаемые пакеты:
 php-pear
Следующие НОВЫЕ пакеты будут установлены:
 libapache2-mod-php libapache2-mod-php7.4 php-common php7.4-cli php7.4-common php7.4-json php7.4-opcache
 php7.4-readline
Обновлено 0 пакетов, установлено 8 новых пакетов, для удаления отмечено 0 пакетов, и 38 пакетов не обновлено.
Необходимо скачать 4.020 kB архивов.
После данной операции объем занятого дискового пространства возрастет на 17,9 МВ.
Хотите продолжить? [Д/Н]
```

Рис. 33.2. Ubuntu: дополнительные пакеты для установки PHP 7

В зависимости от требуемой конфигурации возможно понадобится установить также пакеты `php-imap`, `php-gd`, `php-mysql`, `php-xmlrpc`, обеспечивающие поддержку, соответственно, протоколов IMAP/POP, графической библиотеки GD, СУБД MySQL и XML-RPC (рис. 33.3). 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).



```
den@den-host: ~
den@den-host: $ sudo apt install php-gd php-soap php-imap php-mysql php-xmlrpc
Чтение списков пакетов... Готово
Построение дерева зависимостей
Чтение информации о состояниях... Готово
Будут установлены следующие дополнительные пакеты:
  libc-client2007e libxmlrpc-epi0 mlock php7.4-gd php7.4-imap php7.4-mysql php7.4-soap php7.4-xmlrpc
Предлагаемые пакеты:
  uw-mailutils
Следующие НОВЫЕ пакеты будут установлены:
  libc-client2007e libxmlrpc-epi0 mlock php7.4-gd php7.4-imap php7.4-mysql php7.4-soap php7.4-xmlrpc php7.4-gd php7.4-imap
Обновлено 0 пакетов, установлено 13 новых пакетов, для удаления отмечено 0 пакетов, и 38 пакетов не обновлено.
Необходимо скачать 939 kB архивов.
После данной операции объём занятого дискового пространства возрастёт на 2.831 kB.
Хотите продолжить? [Д/Н]
```

Рис. 33.3. Ubuntu: установка дополнительных библиотек PHP

Мало PHP установить, его нужно еще и настроить. Конфигурация PHP хранится в файле `php.ini`, вот только в системе присутствуют три файла `php.ini`. Каждый из них содержит конфигурацию, подходящую для того или иного случая использования:

- `/etc/php/<номер версии>/apache2/php.ini` — конфигурация скриптов, запущенных посредством веб-сервера;
- `/etc/php/<номер версии>/cli/php.ini` — конфигурация скриптов, запускаемых из консоли (когда вы вводите команду `php <имя скрипта.php>`);
- `/etc/php/<номер версии>/fpm/php.ini` — конфигурация для FPM, т. е. когда PHP запускается через `nginx`.

Как минимум вам нужно изменить следующие директивы:

- `memory_limit` — лимит памяти, доступный скрипту. Для консольных скриптов (конфигурация `cli`) здесь установлено значение `-1`, т. е. память не ограничивается. Для веб-сервера (конфигурации `apache2` и `fpm`) по умолчанию задано значение `128MB`, но для сложных приложений, основанных на фреймворках `Zend`, `Symfony`, `Laravel`, этого мало. Нужно установить как минимум `512MB`, а то и больше, — все зависит от приложения;

- `max_execution_time` — максимальное время выполнения сценария. Для конфигурации `cli` оно не ограничивается (значение 0), для веб-сервера (конфигурация `apache2`) оно установлено, как правило, в 30 секунд, но лучше увеличить это значение до 120 секунд;
- `file_uploads`, `upload_max_filesize` — первая разрешает (On) или запрещает (Off) загрузку файлов на сервер, вторая — задает максимальный размер файла. Значение по умолчанию — 2 Мбайт, но на сегодняшний день этого очень мало, и нужно увеличить его хотя бы до 20 Мбайт (здесь следует руководствоваться здравым смыслом и функционалом приложения: если, например, приложение подразумевает загрузку видеофайлов, то и 20 Мбайт может оказаться недостаточно).

Если вы изменили конфигурацию для Apache, то, чтобы изменения вступили в силу, его нужно перезапустить. Если вы изменили конфигурацию PHP-FPM, то тоже надо выполнить перезапуск командой:

```
sudo systemctl restart php-fpm
```

Теперь о версии PHP. Версия PHP 5.x канула в Лету. Все современные приложения, которые могли работать в версии 5.6, уже давно переведены на версию PHP 7.x из соображений производительности — седьмая версия работает быстрее.

При установке PHP из стандартных репозиториев дистрибутивов устанавливается самая новая версия. Для Ubuntu 20.10 — это PHP 7.4. Но в некоторых случаях нам нужно понизить версию PHP — например, когда устанавливаемое веб-приложение не поддерживает его последнюю версию и требует версии ниже. Рассмотрим, как установить в Ubuntu 20.10 версию PHP 7.2:

1. Установите репозитарий Ondrej:

```
sudo add-apt-repository ppa:ondrej/php
```

2. Обновите список пакетов:

```
sudo apt update
```

3. Установите версию 7.2:

```
sudo apt install php7.2
```

4. Выберите версию 7.2 для использования по умолчанию (это позволяет не удалять версию 7.4, если она уже была установлена):

```
sudo update-alternatives --set php /usr/bin/php7.2
```

5. Отключаем версию 7.4 для Apache:

```
sudo a2dismod php7.4
```

6. Устанавливаем версию 7.2 для Apache:

```
sudo a2enmod php7.2
```

7. Перезапускаем веб-сервер:

```
sudo systemctl restart apache2
```

### 33.3. Тестирование настроек

Теперь протестируем наш веб-сервер. По идеи после установки сервер должен запуститься 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).

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. Для его редактирования надо ввести команду:

```
sudo mcedit /var/www/html/test.php
```

Если команда mcedit у вас недоступна, установите файловый менеджер mc с помощью команды:

```
sudo apt install mc
```

#### Листинг 33.1. Файл test.php

```
<?
phpinfo();
?>
```

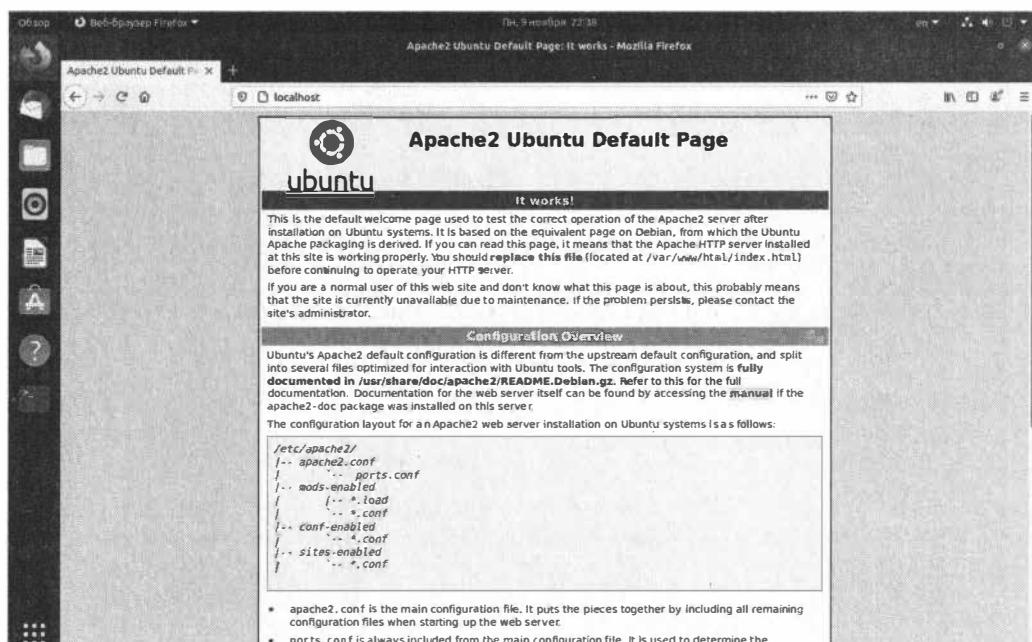


Рис. 33.4. Ubuntu: тестовая страница Apache

Создав файл, введите в строке браузера следующий адрес:

`http://localhost/test.php`

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.

The screenshot shows the Mozilla Firefox browser window with the title "PHP 7.4.9 - phpinfo()". The address bar shows "localhost/test.php". The main content is a table titled "PHP Version 7.4.9" containing detailed information about the PHP installation. Key sections include:

- System:** Linux den-host 5.8.0-26-generic #27-Ubuntu SMP Wed Oct 21 22:29:16 UTC 2020 x86\_64
- Build Date:** Oct 26 2020 15:17:14
- Server API:** Apache 2.0 Handler
- Virtual Directory Support:** disabled
- Configuration File (php.ini) Path:** /etc/php/7.4/apache2
- Loaded Configuration File:** /etc/php/7.4/apache2/php.ini
- Scan this dir for additional .ini files:** /etc/php/7.4/apache2/conf.d
- Additional .ini files parsed:** /etc/php/7.4/apache2/conf.d/00-aspische.ini, /etc/php/7.4/apache2/conf.d/00-pdo.ini, /etc/php/7.4/apache2/conf.d/20-calendar.ini, /etc/php/7.4/apache2/conf.d/20-crypt.ini, /etc/php/7.4/apache2/conf.d/20-exif.ini, /etc/php/7.4/apache2/conf.d/20-fm.ini, /etc/php/7.4/apache2/conf.d/20-mbstring.ini, /etc/php/7.4/apache2/conf.d/20-mysqli.ini, /etc/php/7.4/apache2/conf.d/20-openssl.ini, /etc/php/7.4/apache2/conf.d/20-phar.ini, /etc/php/7.4/apache2/conf.d/20-pspell.ini, /etc/php/7.4/apache2/conf.d/20-readline.ini, /etc/php/7.4/apache2/conf.d/20-shmop.ini, /etc/php/7.4/apache2/conf.d/20-sockets.ini, /etc/php/7.4/apache2/conf.d/20-sysvmsg.ini, /etc/php/7.4/apache2/conf.d/20-sysvshm.ini, /etc/php/7.4/apache2/conf.d/20-tokenizer.ini
- PHP API:** 20190902
- PHP Extension:** 20190902
- Zend Extension:** 320396902
- Zend Extension Build:** AP#20190902.NTS
- PHP Extension Build:** AP#20190902.NTS
- Debug Build:** no
- Thread Safety:** disabled
- Zend Signal Handling:** enabled
- Zend Memory Manager:** enabled
- Zend Multibyte Support:** disabled
- IPv6 Support:** enabled
- DTrace Support:** available, disabled
- Registered PHP Streams:** https, ftps, compress.zlib, zip, file, glob, data, http, ftp, phar
- Registered Stream Socket Transports:** tzc, std, unix, udg, col, fls, flsV1.0, flsV1.1, flsV1.2, flsV1.3

Рис. 33.5. Ubuntu: информация о веб-сервере и о PHP

```
den@den-host: /var/www/html
den@den-host: /var/www/html [----] 0 Lf 1+ 0 1/ 5 *(6 / 21b) 08:00 0x88C
<?php
phinfo();
?>
```

Рис. 33.6. Ubuntu: тестовый сценарий

**СОВЕТ**

Если вместо отображения тестовой страницы, показанной на рис. 33.4, браузер предложит вам сохранить файл `test.php`, перезапустите веб-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 веб-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 веб-сервера при этом называются `httpd.conf` и `httpd2.conf` или `apache.conf` и `apache2.conf`. Названия каталогов и файлов, содержащие фрагмент `apache`, характерны для дистрибутивов Debian и Ubuntu, а содержащие фрагмент `httpd` — для Fedora. В любом случае найти конфигурационные файлы не сложно: ищите или `apache`, или `httpd` — и не промахнетесь! Например, в Ubuntu 20.10 файл конфигурации находится по следующему пути: `/etc/apache2/apache2.conf`.

**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.name
```

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 сервер, 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
ServerName имя	Задает имя веб-сервера — оно должно быть зарегистрировано на DNS-сервере, т. е. обычно — это доменное имя сервера
ServerAdmin e-mail	Задает e-mail администратора сервера
ServerRoot каталог	Определяет каталог с конфигурационными файлами сервера
PidFile файл	Определяет имя файла, в котором будет храниться PID исходного процесса веб-сервера. Обычно изменять эту директиву не нужно
DocumentRoot каталог	Позволяет задать каталог, в котором хранятся документы веб-сервера, — это корневой каталог документов. Обычно это /var/www
StartServers N, MaxSpareServers N, MinSpareServers N, MaxClients N	Директивы, непосредственно влияющие на производительность сервера. Мы их рассмотрим отдельно в разд. 33.6
KeepAlive On   Off, KeepAliveTimeout N	Управляют постоянными соединениями (будут рассмотрены в разд. 33.6)
DirectoryIndex список	Задает имена файлов, которые могут использоваться в качестве главной страницы (индекса). Значение по умолчанию index.html index.cgi index.pl index.php index.xhtml
HostnameLookups On Off	Если директива включена (On), то IP-адрес клиента перед записью в журнал будет разрешен (т. е. веб-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
ErrorLog файл	Задает журнал ошибок
TransferLog 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 пользователь Group группа	Директивы User и Group задают имя пользователя и группы, от имени которых запускается веб-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 блочные 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) приведены 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 DirectoryIndex, 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 символьских 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.

### 33.4.4. Работа сервера на нескольких портах

Как правило, веб-сервер «слушает» порт 80. Но в некоторых сложных конфигурациях нужно, чтобы он слушал несколько портов. Рассмотрим, как разместить на одном сервере три независимых приложения, каждое из которых будет работать на собственном порту: 80, 3128 и 8080.

Первым делом нужно отредактировать файл `/etc/apache2/ports.conf` и задать порты, которые будет слушать сервер (по одному порту в строчке):

```
Listen 80
Listen 3128
Listen 8080
```

Затем надо создать конфигурацию для каждого из приложений (для каждого порта). Вы можете использовать разные файлы конфигурации, а можете поместить все настройки в один файл, — как вам больше нравится. В листинге 33.3 приведен код, где все настройки хранятся в одном большом файле (чтобы не делать три разных листинга).

#### Листинг 33.3. Файл `/etc/apache2/sites-available/000-default.conf`

```
<VirtualHost *:80>
    DocumentRoot      /var/www/landing

    <Directory /var/www/landing>
        Options -Indexes
        AllowOverride All

        # Apache 2.4.x
        <IfModule mod_authz_core.c>
            Require all granted
        </IfModule>
    </Directory>

    CustomLog   /var/log/apache2/landing.access.log "Combined"
    ErrorLog    /var/log/apache2/landing.error.log
</VirtualHost>

<VirtualHost *:3128>
    DocumentRoot      /var/www/sonerezh

    <Directory /var/www/sonerezh>
        Options -Indexes
        AllowOverride All

        # Apache 2.4.x
        <IfModule mod_authz_core.c>
```

```
        Require all granted
    </IfModule>
</Directory>

CustomLog /var/log/apache2/demo.sonerezh.bzh-access.log "Combined"
ErrorLog /var/log/apache2/demo.sonerezh.bzh-error.log
</VirtualHost>

<VirtualHost *:8080>
    DocumentRoot /var/www/sonerezh-ru

    <Directory /var/www/sonerezh-ru>
        Options -Indexes
        AllowOverride All

        # Apache 2.4.x
        <IfModule mod_authz_core.c>
            Require all granted
        </IfModule>
    </Directory>

    CustomLog /var/log/apache2/demo.sonerezh.ru-access.log "Combined"
    ErrorLog /var/log/apache2/demo.sonerezh.ru-error.log
</VirtualHost>
```

Принцип здесь следующий: мы указываем номер порта в директиве `VirtualHost`, а затем готовим разную конфигурацию для каждого узла.

### 33.4.5. Динамические поддомены

Часто для одного домена создаются поддомены — например: `sales.example.com`, `support.example.com` и т. д. Поддомены настраиваются путем использования директивы `VirtualHost`, но если поддоменов много, то вам быстро надоест создавать отдельный конфигурационный файл для каждого поддомена. Гораздо проще использовать следующую конфигурацию:

```
<VirtualHost *:80>
    ServerAdmin site@example.com
    ServerName example.com
    ServerAlias www.example.com
    DocumentRoot /var/www/example.com/htdocs

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>

<VirtualHost *:80>
    ServerAdmin site@example.com
```

```
ServerName example.com
ServerAlias *.example.com

UseCanonicalName Off
VirtualDocumentRoot /var/www/example.com/%-3

ErrorLog ${APACHE_LOG_DIR}/error.log
CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

Все поддомены будут запрашиваться из подкаталогов каталога `/var/www/example.com`. Для работы приведенной конфигурации нужно, чтобы `mod_rewrite` был установлен и активирован. Также надо, чтобы DNS-сервер знал о поддомене. Другими словами, мало создать папку `test` в `/var/www/example.com` — нужно, чтобы DNS-сервер знал, где искать поддомен `test.example.com`.

## 33.5. Управление запуском сервера Apache

Для управления веб-сервером можно использовать команду `service`:

- `sudo systemctl start apache2` — запуск сервера;
- `sudo systemctl stop apache2` — останов сервера;
- `sudo systemctl restart apache2` — перезапуск сервера.

Понятно, что веб-сервер запускается автоматически, поэтому каждый день вам не придется вводить команду `service httpd start`.

Как уже отмечалось ранее, в разных дистрибутивах служба веб-сервера называется по-разному: `httpd`, `httpd2`, `apache` или `apache2`. Поэтому, если система сообщила, что в ней нет вызванного вами сервиса, попробуйте просто использовать другое название.

В старых версиях Linux вместо команды `systemctl` нужно использовать команду `service`:

```
sudo service apache2 start
sudo service apache2 stop
sudo service apache2 restart
```

## 33.6. Оптимизация Apache

В конфигурационном файле сервера Apache `apache2.conf`, находящемся в каталоге `/etc/apache2` 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 веб-сервера, нужно понимать, как он работает: клиент посылает запрос, веб-сервер его обрабатывает и посыпает клиенту ответ. После этого соединение можно закрывать и завершать копию Apache, обслуживающую это соединение. Но зачем завершать копию веб-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`, т. е. 10% от `MaxClients`.
- Вы не задумывались, что если веб-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?

нение? 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 веб-сервера.

## 33.7. Пользовательские каталоги

Если вы когда-нибудь настраивали сервер Apache, то наверняка знакомы с директивой `UserDir`. Я специально ее не показал 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 `/home/<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`<sup>1</sup>. 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.

---

<sup>1</sup> Не забывайте указывать свой путь к файлу конфигурации, поскольку каталог с файлами конфигурации может отличаться в зависимости от версии дистрибутива и самого Apache.

## 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 этого установите следующие пакеты:

- mysql-server;
- mysql-client.

Первый пакет содержит последнюю версию MySQL-сервера (на текущий 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.

### 33.8.2. Изменение пароля root и добавление пользователей

Для базовой настройки введите команду:

```
sudo mysql_secure_installation
```

Программа задаст ряд вопросов (рис. 33.7), на которые нужно ответить нажатием клавиш <Y> (или <y>), если вы хотите ответить утвердительно, или же нажать

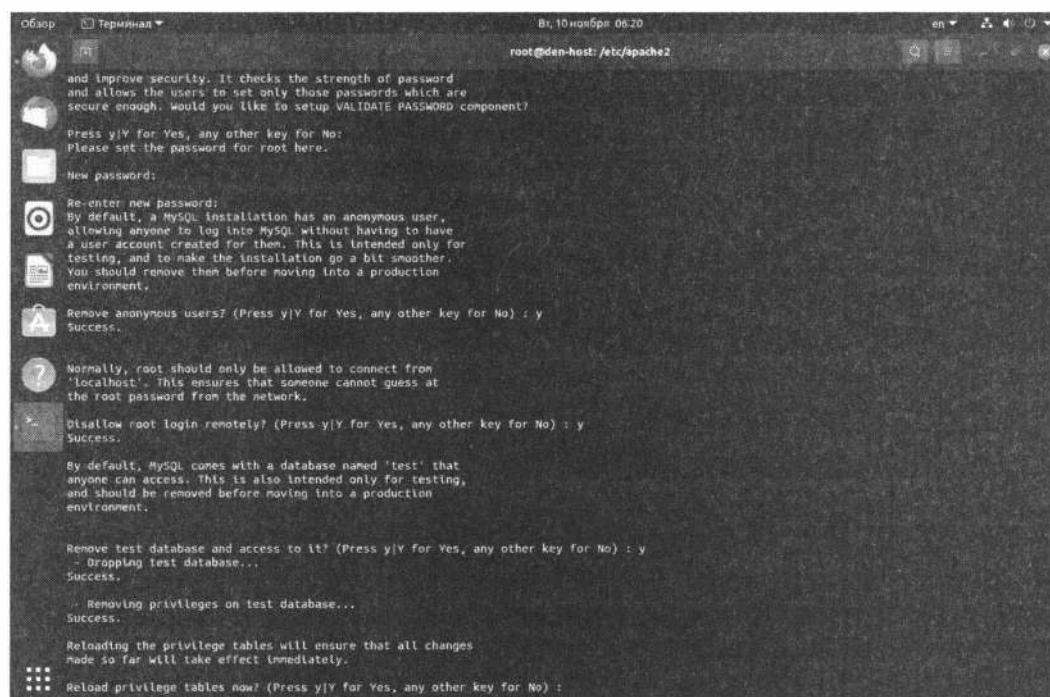


Рис. 33.7. Ubuntu: базовая настройка MySQL

любую другую клавишу, если вы хотите отказаться от предлагаемой возможности. Итак, вот варианты этих вопросов:

- Connecting to MySQL using a blank password** — можно ли подключаться к базе данных с пустым паролем. Нужно нажать любую клавишу, чтобы запретить подключение с пустым паролем;
- Please set the password for root here** — нужно ввести пароль пользователя root базы данных (не путать с системным пользователем) и его подтверждение. Пароль должен состоять из букв разного регистра и цифр. Минимальная длина — 8 символов;
- Remove anonymous users** — удалить анонимных пользователей. Нажмите клавишу <Y>;
- Disallow root login remotely** — отключает вход пользователя root удаленно. Нажмите клавишу <Y>;
- Remove test database and access to it** — удаляет тестовую базу данных и доступы к ней. Нажмите клавишу <Y>;
- Reload privilege tables now** — нажмите клавишу <Y>, чтобы перезагрузить таблицы привилегий прямо сейчас.

Для обычной работы с сервером рекомендуется создать обычного пользователя. Для этого введите команду:

```
mysql -u root -p
```

Программа mysql представляет собой клиент MySQL-сервера. В указанном случае она должна подключиться к базе данных mysql (служебная база данных), используя имя пользователя root (-u root). Поскольку мы ранее задали пароль для пользователя root, то его и нужно указать при запуске программы, на что указывает параметр -p (требовать пароль). При этом если его не указать, а пароль для пользователя задан, то войти на сервер не получится. Далее вы увидите приглашение:

```
mysql>
```

Настало время вводить SQL-запросы:

```
CREATE DATABASE mydb;
CREATE USER myuser@localhost IDENTIFIED BY 'Secret2020';
GRANT ALL ON mydb.* TO myuser@localhost;
ALTER USER myuser@localhost WITH MAX_QUERIES_PER_HOUR 100;
FLUSH PRIVILEGES;
exit
```

Первый MySQL-запрос создает базу данных mydb. Название, разумеется, вы можете изменить. Второй запрос создает пользователя myuser@localhost и задает его пароль. Пароль специально здесь приведен так, чтобы вы видели, что он должен содержать буквы в разных регистрах и цифры. Интересно, что в MySQL 8 вы можете при создании пользователя задать слабый пароль, но пользователь не сможет подключиться к серверу с использованием слабого пароля.

Третий запрос — это предоставление всех полномочий пользователю `myuser` в базе данных `mydb`. Обратите внимание: в MySQL 8 синтаксис этой команды изменился — ранее команда предоставления всех полномочий выглядела иначе. Также заметьте, что мы предоставляем пользователю все полномочия только к данной базе данных, а не ко всем базам.

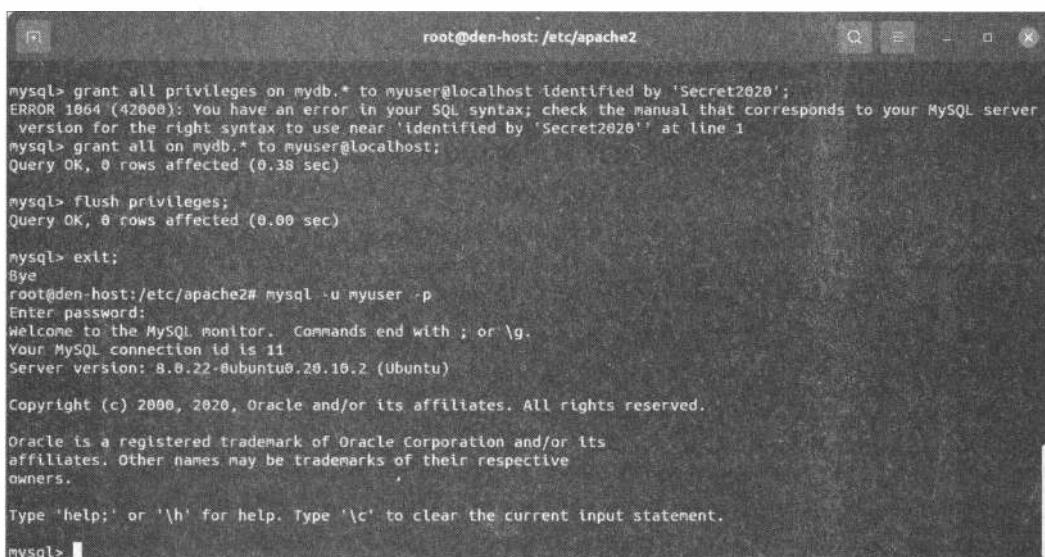
Следующий запрос ограничивает количество запросов в час. За час наш пользователь сможет сделать не более 100 запросов. В реальной жизни это очень мало, но принцип вам понятен: с помощью такого запроса вы можете регулировать этот параметр.

Далее нам нужно обновить полномочия, чтобы произведенные изменения вступили в силу. Последняя команда производит выход из MySQL-клиента.

Теперь введите команду:

```
mysql -u myuser -p
```

Если вы все сделали правильно, то должны увидеть приглашение MySQL, но уже для пользователя `myuser` (рис. 33.8).



```
root@den-host:/etc/apache2# mysql -u myuser -p
mysql> grant all privileges on mydb.* to myuser@localhost identified by 'Secret2020';
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server
version for the right syntax to use near 'identified by 'Secret2020'' at line 1
mysql> grant all on mydb.* to myuser@localhost;
Query OK, 0 rows affected (0.38 sec)

mysql> flush privileges;
Query OK, 0 rows affected (0.00 sec)

mysql> exit;
Bye
root@den-host:/etc/apache2# mysql -u myuser -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 11
Server version: 8.0.22-0ubuntu0.20.10.2 (Ubuntu)

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

Рис. 33.8. Вход на сервер с помощью созданного пользователя

### 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 стоять точка с запятой!

Кроме программы `mysql` в состав MySQL-клиента входит одна очень полезная программа — `mysqlshow`, которая может вывести список баз данных и таблиц, находящихся в той или иной базе данных. Она также еще много чего может, но сейчас нам нужен пока список таблиц — чтобы вы знали, какие таблицы есть в базе данных:

```
mysqlshow
mysqlshow -p <база данных>
```

Первая команда выводит список баз данных, вторая — список таблиц в базе данных (рис. 33.9).

```
Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> exit;
Bye
root@den-host:/etc/apache2# mysqlshow
+-----+
|   Databases   |
+-----+
| information_schema |
| mydb           |
| mysql          |
| performance_schema |
| sys            |
+-----+
root@den-host:/etc/apache2# mysqlshow -p mydb
Enter password:
Database: mydb
+-----+
| Tables |
+-----+
+-----+
root@den-host:/etc/apache2#
```

Рис. 33.9. Команда mysqlshow

### 33.8.3. Запуск и останов сервера

Команды запуска, останова и перезапуска сервера MySQL выглядят так:

```
sudo systemctl start mysql
sudo systemctl stop mysql
sudo systemctl restart mysql
sudo systemctl status mysql
```

### 33.8.4. Программа phpMyAdmin

Самым популярным MySQL-клиентом является веб-клиент phpMyAdmin. Для его работы нужен веб-сервер Apache, но поскольку мы его уже установили, то не установить phpMyAdmin мы просто не можем. Бывают ситуации, когда нам не нужен веб-сервер, тогда придется искать какие-то другие решения, но, поскольку веб-сервер есть, можно смело устанавливать phpMyAdmin:

```
sudo apt install phpmyadmin
sudo systemctl restart apache2
```

В процессе установки (первая команда) нужно будет выбрать, какой веб-сервер у вас используется, чтобы программа произвела его настройку автоматически (рис. 33.10).

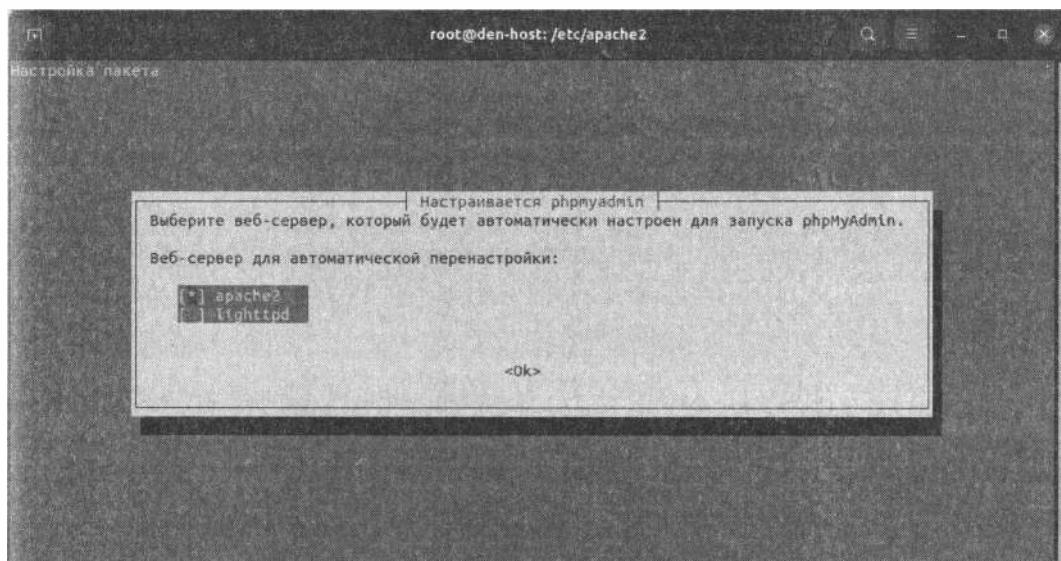


Рис. 33.10. Установка phpMyAdmin

Вторая команда перезапускает веб-сервер. После перезапуска веб-сервера откройте браузер и введите URL: <http://localhost/phpmyadmin>. Далее можно ввести данные нашего пользователя **myuser** — его логин и пароль для входа на сервер баз данных (рис. 33.11).

