Эта книга - 100%-ная гарантия вашей уверенной работы по работе и администрированию Linux

Колисниченко Д. Н.

100% гарантия эффективной работы

LINUX

Полное руководство по работе и администрированию

- Полное описание всех необходимых процедур по использованию и администрированию Linux : от первоначальной настройки до пересборки ядра операционной системы
- Включает рассмотрение рекомендованной в России защищенной Astra Linux

ПОЛНОЕ РУКОВОДСТВО

Колисниченко Д. Н.

LINUX

ПОЛНОЕ РУКОВОДСТВО по работе и администрированию

"Наука и Техника"
Санкт-Петербург

УДК 004.42
ББК 32.973

Колисниченко Д. Н.

LINUX. Полное руководство по работе и администрированию — СПб.: Наука и Техника, 2021. — 480 с., ил.

ISBN 978-5-94387-608-0

Linux в наше время весьма популярен как у обычных пользователей, так и у крупных корпораций – таких как Microsoft, IBM и т.д. Эта книга содержит в себе как теоретические, так и практические материалы, т.е. теория и практика будут объединены в одно целое - не будет отдельных больших и скучных теоретических глав.

Книгу можно разделить на четыре части. В первой части мы поговорим об установке системы, рассмотрим вход и завершение работы, выполним кое-какие действия по настройке системы, рассмотрим основы командной строки;

Вторая часть посвящена настройкам Интернета, установке программного обеспечения, и обзору популярных программ для Linux – вы узнаете не только, как устанавливать программы, но и какую программу установить, что не менее важно. В третьей части будет подробно рассмотрено локальное администрирование в Linux: управление файловыми системами; загрузка операционной системы; системные процессы и основные группы пользователей;

Ну и, конечно, какой Linux без сервера? Четвертая часть посвящена вопросам администрирования Linux-сервера в локальной сети – будет показано, как настроить серверы DNS, SSH, DHCP, FTP; поговорим об интеграции сервера в Windows-сеть; о безопасности сервера; а также настроим брандмауэр и научимся защищать сервер от сетевых атак.

Книга будет полезна для любого уровня читателей – как для тех, кто только заинтересовался Линуксом, так и для тех, кто хочет расширить свои навыки использования этой операционной системы. Каждый найдет здесь для себя что-то полезное и востребованное! Важно, что одним из дистрибутивов (наряду с Ubuntu), на котором показывается работа в Linux, выбран российский Astra Linux, сертифицированный и рекомендованный к использованию на территории России.

ISBN 978-5-94387-608-0

Контактные телефоны издательства:
(812) 412 70 26

Официальный сайт: www.nit.com.ru

Содержание

Введение

Linux - удивительная операционная система. Впервые она появилась в 1991 году, а пик ее популярности приходится на начало 2000-х. Затем интерес к ней пошел на спад и можно было подумать, что через пару лет о ней никто не вспомнит. Но Linux подобен Фениксу, восставшему из пепла. Интерес к ней появился не только у обычных пользователей (иначе бы я сейчас не писал эту книгу), но и крупных корпораций, таких как Microsoft, IBM и т.д.

Некоторые из книг бывают сугубо практическими, некоторые - сугубо теоретическими. Книга, которую вы держите в руках, будет эдаким симбиозом и содержать в себе, как теоретические, так и практические материалы. Теория и практика будут объединены в одно целое - не будет отдельных больших и скучных теоретических глав.

Книга состоит из четырех частей:

1. Начинаем работу с Linux – здесь мы поговорим об установке системы, рассмотрим вход и завершение работы, выполним кое-какие действия по настройке системы, рассмотрим основы командной строки.
2. Linux для пользователя – настройка Интернета, установка программного обеспечения, популярные программы – обо всем этом мы поговорим в этой части книги. Вы узнаете не только, как устанавливать программы, но и какую программу установить, что не менее важно.
3. Локальное администрирование – управление файловыми системами, загрузкой операционной системы, процессами, пользователями – все это рассматривается в третьей части книги.
4. Сервер для локальной сети – здесь мы затрагиваем вопросы администрирования Linux-сервера в локальной сети: будет показано, как настроить серверы DNS, SSH, DHCP, FTP, поговорим об интеграции сервера в Windows-сеть, о безопасности сервера, а также настроим брандмауэр и научимся защищать сервер от сетевых атак.

Данная книга - скорее не учебник, а полноценное руководство и вы можете начать читать ее с любой части или даже главы, которая наиболее интересует вас в данный момент.

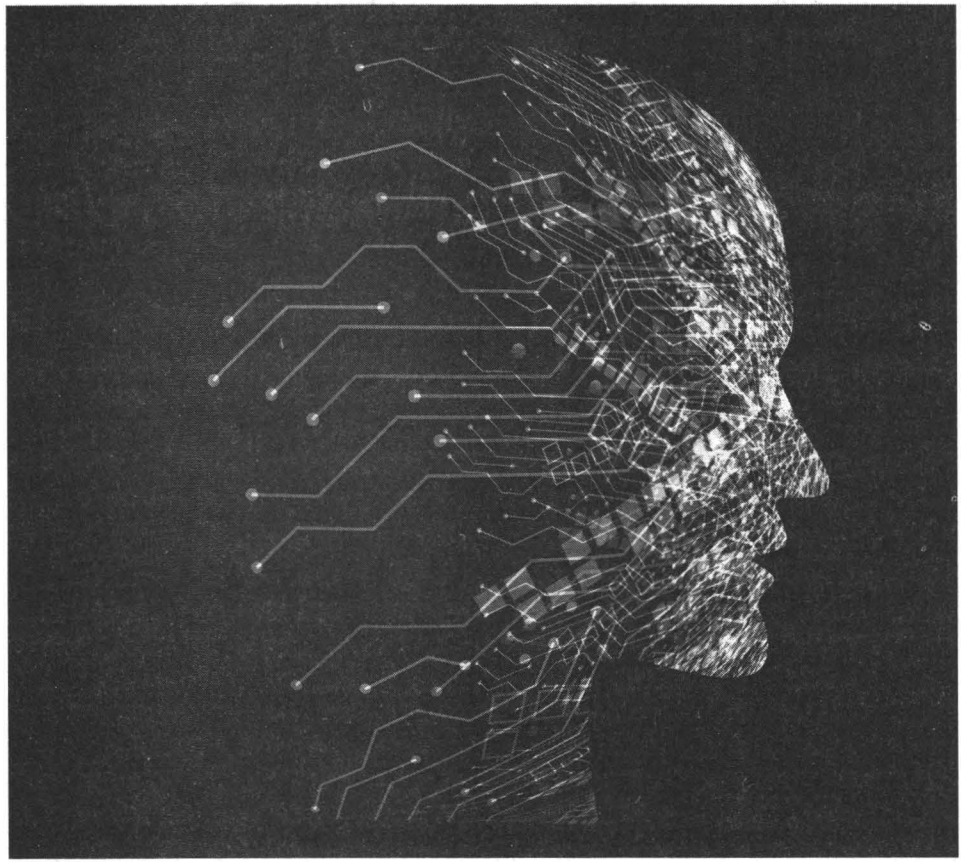
Часть I.

Начинаем работу с Linux

В этой части книги мы начинаем знакомиться и работать с Linux – здесь мы поговорим об установке системы, рассмотрим вход и завершение работы, выполним кое-какие действия по настройке системы, рассмотрим основы командной строки.

Глава 1.

Выбор дистрибутива



1.1. Многообразие выбора

Проблема современного мира IT – в его многообразии выбора. Для выполнения одной и той же задачи часто рынок предлагает множество самых разных решений. Решения различны по своему функционалу, лицензии, стоимости... И во всем этом нужно разобраться.

Порой выбор – не легкий. Ошибка на этапе выбора может стоить, как минимум потраченного времени, как максимум вы потратите время, деньги, а возможно и потеряете репутацию и работу. Например, начальство поручило вам выбрать программный продукт для взаимоотношения с клиентами (CRM). Вы сделали неправильный, как потом окажется в будущем, выбор. В результате компания потеряла огромные средства, потраченные на покупку и внедрение этого программного продукта, не говоря уже о неудобствах для клиентов и простоя при переходе на другую, «правильную» CRM.

Порой кажется, что лучше выбора бы не было. Одна задача – один инструмент. Одна операционная система, один офисный пакет и т.д. Но, к счастью или к сожалению, такая ситуация маловероятна даже в будущем.

Типичный пример – выбор операционной системы для веб-сервера. Что лучше? FreeBSD, Windows Server или Linux? Вы мучаетесь всю ночь и принимаете решение. Первая – слишком сложная, вторая – хороша, но за нее нужно платить, третья – самый универсальный вариант. И не очень сложно, и много документации, и всевозможное ПО. Самое главное – бесплатная. Собственно, какой выбор вы сделали понятно еще и по тому, какую книгу вы сейчас держите в руках.

После выбора в пользу Linux начинается самое интересное. Вы узнаете, что Linux – это только «собирательное название», а по факту выбирать приходится между дистрибутивами Linux, которых очень и очень много. С 2001 года, согласно ресурсу Distrowatch, было создано почти 800 дистрибутивов... Понятно, что количество разрабатываемых дистрибутивов постоянно сокращается – слабые уходят с рынка, но выбор по-прежнему огромен. Несколько сотен дистрибутивов. Только на главной странице Distrowatch есть список TOP-100 – он обновляется ежедневно. Следовательно, в мире есть как минимум 100 активных дистрибутивов, которыми пользуются люди...

1.2. Что такое дистрибутив

Самая первая версия Linux, появившаяся в 1991 году, представляла собой ядро и несколько приложений. В ней запускались компилятор gcc и командный интерпретатор bash. Поставлялась эта версия Linux в виде двух дискет – на первой было ядро, на второй – корневая файловая система с приложениями. Загружалась она тоже специфически – сначала нужно было вставить в дисковод первую дискету и дождаться, пока загрузится ядро, затем был запрос на вставку второй дискеты – с корневой файловой системой.

Первые дистрибутивы появились в 1992 году. Тогда отдельные энтузиасты или группы энтузиастов выпускали разные дистрибутивы (каждый, естественно, под своим именем). Грубо говоря, они отличались второй дискетой, на которой был немного другой набор программ. Далее, с развитием Linux, необходимые программы уже не помещались на одну дискету и пришлось устанавливать Linux на жесткий диск. Появились первые программы установки.

Чем отличаются разные дистрибутивы, кроме, разумеется, названия? Во-первых, у каждого дистрибутива своя программа установки (если не считать дистрибутивов-клонов, которые заимствуют инсталлятор у родительского дистрибутива, меняя только название дистрибутива). Во-вторых, у каждого будет свой набор программ – на усмотрение разработчика. По сути, с 1992 года ничего не поменялось.

Если копнуть дальше, вникнуть в набор программ, то начинаешь видеть более глубинные изменения, например, разницу в менеджере пакетов и системе инициализации. По сути, что менеджер пакетов, что система инициализации – это тоже программы. Но программа-программе – рознь.

Сейчас мы не будем вникать во всех технические тонкости. Для этого есть соответствующие главы это книги. Так, установка пакетов (программ) описывается в главе 10, а система инициализации – в главе 15. Лучше рассмотрим актуальные на данный момент дистрибутивы.

На данный момент Linux-пользователям доступно семь основных дистрибутивов:

- **Debian** – тот самый надежный Debian, появившийся в 1993 году. Это единственный широко распространенный дистрибутив, доживший до наших дней под оригинальным названием.
- **Fedora** – потомок популярного ранее дистрибутива Red Hat, существование которого было прекращено в 2004 году. Тогда пользователям

предоставили выбор: либо они мигрируют на корпоративный (коммерческий) RHEL (Red Hat Enterprise Linux), либо на бесплатный Fedora (ранее Fedora Core). На данный момент Fedora – развивающийся дистрибутив, последняя версия которого вышла 28 апреля 2020 года, а выпуск новых версий производится каждые 6-8 месяцев.

- **Ubuntu** – изначально основан на Debian, первая версия появилась в 2004 году, последняя – 23 апреля 2020 года (версия 20.04). Обновляется каждые 6 месяцев. Как и Fedora, имеет несколько вариантов, в том числе серверный. Популярным неофициальным (не от разработчиков Ubuntu) форком является дистрибутив Mint – «доведенная до ума» версия Ubuntu.
- **openSUSE** – изначально основан на дистрибутиве Slackware и первая его версия вышла в октябре 2005 года (сравнительно молодой дистрибутив). На данный момент доступна версия от 22 мая 2019 года, а обновляется дистрибутив примерно раз в год. В отличие от Ubuntu, использует систему пакетов RPM, что делает его ближе к Fedora – со временем в состав openSUSE включили некоторые решения из Red Hat – систему пакетов RPM, использование sysconfig – что сделало больше похожим на Red Hat, чем на Slackware.
- **ALT Linux** – как ни крути, но этот отечественный дистрибутив заслуживает уважения – хотя бы за то, что дожил до наших дней и не развалился, как многие другие. И учтите, первая его версия появилась в 1999 году (то есть ему больше 20 лет), а не в 2004-2005, как Ubuntu и openSUSE. Последняя версия от 28 октября 2019 года.
- **CentOS** (Community ENTerprise Operating System) – общественная корпоративная операционная система. Основан на RHEL и совместим с ним. Содержит из свободного ПО с открытым кодом. Первая версия вышла в 2004 году, на данный момент последней является версия от 14 января 2020 года – дистрибутив развивается. Дистрибутив очень надежный – иного от корпоративной ОС и не следует ожидать, пусть и не содержит самых новых пакетов ПО, как, например, Fedora.
- **Astra Linux** – дистрибутив специального назначения на базе ядра Linux, созданная для комплексной защиты информации и построения защищенных автоматизированных систем. Сертифицирована в системах сертификации средств защиты информации Минобороны, ФСТЭК и ФСБ России. Первая версия увидела свет в 2009 году, а последняя версия вышла 10 мая 2019 года.

1.3. Что выбрать?

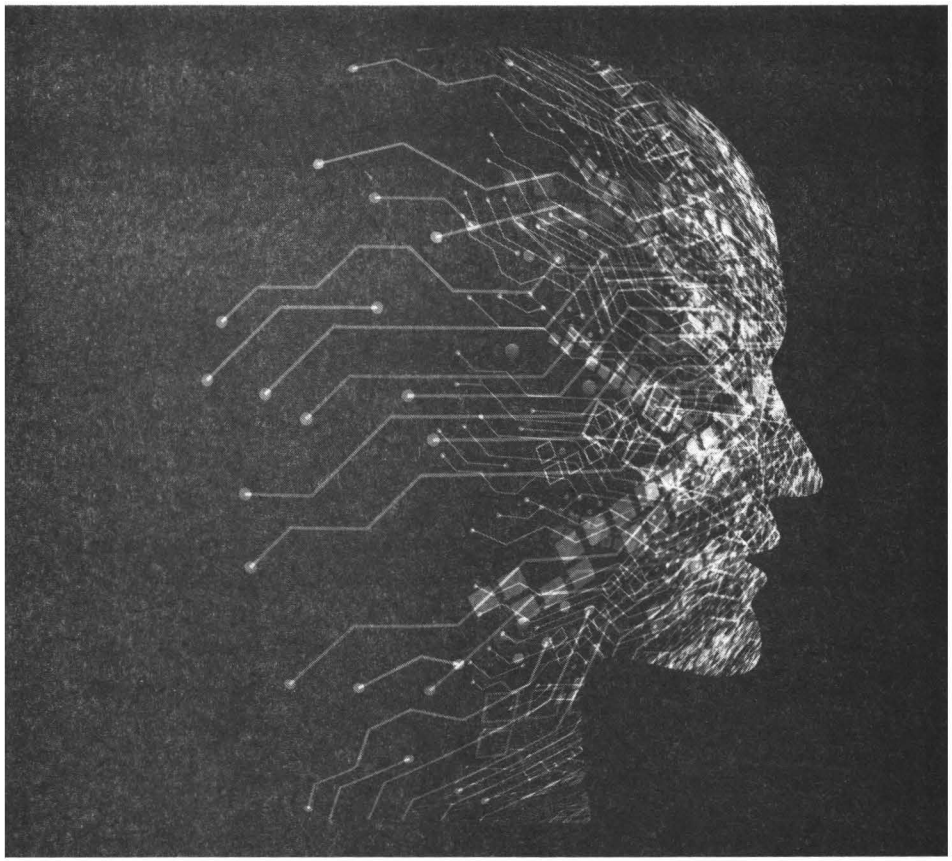
Вопрос довольно распространенный, но однозначного ответа на него нет. Все зависит от применения и личных предпочтений. Например, фанатов Ubuntu ни за что не заставишь установить Fedora и наоборот. Если же у вас своего мнения относительно дистрибутива не сформировалось, то можно выбирать один из следующих дистрибутивов – Fedora, CentOS, Ubuntu, Debian. На сервере я бы рекомендовал более стабильные CentOS и Debian, но поскольку вы только начинаете разбираться с Linux, можно смело использовать Fedora и Ubuntu. С ними вам будет проще и они более универсальные. Оба дистрибутива смело подойдут как для рабочей станции (или домашнего компьютера), так и для сервера.

Когда же речь заходит о работе с персональными данными, то ненароком вмешивается закон ФЗ-152 и расставляет все на свои места. Обработка и хранение таких данных должно происходить только с использованием сертифицированного ПО. Для Windows достаточный большой выбор различного сертифицированного ПО – и программы для шифрования, и антивирусы, и брандмауэры и т.д. Выходит, если вы создаете Интернет-магазин, что подразумевает хранение и обработку персональных данных, то вы или ограничены Windows (что не хочется), или же нужно смотреть в сторону сертифицированных дистрибутивов Linux. Один из таких – Astra Linux. Именно поэтому он рассматривается в этой книге вместе с Ubuntu. В книге, по понятным причинам, мы будем рассматривать Astra Linux общего назначения, поскольку дистрибутив «особого назначения» – платный. Забегая наперед, нужно отметить, что дистрибутив получился весьма неплохой, но имеет многочисленные проблемы с локализацией – некоторые элементы окон программ не переведены на русский язык. Непростительная оплошность для отечественного дистрибутива, где с локализации нужно было начинать.

В книге рассматриваются самые последние на момент написания этих строк дистрибутивы (для Astra Linux – релиз «Орел», для Ubuntu – 20.04). В мире Linux новые дистрибутивы выходят довольно часто, поэтому нет смысла рассматривать предыдущие версии.

Глава 2.

Установка системы



Несмотря на то, что у всех дистрибутивов Linux собственный инсталлятор, свой интерфейс пользователя, разный набор программного обеспечения, устанавливаемого по умолчанию, все они устанавливаются по единому принципу. В этой главе мы рассмотрим попарно установку Ubuntu и Astra Linux.

2.1. Загрузка с инсталляционного диска

Первое с чего нужно начинать установку системы – с получения инсталляционного носителя. Поскольку рассматриваемые дистрибутивы Linux распространяются абсолютно бесплатно, нет никакого смысла загружать ISO-образы со сторонних ресурсов. В нашем случае необходимые ISO-файлы можно получить с сайтов https://ubuntu.com и https://astralinux.ru/.

Далее нужно создать сам инсталляционный носитель. В качестве такового носителя может выступать либо DVD-диск, либо USB-флешка. Эра DVD-дисков давно прошла (сейчас даже в продаже их найти сложно), а вот USB-флешка, как правило, всегда под рукой.

Для подготовки загрузочного USB-диска нам нужна флешка с размером от 4 Гб и компьютер под управлением Windows. Скачайте приложение Rufus (https://rufus.ie/) и выполните следующие действия:

1. Если на флешке есть данные, скопируйте их на жесткий диск, поскольку в процессе создания загрузочного носителя они будут уничтожены.
2. Запустите Rufus.
3. Из списка **Устройство** выберите флешку. Убедитесь, что вы выбрали правильную флешку, если подключено несколько устройств.
4. Нажмите кнопку **Выбрать** для выбора ISO-образа, который будет записан на флешку.
5. Остальные параметры оставьте как есть. Схема раздела должна быть MBR, целевая система – BIOS или UEFI, файловая система – FAT32, размер кластера – 4096.

6. Нажмите кнопку СТАРТ.
7. Дождитесь, пока программа запишет ISO-образ на флешку.
8. Подключите USB-флешку к целевому компьютеру (на который будет происходить установка Linux)
9. Перезагрузите целевой компьютер.
10. Войдите в BIOS SETUP (обычно для этого используется клавиша DEL или F2, но нужная комбинация может отличаться – обратитесь к руководству по материнской плате).
11. В качестве загрузочного устройства выберите созданную флешку.
12. Выйдите из BIOS SETUP с сохранением изменений.

Если вы все сделали правильно, то увидите начальный экран загрузчика Linux.

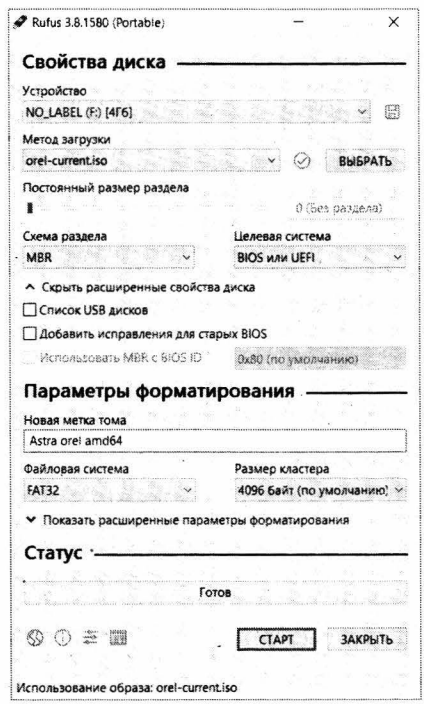


Рис. 2.1. Запись ISO образа с Linux на флешку

2.2. Принцип установки Linux

Самое главное при установке Linux - выполнить разметку жесткого диска и не забыть установленный пароль, указанный при установке. Неправильная разметка жесткого диска означает, что в процессе работы с системой ее придется изменить, а это сделать не всегда просто, особенно, если сервер уже работает. Ну и пароль пользователя желательно тоже не забывать, иначе придется попотеть, чтобы взломать собственную же систему. А возможность такого взлома зависит, прежде всего, от настроек дистрибутива по умолчанию - в одних дистрибутивах все будет хорошо, а в других у вас ничего не получится. Хотя вряд ли можно назвать хорошим возможность взлома локального сервера (к которому у вас есть физический доступ) - так что все относительно.

2.3. Загрузка с инсталляционного носителя

В случае с Ubuntu вы увидите, что экран окрашен в фиолетовый цвет и внизу будет небольшое изображение клавиатуры. Нажмите **Пробел** или любую другую клавишу для выбора языка (рис. 2.2). Если вы не успеете это сделать, то Ubuntu будет запущена в режиме LiveCD на английском языке – не волнуйтесь язык легко изменить при установке системы.

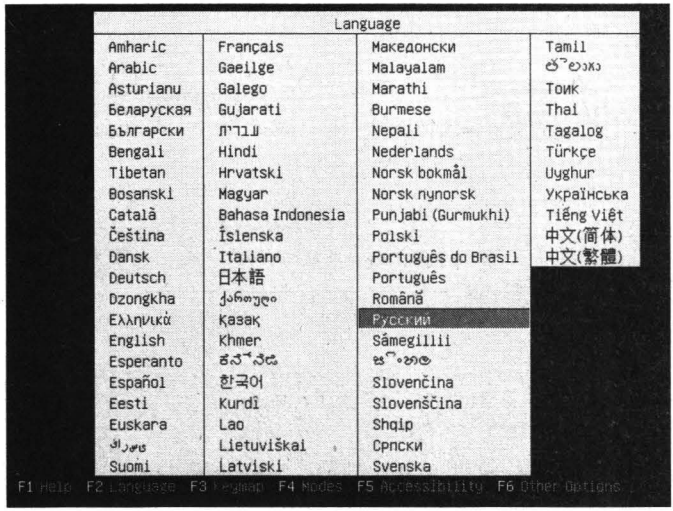


Рис. 2.2. Выбор языка при установке Ubuntu

Если вы успели нажать любую клавишу, вы увидите меню загрузчика (рис. 2.3). Назначение команд загрузчика понятно и в особых комментариях не

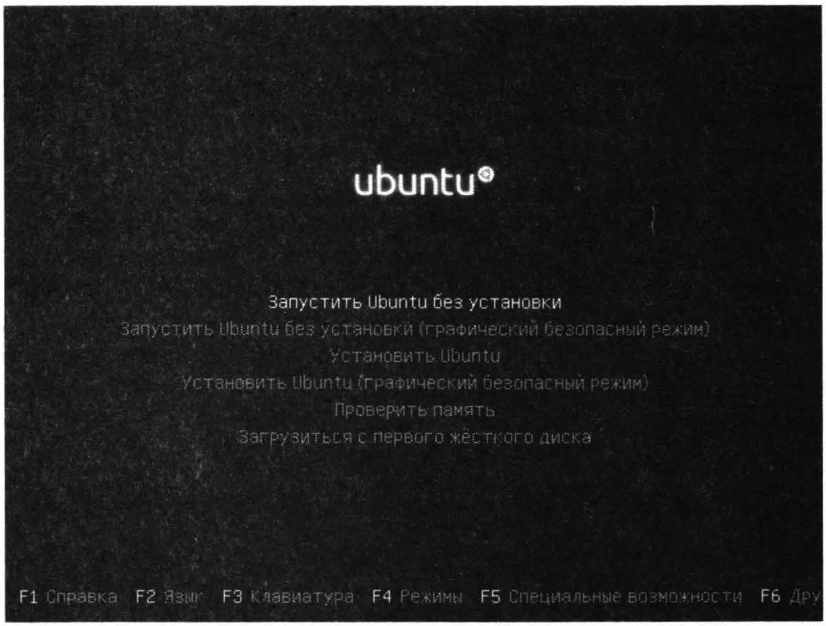


Рис. 2.3. Меню загрузчика Ubuntu

нуждается. Если вы хотите попробовать Ubuntu перед установкой, просто нажмите **Enter**, если же нужно сразу установить систему с помощью стрелок вверх/вниз выберите третий вариант (**Установить Ubuntu**) и нажмите клавишу **Enter**.

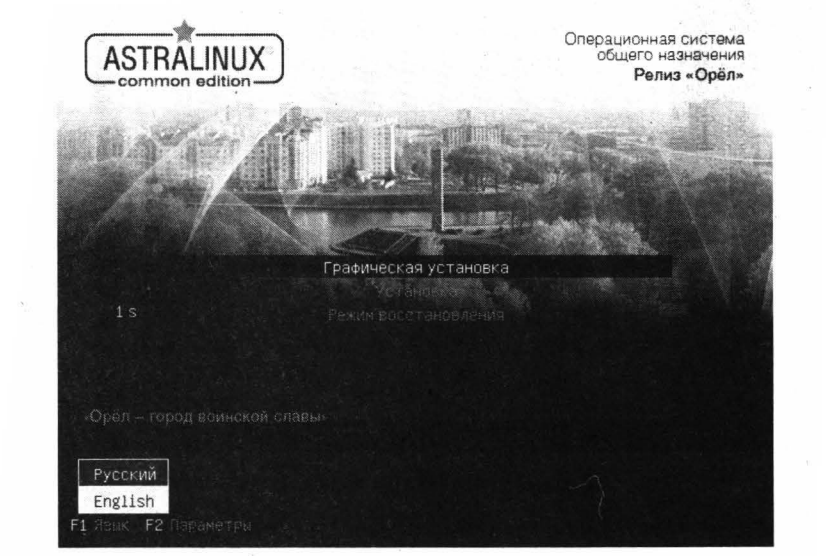


Рис. 2.4. Меню загрузчика Astra Linux

В случае с Astra Linux танцев с бубном меньше – вы сразу видите начальное меню загрузчика и можете выбрать или графическую установку или установку в текстовом режиме (команду **Установка**), что подойдет для слабых компьютеров или для серверов, где графический интерфейс не нужен.

По умолчанию Ubuntu запускается в режиме LiveCD, который позволяет пользователю попробовать дистрибутив перед установкой (рис. 2.5). Для запуска установки дважды щелкните по значку **Установить Ubuntu 20.04** на рабочем столе.

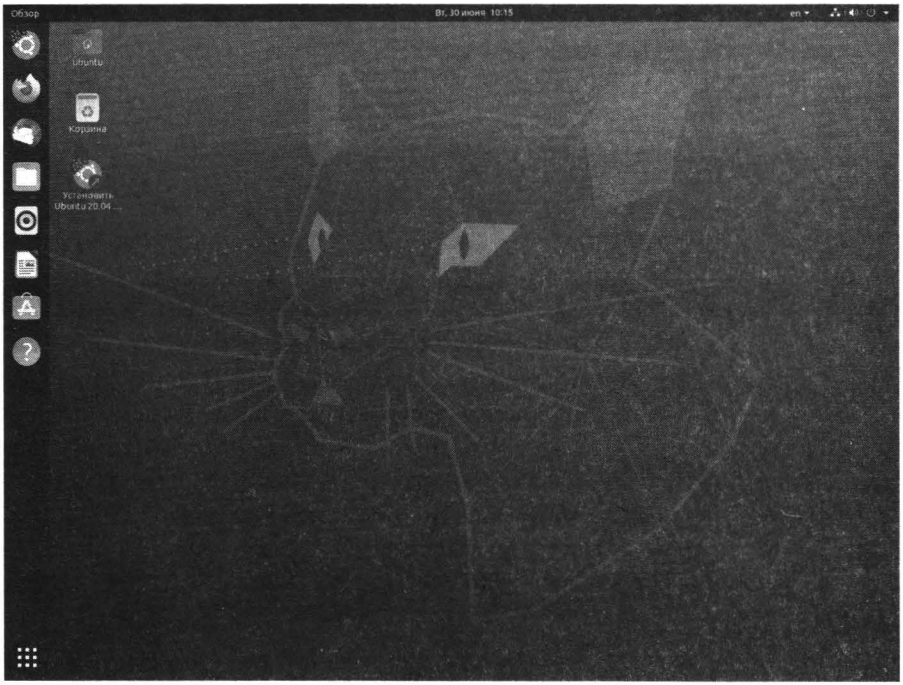


Рис. 2.5. Ubuntu в режиме LiveCD

2.4. Начало установки

Сразу после запуска инсталлятора (который в случае с Astra Linux произойдет автоматически, а в Ubuntu нужно запустить вручную):

- Ubuntu: Нужно выбрать язык и для продолжения установки нажать кнопку **Продолжить**.
- Astra Linux: прочесть лицензионное соглашение и нажать кнопку **Продолжить**.

Дистрибутив Astra Linux основан на дистрибутиве Debian и использует такой же инсталлятор. В некоторых моментах он удобнее, чем инсталлятор

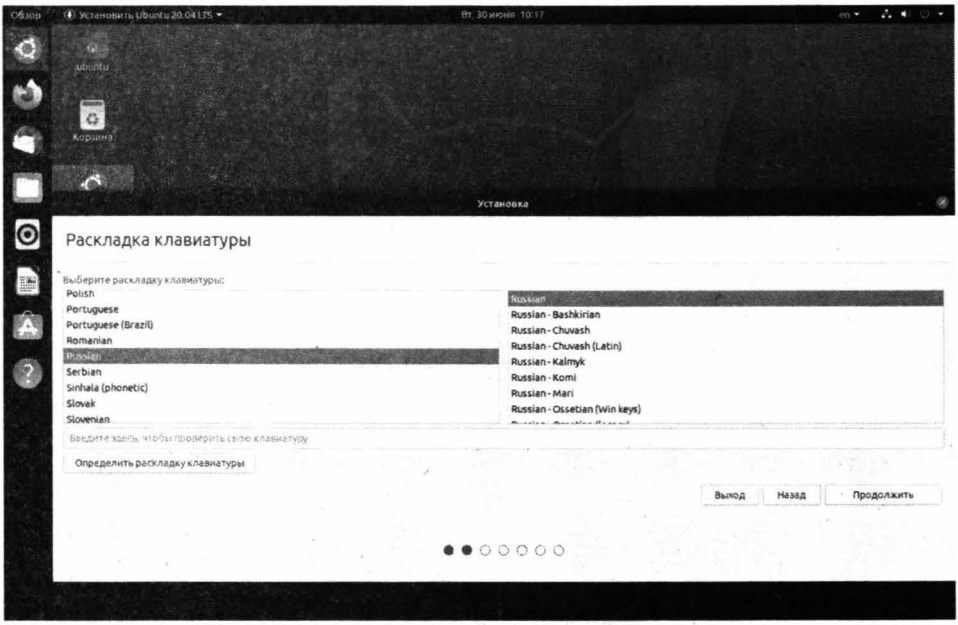


Рис. 2.6. Параметры клавиатуры (Ubuntu)

Ubuntu, в некоторых – нет. Например, после нажатия **Продолжить** в обоих случаях инсталляторы предложат выбрать раскладку клавиатуры (рис. 2.6, 2.7). Вот только в случае с Astra Linux вы можете выбрать комбинацию клавиш для переключения раскладки, что очень удобно. В Ubuntu вам придется это сделать после установки, если стандартная вам не подходит.

Рис. 2.7. Параметры клавиатуры (Astra Linux)

2.5. Разметка диска

2.5.1. Общие сведения о разметке диска

Один из самых важных моментов при установке любого дистрибутива Linux - это разметка жесткого диска. Если вы в процессе установки, например, забудете выбрать какой-то пакет - его очень просто установить после установки. Но вот если разметка диска будет выполнена неправильно, то исправить ситуацию будет гораздо сложнее. В некоторых случаях придется даже переустанавливать всю систему.

Linux использует свою файловую систему (некоторые дистрибутивы используют ext4, некоторые xfs, о файловой системе мы поговорим подробно в главе 13), поэтому ее нельзя установить в уже имеющиеся на жестком диске разделы. Если вы устанавливаете Linux не на новый жесткий диск, тогда желательно удалить все разделы сторонних операционных систем. Не думаю, что на сервере будут сожительствовать две операционных системы. Понятно, что это не нужно делать на домашнем компьютере, где, скорее всего, Linux придется сосуществовать с Windows.

Можно, конечно, положиться на автоматическую разметку инсталлятора, но она не всегда правильна. Даже если вы устанавливаете Linux на новый жесткий диск, то инсталлятор в большинстве случаев создаст два раздела – один под корневую файловую систему, а другой – для подкачки. Для домашнего использования такая схема вполне имеет право на жизнь. Для сервера – нет. Да и серверы бывают разными.

Если мы создаем FTP-сервер (он же сервер хостер-провайдера) и основное его назначение - хранение данных (например, сайтов) пользователей, то для каталога /home нужно выделить львиную долю дискового пространства. Например, если у вас жесткий диск размером 1 Тб, то для самой системы более чем достаточно 15 Гб дискового пространства (всего 1.5% от емкости жесткого диска), а для каталога /home нужно выделить оставшиеся почти 98% (почти - потому что понадобится еще место под раздел подкачки).

Если же у нас - корпоративный почтовик или сервер баз данных или элементарно прокси-сервер (или же корпоративный сервер, который сочетает в себе все эти функции), тогда основное дисковое пространство нужно выделить под точку монтирования /var. Именно в каталоге /var хранятся файлы базы данных, кэш прокси-сервера Squid, почтовые ящики и т.д.

Недостаток автоматической разметки диска в том, что она не предполагает установку назначения компьютера. Если можно было бы выбрать назначение компьютера, а уже потом выполнять саму разметку, все было бы совсем иначе.

Примечание для домашних пользователей. Если вы производите установку Linux на компьютер с уже установленной Windows, обязательно выполните резервное копирование всех важных данных! Иначе знакомство с Linux начнется для вас с потери данных - Linux нельзя установить на имеющийся раздел. Нужно сначала уменьшить размер этого раздела, а затем на освободившемся месте создать Linux-разделы. Для изменения размера раздела рекомендуется использовать сторонние приложения вроде Acronis, а не штатный инсталлятор Linux – чтобы не было больно за потерянные данные.

2.5.2. Введение в точку монтирования

Точка монтирования - это каталог корневой системы, через который осуществляется доступ к тому или иному разделу. По сути, можно раздел диска можно подмонтировать к любому из каталогов, но обычно используются определенные каталоги, которые имеют определенный смысл для системы, а именно:

- / - корневая файловая система, именно к подкаталогам этой файловой системы и осуществляется монтирование.
- /home - здесь хранятся пользовательские файлы.
- /mnt - обычно этот каталог используется для монтирования с целью доступа к данным, находящимся на другом разделе.
- /usr - сюда обычно устанавливается программное обеспечение (за исключением системного набора программ, который устанавливается в каталоги /bin и /sbin)
- /tmp - каталог для временных файлов.
- /var - здесь хранится переменная информация. К такой относят базы данных, почту, журналы и т.д.

Подробно о назначении каталогов и монтировании мы поговорим в главе 13. А пока вам нужно знать следующее. Можно установить Linux на один большой раздел. Скажем, у вас есть жесткий диск (имя устройства /dev/sda), вы на нем создаете всего два раздела - /dev/sda1 и /dev/sda2. Первый будет занимать большую часть дискового пространства, а второй будет размером несколько гигабайтов и будет использоваться для подкачки (подробнее см. след. раздел).

В этом случае все эти каталоги (/home, /usr, /var) будут находиться на одном разделе/диске. С одной стороны ничего страшного, но правильнее будет разместить хотя бы каталоги /home и /var на отдельных разделах, а еще лучше - на отдельных дисках или же организовать RAID-массив. Эти каталоги содержат самое ценное - пользовательские данные, поэтому если

они будут все находиться на одном жестком диске и он выйдет со строя, приятного будет мало.

Идеально, иметь три жестких диска. На первом жестком диске будет сама система и подкачка, на втором - пользовательские данные (каталог /home), на третьем - каталог /var. Однако, учитывая стоимость жестких дисков, такое расточительство можно себе позволить только, если есть прямая необходимость. Если пользователи не будут регистрироваться на сервере и тем более хранить на нем данные, то нет смысла отводить отдельный жесткий диск под каталог /home. Пусть он находится физически на том же диске, что и сама операционная система.

А вот каталог /var желательно разместить на отдельном жестком диске. Если диск всего один, то хотя бы на отдельном разделе.

Даже если жесткий диск один, все равно имеет смысл создавать отдельные разделы под системные точки монтирования. Ведь это позволяет на каждом разделе использовать свою файловую систему. Для корневой файловой системы, например, можно использовать файловую систему ReiserFS. Изюминка этой файловой системы в том, что в одном блоке может быть несколько небольших файлов. Например, если размер блока равен 4 Кб, то в нем может поместиться 4 файла по 1 Кб или 2 файла по 2 Кб. Такая файловая система идеально подходит для корневой /, где много файлов конфигурации - все они текстовые и многие из них имеют небольшой размер. Так дисковое пространство будет расходоваться экономнее.

А вот для каталога /var, где нужна высокая производительность, лучше использовать XFS. Это высокопроизводительная файловая система, рассчитанная на большие носители и большие размеры файлов. В общем, если надумаете развернуть Oracle-сервер - это правильный выбор.

Все сказанное ранее – хорошо для сервера. Для домашнего компьютера или для рабочей станции можно использовать два раздела – один для корневой файловой системы (которая будет содержать и данные, и программы), а второй – для подкачки.

2.5.3. Раздел подкачки

В отличие от Windows, Linux использует не файлы, а разделы подкачки. В принципе, если размера раздела подкачки будет недостаточно, то можно создать и файл подкачки, но правильнее создавать именно раздел - так производительность будет выше.

Теперь о размере раздела подкачки. Если оперативной памяти достаточно (например, 8 Гб или больше), тогда раздела подкачки в 8 Гб будет вполне достаточно (размер раздела подкачки равен размеру ОЗУ).

Если оперативной памяти мало, например, 4 Гб или меньше, тогда размер раздела подкачки должен в 2 раза превышать размер оперативной памяти, то есть 8 Гб будет достаточно. Увеличивать раздел подкачки больше не имеет смысла - ведь производительность от этого выше не станет. Назначение раздела подкачки - продолжение работы системы любой ценой. Когда заканчивается оперативная память, сервер должен продолжить работу, хоть и за счет потери производительности (жестким дискам далеко до производительности оперативной памяти). При небольшом объеме ОЗУ лучший тюнинг производительности - дополнительный модуль оперативной памяти.

Когда оперативной памяти много, иногда возникает соблазн вообще отказаться от раздела подкачки. Этого не нужно делать. Если оперативной памяти не хватит, возможны неприятные ситуации. При современных объемах жесткого диска потеря 8-16 Гб дискового пространства (под раздел подкачки) никак не отразится на работе всего сервера. Конечно, как уже было отмечено, в процессе работы системы можно создать и файл подкачки, но его производительность, как показывает практика, ниже, чем производительность раздела подкачки.

2.5.4. Как правильно разбивать жесткий диск?

Правильная разметка жесткого диска выглядит так:

- Для самой системы (точка монтирования /) будет достаточно всего 15 Гб дискового пространства, если не хотите жадничать, можно выделить 20 Гб, но не более
- Размер раздела подкачки должен быть равен размеру оперативной памяти или превышать его в два раза
- Разделы должны быть созданы, исходя из выполняемых сервером функций. Об этом мы уже говорили

Неправильная разметка чревата тем, что придется изменять конфигурацию диска уже в процессе работы сервера, а это чревато только одним - простоем. Потом вам придется выбирать - или не спать какую-то ночь или же получать недовольные звонки от пользователей и/или начальства - если придется перенастраивать сервер в рабочее время.

2.5.5. Ручная разметка в Ubuntu

Если вы не планируете использовать целевой компьютер для других операционных систем, можете выбрать вариант **Стереть диск и установить Ubuntu** (рис. 2.8). Идеальный выбор для нового домашнего/персонального компьютера.