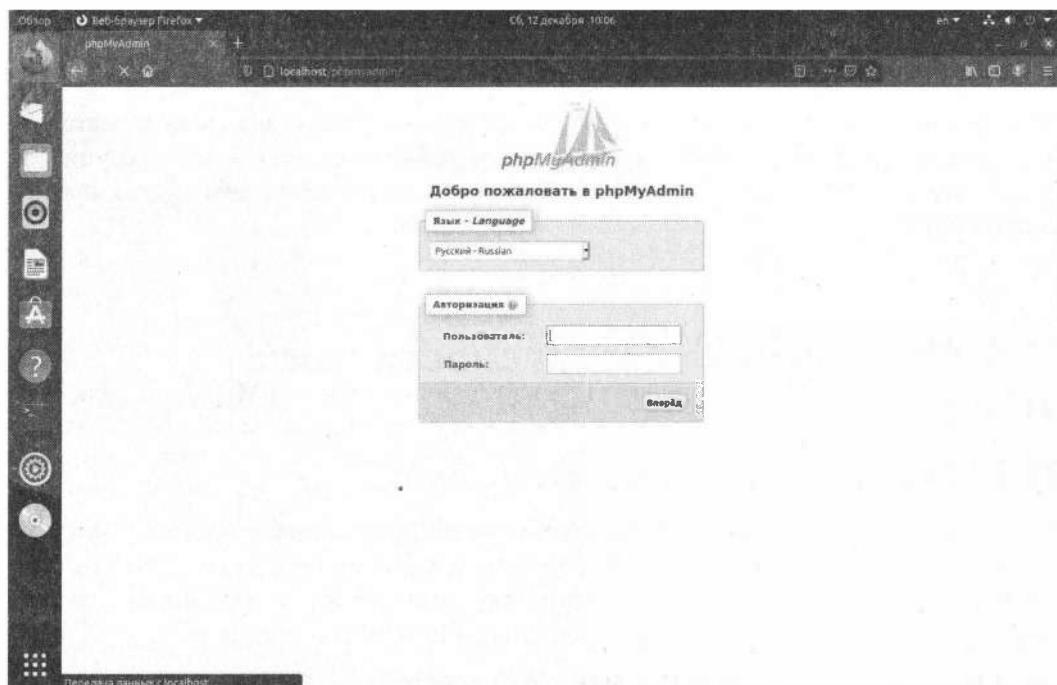


Рис. 33.11. Программа phpMyAdmin

Если вы установили не Apache, а сервер nginx, то процедура установки phpMyAdmin будет чуть сложнее. Дело в том, что phpMyAdmin не поддерживает этот сервер «из коробки», поэтому после установки приложения нужно перейти в каталог /etc/nginx/snippets и создать файл phpmyadmin.conf следующего содержания (листинг 33.4).

#### Листинг 33.4. Файл phpmyadmin.conf

```
location ~ ^/pma/(.*)$ {
    alias /usr/share/phpmyadmin/$1;

    location ~ ^/pma/(.+\.php)$ {
        alias /usr/share/phpmyadmin/$1;
        fastcgi_pass unix:/run/php/php7.2-fpm.sock;
        fastcgi_index index.php;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
        include fastcgi_params;
    }
    location ~* ^/pma/(.+\.(jpg|jpeg|gif|css|png|js|ico|html|xml|txt))$ {
        alias /usr/share/phpmyadmin/$1;
    }
}

location ~* ^/pma/(libraries|setup/lib|setup/frames) / {
    deny all;
}
}
```

Этот файл предполагает, что программа phpMyAdmin установлена в каталоге /usr/share/phpmyadmin (по умолчанию оно так и есть) и что она будет доступна по адресу <http://localhost/pma>. После создания файла phpmyadmin.conf нужно перезапустить nginx:

```
sudo systemctl restart nginx
```

## 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, 2019 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, разработан-

ные 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 ваш_ip
</Files>
```

Даже если у вас динамический IP-адрес, то проще его изменить в файле .htaccess, чем дождаться взлома вашего сайта.

Если жестко ограничить доступ невозможно, в качестве выхода из ситуации для защиты веб-, 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 изменениях файла `/etc/passwd`.

После установки антивируса нужно сразу же обновить антивирусные базы:

```
# freshclam
```

Не забудьте также добавить задание `cron` для обновления этих 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 файлового 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. 33.12 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.

```
/srv/www/      /htdocs/test/_Signedint.php: Php.Malware.Agent-1427575 FOUND  
----- SCAN SUMMARY -----  
Known viruses: 6298827  
Engine version: 0.99.2  
Scanned directories: 11400  
Scanned files: 176394  
Infected files: 1  
Data scanned: 3041.42 MB  
Data read: 6175.80 MB (ratio 0.49:1)  
Time: 241.932 sec (4 m 1 s)
```

Рис. 33.12. Отчет о проверке

### 33.9.4. Сценарий scanner

Для собственного использования я написал простенький bash-сценарий (листинг 33.5), который ищет файлы, содержащие подозрительные PHP-инструкции, а также файлы .html и .php, изменившиеся за последние 20 дней (рис. 33.13). Не обязательно, что все найденные файлы заражены вирусами, но такие файлы нужно держать на контроле, особенно те, которые вы не изменяли.

#### Листинг 33.5. 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. 33.13. Результат работы сценария scanner

## 33.10. SSL-сертификат для сайта

Наличие SSL-сертификата на вашем сайте — это не только статус компании и соответствие требованиям современных браузеров. Выбор сертификата имеет самое прямое отношение к безопасности вашей бизнес-инфраструктуры, а поэтому заслуживает надлежащего внимания.

Существуют различные типы SSL-сертификатов. Но какой из них выбрать? Попробуем разобраться далее.

### 33.10.1. Выбор SSL-сертификата

#### Основные типы сертификатов

Сертификаты различаются не только брендом и ценой. При покупке сертификата нужно обратить внимание на его тип, определяющий функционал сертификата.

Итак, существуют три основных типа сертификатов:

- DV (Domain Validation) — используется для подтверждения домена;
- OV (Organization Validation) — используется для подтверждения организации и домена;
- EV (Extended Validation) — служит для расширенного подтверждения организации и домена.

Обратите внимание, что сертификаты типов OV и EV также можно использовать для подтверждения домена.

Все указанные сертификаты обеспечивают шифрование трафика между сайтом и браузером. Другими словами, если нужно организовать шифрование трафика по протоколу HTTPS, можно выбрать любой тип сертификата.

У всех этих сертификатов есть дополнительные опции: WildCard и SAN. Первая подтверждает домен и все его поддомены следующего уровня. То есть можно ку-

пить сертификат DV с опцией WildCard для домена example.com и использовать его для подтверждения не только example.com, но и всех его поддоменов следующего уровня, т. е. sales.example.com, dev.example.com, docs.example.com и т. д.

Опция SAN подтверждает домены по списку, указанному при получении SSL-сертификата.

## Какой тип сертификата выбрать?

Ответ на этот вопрос зависит от поставленных задач. Если ставится задача «минимум», т. е. обеспечение защиты пользователей вашего сайта от появления у них предупреждений браузера о посещении непроверенного сайта, то достаточно обзавестись самым простым (и самым дешевым) SSL-сертификатом типа DV. Этого будет вполне достаточно, чтобы получить зеленый значок рядом с интернет-адресом сайта (URL) в любом современном браузере (рис. 33.14).



Рис. 33.14. Ubuntu: проверенный сайт

Если у вас есть поддомены вроде sales.example.com, не забудьте заказать опцию WildCard. Если у вашего сайта несколько адресов или у вас несколько сайтов (по отдельному сайту для каждого подразделения компании), то не забудьте указать опцию SAN, чтобы не покупать отдельный SSL-сертификат для каждого домена.

А вот если интернет-площадка используется для операций, требующих повышенного уровня сохранности данных компании и клиентов (например, операций с платежными картами), то лучше выбрать EV-сертификат.

## Особенности SSL-сертификатов разных типов

Различия между всеми современными типами SSL-сертификатов можно разделить на две группы: по методу проверки и по сертифицируемым доменам. Рассмотрим первую группу различий:

- сертификат с проверкой домена (Domain Validation, DV)* — подтверждает, что пользователь находится именно на том сайте, на который он осуществил пере-

ход, т. е. такой сертификат удостоверяет сервер, обслуживающий сайт. DV-сертификат не содержит информации о компании-владельце, поэтому не может считаться безопасным для оказания коммерческих услуг. Представим, что у вас есть интернет-магазин, предоставляющий возможности онлайн-покупки чего-либо. Если вы обрабатываете платеж самостоятельно, то DV-сертификат нельзя считать достаточно безопасным. Однако если при нажатии кнопки **Купить** (**Оплатить** или подобной) пользователь уходит на сайт системы, обрабатывающей платеж, — например, на Яндекс.Деньги, то DV-сертификата будет вполне достаточно, а EV-сертификат пусть покупает себе Яндекс. Вам же он не нужен. Все, что вам нужно, — это чтобы слева от адреса вашего сайта в браузере был зеленый значок, как показано на рис. 33.11;

- **сертификат с проверкой организации** (Organization Validation, OV) — позволяет подтвердить не только доменное имя, но и организацию-владельца веб-сайта. Подлинность организации проверяется по регистрационным данным юридического лица, которые пересылаются провайдеру SSL-сертификата при заказе OV-сертификата. Такие сертификаты наиболее популярны на сегодняшний день.
- **сертификат с расширенной проверкой** (Extended Validation, EV) — обладает самым высоким уровнем доверия со стороны других узлов. Если вам нужна строгая конфиденциальность передаваемых данных (например, обработка платежей именно на вашем сайте), то вам потребуется EV-сертификат.

EV-сертификаты заслуживают отдельного внимания, как самые дорогие. Такие сертификаты подразумевают расширенную периодическую проверку данных владельца сайта для предотвращения подмены этих данных. Так, если OV-сертификат считается действительным на протяжении всего своего срока и проверка организации осуществляется только при покупке сертификата, то в случае с EV-сертификатом такая проверка может осуществляться несколько раз (с определенной периодичностью) на протяжении всего срока действия сертификата.

Изначально все продаваемые SSL-сертификаты были типа OV, т. е. подразумевали проверку данных организации. Однако маркетинг сделал свое дело — и появились DV-сертификаты с упрощенной проверкой, когда проверяется только сам домен. Это сыграло на руку мошенникам — ведь при наличии такого сертификата браузер не будет предупреждать пользователя о возможных проблемах с подлинностью сайта. Отчасти виновником этой проблемы стали браузеры, требующие SSL-сертификатов. В погоне за зеленым значком в адресной строке многие стали покупать SSL-сертификаты, а спрос, как говорится, рождает предложение, и провайдеры SSL-сертификатов стали предлагать более дешевые DV-сертификаты.

EV-сертификат создан для решения этой проблемы. При использовании EV-сертификата в браузере появляется так называемая **зеленая панель** (green bar), в которой выводится не просто зеленый значок защиты, а название организации, которой выдан сертификат. На рис. 33.15 показано, как выглядит эта панель в браузере Firefox.

Если щелкнуть на названии организации, можно получить дополнительную информацию о сертификате: кому выдан, кем выдан (рис. 33.16). Если и этого мало,

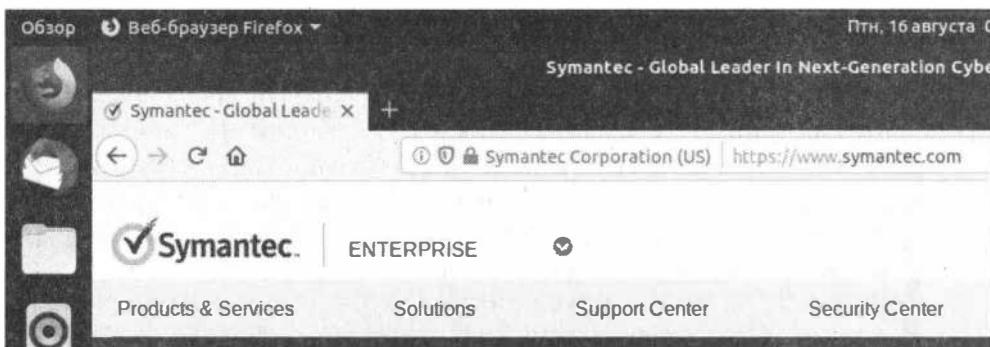


Рис. 33.15. Ubuntu: сведения об организации, которой выдан сертификат

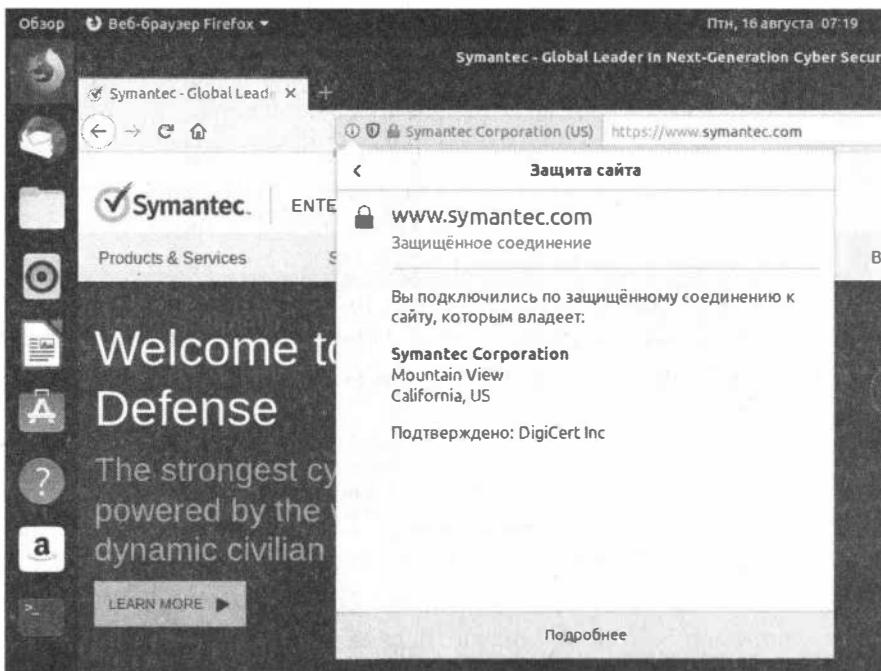


Рис. 33.16. Ubuntu: дополнительные сведения о сертификате

тогда нужно нажать кнопку **Подробнее**, а затем кнопку **Просмотреть сертификат** — это в браузере Firefox, а в браузере Chrome после щелчка на названии организации следует нажать ссылку **Действительный**. На рис. 33.17 приведена информация о сертификате — в частности, видно, что это расширенный сертификат.

Опция **green bar** не предоставляется для DV- и OV-сертификатов. Поэтому если вы видите не просто зеленый значок защиты, а название компании, которой принадлежит сайт, то можете быть уверенными, что компания использует EV-сертификат, и ей можно доверять. По крайней мере, ваши данные не будут украдены третьей стороной, сама компания не считается...

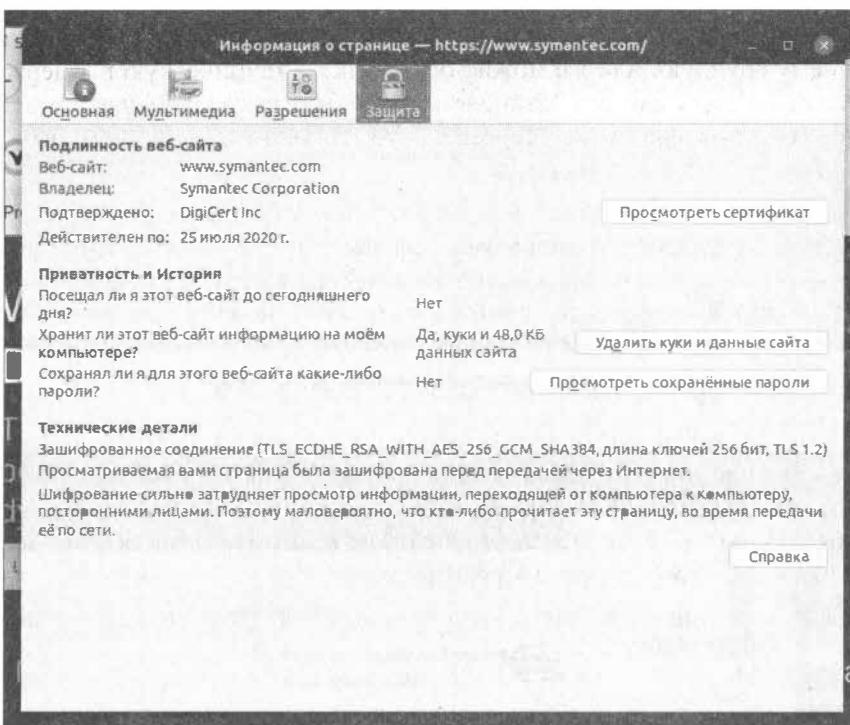


Рис. 33.17. Ubuntu: расширенные сведения о сертификате

Вторая группа различий — по сертифицируемым доменам. Здесь возможны следующие варианты:

- сертификат на один домен (Single Certificate)** — работает только для того доменного имени, которое было указано при заказе сертификата. Если вы купили такой сертификат для своего домена example.com, то он уже не будет действителен для его поддоменов типа sales.example.com;
- WildCard SSL** — используется не только для домена, но и всех его поддоменов, — например: sales.example.com, docs.example.com и т. д.;
- SAN (Subject Alternative Name) или UC (Unified Communications)** — такие сертификаты могут использоваться сразу для нескольких доменов.

## Где купить SSL-сертификат?

Правильнее всего, казалось бы, купить сертификат непосредственно у центра сертификации. Но не все так просто. Многие центры сертификации не работают непосредственно с клиентами, а только с посредниками (реселлерами). Дело в том, что реселлер берет на себя основную работу с клиентом — например, собирает все необходимые документы, а также запрашивает дополнительную информацию, которая может понадобиться в процессе сертификации.

Реселлером, как правило, является ваш хостинг-провайдер. Другими словами, сертификат можно купить у организации, которая предоставляет вам хостинг или вир-

туальный сервер. Если у вас виртуальный сервер, то сертификат придется «прикручивать» к нему вручную, или же можно обратиться в техническую поддержку — за дополнительную плату вам его установят. Далее будет показано, как самостоятельно установить сертификат на сервере, что сэкономит вам немного средств (см. разд. «*Настройка веб-сервера*»).

Сэкономить можно еще больше, если использовать бесплатный сертификат Let's Encrypt. В разд. 33.10.3 будет показано, как его сгенерировать. Недостаток этого способа в том, что каждые три месяца нужно будет этот сертификат обновлять. Впрочем, планировщик crontab отлично справляется с такой задачей.

### 33.10.2. Конвертирование сертификатов

Весьма часто для успешной установки SSL-сертификатов на разных plataформах и устройствах требуется преобразовать их в другие форматы. Так, Windows-серверы используют, как правило, PFX-формат, а для Apache нужны PEM-файлы, имеющие расширение crt или cer. Попытаемся разобраться, какие форматы бывают и как конвертировать их из одного формата в другой.

Табл. 33.3 содержит описание часто используемых форматов SSL-сертификатов.

**Таблица 33.3. Форматы SSL-сертификаты**

Формат	Описание
PEM	Самый распространенный формат сертификата. Файлы в таком формате имеют расширение pem, crt, cer и key (файл приватного ключа). Сами по себе файлы являются обычными ASCII-файлами, закодированными в формате Base64. При открытии такого сертификата в текстовом редакторе вы видите строку ---BEGIN CERTIFICATE---, после чего следует закодированный сертификат, а затем строка ---END CERTIFICATE---. Веб-сервер Apache использует формат PEM. В одном файле может содержаться несколько SSL-сертификатов и даже приватный ключ. В этом случае каждый сертификат отделяется от остальных тегами BEGIN и END. Однако Apache требует, чтобы сертификаты и приватный ключ были в разных файлах
DER	Бинарный тип сертификата — в отличие от PEM, где сертификат хранится в ASCII-файле. Файлы сертификатов в этом формате часто имеют расширение cer, но можно встретить и расширение der. Если перед вами файл с расширением cer, то для вычисления его формата нужно открыть его в текстовом редакторе. Если вы увидите теги начала и окончания сертификата (BEGIN/END), то это формат PEM. Формат DER используется, как правило, на Java-платформах
PKCS # 7 / P7B	Файл сертификата в этом формате хранится в формате Base64 ASCII и имеет расширение p7b или p7c. В файлах находятся теги начала и окончания сертификата: — BEGIN PKCS7 — и — END PKCS7 —. Этот формат поддерживается Windows и Java Tomcat
PFX	Бинарный формат, при его использовании в зашифрованном файле хранятся ваш личный сертификат сервера, промежуточные сертификаты центра сертификации, а также закрытый ключ. PFX файлы, как правило, имеют расширение pfx или p12. Обычно используется в Windows для импорта и экспорта файлов SSL-сертификатов и приватного ключа

Табл. 33.4 содержит команды конвертирования SSL-сертификатов из одного формата в другой.

**Таблица 33.4. Команды конвертирования**

Направление	Команда
PEM > DER	openssl x509 -outform der -in certificate.pem -out certificate.der
PEM > P7B	openssl crl2pkcs7 -nocrl -certfile certificate.cer -out certificate.p7b -certfile CACert.cer
PEM > PFX	openssl pkcs12 -export -out certificate.pfx -inkey privateKey.key -in certificate.crt -certfile CACert.crt
DER > PEM	openssl x509 -inform der -in certificate.cer -out certificate.pem
P7B > PEM	openssl pkcs7 -print_certs -in certificate.p7b -out certificate.cer
P7B > PFX	openssl pkcs7 -print_certs -in certificate.p7b -out certificate.cer openssl pkcs12 -export -in certificate.cer -inkey privateKey.key -out certificate.pfx -certfile CACert.cer
PFX > PEM	openssl pkcs12 -in certificate.pfx -out certificate.cer -nodes

### 33.10.3. Сертификат Let's Encrypt

Если вкратце, то Let's Encrypt — это новый центр сертификации (СА), предоставляющий бесплатные и автоматизированные SSL/TLS-сертификаты. В настоящее время Let's Encrypt поддерживается большинством современных браузеров, в том числе IE и даже старыми операционными системами — такими, как Windows Vista. По сути, все, что вам нужно о нем знать, — он бесплатный и поддерживает автоматическое обновление.

#### Установка клиента Let's Encrypt

Установим клиент с помощью команды:

```
sudo git clone https://github.com/certbot/certbot /opt/letsencrypt
```

Если система git у вас не установлена, то сначала нужно установить ее, а потом уже она устанавливает клиент для Let's Encrypt (далее certbot). Файлы будут загружены в каталог /opt/letsencrypt.

#### Создаем каталог webroot-path/.well-known/acme-challenge/

Этот каталог позволяет серверу Let's Encrypt убедиться, что ваш сайт пытается получить бесплатный SSL-сертификат. Каталог следует создавать в корне веб-серв-

вера. Например, если корень у вас `/var/www/shop`, то в нем и надо создать требуемый каталог:

```
cd /var/www/shop
mkdir .well-known
mkdir .well-known/acme-challenge
find . -type d -exec chown www-data:www-data {} \;
```

## Создаем файл конфигурации

Теперь нужно создать файл конфигурации для вашего домена. Если ваш домен называется `example.com`, то его файл конфигурации будет называться `/etc/letsencrypt/configs/example.com.conf`. Содержимое файла приведено в листинге 33.6.

### Листинг 33.6. Файл `/etc/letsencrypt/configs/example.com.conf`

```
# ваш домен (хотя и можно создать один сертификат для нескольких доменов,
# мы рекомендуем создавать отдельные сертификаты и, следовательно, отдельные
# файлы конфигурации для разных доменов)
domains = example.com

# размер ключа
rsa-key-size = 2048 # или 4096

# сервер сертификации
server = https://acme-v01.api.letsencrypt.org/directory

# адрес, на который будут приходить напоминания об обновлении
email = my-email

# отключаем ncurses UI
text = True

# задаем путь к каталогу .well-known (см. выше)
authenticator = webroot
webroot-path = /var/www/shop/
```

## Запрос сертификата

Настало время запросить сам сертификат. Во второй команде вам нужно вставить точное имя своего файла конфигурации:

```
cd /opt/letsencrypt
$ ./certbot-auto --config /etc/letsencrypt/configs/example.com.conf certonly
```

Вывод последней команды приведен на рис. 33.18. Как видите, файлы сертификата помещены в каталог `/etc/letsencrypt/live/<имя>/` (поскольку скриншот сделан в процессе настройки реального узла, то его адрес затерт — NDA есть NDA<sup>1</sup>). В результате

<sup>1</sup> NDA — соглашение о неразглашении (от англ. Non-disclosure agreement).

```

Would you be willing to share your email address with the Electronic Frontier Foundation, a founding parther of the Let's Encrypt project and the non-profit organization that develops Certbot? We'd like to send you email about our work encrypting the web, EFF news, campaigns, and ways to support digital freedom.

(Y)es/(N)o: Y
Obtaining a new certificate
Performing the following challenges:
http-01 challenge for shop. [REDACTED].ru
Using the webroot path /var/www/[REDACTED] for all unmatched domains.
Waiting for verification...
Cleaning up challenges

IMPORTANT NOTES:
- Congratulations! Your certificate and chain have been saved at:
  /etc/letsencrypt/live/[REDACTED]/fullchain.pem
  Your key file has been saved at:
  /etc/letsencrypt/live/[REDACTED]/privkey.pem
  Your cert will expire on 2019-05-14. To obtain a new or tweaked
  version of this certificate in the future, simply run certbot-auto
  again. To non-interactively renew *all* of your certificates, run
  "certbot-auto renew"
- Your account credentials have been saved in your Certbot
  configuration directory at /etc/letsencrypt. You should make a
  secure backup of this folder now. This configuration directory will
  also contain certificates and private keys obtained by Certbot so
  making regular backups of this folder is ideal.
- If you like Certbot, please consider supporting our work by:

  Donating to ISRG / Let's Encrypt:  https://letsencrypt.org/donate
  Donating to EFF:                  https://eff.org/donate-le

```

**Рис. 33.18. Запрос сертификата**

было сгенерировано два файла: файл сертификата `fullchain.pem` и файл ключа `privkey.pem`.

## Настройка веб-сервера

Итак, в результате генерирования сертификата или его заказа вам будут предоставлены два файла: SSL-файл сертификата (с расширением `pem`) и ключевой файл (с расширением `key` или `pem` — в случае с Let's Encrypt).

Перейдите в каталог `/etc/apache2/sites-available`. В нем хранятся конфигурационные файлы сайтов, работающих на вашем веб-сервере. Откройте файл, содержащий конфигурацию сайта, для которого вы купили сертификат. Далее представим, что наш сайт называется `firma.ru`. Конфигурация для него будет следующей:

```

<VirtualHost *:80>
    ServerName firma.ru
    Redirect permanent / https://firma.ru/
</VirtualHost>

<VirtualHost *:443>
    ServerAdmin admin@firma.ru
    ServerName firma.ru

```

```
ServerAlias www.firma.ru xxxx.xxx.xxx.xxx
DocumentRoot /srv/www/firma.ru/htdocs
ErrorLog /srv/www/firma.ru/logs/error_log
CustomLog /srv/www/firma.ru/logs/access_log combined env=!loopback

SSLEngine on
SSLCertificate /etc/letsencrypt/live/<имя>/fullchain.pem
SSLCertificateKeyFile /etc/letsencrypt/live/<имя>/privkey.pem
</VirtualHost>
```

Разберемся, что здесь к чему. Сначала мы создаем виртуальный хост для порта 80, который будет работать как перенаправление — на HTTPS-версию сайта. Можно было бы сделать это и через файл .htaccess, но поскольку у нас есть доступ к конфигурации сервера, то можно сделать это прямо здесь.

Далее мы описываем виртуальный хост для порта 443 (используется SSL). Настройки такие же, как и для обычной версии сайта: ServerName, DocumentRoot и т. д. Различие заключается только в наличии трех SSL-директив. Первая включает SSL, вторая задает PEM-файл, третья — файл приватного ключа.

После этого нужно сохранить файл конфигурации и перезапустить Apache:

```
sudo systemctl restart apache2.service
```

или (в зависимости от вашего дистрибутива)

```
sudo service apache2 restart
```

Обратитесь к вашему сайту. Если вместо надписи «Не защищено» появилось изображение зеленого замка, то все хорошо, и настройку можно считать завершенной.

Однако иногда замок отображается не зеленым, а серым и не закрытым, а открытым. Это означает, что не все ресурсы сайта загружаются по протоколу HTTPS. Откройте исходный код страницы и произведите поиск по строке: `http://`. Ваша задача — найти адреса ресурсов (JS, CSS, картинок), которые загружаются по протоколу HTTP. Исправьте интернет-адреса (URL) проблемных ресурсов на `https://` и снова обновите страницу сайта. Если вы все сделаете правильно, то увидите зеленый замок соединения.

## Автоматическое обновление сертификата

Наш сертификат будет действителен в течение 90 дней, после чего должен быть обновлен. Для обновления можно использовать следующий сценарий (листинг 33.7).

### Листинг 33.7. Сценарий renew-letsencrypt.sh

```
#!/bin/sh

cd /opt/letsencrypt/
./certbot-auto --config /etc/letsencrypt/configs/my-domain.conf certonly
```

```
if [ $? -ne 0 ]
then
    ERRORLOG=`tail /var/log/letsencrypt/letsencrypt.log`
    echo -e "The Let's Encrypt cert has not been renewed!\n\n"
    $ERRORLOG
else
    nginx -s reload
fi

exit 0
```

А в расписание cron нужно добавить строку:

```
0 0 1 JAN,MAR,MAY,JUL,SEP,NOV * /path/to/renew-letsencrypt.sh
```

И не забудьте создать каталог `/var/log/letsencrypt/` (если он еще не создан) и изменить соответствующим образом права доступа к нему (пользователь, от имени которого выполняется обновление сертификата, должен иметь право записывать в этот каталог).

## 33.11. Ускорение веб-сервера: PageSpeed и Memcached

Хорошо, если движок сайта обладает возможностью кэширования страниц, — это может существенно повысить его производительность. Например, для того же WordPress предусмотрено несколько эффективных плагинов кэширования. Но ведь они не кэшируют «админку» — серверную часть сайта, а она по-прежнему «тормозит». К тому же если сайтом вообще не предусмотрено каких-либо механизмов кэширования, то будет тормозить как «фронт» (интерфейс пользователя), так и «бэк» (админка).

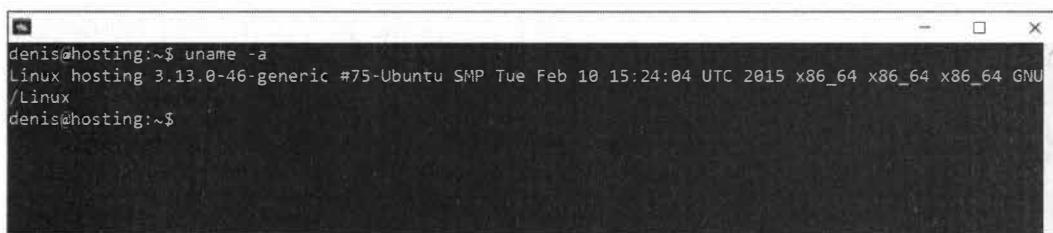
Добиться ускорения в этом случае можно путем установки Google-сервиса PageSpeed (модуль `mod_pagespeed`) и его настройки на работу с системой кэширования данных — демоном Memcached.

### 33.11.1. Установка PageSpeed

Первым делом нужно определить разрядность операционной системы. Конечно, сегодня найти 32-разрядный сервер — еще та задача, но все же... Введите команду:  
`uname -a`

Если увидите в выводе `x86_64` (рис. 33.19) — ваша система 64-разрядная, и надо выполнить следующие команды:

```
cd /tmp
wget https://dl-ssl.google.com/dl/linux/direct/mod-pagespeed-stable_current_amd64.deb
sudo dpkg -i mod-pagespeed-stable_current_amd64.deb
```



```
denis@hosting:~$ uname -a
Linux hosting 3.13.0-46-generic #75-Ubuntu SMP Tue Feb 10 15:24:04 UTC 2015 x86_64 x86_64 x86_64 GNU
/Linux
denis@hosting:~$
```

Рис. 33.19. Вывод команды `uname -a`

Для 32-разрядной системы команды будут такими:

```
cd /tmp
wget https://dl-ssl.google.com/dl/linux/direct/mod-pagespeed-stable_current_
i386.deb
sudo dpkg -i mod-pagespeed-stable_current_i386.deb
```

После этого перезапустите Apache:

```
sudo service apache2 restart
```

### 33.11.2. Установка Memcached

Демон Memcached позволяет добиться существенного ускорения загрузки страниц. Он доступен из репозиториев Ubuntu:

```
sudo apt install memcached
```

После установки демона нужно узнать порт, на котором он работает. Введите команду:

```
netstat -tap | grep memcached
```

Порт будет выведен после слова `localhost` — например: `localhost:11211`. Это стандартный порт для Memcached, который нужно указать в конфигурации модуля `mod_pagespeed`.

Для ускорения PHP-приложений следует установить пакет `php-memcached`:

```
apt install php-memcached
```

Осталось настроить модуль `mod_pagespeed` на работу с помощью Memcached. Для этого откройте файл `/etc/apache2/mods-available/pagespeed.conf`:

```
mcedit /etc/apache2/mods-available/pagespeed.conf
```

Произведите поиск по строке: `ModPagespeedMemcachedServers`. Раскомментируйте строку:

```
# ModPagespeedMemcachedServers localhost:11211
```

Она должна быть такой:

```
ModPagespeedMemcachedServers localhost:11211
```

Перезапустите Apache:

```
sudo service apache2 restart
```

После этого можно наблюдать некоторое ускорение работы вашего сайта. Для более тонкой настройки обратитесь к документации по Memcached.

## 33.12. Протоколирование POST-запросов

Как все мы знаем, HTTP-запросы бывают двух типов: GET и POST. С логированием GET-запросов проблем никаких не возникает — все такие запросы заносятся в файл `access.log`. А вот что с POST-запросами? Как мы узнаем, какой пользователь обращался с помощью POST к нашим сценариям? В этом нам поможет модуль `modsecurity`.

Возможности `modsecurity` весьма большие, но нас интересует сейчас только логирование POST-запросов, на этом и остановимся.

Установите `modsecurity`:

```
sudo apt install libapache2-modsecurity
```

Скопируйте файл конфигурации с рекомендуемыми параметрами, чтобы использовать его в качестве основного файла конфигурации:

```
cd /etc/modsecurity/
sudo cp modsecurity.conf-recommended modsecurity.conf
```

Откройте файл `/etc/modsecurity/modsecurity.conf` для редактирования:

```
sudo mcedit /etc/modsecurity/modsecurity.conf
```

Добавьте в него следующие строки:

```
SecRequestBodyAccess On
SecRule REQUEST_METHOD "POST" "id:200012,phase:2,ctl:auditEngine=On,log,pass"
```

При желании вы можете настроить и другие параметры этого модуля — обратитесь к соответствующей документации (вы без проблем найдете ее в Интернете).

Перезапустим Apache:

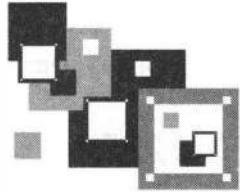
```
sudo service apache2 restart
```

Файл журнала `modsecurity` называется `/var/log/apache2/modsec_audit.log`.

Если на сервере «крутится» несколько сайтов, то соответствующую настройку можно произвести для каждого отдельного виртуального узла. Откройте файл конфигурации виртуального узла и добавьте строки:

```
<IfModule security2_module>
    SecAuditLog /var/log/apache2/audit_example_com.log
    SecRule REQUEST_METHOD "POST" "id:22222224,phase:2,ctl:auditEngine=
                                On,log,pass"
    SecRuleEngine On
</IfModule>
```

Здесь мы не только включаем логирование POST-запросов, но и указываем файл конфигурации.



## 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.

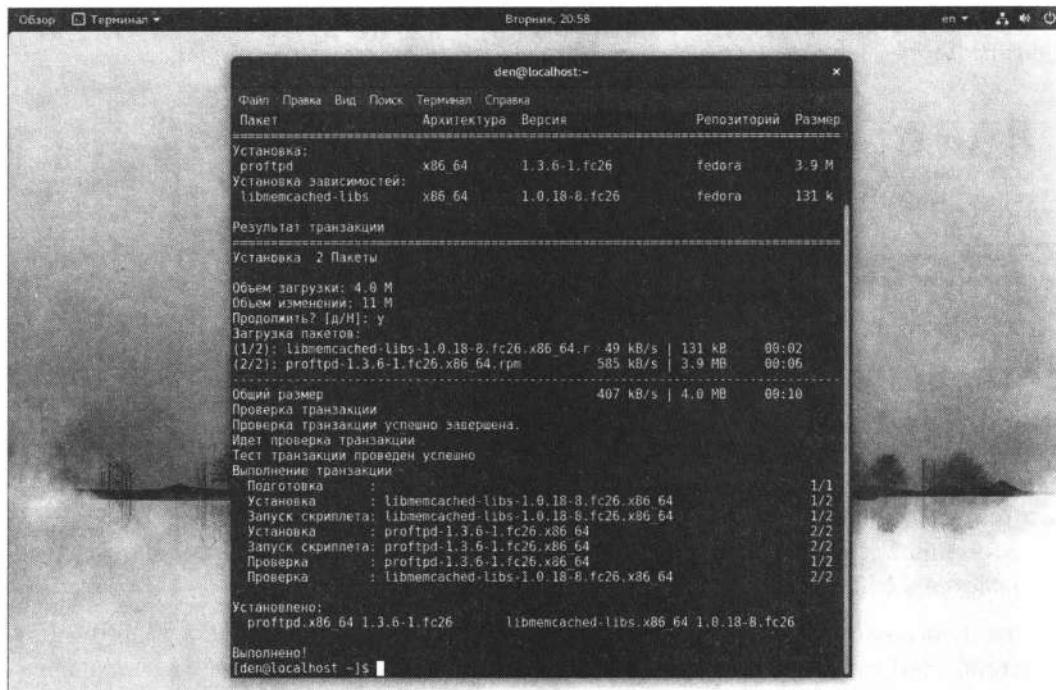
Права пользователя FTP-сервера определяются его администратором. Одним пользователям разрешается загружать файлы в свои личные каталоги на сервере, другие имеют полный доступ к FTP-серверу (имеют право загружать файлы в любые каталоги — как правило, это администраторы FTP-сервера), трети могут только 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-ftpd`, но ProFTPD является одним из самых защищенных и удобных в настройке.

### 34.1. Установка FTP-сервера

Для установки FTP-сервера нужно инсталлировать пакет `proftpd` (рис. 34.1). Хорошо бы также установить и конфигуратор `gproftpd`, если он доступен в вашем дистрибутиве.



**Рис. 34.1. Fedora 30: установка ProFTPD**

Запуск и останов сервера осуществляется командой `service`:

```
sudo service protfpd start  
sudo service protfpd stop
```

Для получения информации о состоянии сервера и его перезапуска обычно используются команды `status` и `restart`:

```
sudo service protfpd status  
sudo service protfpd restart
```

## **34.2. Конфигурационный файл**

Основным конфигурационным файлом сервера ProFTPD является файл `/etc/proftpd/proftpd.conf`. В листинге 34.1 представлен его простейший пример. В зависимости от дистрибутива и версии ProFTPD, файл конфигурации ProFTPD может отличаться от приведенного в листинге 34.1.

### Листинг 34.1. Пример файла конфигурации /etc/proftpd/proftpd.conf

```
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.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
<code>AccessGrantMsg "сообщение"</code>	Задает сообщение, которое будет отправлено пользователю при его регистрации на сервере. Можно задать грозное сообщение, напоминающее о том, что попытка несанкционированного доступа карается статьей какой-то уголовного кодекса
<code>Allow from all   узел   сеть [,узел   сеть[, ...]]</code>	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
<code>AllowAll</code>	Разрешает доступ всем. Может использоваться в блоках <code>Directory</code> , <code>Anonymous</code> , <code>Limit</code>
<code>AllowForeignAddress on   off</code>	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 <code>Anonymous</code> , <code>&lt;Global&gt;</code>
<code>AllowGroup список групп</code>	Разрешает доступ к серверу указанным группам пользователей (группы должны быть зарегистрированы на этом сервере)
<code>AllowOverwrite on   off</code>	Разрешает (он) 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	Задает альтернативный файл групп. По умолчанию /etc/group
AuthUserFile файл	Задает 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>	Задает параметры доступа к каталогу и его подкаталогам
<Global>	Задает 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	Задает тип запуска сервера. Значение по умолчанию — standalone (автономный запуск). Не нужно его изменять
ServerName "имя"	Задает 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	Задает порядок выполнения директив Allow и Deny в блоке Limit
TimeoutIdle секунды	Определяет тайм-аут простоя. Если пользователь не проявил 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	Задает 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 закачки 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                 anonymous ftp

# 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 — не зарегистрированным 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:

```
MaxHostsPerUser 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 его копия). 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 ( $2 \times 10$ ). 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 `ftpwho` и `ftpcount`

Вспомогательные программы `ftpwho` и `ftpcount` помогут администратору FTP-сервера определить, какие пользователи в текущий момент зарегистрированы на сервере (`ftpwho`), и узнать общее число зарегистрированных на сервере в текущий момент пользователей (`ftpcount`). Вывод обеих программ показан на рис. 34.2.

```
den@den-desktop:~$ ftpwho
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 `ftpwho` и `ftpcount`

## 34.6. Несколько слов о защите FTP

ProFTPD — весьма защищенный сервис, и его взлом является только следствием неправильной его настройки.

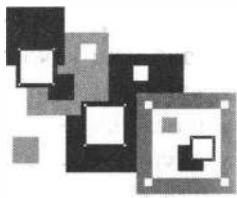
Вспомним директиву `DefaultRoot`, задающую корневой каталог для сервера. Рекомендуется установить значение этой директивы в `~`. Как мы знаем, тильда (`~`) означает домашний каталог пользователя. Следовательно, каждый раз при регистрации пользователя на FTP-сервере корневым каталогом FTP-сервера для него станет его домашний каталог. В результате, пользователь не сможет прочитать (а при неправильных правах доступа — изменить) важные системные файлы.

Также рекомендуется включить директиву `RequireValidShell`:

```
RequireValidShell on
```

Если эта директива включена, злоумышленник не сможет установить в качестве оболочки какую-нибудь вредоносную программу. FTP-сервер будет проверять, указана ли программа-оболочка в файле `/etc/shells`. Если программа не указана в этом файле, то FTP-сервер не будет ее запускать.

# 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) достаточно много: com, biz, org, info, gov, net, ws, домены стран (ru, рф, 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\_N] . ... [домен.TLD]

Например,

ftp.sales.firma.ru

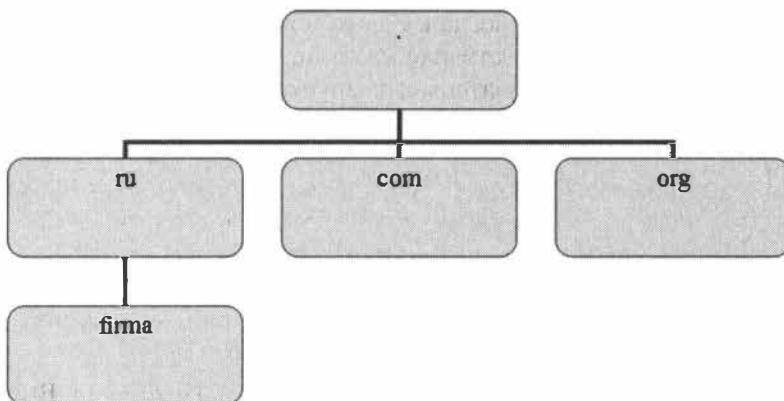


Рис. 35.1. Иерархическая структура DNS

При запросе к DNS-серверу доменное имя обрабатывается в доменном порядке. Сначала наш DNS-сервер посыпает 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. Понятно также и то, что такой запрос занимает весьма 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-тロjanами. 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-

ре, 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 (Berkeley 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. Когдато 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 — 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
    # попробует 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 записываются 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 попытку 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 `rndc.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, это 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 таком 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 firma.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 nslookup:

```
# 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, весьма большая, поэтому для каждого подразделения пришлось организовать свой домен: *sales.firma.ru*, *dev.firma.ru*, *orders.firma.ru* и т. п. Для управления всем доменом *firma.ru* (и всеми его поддоменами) мы можем или использовать 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.firma.ru.
IN A       10.0.0.1
IN MX     100    server.firma.ru.
www      IN CNAME  server.firma.ru.
ftp       IN CNAME  server.firma.ru.
mail      IN CNAME  server.firma.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.firma.ru`. `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.firma.com` можно обратиться по следующим именам: `www.firma.ru`, `ftp.firma.ru`, `mail.firma.ru`.

Далее описаны два компьютера: `c2.firma.ru` (мы не ставили точку после `c2`, поэтому `firma.ru` сервер «допишет» автоматически) и `c3.firma.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
)
@      IN NS       server.firma.ru
1      IN PTR      server.firma.ru
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.firma.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.ru" {
    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 подготовки книги версия файла `named.root`.

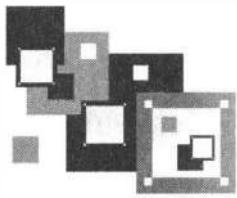
### Листинг 35.7. Файл `named.root` (db.root)

```
; 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.AR'L.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](http://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 (см. главу 38) и подобных сетевых сервисов. Для его установки нужно инсталлировать 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 192.168.0.1:8080

# 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 ваша машина должна использоваться еще для чего-либо,
# можно задать в мегабайтах, но тогда между числом и мегабайтами (МБ)
# 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; третью - 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 31281
```

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.

---

<sup>1</sup> При настройке прокси нужно указать порт, на котором он будет работать. Раньше администраторы, как правило, использовали порт 3128, сейчас «в тренде» порт 8080. Мы разрешаем оба — на всякий случай.

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 blacklist  
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 весьма 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. Большего 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 8080 (именно этот порт мы установили в конфигурационном файле). На рис. 36.1 показана настройка браузера Firefox.

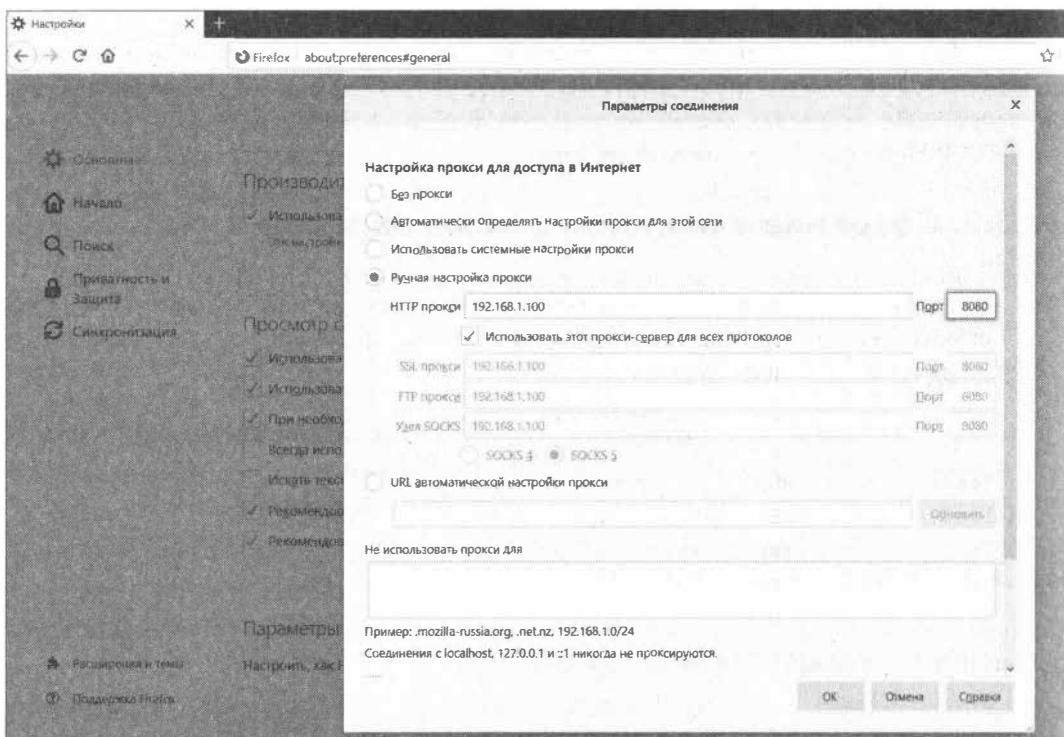


Рис. 36.1. Настройка клиента

## 36.6. Прозрачный прокси-сервер

С прокси-сервером часто связаны две проблемы. Первая заключается в том, что для работы через прокси-сервер нужно настраивать все клиенты. 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 -A PREROUTING -p tcp --dport 80 -j TransProxy
iptables -t nat -A PREROUTING -p tcp --dport 443 -j TransProxy
iptables -t nat -A TransProxy -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 8080
iptables -t nat -A TransProxy -p TCP -j DNAT --to 192.168.1.1:8080
```

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. п.

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 веб-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 none - обратно all, т. е.
        # будет запрещен любой контент. Значение none 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 !redactor !warez
    !ads !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 веб-сервере. Получается, что для работы squidGuard требуется веб-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 blocked.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
```

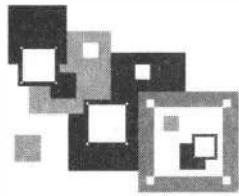
Осталось лишь перезапустить Squid:

```
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 сообщений, — 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 сообщения. 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 в значительной степени «дырявыми» и, мягко говоря, небезопасными. Сейчас с безопасностью у *sendmail* все в порядке, но все равно это слишком 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, *Thunderbird* или *The Bat!*). В процессе своей работы MUA подключается к почтовому серверу и принимает/передает электронные сообщения. Программа, установ-

ленная 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 Linux 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.»<sup>1</sup>, 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.

<sup>1</sup> См. <http://bvh.ru/books/book.php?id=188723>.

## 37.2. Настройка МТА Exim

Процесс настройки четвертой версии МТА Exim, рассмотренный здесь на примере дистрибутивов Ubuntu, Ubuntu Server и Debian (для остальных дистрибутивов соответствующую информацию можно найти в Интернете), существенно отличается от 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 (рис. 37.2), позволяющим настроить большинство параметров Exim4. Это один из немногих случаев, когда я считаю использование конфигуратора оправданным.

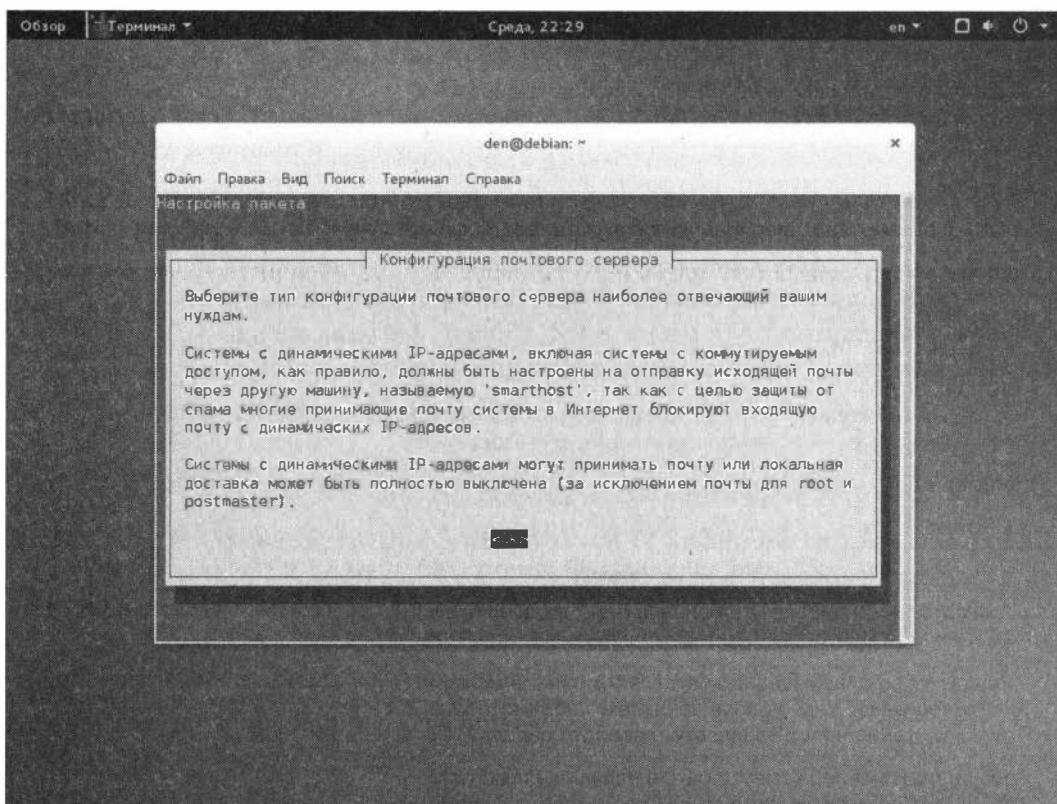


Рис. 37.2. Конфигуратор Exim4

Запустите конфигуратор командой:

```
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_tloptions` 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_PASSWORDS
```

```
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 = ${if saslauthd({$auth1}{$auth2}){1}{0}}
server_set_id = $auth1
.ifndef AUTH_SERVER_ALLOW_NOTLS_PASSWORDS
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 строчку:

START=no

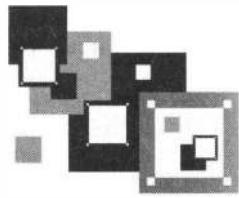
строчкой:

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 мы не раз уделяли внимание взаимодействию Linux-пользователей 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 позвольте себе несколько комментариев для пользователей более ранних версий Samba:

- параметр `server string` ранее назывался `comment`;
- параметры `client code page` и `character set` более 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 внешнему жесткому диску, который монтируется к каталогу `/mnt/ext_usb` через файл `/etc/fstab`.

#### Листинг 38.3. Пример общего доступа к внешнему HDD

```
[extusb]  
comment = External USB HDD for server backup  
read only = no  
writable = yes  
locking = no  
# каталог /mnt/ext_usb должен существовать и являться точкой монтирования  
path = /mnt/ext_usb  
public = yes  
# разрешить просмотр содержимого каталога  
browseable = yes
```

## 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, определяющий, 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`. Учиты-

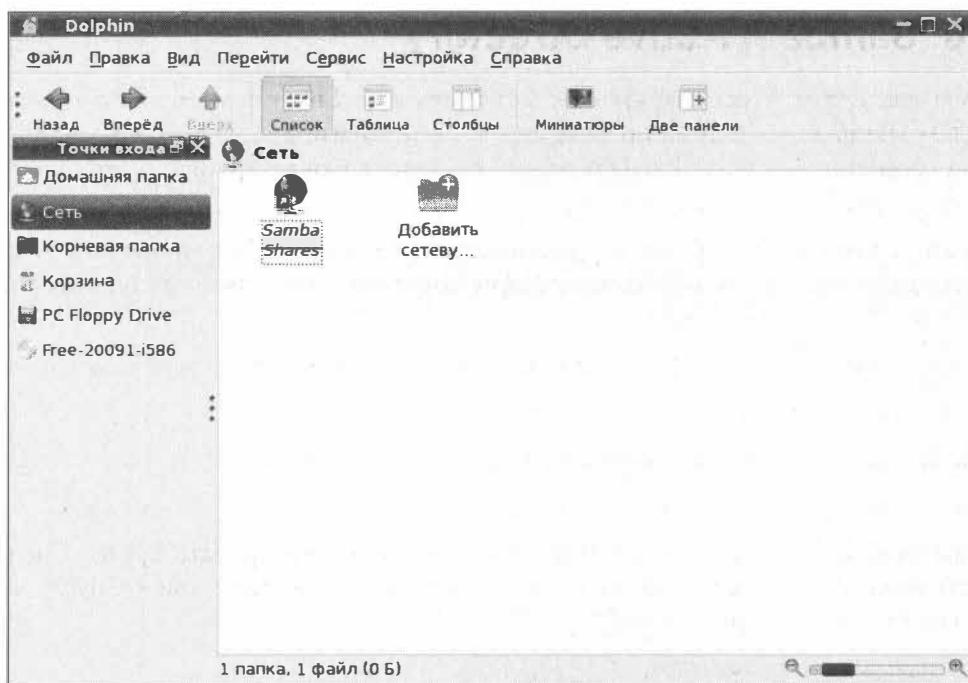


Рис. 38.1. Просмотр ресурсов сети с помощью Dolphin

ваяя, 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 IPTOS\_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 его 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]
    default_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.domain.ru
    default_domain = domain.ru
}

[domain_realm]
.domain.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: compat winbind  
group: compat winbind
```

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 — при этом 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 "dlopen /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
D      0 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 весьма просто:

```
sudo ln -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
```

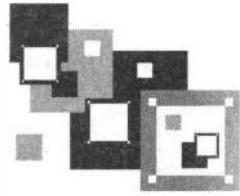
После этого нужно перезапустить сервис ntp, и вы получите полноценный контроллер домена. Не забудьте на Windows-компьютерах указать его IP-адрес в качестве адреса первичного DNS-сервера.

Для управления доменом можно использовать пакет Adminpack.msi, который можно скачать по адресам:

<http://www.microsoft.com/downloads/details.aspx?FamilyID=7D2F6AD7-656B-4313-A005-4E344E43997D&displaylang=ru>

<http://download.microsoft.com/download/3/e/4/3e438f5e-24ef-4637-abd1-981341d349c7/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<sup>1</sup>, или к калькулятору RAID<sup>2</sup>.

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

<sup>1</sup> См. <https://ru.wikipedia.org/wiki/RAID>.

<sup>2</sup> См. <http://altastor.ru/apps/raid-calculator/>.

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 $M \times (N - 1)$ , 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 большого 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. Весьма 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. Достаточно надежен, поскольку 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 и 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
1E0	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

каждый такой уровень, обратимся к данным табл. 39.2, содержащим общие характеристики самых часто используемых уровней RAID, т. е. именно тех уровней, с которыми вы будете работать на практике.

### 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.

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

Таблица 39.2 (окончание)

Уровень	Избыточность	Резервный диск	Чтение	Запись	Емкость
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

## 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 RAID 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** конфигуратора `make menuconfig` (см. главу 20), — не забудьте только после этого перекомпилировать ядро. Загру-

зив 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 простого производственного сервера убедительно вам рекомендую: 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). Напомню, 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:

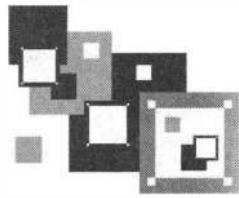
```
sudo mdadm /dev/md0 -a /dev/sdc1  
sudo mdadm /dev/md1 -a /dev/sdc2
```

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 более 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, врашивались 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 (восстановления) 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<sup>1</sup>, что на восстановление сравнительно небольшого массива из пяти 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 недешева! 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

---

<sup>1</sup> См. <http://searchsmbstorage.techtarget.com/tip/Five-ways-to-control-RAID-rebuild-times>.

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-ми и 14-ти 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.

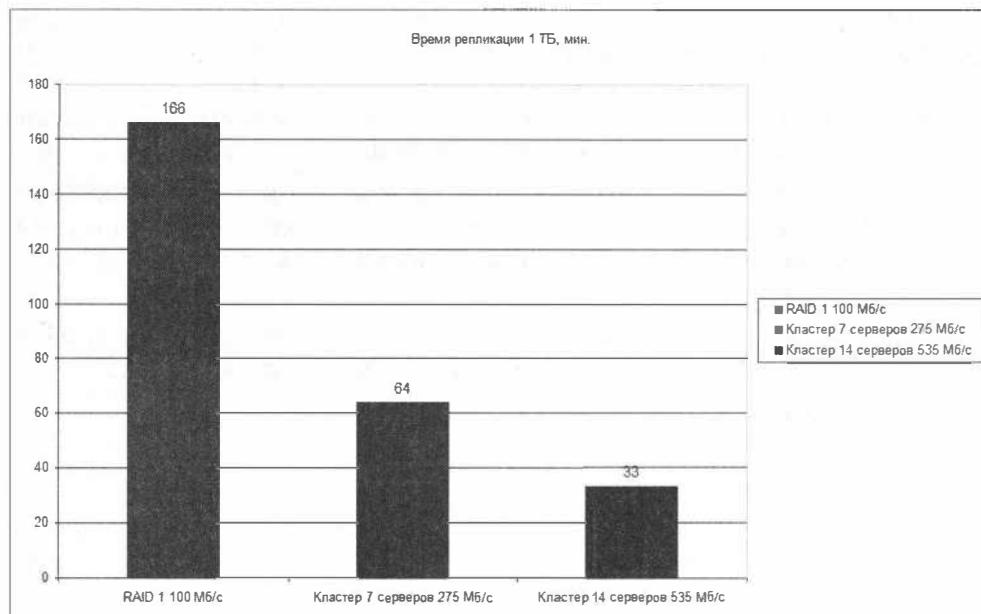


Рис. 40.1. Время репликации 1 Тбайт

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<sup>1</sup>, 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. Таким образом 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, и ссылки на некоторые из них приведены в разд. 40.3.1.

В папке Видео сопровождающего книгу файлового архива (см. *приложение*) вы найдете два анимационных файла: 40.3.gif и 40.4.gif. Считайте, что это третья и четвертая иллюстрации к этой главе. На рис. 40.3.gif показано, как работает Ceph в штатном режиме, а на рис. 40.4.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

---

<sup>1</sup> См. <http://www.yet.org/2012/12/staas/>.

рабочий узел и выполняется процедура замены адреса не читаемого сектора диска резервным (ремаппинг). Новая карта будет содержать новое расположение потерянной копии объекта. Все это выполняется 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 по устройству хранения объектов (OSD, Object Storage Device) псевдослучайно. 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 количеству операций ввода/вывода (IOPS, Input/Output Operations Per Second).

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).