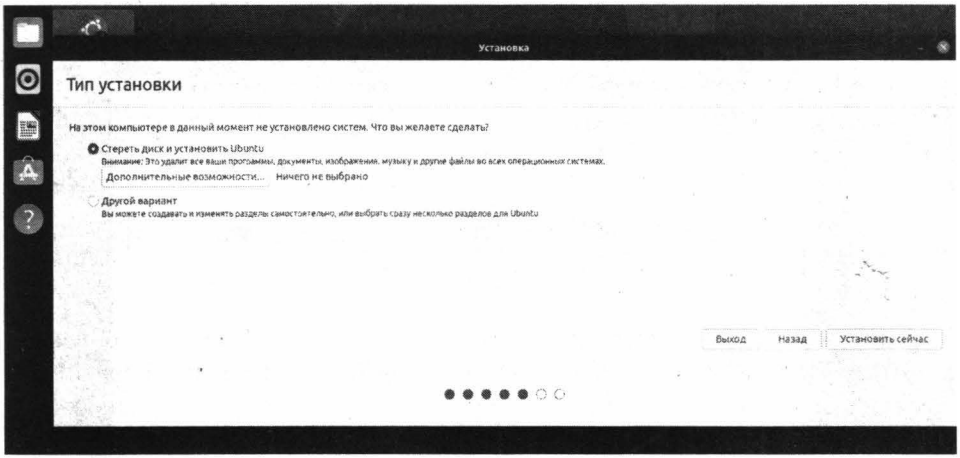


Рис. 2.8. Тип установки

Если же Linux будет использоваться вместе с другой операционной системой или же вы производите настройку сервера, где нужно создать отдельные разделы для разных точек монтирования, нужно выбрать **Другой вариант**.

Инсталлятор Ubuntu довольно удобен для ручной разметки диска. На рис. 2.9 показано, что жесткий диск абсолютно новый и на нем еще не была создана таблица разделов. Для ее создания нажмите кнопку **Новая таблица разделов** (если она уже создана, то кнопка не будет активной). В появившемся окне нажмите кнопку **Продолжить** для подтверждения.

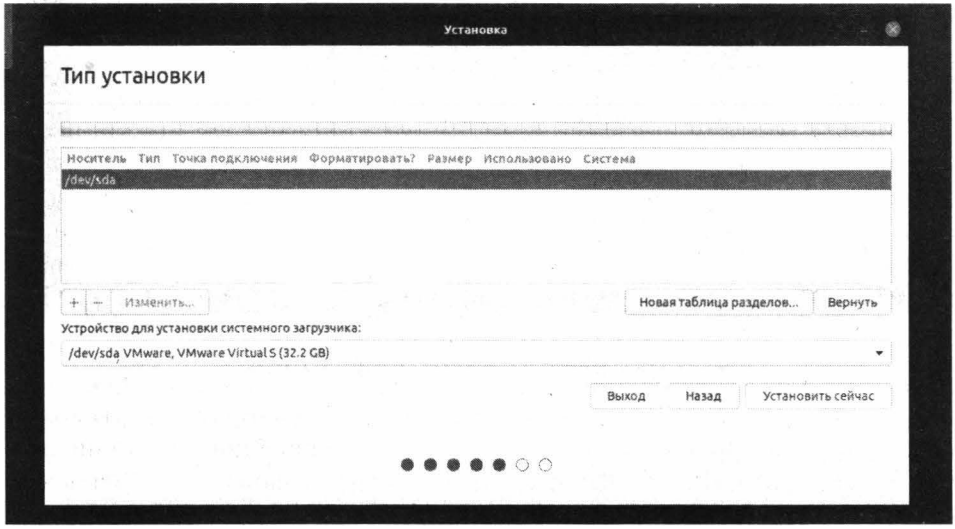


Рис. 2.9. Начало ручной разметки диска

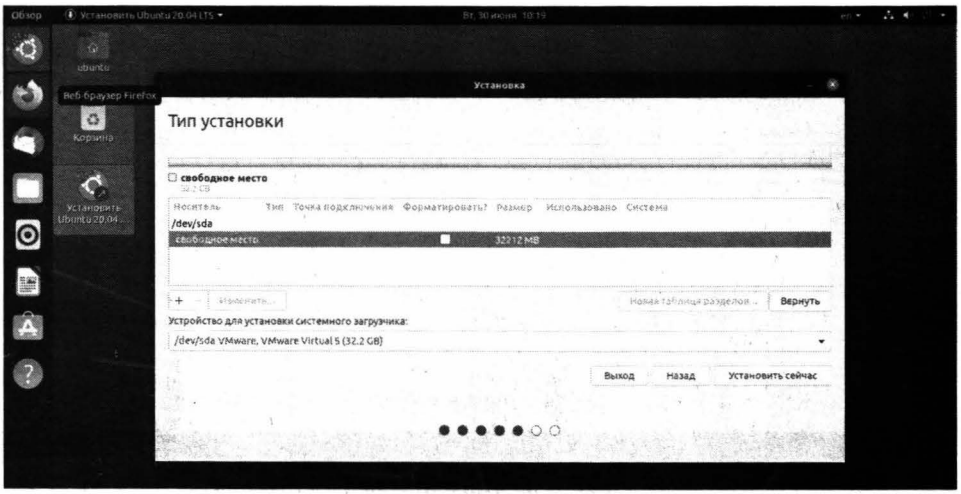


Рис. 2.10. После создания таблицы разделов

После этого вы увидите, что у вас появилось свободное место (рис. 2.10). Нажмите кнопку **+** для создания нового раздела. Введите размер создаваемого раздела, выберите точку монтирования и файловую систему (рис. 2.11).

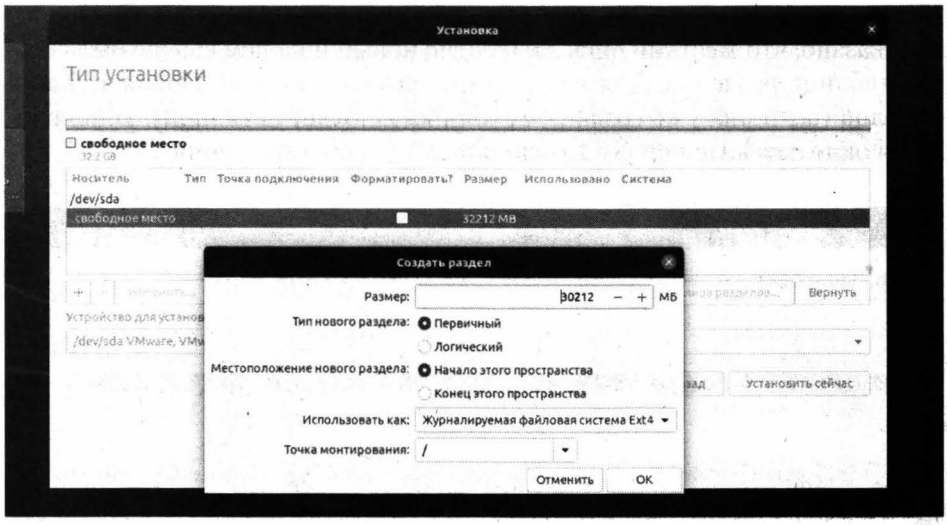


Рис. 2.11. Создание нового раздела

Здесь мы создаем раздел размером около 30 Гб для точки монтирования /, файловая система – ext4. На этом разделе будет содержаться и сама система, и данные пользователя.

Затем снова щелкните на свободном пространстве и снова нажмите **+**. Теперь нужно создать раздел подкачки, как показано на рис. 2.12.

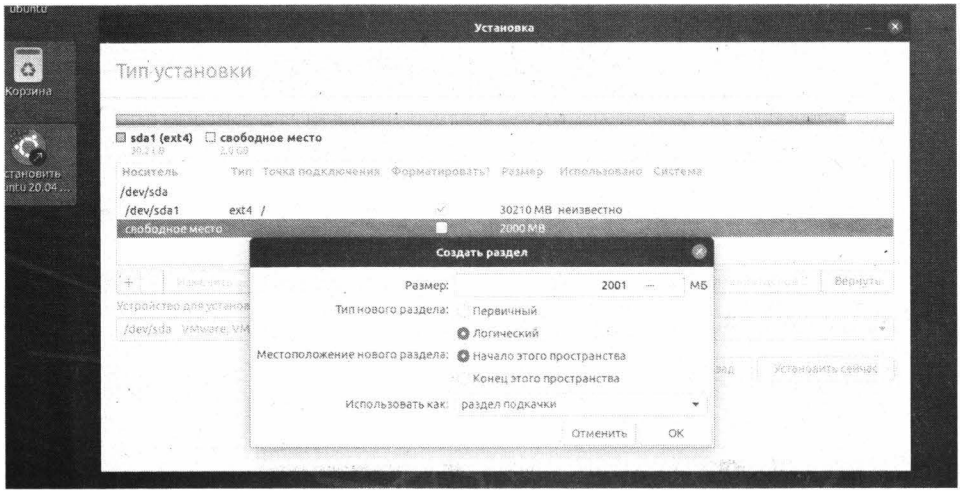


Рис. 2.12. Создание раздела подкачки

Созданная схема разметки изображена на рис. 2.13. Нажмите кнопку **Установить сейчас** для начала установки системы.

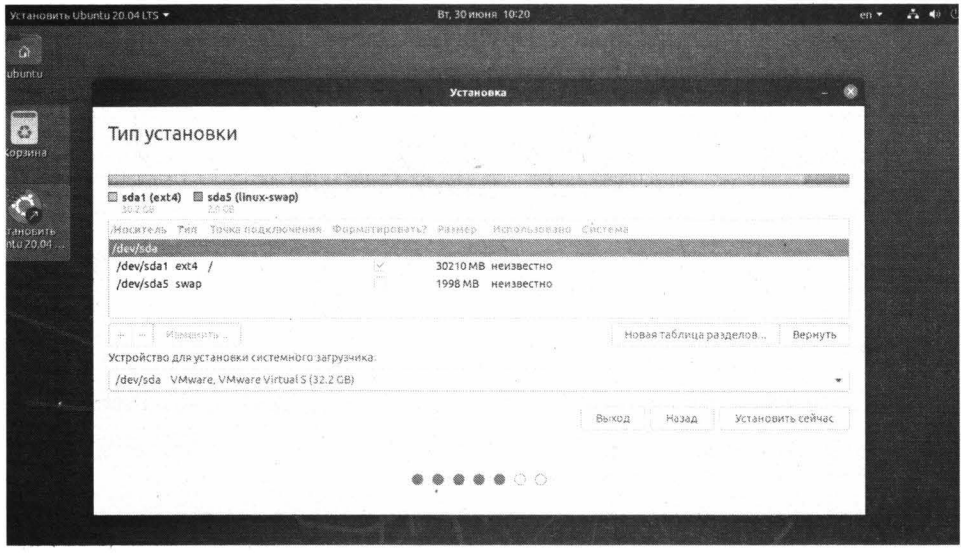


Рис. 2.13. Разметка завершена

2.5.6. Ручная разметка в Astra Linux

Установки в Astra Linux такой же, как в Debian. Debian – неплохой дистрибутив, но у него самый неудобный для разметки диска установщик. Наверное, над ним работали профессионалы по самым неудобным GUI. И у них это получилось!

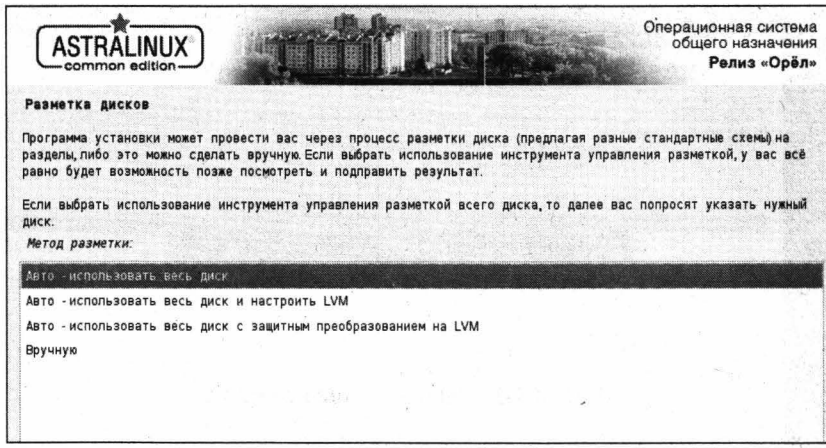


Рис. 2.14. Разметка в Astra Linux

Хорошо, если вы настраиваете персональный компьютер, на котором не будет никаких других операционных систем. Тогда можно выбрать вариант **Авто – использовать весь диск** и не заморачиваться (рис. 2.14)

В противном случае выбираем **Вручную**. Далее выбираем ваш жесткий диск и нажимаем кнопку **Продолжить** (рис. 2.15).

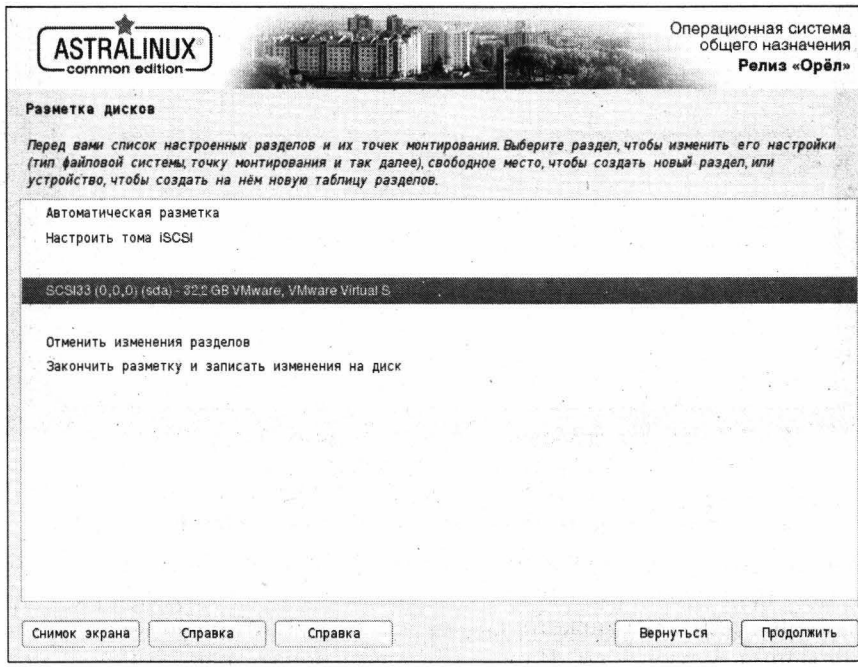


Рис. 2.15. Выберите жесткий диск и нажмите кнопку "Продолжить"

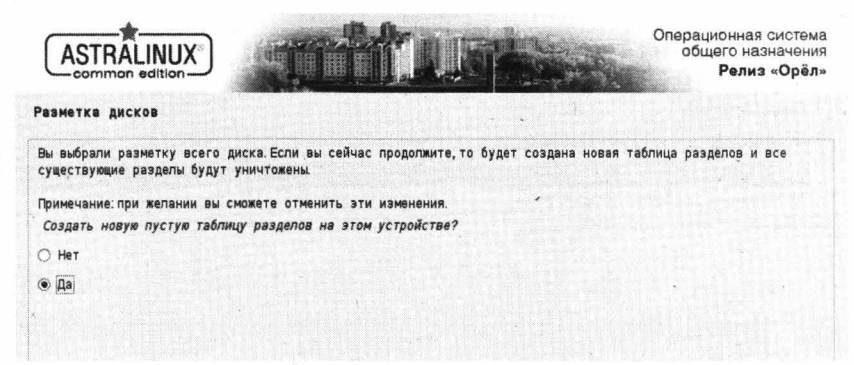


Рис. 2.16. Новая таблица разделов

Создайте новую таблицу разделов (рис. 2.16). Затем выделите свободное место (как показано на рис. 2.17) и снова нажмите кнопку **Продолжить**.

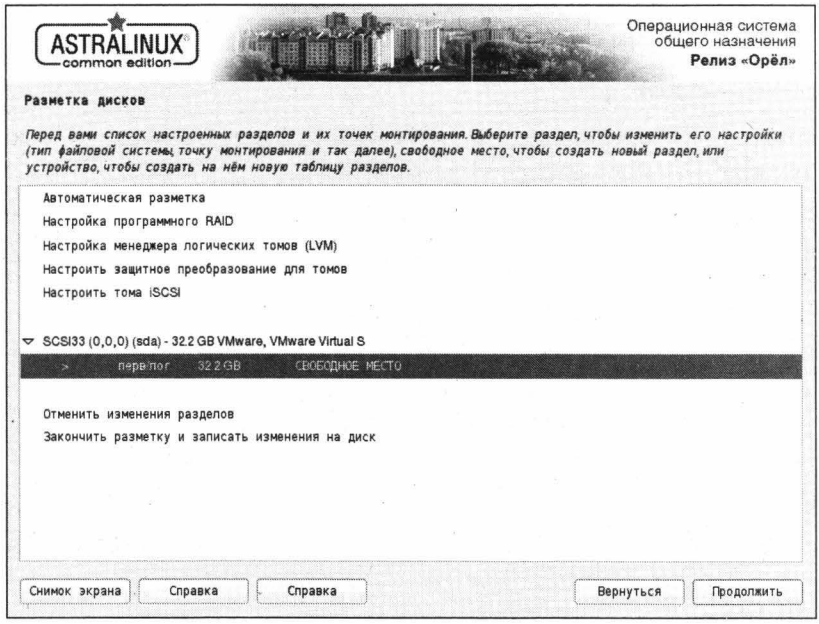


Рис. 2.17. Выберите свободное место и нажмите кнопку "Продолжить"

В появившемся меню (рис. 2.18) выберите команду **Создать новый раздел** и снова нажмите сами знаете какую кнопку. Нужно будет ввести размер раздела (рис. 2.19), выбрать его тип (первичный/логический), расположение (начало/конец), далее появится возможность установить другие параметры раздела (рис. 2.20). Вы можете изменить точку монтирования, параметры монтирования, метку и т.д. Параметр **Использовать как** позволяет изменить файловую систему раздела.

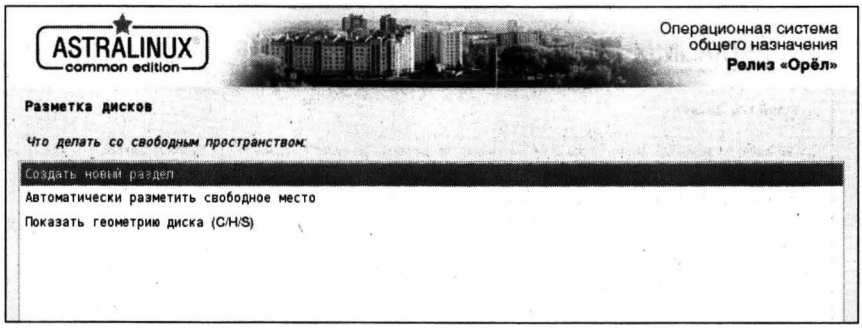


Рис. 2.18. Выберите команду "Создать новый раздел"

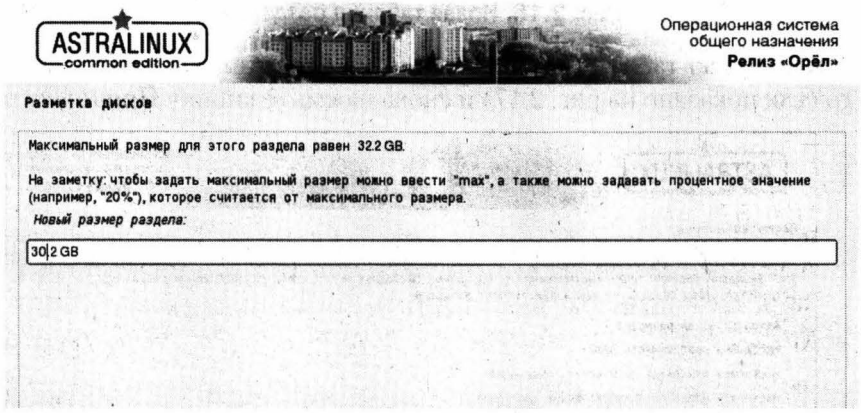


Рис. 2.19. Размер создаваемого раздела

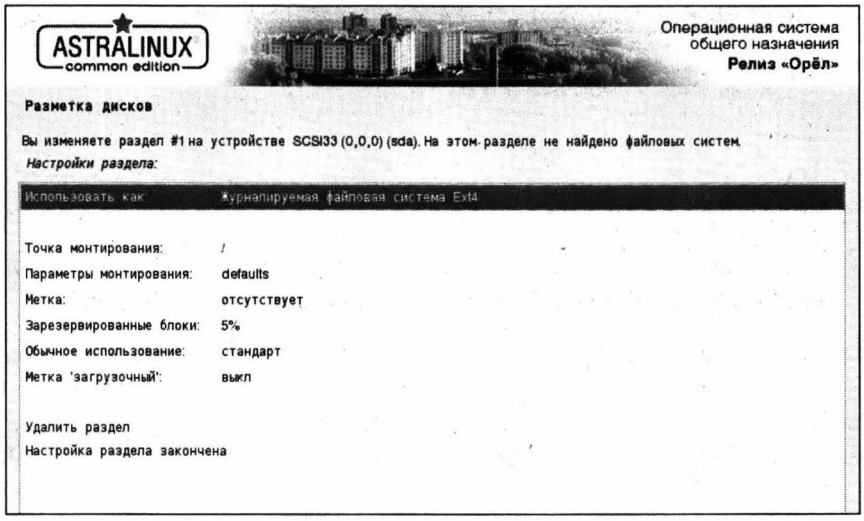


Рис. 2.20. Настройка раздела

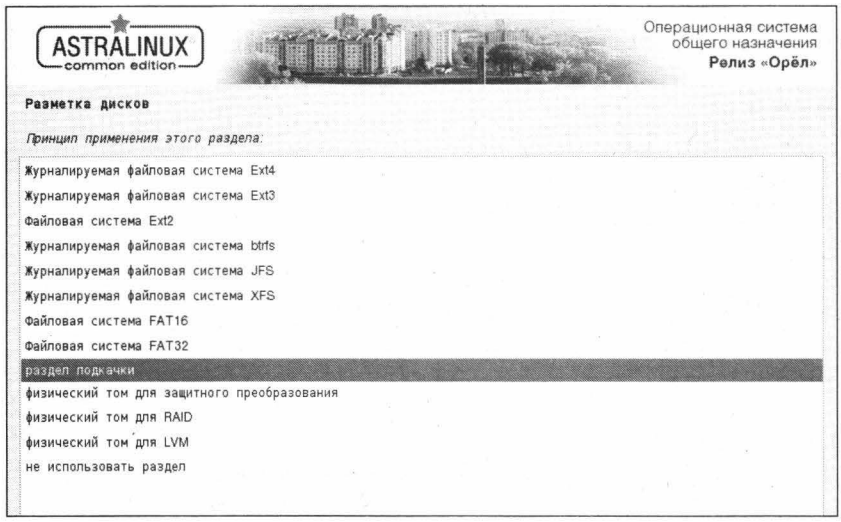


Рис. 2.21. Выбор файловой системы для раздела подкачки

Изменение происходит путем выделения нужного параметра и нажатия кнопки **Продолжить**. Когда все будет готово, используйте команду **Настройка раздела закончена**. Аналогичным образом создайте раздел подкачки (рис. 2.21).

Когда все будет готово, используйте команду **Закончить разметку и записать изменения на диск**. На следующем экране нужно согласиться с форматированием разделов (рис. 2.23). Средство разметки диска очень неудобное, но и к нему можно привыкнуть.

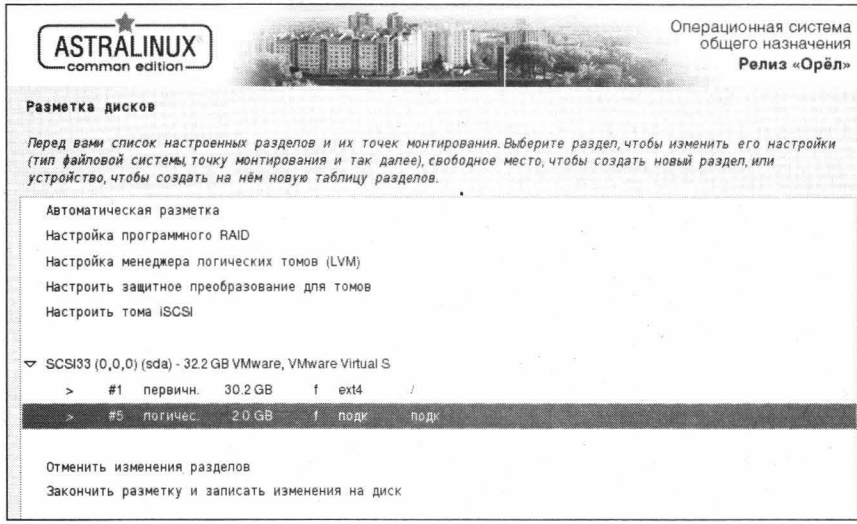


Рис. 2.22. Готовая разметка

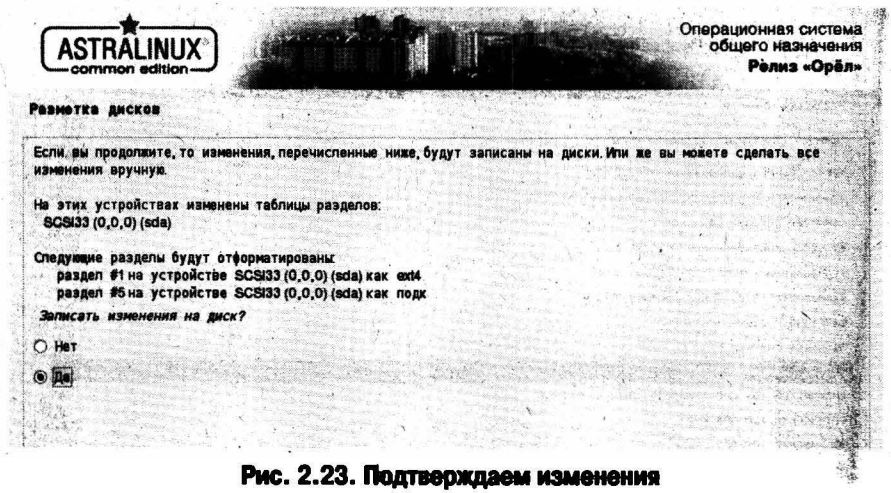


Рис. 2.23. Подтверждаем изменения

2.6. Установка пароля администратора

Раньше при установке Linux предлагалось создать пароль пользователя root – пользователя с максимальными правами и создать обычного пользователя. Сейчас же дистрибутивы отошли от использования пользователя root по соображениям безопасности, а эта учетная запись часто оказывается заблокированной. Делается это по понятной причине – имя root знают все, поэтому злоумышленнику нужно подобрать только один параметр – пароль, а если он не будет знать имя пользователя, то придется угадывать еще и логин, что усложняет задачу.

Современные дистрибутивы предлагают создать так называемого административного пользователя – пользователя, которому разрешено выполнять команду sudo для получения максимальных полномочий. Это и безопасно, и удобно для самого пользователя – ему нужно помнить один пароль, а не два (пользователя root и обычного пользователя).

При создании административного пользователя (рис. 2.24) вы задаете имя пользователя и его пароль. Ubuntu допускает использование слабого пароля: если пароль является слабым для подбора, инсталлятор просто сообщает об этом. А вот Astra Linux не позволяет использовать пустые пароли – пароль должен быть минимум 8 символов и содержать хотя бы буквы с цифрами.

Также обратите внимание на параметр **Входить в систему автоматически** (Ubuntu). Он позволяет не вводить пароль при входе в систему, что хорошо для домашних компьютеров. В Astra Linux подобный параметр доступен после установки системы – он называется **Включить автологин в графиче-**

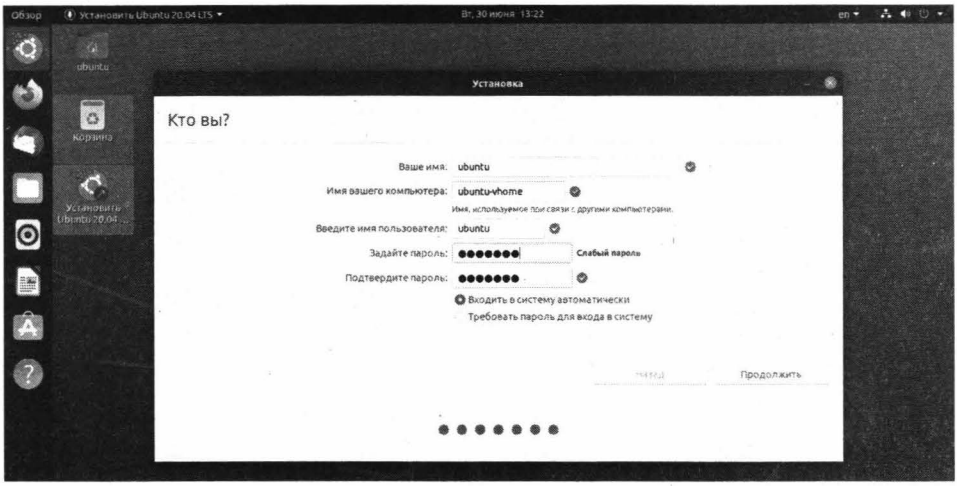


Рис. 2.24. Создание пользователя в Ubuntu

Рис. 2.25. Простые пароли в Astra Linux использовать нельзя

скую сессию (рис. 2.26). В следующей главе будет показано, как включить/выключить автовход уже после установки системы.

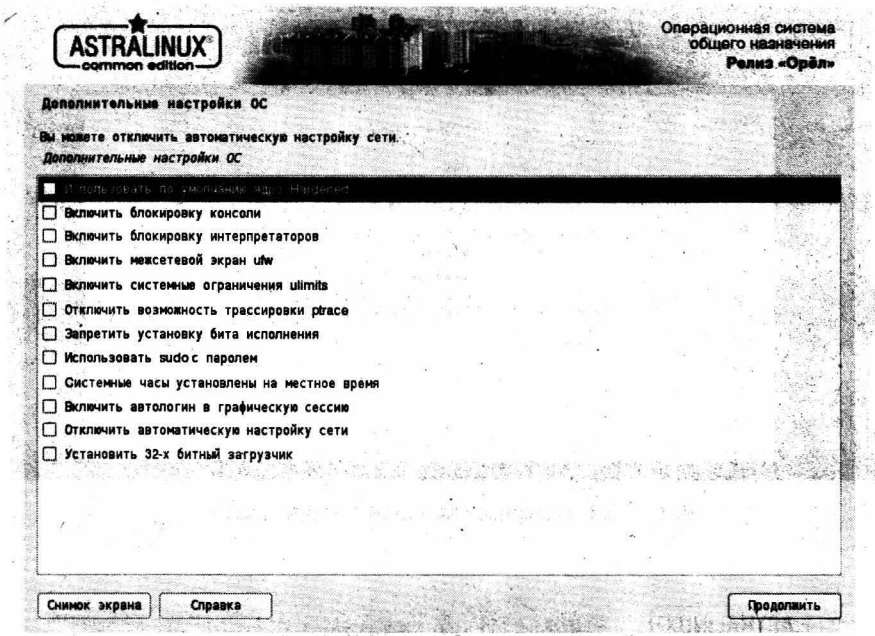


Рис. 2.26. Включение автовхода в Astra Linux

2.7. Параметры программного обеспечения

В Astra Linux позволяют выбрать наборы программного обеспечения (рис. 2.27). В принципе любой пакет можно доустановить после установки системы, но при желании вы можете сразу выбрать нужную группу. Опять-таки если вы заботитесь о дисковом пространстве, лучше не устанавливать готовые наборы, а установить только те пакеты, которые вам нужны. Некоторые, например, Игры можно сразу отключать – вы ними не будете пользоваться.

В Ubuntu ничего такого нет. При установке можно выбрать только обычную или минимальную установку, а также выбрать обновление ПО во время установки и опцию установку стороннего ПО. Рекомендую включить этот переключатель (рис. 2.28), чтобы сразу у вас заработали мультимедиаформаты. Кодеки нельзя по условиям лицензии распространять вместе с Linux, но их можно установить после установки системы (бесплатно). Этот переключатель как раз и производит данную установку.

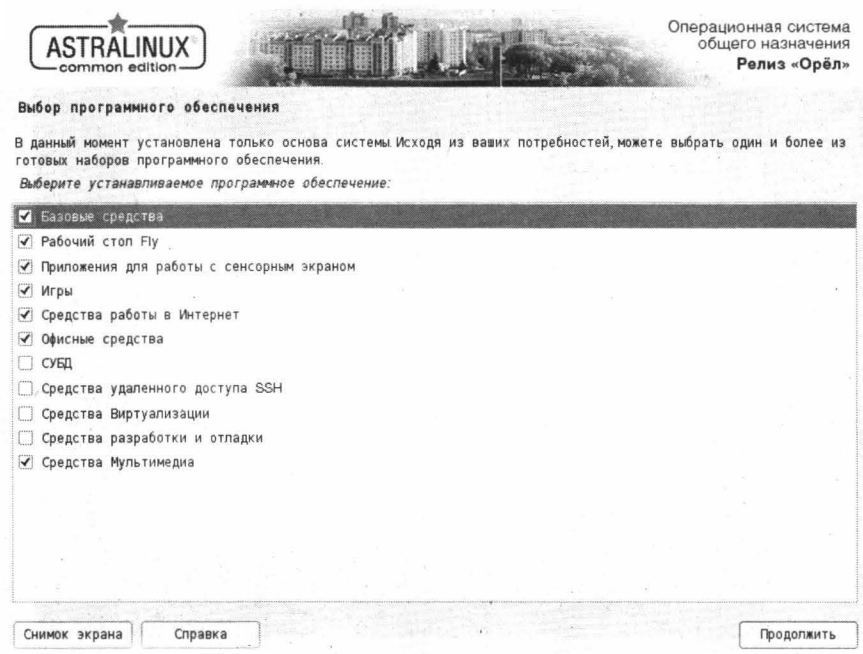


Рис. 2.27. Доступные по умолчанию наборы ПО

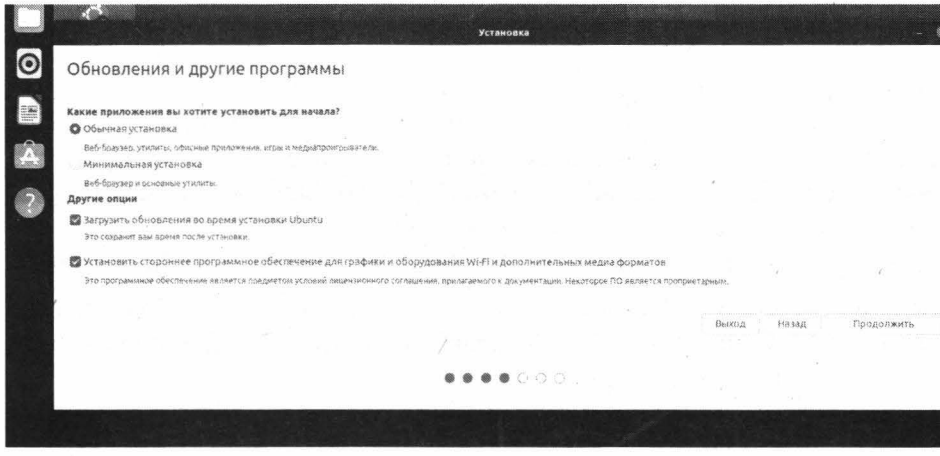


Рис. 2.28. Настройки ПО в Ubuntu

Сразу после установки Ubuntu предложит вам доустановить необходимые программы (рис. 2.29). Вы можете сделать это сразу, а можете по мере необходимости.

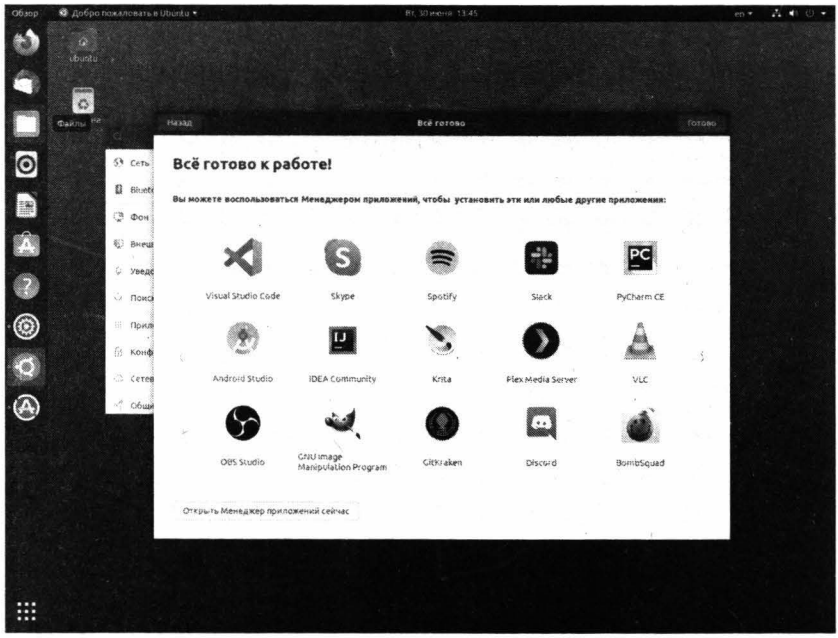


Рис. 2.29. Предложение установить некоторые программы

2.8. Установка загрузчика

Ubuntu устанавливается проще. По сути, самое сложное – это разметка диска. Но в Astra Linux нужно еще выбрать, куда устанавливать загрузчик Linux. Обычно его нужно установить в /dev/sda. Но если у вас есть уже другая операционная система и вы хотите загружать Linux не с помощью родного загрузчика GRUB2, а посредством стороннего загрузчика, тогда нужно или установить загрузчик в другое место или же не устанавливать его вовсе.

Мы рассмотрели все основные моменты, на которые нужно обратить внимание при установке Linux. В следующей главе мы поговорим о входе в систему и о завершении работы.

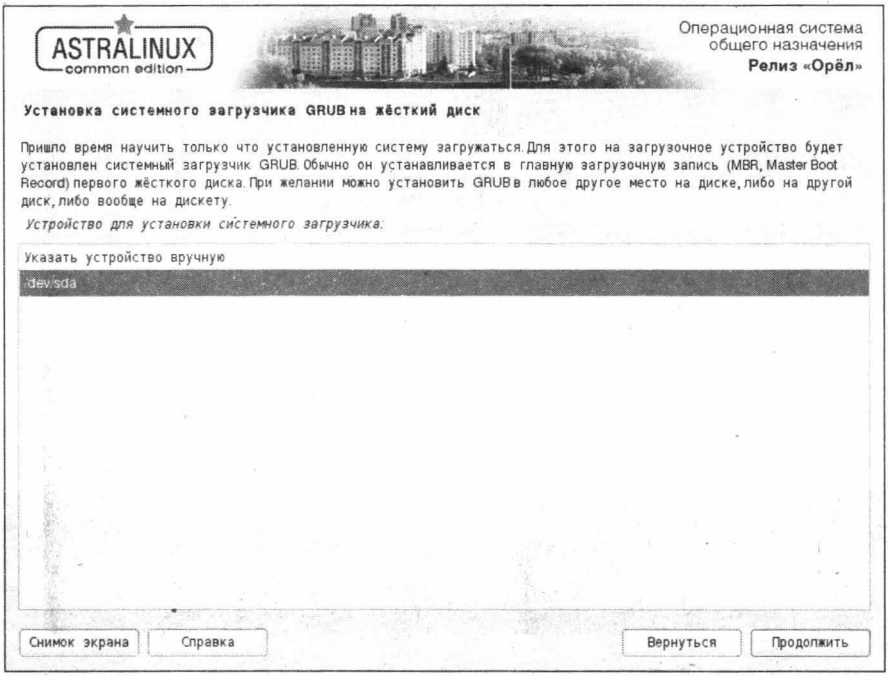
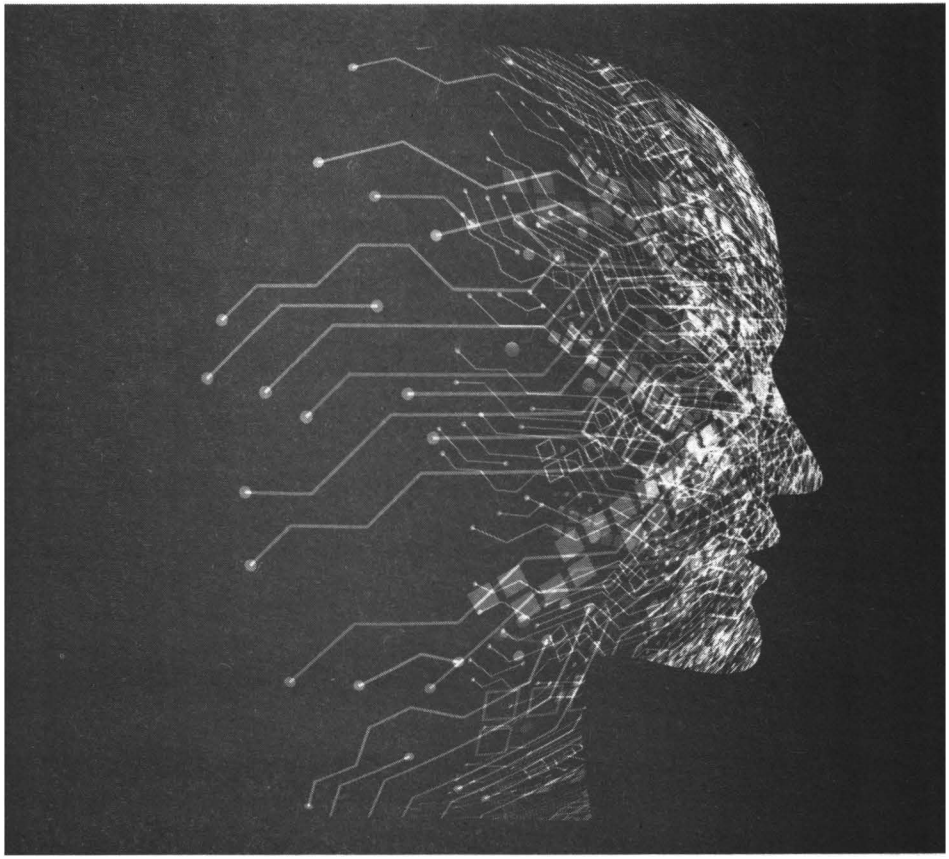


Рис. 2.30. Выбор устройства для установки загрузчика

Глава 3.

Вход в систему



В этой небольшой главе мы рассмотрим вход в систему, основные элементы графического интерфейса, а также правильное завершение работы в системе.

3.1. Вход в консоль и переключение между ними

Вход в систему в Linux ничем не отличается от входа в систему в любой операционной системе. Нужно ввести имя пользователя и пароль. Вход может быть как в текстовом, так и в графическом режиме. На сервере графический интерфейс, как правило, не устанавливается, чтобы не расходовать драгоценные системные ресурсы на никому ненужный графический интерфейс. На рис. 3.1 изображен вход в Astra Linux. На рис. 3.2 изображен графический вход в Astra Linux.