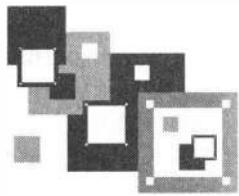
Резюмируя изложенное, приведем преимущества внедрения Ceph:

- аппаратный отказ узла (диска) незаметен для пользователей, как событие;
- времяостоя при отказе равно 0;
- нет единой точки отказа;
- снижения задержки ввода/вывода до уровня SSD-диска;
- доступны снэпшоты, клонирование, инкрементные диффы;
- сверхнизкие цены на услуги благодаря возможности работы на обычном оборудовании.

### 40.3.1. Система Ceph: дополнительная информация

- Подборка слайдов по производительности Ceph:  
<http://slides.com/sebastienhan/ceph-performance-and-benchmarking/#>.
- Каан 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](http://tracker.ceph.com/projects/ceph/wiki/Benchmark_Ceph_Cluster_Performance).
- Описание использования Red Hat Ceph:  
<https://habrahabr.ru/company/ua-hosting/blog/329618/>.

# ГЛАВА 41



## Средства резервного копирования. Создание образа системы на LiveUSB

### 41.1. Необходимость в «живой» резервной копии

Многие из нас помнят «привидение от Нортон» — Norton Ghost®. В мире Windows это в свое время был незаменимый продукт. Здесь же мы поговорим о средствах резервного копирования для Linux — они позволяют создать не просто резервную копию системы, а загрузочный USB-диск (LiveUSB) с резервной копией системы. Да, эти средства способны заменить Norton Ghost, причем они абсолютно бесплатны — в отличие от этого продукта Symantec.

Для начала определимся, зачем нужны средства создания LiveUSB. Наша цель — резервное копирование системы, но причем здесь LiveUSB? 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 сгорел жесткий диск. Опять все заново настраивать? Если же вы накануне создали LiveUSB (это может быть как флешка, которые сейчас достигли размера 64 Гбайт и выше при приемлемой стоимости, так и внешний жесткий диск, — доступного пространства должно хватить), то вам нечего беспокоиться: заменили жесткий диск, загрузились с созданного LiveUSB 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 LiveUSB и развернуть его на всех остальных компьютерах сети. Пусть настройка одного компьютера вместе с установкой системы займет полтора часа, создание 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 флешек. Да, именно на «количество», потому что вам надо будет создать несколько копий LiveUSB, чтобы можно было одновременно устанавливать систему на несколько компьютеров. Затем еще минут 40 ожидания, и будет настроено несколько компьютеров сразу — по числу имеющихся флешек. Удобно? Думаю, да. Без LiveUSB 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 станет расширяться.

Кстати, флешки, на которые записан «бэкап» в виде LiveUSB, могут использоваться не только для копирования/восстановления файлов самой системы. Вы можете копировать на эти флешки и пользовательские данные из каталога `/home` — лишь бы их размер не превысил свободную часть размера носителя, на который записывается «бэкап».

## 41.2. Средства клонирования Linux

Самым мощным средством для клонирования Linux является Clonezilla. Этот продукт может не только создать LiveUSB, но и развернуть систему по сети. На сайте разработчиков<sup>1</sup> 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**

Ранее в этой книге 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 закрыть 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<sup>2</sup> — он позволяет создать как LiveDVD (если для вас это еще актуально), так и LiveUSB.

---

<sup>1</sup> См. <http://clonezilla.org/>.

<sup>2</sup> См. <http://www.linux-live.org/>.

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, macOS (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/> ZIP-архив (если хотите записать Clonezilla на USB) или ISO-образ (если планируете создавать LiveDVD) Clonezilla Live и записать его на носитель. О том, как это сделать, рассказано в официальном руководстве<sup>1</sup>.
2. Загрузиться с носителя 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).

---

<sup>1</sup> См. <https://clonezilla.org/clonezilla-live.php>.

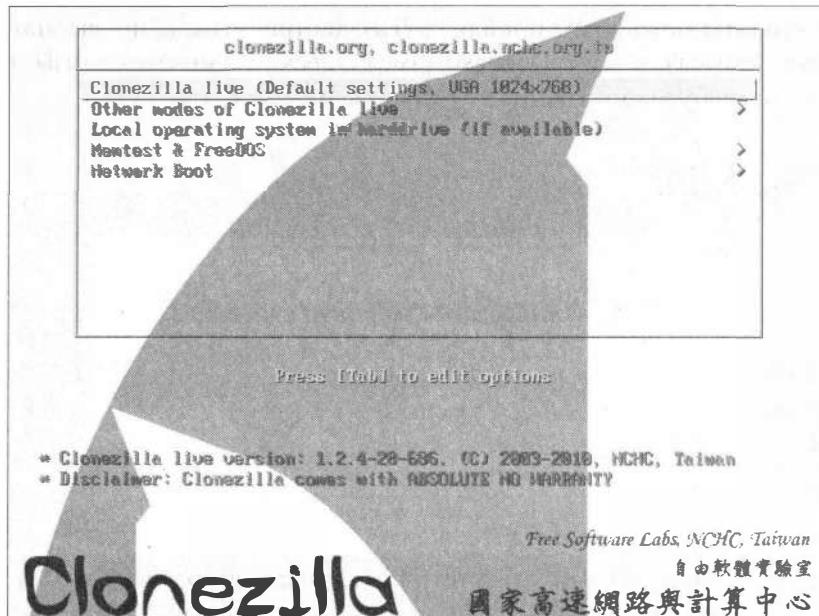


Рис. 41.1. Загрузочное меню Clonezilla Live

```

[ 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 CDR10 1.00 PQ: 0 ANSI: 5
[ 2.350556] sr0: scsi3-mmc drive: 1x/1x xa/form2 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] 31457200 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
[ 2.393087] 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 Louher

```

Рис. 41.2. Debian: процесс загрузки

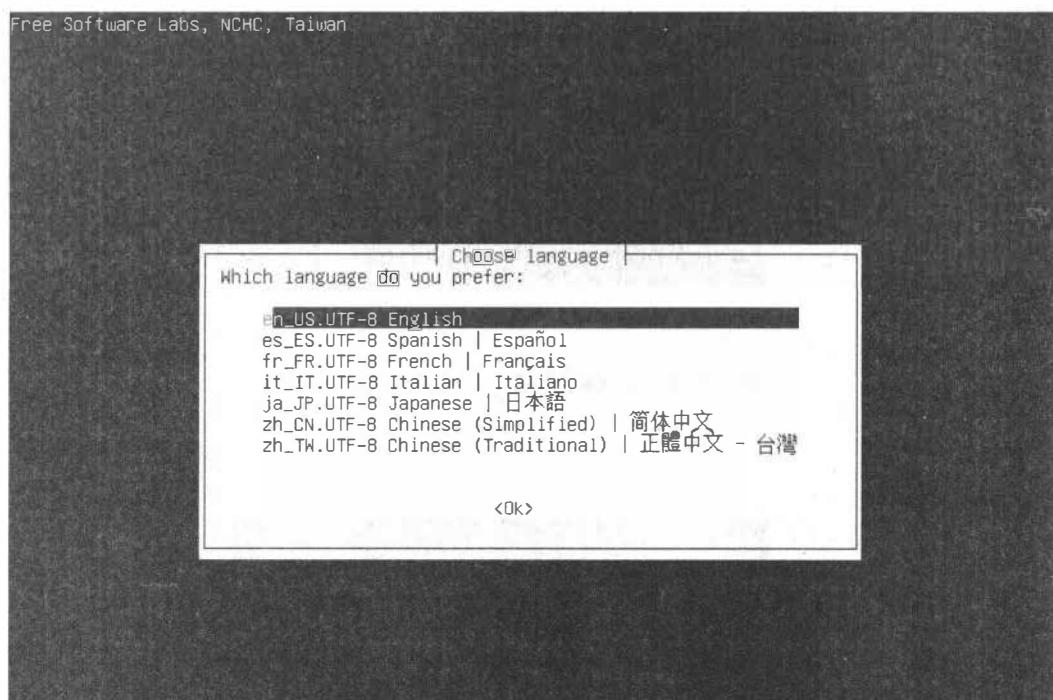


Рис. 41.3. Выбор языка Clonezilla

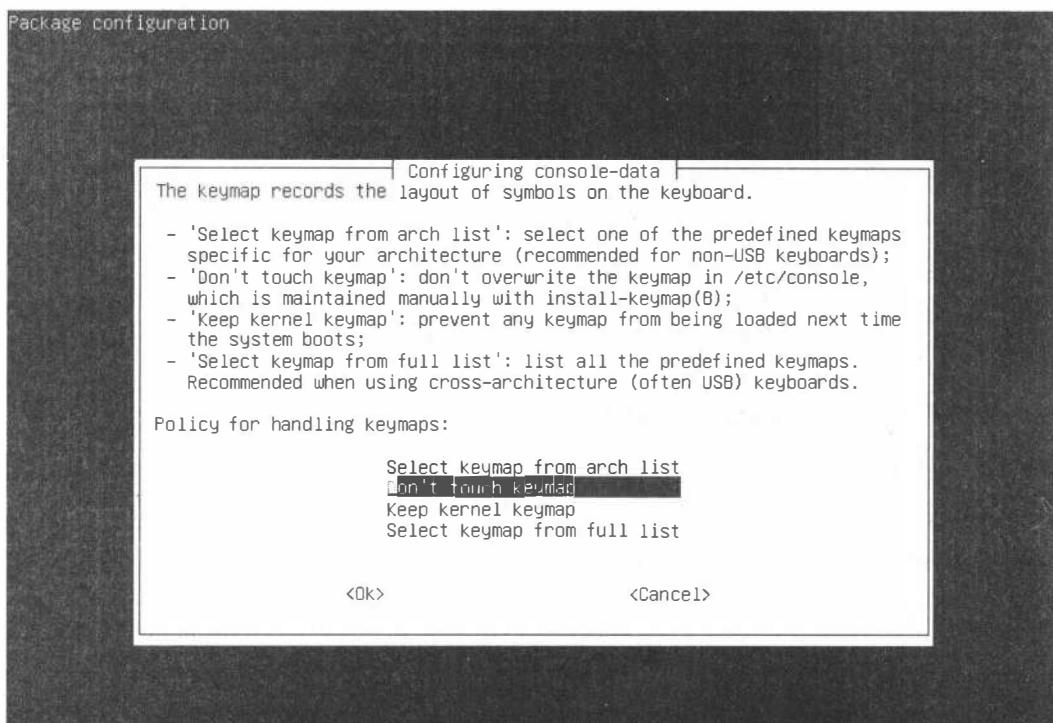


Рис. 41.4. Выбор раскладки

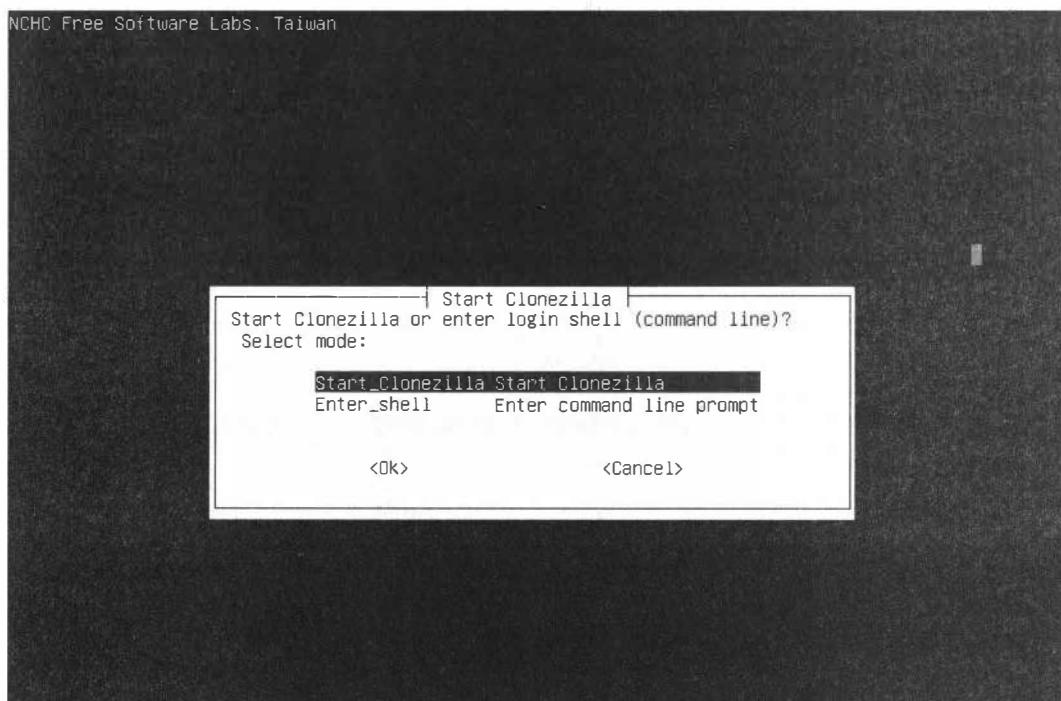


Рис. 41.5. Выберите команду **Start Clonezilla**

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 образу (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).

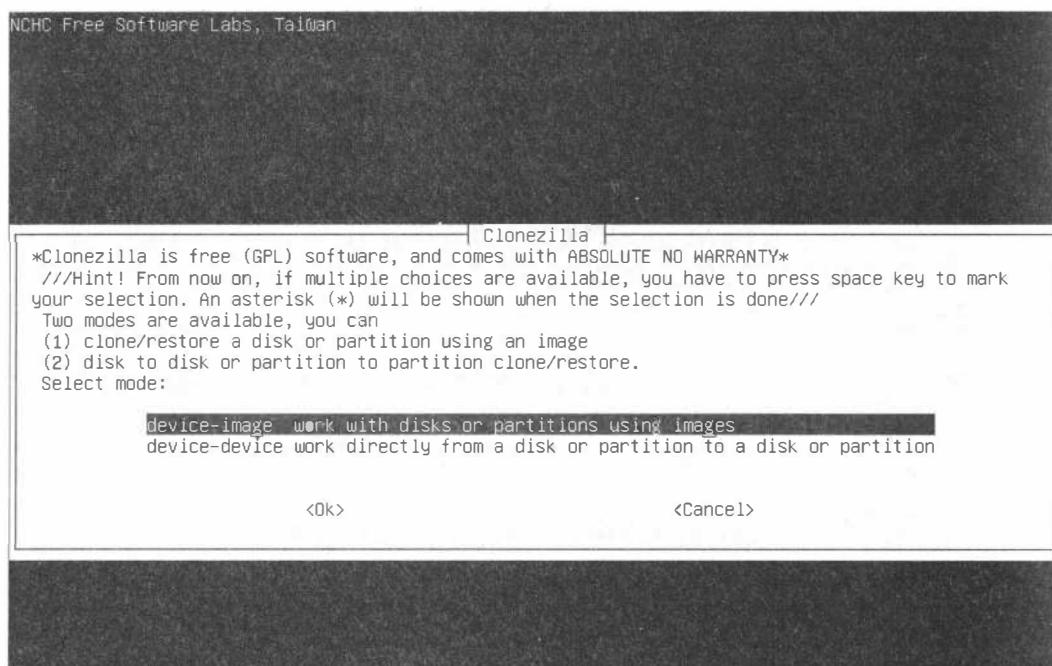


Рис. 41.6. Выберите режим device-image

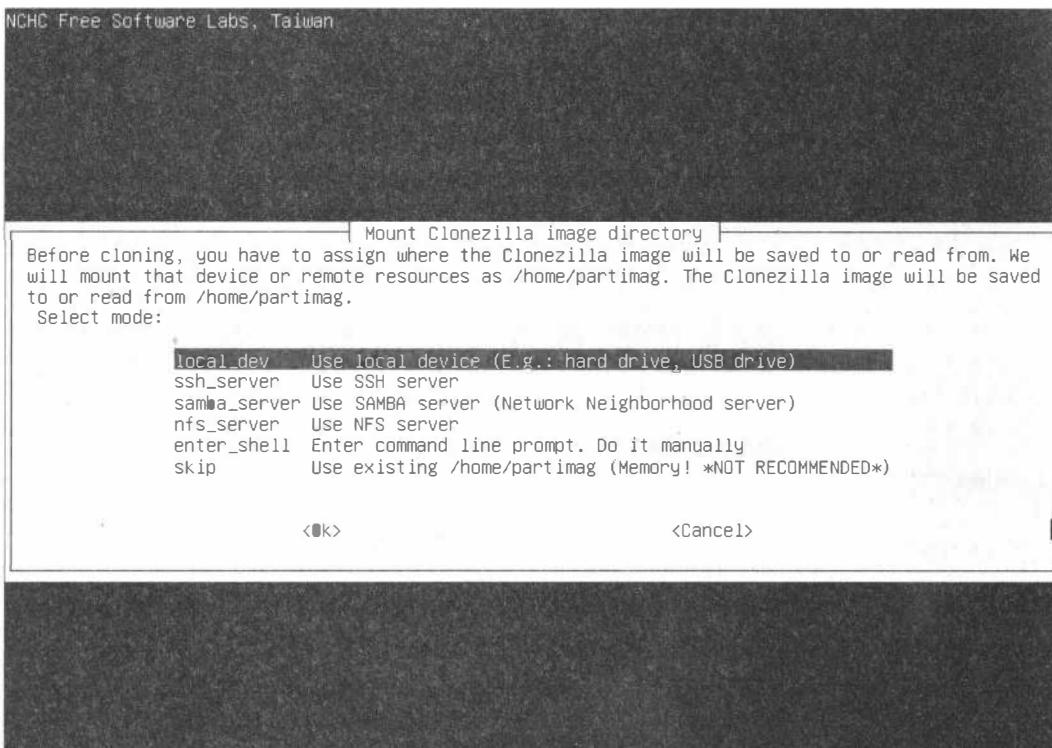


Рис. 41.7. Выбор носителя образа

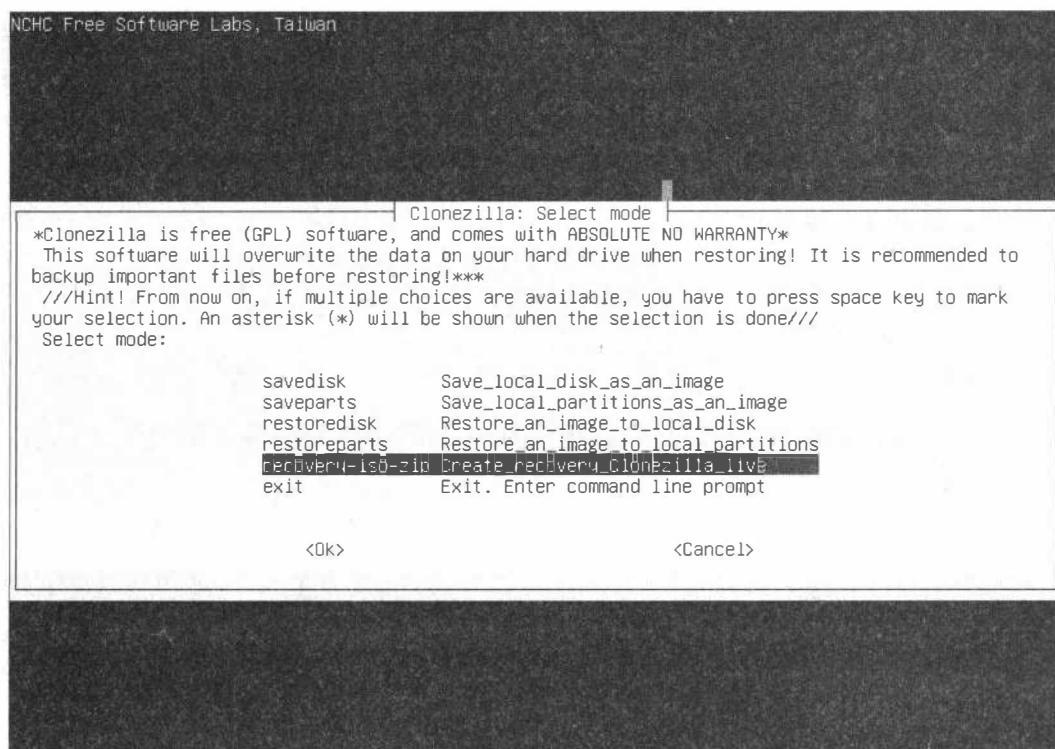


Рис. 41.8. Создать образ или восстановить?

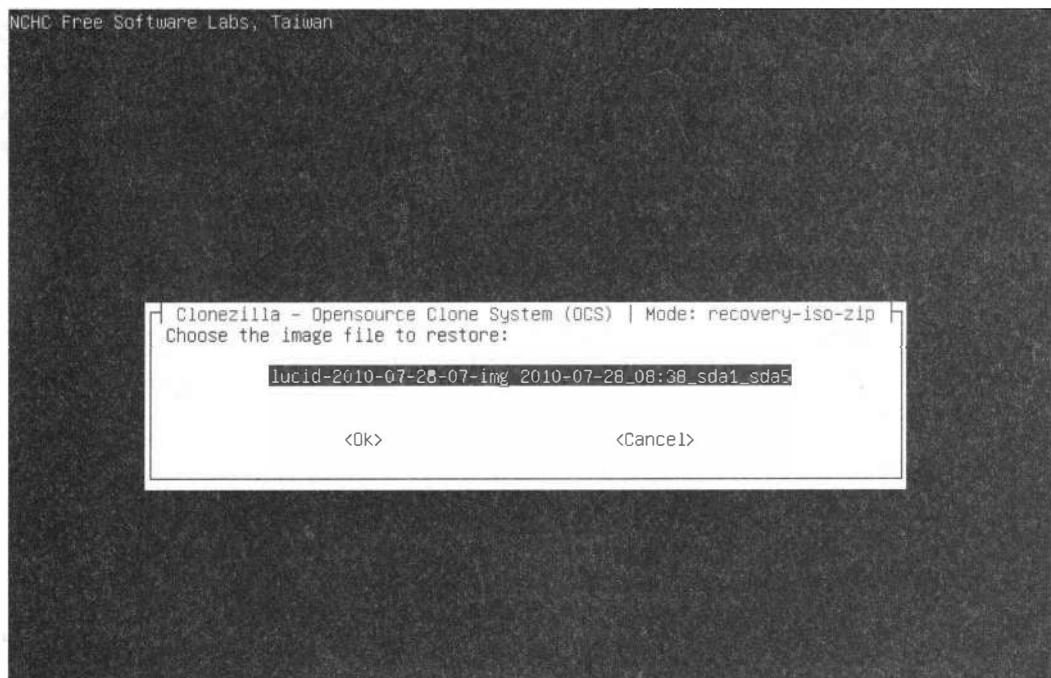


Рис. 41.9. Выбор образа для восстановления

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! Флешка обычно монтируется как `sdb1` (если в системе один SATA-диск), но лучше этот момент уточнить перед запуском Clonezilla или переключившись на другую консоль.

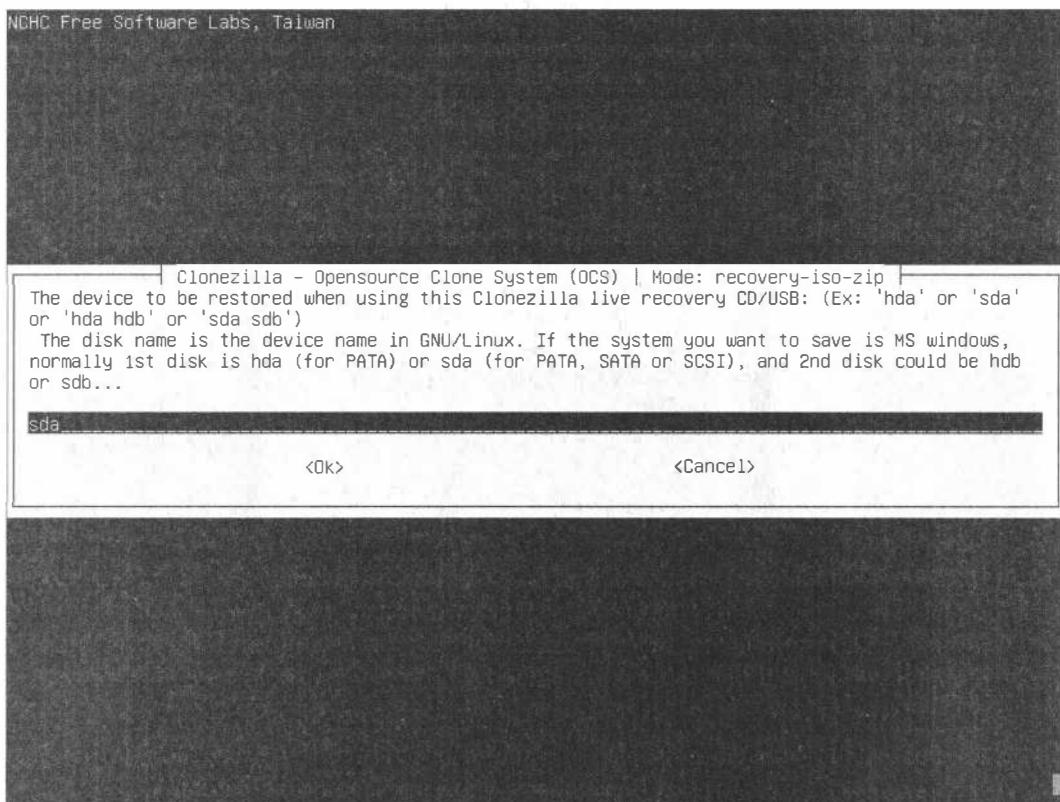


Рис. 41.10. На какое устройство «развернуть» образ

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).

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.

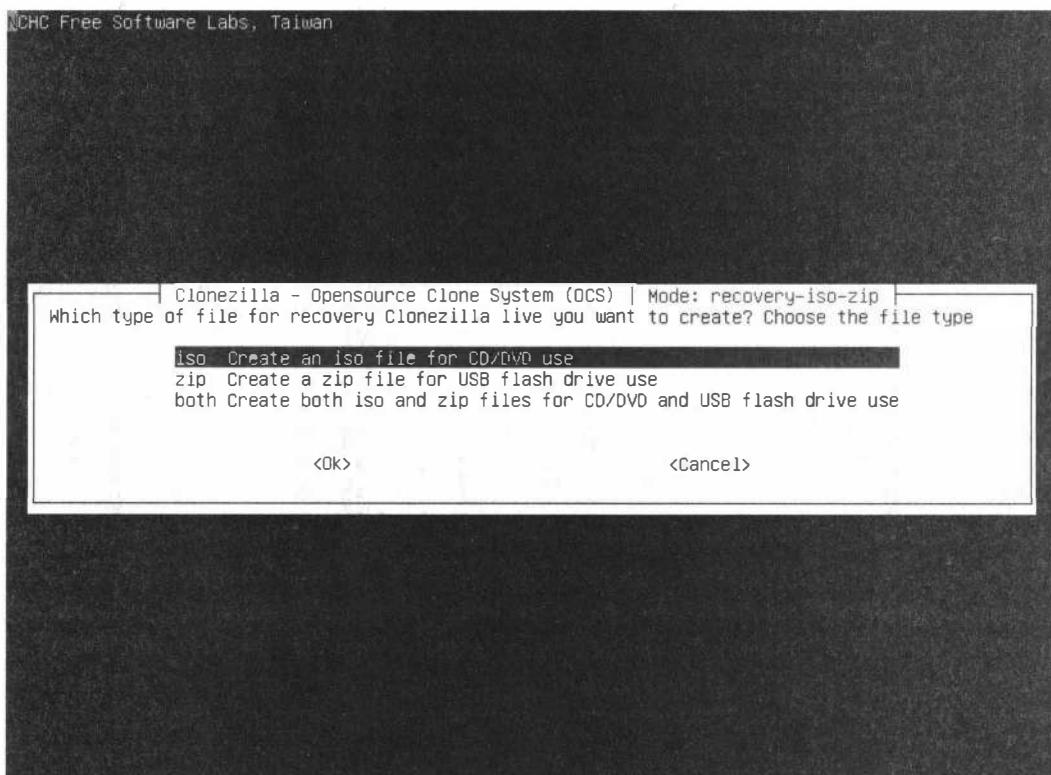


Рис. 41.11. Выбор режима команды `recovery-iso-zip`

```
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`

Если у вас есть необходимость в серверной версии (Clonezilla Server Edition), найти руководство по ее использованию вы можете по адресу: <https://clonezilla.org/clonezilla-SE/>.

## 41.4. Linux Live

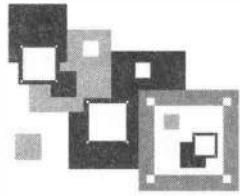
Теперь очередь дошла и до Slackware. Это очень хороший дистрибутив, пусть, может, и не такой удобный как Ubuntu, зато весьма надежный. Для создания LiveUSB в Slackware мы воспользуемся уже упомянутым ранее скриптом Linux Live. На сайте этого проекта<sup>1</sup> указано, что для работы с Linux Live рекомендуется дистрибутив

<sup>1</sup> См. <http://www.linux-live.org/>.

Slackware, — именно рекомендуется, а не требуется, то есть, 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.
2. Удалите все лишнее — например, лишнюю документацию и лишние программы, — чтобы уменьшить размер дистрибутива.
3. Скачайте скрипты Linux Live с сайта <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 для образа: /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 осущест-  
вим мы его стандартными средствами — с помощью криптографической файловой  
системы eCryptfs. Различные сторонние решения, которых вполне достаточно  
(взять тот же TrueCrypt или его форк CipherShed), мы здесь рассматривать не ста-  
нем — 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 ал-  
горитмами шифрования).

### **ПРОЕКТ TRUECRYPT**

Пользуясь моментом и «служебным положением», хочу выразить свое мнение относи-  
тельно проекта TrueCrypt. Этот проект ни разу не был никем скомпрометирован, и, види-  
мо, 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 по-  
водом для выпуска новой версии, но никак не для закрытия проекта, да еще и с реко-  
мендацией перейти на проприетарный BitLocker, который разработчики TrueCrypt вы-  
смеивали ранее...

### **42.1. Шифрование папки**

Для знакомства с криптографической файловой системой eCryptfs мы сейчас за-  
шифруем мой домашний каталог (`/home/den`). Причин шифрования данных может  
быть много, но у меня сейчас причина одна — сугубо академический интерес: по-  
пробовать и вам рассказать.

#### **ШИФРОВАНИЕ СРЕДСТВАМИ ДИСТРИБУТИВА**

Шифрование файловой системы с помощью средств, включенных в дистрибутив, опи-  
сано в главе 2.

Прежде всего установите утилиты eCryptfs. Поскольку работаю я сейчас в дистри-  
бутиве Debian, то для их установки буду использовать apt:

```
sudo apt install ecryptfs-utils
```

В дистрибутиве Fedora нужно выполнить команду dnf:

```
sudo dnf 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 клавишу 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 надоест 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, приkleенной 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:

<code>/dev/sdb1</code>	<code>/mnt/usb</code>	<code>vfat</code>	<code>ro</code>	<code>0 0</code>
<code>/home/den</code>	<code>/home/den</code>	<code>ecryptfs</code>	<code>defaults</code>	<code>0 0</code>

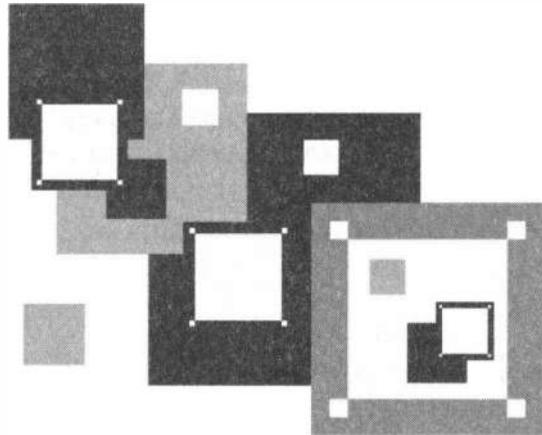
Первая строка монтирует флешку к каталогу `/mnt/usb`, а вторая — зашифрованную файловую систему. Понятно, что флешка должна быть смонтирована до монтирования зашифрованной файловой системы.

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.

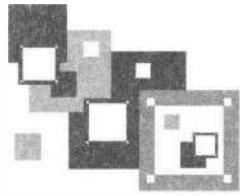


## ЧАСТЬ VII

### Виртуальные серверы

В настоящее время все чаще серверы делаются 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

Первым делом выберите 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.

Далее 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» под реальной нагрузкой при одновременной работе даже пяти-десяти пользователей, например, в программе «1С»?

Если вы себе представляете сервер именно так: отдельный компьютер, скажем, с 16 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 такое устройство назвать нельзя.

В моем же представлении сервер — это машина с серверным процессором Xeon, регистровой памятью с ECC (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 data-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.

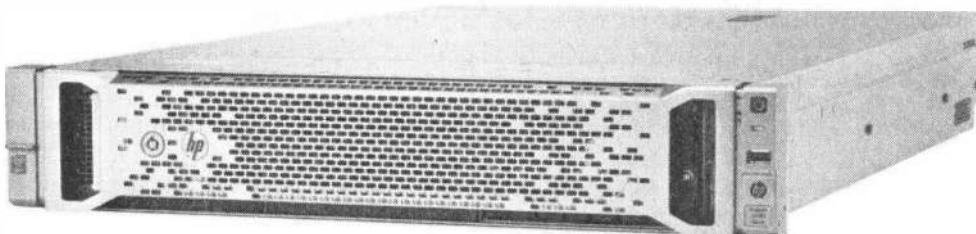


Рис. 43.1. Сервер HP ProLiant DL180 Gen9

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<sup>1</sup>. 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 «середнячок» — 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 вовсе не нужен. Аргументы «У соседа есть» или «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. п.;

---

<sup>1</sup> Понятно, что при нынешней волатильности цен и курсов валют конкретные цены на товары и услуги, приведенные в этой главе, могут на момент выхода книги из печати несколько отличаться от реальных, однако это не делает неверными их соотношения, на которых построен анализ эффективности применения физических и виртуальных устройств.

- портал с объемным контентом* — фотобанк 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 подразумеваются 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 впрямую не спросите об этом.

Второй момент, который стоит уточнить перед покупкой сервера, — тип накопителя, использующийся 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, и 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 «оверсэллить» ресурсы — т. е. продавать одни и те же ресурсы несколько раз разным пользователям.

Рассмотрим недостатки OpenVZ:

- **оверсэллинг** — 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 гигабайта 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 четырех гигабайт оказалось мало, и она узурпировала еще и ваши незадействованные 3,5 гигабайта. А если этого и 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 «состворить» с ядром операционной системы тоже не удастся. Хорошо хоть, что это нужно далеко не всем.

Справедливо 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. п.;
- **нет оверселинга** — вы честно получаете те ресурсы, за которые платите. Если вы заказали сервер с четырьмя гигабайтами 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 вашего виртуального сервера;
- **надежность и стабильность**, сравнивая 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 data-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 в настоящее время необходимо пространства для вашего проекта? Именно сейчас, а не через год или два, — в отличие от физического сервера, где вы поку-

паете 300 Гбайт сразу, в случае с VDS вы сможете покупать 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 оккупит 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 или 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 ощущимо.

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... 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 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

Физический 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 сервера, — 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 руб./мес. 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 вычертить 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/>;
- облачный сервис «1cloud» — [1cloud.ru](http://1cloud.ru).

Это не реклама провайдеров, поэтому тестирование будет объективным. Во время тестирования мы оценим пропускную способность, скорость работы дисковой подсистемы, по возможности выполним нагружочное тестирование, а также разберемся с ценами. Здесь надо уточнить, что эта глава готовилась в июле 2019 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 весьма демократична — конфигурация **Альфа** предлагается весьма задешево — 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 скучненькая, 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.

Если сравнить эту цену с предложением германского провайдера [hetzner.com](http://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

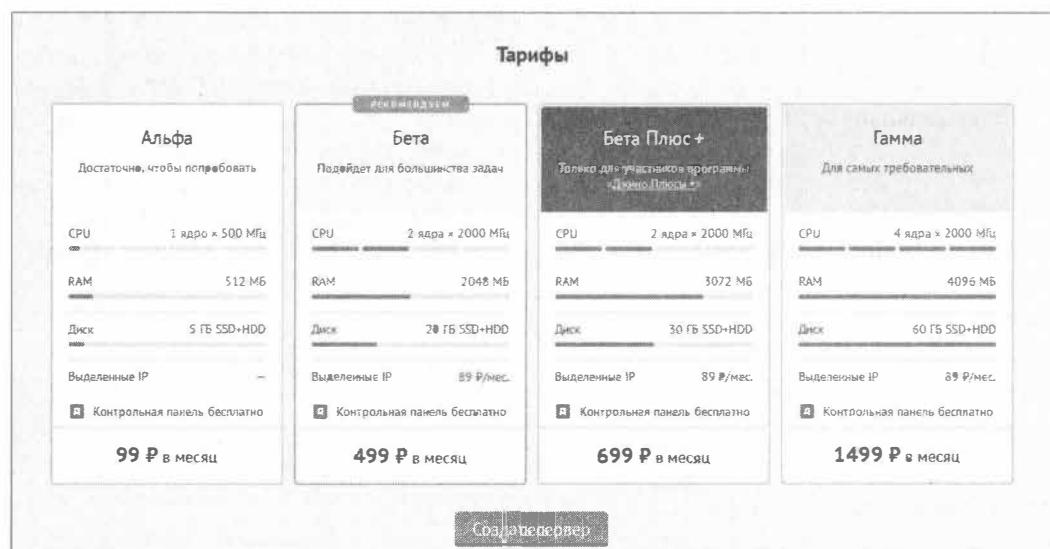


Рис. 43.2. «Джино»: стоимость VPS

5 Гбайт больше). Выходит даже немного дешевле. Но нужно учитывать, что у **hetzner.com** трафик платный — бесплатные только первые 2 Тбайт, да и конфигурация 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

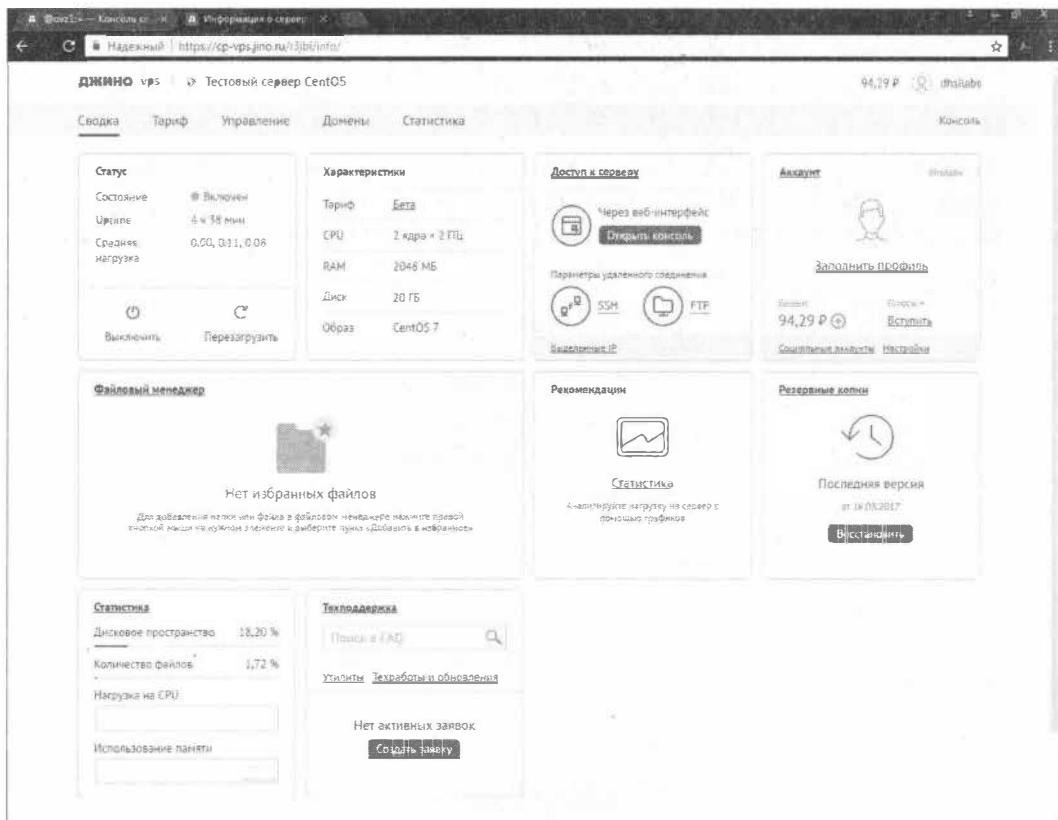


Рис. 43.3. «Джино»: панель управления

тариф, просмотреть статистику использования ресурсов. Ссылка **Открыть консоль** в разделе **Доступ к серверу** позволяет открыть консоль root (рис. 43.4).

Надо отметить, что веб-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 веб-консоль предлагается только как экстренная мера. Обычно нужно использовать SSH, клиенты которого есть даже для Android.

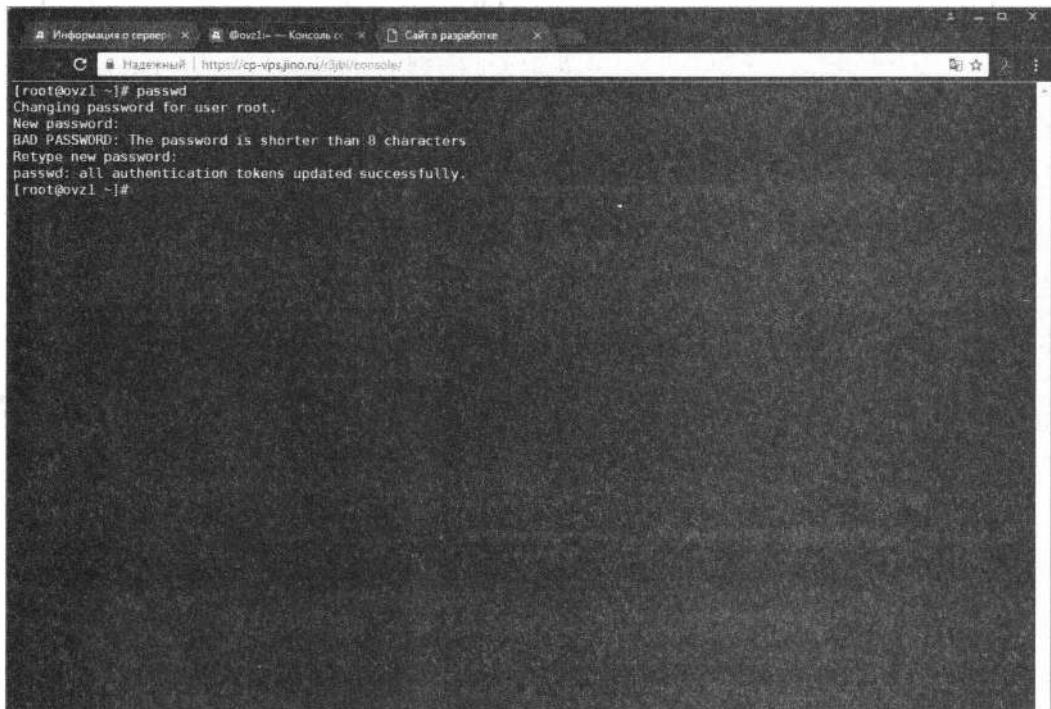


Рис. 43.4. «Джино»: консоль root

## Тестирование

Проведем простенькое нагрузочное тестирование. Можно было бы использовать неприхотливую утилиту *ab* из комплекта пакета Apache, но это неинтересно, поэтому я установил *siege* — специализированную утилиту для нагрузочного тестирования веб-серверов:

```
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.

```

HTTP/1.1 200 0.81 secs: 768 bytes => GET /
HTTP/1.1 200 0.79 secs: 768 bytes => GET /
HTTP/1.1 200 0.78 secs: 768 bytes => GET /
HTTP/1.1 200 0.75 secs: 768 bytes => GET /
HTTP/1.1 200 0.67 secs: 232736 bytes => GET /static/main.js
HTTP/1.1 200 0.58 secs: 768 bytes => GET /
HTTP/1.1 200 0.58 secs: 232736 bytes => GET /static/main.js
HTTP/1.1 200 0.58 secs: 232736 bytes => GET /static/main.js
HTTP/1.1 200 0.55 secs: 768 bytes => GET /
HTTP/1.1 200 0.56 secs: 232736 bytes => GET /static/main.js
HTTP/1.1 200 0.56 secs: 232736 bytes => GET /static/main.js
HTTP/1.1 200 0.54 secs: 232736 bytes => GET /static/main.js
HTTP/1.1 200 0.56 secs: 232736 bytes => GET /static/main.js
HTTP/1.1 200 0.45 secs: 768 bytes => GET /
HTTP/1.1 200 0.26 secs: 232736 bytes => GET /static/main.js
HTTP/1.1 200 0.73 secs: 232736 bytes => GET /static/main.js
HTTP/1.1 200 0.26 secs: 232736 bytes => GET /static/main.js
HTTP/1.1 200 0.28 secs: 232736 bytes => GET /static/main.js
HTTP/1.1 200 0.30 secs: 232736 bytes => GET /static/main.js
HTTP/1.1 200 0.27 secs: 232736 bytes => GET /static/main.js
HTTP/1.1 200 0.26 secs: 768 bytes => GET /
HTTP/1.1 200 0.25 secs: 232736 bytes => GET /static/main.js
HTTP/1.1 200 0.15 secs: 232736 bytes => GET /static/main.js
HTTP/1.1 200 0.07 secs: 232736 bytes => GET /static/main.js

Transactions: 5100 hits
Availability: 100.00 %
Elapsed time: 75.56 secs
Data transferred: 567.85 MB
Response time: 3.11 secs
Transaction rate: 67.58 trans/sec
Throughput: 7.52 MB/sec
Concurrency: 210.21
Successful transactions: 5100
Failed transactions: 0
Longest transaction: 18.38
Shortest transaction: 0.00

[root@ovz1 ~]#

```

Рис. 43.5. «Джино»: результат нагрузочного тестирования

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
```

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 такие 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 (системы управления контентом), т. е. siege обращался к веб-серверу, возвращающему тестовую страничку. Если 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

```

[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: 199584008 connection timed out.: Connection timed out
[error] socket: -1746299024 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: -1605224784 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: -1956806144 connection timed out.: Connection timed out
[error] socket: 1861597448 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: -1599351940 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:        98.20 %
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

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
```

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).

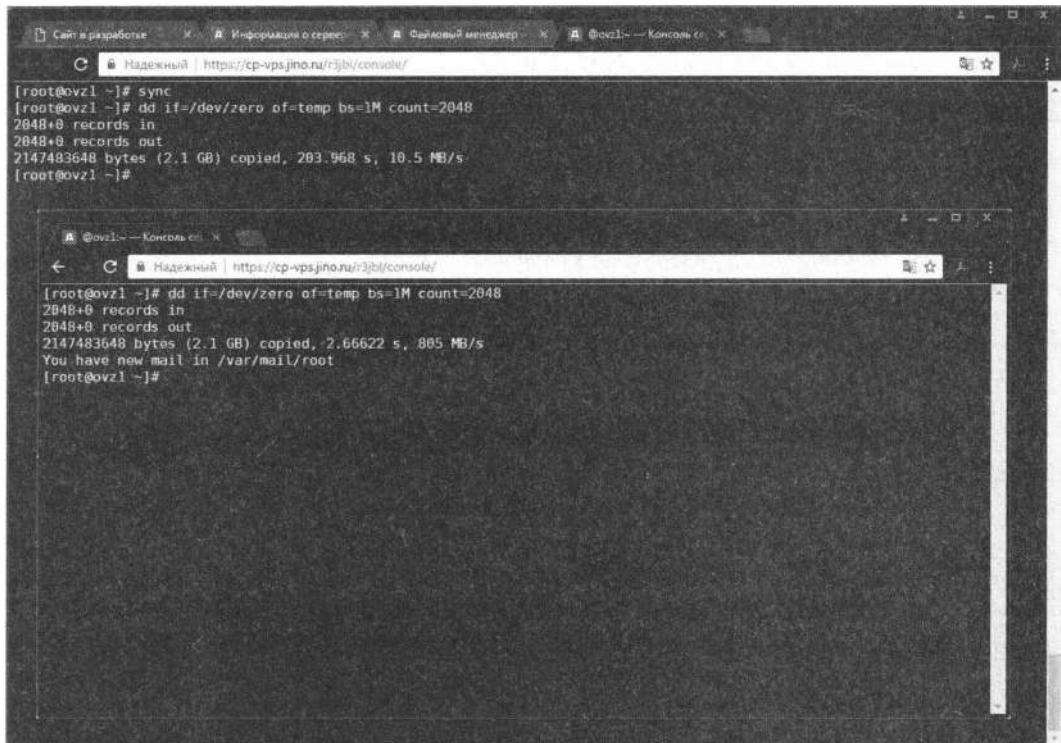
#### ПРИМЕЧАНИЕ

Если бы я использовал не /dev/zero, а «забивал» диск случайными данными (/dev/random), то результаты, возможно, были бы хуже, но не намного.

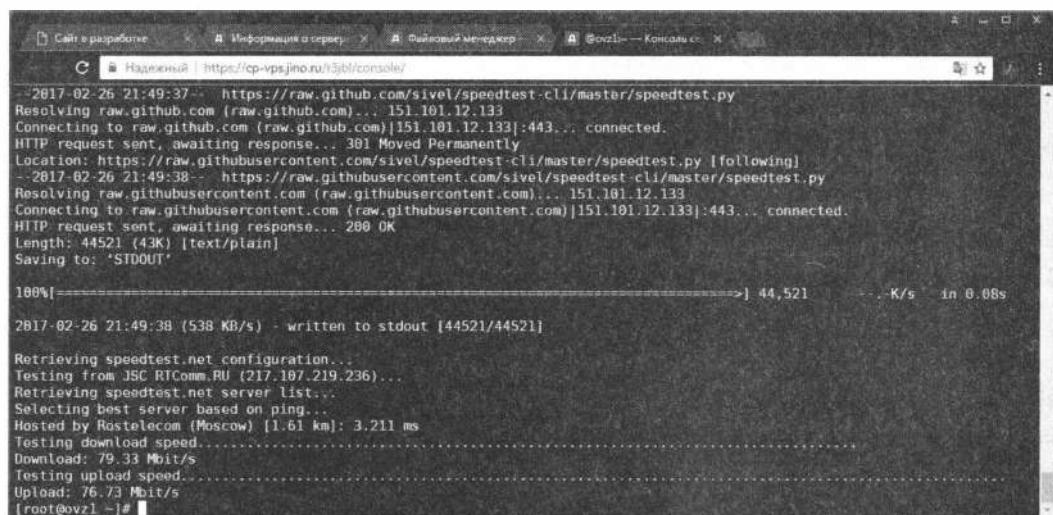
Напоследок посмотрим на пропускную способность, для оценки которой я воспользуюсь консольной версией speedtest.net:

```
$ wget -O - https://raw.github.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



**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

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;
- бесплатная панель собственной разработки;
- неограниченный трафик;
- посutoчная 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, присутствуют кнопки управления сервером. Можно остановить, перезапустить сервер, создать бэкап и даже удалить сервер, если он больше не нужен.

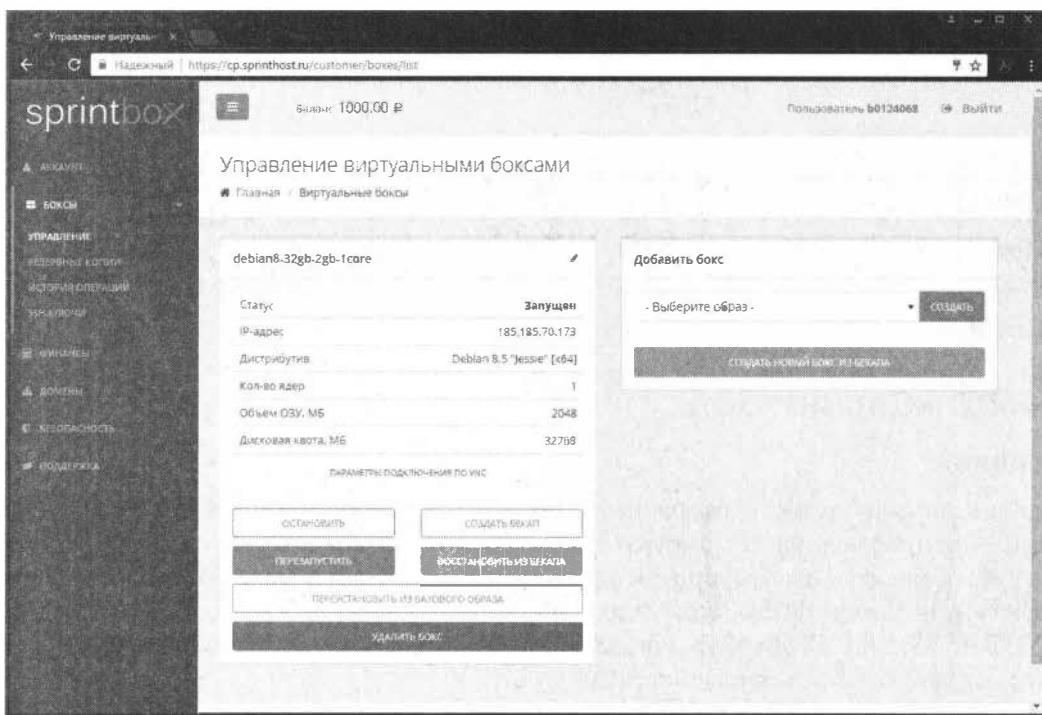


Рис. 43.9. «Спринтхост»: VDS готов к работе

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. Веб-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).

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.

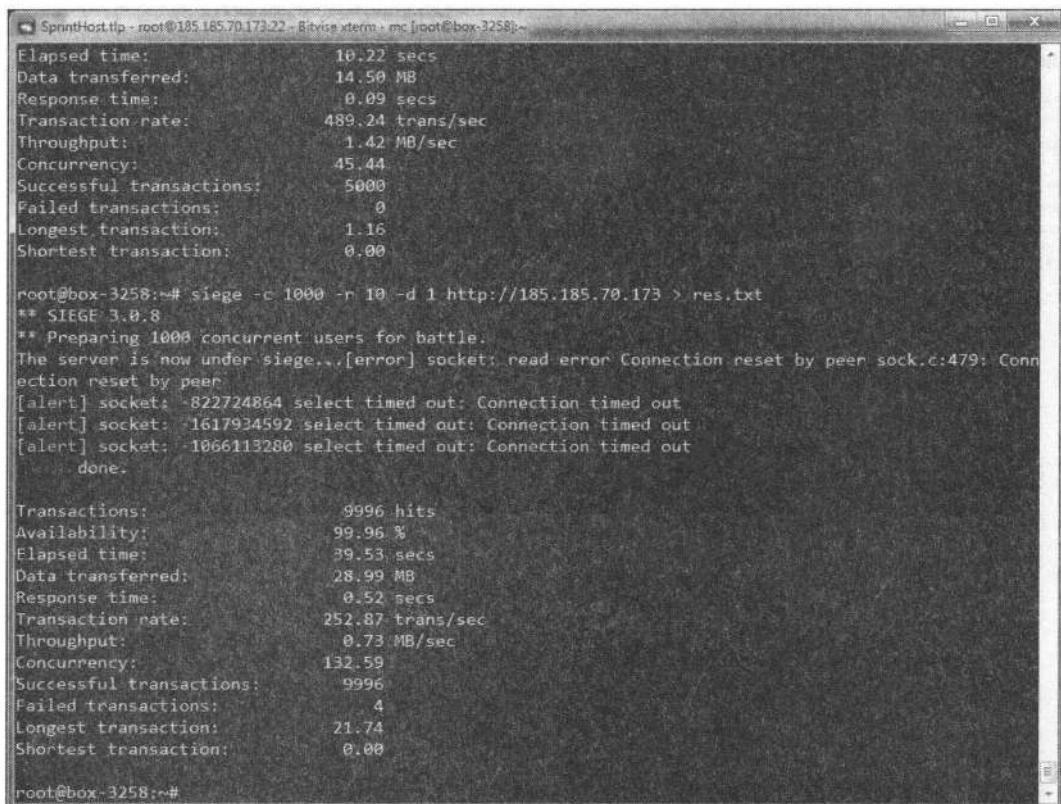
Как вы уже догадались, следующий этап теста посвящен тестированию диска. Команда будет той же самой, чтобы читатели могли сравнить полученные результаты с результатами других провайдеров:

```
dd if=/dev/zero of=temp bs=1M count=2048
```

The screenshot shows a terminal window titled "Sprinthost.tlp" with the command "root@185.185.70.173:22 - Bitvise xterm - mc [root@box-3258]~". The window displays two sets of Siege test results. The first set is for 255 concurrent users, and the second is for 500 concurrent users. Both tests were run against the URL <http://185.185.70.173> and wrote the results to a file named res.txt.

Test Type	Transactions	Availability (%)	Elapsed time (secs)	Data transferred (MB)	Response time (secs)	Transaction rate (trans/sec)	Throughput (MB/sec)	Concurrency	Successful transactions	Failed transactions	Longest transaction (secs)	Shortest transaction (secs)
255 concurrent users	2550	100.00	9.32	7.48	0.03	273.61	0.79	7.66	2550	0	0.39	0.00
	500 concurrent users	48924	100.00	10.22	14.50	0.09	489.24	1.42	45.44	5000	0	1.16

**Рис. 43.10.** «Спринтхост»: результаты нагрузочного тестирования для 255 и 500 одновременных пользователей



```

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.

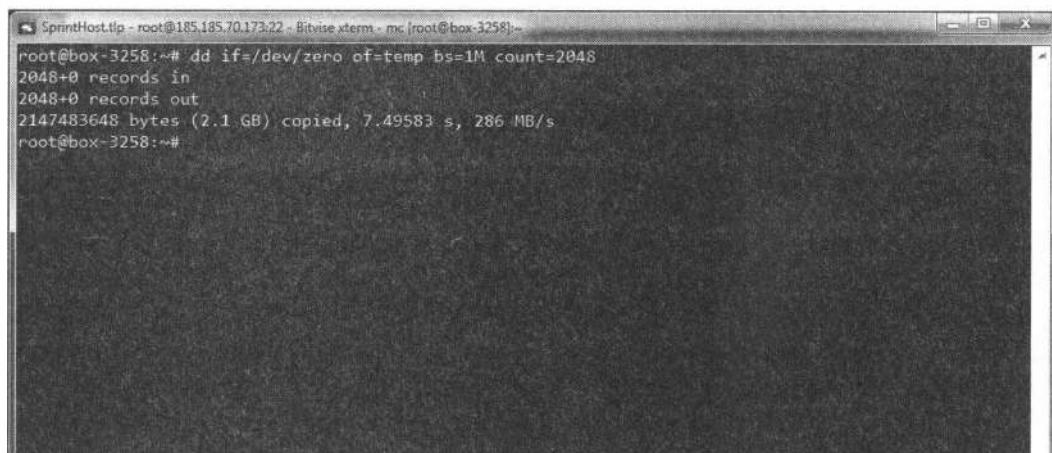
Transactions:          9996 hits
Availability:          99.96 %
Elapsed time:           39.53  secs
Data transferred:       28.99  MB
Response time:          0.52  secs
Transaction rate:      252.87  trans/sec
Throughput:             0.73  MB/sec
Concurrency:           132.59
Successful transactions: 9996
Failed transactions:    4
Longest transaction:   21.74
Shortest transaction:  0.00

root@box-3258:~#

```

**Рис. 43.11.** «Спринтхост»: результаты нагрузочного тестирования — 1000 пользователей

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.



```

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

Наконец, протестируем пропускную способность (рис. 43.13) — speedtest показал практически синхронный канал: 101 Мбит/с **download**, 111 Мбит/с **upload** — хорошие показатели.

```
Sprinthonst.lip - root@185.185.70.173:22 - Bitvise xterm - mc [root@box-3258]~# python speedtest.py
root@box-3258:~# python speedtest.py
Retrieving speedtest.net configuration...
Testing from SPRINTHONST.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	Год	Год	Год
<b>879 ₽ в месяц</b>	<b>1583 ₽ в месяц</b>	<b>2903 ₽ в месяц</b>	<b>5543 ₽ в месяц</b>
при оплате на год			
⊕ Экономия 1439 ₽	⊕ Экономия 2591 ₽	⊕ Экономия 4751 ₽	⊕ Экономия 9071 ₽
<b>Заказать</b>	<b>Заказать</b>	<b>Заказать</b>	<b>Заказать</b>

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. п. 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 «Макхост» 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 (рис. 43.15), 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».

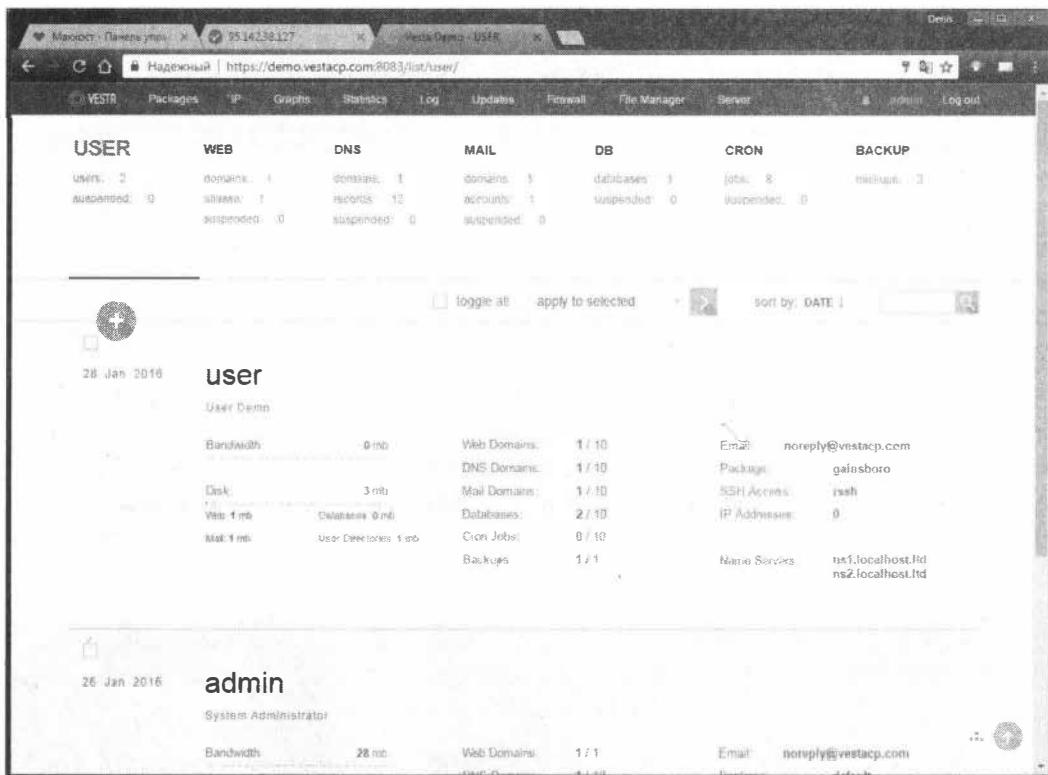


Рис. 43.15. «Макхост»: панель управления Vesta

## 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 «Макхост», я не сделал тест на нагружочное 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 что на спорткар установить

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...