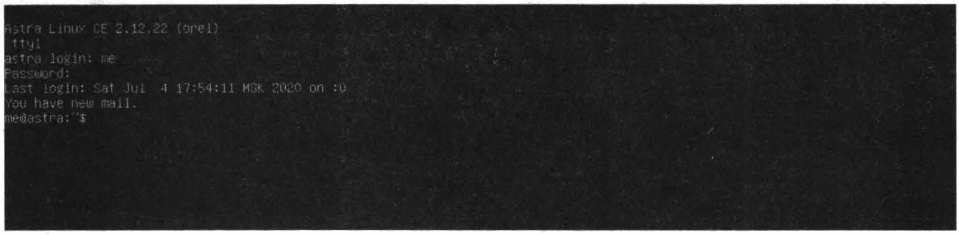


Рис. 3.1. Вход в систему (Astra Linux)

Давайте разберемся, что есть что. Самая первая строка (рис. 3.1) - название и версия дистрибутива. Далее выводится имя терминала, на котором вы сейчас находитесь (tty1). Некоторые другие дистрибутивы еще сообщают версию ядра и архитектуру систему. Здесь разработчики посчитали лишним вывод этой информации.

Далее нужно ввести логин (в нашем случае root) и пароль. Обратите внимание, что пароль не отображается при вводе, не отображаются даже звездочки. В графическом режиме вместо введенных символов могут отображаться звездочки или точки.

После входа в систему вы увидите или приглашение командной строки или графический интерфейс. В первом случае приглашение будет иметь вид #

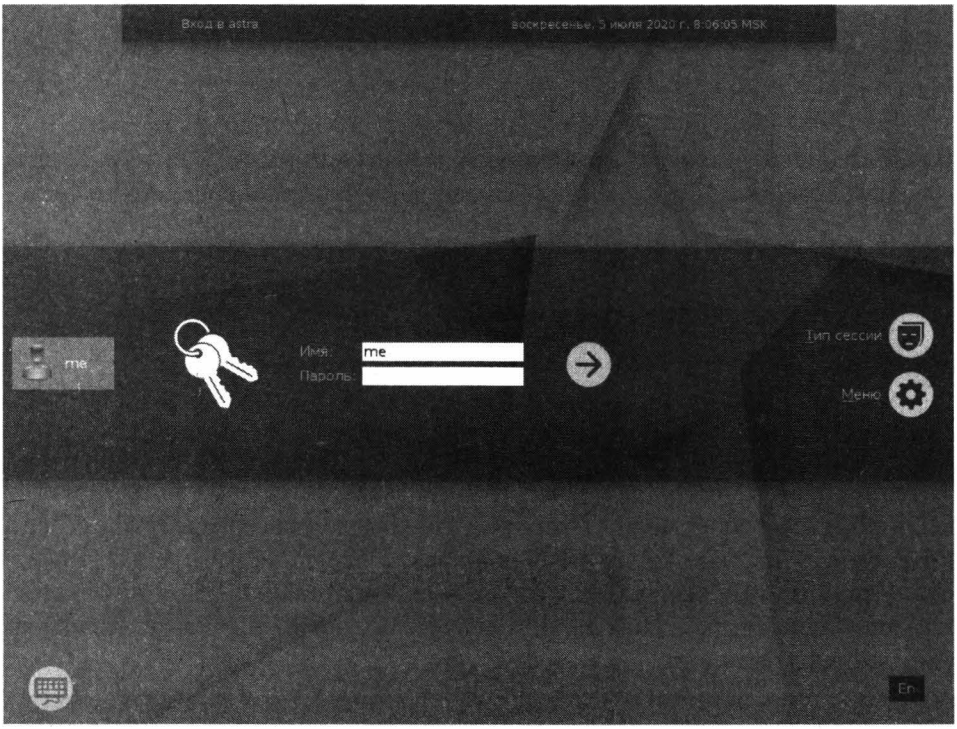


Рис. 3.2. Графический вход (Astra Linux)

(если вы вошли как root) или **$** (если вы вошли, как обычный пользователь).

Перед приглашением командной строки выводится время и дата последнего входа в систему, а также терминал, на котором был осуществлен вход (в нашем случае - tty1). Значение :0 означает, что последний вход был в графическом режиме.

Строка **You have mail** означает, что у вас есть новые сообщения электронной почты. Для их чтения нужно или использовать какую-то почтовую программу, настроенную на чтение системных сообщений e-mail, либо же команду mail.

Для переключения между терминалами используются комбинации клавиш Alt + F1 ... Alt + F6. Комбинация клавиш Alt + F7 переключит вас из консоли в графический режим, если система X11 запущена.

Находясь в графическом режиме, вы можете нажать комбинацию клавиш Ctrl + Alt + Fn, где n - это число от 1 до 6 для переключения на нужную вам консоль. Функционал экрана входа в систему отличается в зависимо-

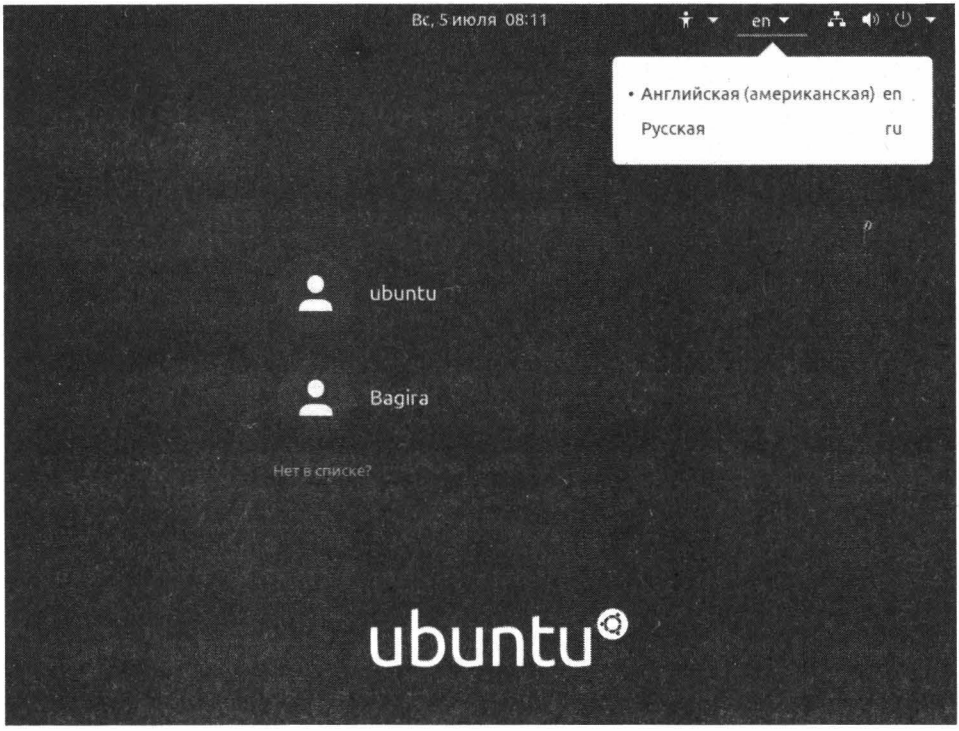


Рис. 3.3. Вход в систему (Ubuntu 20.04)

сти от дистрибутива – здесь все на усмотрение разработчиков. В Ubuntu он минималистичен – выводятся только имена пользователей, после выбора пользователя можно будет ввести пароль и войти в систему. Из полезных «фич» – только смена раскладки клавиатуры (рис. 3.3).

В Astra Linux можно выбрать тип сессии (рис. 3.4), а также открыть меню (рис. 3.5), в котором будут полезные команды, такие как: переключение на консоль (на случай, если вы не знаете ничего о комбинации Ctrl + Alt + F1), вызов экранной клавиатуры, перезапуск X-сервера и команда выключения системы.

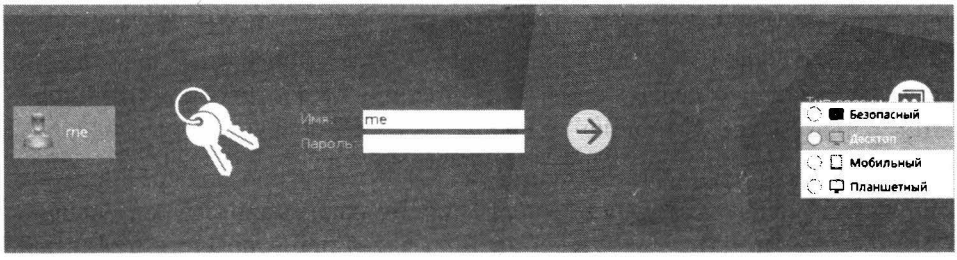


Рис. 3.4. Выбор типа сеанса (Astra Linux)

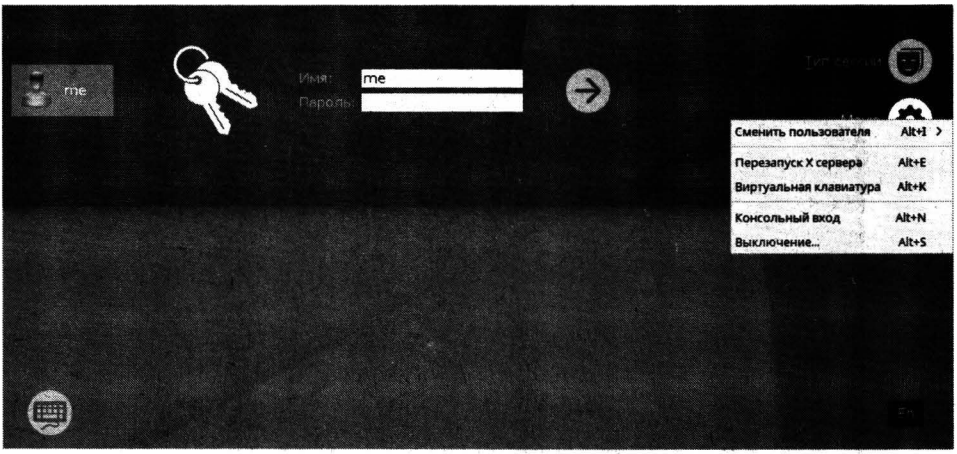


Рис. 3.5. Меню входа в систему (Astra Linux)

3.2. Основные элементы графического интерфейса

Вы вошли в систему, но что делать дальше? Для бывшего Windows-пользователя слегка непривычно. Далее мы рассмотрим основные элементы интерфейса пользователя Ubuntu и Astra Linux.

3.2.1. Интерфейс Ubuntu

По умолчанию в Ubuntu установлена графическая среда GNONE 3.36. Интерфейс минималистичен. Ничего лишнего.

Слева выводится панель, на которую можно поместить кнопки вызова часто используемых приложений (рис. 3.6). В самом низу этой панели есть кнопка, позволяющая открыть экран Приложения (рис. 3.7).

Обратите внимание: по умолчанию этот экран отображает только популярные приложения (те, которые вы чаще всего использовали) и может показаться, что приложений очень мало. На самом деле это не так. Перейдите на вкладку **Все** (внизу экрана) и вы увидите все установленные приложения. Поле **Найти** вверху экрана позволяет быстро найти то или иное приложений.

Чтобы добавить значок приложения на панель слева, просто перетащите его в нужную позицию или же щелкните по значку правой кнопкой мыши и выберите команду **Добавить в избранное**. Аналогично, команда **Удалить из избранного** удаляет этот значок с панели избранного.

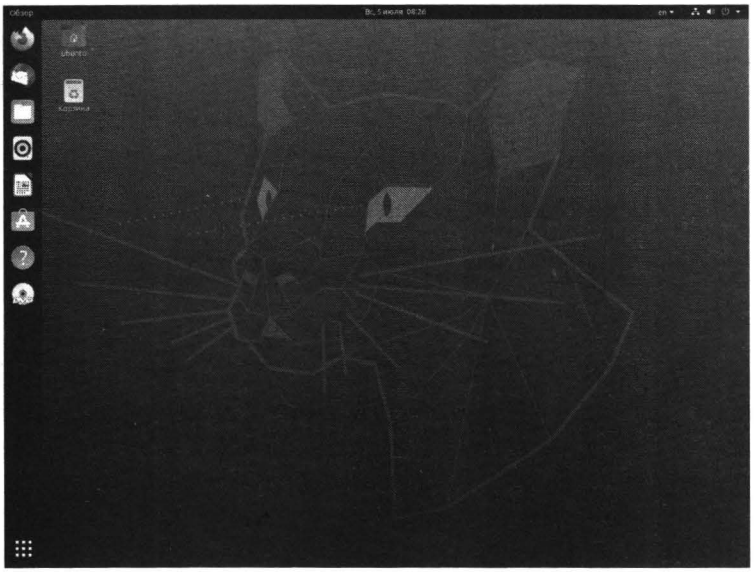


Рис. 3.6. Рабочий стол Ubuntu 20.04

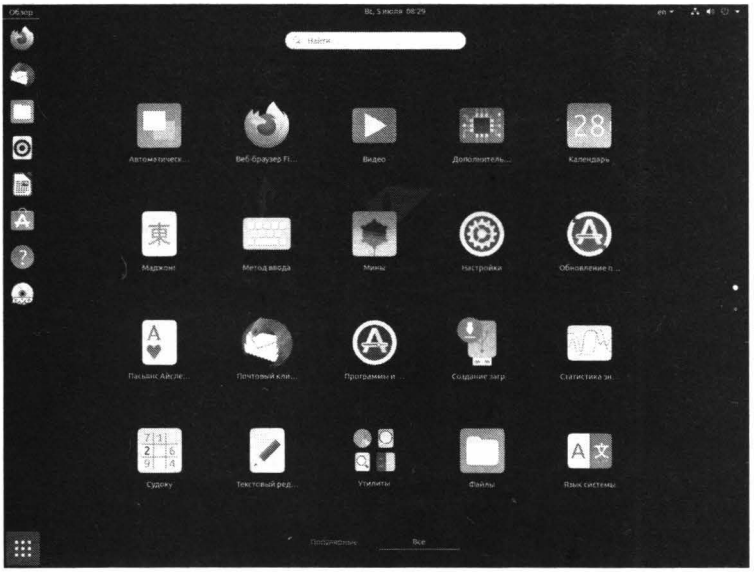


Рис. 3.7. Приложения

По умолчанию на панели избранного находятся следующие приложения (сверху вниз):

- Браузер Firefox.

- Почтовый клиент Thunderbird.
- Файловый менеджер.
- Музыкальный проигрыватель.
- Текстовый процессор Writer.
- Центр установки ПО Ubuntu Software.
- Кнопка вызова справочной системы.
- Кнопка быстрого доступа к DVD.

Когда вы запускаете приложение, его значок также добавляется на панель избранного. Другими словами, она выполняет роль еще и панели задач, которую вы можете использовать для переключения между приложениями. Запущенные приложения отмечаются кружочком слева от значка кнопки. На рис. 3.8 показано, что запущены браузер, файловый менеджер и терминал (средство для ввода команд).

Рис. 3.8. Демонстрация запущенных приложений

Существует несколько способов переключения между запущенными приложениями:

1. Нажатие на кнопку приложения на панели избранного.
2. Комбинация клавиш Alt + Tab, выводящая все запущенные приложения (как в Windows).
3. Кнопка **Обзор** в верхнем левом углу экрана (рис. 3.9).

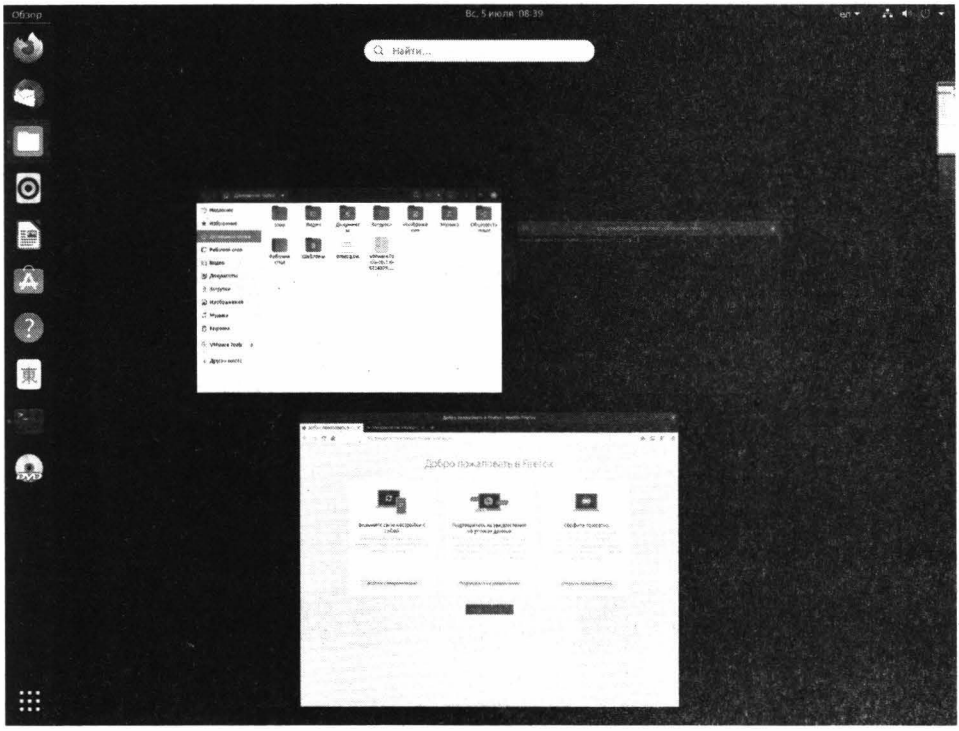


Рис. 3.9. Экран Обзор

В верхнем правом углу экрана находятся довольно важные элементы графического интерфейса: переключатель раскладки клавиатуры и меню, позволяющее управлять сетью, звуком и завершением работы. Нажмите на него (рис. 3.10). Вы увидите ползунок, позволяющий регулировать громкость, далее надпись «Проводное сеть подключена» означает, что на данный момент компьютер подключен к проводной локальной сети, нажатие по этой надписи позволит управлять другими сетевыми соединениями. Кнопка **Настройки** вызывает средство настройки графического интерфейса. Кнопка **Заблокировать** позволяет заблокировать экран пользователя – полезно, если вы хотите ненадолго отойти. Кнопка **Выключить / Завершить сеанс** открывает меню завершения работы:

- **Завершить сеанс** – производит выход пользователя из системы, но не выключает компьютер. Если компьютер используется не только вами, данная команда используется, когда вы закончили работу и готовы передать компьютер другому пользователю.
- **Сменить пользователя** – позволяет войти под именем другого пользователя, но не завершать сеанс текущего пользователя.

- **Ждущий режим** – переводит компьютер в ждущий режим для экономии электричества. Компьютер не завершает работу, приложения не закрываются, но отключать его от питания нельзя, иначе все несохраненные изменения будут потеряны.
- **Выключение** – вызывает дополнительное меню, в котором вы сможете или выключить компьютер или перезагрузить его.

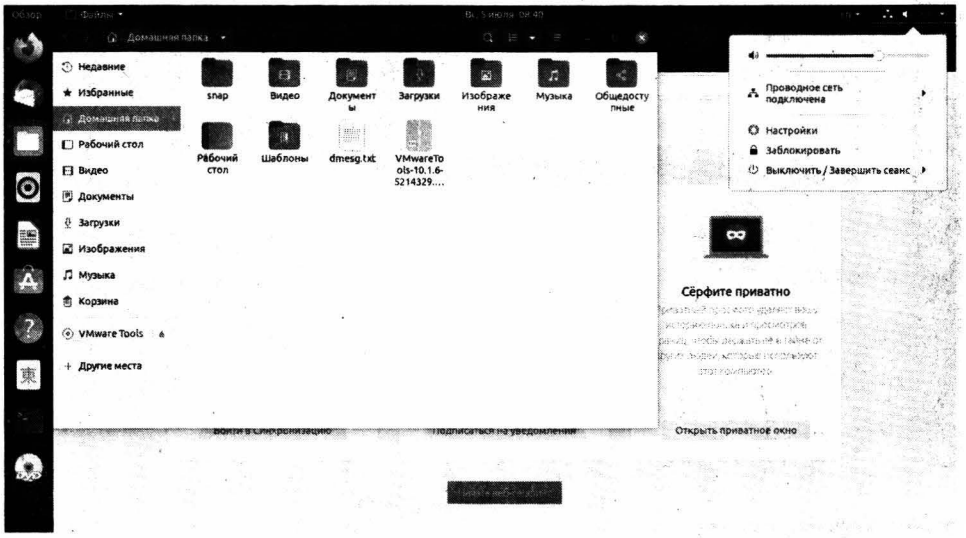


Рис. 3.10. Системное меню

Выберите команду **Настройки**. Сейчас «пройдемся» по наиболее полезным настройкам графического интерфейса. Настроек очень много, учитывая, что интерфейс достаточно качественно переведен на русский язык, не составит особого труда разобраться со всеми настройками самостоятельно. Но на некоторые следует обратить внимание.

В разделе **Фон** можно изменить фон рабочего стола (рис. 3.11). Кнопка **Добавить изображение** позволяет добавить в галерею пользовательское изображение, если стандартные вам не нравятся.

> Примечание. Много хороших (и главное бесплатных) обоев можно найти по адресу https://unsplash.com/wallpapers. На этом сайте вы найдете обои на любой вкус.

Раздел **Внешний вид** позволяет выбрать модную нынче темную тему оформления. Нужно отметить, что темная тема впервые появилась в Ubuntu 20.04, так что на сегодняшний день – это новинка. Также здесь можно выбрать расположение панели задач. Уверен, что пользователи Windows выберут положение **Снизу**, чтобы было привычнее.

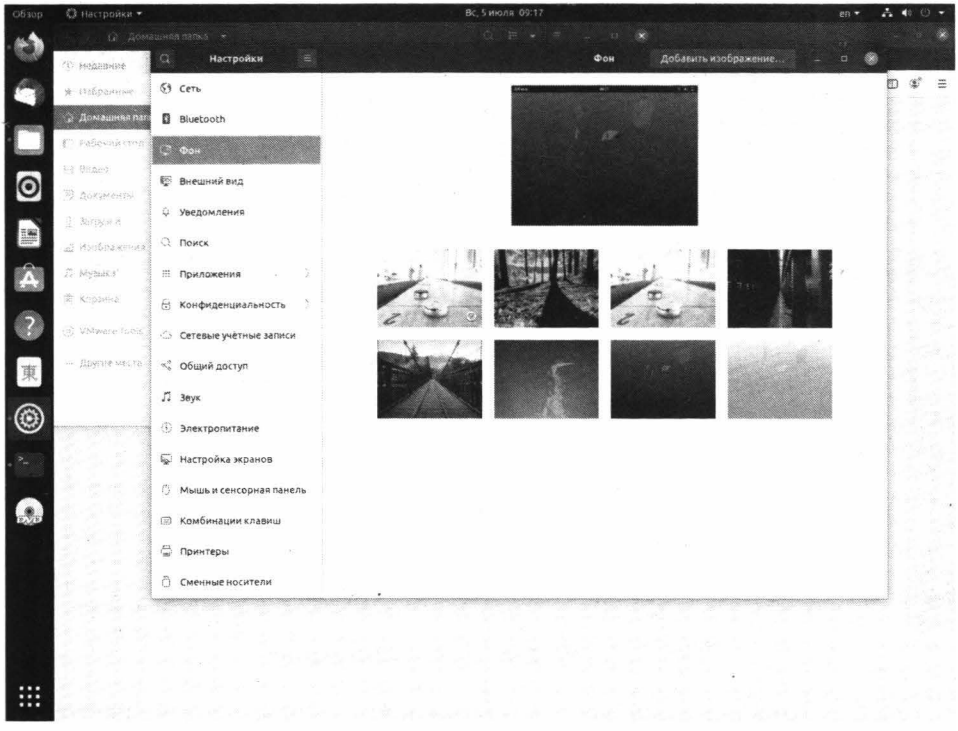


Рис. 3.11. Раздел "Фон"

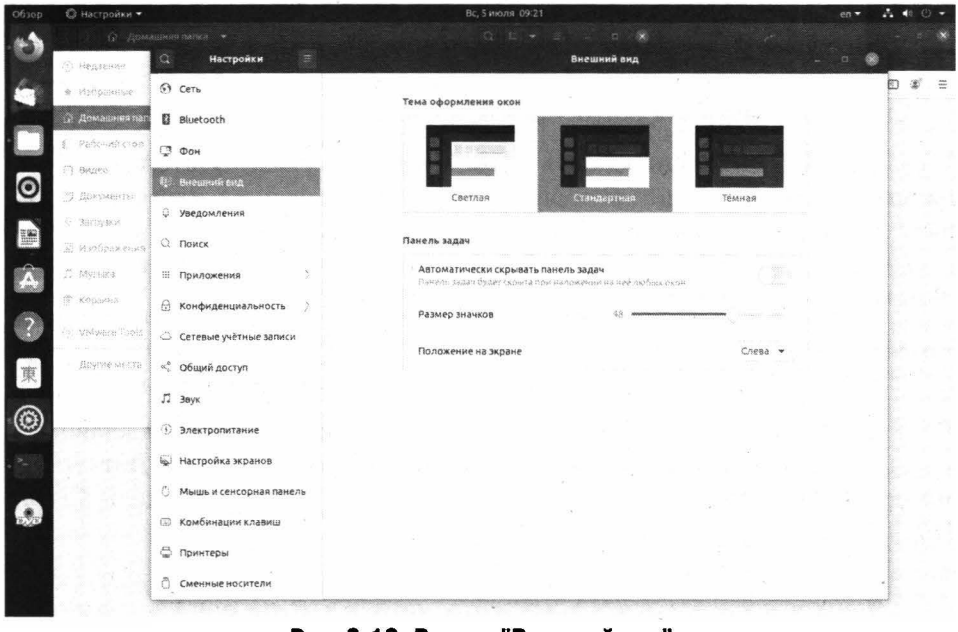


Рис. 3.12. Раздел "Внешний вид"

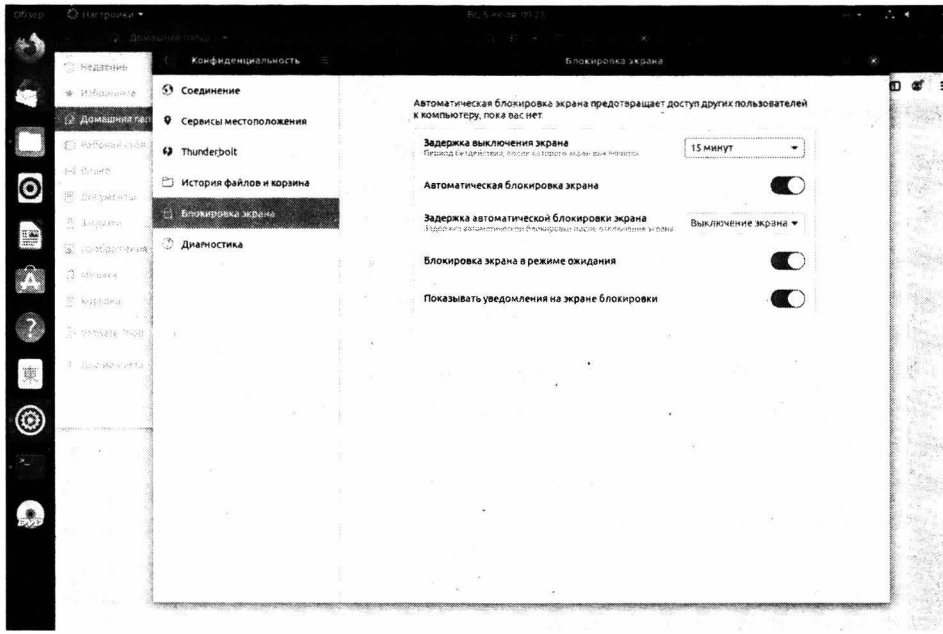


Рис. 3.13. Раздел "Блокировка экрана"

В разделе **Конфиденциальность, Блокировка экрана** можно настроить или вовсе отключить блокировку экрана. Задержка выключения экрана в 5 минут просто раздражает – отвлекся и экран уже потух и заблокировался, после чего нужно вводить пароль заново. Поэтому, если вы не страдаете паранойей, задержку выключения экрана можно установить в 15 минут, остальные параметры оставьте без изменения. Так будет комфортнее.

В разделе **Конфиденциальность** также находится раздел **История файлов и корзина**. Ubuntu хранит историю использования файлов. Здесь можно задать продолжительность хранения истории и очистить ее. Также здесь можно очистить корзину и временные файлы, а также задать интервал автоматического удаления, который по умолчанию равен 30 дней.

Раздел **Настройка экранов** позволяет выбрать разрешение экрана монитора, если разрешение по умолчанию выбрано неправильно (рис. 3.15).

Раздел **Комбинации клавиш** позволяет изменить комбинации клавиш, в том числе для переключения раскладки клавиатуры, а в разделе **Дата и время** можно выбрать часовой пояс, установить дату и время. Разделы **Сеть** и **Пользователи** будут рассмотрены в других главах этой книги.

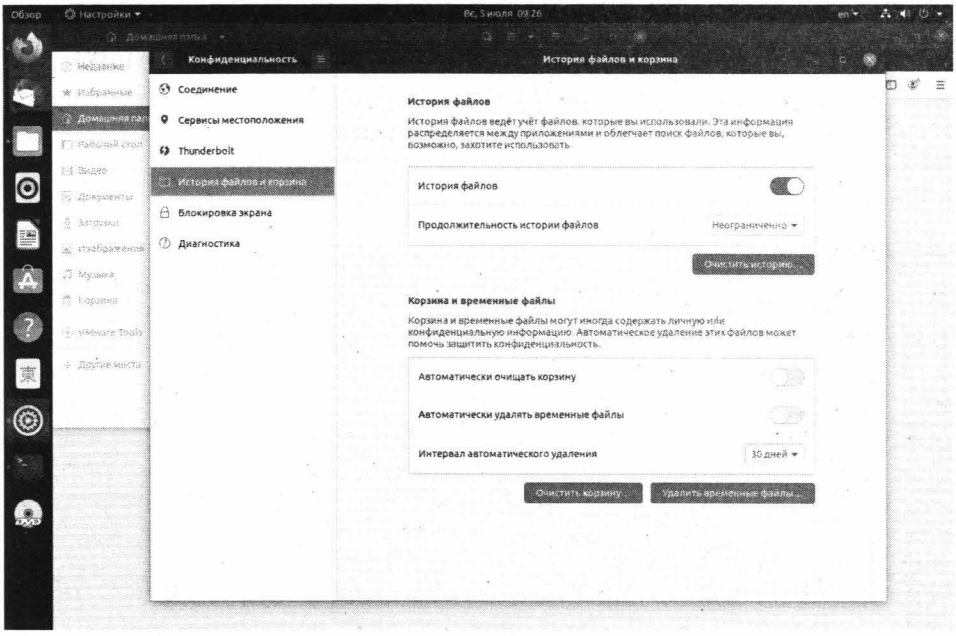


Рис. 3.14. История файлов и корзина

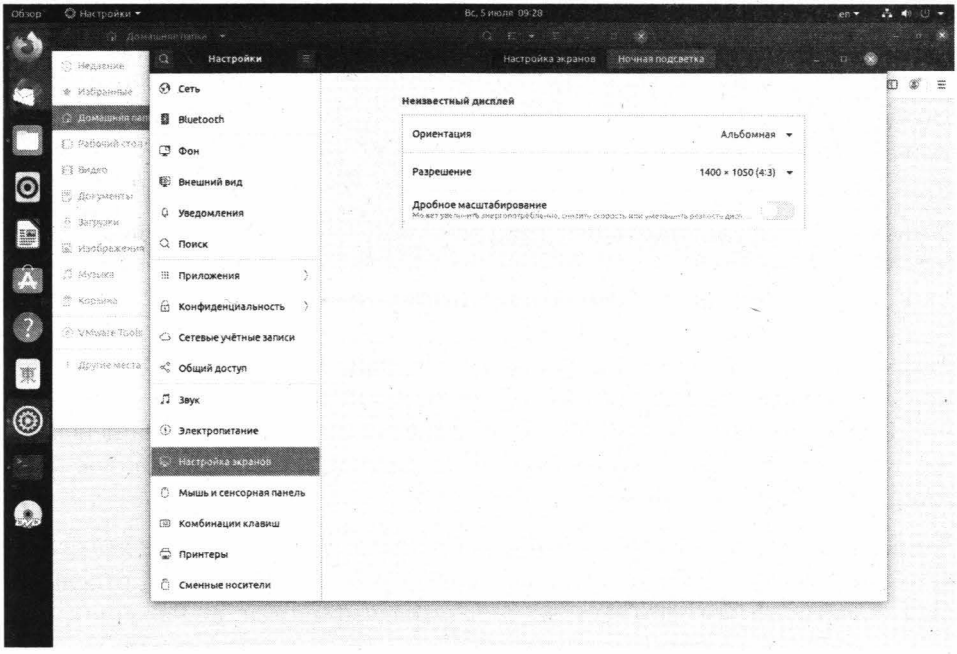


Рис. 3.15. Настройка экранов

3.2.2. Интерфейс Astra Linux

Интерфейс Astra Linux пытались максимально заточить под Windows. Панель задач снизу, меню в стиле Пуск, только открывается окно не кнопкой с логотипом Windows, а кнопкой с лого Astra (рис. 3.16). Бывшие Windows-пользователи оценят такое решение – для них такой интерфейс будет привычнее интерфейса Ubuntu.

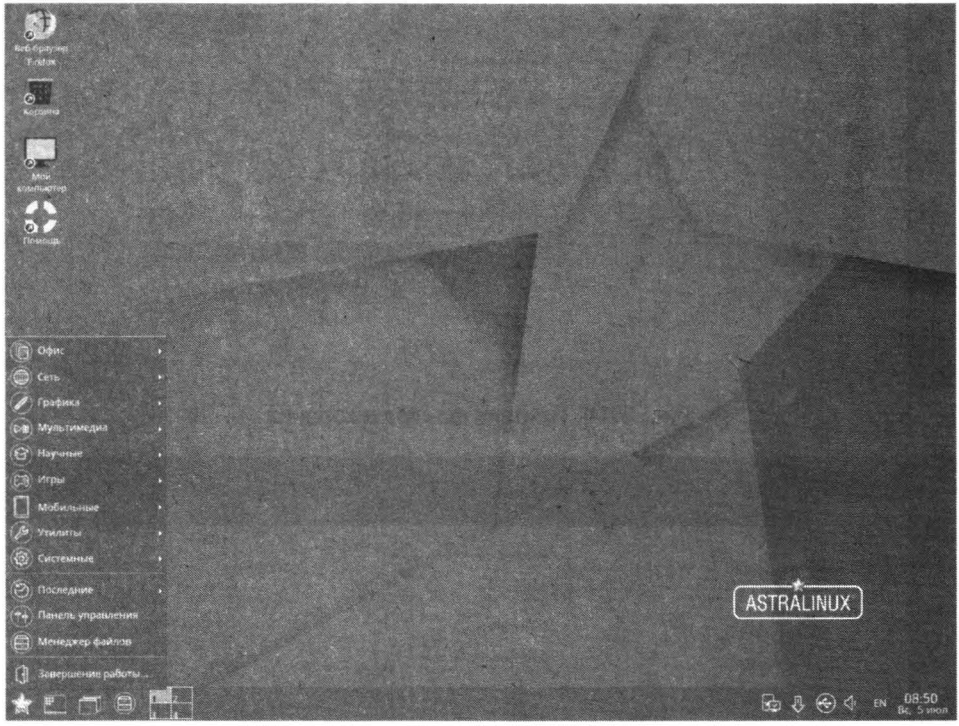


Рис. 3.16. Рабочий стол и главное меню Astra Linux

Рассмотрим содержимое панели задач. **Первая** кнопка открывает главное меню, изображенное на рис. 3.16. **Вторая** кнопка – сворачивает все окна, предоставляя доступ к рабочему столу. **Третья** используется для переключения между окнами – когда вам нужно быстро найти запущенное окно (рис. 3.17).

Четвертая кнопка открывает файловый менеджер (рис. 3.18). Далее следует переключатель рабочих столов. Каждый рабочий стол – это отдельное пространство, на котором вы можете запускать приложения. Рабочие столы позволяют эффективно организовать рабочее пространство, когда запущенных окон много. Для перемещения уже открытого окна на другой рабочий

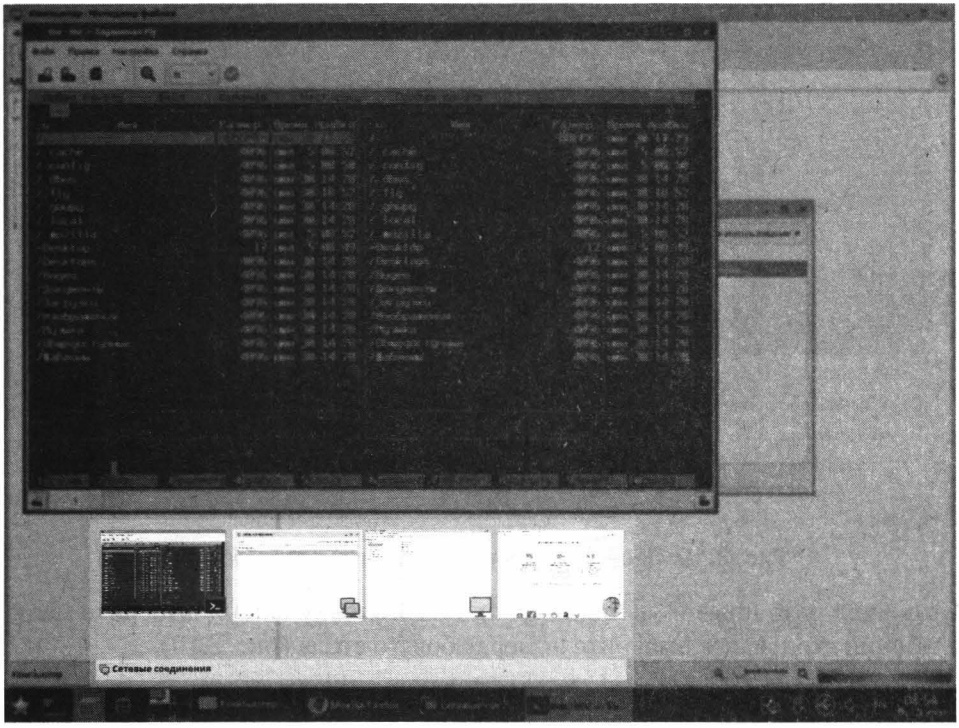


Рис. 3.17. Переключение между окнами в Astra Linux

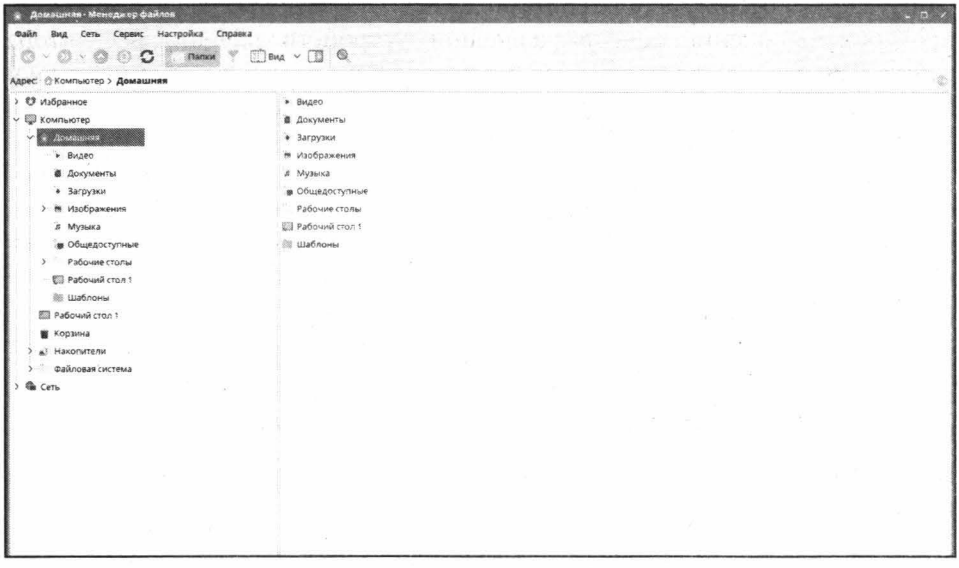


Рис. 3.18. Файловый менеджер Astra Linux

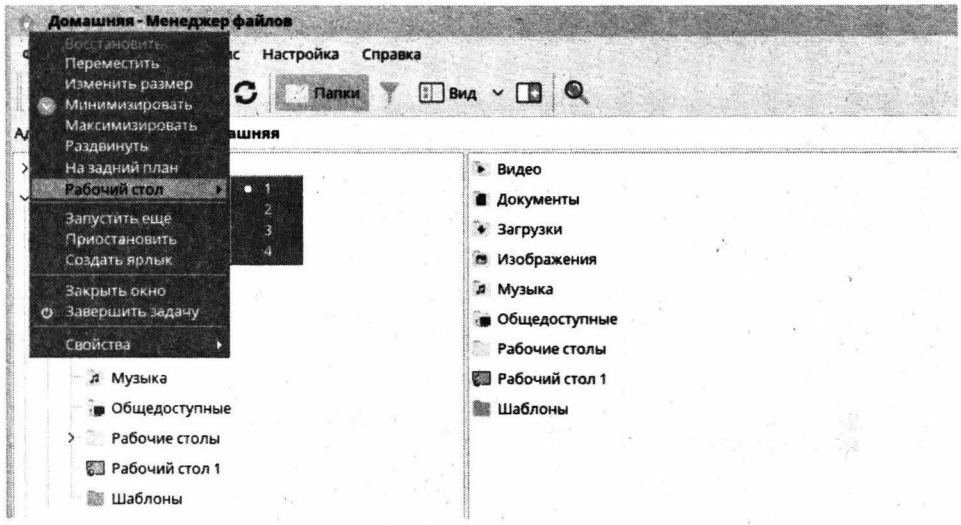


Рис. 3.19. Перемещения окна на другой рабочий стол

стол щелкните правой кнопкой мыши по заголовку окна и выберите меню **Рабочий стол**, далее выберите номер рабочего стола (рис. 3.19).