```
[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

Теперь проверим пропускную способность, воспользовавшись консольным скриптом speedtest.py:

```
$ wget -O - https://raw.github.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 радуйтесь: 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.github.com/sivel/speedtest-cli/master/speedtest.py | python
-- 2017-01-29 10:15:30 -- https://raw.github.com/sivel/speedtest-cli/master/speedtest.py
Распознаётся raw.github.com... 151.101.36.133
Устанавливается соединение с raw.github.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 веб-портала, так и для «1С». Желающим 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

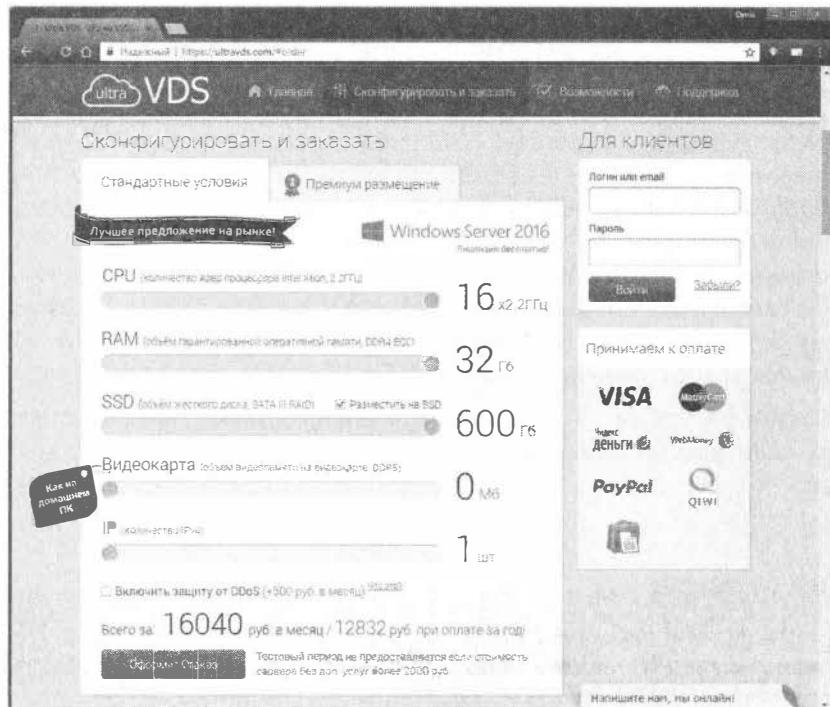
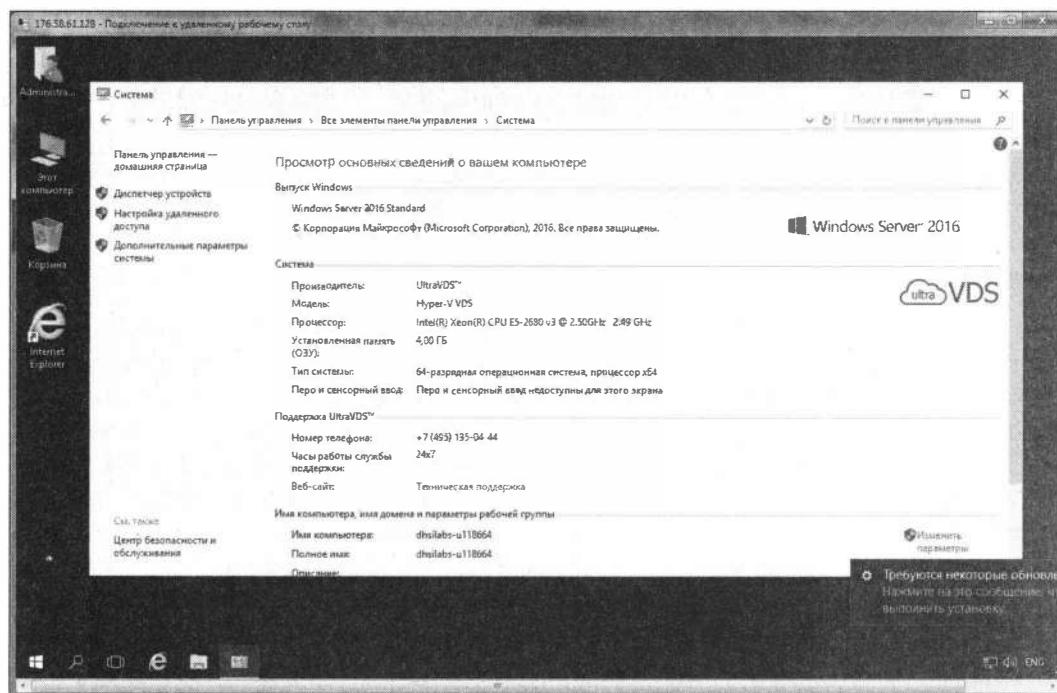


Рис. 43.19. «UltraVDS»: конфигуратор



**Рис. 43.20.** «UltraVDS»: панель управления

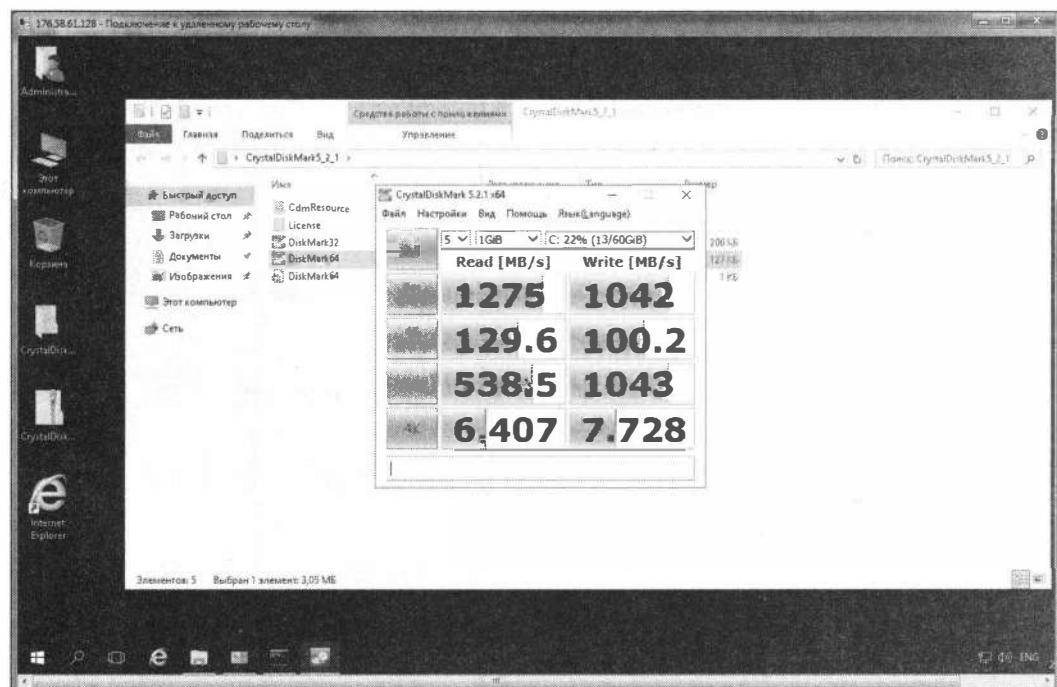


Рис. 43.21. «UltraVDS»: подключение по RDP

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 (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 веб-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 teste на последовательное чтение и 484 Мбайт/с в teste 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 <a href="http://beta.speedtest.net/result/5998827299">http://beta.speedtest.net/result/5998827299</a>	599,73	102,79	3
Rostelecom, Krasnodar <a href="http://beta.speedtest.net/result/5998838629">http://beta.speedtest.net/result/5998838629</a>	258,45	46,71	31
Megafon, Moscow <a href="http://beta.speedtest.net/result/5998841560">http://beta.speedtest.net/result/5998841560</a>	325,34	58,44	20
Rostelecom, Saint Petersburg <a href="http://beta.speedtest.net/result/5998847918">http://beta.speedtest.net/result/5998847918</a>	418,90	69,92	12
DEAC, Amsterdam <a href="http://beta.speedtest.net/result/5998855369">http://beta.speedtest.net/result/5998855369</a>	143,75	50,53	66
Orange, Paris <a href="http://beta.speedtest.net/result/5998852439">http://beta.speedtest.net/result/5998852439</a>	202,57	59,96	54
Rostelecom, Yaroslavl <a href="http://beta.speedtest.net/result/5998859940">http://beta.speedtest.net/result/5998859940</a>	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.

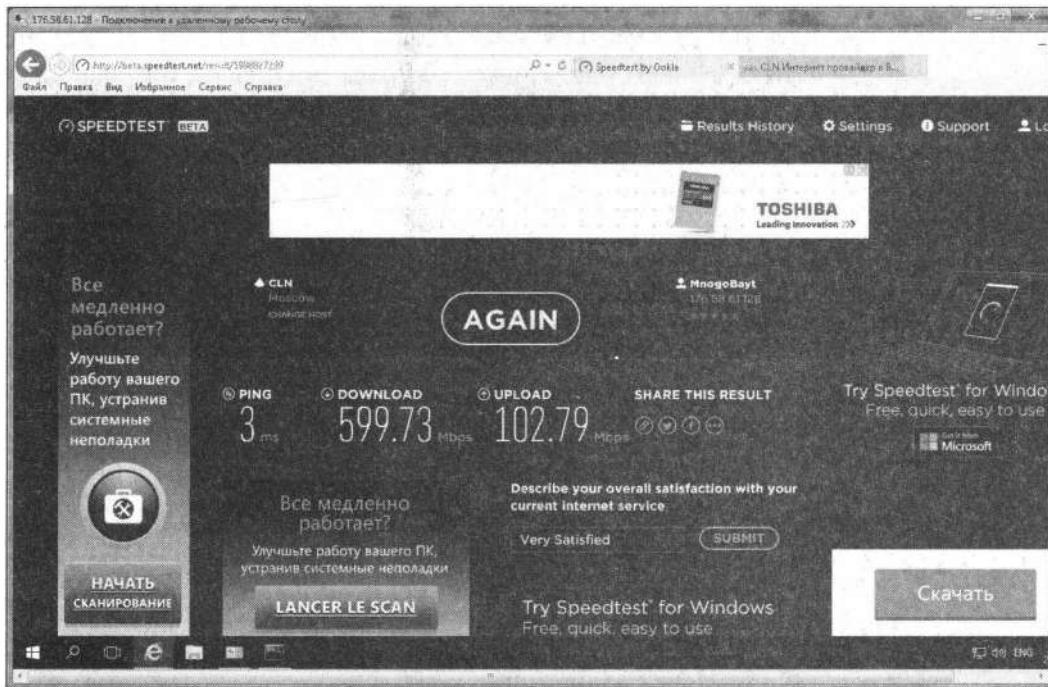


Рис. 43.22. «UltraVDS»: тестирование пропускной способности (CLN, Moscow)

## 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 нагружочных тестов действия этих лимитов не выявлены.

## 43.2.5. Облачный сервис «1cloud»

### О ценах

У «1cloud» 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.

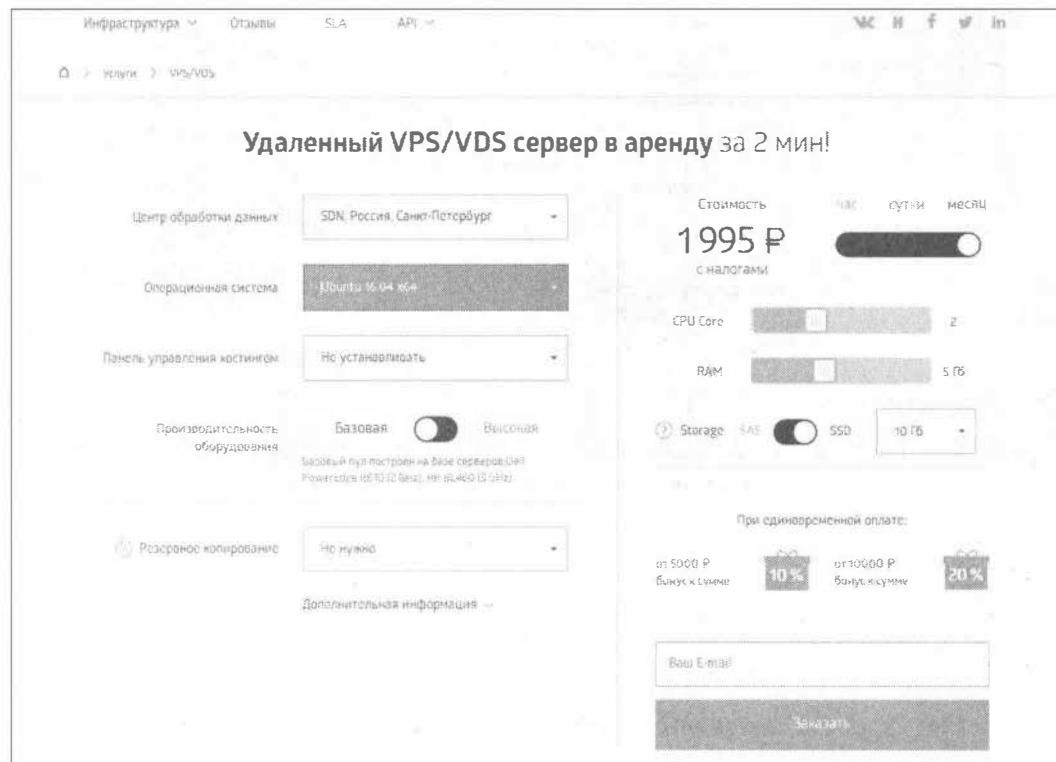


Рис. 43.23. «1cloud»: стоимость сервера

При базовой производительности сервер будет помещен в пул, построенный на основе 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 станет 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.

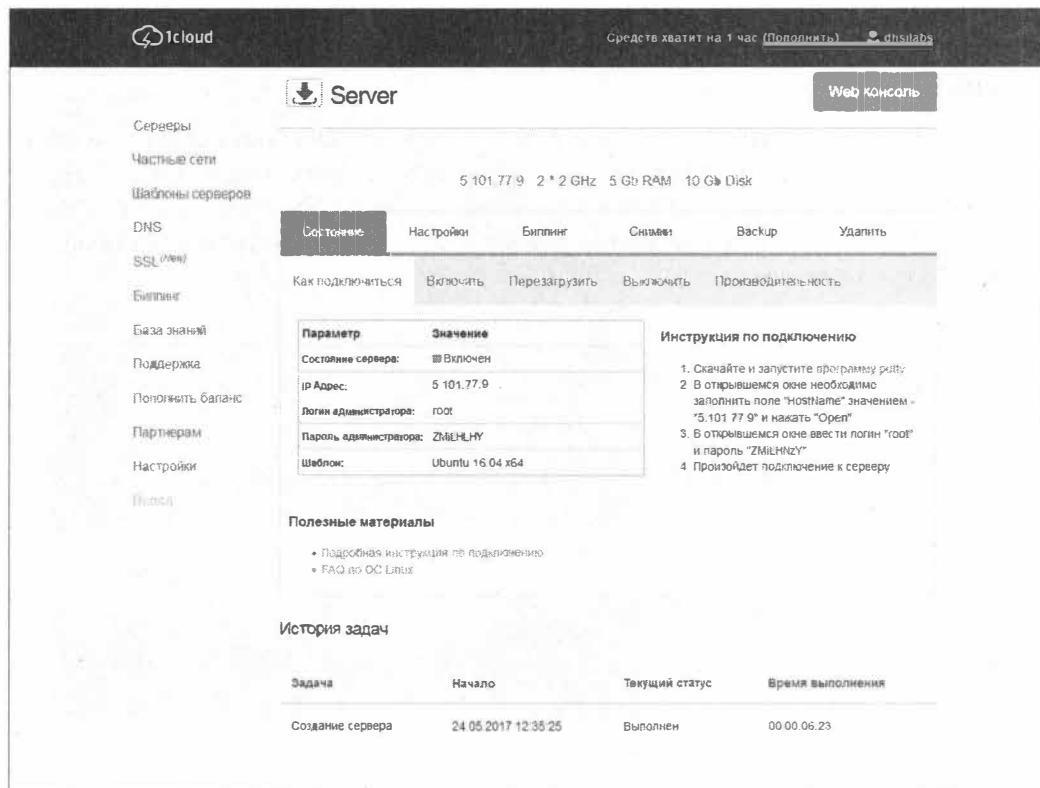


Рис. 43.24. «1cloud»: панель управления услугой

## Тестирование

Тест строился по привычной схеме: пропускная способность, производительность SSD и нагружочное тестирование. Так вот, получив весьма посредственные результаты пропускной способности (рис. 43.25), я уже 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 руб./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 нагружочного 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

```
root@Ubuntu1604x64:~# python speedtest.py
Retrieving speedtest.net configuration...
Testing from IT-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 IT-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: 30.38 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)

доступность 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 [1cloud.ru](http://1cloud.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
[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 «1cloud»:

- 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 запущенная конфигурация не используется на 100%, можно временно ее упростить, 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. п.;
- возможность бесплатного тестирования.

Недостатки:

- высокая стоимость;
- низкая пропускная способность по умолчанию, а за дополнительную пропускную способность нужно доплачивать.

### 43.3. Сравнительная таблица

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»	«Маххост»	«UltraVDS»	«1cloud»
ТТХ	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 способность, 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

## 43.4. Сразу после покупки виртуального Linux-сервера. Шесть шагов к безопасности сервера

Рассмотрим небольшой чек-лист, который нужно сделать после покупки виртуального сервера. Игнорирование этих простых правил приведет к серьезной дыре в безопасности системы.

### 43.4.1. Меняем пароль пользователя root

Как правило, после заказа сервера вам сообщают его IP-адрес и пароль пользователя. Причем пароль пользователя виден в панели управления виртуальным сервером. Если пароль сгенерировал провайдер, следовательно, он его знает и хранит в базе данных — ведь пароль не просто сгенерировали, его где-то сохранили, чтобы показывать вам каждый раз в панели управления. К тому же пароли от сервера обычно генерируются случайным образом и сложны для подбора, а вот какой пароль указали вы для панели управления — тот еще вопрос. Поэтому первым делом меняем пароль root. Считайте это не паранойей, а простой предосторожностью. И не забываем менять его каждые 2–3 месяца — также из соображений безопасности. Для этого войдите в систему как root и введите команду:

```
passwd
```

### 43.4.2. Создаем обычного пользователя

Администрация сервера сообщает пользователю пароль root для приобретенного им VDS, после чего он обычно копирует его в профиль своего SSH-клиента. Но это в корне неправильно! Во-первых, учетная запись root — это так называемое всем известное имя, и когда злоумышленник попытается взломать ваш сервер методом грубой силы, он будет пытаться скомпрометировать именно учетную запись root. Все просто — каких пользователей вы создали, он не знает, а root есть у всех. Поэтому нам нужен обычный пользователь, под которым мы будем работать постоянно. Создаем пользователя (имя user вы здесь можете изменить на свое собственное):

```
adduser user  
passwd user
```

Первая команда добавляет учетную запись с именем user, вторая — изменяет пароль для нее.

Чтобы этот пользователь мог использоваться как root, нужно добавить его в файл sudoers, но сначала нам надо выполнить один промежуточный шаг.

### 43.4.3. Установка удобного редактора

Используемый по умолчанию во многих дистрибутивах Linux редактор vi неудобный и требует для работы с ним специальных знаний, что немножко шокирует начи-

нающих пользователей. Редактор был разработан в 70-х годах прошлого века, и только в 2020 году Fedora (в версии 33) отказалась от его использования по умолчанию.

Итак, установите любой текстовый редактор, который вам нравится. Я же предлагаю установить файловый менеджер MC — в его составе есть очень удобный редактор mcedit:

```
# apt install mc
```

Запустить собственно редактор можно командой `mcedit`. Смотрим, где находится этот редактор:

```
which mcedit
```

Обычно это каталог `/usr/bin`. Теперь надо сделать так, чтобы редактор `mcedit` использовался в качестве редактора по умолчанию. В современной Ubuntu для этого выполните команду:

```
update-alternatives --config editor
```

После чего вам нужно просто выбрать `mcedit` из списка предлагаемых вариантов. Если же это способ не сработает (в вашем дистрибутиве нет команды `update-alternatives`), можно пойти простым путем (по старинке) и назначить переменную окружения `EDITOR`:

```
export EDITOR=/bin/mcedit
```

Чтобы переменная окружения `EDITOR` устанавливалась при каждом входе в систему, отредактируйте файл `.bashrc`, который находится в домашнем каталоге пользователя. В него нужно добавить команду:

```
export EDITOR=/bin/mcedit
```

Обратите внимание, что это нужно сделать для каждого пользователя. Для пользователя `root` его домашний каталог — `/root`. Для пользователя `denis` — `/home/denis`.

После выполненных настроек все команды, вызывающие редактор по умолчанию, будут обращаться к удобному редактору `mcedit`. Одна из таких команд: `visudo`.

#### 43.4.4. Превращаем обычного пользователя в администратора

Ранее мы создали обычного пользователя `user`. Но чтобы вы могли полноценно использовать его учетную запись, нужно превратить этого пользователя в администратора, т. е. предоставить ему возможность использовать команду `sudo`. Для этого выполните команду:

```
visudo
```

Команда запустит текстовый редактор для изменения файла конфигурации `sudoers`. Найдите и раскомментируйте в нем строку:

```
wheel    ALL=(ALL)      ALL
```

Эта строка разрешает всем пользователям группы `wheel` использовать команду `sudo`. Если эта строка уже раскомментирована, ничего делать не нужно. Нажмите для выхода из редактора клавишу `<F10>`.

После этого нужно добавить нашего пользователя `user` в группу `wheel`:

```
usermod -aG wheel user
```

Теперь отключитесь от SSH-сервера и попытайтесь войти как пользователь `user` с ранее заданным паролем. Обратите внимание, что, когда вы работали как пользователь `root`, приглашение командной строки имело вид `#`, теперь же оно выглядит так: `$`. Введите команду:

```
sudo bash
```

Если приглашение командной строки поменялось на `#`, значит, теперь у вас появились права `root`.

#### 43.4.5. Запрещаем вход как root по SSH

Дело осталось за малым — запретить пользователю `root` входить по SSH. Для этого, работая уже под новым пользователем `user`, введите команду:

```
sudo mcedit /etc/ssh/sshd_config
```

Добавьте в конфигурационный файл сервера SSH строку (или раскомментируйте ее):

```
PermitRootLogin no
```

Сохраните файл конфигурации и попробуйте подключиться как пользователь `root` — у вас это не получится.

#### 43.4.6. Настройка брандмауэра

Разобравшись с безопасностью пользователей, настройте брандмауэр. Традиционно в качестве брандмауэра (фильтра пакетов) в Ubuntu используется `iptables`, но поскольку Ubuntu позиционируется как простой дистрибутив, то и оболочка для `iptables` была разработана соответствующая — `UFW` (Uncomplicated Firewall), что буквально означает «несложный файрвол». Первым делом нужно убедиться, что пакет `ufw` вообще установлен, или установить его, если это не так:

```
sudo apt install ufw
```

##### Базовая настройка

Теперь посмотрим состояние брандмауэра:

```
sudo ufw status verbose
```

По умолчанию брандмауэр выключен, поэтому вы получите сообщение:

```
Status: inactive
```

Не спешите включать брандмауэр — сначала его надо настроить. Ведь если порт 22 окажется по умолчанию недоступен, вы потеряете доступ к своему VDS. Конечно, в службе поддержки вам помогут, но все это — потеря времени.

По умолчанию брандмауэр запрещает все входящие соединения с сервером и разрешает все исходящие. Такая политика идеальна с точки зрения безопасности — ведь если кто-то (и вы в том числе) захочет подключиться к серверу, у него это не получится. В то же время приложения на сервере смогут создавать исходящие соединения.

Рассмотрим две команды:

```
ufw default deny incoming  
ufw default allow outgoing
```

Эти два правила как раз и задают политику по умолчанию: запрещаются все входящие соединения и разрешаются все исходящие.

Итак, все входящие соединения запрещены. И чтобы до сервера можно было «достучаться» по определенному порту, его нужно сначала открыть. UFW хорош тем, что вам даже не нужно помнить номер порта — нужно знать только название сервиса. Например, вот как можно разрешить подключение по SSH:

```
ufw allow ssh
```

При этом UFW сам создаст правило для порта 22 — именно этот порт используется для SSH. Брандмауэр знает порты и имена всех распространенных служб (HTTP, SSH, FTP, SFTP и т. д.).

Однако если вы перенастроили SSH на нестандартный порт из соображений той же безопасности, следует явно указать номер порта:

```
ufw allow 3333
```

Вот теперь, разрешив подключение по SSH (а это главное, чтобы брандмауэр не разорвал соединение), можно включить UFW командой:

```
ufw enable
```

И подтвердить запуск, ответив нажатием клавиши <y> на запрос:

**Command may disrupt existing ssh connections. Proceed with operation (y|n) ?**

Посмотрите на рис. 43.29 — он иллюстрирует выполненные нами действия. Сначала мы установили собственно брандмауэр:

```
apt install ufw
```

Затем запросили его статус:

```
ufw status verbose
```

После чего разрешили SSH:

```
ufw allow ssh
```

на что получили ответ, что правила обновлены:

**Rules updated**

The screenshot shows a terminal window with the following command history:

```
root@Ubuntu1804x64:~# apt install ufw
Reading package lists... Done
Building dependency tree
Reading state information... Done
ufw is already the newest version (0.35-5).
0 upgraded, 0 newly installed, 0 to remove and 1 not upgraded.
root@Ubuntu1804x64:~# ufw status verbose
Status: inactive
root@Ubuntu1804x64:~# ufw allow ssh
Rules updated
Rules updated (v6)
root@Ubuntu1804x64:~# ufw enable
Command may disrupt existing ssh connections. Proceed with operation (y|n)? y
Firewall is active and enabled on system startup
root@Ubuntu1804x64:~#
```

Рис. 43.29. Брандмауэр UFW: базовая настройка

И в завершение мы включили брандмауэр:

```
ufw enable
```

и получили сообщение, что он активен и будет запускаться при загрузке системы.

На этом базовая настройка выполнена: SSH успешно работает, и мы можем приступить к дальнейшей настройке брандмауэра как фильтра пакетов.

## Создание правил для сервисов

Разрешим работу других приложений. Как правило, нужно разрешить службу HTTP (веб-сервер), FTP (если этот сервис вам нужен) и постараться не забыть о HTTPS (что очень важно в последнее время):

```
ufw allow http
ufw allow https
ufw allow ftp
```

Сделать то же самое можно было бы и по номерам портов:

```
ufw allow 80
ufw allow 443
ufw allow 21
```

При желании можно разрешить целый диапазон портов, указав при этом транспортный протокол (UDP или TCP):

```
sudo ufw allow 2000:2200/tcp  
sudo ufw allow 4000:4400/udp
```

## Разрешаем IP-адреса

UFW позволяет разрешить определенному IP-адресу (например, 46.229.220.16) доступ ко всем портам сервера:

```
ufw allow from 46.229.220.16
```

Если нужно разрешить доступ какому-либо IP-адресу (например, тому же 46.229.220.16) только к конкретному порту, то делается это так:

```
ufw allow from 46.229.220.16 to any port 22
```

Здесь мы разрешаем не все подключения к SSH, а только подключения с IP-адреса 46.229.220.16.

Разрешить доступ целого диапазона IP-адресов (например, когда у админа динамический IP-адрес), можно так:

```
ufw allow from 123.45.67.89/24 to any port 22
```

## Запрещаем IP-адреса и службы

Запретить доступ с определенного IP-адреса можно аналогично:

```
ufw deny from 123.45.67.89
```

При желании можно запретить все подключения к определенной службе:

```
ufw deny ftp
```

## Удаление/сброс правил

Сбросить все правила можно командой:

```
ufw reset
```

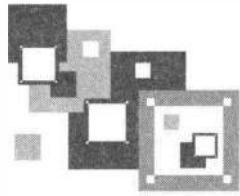
Но убедитесь, что на момент ввода этой команды вы уже отключили брандмауэр, иначе вы потеряете доступ к серверу по SSH.

Удалить конкретное правило можно по его номеру. Для этого сначала введите следующую команду, чтобы узнать номер правила:

```
ufw status numbered
```

А уже затем удаляйте его:

```
sudo ufw delete <номер_правила>
```



## 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 компаний. 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 веб-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-ю гигабайтами оперативной памяти, которую вы желаете разделить между виртуальными серверами. Пусть каждому серверу вы планируете выделить по одному гигабайту — 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](http://kernel.org)) — следовательно, вам понадобится специальное ядро OpenVZ. Само ядро — не проблема, проблема в его версии. Несмотря на то, что уже вышла 5-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 веб-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 достаточно «свежие» дистрибутивы: Fedora 22, OpenSUSE 13.2. Однако не забывайте, что приведенные 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
<b>Установка:</b>			
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
<b>Установка зависимостей:</b>			
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
<b>Результат операции</b>			
Установить	9 пакет(а,ов)		
Объем загрузки: 31 М			
Будет установлено: 126 М			
Продолжить? [y/N]: █			

Рис. 44.1. Процесс установки OpenVZ

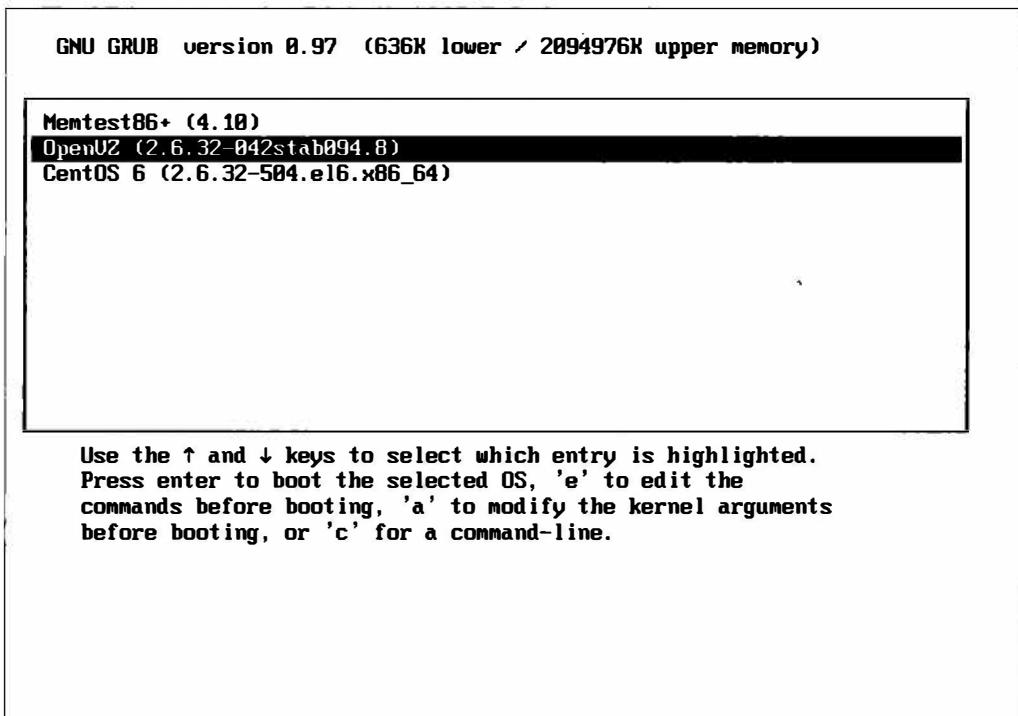


Рис. 44.2. Загрузка сервера виртуализации

## 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 23), 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 vsl.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). Мне 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 рассматриваемом 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.

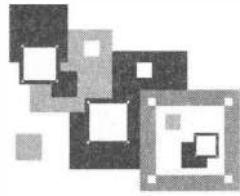
Что делать дальше? Войти в виртуальный сервер по `ssh` и приступить к его настройке так, как если бы это был самый обычный физический сервер:

```
# ssh <IP-адрес>
```

The screenshot shows a terminal window titled "den@host-server:/home/den". The window contains the following command-line session:

```
[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, борясь 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 веб-серверы и продавать их (типичное решение для VPS-провайдера).

Схема виртуализации, представленная на рис. 45.1, позаимствована из документации по Virtuozzo. Как можно видеть, здесь присутствует «железо» сервера (**Server Hardware**), выделены уровни виртуализации (**OS Virtualization Layer**) и контейнеры (**Container**).

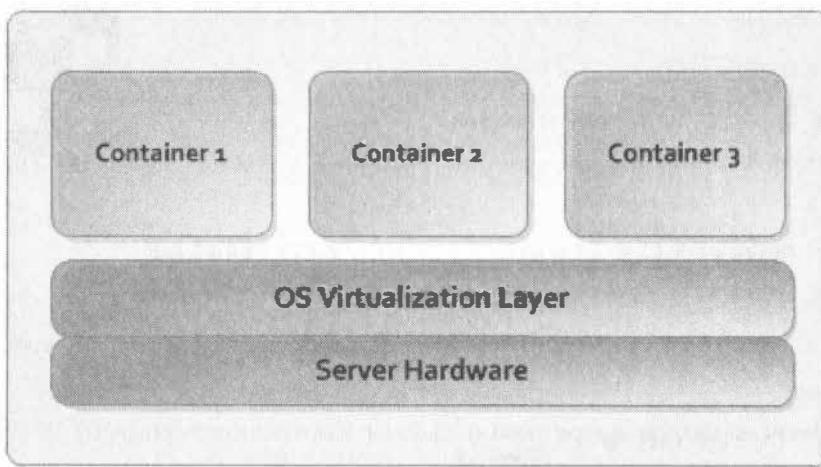


Рис. 45.1. Схема виртуализации Virtuozzo

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 vmscap.py | grep -i unrest
```

В результат 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 partitioning** и нажмите кнопку **Done** (рис. 45.3).

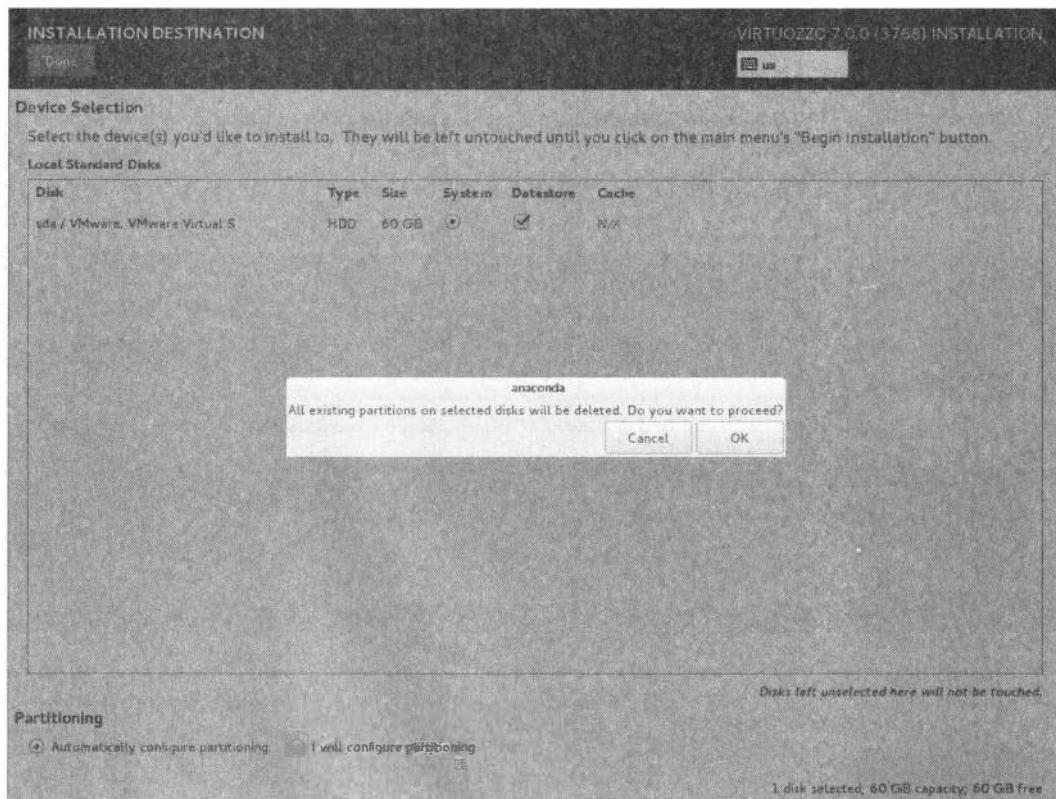


Рис. 45.3. Virtuozzo: выбор способа разметки диска

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).

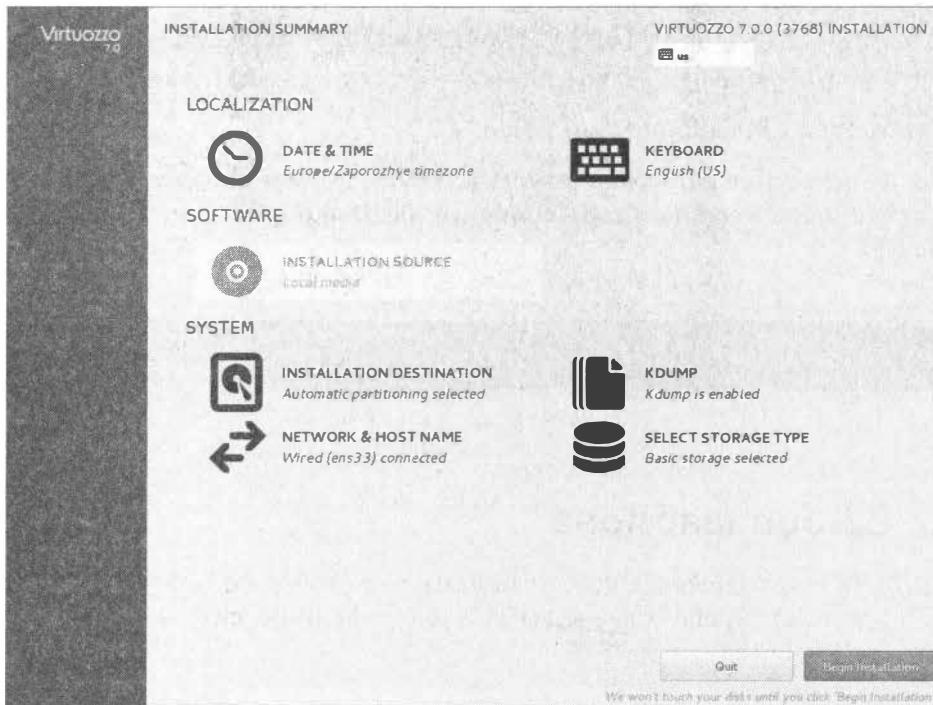


Рис. 45.4. Virtuozzo: нажмите кнопку **Begin Installation**

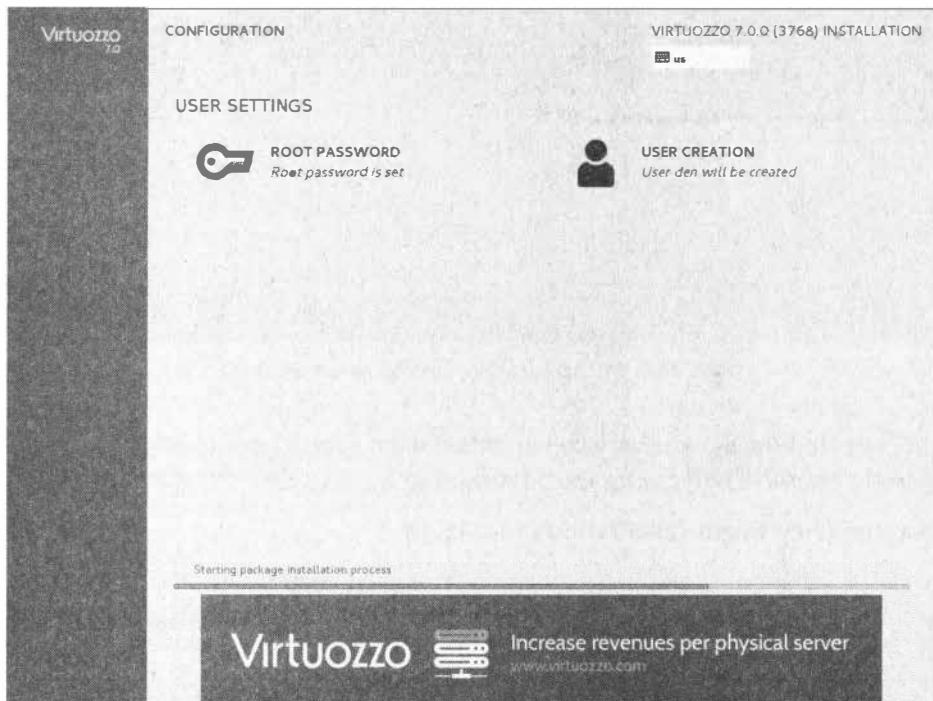


Рис. 45.5. Virtuozzo: задание пароля root и создание обычного пользователя

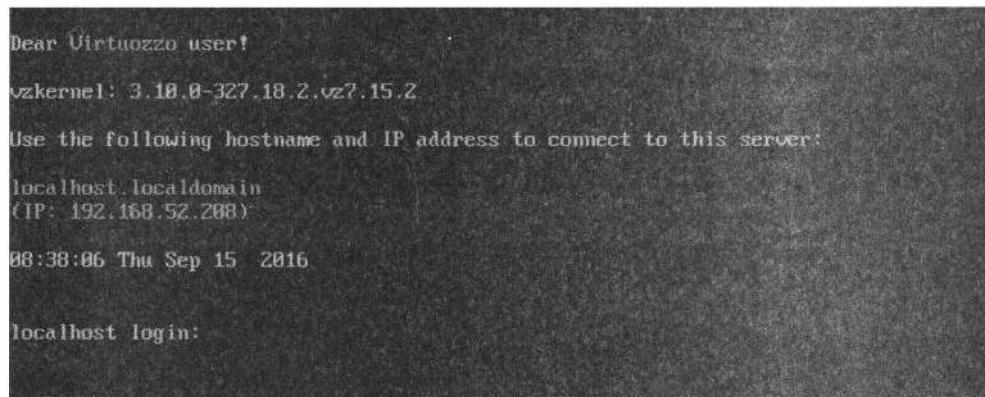


Рис. 45.6. Virtuozzo: вход в систему

## 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 For Ubuntu 16.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64	phpadmin For Ubuntu 16.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64	jre For Ubuntu 16.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64	cyrus-imap For Ubuntu 16.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64	imap For Ubuntu 16.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64	devel For Ubuntu 16.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64	spamassassin For Ubuntu 16.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64	mysql For Ubuntu 16.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64	phpadmin For Ubuntu 16.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64	jdk For Ubuntu 16.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64	mod_perl For Ubuntu 16.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64	squidmail For Ubuntu 16.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64	mailman For Ubuntu 16.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64	webalizer For Ubuntu 16.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64	tomcat For Ubuntu 16.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64	wordpress For Ubuntu 16.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64	profptd For Ubuntu 16.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-16.04-x86_64	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 For Ubuntu 14.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64	phpadmin For Ubuntu 14.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64	jre For Ubuntu 14.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64	cyrus-imap For Ubuntu 14.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64	imap For Ubuntu 14.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64	devel For Ubuntu 14.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64	spamassassin For Ubuntu 14.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64	mysql For Ubuntu 14.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64	phpadmin For Ubuntu 14.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64	jdk For Ubuntu 14.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64	mod_perl For Ubuntu 14.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64	squidmail For Ubuntu 14.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64	mailman For Ubuntu 14.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64	webalizer For Ubuntu 14.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64	tomcat For Ubuntu 14.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64	wordpress For Ubuntu 14.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64	profptd For Ubuntu 14.04 (for AMD64/intel EM64T) Virtuozzo Template
ubuntu-14.04-x86_64	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 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	jdk For Centos 7 (for AMD64/intel EM64T) Virtuozzo Template

Рис. 45.7. Virtuozzo: 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          :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          :jdk 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          :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          :jdk 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. Virtuozzo: 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 ~]# vctl 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. Virtuozzo: 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
PHYS PAGES="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-e16"
OSRELEASE="2.6.32"
VEID="6ba9cd71-84fd-4e5e-b5d1-cdd73dfa0ea3"
UUID="6ba9cd71-84fd-4e5e-b5d1-cdd73dfa0ea3"
```

Рис. 45.10. Virtuozzo: 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 к неудобным идентификаторам 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 --cpunits 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):

```
PHYS PAGES="65536:65536"  
SWAPPAGES="65536"
```

Изменить размер виртуального диска позволяет команда **prl\_disk\_tool**:

```
prl_disk_tool resize --hdd /vz/vmpriate/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
```

```
[root@localhost ~]# prlctl set MyCT --cpus 1
set cpus(2): 1

The CT has been successfully configured.
[root@localhost ~]# prlctl set MyCT --memsize 256M
Set the memsize parameter to 256Mb.

The CT has been successfully configured.
[root@localhost ~]# prlctl set MyCT --hostname myct.example.com

The CT has been successfully configured.
[root@localhost ~]# prlctl 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.
[root@localhost ~]# ...
```

Рис. 45.11. Virtuozzo: 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:

```
# prlctl start MyCT
```

Запустив контейнер, сразу вводим команду просмотра состояния `prlctl 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:

```
# prlctl stop MyCT
# prlctl restart MyCT
```

Для удаления контейнера его нужно сначала остановить, а потом удалить:

```
# prlctl stop MyCT
# prlctl delete MyCT
```

```
[root@localhost ~]# prlctl start MyCT
Starting the CT...
The CT has been successfully started.
[root@localhost ~]# prlctl list -a
UUID STATUS IP_ADDR T NAME
{6ba9cd71-84fd-4e5e-b5d1-cdd73dfa0ea3} 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. Virtuozzo: 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):

```
# prlctl exec MyCT passwd
```

```
[root@localhost ~]# prlctl exec MyCT passwd
New password:
Retype new password:
Changing password for user root.
passwd: all authentication tokens updated successfully.
[root@localhost ~]#
```

Рис. 45.13. Virtuozzo: 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 cc:1a:8b:bf:d4:cb:03:ff:d8: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@myct ~]# df -H
Filesystem      Size  Used Avail Use% Mounted on
rootfs          11G   715M  9.2G  8% /
/dev/ploop3?668p1 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@myct ~]# free
              total        used        free      shared  buffers   cached
Mem:       262144      42664     219488          0        48        0    26784
/+ buffers/cache:  158880     246264
Swap:      262144          0     262144
[root@myct ~]#
```

Рис. 45.14. Virtuozzo: подключение к контейнеру по ssh

После установки ssh-подключения можно вводить команды без префикса `prlctl 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.

К этому моменту 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 единичка), 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 подсетки 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 обращайте 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.vl7.2.x86_64 already installed and latest version
Nothing to do
[root@localhost ~]# cat /proc/sys/net/ipv4/ip_forward
1
[root@localhost ~]# prlctl 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 192.168.52.181
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 7534ms
rtt min/avg/max/mdev = 52.412/88.497/274.465/73.316 ms
[root@first ~]# exit
logout
Connection to 192.168.52.101 closed.
[root@localhost ~]# prlctl stop first
Stopping the CT...
The CT has been successfully stopped.
[root@localhost ~]# prlctl set first --nameserver 8.8.8.8

The CT has been successfully configured.
[root@localhost ~]# prlctl start first
Starting the CT...
The CT has been successfully started.
[root@localhost ~]#

```

**Рис. 45.15. Virtuozzo: 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:

```
# prlctl 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 досгучаться:

```
# 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. Так же 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 12:46:08 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.** Virtuozzo: 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 eth0 -j DNAT --to-destination ve_address:80
# iptables -t nat -A POSTROUTING -s ve_address -o eth0 -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>.

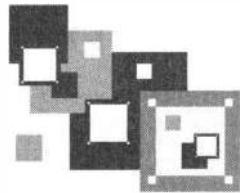
## 45.11. Делаем работу с Virtuozzo удобнее

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, так как Virtuozzo Linux содержит весьма богатые репозитории, из которых вы можете установить свои любимые программы. Я, например, установил mc:

```
# yum install mc
```

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

# Сервер виртуальной частной сети

### 46.1. Настройка собственного VPN-сервера

Здесь не будет длинного и скучного введения на тему, что такая виртуальная частная сеть (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 здесь производится на примере Ubuntu, но в других дистрибутивах все настраивается аналогично, — будут различаться разве что команды установки пакетов и, возможно, пути к некоторым файлам конфигурации.

Прежде, чем приступать к работе, нужно понимать, что именно мы будем настраивать. Виртуальную частную сеть можно организовать по-разному, соответственно, настройка VPN-сервера тоже будет различна, — в зависимости от преследуемой цели. Можно, например, соединить несколько филиалов организации одной большой виртуальной сетью. 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), в организации и создается VPN-сервер. При этом 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. Вернее, злоумышленник, «работающий» в сети отеля, даже если и перехватит ваш трафик, толку от него получит немного — ведь трафик будет зашифрован.

К этому моменту мы предполагаем, что вы уже выполнили начальную настройку компьютера, работающего под управлением Ubuntu: настроили интернет-соединение, правила брандмауэра и т. п.

## 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

OpenVPN использует протоколы TLS/SSL, поэтому для шифрования трафика между сервером и клиентом нам понадобятся соответствующие сертификаты. Чтобы не покупать эти 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 def1 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 пересыпать 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 надо 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
```

Здесь 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 2019-08-11 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 udp

# 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).

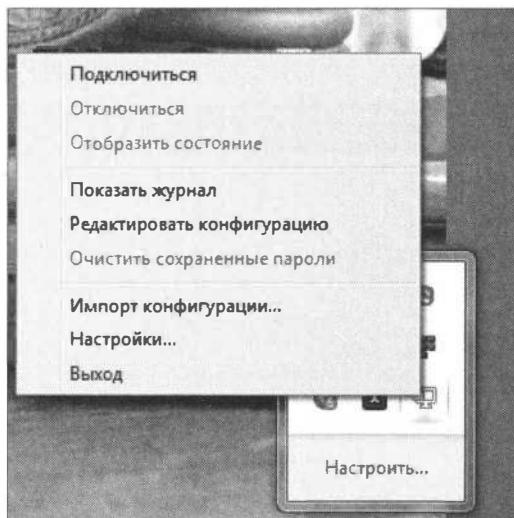


Рис. 46.1. OpenVPN для Windows

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`:

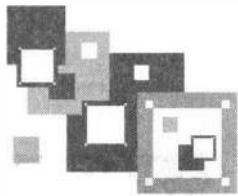
```
group nobody
```

Теперь подключитесь к VPN-серверу:

```
sudo openvpn --config user1.ovpn
```

Собственно, на этом настройка сервера и клиентов окончена — пользоваться Интернетом стало немного безопаснее!

# ГЛАВА 47



## Виртуальные диски на виртуальном сервере

Рано или поздно ваш виртуальный сервер попросит «апгрейда». Если добавить оперативную память — это дело двух щелчков мышью, то с накопителями все сложнее. Расширить существующий диск или добавить еще один в панели управления виртуальным сервером так же просто, как и оперативную память. Сложность операции заключается в том, что операционная система не видит такую модернизацию, пока ей явно не укажешь.

### 47.1. Добавление еще одного виртуального диска

Первым делом нужно добавить еще один диск в панели управления виртуальным сервером. Как это сделать, зависит от самой панели. Как правило, нужно выбрать сервер, перейти к списку его дисков и добавить еще один. Конкретные действия можно просмотреть в справочной системе по панели управления или, в крайнем случае, обратиться в техническую поддержку — там точно подскажут, как добавить еще один диск. Как правило, для добавления нового устройства придется выключить сервер. Панель управления предложит это сделать самостоятельно перед добавлением диска или же перезагрузит сервер в процессе операции добавления — все как на настоящем сервере. Автор знает о «горячей замене», но на виртуальных серверах она работает не всегда.

После добавления диска необходимо подключиться к серверу. Как вы это сделаете — через веб-интерфейс консоли или же через SSH — разницы нет.

Посмотрим, какие диски у нас теперь есть. Для этого введите команду:

```
fdisk -l
```

Вывод будет примерно таким (рис. 47.1) — он зависит от размеров созданных дисков.

Из вывода видно, что есть два диска: **/dev/sda** и **/dev/sdb**. Первый является загрузочным, а второй — с размером **10 GiB** — это и есть диск, который мы только что добавили.

Консоль доступа к серверу: Server

Текущий статус: Соединение установлено

```
root@ubuntu1004:~# fdisk -l
I: 220.5710051 print_req_error: I/O error, dev fd0, sector 0
I: 220.5949771 print_req_error: I/O error, dev fd0, sector 0
Disk /dev/sda: 30 GiB, 32212254720 bytes, 62914560 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
Disklabel type: dos
Disk identifier: 0x4b30d60f

Device      Boot   Start     End   Sectors  Size Id Type
/dev/sda1    *       2048  1953791  1951744  953M 83 Linux
/dev/sda2        1953792 6289625 60936834 29.1G 8e Linux LVM

Disk /dev/sdb: 10 GiB, 10737418240 bytes, 20971520 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 /dev/mapper/vgroup1-root: 28.1 GiB, 30173822976 bytes, 58933248 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
```

**Рис. 47.1.** Используем веб-интерфейс консоли: список дисков и разделов

На втором диске нужно создать разметку. Для этого введите команду:

`fdisk /dev/sdb`

Основные команды `fdisk` (если кто забыл):

- `n` — создать новый раздел;
- `d` — удалить раздел;
- `p` — вывести список разделов;
- `m` — справка;
- `q` — выход без сохранения;
- `w` — сохранить и выйти.

Введите команду `n`. Затем введите тип раздела (`p` — основной), номер раздела (`1` — первый), номер первого сектора (просто нажмите клавишу `<Enter>`), номер последнего сектора (просто нажмите клавишу `<Enter>`). Так мы создали один первичный раздел размером на весь диск (рис. 47.2). Затем введите команду `w` для сохранения изменений и выхода из `fdisk`.

Итак, раздел создан. Осталось создать файловую систему и подмонтировать его. Введите команду:

`mkfs.ext4 /dev/sdb1`

Эта команда создает файловую систему `ext4` на разделе `/dev/sdb1` (рис. 47.3).

```

Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
root@ubuntu1804:~# fdisk /dev/sdb

Welcome to fdisk (util-linux 2.31.1).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Device does not contain a recognized partition table.
Created a new DOS disklabel with disk identifier 0xb245b7df.

Command (m for help): n
Partition type
  p  primary (0 primary, 0 extended, 4 free)
  e  extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1):
First sector (2048-20971519, default 2048):
Last sector, +sectors or +size(K,M,G,T,P) (2048-20971519, default 20971519):

Created a new partition 1 of type 'Linux' and of size 10 GiB.

Command (m for help): w

```

Рис. 47.2. Создан раздел диска

```

Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.

root@ubuntu1804:~# mkfs.ext4 /dev/sdb1
mkfs 1.44.1 (24-Mar-2018)
Creating filesystem with 2621184 4k blocks and 655360 inodes
Filesystem UUID: e5e029af-7253-4d41-b8b6-3091eb7c27f4
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632
Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done

root@ubuntu1804:~#

```

Рис. 47.3. Файловая система создана

Затем нужно подмонтировать диск к какому-нибудь каталогу:

```

mkdir /mnt/sdb1
mount /dev/sdb1 /mnt/sdb1

```

Каталог `/mnt/sdb1` — это точка монтирования (рис. 47.4), он может называться иначе, но должен существовать на момент монтирования диска. Обратиться к файлам нового диска можно будет через этот каталог.

```

Command (n for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.

root@ubuntu1804:~# mkfs.ext4 /dev/sdb1
mke2fs 1.44.1 (24-Mar-2018)
Creating filesystem with 2621184 4k blocks and 655360 inodes
Filesystem UUID: e5e029af-7253-40d1-b8b6-3091eb7c27f4
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632

Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done

root@ubuntu1804:~# mkdir /mnt/sdb1
root@ubuntu1804:~# mount /dev/sdb1 /mnt/sdb1
root@ubuntu1804:~# ls /mnt/sdb1
lost+found
root@ubuntu1804:~#

```

Рис. 47.4. Диск подмонтирован

Осталось только обеспечить автоматическое монтируние этого диска при загрузке сервера. Для этого введите команду:

```
echo '/dev/sdb1 /mnt/sdb1 ext4 defaults 0 0' >> /etc/fstab
```

Эта команда добавит нужную строку в файл */etc/fstab*.

На этом все: мы создали раздел на диске, создали файловую систему (отформатировали — в терминологии Windows) и подмонтировали созданный раздел к каталогу корневой файловой системы.

## 47.2. Расширение существующего диска

Как и с добавлением нового диска, в панели управления сервером достаточно просто выполнить эту операцию, однако операционная система не увидит изменения, пока вы ей явно на них не укажете. На этот раз мы воспользуемся утилитой *parted*, обладающей необходимым функционалом, и продемонстрируем процесс расширения группы томов LVM в Ubuntu 18.04. Вы редко найдете сервер без LVM, разве что сами установите Linux (без LVM) в виртуальную машину, а потом переместите ее в облако.

Первым делом посмотрим, сколько сейчас дискового пространства доступно:

```
df -h
```

Как видите, общий размер группы томов **/dev/mapper/vgroup1-root** составляет 19 Гбайт (рис. 47.5). Наша задача — расширить размер этой группы томов до полного размера диска.

Пересканируем текущую аппаратную конфигурацию, чтобы система увидела новый объем винчестера:

```
echo 1 > /sys/block/sda/device/rescan
```

```

Last login: Fri Jun 29 07:56:02 2018 from 5.101.73.10
root@ubuntu1804:~# df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            1.5G   0    1.5G  0% /dev
tmpfs           301M  4.5M  296M  2% /run
/dev/mapper/vgroupl-root  19G  2.1G  17G  12% /
tmpfs           1.5G   0    1.5G  0% /dev/shm
tmpfs           5.0M   0    5.0M  0% /run/lock
tmpfs           1.5G   0    1.5G  0% /sys/fs/cgroup
/dev/sda1        922M  140M  719M  17% /boot
tmpfs           301M   0    301M  0% /run/user/0
root@ubuntu1804:~#

```

Рис. 47.5. Доступно 19 гигабайт

Запустите утилиту `parted` (используется для работы с разделами диска):

`parted`

Введите команду `r` для просмотра имеющихся разделов (рис. 47.6). Запомните номер раздела, который мы будем расширять (**2**), и новый размер диска (**42.9GB**).

Запустим команду изменения раздела:

`resizepart`

Укажем номер раздела:

`Partition number? 2`

А затем — конец раздела. Нужно указать как раз то самое значение `42.9GB` — именно так, без пробелов (рис. 47.7).

Введите команду `quit` для выхода из `parted`.

Сообщим ядру об изменениях размера:

```

pvresize /dev/sda2
Physical volume "/dev/sda2" changed
 1 physical volume(s) resized / 0 physical volume(s) not resized

```

Изменим логический том:

`lvextend -r -l +100%FREE /dev/mapper/vgroupl-root`

The screenshot shows a terminal window titled "Bitvise xterm - root@ubuntu1804: ~". The terminal displays the output of the "parted" command. It shows a disk with two partitions. The first partition is primary, type ext4, size 999MB, and has the "boot" flag set. The second partition is primary, type lvm, size 20.5GB. Two red arrows point to the "boot" flag of the first partition and the "lvm" flag of the second partition.

```
Last login: Fri Jun 29 06:08:28 2018 from 188.163.24.170
root@ubuntu1804:~# echo 1 > /sys/block/sda/device/rescan
root@ubuntu1804:~# parted
GNU Parted 3.2
Using /dev/sda
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted) p
Model: VMware Virtual disk (scsi)
Disk /dev/sda: 42.9GB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Partition Flags:

Number Start   End     Size    Type      File system  Flags
 1      1049kB  1000MB  999MB   primary   ext4        boot
 2      1000MB  21.5GB  20.5GB  primary   lvm

(parted)
```

Рис. 47.6. Текущая таблица разделов

The screenshot shows a terminal window titled "Bitvise xterm - root@ubuntu1804: ~". The terminal displays the output of the "parted" command. It shows the "resizepart" command being run on partition 2, which is currently 20.5GB. The user is prompted for the new end point, which is set to 42.9GB. The "quit" command is then entered, and a message indicates that the user may need to update /etc/fstab. The terminal then ends with a prompt for "root@ubuntu1804:~#".