В Ubuntu также можно организовать работу нескольких рабочих столов, но в последних версиях Ubuntu стараются использовать концепцию одного рабочего стола – минимизация интерфейса.

Снизу справа отображаются системные значки – менеджера сети, средства проверки обновлений, менеджера внешних устройств хранения, регулятора громкости, переключателя раскладки клавиатуры и средства вывода даты и времени. Все эти значки – интерактивные, то есть вы можете щелкнуть по значку менеджера сети, чтобы отключиться от текущего соединения. Если щелкнуть по нему правой кнопкой мыши, появится команда **Изменить соединения** – для управления соединения. Нам понравилась команда **Включить поддержку сети**, которая включает/выключает поддержку сети. Она может понадобиться, если Wi-Fi соединение неожиданно «отвалится» – не перезагружать же компьютер из-за этого?

Щелчок по регулятору громкости открывает микшер громкости, позволяющий регулировать громкость для динамиков, микрофонов и отдельный регулятор предусмотрен для звуков уведомлений (рис. 3.20).

Много полезных утилит вы найдете в программных группах **Утилиты** и **Системные**. Так, здесь вы найдете:

- **Терминал Fly** – средство для ввода команд Linux.

Рис. 3.20. Регулятор громкости в Astra Linux

- **Политика безопасности** – здесь можно добавлять других пользователей Linux и изменять пароль уже имеющихся.
- **Менеджеров пакетов Synaptic** – используется для установки программного устройства.
- **Менеджер устройств** – позволяет просматривать имеющиеся в системе устройства.
- **Редактор разделов Gparted** – графический редактор разметки, который может использоваться для разметки жесткого диска, например, при подключении нового жесткого диска.
- **Запись ISO образа на USB носитель** – название этой утилиты вполне описывает ее назначение.
- **Менеджер файлов MC** – запускает двухпанельный менеджер файлов Midnight Commander, очень популярный у Linux-пользователей. К сожалению, в Ubuntu 20.04 его установить уже нельзя. Последней версией, в репозитарии которой был mc, была 19.04.

Команда **Завершение работы** открывает меню, в котором будут различные варианты завершения работы – выключение, перезагрузка, сон, блокировка, выход, гибернация. Последний режим позволяет сохранить состояние компьютера. При следующей загрузке состояние компьютера будет восстановлено. Режим гибернации похож на режим сна, но поскольку состояние оперативной памяти будет сохранено на жестком диске, вы можете выключить питание компьютера, что никак не повлияет на состояние системы, в отличие от режима сна. Не все компьютеры поддерживают режим гибер-

нации, также для перехода в этот режим нужно, чтобы на жестком диске было достаточно свободного места – ведь нужно сохранить состояние оперативной памяти. Если в вашем компьютере установлено 8 Гб оперативной памяти, то на жестком диске должно быть как минимум 8 Гб свободного пространства для перехода в режим гибернации.

Откройте главное меню и выберите команду **Панель управления**. Откроется панель управления системой (рис. 3.21).

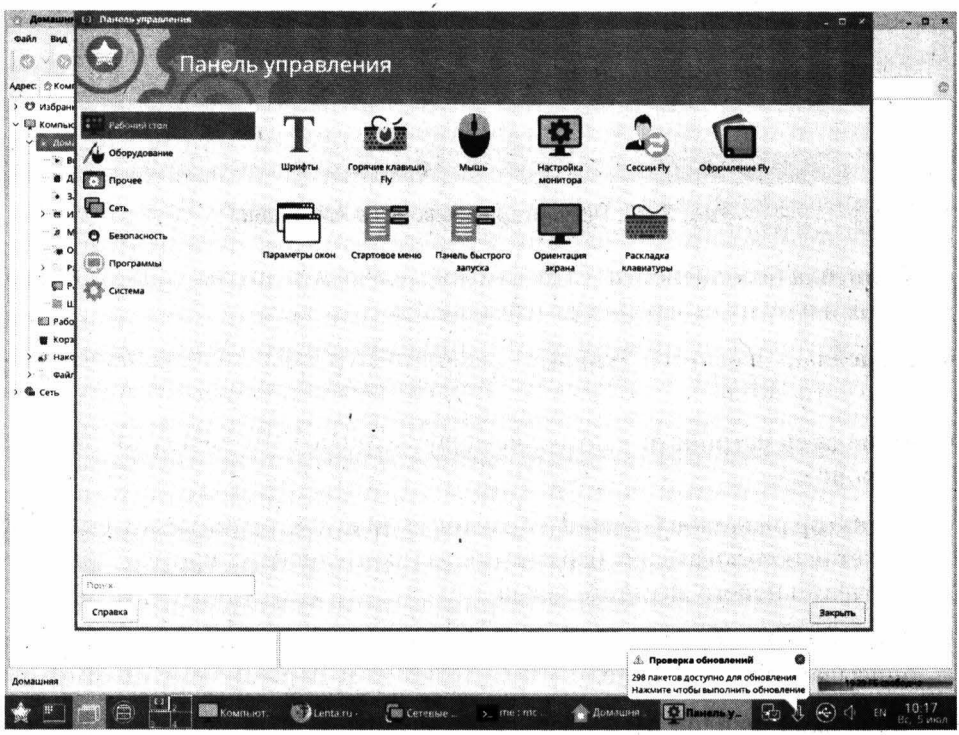


Рис. 3.21. Панель управления Astra Linux

Раздел **Оформление Fly** позволяет изменить оформление рабочего стола – выбрать другие обои, настроить блокировку экрана, выбрать тему оформления, выбрать различные графические эффекты.

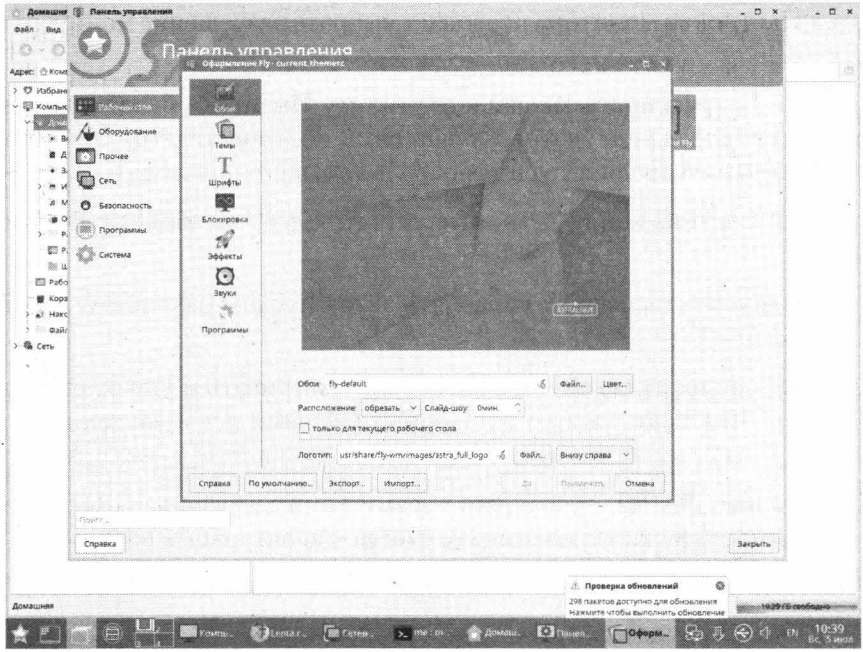


Рис. 3.22. Изменение оформления Astra Linux

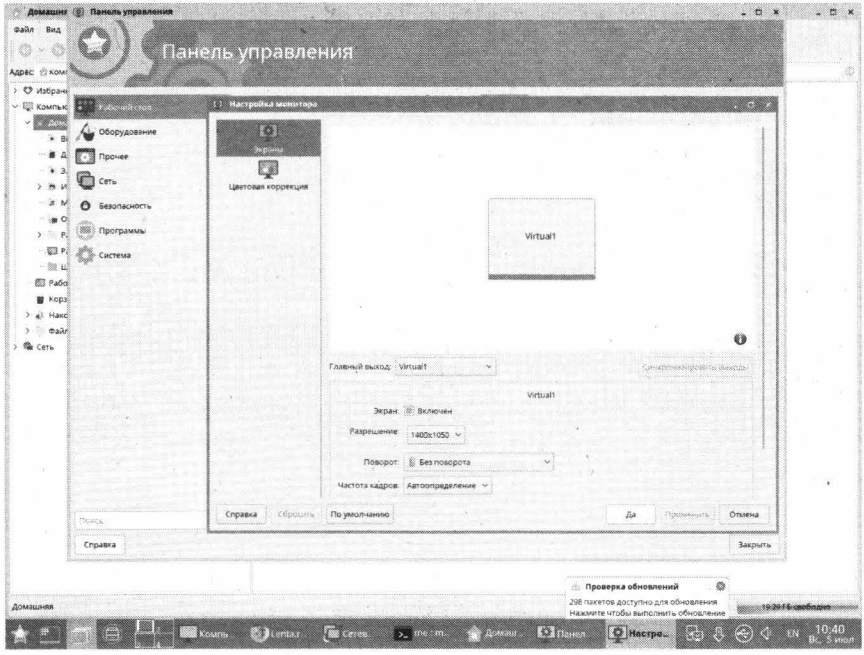


Рис. 3.23. Выбор разрешения монитора

Раздел **Настройка монитора** позволяет выбрать разрешение монитора (рис. 3.23).

В разделе **Безопасность** вы найдете утилиту **Политика безопасности**, позволяющую управлять учетными записями пользователями. Утилита **Изменить пароль** позволяет сменить пароль текущего пользователя.

В разделе **Система** много чего интересного. Здесь вы найдете следующие утилиты:

- **Автозапуск** – позволяет управлять автозапуском различных приложений GNOME;
- **Планировщик заданий** – позволяет просматривать и управлять задачами cron, что очень удобно и подобных решений нет в других дистрибутивах;
- **Загрузчик GRUB2** – управляет параметрами загрузчика GRUB2 без необходимости редактирования его файла конфигурации вручную;
- **Система инициализации** – позволяет управлять системными службами.

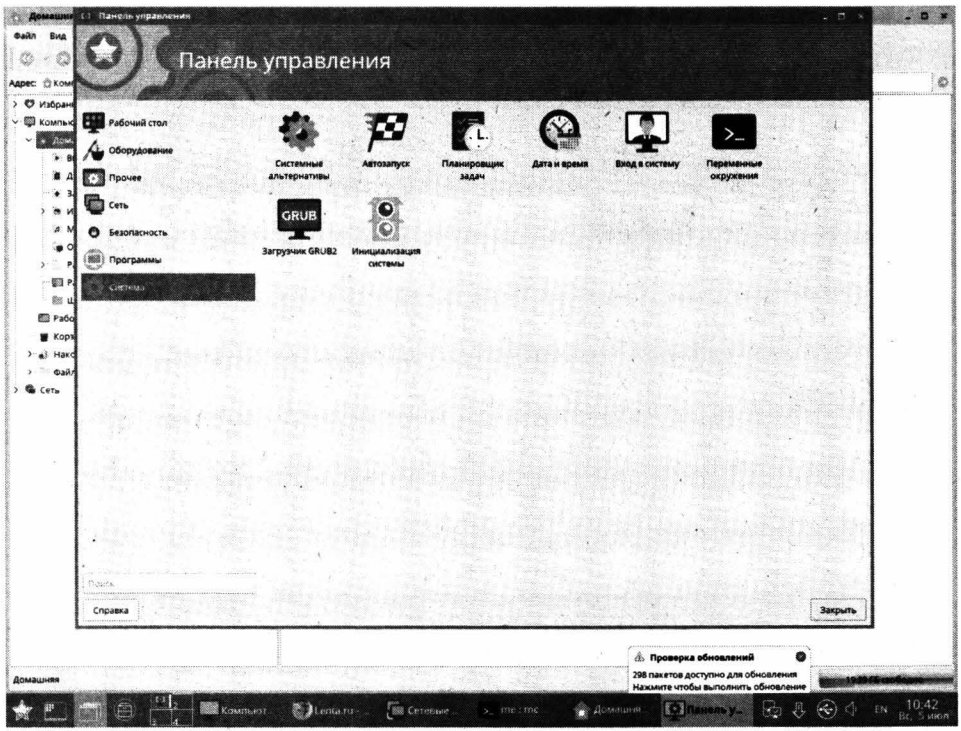


Рис. 3.24. Раздел Система

3.3. Автоматический вход в систему

Если вы не страдаете паранойей, существует возможность настройки автоматического входа в систему. При этом система не будет запрашивать пароль, а будет обеспечивать вход выбранного пользователя в систему сразу после загрузки. Для пользователя домашнего компьютера, который работает в гордом одиночестве – это идеальный вариант. Конечно, если этому пользователю нечего скрывать от других членов семьи. Для настройки автоматического входа в Astra Linux выполните следующие действия:

1. Откройте **Панель управления;**
2. Перейдите в раздел **Система;**
3. Вызовите утилиту **Вход в систему**;
4. Перейдите на вкладку **Дополнительно**, включите параметр **Разрешить автоматический вход в систему**;
5. Выберите пользователя, автоматический вход в систему которого нужно обеспечить (рис. 3.25);
6. Нажмите кнопку **Да**;
7. Перезагрузите систему.

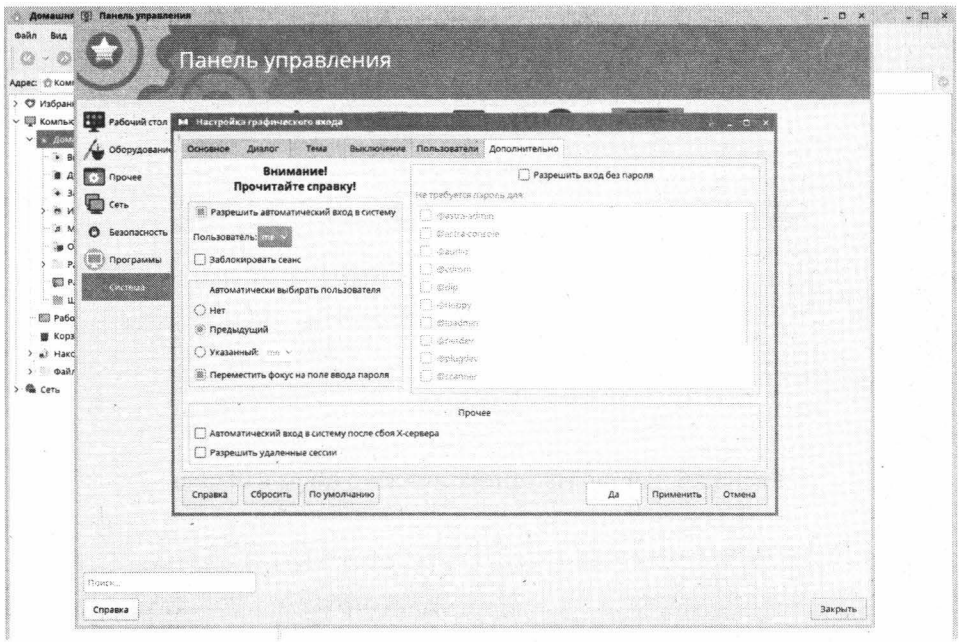


Рис. 3.25. Настройка автоматического входа в Astra Linux

В Ubuntu нужно выполнить следующие операции:

1. Откройте экран **Настройки**;
2. Перейдите в раздел **Пользователи**;
3. Нажмите кнопку **Разблокировать** для разблокирования интерфейса управления учетными записями;
4. Выберите пользователя, автоматический вход которого нужно обеспечить;
5. Включите параметр **Автоматический вход**
6. Закройте окно **Настройки**;
7. Перезагрузите систему.

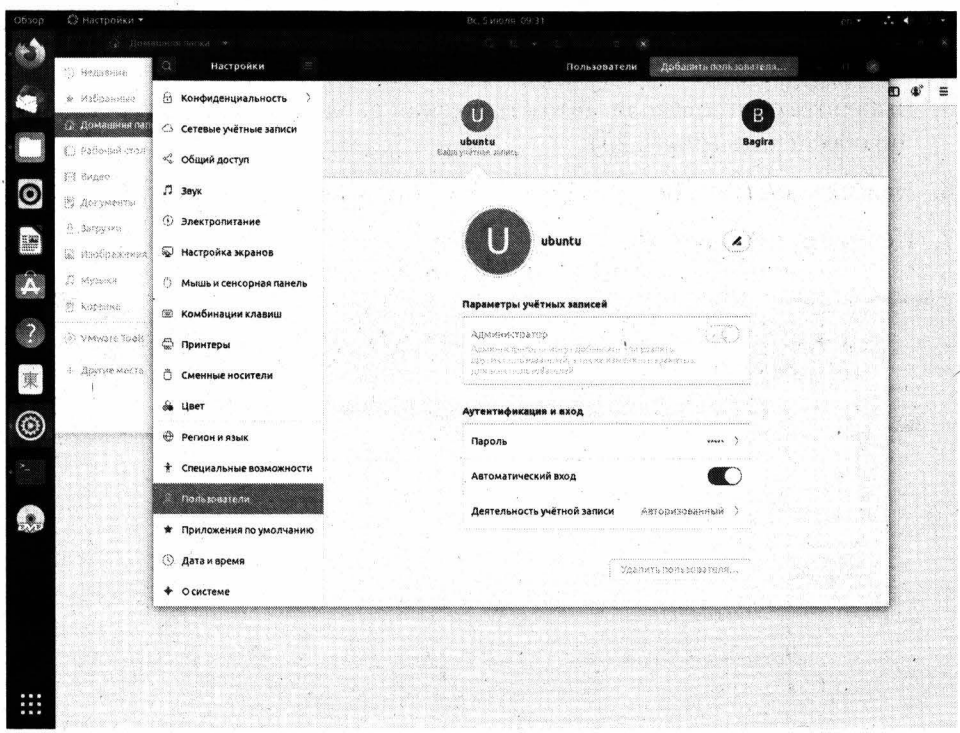


Рис. 3.26. Настройка автоматического входа в Ubuntu

3.4. Завершение работы из консоли

Ранее было рассказано, как завершить работу в графическом режиме. Но Linux поддерживает ряд команд, позволяющих завершить работу системы

из консоли. Данные команды вы можете, как вводить вручную, так и использовать их в сценариях командной оболочки (см. приложение 1).

Для завершения работы в Linux используются следующие команды:

- `poweroff` - завершает работу системы Linux и отключает питание компьютера;
- `halt` - завершает работу системы, но не отключает питание;
- `reboot` - перезагружает систему;
- `shutdown` - обеспечивает более гибкое завершение работы.

При завершении работы (в том числе и при перезагрузке) производится ряд очень важных действий, а именно синхронизация буферов ввода/вывода и размонтирование смонтированных файловых систем. Именно поэтому очень важно правильно завершить работу системы.

Программа `shutdown` может не просто завершить работу системы, а сделать это в указанное время. Причем всем зарегистрированным в системе пользователям на их консоль будет отправлено сообщение (определяется администратором) о необходимости завершения работы или перезагрузки. Формат вызова команды `shutdown` следующий:

```
shutdown [параметры] [время] [сообщение]
```

Пример вызова команды shutdown:

```
sudo shutdown -r now Bye!
sudo shutdown -h 19:00
```

В первом случае система будет перезагружена (-r) моментально (время - now), а всем пользователям будет отправлено сообщение «Bye!». Сообщение не обязательно и вы можете его не указывать, что и продемонстрировано на примере второй команды. Во втором случае работа системы будет завершена (-h) в 19:00. Пользователи получат стандартное сообщение, что работа системы будет завершена.

> Примечание. Команды завершения работы требуют полномочий root, поэтому вводит их нужно через команду sudo.

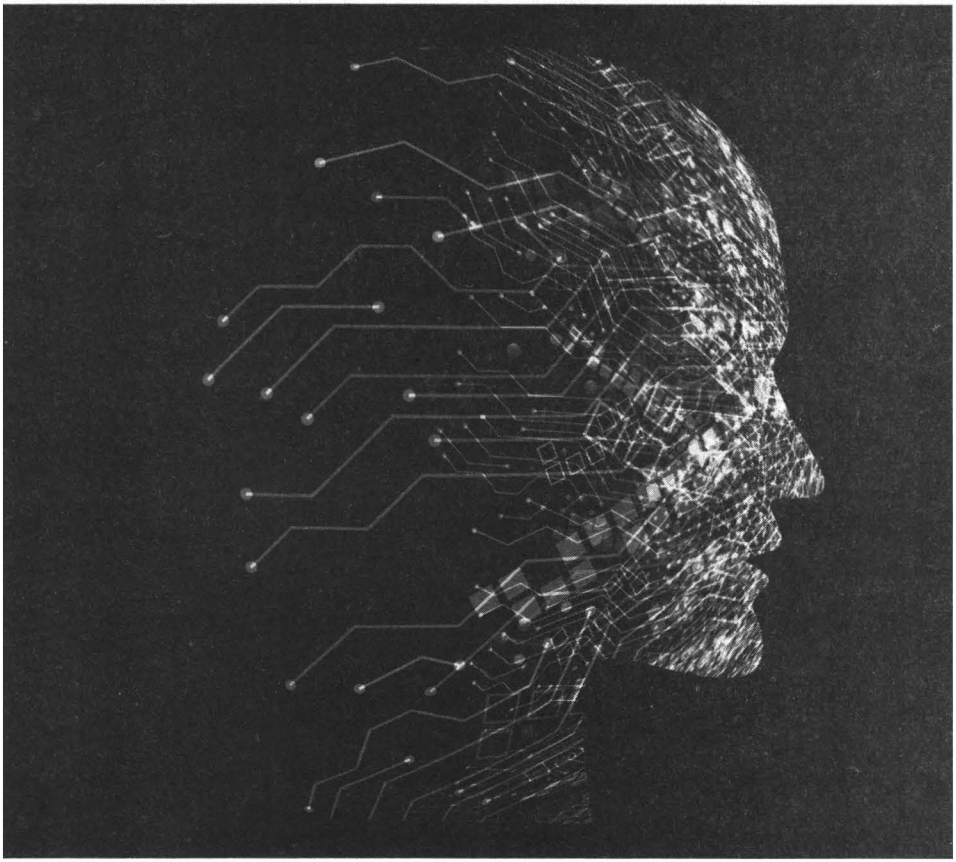
Дополнительные параметры shutdown приведены в таблице 3.1.

Таблица 3.1. Параметры команды shutdown

Параметр	Описание
-H (или --halt)	Завершает работу системы, питание не отключается
-P (или --power-off)	Завершает работу системы, питание отключается
-r (или --reboot)	Перезагружает компьютер
-h	Аналогично --poweroff, то есть завершение работы с отключением питания
-k	Работа системы не завершается, просто каждый пользователь получит указанное сообщение.
--no-wall	При завершении работы системы (в том числе перезагрузке) пользователям не будут выводиться
-c	Отменяет отложенное завершение работы (если вы передумали завершать работу или перезагружать систему, но при условии, что процесс завершения работы еще не начат)

Глава 4.

Сразу после установки



Есть некоторые вещи, которые нужно настроить сразу после установки системы. В прошлой главе мы разобрались, как войти в систему, как завершить ее работу, как вызвать средства конфигурации системы. В этой главе мы изменим под себя некоторые параметры системы и установим некоторое важное программное обеспечение. Подробно об установке программ мы поговорим в главе 10.

4.1. Проверяем и устанавливаем обновления

Очень важно поддерживать систему в актуальном состоянии. Если вы не выбрали установку обновлений при установке системы или не вы устанавливали систему, самое время проверить наличие обновлений.

В Astra Linux щелкните по значку средства обновлений, откроется окно со списком пакетов, требующих обновления. Нажмите **Да**, чтобы произвести обновление системы.

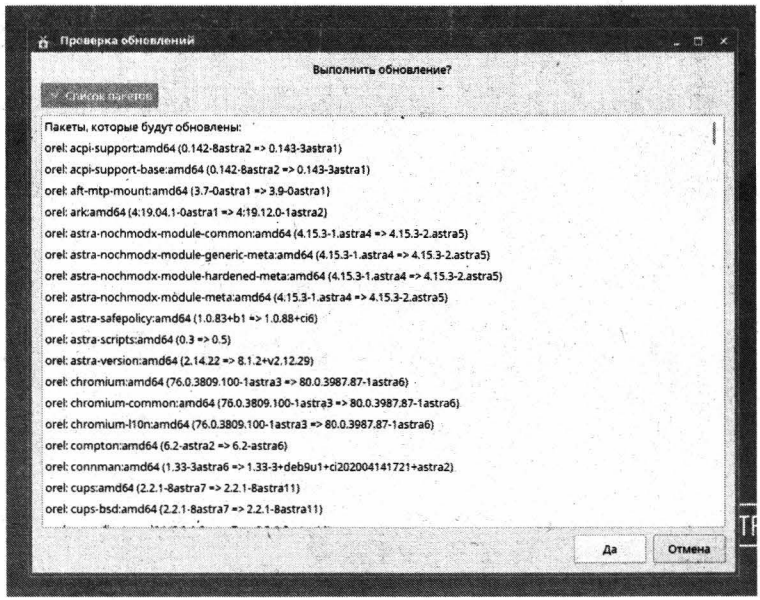


Рис. 4.1. Обновление системы в Astra Linux

В Ubuntu нажмите Alt + F2, в появившемся окне введите команду **update-manager**. Откроется окно менеджера обновлений (рис. 4.2). Менеджер произведен поиск обновлений и, если таковые будут найдены, предложит их установить. Обычно обновления в Ubuntu устанавливаются автоматически, поэтому при вызове менеджера вы можете часто увидеть картину, изображенную на рис. 4.2. Она означает, что обновления уже установлены и для их применения нужно перезагрузить компьютер.

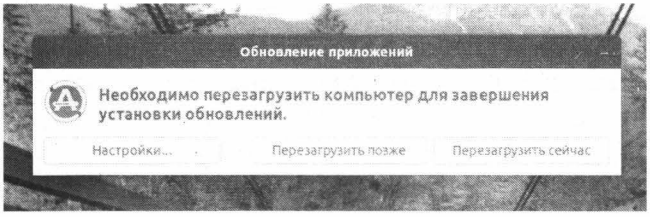


Рис. 4.2. Менеджер обновлений в Ubuntu

Нажмите кнопку **Настройки** (рис. 4.3), чтобы настроить периодичность проверки обновлений. Если вы не хотите, чтобы система автоматически проверяла наличие обновлений, из списка **Автоматически проверять наличие обновлений** выберите **Никогда**.

Рис. 4.3. Параметры обновлений по умолчанию

Если вы предпочитаете командную строку, то откройте терминал и введите команду (для полного обновления системы):

```
sudo apt update && sudo apt full-upgrade
```

4.2. Настройке Livepatch (только для Ubuntu)

Livepatch (или Canonical Livepatch Service) позволяет пользователям Ubuntu применять критические исправления ядра без перезагрузки. Livepatch также помогает поддерживать безопасность вашей системы, применяя обновления безопасности без перезагрузки системы. Сервис бесплатный (до 3 компьютеров) и все, что вам нужно для его активации – настроить учетную запись Ubuntu.

Откройте окно **Программы и обновления** (команда update-manager, как было показано ранее) и перейдите на вкладку Livepatch. Нажмите **Войти** для входа в учетную запись Ubuntu или ее создания, а после того, как вход будет выполнен, включите Livepatch, включив единственный переключатель на этой странице.

Рис. 4.4. Активация Livepatch

4.3. Отключаем уведомления об ошибках

Для отключения оповещения об ошибках, откройте экран **Настройки**, перейдите в раздел **Конфиденциальность**, затем – в раздел **Диагностика**, для параметра **Отправлять отчеты об ошибках в Canonical** выберите значение **Никогда**.

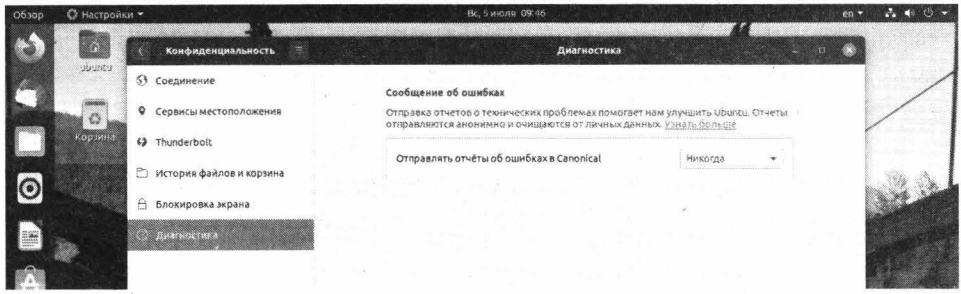


Рис. 4.5. Отключаем уведомления об ошибках

4.4. Настраиваем почтовый клиент

На панели задач Ubuntu есть кнопка вызова почтового клиента Thunderbird. В Astra Linux команда вызова почтового клиента находится в программной группе **Сеть**. Запустите почтовый клиент и просто введите ваш e-mail и пароль от почтового ящика. Почтовый клиент Thunderbird умный и сам установит остальные параметры почтового сервиса (SMTP, IMAP, порты и т.д.).

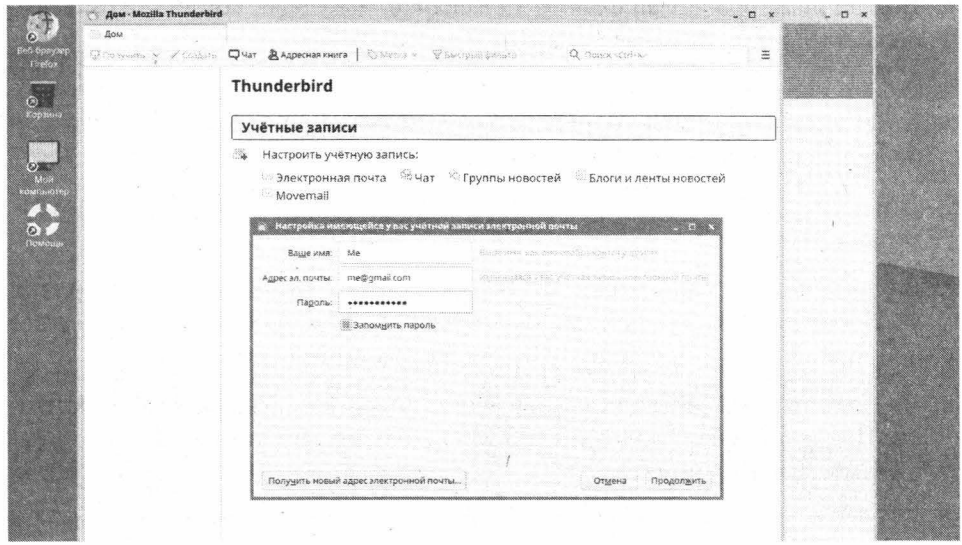


Рис. 4.6. Почтовый агент Thunderbird в Astra Linux

> Примечание. Если у вас почтовый ящик на Gmail, нужно в настройках аккаунта включить использование небезопасных приложений – таковыми считаются все не Google приложения, иначе вы не сможете подключиться к своему почтовому ящику!

Некоторые пользователи предпочитают использовать веб-интерфейс в браузере, а не почтовую программу. В этом случае можно смело удалить кнопку вызова почтового клиента, чтобы она даром не занимала пространство на панели задач. Для этого щелкните по ней правой кнопкой мыши и выберите команду **Удалить из избранного**. Вы освободите пространство для одной полезной для вас кнопки!

4.5. Установите ваш любимый браузер

По умолчанию в Linux используется браузер Firefox. Это очень хороший браузер, но он нравится далеко не всем. Установить Chrome можно путем загрузки его пакета с официального сайта и его установки. Рассмотрим весь процесс подробнее.

Откройте Firefox и перейдите по ссылке https://www.google.com/intl/ru/chrome.

1. Нажмите кнопку **Скачать Chrome**.
2. Выберите DEB-пакет.
3. В появившемся окне выберите **Сохранить файл.**
4. В окне браузера, когда файл будет скачан, выберите команду **Показать все загрузки**.
5. В окне менеджера загрузок нажмите значок папки напротив загруженного DEB-файла, чтобы перейти в папку **Домашняя папка/Загрузки** или откройте файловый менеджер и перейдите в эту папку самостоятельно .
6. Щелкните правой кнопкой мыши и выберите команду **Открыть в терминале**.
7. Введите команду sudo dpkg –i *.deb (убедитесь, что в каталоге **Загрузки** у вас нет других deb-файлов, поскольку эта команда устанавливает все deb-файлы из текущей папки, что может быть нежелательно).
8. Введите пароль своего пользователя и дождитесь завершения установки. Пакет довольно большой (65 Мб), поэтому его установка может занять несколько минут .
9. Откройте экран **Приложения** (кнопка в самом низу на панели задач) и щелкните правой кнопкой на Chrome.

10. Выберите команду **Добавить в избранное**, чтобы добавить кнопку запуска нового браузера на панель задач

11. Осталось дело за малым – использовать Chrome.

4.6. Установка проигрывателя VLC

VLC – культовый медиа-проигрыватель, поддерживающий множество самых разных медиа-форматов. В Ubuntu для его установки нужно ввести команду:

```
$ sudo snap install vlc
```

Теперь проигрыватель распространяется в виде снапа, а не пакета, что упрощает его установку.

Пользователям Astra Linux повезло больше: VLC уже установлен по умолчанию, поэтому все, что нужно – запустить его из программной группы **Мультимедиа**.

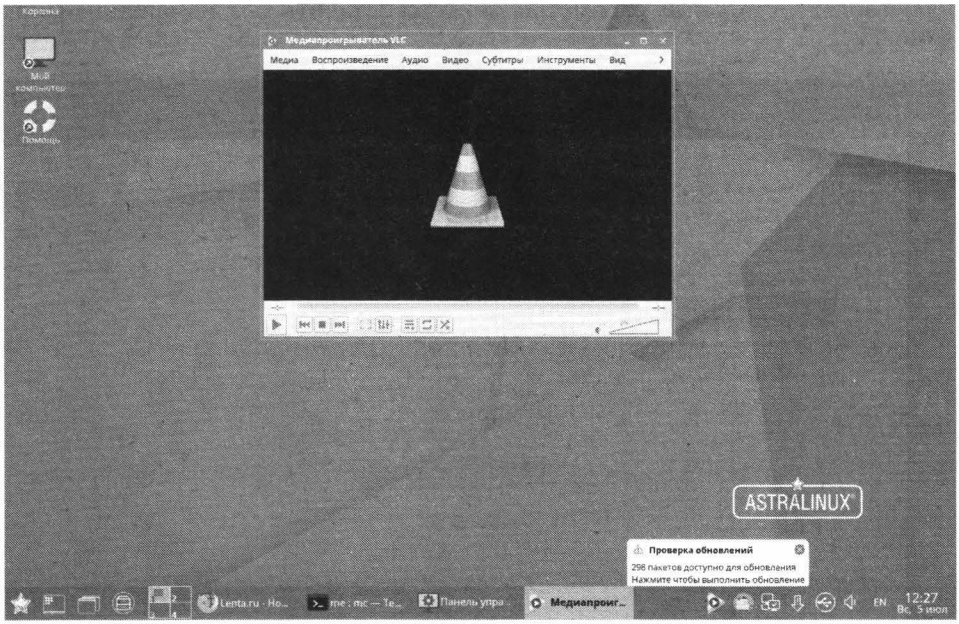


Рис. 4.7. Проигрыватель VLC

4.7. Установка кодеков

Разработчики Ubuntu включают в состав дистрибутива только бесплатное ПО с открытым исходным кодом. К таковому не относятся кодеки – специ-

альное ПО для кодирования/декодирования мультимедиа-форматов. Кодеки бесплатные, но их разработчики не хотят публиковать исходные коды, поэтому по умолчанию кодеки не включены в состав Ubuntu. Для воспроизведения распространенных аудио- и видеофайлов, таких как MP3, AVI, MPEG4 нужно установить кодеки вручную.

Чтобы установить их, вам нужно установить метапакет ubuntu-limited-extras, выполнив следующую команду:

```
$ sudo apt install ubuntu-restricted-extras
```

4.8. Включение ночного режима

Чтобы глаза меньше уставали при работе ночью, рекомендуется включить ночную подсветку, которая сделает цвета более теплыми. Для этого откройте экран **Настройки**, перейдите в раздел **Настройка экранов**, перейдите на вкладку **Ночная подсветка** и активируйте единственный доступный переключатель. Остальные параметры можете оставить без изменений (рис. 4.8).

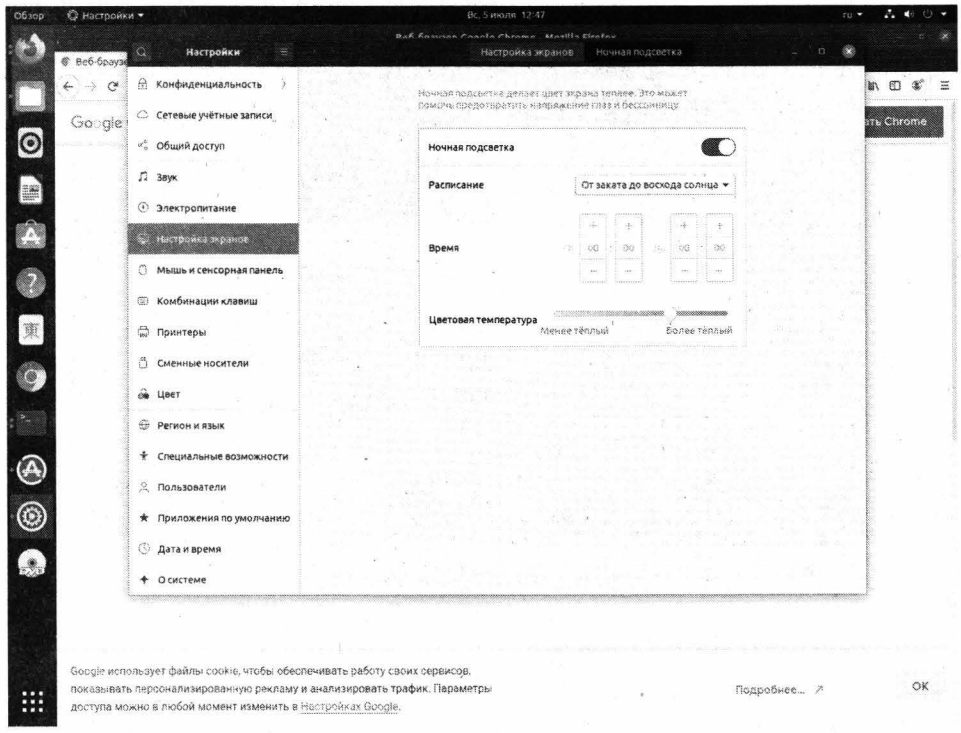


Рис. 4.8. Активация ночного режима

В Astra Linux, к сожалению, подобного режима нет. В нем есть возможность цветовой коррекции, и вы сами можете установить более теплые цвета, но автоматической смены оттенков там нет и вам придется менять цветовые настройки дважды в сутки – днем и ночью, что очень неудобно.

4.9. Установка wine для запуска Windows-приложений

Подробно о запуске Windows-приложений в Linux мы поговорим в главе 12, а пока установите Wine – средство, обеспечивающее запуск Windows-приложений. Для его установки введите команду:

```
$ sudo apt install wine winetricks
```

4.10. Установка дополнительных архиваторов

Поддержка дополнительных форматов архивов никогда не бывает лишней. Для установки дополнительных архиваторов введите команду:

```
$ sudo apt install rar unrar p7zip-full p7zip-rar
```

4.11. Попробуйте другие графические окружения

Ubuntu поставляется только с рабочим столом Gnome, но вы можете установить другие графические окружения и выбрать то, которое больше нравится вам. Например, следующая команда устанавливает графическую среду Cinnanmon (рис. 4.9):

```
$ sudo apt install cinnamon-desktop-environment
```

А эти команды устанавливают графическую среду MATE:

```
$ sudo apt install tasksel
$ sudo tasksel install ubuntu-mate-desktop
```

Какую из них использовать, дело вкуса и здесь каждый решает сам.

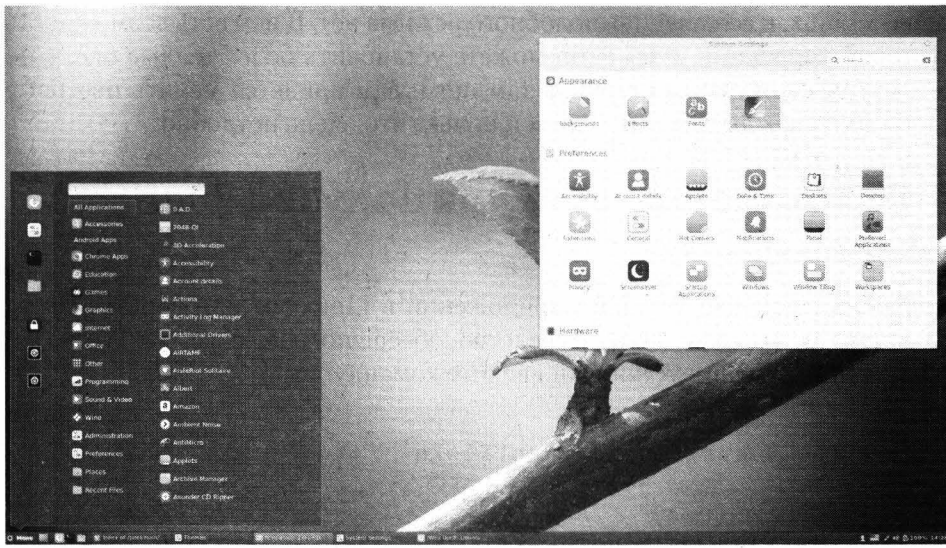


Рис. 4.9. Графическая среда Cinnanmon

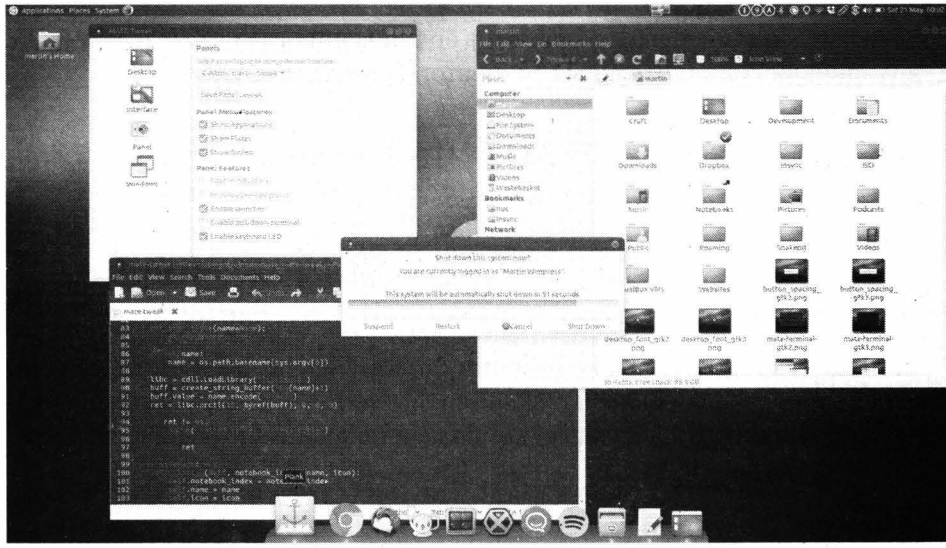


Рис. 4.10. Графическая среда MATE

4.12. Установите полезные утилиты

Установите двухпанельный файловый менеджер Midnight Commander (работает только в консоли) и менеджер пакетов Synaptic, который облегчит поиск нужного пакета при установке программного обеспечения:

```
sudo apt install mc
sudo apt install synaptic
```

4.13. Тонкая настройка GNOME. Установка темы оформления в стиле macOS

Множество настроек графической среды GNOME скрыто от глаз пользователя. Для более тонкой настройки GNOME вы можете использовать утилиту Gnome Tweaks, позволяющую легко кастомизировать ваш рабочий стол. Для ее установки введите команды:

```
$ sudo add-apt-repository universe
$ sudo apt install gnome-tweak-tool
```

Первая команда включает репозитарий universe, в котором находится нужный нам пакет. Вполне возможно, что он уже включен у вас, но лучше убедиться в этом явно. Вторая – устанавливает сам пакет.