```
Last login: Fri Jun 29 08:09:14 2018 from 188.163.24.170
root@ubuntu1804:~# echo 1 > /sys/block/sda/device/rescan
root@ubuntu1804:~# parted
GNU Parted 3.2
Using /dev/sda
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted) p
Model: VMware Virtual disk (scsi)
Disk /dev/sda: 42.9GB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Partition Flags:

Number Start   End     Size    Type      File system  Flags
 1      1049kB  1000MB  999MB   primary   ext4        boot
 2      1000MB  21.5GB  20.5GB  primary   lvm

(parted) resizepart
Partition number? 2
End? [21.5GB]? 42.9GB
(parted) quit
Information: You may need to update /etc/fstab.

root@ubuntu1804:~#
```

Рис. 47.7. Изменение размера раздела

По окончании работ введем команду `df -h`, чтобы убедиться, что дисковое пространство расширилось.

Посмотрите на рис. 47.8. Здесь показан результат выполнения команд `pvresize`, `lvextend` и `df -h`. Помеченная строка вывода сообщает нам, что размер группы томов `vgroup1-root` теперь составляет 41 Гбайт. Поставленная задача выполнена.

The screenshot shows a terminal session on an Ubuntu 18.04 system. The user performs the following steps:

- Enters `(parted) resizepart` and specifies partition number 2.
- Enters `End? [21.5GB]? 42.9GB`.
- Enters `(parted) quit`.
- Enters `Information: You may need to update /etc/fstab.`
- Enters `root@ubuntu1804:~# pvresize /dev/sda2`.
- Shows output indicating 1 physical volume resized.
- Enters `root@ubuntu1804:~# lvextend -r -l +100%FREE /dev/mapper/vgroup1-root`.
- Shows output indicating the logical volume `vgroup1/root` was resized from 18.11 GiB to 38.07 GiB.
- Shows detailed metadata for the logical volume `vgroup1/root`.
- Enters `root@ubuntu1804:~# df -H` to view disk usage.
- The output of `df -H` is shown, with the line for `/dev/mapper/vgroup1-root` highlighted in yellow, indicating it has been resized to 41G.

Рис. 47.8. Процедура расширения тома

# ПРИЛОЖЕНИЕ

## Описание файлового архива

Файловый архив, сопровождающий книгу, можно скачать с FTP-сервера издательства по ссылке <ftp://ftp.bhv.ru/9785977567732.zip>, а также со страницы книги на сайте <https://bhv.ru/>.

Файловый архив включает две папки:

в папке Дополнения в PDF-файлах с соответствующими названиями приведены следующие материалы:

- Автоматизация задач с помощью `tcsch` (введение в язык TC Shell);
- Автоматизация обработки текста средствами `gawk` (введение в язык `gawk`);
- 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;
- Лазерные диски и программы для их прожига — глава 16 из 6-го издания, в которой описываются программы для прожига CD/DVD-дисков;
- Настройка ADSL-доступа к Интернету — материал главы 9 из 5-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);

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:

- `40.3.gif` — работа системы хранения данных Серф в штатном режиме (анимация);
- `40.4.gif` — работа системы хранения данных Серф в случае сбоя (анимация);
- 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 20.10 — простое видео установки Ubuntu на новый компьютер.

Для воспроизведения видеоуроков необходимо установить кодек XViD.

Звукового сопровождения видеоуроков не предусмотрено.



# Предметный указатель

## 3

3DES 473

## A

ACL 544

Apache 479

APM 342

## B

bash 128

BIOS 365

Blowfish 147, 473

BogoMIPS 334

## C

CentOS 455

Ceph 578, 580

Clonezilla 583, 584

cyrus-pop3d 552

## D

DEB-пакеты 162

Destination NAT 462

DHCP 54

DirectX 323

DNS 528

DNS-сервер

◊ вторичный 539

◊ первичный 534

DoS 440

DOS 85

DSL-маршрутизаторы с функциями Wi-Fi

220

## E

eCryptfs 593

EDD 58

EFI 122

Ext2Fsd 80

## F

Firefox 301

firewall 460

Flash 302

FreeOpenVPN 229

FTP 517

FTP-клиент 517

FTP-клиент FileZilla 307

FTP-сервер 517

◊ ProFTPD 517

◊ wu-ftpd 517

## G

GIMP 283

GlusterFS 578

Google-сервис PageSpeed 514

GPT 122

GrSecurity 436

GRUB 335

GRUB2 362

## H

Hetzner 604

## I

IDEA 473

IMAP 551

initrd 366

Intel VT-x 645

iptables 463, 547

IPv4 forwarding 459, 463

i-узел 90

**K**

KDM 435  
 KVM 601  
 ◇ преимущества 603

PID 383  
 POP 551  
 POST 34  
 proc 389  
 PStorage 579  
 PXE 53

**L**

LIDS 436  
 Linux Live 583, 584, 591, 592  
 LiveCD 582–584, 591, 592  
 logrotate 410  
 Lustre 578  
 LXDE 50

**R**

RAID 568, 576  
 ◇ недостатки 577  
 RAID-массив 568, 569, 572–574, 576–578  
 RDIMM 600  
 Red Hat Ceph Storage 581  
 Remastersys Backup 583  
 Remote Administrator 138  
 rpm 166  
 RPM-пакеты 162, 164  
 RSA 473  
 rsyslogd 423

**S**

MAC 54  
 Macromedia Flash Player 302  
 MBR 89, 123, 365  
 MD5 147  
 MooseFS 578  
 MTA 551  
 MTA Exim 552, 553  
 MTA QMail 552  
 MTU 216

SecurityKISS: настройка 230  
 SELinux 334, 436  
 sendmail 412, 551  
 SGID 101  
 Slackware 62  
 SMP 333  
 SMTP 551  
 SMTP-сервер 554  
 SOA 536  
 Source NAT 463  
 speedtest.net  
 ◇ консольная версия 611  
 squidGuard 548  
 SSID 225  
 SSL-сертификат 504–506  
 SUID 101  
 Synaptic 181  
 sysfs 389  
 syslog-ng 423  
 SysRq 76

**N**

NAS 576  
 NAT 460, 655  
 NCSA HTTPd 1.3 479  
 Norton Ghost 582  
 ntfs-3g 108  
 NX-защита 332

**O**

OpenLDAP 412  
 OpenVPN 660, 662  
 OpenVZ 599, 601, 637–641, 644, 659, 668  
 ◇ недостатки 602  
 OS EZ Template 649  
 OSD 581  
 OSI 456

**T**

P2P 308  
 PAM 437  
 Parallels Cloud Storage 578  
 Photoshop 283

Telnet 473  
 TFTP 54  
 TLD 528  
 TLS/SSL 660  
 TrueCrypt 593

**P**

**У**

Ubuntu: DNS 534  
 Unity 50  
 UNIX 293  
 UUID 111

**В**

VDS 601  
 virtual environments 637  
 Virtual File System 389  
 VirtualBox 637, 639, 641  
 Virtuozzo 601, 644  
 ◇ установка 647

Virtuozzo Storage Cluster 646  
 VLAN 206  
 VMware 454, 637, 639, 641  
 VPN 659  
 VPN-доступ 228  
 VPN-сервер 659  
 VPS 601

**W**

Web-сервер Apache 550  
 Wi-Fi 220  
 Windows-версия GIMP 291

**А**

«Аварийные» комбинации клавиш 76  
 Автодополнение 77  
 Автоматизация выполнения задач 448  
 Алгоритм шифрования AES 593  
 Архитектура файловой системы 88  
 Атака на отказ 440  
 Аутентификация SMTP-AUTH 554

Восстановление загрузчика GRUB/GRUB2 358  
 Вращение изображения 287  
 Вторичный загрузчик 365  
 Выбор  
 ◇ графической среды 50  
 ◇ дистрибутива 21  
 ◇ пакетов для установки 49  
 ◇ языка установки 37

**Б**

Блок 90  
 Борьба с простыми паролями 443  
 Брандмауэр 460  
 ◇ iptables 463, 547  
 ◇ nftables 463  
 Браузер  
 ◇ Chromium 302  
 ◇ Google Chrome 302

**В**

Виртуализация на уровне ядра 637  
 Виртуальная машина 310  
 ◇ VirtualBox 311, 572  
 ◇ VMware 310, 572  
 Виртуальное окружение 637  
 Виртуальные  
 ◇ серверы 602  
 ◇ файловые системы 389  
 Виртуальный контейнер 637  
 Включение IPv4-переадресации 460

**Г**

Главная загрузочная запись 89, 365  
 Гостевая операционная система 310  
 Графическая  
 ◇ подсистема X.Org 445  
 ◇ среда 61  
 Графический  
 ◇ интерфейс 261
 

- GNOME 445
- KDE 445
- X Window 25

 ◇ менеджер регистрации 61  
 ◇ редактор GIMP 283  
 ◇ режим 62

**Д**

Двухканальный режим работы памяти 448  
 Демон  
 ◇ SASL 555  
 ◇ saslauthd 554

**Демоны протоколирования** 417

**Дефрагментация** 40

**Директива**

- ◊ **DefaultRoot** 522
- ◊ **Directory** 487
- ◊ **Files** 489
- ◊ **Limit** 488
- ◊ **MaxClients** 522
- ◊ **ServerName** 486

**Директивы файла конфигурации**  
profptd.conf 520

**Диск**

- ◊ **USB-диск** 114
  - ◊ **виртуальный** 366
  - ◊ **гибкий** 105
- Дистрибутив**
- ◊ **ALT Linux** 28
  - ◊ **CentOS** 28
  - ◊ **Debian Sarge** 29
  - ◊ **Fedora** 27
  - ◊ **Mageia** 27
  - ◊ **Mandrake** 27
  - ◊ **Mandriva** 27, 44
  - ◊ **openELEC** 249
  - ◊ **openSUSE** 30
  - ◊ **Red Hat** 27
  - ◊ **Slackware** 30
  - ◊ **Ubuntu** 29

**Дистрибутивы Linux** 25

**Длинные имена дисков** 111

**Домашний каталог** 94

**Домен** 534

**Дополнительные файлы конфигурации**  
PAM 438

## Ж

**Журналы** 80

## З

**Загрузчик**

- ◊ **ASPLoader** 350
- ◊ **GRUB** 34, 350
- ◊ **GRUB2** 34, 335, 350
- ◊ **LILO** 34, 350, 365

**Защита**

- ◊ **загрузчика паролем** 360
- ◊ **от «восстановления пароля root»** 433
- ◊ **от перезагрузки** 432, 433

**Зона** 534

## И

**Изменение**

- ◊ **размера изображения** 285
- ◊ **размера существующих разделов** 45
- ◊ **таблицы маршрутизации** 457
- ◊ **таблицы разделов** 39

**Имена разделов диска в GRUB** 352

**Информационные proc-файлы** 390

## К

**Кадрирование изображения** 288

**Каталог**

- ◊ **/etc/cron.daily** 449
- ◊ **/etc/cron.hourly** 449
- ◊ **/etc/cron.weekly** 449
- ◊ **/etc/rc.d** 369
- ◊ **/etc/rc.d/init.d** 369
- ◊ **/etc/skel** 148
- ◊ **/etc/zypp/repos.d** 187
- ◊ **/var/cache/apt/archives** 177
- ◊ **домашний** 93
- ◊ **признак каталога** 99
- ◊ **родительский** 93
- ◊ **текущий** 93

**Квотирование** 159

**Классический интерфейс GNOME** 64

**Клиент обмена мгновенными сообщениями**

Pidgin 304

**Кодек** 243

**Концепция модулей** 372

**Команда**

- ◊ **/sbin/grub-install** 356
- ◊ **/sbin/init** 369
- ◊ **adduser** 145
- ◊ **alien** 178
- ◊ **apt** 165
- ◊ **apt-get** 177
- ◊ **arch** 394
- ◊ **at** 451
- ◊ **atq** 451
- ◊ **atrm** 451
- ◊ **cat** 91
- ◊ **cd** 93
- ◊ **chmod +x** 130
- ◊ **chmod 100**
- ◊ **chown** 101
- ◊ **clear** 78, 394
- ◊ **cmp** 398
- ◊ **configure** 163

- ◊ convert 357
- ◊ cp 91
- ◊ date 394
- ◊ df 402
- ◊ diff 397
- ◊ dmesg 331
- ◊ dpkg 165, 175
- ◊ echo 395
- ◊ edquota 160
- ◊ egrep 398, 399
- ◊ exit 141, 395
- ◊ fdisk 119
- ◊ find 103
- ◊ free 402, 445
- ◊ fsck 109
- ◊ ftp 400
- ◊ grep 398
- ◊ groupadd 148
- ◊ grub 360
- ◊ grub-mkconfig 356
- ◊ grub-mkpasswd-pbkdf2 363
- ◊ gzip 357
- ◊ halt 63
- ◊ head 399
- ◊ ifconfig 78, 208
- ◊ kdesu 141
- ◊ kill 383
- ◊ killall 384
- ◊ less 91, 399
- ◊ ln 96
- ◊ locate 91, 104
- ◊ logout 63, 78
- ◊ ls 93
- ◊ lynx 401
- ◊ mail 402
- ◊ make 163
- ◊ md5sum 402
- ◊ mii-tool 218
- ◊ mkdir 93
- ◊ mkraid 572
- ◊ more 399
- ◊ mv 91
- ◊ netstat 453
- ◊ nice 387
- ◊ nslookup 534
- ◊ passwd 145, 395
- ◊ ping 211
- ◊ poweroff 63
- ◊ prlctl 650
- ◊ ps 383
- ◊ quotacheck 160
- ◊ quotaon 161
- ◊ raidhotadd 572
- ◊ raidhotremove 572
- ◊ reboot 63
- ◊ repquota 161
- ◊ rm 91, 93
- ◊ rmdir 93
- ◊ rndc-confgen 535
- ◊ route 453, 456
- ◊ rpm 165
- ◊ service 369
- ◊ shutdown 63
- ◊ ssh 403, 474
- ◊ startx 62, 395
- ◊ su 141
- ◊ sudo 140
- ◊ swapon 447
- ◊ tac 91
- ◊ tail 399
- ◊ telnet 403
- ◊ touch 91
- ◊ tracepath 211
- ◊ traceroute 211
- ◊ umount 105
- ◊ update-grub 356
- ◊ uptime 396
- ◊ userdel 147
- ◊ usermod 146
- ◊ users 396
- ◊ vzpkg 650
- ◊ wc 400
- ◊ which 91, 104
- ◊ who 139
- ◊ who 396
- ◊ xf86config 397
- ◊ yum 165, 168
- ◊ Компания TransGaming 324
- ◊ Коннектор сетевого кабеля 198
- ◊ Консоль 62
- ◊ Конфигуратор
- ◊ bum 370
- ◊ gproftpd 517
- ◊ pppoeconf 210
- ◊ system-config-services 370
- ◊ YaST 370
- ◊ Конфигурационный файл
- ◊ GRUB 351
- ◊ GRUB2 353
- ◊ X.Org 262

Корневая файловая система 86  
 Корневой раздел 34  
 Коэффициент подкачки 446  
 Криптографическая файловая система eCryptfs 593  
 Кросс-платформенная совместимость офисных пакетов 281  
 Кэширующий сервер DNS 529, 530

**Л**

Линус Торвальдс 24

**М**

Маршрутизатор 460  
 Маршрутизация 452  
 ◇ пакетов 452  
 Массивы 132  
 Масштабирование изображения 285  
 Менеджер пакетов 188, 190  
 Метод  
 ◇ SASL 554  
 ◇ отключения учетной записи root 433

**Н**

Настройка  
 ◇ X.Org в современных дистрибутивах 261  
 ◇ анонимного FTP-сервера 523  
 ◇ межсетевого экрана 460  
 ◇ неанонимного FTP-сервера 522  
 ◇ планировщика ввода/вывода 447  
 ◇ ядра 341

**О**

Обновление базы данных корневых серверов 539  
 Оболочка Unity 65  
 Образ жесткого диска 313  
 Объединение интернет-каналов 233  
 Оператор  
 ◇ case 135  
 ◇ if 134  
 Операционная система  
 ◇ AIX 82  
 ◇ QNX 348  
 ◇ UNIX 24  
 ◇ UNIX-подобная 24  
 Опции DNS-сервера 531  
 Основные особенности systemd 374

Отключение учетной записи root 435  
 Отключить блокировку экрана 71  
 Офисный пакет  
 ◇ Calligra Suite 279  
 ◇ Kingsoft Office 280  
 ◇ LibreOffice 277  
 ◇ OpenOffice.org 282  
 Очистка дерева исходного кода 343, 348

**П**

Пакет 162  
 ◇ bind 530  
 ◇ mysql-client 495  
 ◇ mysql-server 495  
 ◇ raidtools 572  
 ◇ resolvconf 230  
 ◇ samba 556  
 ◇ samba-server 556  
 ◇ зависимости 163  
 ◇ конфликты 164  
 Параллельный запуск сервисов 371  
 Параметры  
 ◇ виртуальной машины 316  
 ◇ фильтрации пакетов 465  
 ◇ ядра 334  
 Пароль на вход в BIOS Setup 431  
 Первичный загрузчик 365  
 Первый дистрибутив Linux 25  
 Переключение языков ввода 74  
 Перекомпиляция ядра 339  
 Переменные 131  
 ◇ окружения 131  
 ◇ специальные 132  
 Перенаправление ввода/вывода 78  
 Пиринговый протокол BitTorrent 308  
 Планировщик

◇ anacron 450  
 ◇ atd 451  
 ◇ crond 448  
 Повышение отказоустойчивости интернет-соединения 233  
 Подключение Linux к сети Microsoft 556  
 Пользователь root 51  
 Порядок установки операционных систем 53  
 Почтовый  
 ◇ клиент 551  
     ◦ Evolution 303  
     ◦ KMail 303  
     ◦ Mozilla Thunderbird 303  
 ◇ сервер 551

**Правила**

- ◊ брандмауэра 469
- ◊ маршрутизации 452
- Пример файла конфигурации X.Org 264
- Проблемы при установке Linux 56
- Проверка установочного DVD 39
- Программа
  - ◊ /usr/sbin/grub-mkconfig 353
  - ◊ ftpcount 526
  - ◊ ftpwho 526
  - ◊ GParted 122
  - ◊ installpkg 184
  - ◊ pkgtool 183
  - ◊ removepkg 184
  - ◊ rndc 533
  - ◊ rpm2tgz 185
  - ◊ slackpkg 186
  - ◊ smbclient 559
  - ◊ Transmission 308
  - ◊ upgradepkg 184
  - ◊ urpmi 167
  - ◊ xpkgtool 184
  - ◊ zypper 189
- Программные RAID-массивы 571
- Программы (агенты) передачи почты (MTA) 551
- Программы-«заглушки» 129
- Прозрачный прокси-сервер 547
- Проигрыватель
  - ◊ Videos 244
  - ◊ VLC 244
- Производительность файловых систем 83
- Прокси-сервер 542
  - ◊ Squid 542
  - ◊ прозрачный 546
- Протокол
  - ◊ Kerberos 561
  - ◊ TLS 554
  - ◊ отправки почты (SMTP) 551
- Протоколирование POST-запросов 516
- Протоколы получения почты (POP, IMAP) 551
- Прототипы 161
- Псевдонимы команд 77, 78
- Псевдофайловая система
  - ◊ proc 390
  - ◊ sysfs 389
- Псевдофайловые системы 389

**P**

- Работа с консолью 91
- Раздел подкачки 445, 446
- Разметка диска 40, 44, 45
- Размытие изображения 288
- Распространение Linux 25
- Расширение имени файла 85
- Редактирование
  - ◊ конфигурации GRUB2 356
  - ◊ параметров ядра 35, 337
- Редактор
  - ◊ joe 296
  - ◊ mcedit 296
  - ◊ nano 296
  - ◊ pico 296
  - ◊ vi 293
- Резервное копирование 129
- Репозиторий 164

**C**

- Сервер 599
  - ◊ FTP 517
  - ◊ POP 551
  - ◊ Samba 517
  - ◊ SMTP 551
  - ◊ аутентификации SASL 554
  - ◊ физический 600
- Серверный дистрибутив 31
- Сервис
  - ◊ Samba 556
  - ◊ systemd-journald.service 417
  - ◊ интернет-телефонии Skype 304
- Сертификат Let's Encrypt 509, 510, 512, 514
- Сетевой протокол аутентификации Kerberos 556
- Сетевые интерфейсы 456
- Сеть: отказ работы 208
- Система
  - ◊ кэширования данных Memcached 514–516
  - ◊ доменных имен 528
  - ◊ инициализации
    - ◊ init 366
    - ◊ Slackware 381
    - ◊ systemd 367, 417, 422
    - ◊ upstart 366
  - ◊ хранения данных
    - ◊ Ceph 580, 581, 675
    - ◊ распределенная 580, 675

- Системные требования Linux 32
- Системы управления доступом 436
- Создание
  - ◊ виртуальной машины 312
  - ◊ учетных записей пользователей 52
- Специальный демон WinBind 556
- Список управления доступом (ACL) 544
- Способы
  - ◊ аутентификации 437
  - ◊ виртуализации 637
- Сравнение дистрибутивов 26
- Статическая маршрутизация 234
- Суперблок 90
- Схема разрешения доменного имени 529
- Сценарии (скрипты) GIMP 291
- Сценарий 129
  - ◊ .bashrc 129

## Т

- Таблица
  - ◊ маршрутизации 453, 455, 456
  - ◊ разделов 41
- Текстовый клиент ftp 306
- Терминал 78
- Терминалы 64, 78
- Терминальный мультиплексор 128
- Технология виртуализации
  - ◊ KVM 637
  - ◊ OpenVZ 637
- Точка
  - ◊ доступа Wi-Fi на смартфоне 226
  - ◊ монтирования 42, 110

## У

- Увеличение скорости доступа к Интернету 233
- Уровни RAID-массивов 568
- Установка
  - ◊ Linux 32, 53, 57
    - по сети 53
  - ◊ кодеков в openSUSE 244
  - ◊ пароля
    - root 51
    - загрузчика 431
  - ◊ программ из исходных кодов 162
  - ◊ проприетарных драйверов NVIDIA 270
  - ◊ разрешения монитора 261
- Утилита journalctl 417

## Ф

- Файл
  - ◊ .{ICE,X}authority 142
  - ◊ .bash\_history 129
  - ◊ .bash\_profile 78
  - ◊ /boot/boot.b 365
  - ◊ /boot/grub/grub.conf 351
  - ◊ /boot/map 366
  - ◊ /etc/anacrontab 450
  - ◊ /etc/apt/sources.list 177
  - ◊ /etc/audit/auditd.conf 406
  - ◊ /etc/crontab 448
  - ◊ /etc/cups/printers.conf 407
  - ◊ /etc/default/grub 354
  - ◊ /etc/default/ufw 663
  - ◊ /etc/dhcp3/dhcpd/dhcpd.conf 54
  - ◊ /etc/fstab 109, 159
  - ◊ /etc/group 148
  - ◊ /etc/hostname 216
  - ◊ /etc/HOSTNAME 216
  - ◊ /etc/httpd/conf 492
  - ◊ /etc/inetd.conf 54
  - ◊ /etc/inittab 367
  - ◊ /etc/network/interfaces 216, 456
  - ◊ /etc/NetworkManager/system-connections 223
  - ◊ /etc/openvpn/server.conf 662
  - ◊ /etc/passwd 147
  - ◊ /etc/proftpd/proftpd.conf 518
  - ◊ /etc/resolv.conf 533
  - ◊ /etc/resolvconf/resolv.conf.d/base 230
  - ◊ /etc/route.conf 455
  - ◊ /etc/samba/smb.conf 556
  - ◊ /etc/shadow 147
  - ◊ /etc/shells 128, 129
  - ◊ /etc/sshd\_config 474
  - ◊ /etc/sudoers 141
  - ◊ /etc/sysconfig/network 213
  - ◊ /etc/sysconfig/network/config 215
  - ◊ /etc/sysconfig/network/dhcp 216
  - ◊ /etc/sysconfig/network/ifcfg-eth0 215
  - ◊ /etc/sysconfig/network/routes 215, 455
  - ◊ /etc/sysconfig/network-scripts/ifcfg-eth0 214
  - ◊ /etc/sysconfig/static-routes 215
  - ◊ /etc/ufw/before.rules 663
  - ◊ /etc/urpmi/urpmi.conf 167
  - ◊ /etc/yum.conf 171
  - ◊ /etc/zypp/zypp.conf 188

- ◊ /proc/filesystems 389
  - ◊ /proc/sys/vm/swappiness 446
  - ◊ /var/log/messages 209
  - ◊ apache.conf 485
  - ◊ apache2.conf 485
  - ◊ aquota.user 159
  - ◊ etc/squid/squid.conf 542
  - ◊ fstab 116
  - ◊ httpd.conf 485
  - ◊ httpd2.conf 485
  - ◊ resolv.conf 534
  - ◊ smb.conf 559
  - ◊ xorg.conf 264
  - ◊ подкачки 446
  - ◊ права доступа 99
  - ◊ устройства 86, 105
- Файловая система 88
- ◊ Btrfs 82
  - ◊ eCryptfs 593
  - ◊ ext2 80, 573
  - ◊ ext3 80
  - ◊ ext4 80, 117
  - ◊ JFS 82
  - ◊ NTFS 84
  - ◊ Reiser4 81
  - ◊ ReiserFS 81, 573
  - ◊ Tux2 82
  - ◊ Tux3 82
  - ◊ XFS 81
  - ◊ Xiafs 82
  - ◊ ZFS 82
  - ◊ журналируемая 80, 116
- Файловые системы 81
- Файловый менеджер
- ◊ Dolphin 559
- Файлы
- ◊ конфигурации РАМ 437
  - ◊ устройств 86
- Фильтрация пакетов 461
- Флеш-память 107
- Фон рабочего стола 75
- Форвард-сервер 532

## X

- Хост-клавиша виртуальной машины 322
- Хост-машина 639

## Ц

- Цепочка правил брандмауэра 461
- Цикл
  - ◊ for 133
  - ◊ while 133

## Ч

- Черный список интернет-адресов 545

## Ш

- Шифрование файловой системы 48
- Шлюз 460, 467
  - ◊ по умолчанию 452, 455

## Э

- Эмулятор 323
  - ◊ Cedega 324
  - ◊ VirtualBox 323
  - ◊ VMware 323
  - ◊ Wine 323
  - ◊ Winex 323
  - ◊ виртуальной машины 311

## Я

- Ядро 34, 331
  - ◊ модуль ядра 343
  - ◊ параметры ядра 35
  - ◊ системный вызов 35



## Внутреннее устройство Linux, 2-е изд.

Отдел оптовых поставок:  
e-mail: opt@bhb.ru



- Пользовательское окружение и интерфейс командной строки CLI
- Файлы, каталоги и файловые системы
- Дискреционное, мандатное разграничение доступа и привилегии
- Процессы и нити
- Виртуальная память и отображаемые файлы
- Каналы, сокеты и разделяемая память
- Сетевая подсистема и служба SSH
- Графический интерфейс GUI: оконные системы X Window и Wayland
- Программирование на языке командного интерпретатора
- Контейнеры и виртуализация
- Linux своими руками

Книга, которую вы держите в руках, адресована студентам, начинающим пользователям, программистам и системным администраторам операционной системы Linux. Она представляет собой введение во внутреннее устройство Linux — от ядра до сетевых служб и от утилит командной строки до графического интерфейса.

Все части операционной системы рассматриваются в контексте типичных задач, решаемых на практике, и поясняются при помощи соответствующего инструментария пользователя, администратора и разработчика.

Все положения наглядно проиллюстрированы примерами, разработанными и проверенными автором с целью привить читателю навыки самостоятельного исследования постоянно эволюционирующей операционной системы Linux.

**Кетов Дмитрий Владимирович**, инженер в Санкт-Петербургском исследовательском центре LG Russia R&D Lab. Профессионально занимается теорией построения и практикой разработки операционных систем и системного программного обеспечения. Имеет многолетний опыт преподавания в Санкт-Петербургском политехническом университете (СПбПУ) в области операционных систем и сетевых технологий.



www.bhv.ru

Волох С.

## Ubuntu Linux с нуля, 2-е изд.

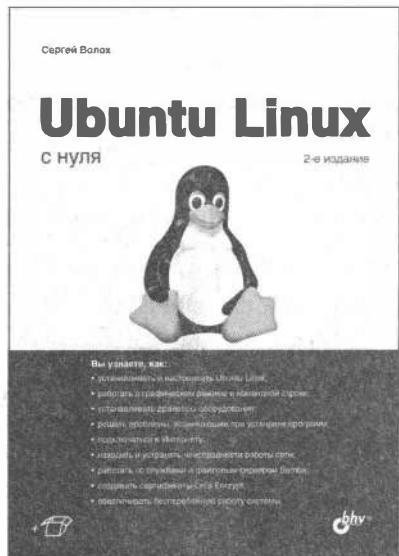
Отдел оптовых поставок:

e-mail: opt@bhw.ru

**Ubuntu Linux — это проще, чем кажется**

Вы узнаете, как:

- устанавливать и настраивать Ubuntu Linux;
- работать в графическом режиме и командной строке;
- устанавливать драйвера оборудования;
- решать проблемы, возникающие при установке программ;
- подключаться к Интернету;
- находить и устранять неисправности работы сети;
- работать со службами и файловым сервером Samba;
- создавать сертификаты Let's Encrypt;
- обеспечивать бесперебойную работу системы.



Эта книга — проводник в мир операционной системы Linux! В ней рассмотрен самый популярный дистрибутив Ubuntu Linux. С позиции пользователя описан весь цикл работы, начиная от установки и заканчивая восстановлением работы системы после сбоев. Рассмотрены вопросы подключения принтеров, сканеров и других периферийных устройств. Особое внимание уделено работе в командной строке, при этом все команды подробно раскрыты. Материал книги ориентирован на текущую версию, но будет применим и к будущим версиям Ubuntu Linux. Более того, полученных в книге знаний достаточно для перехода на любой другой дистрибутив Linux без дополнительного его изучения.

**Волох Сергей Васильевич**, программист, специалист в области информационных технологий, преподавал информатику в школе. Благодаря опыту работы в педагогической сфере умеет простым и понятным языком рассказывать о сложных вещах. Ведет личный сайт <https://volokh.info>, посвященный информационным технологиям, где активно помогает пользователям в решении возникших проблем.



# Linux

## от новичка к профессионалу



**Колисниченко Денис Николаевич**, инженер-программист, системный администратор и IT-консультант. 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 системного администратора», «Программирование для Android», «Секреты безопасности и анонимности в Интернете» и др.



Дополнительные главы в PDF-файлах и видеоуроки можно скачать по <ftp://ftp.bhv.ru/9785977567732.zip>, а также со страницы книги на сайте [www.bhv.ru](http://www.bhv.ru).

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 веб-сервера. Материал ориентирован на последние версии дистрибутивов Fedora, openSUSE, Slackware, Ubuntu.

В восьмом издании рассмотрены Fedora 33, модуль zRAM, файловая система Btrfs, настройка Apache для работы на нескольких портах, организация поддоменов \*.example.com, выбор и настройка VDS, брандмауэр ufw, лайфхаки для начинающих администраторов.

191036, Санкт-Петербург,  
Гончарная ул., 20  
Тел.: (812) 717-10-50,  
339-54-17, 339-54-28  
E-mail: [mail@bhv.ru](mailto:mail@bhv.ru)  
Internet: [www.bhv.ru](http://www.bhv.ru)

ISBN 978-5-9775-6773-2



9 785977 567732

В ПОДЛИННИКЕ®