Далее запустите средство командой или выберите из меню команду **Дополнительные настройки GNOME**:

```
$ gnome-tweaks
```

Данная утилита – настоящая находка для любителей кастомизации (рис. 4.11). Кстати, только с ее помощью можно изменить тему оформления в

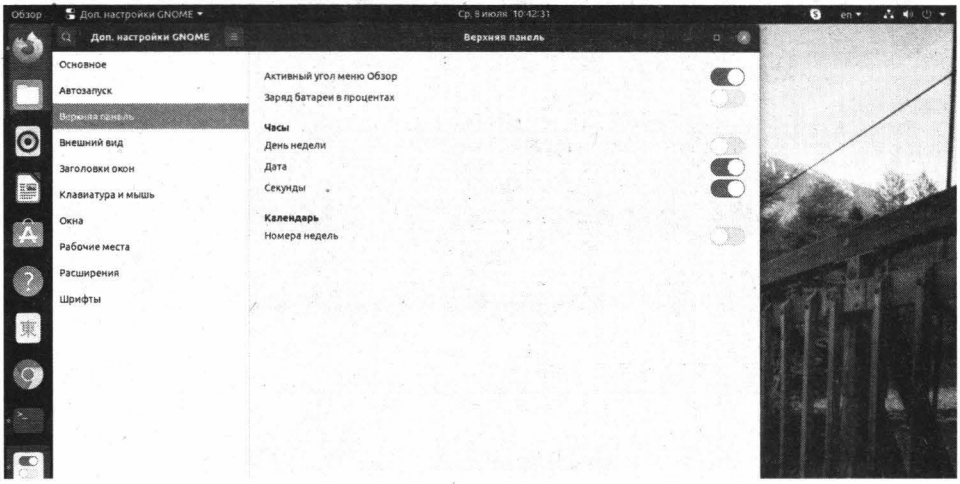


Рис. 4.11. Приложение Gnome Tweaks

Ubuntu на любую другую. Перейдите в раздел **Внешний вид** и выберите другую тему оформления, как показано на рис. 4.12. На рис. 4.13 показано, что тема изменена. Далее будет показан пример наиболее популярного «тюнинга» Ubuntu – установка темы оформления в стиле macOS.

Рис. 4.12. Изменение темы оформления

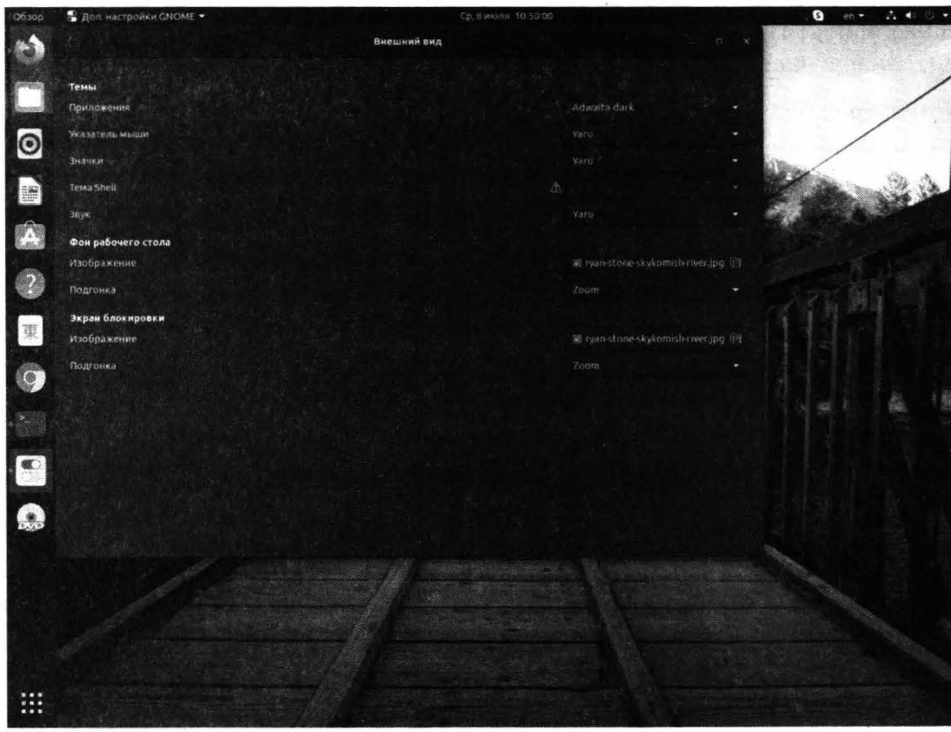


Рис. 4.13. Тема оформления изменена

Установить новую тему достаточно просто. Скачайте архив с темой. Много тем оформления доступно на сайте https://www.gnome-look.org/, например, по адресу https://www.gnome-look.org/p/1275087/ доступна тема в стиле macOS. Перед установкой этой темы нужно установить два пакета:

```
$ sudo apt install gtk2-engines-murrine gtk2-engines-pixbuf
```

Далее нужно скачать архив с темой и распаковать его в каталог .themes:

```
$ tar xf Mojave-dark.tar.xz
$ mkdir ~/.themes
$ mv Mojave-dark ~/.themes/
```

Затем откройте Gnome Tweaks и в качестве темы приложений выберите Mojave-dark. Закройте Gnome Tweaks.

Следующий шаг – скачать значки в стиле macOS. Они доступны по адресу https://www.gnome-look.org/p/1210856/. Аналогично, значки нужно распаковать:

```
$ tar xf Mojave-CT-Night-Mode.tar.xz
$ mkdir ~/.icons
$ mv Mojave-CT-Night-Mode ~/.icons/
```

После этого опять запустите Gnome Tweaks и в качестве темы значков выберите Mojave-CT-Night-Mode. Наконец, нужно установить тему для курсоров мыши. Скачайте архив по адресу https://www.gnome-look.org/p/1148748/ и распакуйте архив в соответствующий каталог:

```
$ unzip -qq macOS\ Cursor\ Set.zip
$ mv macOS\ Cursor\ Set ~/.icons/
```

Опять запустите Gnome Tweaks и выберите MacOS Cursor Set в качестве темы курсоров.

На рис. 4.13 показан процесс распаковки необходимых архивов, а на рис. 4.14 – настройки, сделанные в Gnome Tweaks. У вас должно получиться так, как показано на рис. 4.14. Обратите внимание, как изменились значки в заголовках окон и значки приложений на панели задач.

Следующий шаг (по желанию) – скачать и установить в качестве фонового следующее изображение: **https://www.reddit.com/r/wallpapers/comments/e4fz6s/a_more_purpleish_version_of_the_mac_os_mojave/**

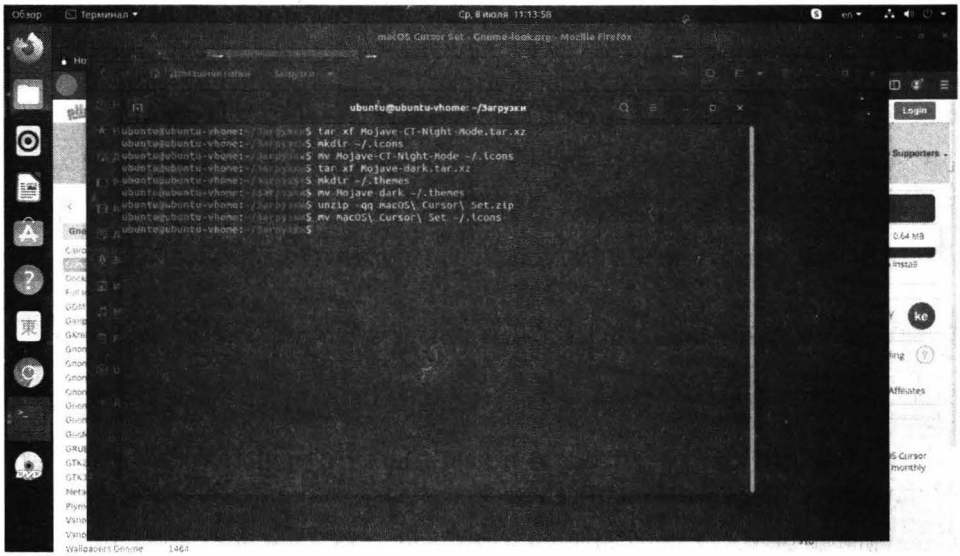


Рис. 4.14. Распаковка ресурсов

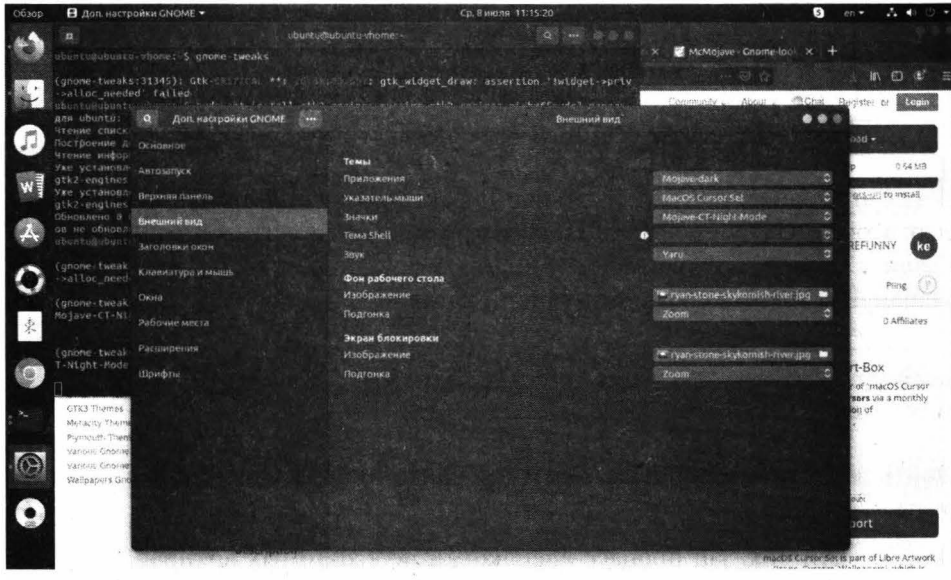


Рис. 4.15. Настройки в Gnome Tweaks

Щелкните правой кнопкой по файлу изображения в папке **Загрузки** и выберите команду **Установить как фон**.

Вишенка на торте – панель задач в стиле macOS. Установить ее можно командой: `$ sudo apt install plank`

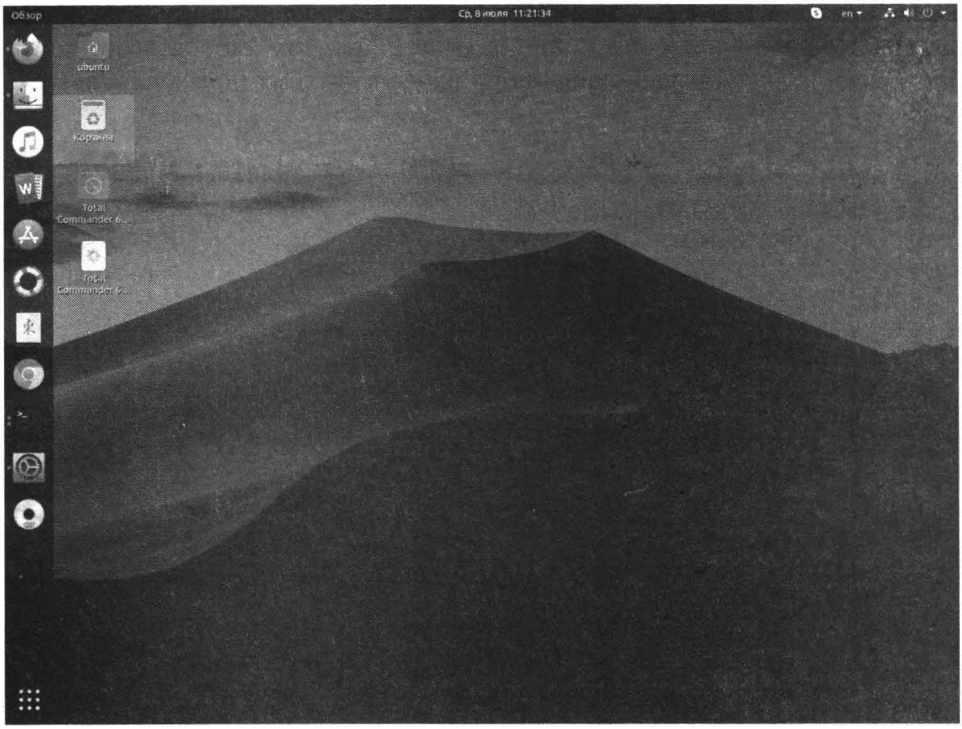


Рис. 4.16. Как будет выглядеть ваша Ubuntu после установки фонового изображения

Запустите новую панель задач:

```
plank
```

В нижней части экрана вы увидите ту самую панель (рис. 4.17).

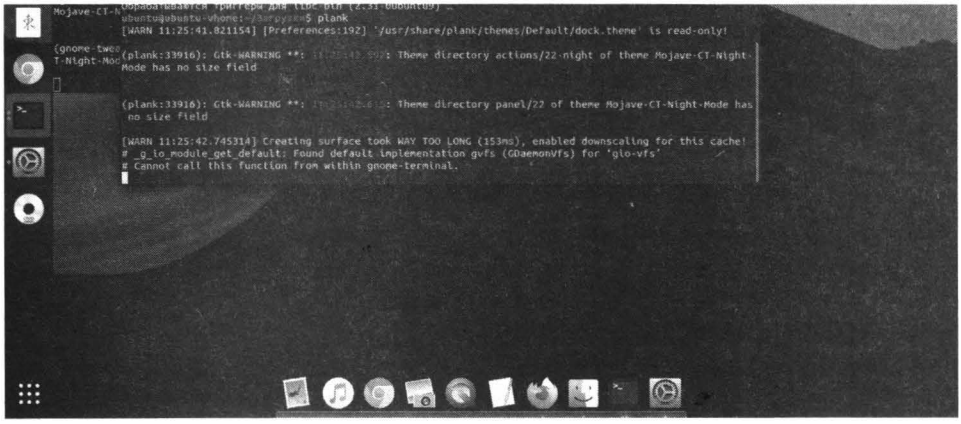


Рис. 4.17. Панель задач в стиле macOS

Откройте Gnome Tweaks и перейдите в раздел **Автозапуск**. Нажмите кнопку + и добавьте в автозапуск приложение Plank. Так мы обеспечим автоматический запуск нашей панели при входе в систему пользователя (рис. 4.18).

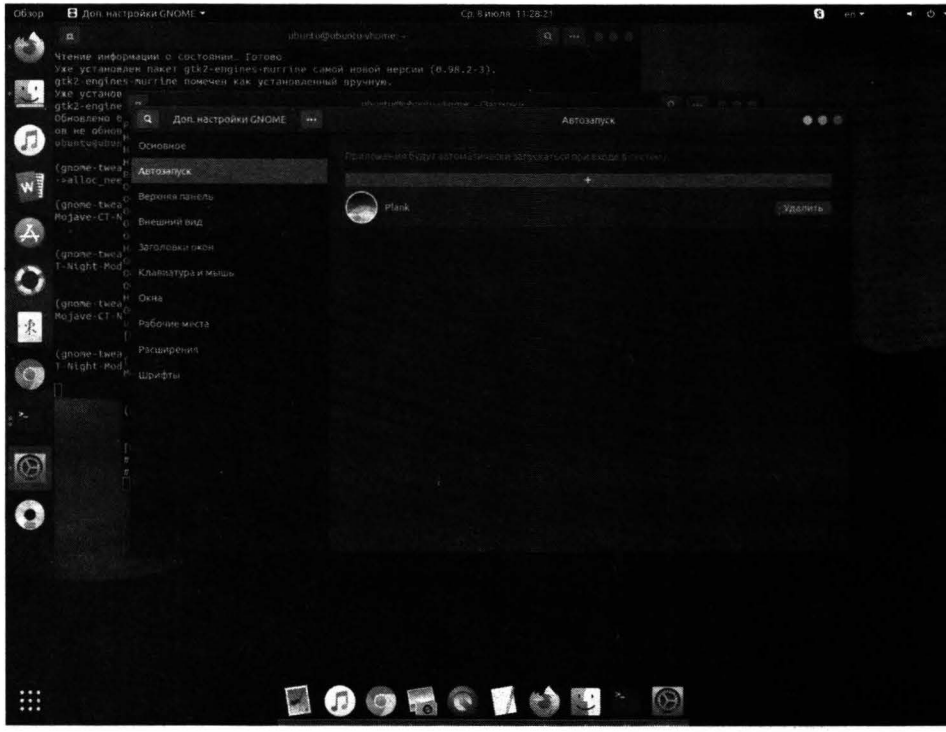


Рис. 4.18. Автоматический запуск панели задач Plank

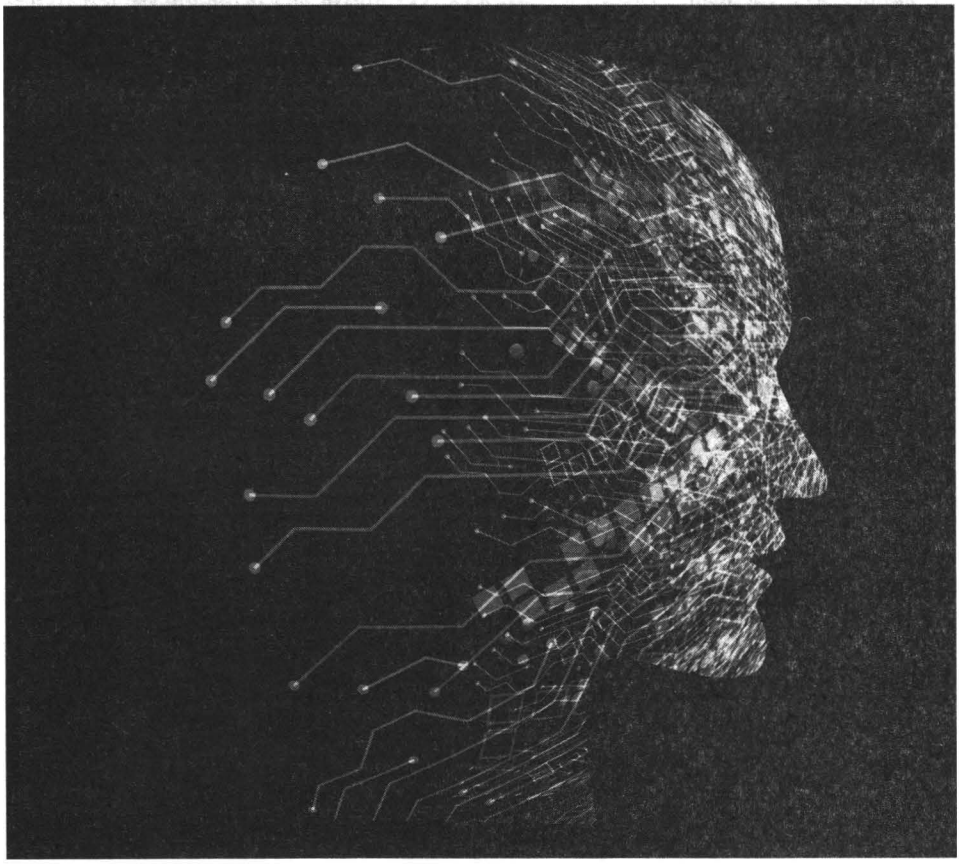
Осталось удалить стандартную панель задач. Для этого введите команду:

```
$ sudo apt remove gnome-shell-extension-ubuntu-dock
```

После этой команды нужно выйти из системы и снова в нее войти. В результате у вас должна получиться «почти» macOS.

Глава 5.

Основы командной строки



Командная строка – неотъемлемая часть Linux. Посредством командной строки осуществляется выполнение команд Linux. Эффективно сможет работать с Linux только тот, кто освоил принципы работы в командной строке.

5.1. Ввод команд

Ввод команд осуществляется в приложении **Терминал**. Это эмулятор консоли Linux, позволяющий вводить команды. Можно переключиться из графического режима в консоль (нажав Ctrl + Alt + F1) и вводить команды непосредственно в консоли Linux. Но для большинства пользователей будет удобнее работа с эмулятором терминала в графическом режиме. Приложение Терминал изображено на рис. 5.1.

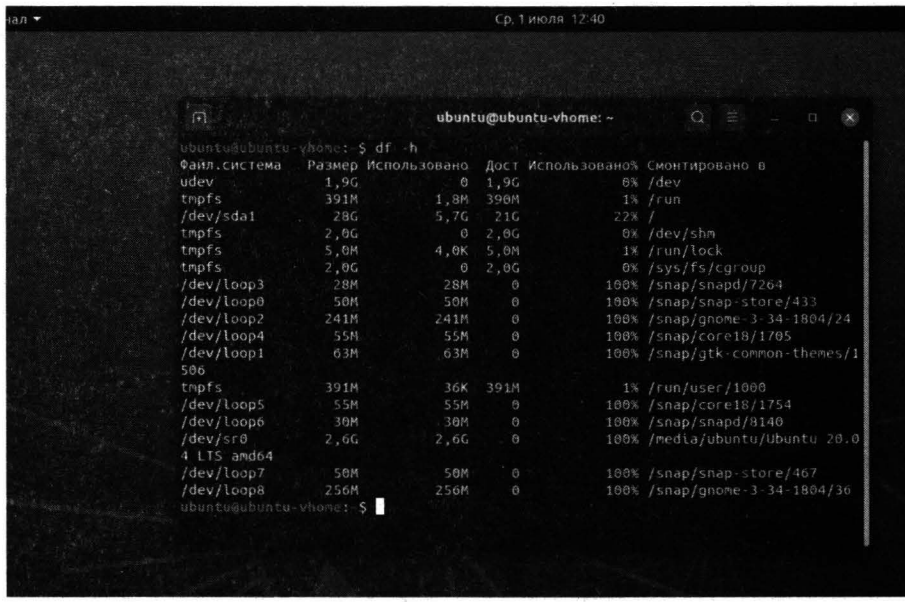


Рис. 5.1. Приложение "Терминал" (Ubuntu)

Для ввода команды нужно ввести ее в приглашение командной строки и нажать **Enter**. После чего вы увидите результат выполнения команды. Обычно приглашение командной строки имеет вид:

```
пользователь@компьютер:текущий_каталог<$|#>
```

Посмотрите на рис. 5.2. В первом случае наш пользователь называется ubuntu, имя компьютера ubuntu-vhome, каталог ~ (так сокращенно обозначается домашний каталог пользователя), далее следует символ **$**. Символ **$** обозначает, что вводимая команда будет выполняться с привилегиями обычного пользователя.

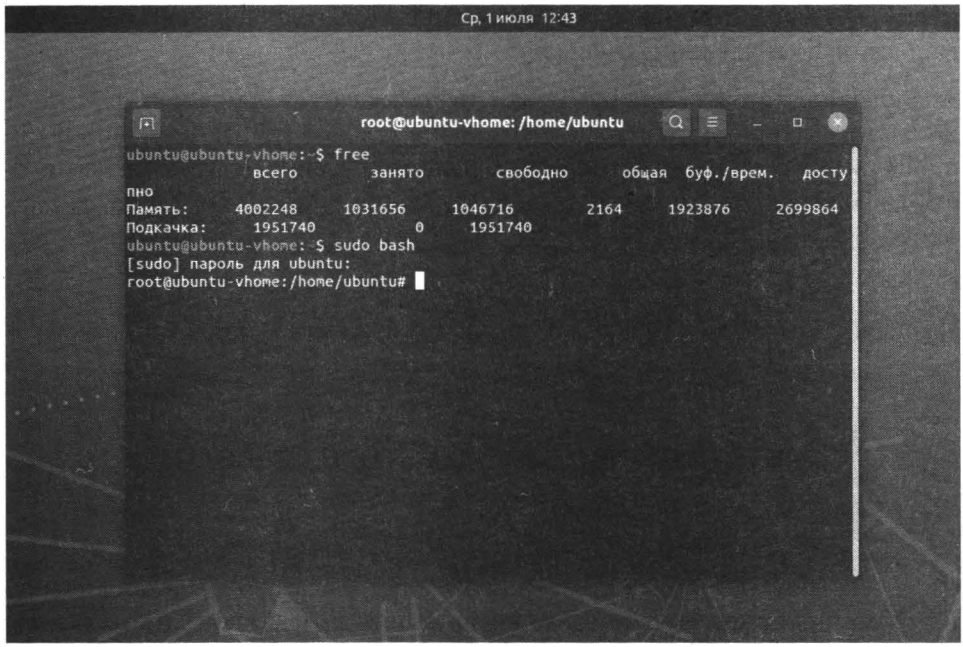


Рис. 5.2. Разные формы приглашения командной строки

Далее мы выполняем команду sudo bash. Команда sudo запрашивает привилегии суперпользователя root для командной оболочки bash. По сути, после этого мы получим командную строку с привилегиями root. Все вводимые данные команды будут выполняться с максимальными правами.

Посмотрим, как поменялось приглашение командной строки. Мы видим, что пользователь уже не ubuntu, а root и что каталог выводится как /home/ubuntu – это домашний каталог пользователя ubuntu, в котором мы находимся. Символ ~ не выводится, поскольку домашний каталог пользователя root называется /root. Как только мы перейдем в этот каталог, то получим ~ вместо имени каталога. Последний символ # обозначает, что вводимая команда будет выполняться с максимальными правами. С максимальными правами нужно быть осторожнее и стараться, как можно реже использовать данный режим во избежание нанесения системе вреда.

5.2. Автодополнение командной строки

Linux содержит множество команд. Если вы забыли точное название команды или просто хотите ускорить ее ввод, вы можете использовать автодополнение командной строки. Для этого введите первые буквы названия команды и нажмите **Tab**. Далее система или автоматически дополнит команду или выведет список доступных вариантов, если доступно несколько вариантов по введенной команде.

Аналогично, автодополнение может использоваться для имен файлов и каталогов. Например, вы хотите перейти в каталог applications/xnfbdh73/public_html. Вместо того, чтобы вводить этот длинный путь (и помнить его!), вводите команду так:

```
cd a <нажмите Tab> / x <нажмите Tab> / p <нажмите Tab>
```

В конечном итоге вместо ввода 33 символов пути вам нужно будет ввести 5!

5.3. Перенаправление ввода/вывода

Рассмотрим следующую команду:

```
cat very_long_file.txt | less
```

Здесь мы пытаемся просмотреть очень длинный файл very_long_file.txt. Поскольку он очень длинный и не помещается на одном экране, мы перенаправили вывод первой части команды (cat very_long_file.txt) на стандартный вывод команды less, которая обеспечивает удобный просмотр длинных файлов.

Синтаксис следующий:

```
команда_1 | команда_2
```

Особых ограничений не существует, и вы можете передать вывод второй команды на ввод третьей и т.д.:

```
команда_1 | команда_2 | команда_3
```

С помощью такого перенаправления можно автоматизировать некоторые команды, что важно в сценариях bash, например, вот как можно утвердительно ответить на запрос об удалении файла:

```
echo y | rm file.old
```

Очень часто перенаправление ввода/вывода используется в таком контексте, в котором использовали его мы: большой вывод перенаправляется на программу просмотра (less) или на программу-фильтр, такую как grep.

Кроме перенаправления вывода программы на ввод другой программы, его можно перенаправить в файл, например:

```
ps -A > processes.txt
```

Если файл processes.txt не существует, он будет создан. Если существует - перезаписан. Если нужно дописать вывод программы в конец файла, не удаляя существующий файл, тогда используйте два знака больше:

```
ps -A >> processes.txt
```

В этом случае, если файл не существует, то он будет создан, а если существует, то информация будет дописана в конец файла.

5.4. Справочная система man

В Linux справочная система всегда под рукой. Например, вы забыли параметры команды **df**, просто введите команду:

```
man df
```

Откроется страница руководства (в большинстве случаев – на русском языке), в котором будут описаны все возможные параметры по интересующей вас команде и даны рекомендации по их применению. Для работы справочной системы не требуется соединение с Интернетом, поскольку все страницы справочного руководства уже загружены на ваш компьютер.

Далее будут рассмотрены некоторые полезные команды, знание которых просто обязательно для каждого пользователя Linux.

5.5. Команды для работы с файлами и каталогами

5.5.1. Команды для работы с файлами

В каждой операционной системе есть команды для работы с файлами и каталогами. Linux - не исключение. Рассмотрим стандартные команды Linux для работы с файлами (см. табл. 5.1).

Таблица 5.1. Стандартные команды Linux для работы с файлами

Команда	Описание
cat файл	Выводит текстовый файл. Файлы могут быть довольно длинными, поэтому лучше использовать ее в паре с командой less, например, cat /etc/services | less
tac файл	Подобна команде cat, но выводит файл в обратном порядке. Данная команда удобна для чтения журналов, в которых самые свежие сообщения заносятся в конец файла, например, tac /var/log/messages | less
cp файл1 файл2	Копирует файл1 в файл2. Если второй файл существует, программа спросит вас, нужно ли его перезаписать
mv файл1 файл2	Используется для перемещения файла1 в файл2. Можно использовать для переименования файлов
rm файл	Удаляет файл
touch файл	Используется для создания нового пустого файла
locate файл	Быстрый поиск файла. Позже мы рассмотрим процесс поиска подробнее
which исполнимый_файл	Производит быстрый поиск программы (исполнимого файла). Если программа находится в пути PATH, то which выведет каталог, в котором находится программа.

В таблице 5.1 представлены основные команды, которые используются для создания (touch), копирования (cp), перемещения (mv) и удаления (rm) файлов, а также несколько дополнительных команд.

Рассмотрим несколько примеров:

```
$ dmesg > kernel.messages
$ cat kernel.messages | less
$ cp kernel.messages krn.msg
$ rm kernel.messages
```

Первая команда выводит загрузочные сообщения ядра в файл kernel.messages. Вторая выводит содержимое этого файла на экран, а команда less огранизует удобный постраничный просмотр этого файла. Далее команда cp копирует файл kernel.messages в файл krn.msg, а последняя команда удаляет наш исходный файл kernel.messages. В принципе, вместо последних двух команд можно было использовать одну команду mv:

```
mv kernel.messages krn.msg
```

При указании имени файла вы можете использовать маски * и ?. Как обычно, символ * заменяет любую последовательность символов, а ? - один символ. Например:

```
rm *.tmp
rm /tmp/*
cp *.txt /media/ext-usb
cp ???.txt /media/ext-usb
```

Первая команда удаляет все файлы, заканчивающиеся на «.tmp», в текущем каталоге. Вторая - удаляет все файлы из каталога /tmp. Третья копирует все файлы с «расширением» .txt из текущего каталога в каталог /media/ext-usb. Четвертая команда копирует все файлы, имя которых состоит из трех любых символов и заканчивается на «.txt», например, abc.txt, в каталог /media/ext-usb.

> Примечание. Как вы уже догадались, к каталогу /media/ext-usb можно подмонтировать внешний USB-диск и тогда копируемые файлы физически окажутся на внешнем жестком диске. Подробнее о монтировании мы поговорим в главе 13.

> Примечание. Обратите внимание, что в таблице 5.1 команды представлены без параметров. Хотя дополнительные параметры есть у каждой команды. Вы не обязаны помнить все параметры, для этого есть справочная система man. Вам нужно помнить только названия команд, а параметры вы всегда сможете «подсмотреть» в man.

Мы не рассмотрели команды редактирования текстовых файлов. Они не являются стандартными (кроме программы vi, которой пользоваться очень

неудобно), но в вашей системе по умолчанию могут быть установлены следующие текстовые редакторы:

- **nano** - удобный текстовый редактор;
- **joe** - небольшой и удобный текстовый редактор;
- **pico** - текстовый редактор, устанавливаемый вместе с почтовым клиентом pine;
- **mcedit** - текстовый редактор, устанавливаемый вместе с файловым менеджером mc.

Если у вас нет этих редакторов, вы можете установить их. Например, установите mc - вы получите и файловый менеджер и текстовый редактор сразу:

```
sudo apt install mc
```

5.5.2. Команды для работы с каталогами

Аналогично командам для работы с файлами, команды для работы с каталогами представлены в таблице 5.2.

Таблица 5.2. Стандартные команды Linux для работы с каталогами

Команда	Описание
cd каталог	Изменение каталога
ls каталог	Выводит содержимое каталога
rmdir каталог	Удаляет пустой каталог
rm -r каталог	Рекурсивное удаление непустого каталога
mkdir каталог	Создает каталог
cp каталог1 каталог2	Команду **cp** можно использовать и для копирования каталогов. В данном случае **cp** копирует каталог1 в каталог 2
mv каталог1 каталог2	Команду **mv** можно использовать и для перемещения каталогов. В данном случае **mv** перемещает каталог1 в каталог2

Обратите внимание, что команда **rmdir** не может удалить непустой каталог, поэтому если не хотите удалять сначала файлы и подкаталоги из удаляемого каталога, то лучше использовать команду `rm -r каталог`. Например:

```
$ mkdir /home/bagira/test
$ touch /home/bagira/test/test-file
$ rm -r /home/bagira/test
```

Как и в случае с командой **rm**, вы можете задать параметр -r для команд **cp** и **mv**. В этом случае операция копирования или перемещения будет выполняться рекурсивно.

Очень важной операцией является просмотр содержимого каталога, для которой используется команда **ls**. Поэтому сейчас сделаем исключением для этой команды и рассмотрим ее параметры (табл. 5.3). А общий формат вызова этой команды таков:

```
ls [параметры] [каталог]
```

Таблица 5.3. Параметры команды ls

Параметр	Описание
-C	Выводит список файлов в колонках с вертикальной сортировкой
-F	Для каждого каталога добавлять суффикс '/', а для каждого исполняемого файла - '*', а для каждого FIFO-канала - '|'
-R	Рекурсивный вывод, то есть команда **ls** будет выводить не только содержимое каталога, но и подкаталогов
-a	Показывать скрытые файлы.
-i	Показывать иноды для каждого файла (будет показан серийный номер файла)
-l	«Длинный» формат вывода, в котором отображается тип файла, права доступа, количество ссылок на файл, имя владельца, имя группы, размер файла, метка времени создания файла и имя файла. В колонке типа файла могут быть следующие значения: d (каталог), b (блочное устройство), c (символьное устройство), l (символическая ссылка), p (FIFO-канал), s (сокет).
-r	Сортировка в обратном порядке

В таблице 5.3 приведены не все параметры команды **ls**, а только самые основные.

При задании имени каталога можно использовать следующие специальные имена:

- . - обозначает текущий каталог.
- .. - обозначает родительский каталог.
- ~ - домашний каталог пользователя, например, если вы вошли под пользователем bagira, то путь ~/file.txt равноценен /home/bagira/file.txt.

5 Команды системного администратора

Существуют команды, которые нужно знать каждому системному администратору. В этой главе рассматривается необходимый минимум таких команд. Нужно отметить, что команд системного администратора гораздо больше в Linux, по сути, в каждой главе мы рассматриваем те или иные команды администратора. В этой главе мы рассмотрим некоторые базовые команды. Возможно, они не пригодятся вам прямо сейчас, но вы еще ни раз вернетесь к этой главе в будущем.

5.6.1. Команды для работы с устройствами и драйверами

В таблице 5.4 приведены некоторые команды, которые помогут вам обнаружить аппаратную проблему - проблему с устройством, либо с его драйвером.

Таблица 5.4. Команды, предоставляющие информацию об устройствах

Команда	Описание
`uname -a`	Очень важная команда, сообщающая версию ядра. Очень важно, чтобы устанавливаемые модули были откомпилированы под вашу версию ядра
`lsdev`	Выводит информацию об устройствах. По умолчанию эта команда не установлена, нужно установить пакет procinfo
`lshal`	Выводит параметры всех устройств
`lspci, lsusb, lshw`	Выводят соответственно список PCI-устройств, USB-устройств и список оборудования компьютера

`lsmod`	Выводит список загруженных модулей ядра
`dmidecode`	Отображает информацию о BIOS компьютера
`cat /proc/cupinfo`	Выводит информацию о процессоре
`cat /proc/meminfo`	Отображает информацию о памяти
`cat /proc/mounts`	Показывает точки монтирования
`cat /proc/net/dev`	Выводит сетевые интерфейсы и статистику по ним
`cat /proc/version`	Похожа на uname, выводит версию ядра
`cat /proc/interrupts`	Отображает информацию по прерываниям
`cat /proc/swaps`	Выводит информацию о файлах подкачки

5.6.2. Команды настройки сетевых интерфейсов

Подробно настройка сети будет рассматриваться в следующей главе, а пока рассмотрим таблицу 5.5, в которой представлен короткий список команд, которые вам могут пригодиться при настройке сети.

Таблица 5.5. Некоторые команды настройки сети

Команда	Описание
`route`	Просмотр и изменение таблицы маршрутизации
`dmesg | less`	Просмотр сообщений ядра, которые выводятся ядром при загрузке системы
`iwconfig`	Выводит информацию обо всех беспроводных интерфейсах
`iwlist scan`	Поиск беспроводных сетей
`dhclient wlan0`	Обновляет IP-адрес и другую сетевую информацию беспроводного интерфейса wlan0

`iwevent`	Просмотреть события беспроводной сети
`sudo /etc/init.d/dbus restart`	Перезапуск всех сетевых служб (работает не во всех дистрибутивах)
`sudo systemctl restart <служба>` или `service <служба> restart`	Перезапуск службы. Например, sudo systemctl restart networking перезапускает сеть

5.6.3. Программы тестирования и настройки жесткого диска

Команды для тестирования и настройки жесткого диска, подобно ранее приведенным командам, также представлены в виде таблицы - табл. 5.6.

Таблица 5.6. Команды для тестирования и настройки жесткого диска

Команда	Описание
`badblocks -v <имя_устройства>`	Осуществляет проверку жесткого диска на наличие «плохих» блоков. Параметр -v включает подробный отчет.
`hdparm`	Тестирование производительности и настройка жесткого диска. Например, параметр -t может протестировать производительность (`hdparm -t /dev/sda`), а параметр -E установить скорость привода CD/DVD: `hdparm -E 2 /dev/sr0`
`hddtemp`	Отображает температуру жесткого диска
`bonnie`	Тестирует производительность жесткого диска
`cpuburn`	Тестирование процессора (стресс-тест процессора)

screentest	Тестирование и настройка монитора
smartmontools	SMART-мониторинг. Нужно, чтобы ваши жесткие диски поддерживали S.M.A.R.T

5.7. Команды обработки текста

5.7.1. Редактор sed

Команда **sed** - мощный потоковый редактор и ему можно было посвятить целую главу, но не вижу такой необходимости, поскольку в современных дистрибутивах имеется документация на русском языке. Главное знать, что такая программа есть. А чтобы вы заинтересовались, давайте рассмотрим несколько примеров использования этой программы:

Заменить строку «string1» на «string2» в файле report.txt, результат вывести на стандартное устройство вывода:

```
sed 's/stringa1/stringa2/g' report.txt
```

вывести пятую строку файла report.txt:

```
sed -n '5p;5q' report.txt
```

Удалить пустые строки из файла:

```
sed '/^$/d' report.txt
```

Удалить строку «string1» из текста, не изменяя всего остального:

```
sed -e 's/string1//g' report.txt
```

Удалить пустые символы в в конце каждой строки:

```
sed -e 's/ *$//' report.txt
```

Удалить пустые строки и комментарии из файла:

```
sed '/ *#/d; /^$/d' report.txt
```

Преобразовать символы из нижнего регистра в верхний:

```
echo 'test' | tr '[:lower:]' '[:upper:]'
```

Удалить первую строку из файла:

```
sed -e '1d' report.txt
```

5.7.2. Подсчет количества слов/символов

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

5.7.3. Сравнение файлов

Команда **cmp** используется для сравнения двух файлов. Если файлы идентичны, то **cmp** вообще никак не реагирует. А вот если файлы отличаются, то **cmp** выводит номер строки и номер символа в строке, откуда начинается различие.

Команда **cmp** более универсальна, поскольку она может использоваться как для сравнения текстовых, так и двоичных файлов. А вот команда **diff** и ее аналоги умеют сравнивать только текстовые файлы.

Формат вызова команды следующий:

```
cmp [параметры] файл1 файл2
```

Параметры команды cmp указаны в табл. 5.7.

Таблица 5.7. Параметры команды cmp

Параметр	Описание
-c	Вывод отличающихся символов
-i n	Игнорировать первые n символов
-l	Вывод позиций всех отличий, а не только первого

-s	Не выводить информацию на экран, при этом код возврата будет следующим: 0 — файлы одинаковые; 1 — файлы отличаются; 2 — ошибка при открытии одного из файлов

5.7.4. Разбивка текста на колонки

Команда **column** используется для разбивки текста на несколько столбцов. Текст может быть прочитан как из файла, так и со стандартного ввода, если файл не указан.

Формат вызова команды:

```
column [параметры] [файл]
```

Параметры команды column приведены в табл. 5.8.

Таблица 5.8. Параметры команды column

Параметр	Описание
-c n	Задает количество столбцов (число n)
-s символ	Указанный символ будет использоваться в качестве разделителя столбцов
-t	Текст будет форматироваться как таблицы. По умолчанию разделителем полей считается пробел, но с помощью параметра -s можно задать другой разделитель
-x	Сначала будут заполняться столбцы, а потом строки

5.7.5. Команды diff и diff3

Команда используется для сравнения двух файлов. Формат вызова программы **diff**:

```
diff [параметры] файл1 файл2
```

В выводе программы отличающиеся строки помечаются символами > и <:

- строка из первого файла помечается символом <;
- строка из второго файла — символом >.

Самые полезные параметры программы diff приведены в табл. 5.9.

Таблица 5.9. Параметры команды diff

Параметр	Описание
-a	Сравнение всех файлов, в том числе бинарных
-b	Программа будет игнорировать пробельные символы в конце строки
-B	Игнорирует пустые строки
-e	Применяется для создания сценария для редактора ed, который будет использоваться для превращения первого файла во второй
-w	Игнорирует 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 diff3 похожа на diff, только используется для сравнения трех файлов. Формат вызова команды таков:

```
diff3 [параметры] файл1 файл2 файл3
```

Программа выводит следующую информацию:

- ==== — все три файла разные;
- ===1 — первый файл отличается от второго и третьего;
- ===2 — второй файл отличается от первого и третьего;
- ===3 — третий файл отличается от первого и второго.

Параметры команды diff3 указаны в таблице 5.10.

Таблица 5.10. Параметры команды diff3

Параметр	Описание
-a	Сравнивать файлы как текстовые, даже если они являются бинарными
-A	Создание сценария для редактора **ed**, который показывает в квадратных скобках все отличия между файлами
-e	Создает сценарий для **ed**, который помещает все отличия между файлами файл2 и файл3 в файл файл1 (будьте осторожны!)
-i	Добавить команды **w** (сохранить файл) и **q** (выйти) в конец сценария **ed**
-x	Создание сценария редактора **ed**, который помещает отличия между файлами в файл файл1
-X	То же, что и -x, но отличия выделяются
-3	Создает сценарий **ed**, который помещает все различия между файлами файл1 и файл3 в файл1

5.7.6. Команда grep

Предположим, что у нас есть файл какой-то большой файл и мы хотим найти в нем все упоминания строки hello. Сделать это можно так:

```
cat file.txt | grep hello
```

Команда cat file.txt передаст содержимое файла file.txt на стандартный ввод команды grep, которая, в свою очередь, выделит строки, содержащие строку hello.

5.7.7. Замена символов табуляции пробелами

Команда **expand** заменяет в указанных файлах символы табуляции на соответствующее количество пробелов. Команде можно передать лишь один параметр -i, означающий, что замена должна быть только в начале строки.

Формат вызова команды:

```
expand [-i] файлы
```

5.7.8. Форматирование текста

Команда **fmt** форматирует текст, выравнивает его по правой границе и удаляет символы новой строки. Синтаксис вызова команды:

```
fmt [параметры] файлы
```

Параметры команды **fmt** приведены в табл. 5.11.

Таблица 5.11. Параметры команды fmt

Параметр	Описание
-c	Не форматировать первые две строки
-p префикс	Форматировать только строки, начинающиеся с указанного префикса
-s	Не объединять строки
-t	Начинать параграф с красной строки
-w n	Задает максимальную длину строки в n символов (по умолчанию 72)

5.7.9. Команды постраничного вывода more и less

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
```

5.7.10. Команды head и tail: вывод первых и последних строк файла

Команда **head** выводит первые десять строк файла, а **tail** — последние десять. Количество строк может регулироваться с помощью параметра -n.

Пример использования:

```
head -n 10 /var/log/messages
tail -n 15 /var/log/messages
```

5.7.11. Команда split

Используется для разделения файлов на части. По умолчанию создаются части размером в 1000 строк. Изменить размер можно, указав количество строк, например:

```
split -200 файл1
```

В данном случае файл будет разбит на части по 200 строк в каждой (кроме, возможно, последней части, где может быть меньше строк).

Команду можно также использовать для разделения файлов на части по размеру информации, а не по количеству строк, например с помощью параметра -b можно указать количество символов в каждой части. Примеры вызова команды:

```
split -b100b файл
split -b100k файл
split -b100m файл
```

Первая команда разделит файл на части по 100 байтов каждая, вторая — на части по 100 Кбайт каждая, третья — по 100 Мбайт каждая.

5.7.12. Команда unexpand

Заменяет последовательные пробелы символами табуляции. По умолчанию 8 пробелов заменяются одним символом табуляции. Количество пробелов можно задать с помощью параметра -t n (где n — количество пробелов).

Синтаксис вызова:

```
unexpand [параметры] файл
```

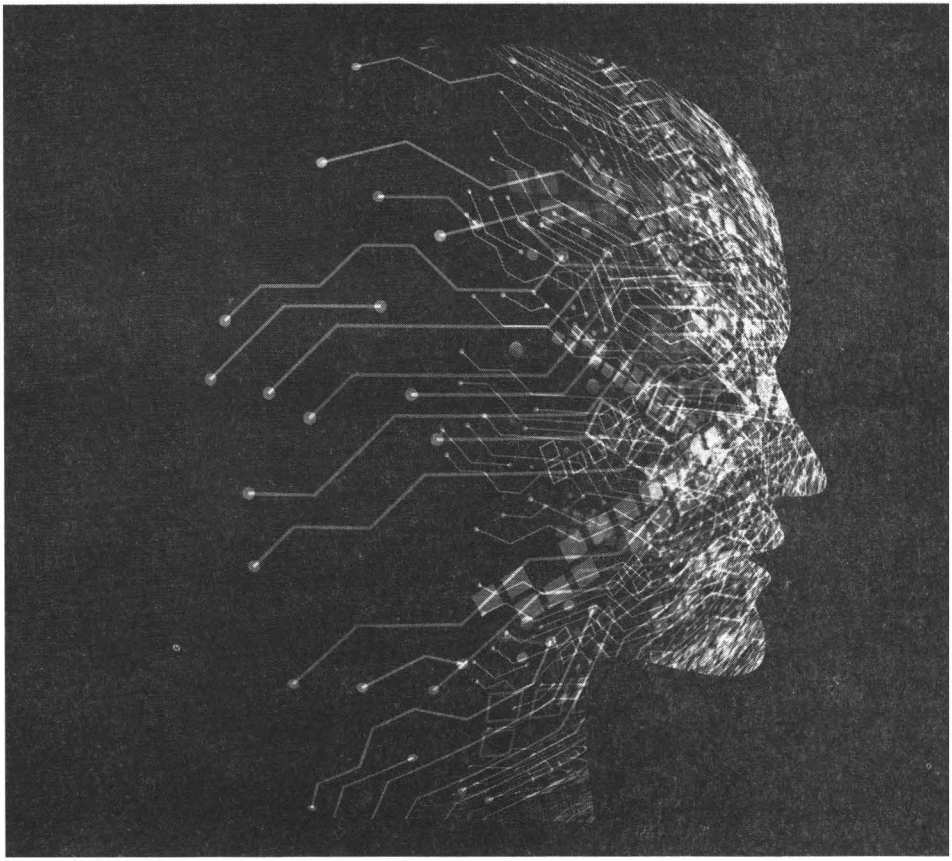
Часть II.

Linux для пользователя

В этой части книги вы узнаете о настройки сети. Будет рассмотрено как настроить локальную Ethernet и беспроводную Wi-Fi-сеть; как установить VPN-соединение. А уже после этого мы рассмотрим установку программного обеспечения, популярные программы и все остальное. Вы узнаете не только как устанавливать программы, но и какую программу установить, что не менее важно.

Глава 6.

Локальная сеть



6.1. Физическая настройка сети Ethernet

Существует много сетевых технологий, но в этой книге мы будем рассматривать настройку локальной сети, построенной на технологии Fast Ethernet. Однако мы рассмотрим ее полностью — от обжатия кабеля до конфигурирования сети в Linux.

Не смотря на наличие уже стандарта Gigabit Ethernet (1000 Мбит/с), стандарт Fast Ethernet (100 Мбит/с) все еще актуален и его скорости вполне хватает для обеспечения работы локальной сети. А стандарт Gigabit Ethernet пока используется в качестве магистрали.

Прежде всего, вам нужно убедиться, что у вас есть сетевой адаптер, поддерживающий технологию FastEthernet. Большинство современных компьютеров оснащены такими сетевыми адаптерами. Как правило, в современных компьютерах и ноутбуках сетевые адаптеры являются интегрированными в материнскую плату и устанавливать их отдельно не нужно.

После это вам нужно подключить сетевой кабель к вашему сетевому адаптеру. Как правило, кабель обжимается администратором сети.

Для создания Fast Ethernet сети вам нужны следующие устройства:

- Сетевые адаптеры — с ними мы уже разобрались;
- Коммутатор (switch) — его можно купить в любом компьютерном магазине. Вместо него сойдет машрутизатор (router), который, как правило, обладает 4-8 портами для подключения локальных компьютеров – этого вполне достаточно для построения небольшой SOHO-сети (Small Office Home Office);
- Витая пара пятой категории — спрашивайте именно такой тип кабеля[1];
- Коннекторы RJ-45 — таких коннекторов вам нужно будет в два раза больше, чем число компьютеров, поскольку кабель нужно будет обжать с двух концов.

1 Для Gigabit Ethernet нужна витая пара 6-ой категории

- Инструмент для обжимки витой пары — хороший инструмент стоит относительно дорого (примерно как коммутатор), а плохой лучше не покупать. Если не хотите выкладываться, возьмите у кого-нибудь на пару дней.

Теперь приступим к самому процессу обжимки. Внутри кабеля будут 4 витые пары, причем у каждого провода будет своя цветовая маркировка. Суть процесса обжимки заключается в том, чтобы подключить каждый из проводов к нужному контакту коннектора. Сначала нужно поместить провода в коннектор (защищать их необязательно — за вас это сделает инструмент), затем коннектор обратной частью (той, которой он будет вставляться в сетевой адаптер) помещается в инструмент для обжимки и крепко обжимается. Используя приведенную ниже таблицу (табл. 6.1), вы без проблем сможете обжать кабель:

Таблица 6.1. Обжимка витой пары

Контакт	Цвет провода
1	Бело-оранжевый
2	Оранжевый
3	Зелено-белый
4	Синий
5	Сине-белый
6	Зеленый
7	Бело-коричневый
8	Коричневый

Обжимать кабель нужно с двух сторон. Один конец подключается к концентратору (или коммутатору), а второй — к сетевому адаптеру. Если вы неправильно (или слабо) обожмете кабель, то ваша сеть работать не будет или же будет работать только на скорости 10 Мбит/с.

Проверить, правильно ли вы обжали кабель очень просто: обратите внимание на коммутатор. Возле каждого порта будет два индикатора. Если горят оба, значит все нормально. Если же горит только один из них, значит, данный порт работает в режиме 10 Мбит/с. А если вообще не горит ни один из индикаторов, значит, вам нужно переобжать кабель.

6.2. Настройка сети с помощью графического конфигуратора

В каждом дистрибутиве Linux есть графические конфигураторы. Конечно, на сервере далеко не всегда устанавливается графический интерфейс, поэтому такие конфигураторы будут недоступны, однако не сказать о них тоже нельзя. Сначала мы рассмотрим графические конфигураторы, а затем рассмотрим настройку сети в консоли - с помощью конфигурационных файлов.

Прежде, чем приступить к настройке сети, ради справедливости нужно отметить, что в большинстве случаев настраивать ничего не придется - во всех современных сетях есть DHCP-сервер, который и настраивает все остальные компьютеры. Настройка сети может понадобиться разве что на самом DHCP-сервере, но это только в том случае, когда вы настраиваете сеть с нуля. Опять-таки, если вы подключили свои домашние компьютеры к маршрутизатору, то в качестве DHCP-сервера будет выступать сам маршрутизатор и вам в большинстве случаев вообще не придется ничего настраивать.

В Ubuntu для настройки локальной сети щелкните по значку сети в правом верхнем углу (рис. 6.1). Выберите команду **Параметры соединения**, чтобы открыть конфигуратор сети.

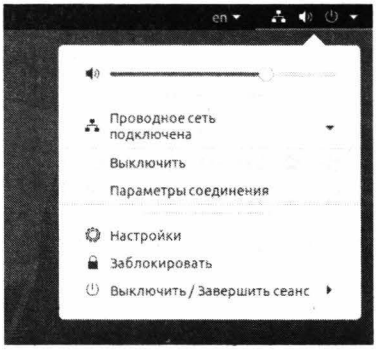


Рис. 6.1. Системное меню

В появившемся окне **Сеть** нажмите кнопку с изображением шестеренки напротив строки **Подключено** – 1000 Мбит/с (рис. 6.2).

В появившемся окне на вкладке **Сведения о системе** убедитесь, что включен флажок **Подключаться автоматически**, чтобы соединение устанавливалось автоматически при загрузке системы (или при входе в систему, если выключен флажок **Сделать доступным для других пользователей**).

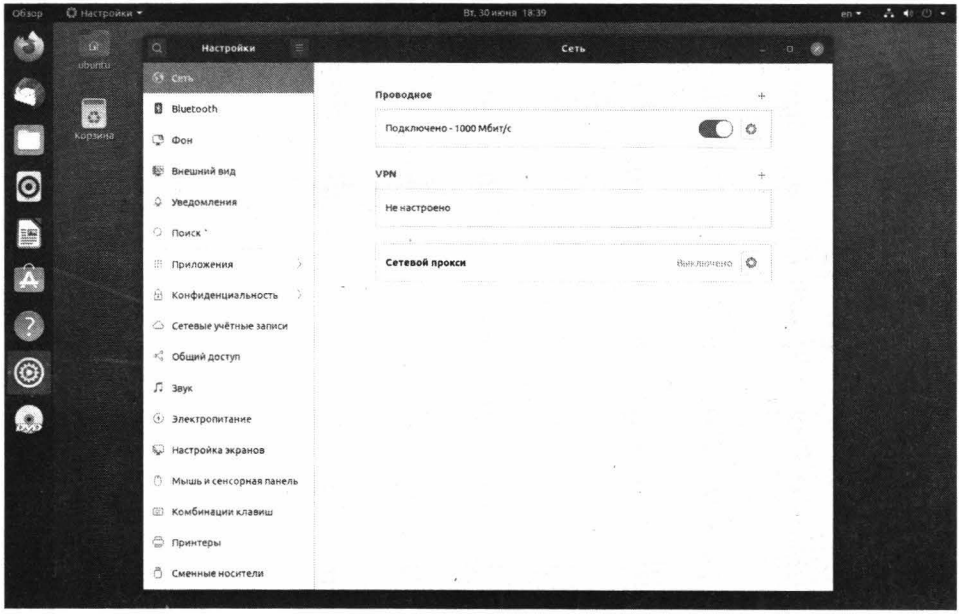


Рис. 6.2. Окно "Сеть"

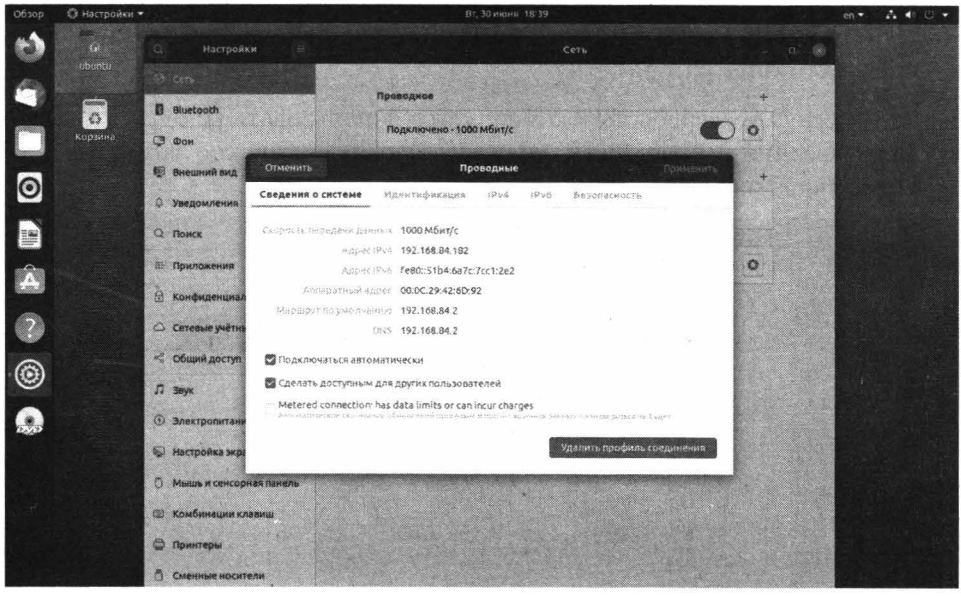


Рис. 6.3. Сведения о системе

Далее перейдите на вкладку IPv4 для изменения параметров протокола IP. По умолчанию сеть настроена на использование протокола DHCP (рис.

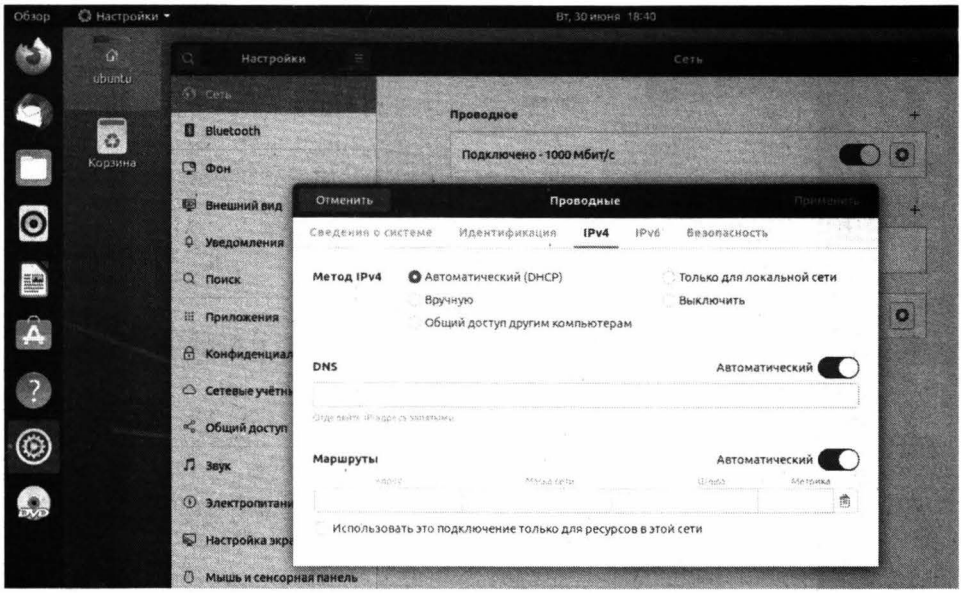


Рис. 6.4. Используем DHCP

6.4). Для ручной настройки выберите **Вручную** и укажите (рис. 6.5):

- IP-адрес, маску сети, IP-адрес шлюза (все эти параметры можно получить у администратора сети).
- IP-адреса DNS-серверов, рекомендуется использовать IP-адреса Google. При указании несколько IP-адресов разделять их нужно запятыми, как показано на рис. 6.5.

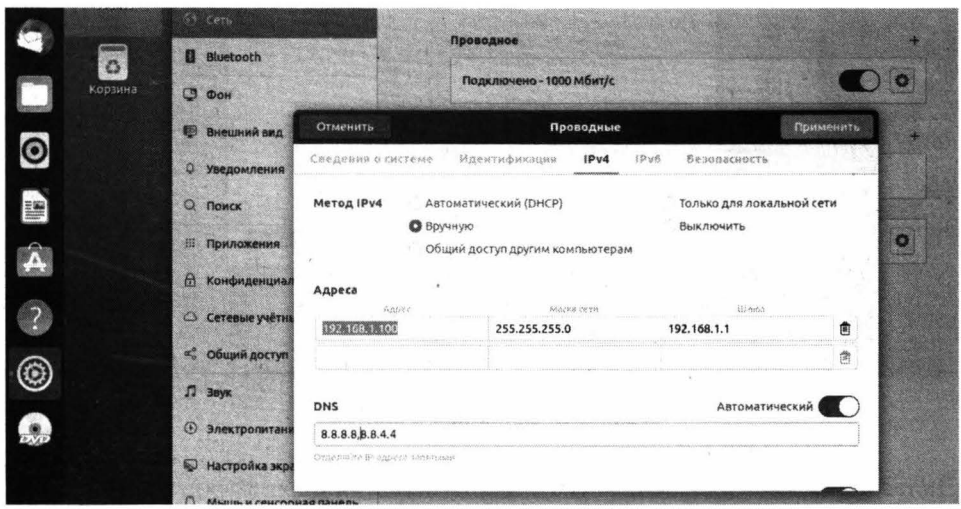


Рис. 6.5. Ручная настройка

Для сохранения настроек нажмите кнопку **Применить**.

В Astra Linux настройка осуществляется аналогично, только интерфейс конфигуратора немного другой. Щелкните правой кнопкой мыши на значке сетевого соединения и выберите команду **Изменить соединения** (рис. 6.6). В появившемся окне выберите сетевое соединение и нажмите кнопку с изображением шестеренки (рис. 6.7).

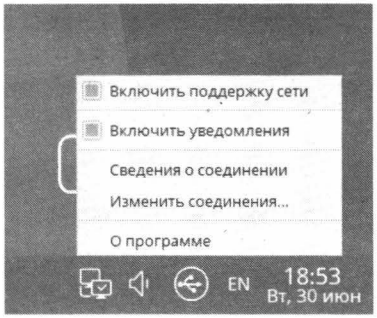


Рис. 6.6. Вызов конфигуратора сети в Astra Linux

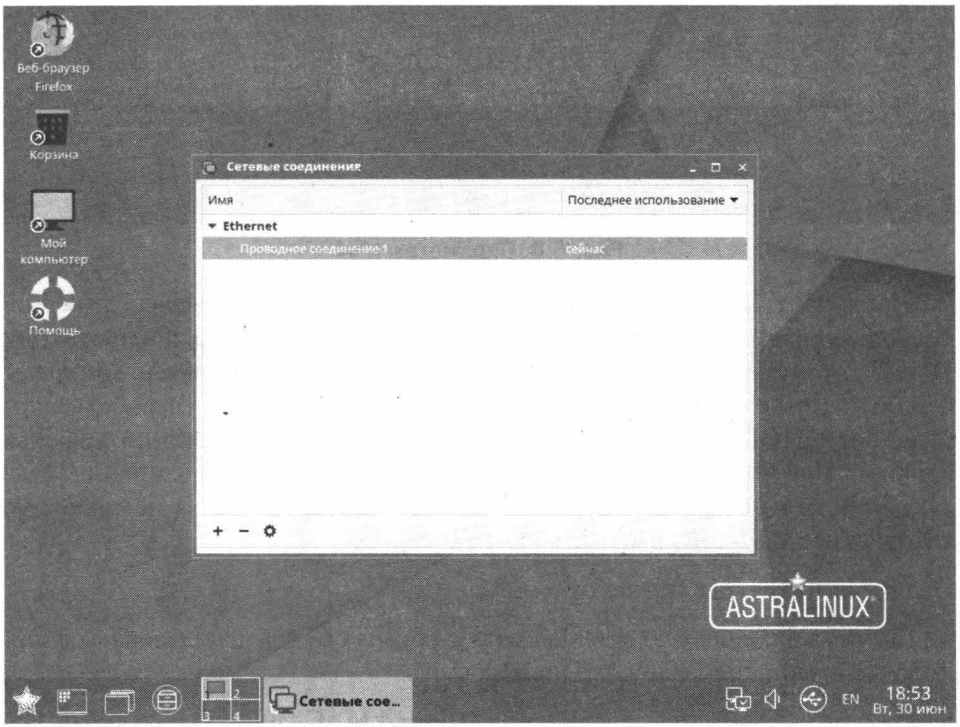


Рис. 6.7. Конфигуратор сети

Далее нужно перейти на вкладку **Параметры IPv4** и установить параметры сети, как это было сделано ранее в Ubuntu. IP-адреса DNS-серверов также указываются через запятые. Для применения настроек нажмите кнопку **Сохранить**.

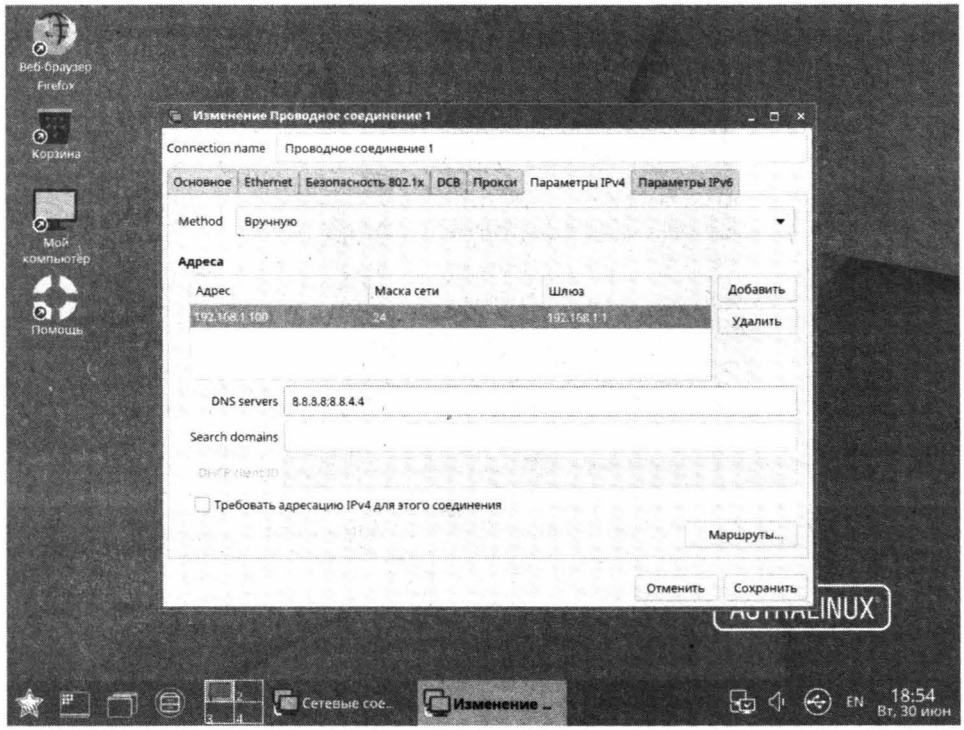


Рис. 6.8. Параметры IPv4

В 99% случаев вам не нужно редактировать параметры Ethernet-соединения, поскольку настройка данного соединения осуществляется по протоколу DHCP автоматически. Конечно, в некоторых случаях, например, когда вам нужно по тем или иным причинам клонировать MAC-адрес устройства или у вас нет DHCP-сервера и нужно определить конфигурацию интерфейса вручную, придется редактировать параметры проводного соединения. Также настройка интерфейса вручную может потребоваться на компьютере, который используется в качестве DHCP-сервере.

На вкладке **Ethernet** можно (рис. 6.9):

- Выбрать физическую сетевую плату, которая будет использоваться для этого соединения (список **Device**);
- Изменить MAC-адрес соединения (**Cloned MAC address**);
- Задать MTU соединения.

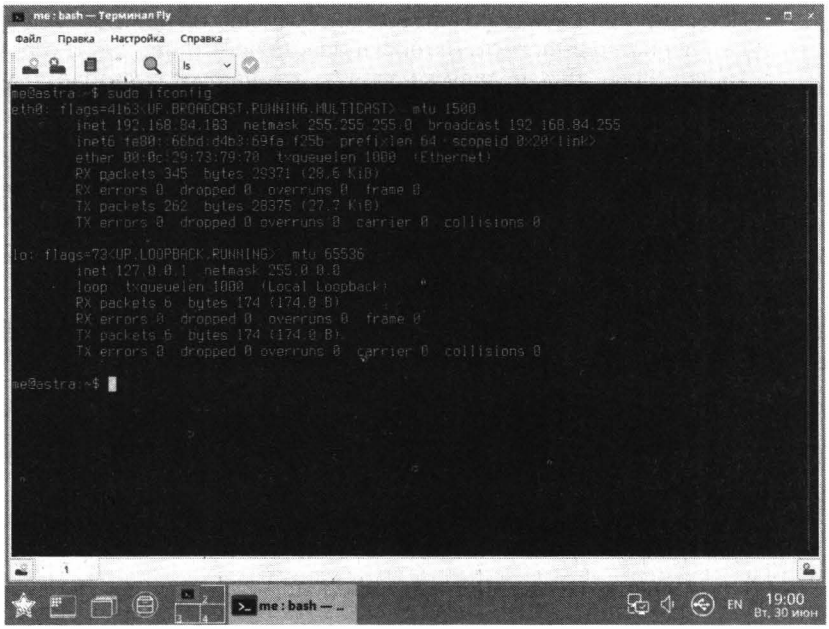


Рис. 6.9. Вкладка Ethernet в Astra Linux

6.3. Команда ifconfig

Команда **ifconfig** используется для настройки и отображения параметров сетевого интерфейса. Если ввести команду ifconfig без параметров, то мы получим список всех активных интерфейсов, например:

```
ens33     Link encap:Ethernet  HWaddr 00:0C:29:C2:0D:D1
          inet addr:192.168.52.154  Bcast:192.168.52.255 Mask:255.255.255.0
           inet6 addr: fe80::20c:29ff:fec2:dd1/64 Scope:Link
           UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
           RX packets:425 errors:0 dropped:0 overruns:0 frame:0
           TX packets:406 errors:0 dropped:0 overruns:0 carrier:0
           collisions:0 txqueuelen:1000
           RX bytes:41540 (40.5 Kb)  TX bytes:39618 (38.6 Kb)

lo         Link encap:Local Loopback
           inet addr:127.0.0.1  Mask:255.0.0.0
           inet6 addr: ::1/128 Scope:Host
           UP LOOPBACK RUNNING  MTU:65536  Metric:1
           RX packets:96 errors:0 dropped:0 overruns:0 frame:0
           TX packets:96 errors:0 dropped:0 overruns:0 carrier:0
           collisions:0 txqueuelen:0
           RX bytes:5760 (5.6 Kb)  TX bytes:5760 (5.6 Kb)
```

> Примечание. Даже если сетевые интерфейсы не настроены, в выводе ifconfig должен быть интерфейс lo. Если его нет, служба network не запущена. Для ее запуска введите команду от пользователя root: `service network start`.

Разберемся, что есть что. У нас есть два сетевых интерфейса - ens33 (Ethrenet-адаптер) и lo (интерфейс обратной петли). Второй используется для тестирования работы сети и всегда имеет IP-адрес 127.0.0.1. Также он может использоваться для обращения к локальному компьютеру. Например, если на вашем компьютере установлен MySQL-сервер, то в PHP-сценариях можно смело указывать адрес 127.0.0.1 или localhost, чтобы указать, что вы обращаетесь к MySQL-серверу, запущенному на этом компьютере.

Рассмотрим поля вывода ifconfig:

- HWaddr - содержит аппаратный MAC-адрес (в нашем случае это 00:0C:29:C2:0D:D1);
- inet addr - содержит IPv4-адрес интерфейса (192.168.52.154);
- inet6 addr - содержит IPv6-адрес интерфейса;
- Bcast - адрес для широковещательной передачи;
- Mask - маска сети;
- MTU - значение MTU (Maximum transmission unit);
- collisions - счетчик коллизий, если количество коллизий больше 0, с вашей сетью творится что-то неладное. Как правило, в современных коммутируемых сетях коллизии отсутствуют вообще;
- RX packets - количество принятых пакетов;
- TX packets - количество переданных пакетов;
- RX bytes - количество принятых байтов;
- TX bytes - количество переданных байтов.

Получить список всех интерфейсов, а не только активных, можно с помощью параметра -a:

```
sudo ifconfig -a
```

С помощью команды **ifconfig** можно деактивировать (down) и активировать (up) любой интерфейс:

```
sudo ifconfig ens33 down
sudo ifconfig ens33 up
```

Иногда такой «перезапуск» интерфейса помогает наладить его работу. Особенно это полезно для «подвисших» беспроводных интерфейсов (wlan*).

С помощью ifconfig можно назначить IP-адрес и другие сетевые параметры интерфейса, например:

```
sudo ifconfig ens33 down
sudo ifconfig ens33 192.168.1.1 up
```

Сначала мы деактивируем интерфейс, затем назначаем новый IP-адрес и «поднимаем» (параметр up) интерфейс. Можно эту последовательность действий выполнить и за три команды:

```
sudo ifconfig ens33 down
sudo ifconfig ens33 192.168.1.1
sudo ifconfig ens33 up
```

Аналогичным образом можно назначить маску сети и широковещательный адрес:

```
sudo ifconfig eth0 10.0.0.1 netmask 255.255.255.240 broadcast
10.0.0.15
```

После изменений параметров интерфейса не забудьте его активировать с помощью команды up:

```
sudo ifconfig eth0 up
```

Изменить MTU интерфейса можно так:

```
sudo ifconfig eth0 down
sudo ifconfig eth0 mtu XXXX up
```

Самой интересной возможностью ifconfig является перевод интерфейса в так называемый promiscuous mode. В обычном режиме сетевая карта, если принимает пакет, который был адресован не ей, она его просто отбрасывает. В promiscuous mode карта будет принимать пакеты, даже которые не были ей адресованы. Такая возможность может понадобиться для перехвата трафика:

```
sudo ifconfig eth0 promisc
```

Перевести карту в обычный режим можно командой:

```
sudo ifconfig eth0 -promisc
```

Помните, что параметры, заданные с помощью ifconfig, хранятся только до следующей перезагрузки. Если вы хотите сохранить назначенные параме-

тры, нужно или использовать конфигураторы сети (описаны ранее) или редактировать файлы конфигурации (будут описаны далее).

6.4. Имена сетевых интерфейсов в Linux

В Linux у каждого сетевого интерфейса есть свое имя, чтобы администратор сразу понимал, что это за соединение. Например, у всех PPP-соединений имя ppp?, где ? - номер соединения: первое соединение называется ppp0, второе - ppp1 и т.д. Нумерация соединений начинается с нуля. В таблице 6.2 приведены названия часто используемых сетевых интерфейсов.

Таблица 6.2. Имена сетевых интерфейсов в Linux

Название интерфейса	Описание
lo	Интерфейс обратной петли
eth, он же ens	Сетевой интерфейс Ethernet
ppp	Соединение PPP (Point-to-Point)
wlan	Беспроводной сетевой интерфейс
tr	Сетевой интерфейс Token Ring
plip	Сетевой интерфейс PLIP (Perallel Line IP). Очень старый интерфейс, вряд ли вы будете с ним иметь дело
sl	Сетевой интерфейс SLIP (Serial Line IP). Тоже «древний» интерфейс.
fddi	Интерфейс к карте FDDI
ax	Сетевой интерфейс любительского радио AX.25

Интерфейсы создаются автоматически ядром для каждого обнаруженного сетевого устройства. Исключения составляют разве что интерфейсы ppp, которые создаются во время настройки сети. Например, у вас может быть PPPoE-соединение к провайдеру и сетевая карта eth0. Сетевой интерфейс eht0 для сетевой карты будет создан автоматически ядро. Но ядро не знает, как вы будете использовать эту карту - или для подключения к локальной сети или как PPPoE-соединение с провайдером, поэтому оно не может создать интерфейс ppp0, прочитав ваши мысли. Поэтому интерфейс ppp0 будет создан уже во время настройки PPPoE/ADSL-соединения с вашим провайдером. О таких соединениях мы поговорим в главе 9.

Все было бы хорошо, если по непонятным причинам в последних версиях ядра интерфейсы не переименовали. Так, старый добрый eth0 теперь почему-то называется ens33, в некоторых дистрибутивах даже enp3s0. Логики в таких именах не очень много, да и хочется, чтобы имена интерфейсов были такие же, как раньше - как-никак 20-летняя традиция. Ради справедливости нужно отметить, что в Astra Linux интерфейс называется как нужно – eth0 (рис. 6.10).

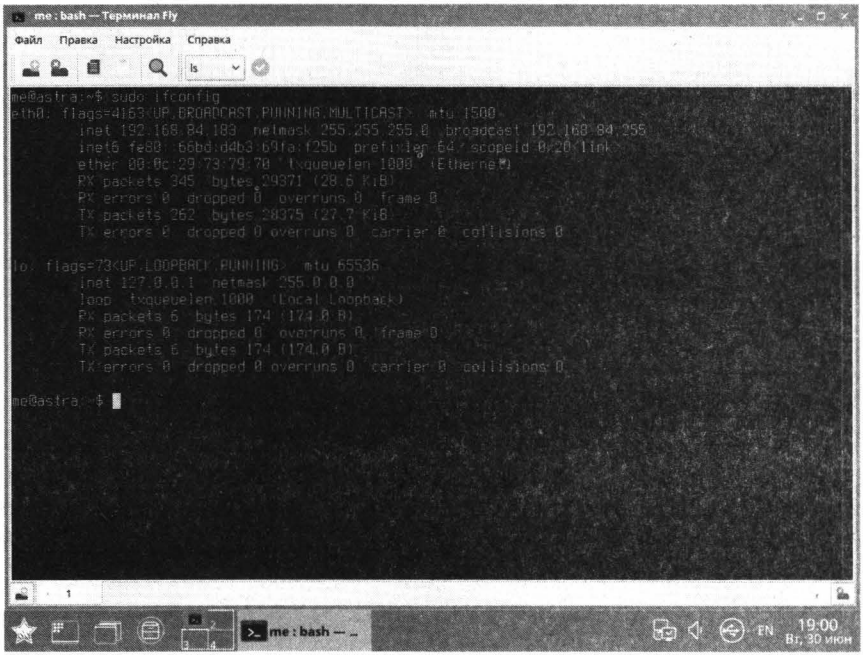


Рис. 6.10. Вывод ifconfig в Astra Linux

На самом деле, логика в новых названиях есть, но только с точки зрения самого компьютера и разработчиков udev. Пользователям и администраторам такая логика не очень понятна. Например, имена, вовлекающие предоставленный BIOS индекс для встроенных в материнскую плату устройств (ID_NET_NAME_ONBOARD) выглядят так - eno1. Имена адаптеров, подключающихся к шине PCI Express, будут содержать номер слота (ID_NET_NAME_SLOT), например, ens1. Имена, вовлекающие физическое/географическое расположение коннектора (ID_NET_NAME_PATH), выглядят еще сложнее - enp2s0. Ну и самые «страшные» - это имена с MAC-адресами интерфейсов (ID_NET_NAME_MAC). Пример такого имени: enx78e7d1ea46da.

Конечно, можно оставить все как есть - все будет прекрасно работать. Но если вам привычнее обычные имена, тогда есть несколько способов вернуть все, как было раньше.

Первый способ заключается в редактировании файла /etc/udev/rules.d/70-my-network.rules (или 70-persistent-net.rules). Если такого файла у вас нет, значит, его нужно создать. Примерное содержимое файла следующее:

```
# PCI device (r8169)
SUBSYSTEM==»net», ACTION==»add», ATTR{address}==»xx:xx:xx:x
x:xx:xx», NAME=»eth0»

# USB device (usb)
SUBSYSTEM==”net”, ACTION==”add”, ATTR{address}==”yy:yy:yy:y
y:yy:yy”, NAME=”wlan0”
```

Здесь xx нужно заменить MAC-адресом вашей сетевой карты, а yy - на MAC-адрес вашего беспроводного адаптера.

Есть и второй способ - передать ядру параметр net.ifnames=0. О том, как это сделать, будет рассказано в главе 18.

6.5. Общие конфигурационные файлы

Наверняка вам интересно, куда конфигураторы сохраняют введенные вами настройки? Об этом мы обязательно поговорим, тем более, что файлы конфигурации сети отличаются в зависимости от используемого дистрибутива. Но сначала мы рассмотрим общие файлы конфигурации, имеющие отношение к сети. Эти файлы есть во всех дистрибутивах Linux и их назначение одинаково.

Файл /etc/hosts

Когда еще не было системы разрешения доменных имен (DNS) в этом файле прописывались IP-адреса узлов и соответствующие им имена узлов. Эти файлы тиражировались от компьютера к компьютеру. Если в сеть добавлялся новый компьютер, изменения нужно было внести в файлы /etc/hosts на всех компьютера. Сегодня этот файл практически не используется и в нем содержится только IP-адрес узла localhost - 127.0.0.1 и аналогичные IPv6-адреса. Формат записей этого файла следующий:

```
IP-адрес    Имя
```

Например:

```
127.0.0.1   localhost
```

Файлы /etc/hosts.allow и /etc/hosts.deny

Содержит IP-адреса узлов, которым разрешен (hosts.allow) или запрещен (hosts.deny) доступ к сервисам данного узла. Опять-таки - это устаревшие конфигурационные файлы. Вы можете их использовать, однако в настоящее время ограничения доступа к определенной службе или узлу достигается путем редактирования правил брандмауэра (глава 25), а не путем редактирования этих файлов. При желании, конечно, вы можете их использовать, но среди своих коллег вы будете выглядеть настоящим динозавром.

Файл /etc/host.conf

Содержит параметры разрешения доменных имен, которые указываются директивой order. Так, `order hosts, bind` означает, что сначала поиск IP-адреса по доменному имени будет произведен в файле /etc/hosts, а затем - с помощью DNS-сервера. Если для вас это существенно, можете изменить порядок на `bind, hosts`, ведь файл hosts уже никем давно не используется и искать там нечего.

Директива `multi on` в этом файле означает, что одному доменному имени могут соответствовать несколько IP-адресов.

Файл /etc/hostname

Обычно в этом файле содержится имя узла. Оно записано одной строкой, больше никаких директив в этом файле нет.

Файл /etc/motd

Содержит сообщение дня, которое может использоваться некоторыми сетевыми службами (например, FTP) и отображаться при входе пользователя в систему. Чтобы ваша сетевая служба использовала именно этот файл, нужно проверить ее файл конфигурации.

Файл /etc/resolv.conf

Обычно в этом файле содержатся параметры резолвера доменных имен. Директива search задает список доменов, в которых будет произведен поиск доменного имени, если оно введено не полностью. Например, пусть директива search определена так:

```
search org org.ua net
```

Вы вводите доменное имя не полностью (например, в браузере) - dkws. Тогда резолвер попытается сначала разрешить доменное имя dkws.org, потом

- dkws.org.ua, а уж потом, если ни один из первых двух вариантов не был разрешен в IP-адрес - dkws.net.

Директива nameserver позволяет определить IP-адрес сервера DNS. В одной директиве можно указать только один сервер. Всего в файле конфигурации может быть определено до четырех серверов имен:

```
nameserver 8.8.8.8
nameserver 8.8.4.4
```

Пример файла конфигурации /etc/resolv.conf приведен в листинге 6.1.

Листинг 6.1. Файл /etc/resolv.conf

```
search dkws.org.ua

nameserver 8.8.8.8
nameserver 8.8.4.4
```

Файл /etc/services

Все мы знаем, что за сетевыми службами закрепляются определенные номера портов. Например, службе pop3 соответствует порт 110, ftp - порт 21. Эти соответствия и определяются в файле /etc/services. Обычно вам не придется редактировать этот файл. Скорее, вы будете его использовать в качестве информационного, чтобы знать, какой порт соответствует какой службе.

Файл /etc/protocols

Тоже сугубо информационный файл (его можно редактировать, но вряд ли вы будете это делать). В нем описываются имена протоколов, их номера, псевдонимы и комментарии, а также ссылки на RFC-документы. Полезный файл, если вы собрались разобраться по очереди во всех сетевых протоколах.

Файл /etc/network/interfaces: конфигурация сети в Astra Linux

Основной конфигурационный файл сети в Astra Linux называется /etc/network/interfaces. Он содержит параметры сетевых интерфейсов, если не используется Network Manager. Вот пример описания интерфейса eth1 со статическим IP-адресом:

```
iface eth1 inet static
   address 192.168.1.2                  # IP-адрес
   netmask 255.255.255.0                # маска сети
   gateway 192.168.1.1                  # Шлюз
```

Думаю, все ясно. Если интерфейс настраивается по DHCP, его конфигурация будет выглядеть так:

```
auto eth0
iface eth0 inet dhcp
```

Каталог /etc/NetworkManager/system-connections: конфигурация сети в Ubuntu

В Ubuntu параметры сетевого интерфейса хранятся в каталоге /etc/NetworkManager/system-connections. В этом каталоге находятся несколько файлов, имена этих файлов соответствуют названиям соединений в NetworkManager. Например, в файле 'Проводное соединение 1.nmconnection' находятся параметры первого проводного соединения (лист. 6.2).

Листинг 6.2. Пример настройки интерфейса со статическим IP-адресом

```
[802-3-ethernet]
duplex=full

[connection]
id=Проводное соединение 1
uuid=00779167-0a51-44ad-a2b3-b5765a65b496
type=802-3-ethernet
timestamp=1419337229

[ipv6]
method=auto
ip6-privacy=2

[ipv4]
method=manual
dns=8.8.8.8;
addresses1=192.168.1.101;24;192.168.1.1;
```

Как видите, протокол IPv4 настраивается вручную (method=manual). Был задан DNS-сервер (8.8.8.8), а также IP-адрес узла (192.168.1.101), маска

сети (24 - соответствует 255.255.255.0), а также адрес шлюза - 192.168.1.1. Если нужно добавить еще один IP-адрес, просто добавьте строку addressesN:

```
addresses2=192.168.1.102;24;192.168.1.1;
```

Для настройки интерфейса по DHCP измените параметр **method**:

```
[ipv4]
method=auto
```

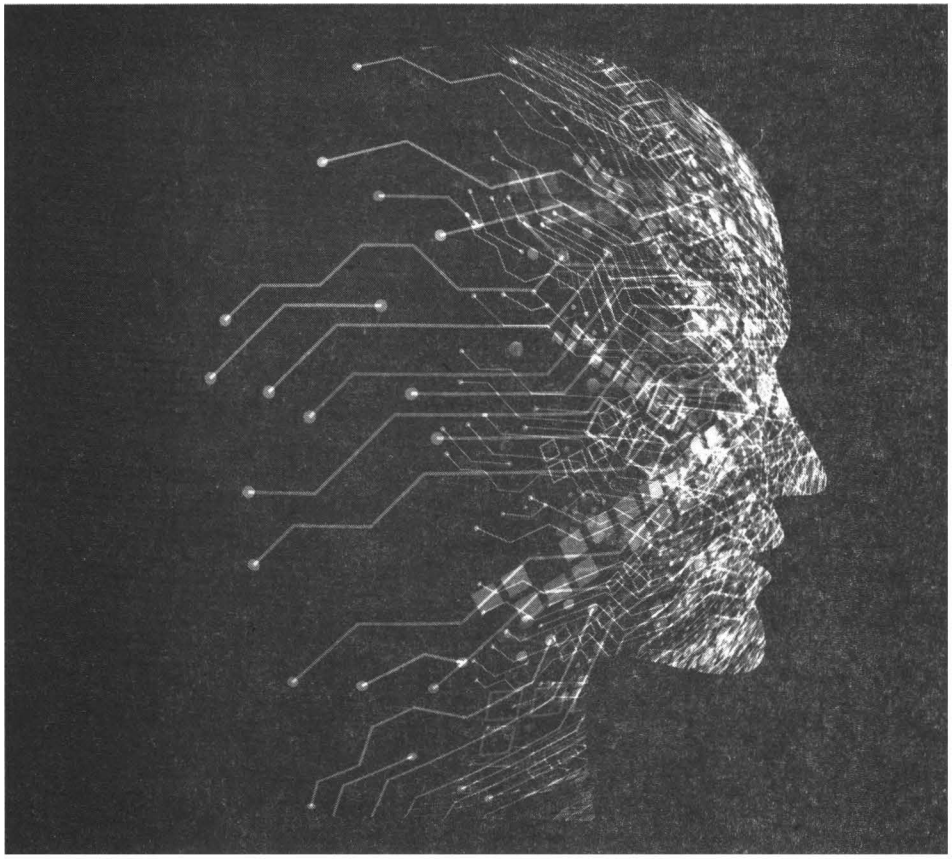
Если вы изменили параметры интерфейса, контролируемого Network Manager и не через интерфейс NM, а вручную, тогда нужно перезапустить Network Manager:

```
# service network-manager restart
```

На этом все. В следующей главе будет рассмотрена конфигурация беспроводной сети.

Глава 7.

Беспроводная Wi-Fi сеть



Беспроводные сети быстро завоевали популярность. Оно и понятно – быстро, просто, дешево. Судите сами: не нужно прокладывать кабели, обжимать их. Все, что нужно – купить «коробочку» (самые дешевые сейчас можно встретить чуть дороже 1000 рублей) и включить ее. Можно даже не настраивать, а использовать те пароли, которые указаны на самой «коробочке». В итоге, начальное «разворачивание» сети у вас займет 5 минут на выбор и размещение Wi-Fi-маршрутизатора и его включение, и еще по 1 минуте на настройку каждого узла, который будет к подключаться к беспроводной сети. Это потом уже вам захочется улучшить прием, выбрав другой канал, поменять пароли по умолчанию и т.д. Если у вас к беспроводной сети подключается 5 устройств, то на первоначальную настройку всей сети будет потрачено 10-15 минут.

Давно прошли те времена, когда настройка беспроводной сети в Linux напоминала шаманские заклинания и пляски с бубном вокруг компьютера. Сейчас все происходит так же, как и в случае с «нормальными» операционными системами – выбрал сеть, ввел пароль и работаешь.

Мы рассмотрим, как подключиться к Wi-Fi с помощью графического интерфейса, и как проделать то же самое в консоли. Такой «лайфхак» может понадобиться вам при администрировании сервера без графического интерфейса. Обычно серверы подключают посредством проводной сети, что надежнее, но и с проводной сетью может что-то случиться, а какой-никакой канал нужно обеспечить, поэтому в качестве резервного канала можно использовать беспроводную сеть. Вот ее настройку мы и рассмотрим.

7.1. Настройка беспроводной сети с помощью графического интерфейса

Мы будем считать, что беспроводной адаптер подключен к компьютеру и включен. Если у вас стационарный компьютер, убедитесь, что вы подключили WiFi-адаптер к USB-порту. Также убедитесь, что он работает – на самих адаптерах иногда есть переключатели, позволяющие включать/выключать их. Если адаптер работает, индикатор на нем светится.

С ноутбуками может быть все немного сложнее. Адаптер, как правило, встроен в ноутбук, но может быть выключен. Обычно включение/выключение адаптера происходит посредством комбинации клавиш Fn + Fx, где Fn – это клавиша Fn (находится, как правило, справа от левого Ctrl), а Fx – одна из функциональных клавиш (F1 – F12). Над такой клавишей обычно есть изображение беспроводной сети или самолета. Так, на ноутбуках HP на клавише F12 есть изображение самолета, нажатие комбинации Fn + F12 переводит ноутбук в режим полета, когда WiFi выключается, повторное нажатие переключает ноутбук в обычный режим – в нем WiFi доступен. На некоторых ноутбуках могут быть аппаратные переключатели (выполненные в виде переключателя или кнопки) для включения/выключения WiFi. Если не разберетесь, обратитесь к документации по ноутбуку.

Итак, если беспроводной адаптер подключен и включен, в меню NetworkManager появится возможность подключения к беспроводной сети (рис. 7.1). Если в меню не появился пункт **Wi-Fi сеть не подключена**, значит, есть какая-то проблема с беспроводным адаптером. Как правило, он просто выключен. Если данный пункт в меню появился, выберите его.

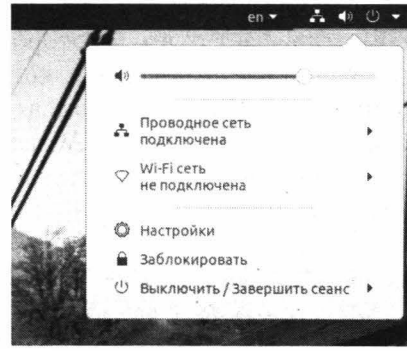


Рис. 7.1. WiFi активирован, но не подключен к сети

Далее нужно выбрать команду **Выбрать сеть** (рис. 7.2). Появится список сетей, в котором нужно выбрать сеть и нажать кнопку **Соединиться** (рис. 7.3). Далее введите пароль и дождитесь подключения к сети.

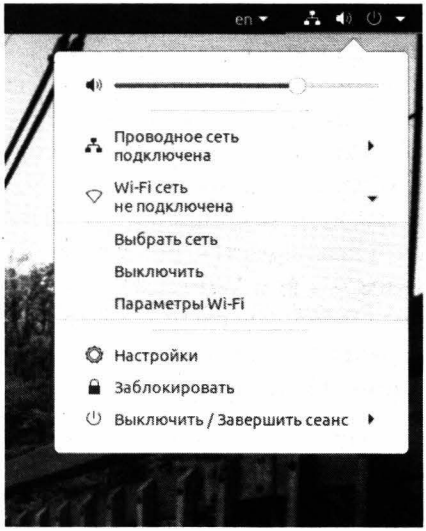


Рис. 7.2. Используйте команду Выбрать сеть

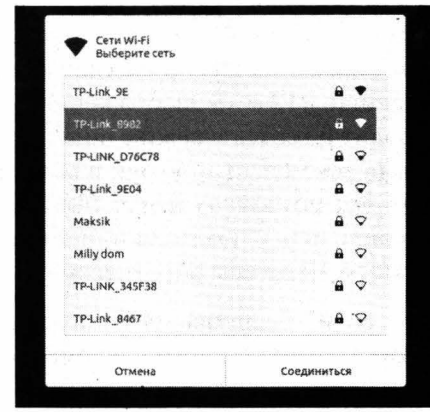


Рис. 7.3. Список сетей

Команда **Параметры WiFi** открывает окно, в котором будет список сетей. Нажмите значок шестеренки напротив сети, к которой вы подключены, чтобы открыть ее параметры. Особо интересного в этих параметрах нет, разве что вы можете узнать аппаратный адрес (MAC-адрес) беспроводного адаптера, а также разрешить/запретить подключаться к сети автоматически и сделать это подключение доступным для других пользователей компьютера (рис. 7.4)

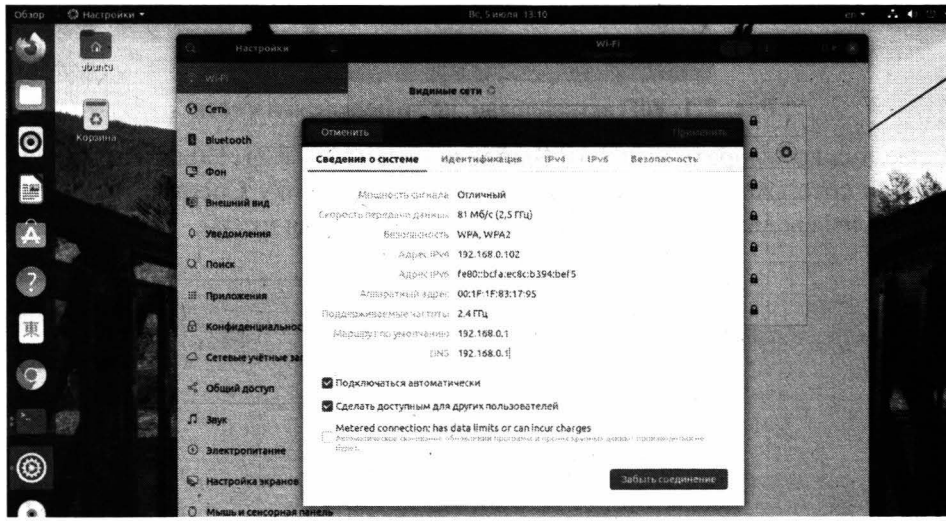


Рис. 7.4. Параметры беспроводного соединения

Параметр **Metered connection: has data limits or can incur charges** полезен, если вы подключаетесь к WiFi, созданной путем «расшаривания» соединения на мобильном телефоне или же при подключении к платной WiFi отеля, где есть тарификация трафика. В этом случае система при подключении по тарифицируемому соединению не будет загружать обновление программ и прочих больших объемов данных.

Отдельного внимания заслуживает подключение к скрытой сети. Такие сети не отображаются в списке сетей. Для подключения к такой сети в Ubuntu нужно открыть окно **Параметры WiFi**, далее из меню выбрать команду **Подключиться к скрытой сети** (рис. 7.5). После этого нужно будет ввести SSID сети и пароль для подключения к ней.

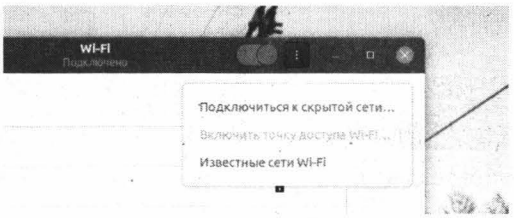


Рис. 7.5. Подключение к скрытой сети

В Astra Linux подключение к WiFi еще удобнее, чем в Ubuntu. Хотя бы потому, что не нужно щелкать по пункту **Wi-Fi сеть не подключена** и не нужно выбирать команду **Выбрать сеть** – вы экономите целых два клика мышки. Щелкните по значку NetworkManager и вы сразу получите список сетей. Если нужно сети нет в списке, используйте меню **Еще сети**, а для подключения к скрытой сети – команду **Подключиться к скрытой сети Wi-Fi**.

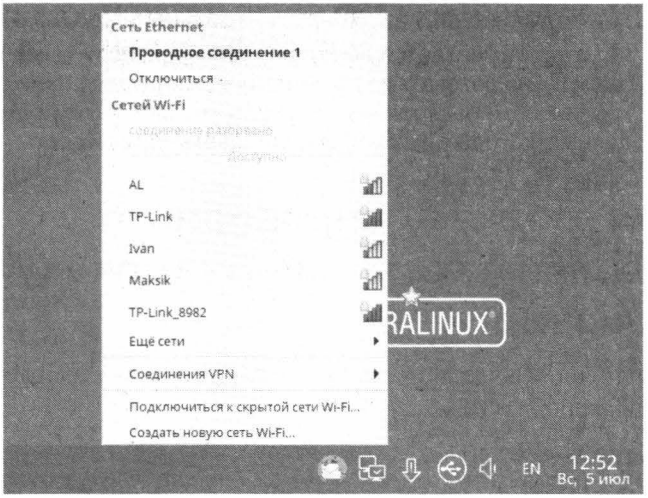


Рис. 7.6. Выбор сети

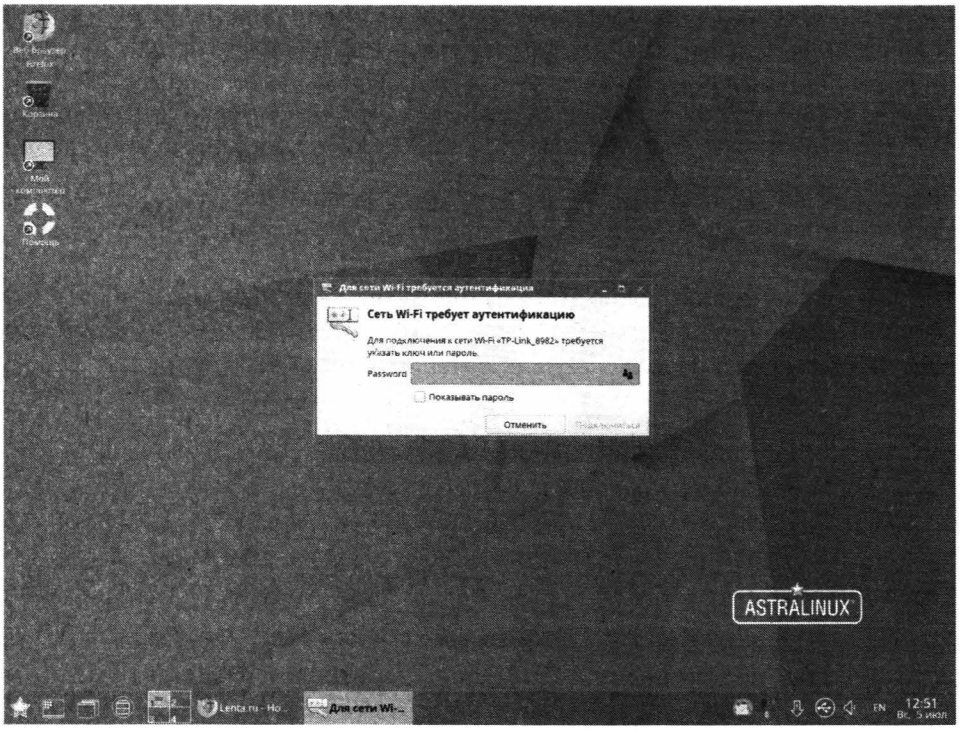


Рис. 7.7. Ввод пароля

Далее нужно ввести только пароль для подключения к сети и дождаться самого подключения.

> Примечание. Обычно WiFi-сети требуют пароль для подключения к ним. Такие сети в списке сетей отмечаются замочком. Если замочка нет, то сеть считается доступной для всех. Если вы не знаете, что это за сеть, подключаться к ней не рекомендуется, поскольку часто открытые сети разворачиваются злоумышленниками для кражи ваших данных, которые вы будете передавать через их сеть.

7.2. Настройка беспроводного соединения в командной строке (WEP-шифрование)

Первое, что нужно сделать - посмотреть, какие сетевые адаптеры имеются у нас на компьютере:

```
sudo ifconfig -a
```

Вывод будет содержать имена и подробное описание всех сетевых интерфейсов, которые удалось обнаружить утилите ifconfig. Если не был обна-

ружен желаемый, то причина заключается только в одном - нет драйверов для него и не включена поддержка этого интерфейса в ядре Linux. Обычно беспроводной интерфейс называется wlan0. Запустим его:

```
sudo ifconfig wlan0 up
```

Опция **up** говорит команде ifconfig запустить для работы («поднять») сетевое устройство.

Теперь нам надо сканировать эфир вокруг себя на наличие доступных беспроводных сетей:

```
sudo iwlist wlan0 scan
```

Здесь wlan0 – имя нашего адаптера, scan – режим сканирования сетей.

Результатом работы **iwlist** будет детальный отчет, из которого на данном этапе нас интересует только одна строчка: ESSID:»Some_Name». Значение параметра ESSID («Some_Name») - это имя беспроводной точки доступа. Теперь мы знаем, к какой конкретно wifi-сети мы будем подключаться.

Выполняем подключение:

```
sudo iwconfig wlan0 essid Имя_сети key Пароль
```

Здесь все просто: мы указываем имя адаптера, SSID сети и пароль для подключения к ней.

Команда iwconfig по умолчанию использует для ключа шифрования данные в шестнадцатеричном виде HEX. Если вы хотите указать ключ в виде простого текста (ASCII), вам необходимо использовать опцию s.

Например, так:

```
sudo iwconfig wlan0 essid Some_Name key s:Wireless_Key
```

Соединение установлено.

Последний шаг - получаем от dhcp-сервера wifi-точки IP-адрес:

```
sudo dhclient wlan0
```

Естественно, вышеуказанные шаги выполнять каждый раз утомительно. Можно упростить процесс установки соединения, написав скрипт подключения, в котором мы объединим все эти команды в одно целое:

```
#!/bin/bash
ifconfig wlan0 up
iwconfig wlan0 essid Some_Name key s:Wireless_Key
```

```
sleep 10
dhclient wlan0
```

Здесь мы добавили еще одну команду **sleep** с параметром 10 секунд. Это рекомендуется делать перед получением IP-адреса для надежности установки соединения.

Сохраняем этот файл под каким-либо именем (например, wireless_up) и делаем его исполняемым командой:

```
sudo chmod u+x wireless_up
```

Переносим wireless_up по пути /usr/local/bin, чтобы сделать его глобально видимым всей системой.

Теперь вам достаточно набрать в командной строке:

```
sudo wireless_up
```

... и соединение будет установлено.

7.3. Соединение с точкой доступа по WPA-шифрованию

Соединение с таким шифрованием поддерживает только утилита wpa_supplicant, поэтому она нам понадобиться. Также, опять-таки, предполагаем, что мы знаем ключ (пароль) шифрования этой точки доступа.

Генерируем пароль на основе этого ключа с помощью утилиты wpa_passphrase, которая входит в состав пакета wpa_supplicant. Дело в том, что пароль, который мы будем использовать далее, должен быть в виде шестнадцатеричного числа:

```
sudo wpa_passphrase ssid password
```

Утилита выдаст сгенерированную строку psk, которую мы вставим в конфигурационный файл wpa_supplicant.conf:

```
Network={
ssid=SSID
psk=PSK }
```

Это очень упрощенный файл конфигурации, но он будет работать. Возможно, вам потребуется добавить в шапку этого файла еще одну строку:

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=wheel
```

для предоставления необходимых прав доступа.

Поднимаем» интерфейс wlan0:

```
sudo ifconfig wlan0 up
```

Указываем, к какой точке мы хотим подключиться:

```
sudo iwconfig wlan0 essid ssid
```

Запускаем утилиту wpa_supplicant на установку соединения:

```
sudo wpa_supplicant -B -Dwext -i wlan0 -c /etc/wpa_
supplicant.conf
```

Разберемся, какие параметры мы указали:

- -B - запускать команду wpa_supplicant в фоновом режиме;
- -Dwext - говорим утилите wpa_supplicant использовать драйвер wext для интерфейса wlan0;
- -i - задаем настраиваемый сетевой интерфейс (wlan0 в нашем случае);
- -c - указываем путь к конфигурационному файлу wpa_supplicant.conf.

Проверяем, что соединение установлено:

```
sudo iwconfig wlan0
```

На выводе увидим подробную информацию об интерфейсе wlan0.

Получаем локальный IP-адрес:

```
sudo dhclient wlan0
```

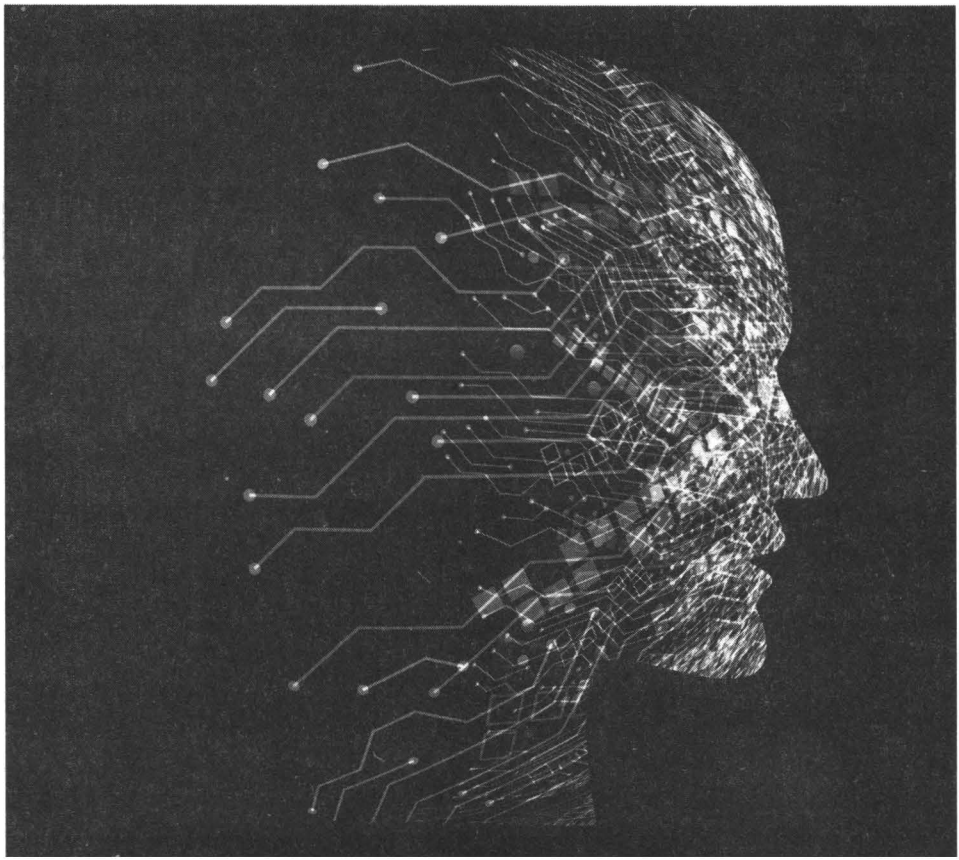
Упрощаем процесс, создав в файле /etc/network/interfaces запись следующего вида:

```
auto wlan0
iface wlan0 inet dhcp
pre-up wpa_supplicant -Bw -Dwext -i wlan0 -c /etc/wpa_
supplicant.conf
post-down killall -q wpa_supplicant
```

В зависимости от дистрибутива Linux, существует множество способов настройки wifi-соединений. Приведенные примеры позволяют настроить соединение практически в любой Linux-системе. Главное, чтобы сам беспроводной адаптер поддерживался в Linux на уровне драйверов.

Глава 8.

VPN-соединение



8.1. Зачем нужна VPN

VPN (Virtual Private Network) – очень популярная в последнее время технология. Технология довольно многогранная и может использоваться для самых разных целей. Основная ее задача – шифрование трафика, передаваемого по VPN-соединению. Пользователи используют VPN для достижения одной из целей – обеспечение анонимности и/или обход блокировки ресурса, получение безопасного доступа к сети предприятия из небезопасной сети.

Бывает так, что контролирующие органы закрывают доступ к тем или иным ресурсам, которые пользователь хочет получить. Законность и моральную сторону этих вопросов рассматривать не будем, тем более, что иногда ресурсы закрываются по политическим причинам. Вовсе не означает, что если ресурс заблокирован, то сразу на нем продают наркотики, оружие и детскую порнографию. Например, в Украине запретили доступ к популярным социальным сетям сугубо по политическим причинам и тем самым способствовали совершенствованию технических навыков большей части населения – даже обычная домохозяйка знает, что такое VPN и как его использовать для подключения к Одноклассникам.

Вторая распространенная причина – подключение к офисной сети для передачи документов, составляющих коммерческую тайну. Поскольку не всегда сеть, через которую проходит соединение с офисом, безопасна (например, это может быть сеть отеля, ресторана, аэропорта), то доверять такому соединению нельзя и поэтому настраивают передачу данных по зашифрованному VPN-соединению. Так можно быть уверенным, что важные документы не будут перехвачены.

Разберемся, как настроить VPN-подключение в Ubuntu и Astra Linux.

8.2. Настройка VPN-подключения в Ubuntu

Для настройки подключения к виртуальной частной сети в Ubuntu выполните следующие действия:

1. Откройте окно **Настройки.**

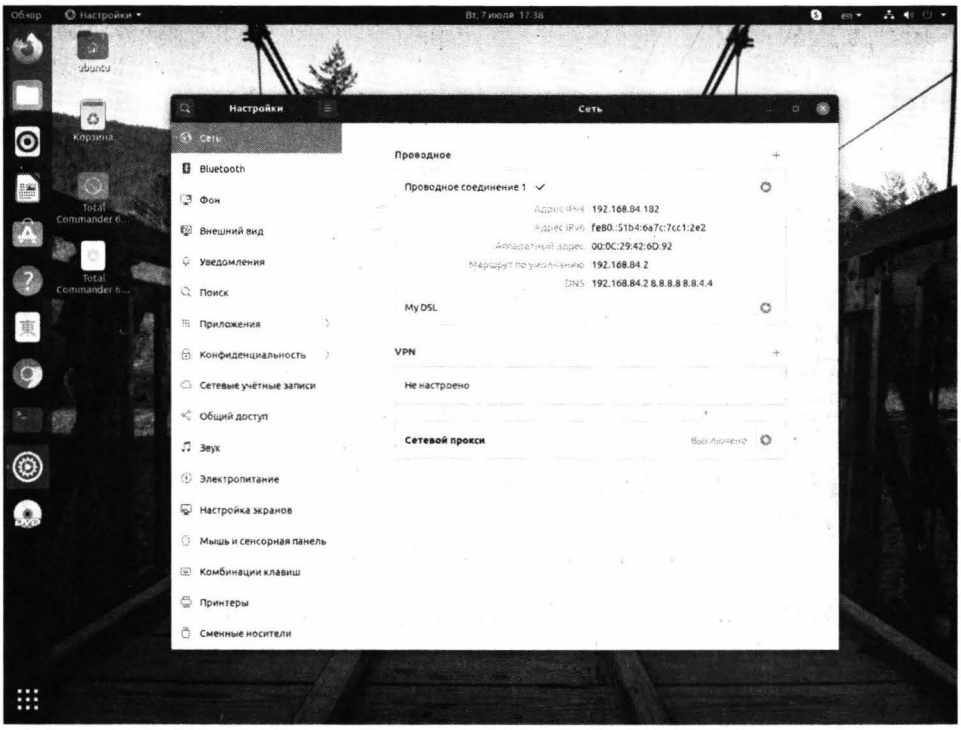


Рис. 8.1. Настройки, Сеть

2. Перейдите в раздел **Сеть.**
3. В группе **VPN** нажмите кнопку **+**
4. Выберите тип протокола – OpenVPN или PPTP. Выберите OpenVPN. Можно также выбрать вариант **Импортировать из файла** и выбрать файл, созданный нами при настройке учетной записи пользователя. Тогда можно пропустить указанные далее действия.
5. Выберите сертификаты, сгенерированные в процессе настройки сервера и создания учетной записи пользователя, введите пароль от сертификата.
6. Введите IP-адрес или доменное имя сервера VPN. Остальные параметры можно не изменять.
7. Если нужно изменить порт соединения, нажмите кнопку **Дополнительно** и укажите порт и/или другие дополнительные параметры.
8. Нажмите кнопку **Добавить** в верхнем правом углу.
9. Щелкните по созданному соединению для установки подключения.

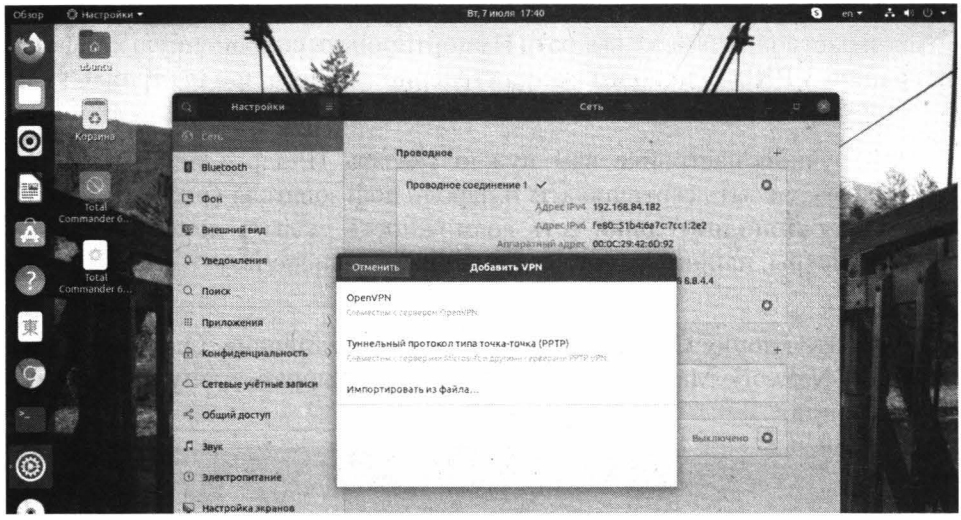


Рис. 8.2. Выбор типа протокола

Рис. 8.3. Параметры соединения

8.3. Настройка VPN-подключения в Astra Linux

Настроить VPN-соединение можно так:

1. Из меню **Network Manager** выберите команду **Соединения VPN, Добавить VPN-соединение** (рис. 8.4);

2. Появится окно, в котором нужно или выбрать пункт **OpenVPN** для ручной настройки или же выбрать **Импортировать сохраненную конфигурацию VPN** для использования сгенерированного при настройке учетной записи файла;

3. При ручной настройке вам нужно указать IP-адрес или имя VPN-сервера, указать сертификаты и пароль пользователя (рис. 8.5). Нажав кнопку **Дополнительно**, есть возможность указать дополнительные параметры, например, порт соединения, если используется нестандартный;

4. Нажмите кнопку **Сохранить** для создания соединения. Оно появится в меню Network Manager и его можно будет выбрать для установки соединения.

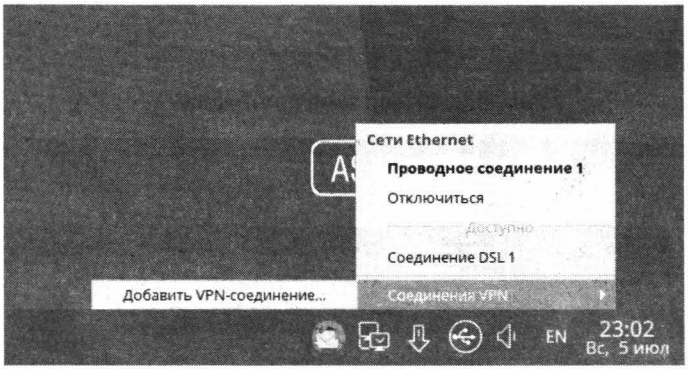


Рис. 8.4. Меню Network Manager

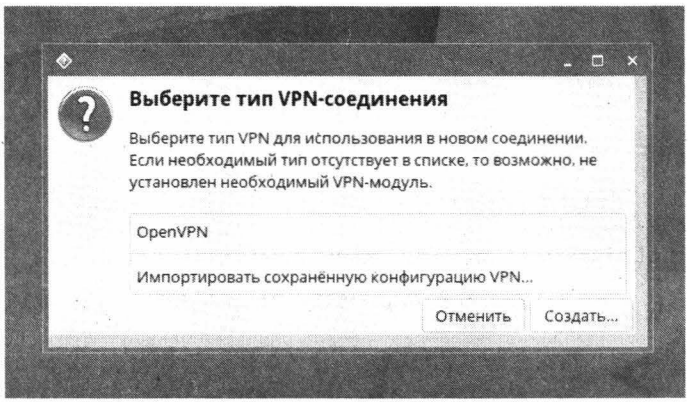
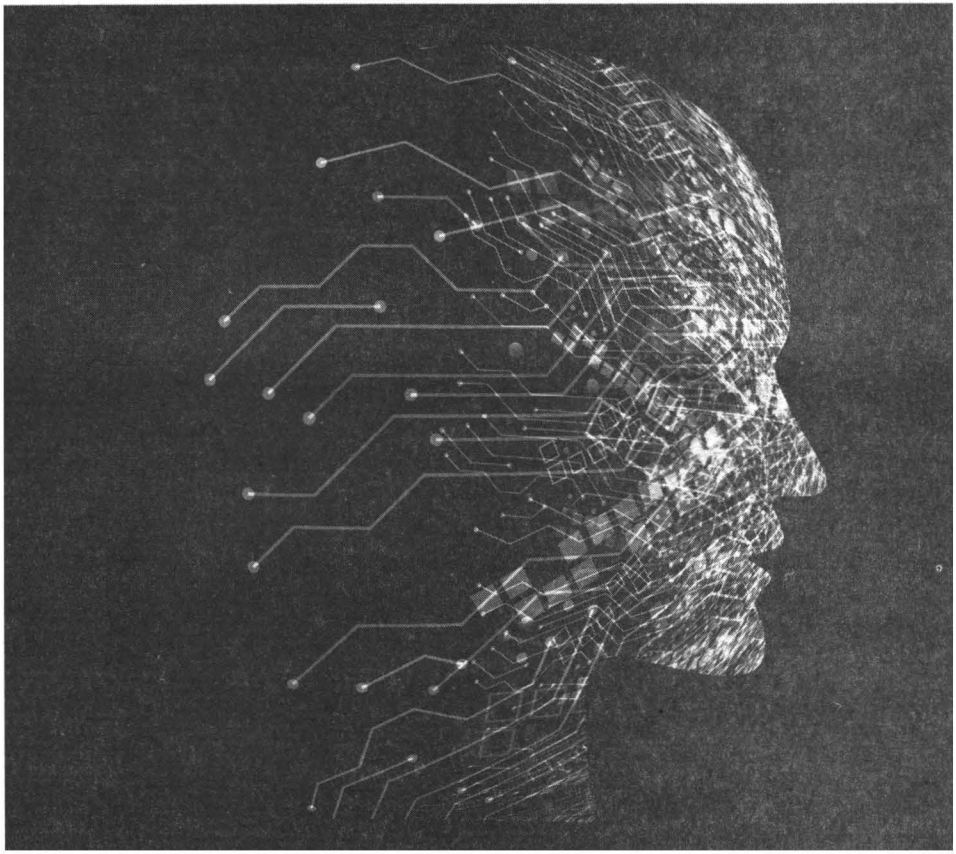


Рис. 8.5.

Глава 9.

DSL-соединение



9.1. Несколько слов о DSL-доступе

Технология DSL (Digital Subscriber Line) является довольно популярной, поскольку сочетает в себе надежность, невысокую стоимость и высокую скорость соединения.

Существует различные варианты предоставления DSL-доступа конечному пользователю. Самый распространенный из них - пользователю предоставляется DSL-модем, который, с одной стороны, подключается к Ethernet-адаптеру, а с другой - к телефонной линии через специальный ADSL-сплиттер (чтобы была возможность принимать обычные звонки).

Второй вариант немного проще - к компьютеру пользователя подведен лишь Ethernet-кабель, ведущий к оборудованию провайдера, а само оборудование пользователь даже и не видит.

В обоих случаях нужно настроить PPPoE-соединение. Протокол PPPoE (PPP over Ethernet) подразумевает передачу кадров протокола PPP по Ethernet-линии. В этой главе будет рассмотрена настройка PPPoE-клиента, а рассмотрение настройки PPPoE-сервера выходит за рамки этой книги.

Соединения DSL/PPPoE уже не такие популярные, какими они были лет пять назад и мы столкнулись с небольшой дилеммой: стоит ли включать описание процесса настройки в книгу? С одной стороны, соединение устаревает, и провайдеры постепенно от него отказываются. С другой стороны, все еще есть провайдеры, использующие этот тип соединений. Масла в огонь подлили разработчики Ubuntu: если раньше для создания DSL-соединения нужно было нажать кнопку + и ввести его параметры, то в Ubuntu 20.04 пришлось разбираться. Именно поэтому в данной книге есть описание процесса настройки этого соединения.

9.2. Настройка DSL/PPPoE в Ubuntu 20.04

Откройте окно настройки и перейдите в раздел Сеть. Вы будете удивлены, но вы не найдете там возможности создать DSL-соединение. Процедура настройки будет выглядеть так:

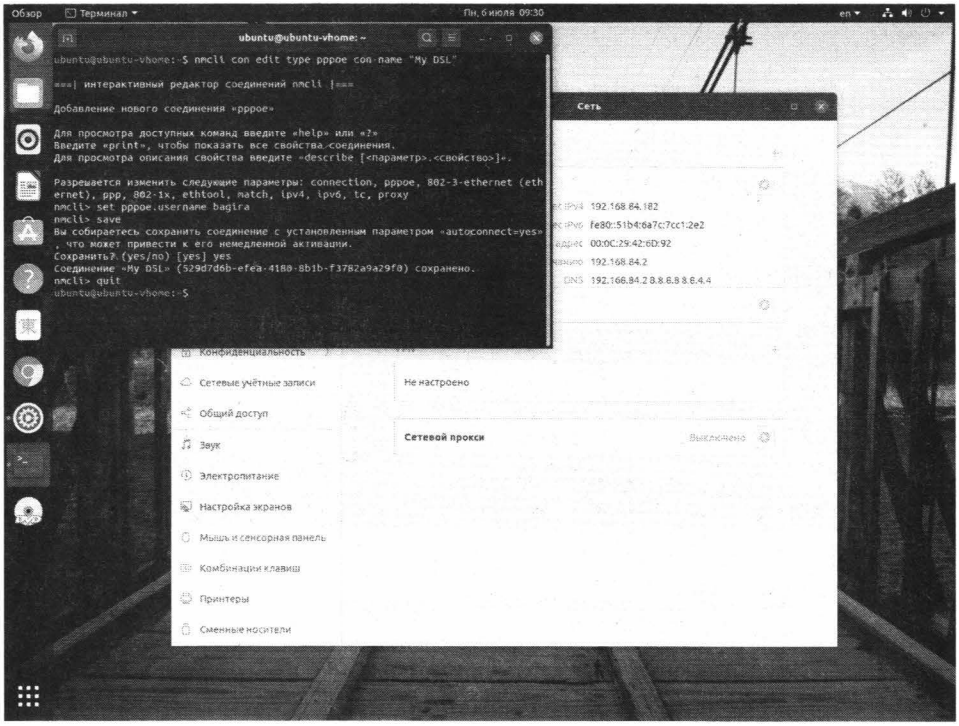


Рис. 9.1. Процесс создания DSL-соединения

1. Нажмите Ctrl + Alt + T, чтобы быстро открыть терминал.
2. Введите команду `nmcli con edit type pppoe con-name "My DSL"`
3. В появившемся приглашении введите команду `pppoe.username Имя`, где Имя нужно заменить на имя пользователя, полученное от вашего провайдера.
4. Введите команду **save**, а затем команду **exit**.

Процесс создания соединения показан на рис. 9.1. В процессе будет создан конфигурационный файл /etc/NetworkManager/system-connections/My DSL.nmconnection, содержимое которого показано на рис. 9.2. Если соединение не требует каких-либо специальных параметров, то в секции [pppoe] добавьте параметр password и соединение готово к использованию:

```
[pppoe]
password=Пароль
username=Имя
```

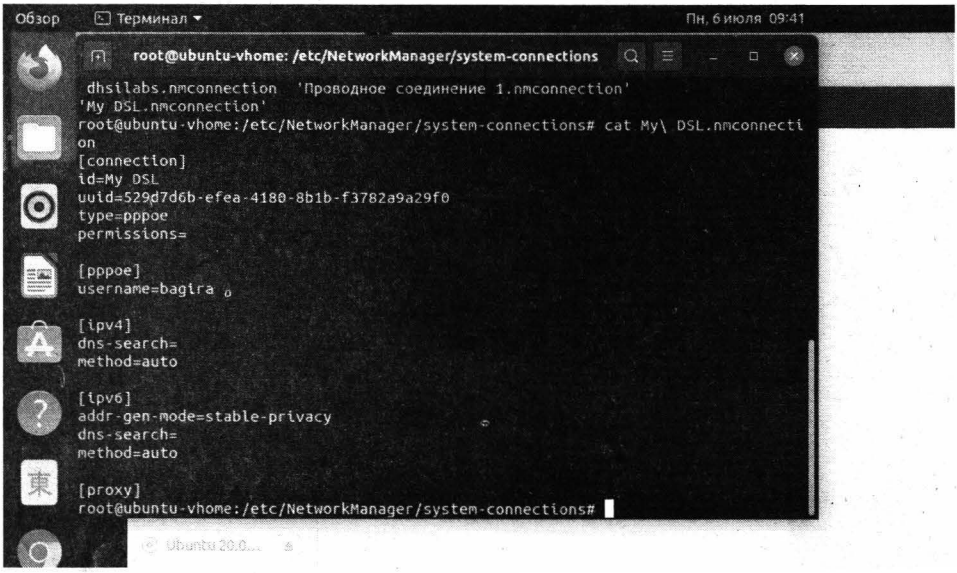


Рис. 9.2. Содержимое файла конфигурации соединения

В результате выполнения этой команды в разделе **Сеть** появится созданное нами соединение (рис. 9.3). Далее, нажав шестеренку, вы можете отредактировать другие параметры соединения (рис. 9.4)

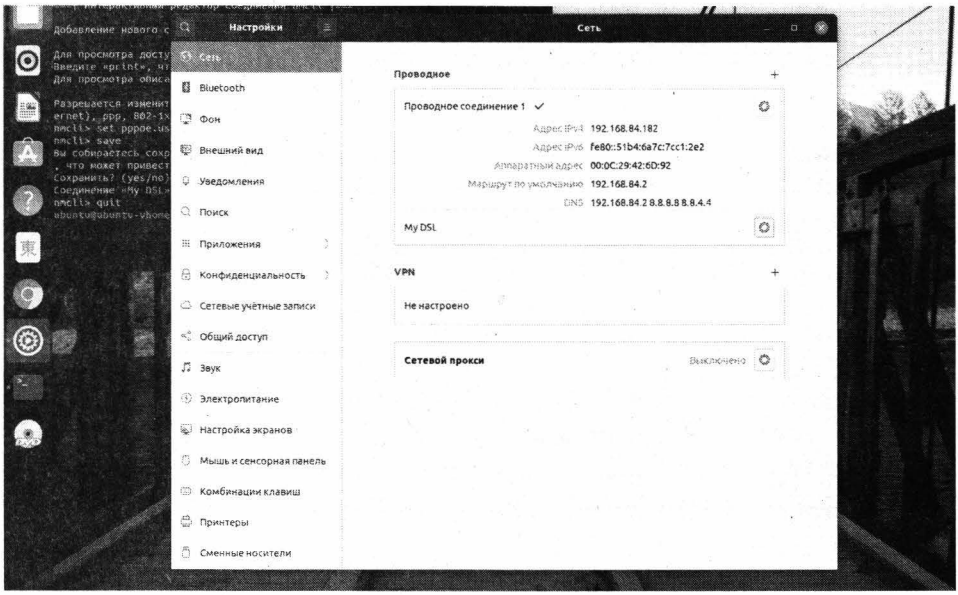


Рис. 9.3. Раздел Сеть

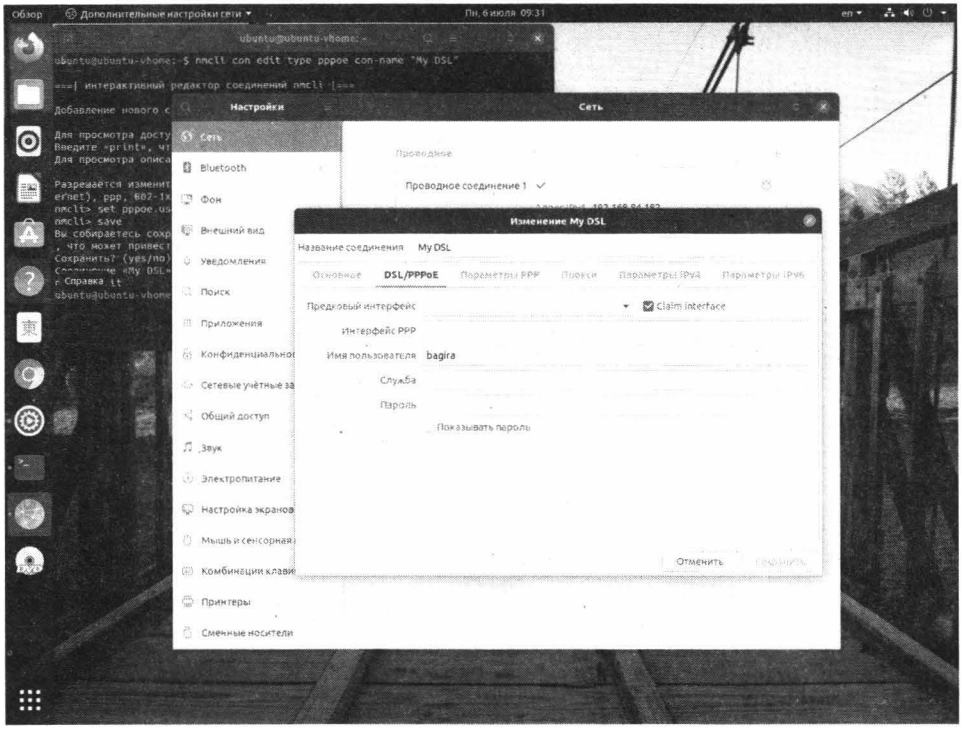


Рис. 9.4. Редактирование параметров соединения

Для подключения просто щелкните по соединению в разделе Сеть. Соединение будет установлено, если его параметры (как минимум имя пользователя и пароль) установлены правильно.

В более старых версиях Ubuntu и других Debian/Ubuntu-совместимых дистрибутивах, в том числе Astra Linux, используется приложение nm-connection-editor – редактор соединений Network Manager. Соединение PPPoE (DSL) настраивается следующим образом:

1. Нажмите комбинацию клавиш Alt + F2 и введите команду nm-connection-editor.
2. В окне **Сетевые соединения** (рис. 9.5) нажмите кнопку **+**.
3. Выберите тип соединения – **DSL/PPPoE** и нажмите кнопку **Создать** (рис. 9.6).
4. В появившемся окне (рис. 9.7) введите имя пользователя и пароль. В поле **Service** ничего вводить не нужно. Обязательно выберите **Parent interface** – это родительский интерфейс, по которому будут проходить PPPoE-пакеты. Поскольку у нас PPPoE – это PPP over Ethernet,

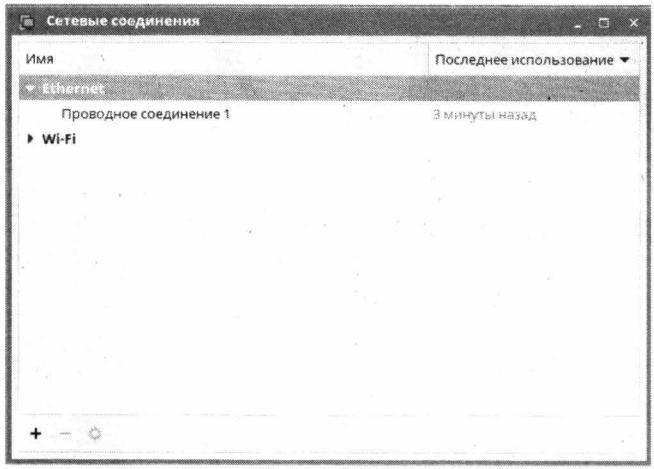


Рис. 9.5. Сетевые соединения

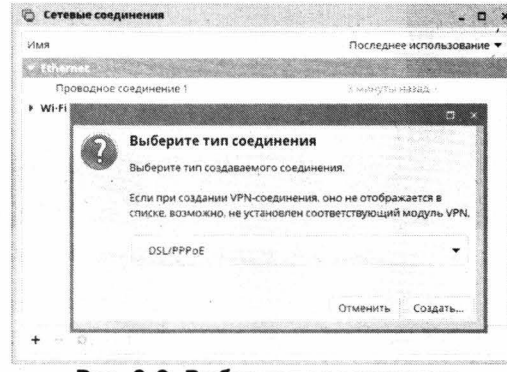


Рис. 9.6. Выбор типа соединения

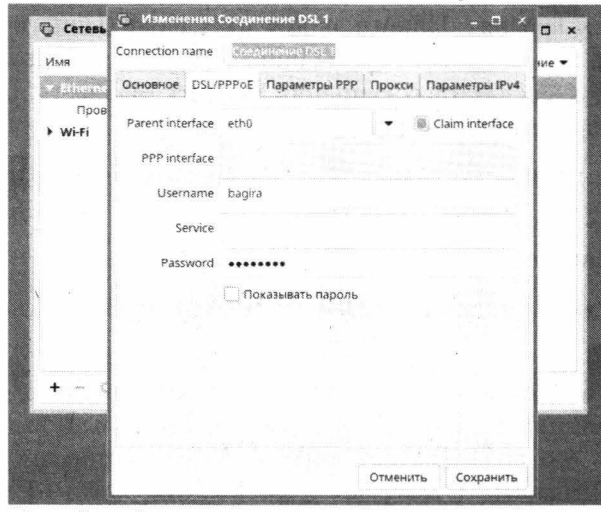


Рис. 9.7. Основные параметры PPPoE-соединения

то нужно выбрать сетевой интерфейс Ethernet, к которому подключен DSL-модем. Если у вас одна сетевая карта, выберите eth0.

5. Перейдите на вкладку **Основные**.

6. Если вы хотите, чтобы соединение было доступно всем пользователям, включите флажок **Все пользователи могут подключаться к этой сети** на вкладке **Основные**.

7. Если вы хотите, чтобы соединение устанавливалось автоматически, включите флажок **Connect automatically with priority** и выберите приориет соединения. Да, локализация Astra Linux оставляет желать лучшего, хотя позиционируется он как отечественный дистрибутив.

8. На вкладке **Параметр PPP** ничего интересного. Можно разве что отключить неиспользуемые методы аутентификации, хотя не нужно думать, что если вы это сделаете, ваше соединение заработает лучше.

9. На вкладке **Параметры IPv4** можно указать статический IP-адрес соединения. Обычно соединения настраиваются по протоколу PPPoE, но если у вашего провайдера другие мысли на этот счет, можно выбрать значение **Вручную** из списка **Method** и указать IP-адрес, маску сети и шлюз по умолчанию, а также серверы DNS и домены поиска (рис. 9.8).

10. Если у вас несколько DSL-подключений к разным провайдерам, тогда введите название соединения в верхней части экрана. В противном случае его можно оставить без изменений.

11. Нажмите кнопку **Сохранить**

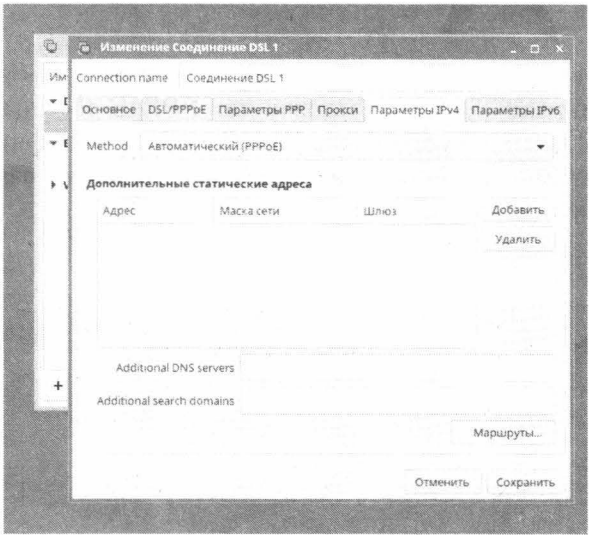


Рис. 9.8. Параметры IPv4

9.3. Программа pppoeconf: настройка DSL-соединения на сервере без графического интерфейса

В дистрибутивах Debian и Ubuntu (кроме версии 20.04), а также во многих других, основанных на Debian, имеется утилита pppoeconf, которая представляет собой псевдографический конфигуратор PPPoE-соединения.

Единственный неприятный момент, связанный с использованием этого конфигуратора - то, что по умолчанию он не установлен (во всяком случае, в Debian 7, в Ubuntu эта программа установлена по умолчанию) это так. Для его установки введите команду:

```
sudo apt install pppoeconf
```

Не беспокойтесь! Даже если у вас в данный момент нет графического интерфейса (вы его не устанавливали, поскольку настраиваете сервер, следовательно, вы не можете запустить NetworkManager для настройки DSL) и нет соединения с Интернетом (поскольку у вас DSL-подключение, которое вам нечем пока что настроить), программа все равно будет установлена. Она имеется на установочном DVD-диске Debian и будет загружена с его репозитария.

После установки программы выполните команду:

```
# pppoeconf
```

Запустите конфигуратор, который или предложит выбрать Ethernet-интерфейс, который будет использован для PPPoE-доступа или сообщит, что найдено только одно устройство (рис. 9.9). Далее конфигуратор начнет поиск активных PPPoE-концентраторов через выбранное вами устройство. Придется немного подождать.

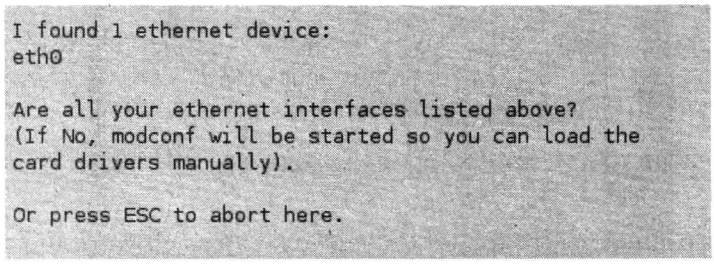


Рис. 9.9. Найдено только одно Ethernet-устройство

Следующий шаг - программа предложит вам добавить в файл конфигурации опции `noauth` (вы не будете требовать, чтобы сервер аутентифицировался) и `defaultroute` (без назначения маршрута по умолчанию доступа к Интернету не будет и вам все равно придется редактировать таблицу маршрутизации, но делать это нужно будет вручную!). Отказываться от добавления этих опций не стоит (рис. 9.10).

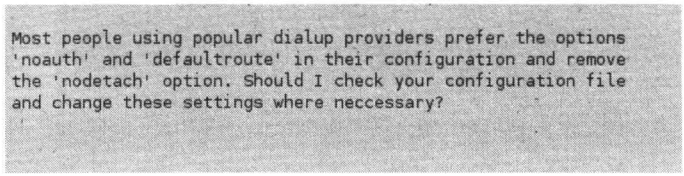


Рис. 9.10. Выберите Yes

После этого нужно ввести имя пользователя и пароль, необходимые для подключения к вашему Интернет-провайдеру

Сервер может сконфигурировать систему клиента, в том числе назначить ей IP-адреса DNS-серверов. Нужно ли автоматически внести эти IP-адреса в /etc/resolv.conf? Думаю, не нужно отказываться от этой возможности (рис. 9.11).

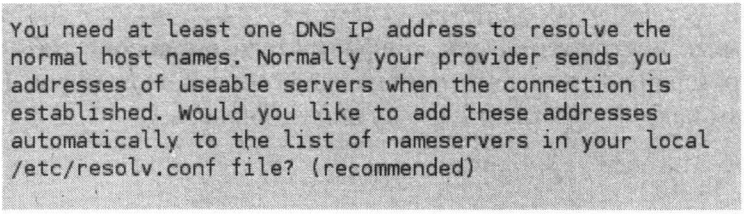


Рис. 9.11. Внести IP-адреса DNS-серверов в /etc/resolv.conf?

Далее будет традиционный вопрос о размере MSS. Нужно выбрать **Yes**, особенно, если вы не уверены в своих действиях и не понимаете, чего от вас хочет конфигуратор (рис. 9.12).

Предпоследний вопрос - нужно ли устанавливать соединение при загрузке системы. Если трафик у вас неограниченный (а в большинстве случаев оно так и есть), то удобнее устанавливать соединение именно при загрузке.

Наконец, конфигуратор предложит подключиться к Интернету - не стоит себе в этом отказываться.

Если у вас возникнет необходимость вручную управлять соединением, то установить соединение можно так:

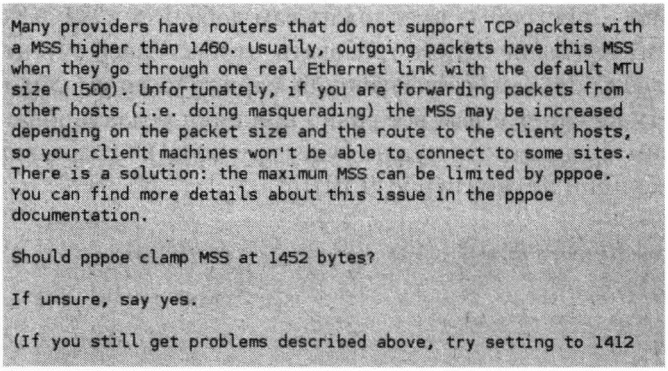


Рис. 9.12. Опять выберите Yes

```
# pon dsl-provider
```

Разорвать соединение можно командой poff:

```
# poff dsl-provider
```

9.4. Программа pppoe-setup

В некоторых дистрибутива нет конфигуратора **pppoeconf**. Зато имеется конфигуратор pppoe-setup, который находится в пакете rp-pppoe. По умолчанию этот пакет отсутствует и его нужно установить:

```
sudo apt install rp-pppoe
```

Далее запустите конфигуратор pppoe-setup. Здесь нет даже псевдографического интерфейса. Конфигуратор просто задает вам ряд вопросов, на которые вы должны ответить. Далее будет приведен вывод конфигуратора, а ответы на вопросы будут выделены жирным (то, что вы должны ответить), мои комментарии - курсивом

```
sudo pppoe-setup

Welcome to the PPPoE client setup.  First, I will run some checks on
your system to make sure the PPPoE client is installed properly...

LOGIN NAME
Enter your Login Name (default root): <введите ваш логин>
INTERFACE
```

Далее нужно ввести имя сетевого интерфейса для PPPoE-подключения. Если у вас одна плата и вы перешли на классические имена, можете ввести eth0. Если вы используете новую схему именования интерфейсов или у вас несколько сетевых карт, введите ifconfig в другом терминале, чтобы узнать правильное имя интерфейса.

```
Enter the Ethernet interface connected to the PPPoE modem
For Solaris, this is likely to be something like /dev/hme0.
For Linux, it will be ethX, where 'X' is a number.
(default eth0): eth0
```

Если ответить no, соединение будет доступно постоянно. Если указать количество секунд простоя, то по их истечению, соединение будет разорвано.

```
Do you want the link to come up on demand, or stay up continuously?
If you want it to come up on demand, enter the idle time in seconds
after which the link should be dropped.  If you want the link to
stay up permanently, enter 'no' (two letters, lower-case.)
NOTE: Demand-activated links do not interact well with dynamic IP
addresses.  You may have some problems with demand-activated links.
Enter the demand value (default no): no
```

DNS

Далее нужно ввести IP-адреса первичного и вторичного DNS-серверов

```
Please enter the IP address of your ISP's primary DNS server.
If your ISP claims that 'the server will provide dynamic DNS
addresses',
enter 'server' (all lower-case) here.
If you just press enter, I will assume you know what you are
doing and not modify your DNS setup.
Enter the DNS information here: 8.8.8.8
Please enter the IP address of your ISP's secondary DNS server.
If you just press enter, I will assume there is only one DNS server.
Enter the secondary DNS server address here: 8.8.4.4
```

PASSWORD

Введите пароль для соединения и повторите ввода

```
Please enter your Password: <ваш пароль>
Please re-enter your Password: <ваш пароль>
```

USERCTRL

Можно или нет обычному пользователю запускать и останавливать DSL-соединение? На сервере имеет смысл запретить это, поэтому выбираем no. А на обычной рабочей станции для настройки PPPoE вы вряд ли будете использовать этот конфигуратор.

```
Please enter 'yes' (three letters, lower-case.) if you want to allow
normal user to start or stop DSL connection (default yes): no

FIREWALLING

Please choose the firewall rules to use.  Note that these rules are
very basic.  You are strongly encouraged to use a more sophisticated
firewall setup; however, these will provide basic security.  If you
are running any servers on your machine, you must choose 'NONE' and
set up firewalling yourself.  Otherwise, the firewall rules will deny
access to all standard servers like Web, e-mail, ftp, etc.  If you
are using SSH, the rules will block outgoing SSH connections which
allocate a privileged source port.

The firewall choices are:
0 - NONE: This script will not set any firewall rules.  You are
responsible
          for ensuring the security of your machine.  You are
STRONGLY
          recommended to use some kind of firewall rules.
1 - STANDALONE: Appropriate for a basic stand-alone web-surfing
workstation
2 - MASQUERADE: Appropriate for a machine acting as an Internet
gateway
                for a LAN
```

Укажите тип брандмауэра, просто введите пока 0, чтобы конфигуратор не устанавливал никаких правил брандмауэра. Если вы настраиваете шлюз, укажите 2. Если вы настраиваете обычную рабочую станцию и Интернетом будете пользоваться только вы, тогда введите 1. Поскольку брандмауэр будем настраивать только в следующей главе, пока введите 0.

```
Choose a type of firewall (0-2): 0

Start this connection at boot time
```

Устанавливать ли соединение при загрузке системы?

```
Do you want to start this connection at boot time?
```

```
Please enter no or yes (default no):yes

Сводка по указанным вам данным

** Summary of what you entered **

Ethernet Interface: eth0
User name:          user101
Activate-on-demand: No
Primary DNS:        8.8.8.8
Secondary DNS:      8.8.4.4
Firewalling:        NONE
User Control:       none

Сохранить изменения?

Accept these settings and adjust configuration files (y/n)? y
Adjusting /etc/sysconfig/network-scripts/ifcfg-ppp0
Adjusting /etc/resolv.conf
  (But first backing it up to /etc/resolv.conf.bak)
Adjusting /etc/ppp/chap-secrets and /etc/ppp/pap-secrets
  (But first backing it up to /etc/ppp/chap-secrets.bak)
  (But first backing it up to /etc/ppp/pap-secrets.bak)

Congratulations, it should be all set up!

Type '/sbin/ifup ppp0' to bring up your xDSL link and '/sbin/ifdown
ppp0'
to bring it down.
Type '/sbin/pppoe-status /etc/sysconfig/network-scripts/ifcfg-ppp0'
to see the link status.
```

Внимательно прочитайте все, что сообщит вам конфигуратор после сохранения настроек. Он сообщает, что вся конфигурация будет записана в файл /etc/sysconfig/network-scripts/ifcfg-ppp0. Также будет изменен файл /etc/resolv.conf - в него будет внесена информация DNS, а старая версия этого файла будет называться /etc/resolv.conf.bak.

Имя пользователя и пароль будут добавлены в файлы /etc/ppp/chap-secrets и /etc/ppp/pap-secrets (методы аутентификации CHAP и PAP используются наиболее часто).

Чтобы вручную установить PPPoE-подключение, введите команду:

```
sudo /sbin/ifup ppp0
```

```
firewall setup; however, these will provide basic security.  If you
are running any servers on your machine, you must choose 'NONE' and
set up firewalling yourself.  Otherwise, the firewall rules will deny
access to all standard servers like Web, e-mail, ftp, etc.  If you
are using SSH, the rules will block outgoing SSH connections which
allocate a privileged source port.

The firewall choices are:
0 - NONE: This script will not set any firewall rules.  You are responsible
          for ensuring the security of your machine.  You are STRONGLY
          recommended to use some kind of firewall rules.
1 - STANDALONE: Appropriate for a basic stand-alone web-surfing workstation
2 - MASQUERADE: Appropriate for a machine acting as an Internet gateway
                for a LAN
Choose a type of firewall (0-2): 0

Start this connection at boot time

Do you want to start this connection at boot time?
Please enter no or yes (default no):

** Summary of what you entered **

Ethernet Interface: eth0
User name:          user101
Activate-on-demand: No
Primary DNS:        0.0.0.0
Secondary DNS:      0.0.0.1
Firewalling:        NONE
User Control:       no
Accept these settings and adjust configuration files (y/n)? y
Adjusting /etc/sysconfig/network-scripts/ifcfg-ppp0
Adjusting /etc/resolv.conf
  (But first backing it up to /etc/resolv.conf.bak)
Adjusting /etc/ppp/chap-secrets and /etc/ppp/pap-secrets
  (But first backing it up to /etc/ppp/chap-secrets.bak)
  (But first backing it up to /etc/ppp/pap-secrets.bak)


Congratulations, it should be all set up!

Type '/sbin/ifup ppp0' to bring up your xDSL link and '/sbin/ifdown ppp0'
to bring it down.
Type '/sbin/pppoe-status /etc/sysconfig/network-scripts/ifcfg-ppp0'
to see the link status.

[root@localhost ~]#
```

Рис. 9.13. Процесс настройки PPPoE с помощью pppoe-setup

Чтобы вручную разорвать PPPoE-подключение, используется другая команда:

```
/sbin/ifdown ppp0
```

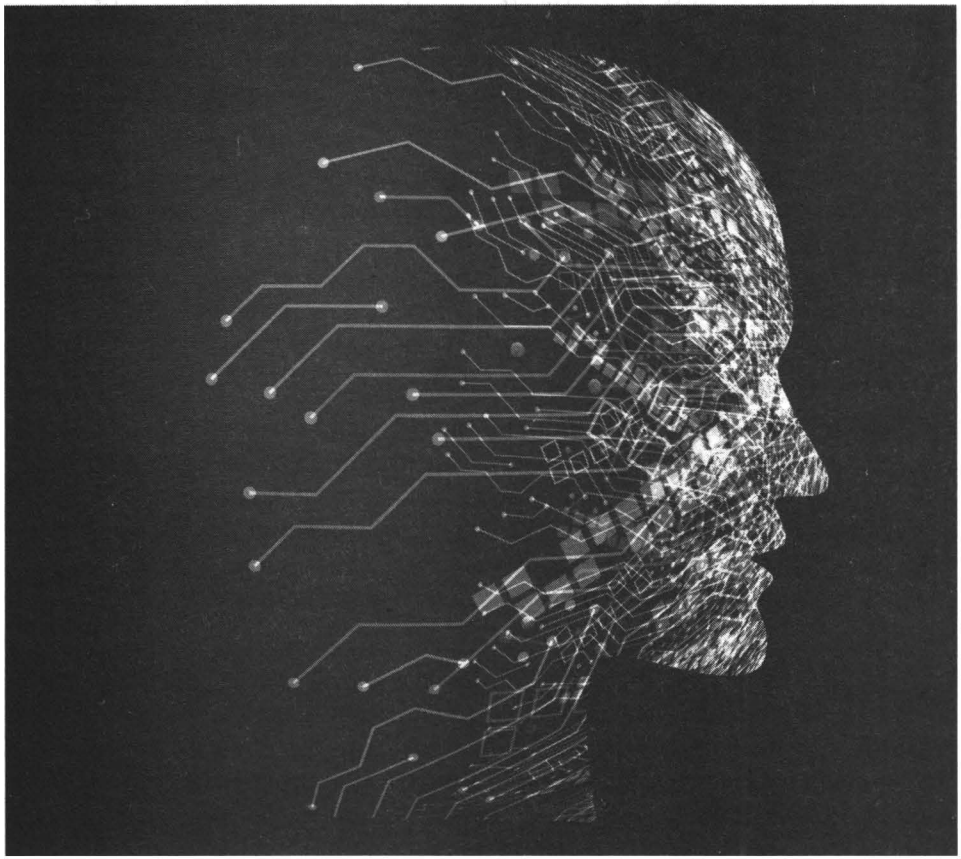
Просмотреть состояние вашего соединения можно с помощью команд:

```
/sbin/pppoe-status
/etc/sysconfig/network-scripts/ifcfg-ppp0
```

На этом все, а в следующих главах будут рассмотрены установка программ и настройка брандмауэра в Linux.

Глава 10.

Установка программ в Linux



10.1. Способы установки программ

Существует три способа установки программ: компиляция из исходных кодов, из пакетов, из снапов. Первый способ устарел и сейчас используется очень редко – когда нет другого способа установить ту или иную программу на свой компьютер. Выглядит все это так: вы сначала устанавливаете программное обеспечение разработчика (заголовочные файлы, компилятор **gcc**, библиотеки, различные утилиты вроде **make** и т.д.), затем загружаете и распаковываете архив с исходными кодами программы. Запускаете компиляцию и ждете, пока программа будет «собрана» из исходников. Даже если все пройдет гладко и вам не придется адаптировать исходные коды (что требует определенных знаний) под свою систему, процесс компиляции занимает много времени. Удалить «установленную» таким образом программу можно только вручную.

Сегодня все мыслимое и немыслимое программное обеспечение поставляется в уже откомпилированном виде. Самостоятельная сборка (компиляция) программного обеспечения имеет смысл только в случаях, если вам нужно установить программу, когда для вашей архитектуры[1]/дистрибутива нет пакета или же когда вы хотите получить самую последнюю версию программы - когда разработчики еще не успели создать пакет для вашего дистрибутива.

Установка из пакета гораздо проще. Вы получаете пакет с программой и устанавливаете его. Пакет содержит уже откомпилированную программу и необходимые для работы этой программы файлы, например, файлы конфигурации, страницы руководства и др. При удалении пакета вся эта информация будет централизована и удалена и вам не придется исследовать каждый каталог системы и вручную удалять остатки программы.

Максимум, что может пойти не так при установке пакета – это нарушение зависимостей. Об этом мы поговорим далее в этой главе. Забегая наперед, обычно для удовлетворения зависимостей нужно удалить или установить какие-то другие пакеты. Это несложно и в некоторых случаях система сама справляется с поставленной задачей. Установка программы из пакета занимает считанные секунды, в крайнем случае – минуты, большая часть времени уходит на загрузку самого пакета из Интернета.

1 Хотя есть большая вероятность того, что раз для вашей архитектуры нет пакета, то у вас не получится откомпилировать программу

Получить пакет можно, как вручную, собственноручно скачав его из сайта разработчика, так и воспользоваться менеджером установки пакетов, который сам скачает пакет из репозитария и установит его.

Третий способ, снапы – относительно недавний. Как уже было отмечено, при установке пакета могут понадобиться дополнительные пакеты. Как правило, это пакеты с какими-то библиотеками. Иногда происходит так, что нужная библиотека отсутствует в вашем дистрибутиве как таковая или конфликтует с имеющейся версией библиотеки. Особенно часто такое происходит, если вы пытаетесь установить пакет не из репозитария своего дистрибутива, а скачанный с сайта разработчика. Чтобы избавиться от подобных танцев с бубном, были придуманы снапы. Снап содержит свою программу и все необходимые для ее работы библиотеки. При этом программа, установленная со снапа, будет использовать не системную версию библиотеки, а ту версию, которая шла со снапом. В результате программа будет выполняться корректно и ее установка никак не отразится на других программах, поскольку процесс установки никак не затрагивает имеющиеся в системе библиотеки. О снапах мы еще поговорим отдельно в этой главе.

10.2. Типы пакетов и их содержимое

Существует два формата пакетов – DEB (расширение .deb) и RPM (.rpm). Первые используются в Debian-совместимых дистрибутивах – Ubuntu, Astra Linux, Mint, Denix и др. Второй тип используется в RedHat-совместимых – RHEL, Fedora, CentOS, openSUSE и т.д.

Независимо от формата пакетов в самом пакете кроме устанавливаемой программы и вспомогательных файлов содержится различная служебная информация: информация о разработчике, о версии программного продукта, информация о зависимостях и конфликтах, пути для установки (указывают, куда должны быть скопированы файлы, имеющиеся в пакете в процессе установки пакета).

Отдельного разговора заслуживает информация о зависимостях и конфликтах. Некоторые пакеты для своей работы требуют установки дополнительных пакетов. Например, пакет ubuntu-desktop требует для своей работы множества других пакетов – alsa-base (файлы звукового драйвера), alsa-utils, bc, anacron и т.д. Говорят, что пакет ubuntu-desktop зависит от пакетов alsa-base, alsa-utils и т.д. В свою очередь эти пакеты могут зависеть от других пакетов. При установке пакета ubuntu-desktop менеджер пакетов выполнит разрешение зависимости, то есть установит все пакеты, от которых зависит наш пакет и все пакеты, от которого зависят другие устанав-

ливаемые пакеты. Может получиться, что для установки одной небольшой программы будет установлено очень много других пакетов, от которых зависит эта программа. Но ничего не поделаешь – если программа вам нужна, то придется пойти на это. Как вариант – найти аналогичную программу.

Некоторые программы могут находиться в системе только в единственном экземпляре. Например, почтовый агент может быть установлен только один, иначе между ними произойдет конфликт. Поэтому пакет А может конфликтовать с пакетом Б. Если у вас установлен А, а вы пытаетесь установить Б, менеджер пакетов сообщит о конфликте и предложит или отказаться от установки или перед установкой пакета Б удалить пакет А.

10.3. Источники пакетов

Список источников пакетов в порядке убывания их популярности:

- **Репозитарии** - каждый дистрибутив работает с собственным репозитарием (хранилищем) пакетов. Да, для установки программного обеспечения вам понадобится доступ к Интернету, но это самый современный способ установки. Репозитарии позволяют эффективно управлять обновлением программного обеспечения. Используя репозитарии, вы можете быть уверенными, что у вас будет установлено самое новое программное обеспечение.
- **Установочный диск** - ранее дистрибутивы Linux распространялись на DVD-дисках (некоторые - на одном, некоторые - на нескольких). На DVD, кроме самой системы, была и львиная доля программного обеспечения, которая только может понадобиться пользователю. Все программное обеспечение при установке системы не устанавливалась, но была возможность установки с инсталляционного носителя после установки. Когда доступ к Интернету был медленным и дорогим, пользователи предпочитали устанавливать пакеты с DVD-диска. Преимущество этого способа в том, что не нужно загружать пакеты с удаленного сервера, что экономило время и деньги. Сейчас же доступ к Интернету в большинстве случаев высокоскоростной (даже 10 Мбит/с считается высокоскоростным) и безлимитный, поэтому чаще всего на DVD-диске содержится только самое необходимое программное обеспечение, а все остальное загружается из Интернета. К тому же, ПО с DVD наверняка устареет к моменту его установки, а из репозитария будет загружена более новая версия – пакеты в репозитариях периодически обновляются.

- **Сайт разработчика программного обеспечения** - иногда программное обеспечение не включено в состав репозитария дистрибутива. Это может произойти по разным причинам. Например, программное обеспечение является проприетарным (коммерческим) или же просто узкоспециализированным и его не стали включать в состав дистрибутива. В этом случае вам нужно самостоятельно скачать пакеты и установить их. В некоторых случаях разработчики предоставляют доступ к собственным репозитариям, из которых можно загрузить все необходимые пакеты.

10.4. Менеджеры пакетов

Изначально в RedHat и Debian использовались программы **rpm** и **dpkg**. Сейчас обе эти программы все еще входят в состав дистрибутивов, но использовать их крайне не рекомендуется. Дело в том, что эти программы ничего не «знают» о зависимостях между пакетами. Если для работы пакета А нужен пакет Б, то программа даже толком не сообщит о том, какой пакет нужен для установки пакета А. Она сообщит о том, что для установки пакета А нужна, например, библиотека lib40, а в каком DEB-пакете находится эта библиотека и откуда ее нужно заполучить - вам придется догадаться самому.

Также программы rpm и dpkg ничего не знают о репозиториях, поэтому удел этого менеджера - установка пакета из локального источника. Желательно, чтобы для работы устанавливаемого пакета не требовались другие пакеты - ведь о зависимостях эти программы, как уже было отмечено, ничего не знают. В репозиториях содержится информация обо всем дереве зависимостей. Например, пакет А может зависеть от пакета Б, пакет Б - от пакетов В, Г и Д. Менеджеры пакетов, такие как apt установят все необходимые пакеты в необходимом порядке, чего не дождешься от программ rpm и dpkg.

Именно поэтому для управления пакетами рекомендуется использовать менеджеры пакетов. Вот неполный список возможностей таких программ:

- Поиск пакетов в репозиториях;
- Установка пакетов из репозиториев;
- Установка/удаление пакетов с разрешением зависимостей и конфликтов;
- Обновление пакетов.

Принцип работы менеджера пакетов простой. Представим, что вы устанавливаете пакет А. Менеджер производит список вашего пакета по списку

всех установленных репозитариев. В одном из репозитариев ваш пакет точно будет, в противном случае менеджер сообщит о том, что пакет не найден. Далее менеджер смотрит на список зависимостей и если есть пакеты, от которых зависит наш пакет А, то они также загружаются и устанавливаются - до установки пакета А. Затем производится загрузка и установка самого пакета А.

Установка программного обеспечения в Linux требует прав root. Поэтому нужно сначала запускать любые команды управления программным обеспечением через команду sudo. Например:

```
sudo apt install mc
```

В дистрибутивах Debian, Ubuntu, Astra Linux и других, которые основаны на Debian, используется менеджер пакетов apt. В других дистрибутивах могут использоваться другие менеджеры, например, в Fedora используется dnf, в CentOS и старых версиях Fedora – yum. В openSUSE используется свой менеджер пакетов zypper.

Далее будет рассмотрен менеджер пакетов apt. При установке пакетов apt разрешает зависимости. Вы просто указываете, какой пакет вы хотите установить, а что будет происходить дальше - уже не ваша забота. Вас интересует конечный результат. Лишь бы требуемый пакет был в репозитариях.

> Примечание. В старых версиях Debian и Ubuntu использовалась команда apt-get вместо apt. Основные параметры (см. табл. 10.1) у этих двух команд такие же, нет смысла рассматривать устаревшую версию менеджера пакетов. Если вам нужно работать со старой версией, обратитесь к странице руководства (man apt-get).

Список репозитариев хранится в файле /etc/apt/sources.list. Содержимое этого файла показано на рис. 10.1.

Вы вряд ли будете вручную редактировать этот файл. Исключения могут составить лишь ситуации, когда вы добавляете сторонний репозитарий, содержащий какую-то программу, чтобы установить ее впоследствии с помощью apt. Что же касается стандартных репозитариев, то ними проще управлять (включать/выключать) посредством приложения **Программы и обновления**, изображенном на рис. 10.2. На вкладке **Программное обеспечение Ubuntu** вы можете включить/выключить стандартные репозитарии. Также можно включить использование в качестве источника пакетов инсталляционный DVD-диск Ubuntu. Обратите внимание: вы можете не только включить/выключить репозитарии, но и выбрать местоположение сервера, с которого будут пакеты загружаться. На рис. 10.2 показано, что пакеты будут загружаться с сервера, который находится в РФ.

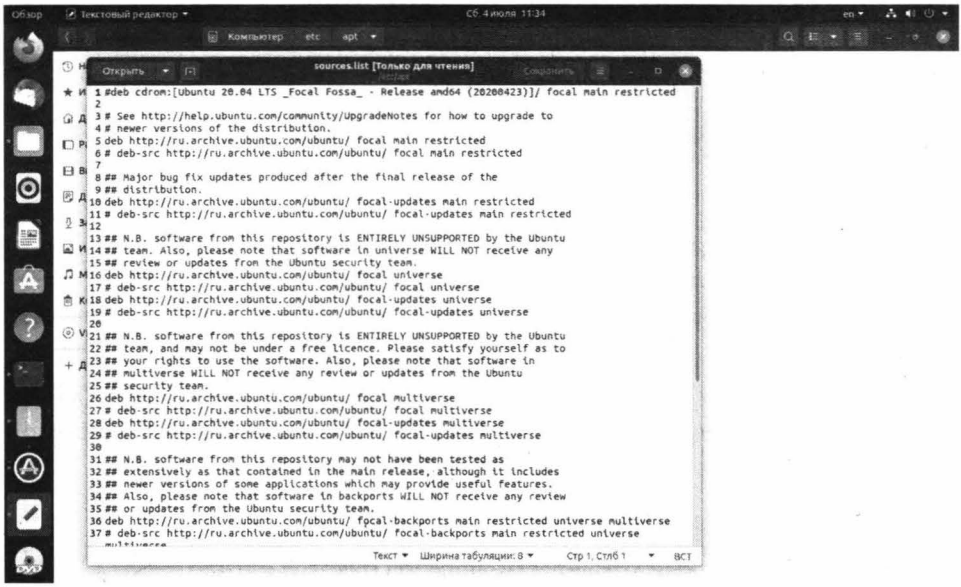


Рис. 10.1. Файл sources.list

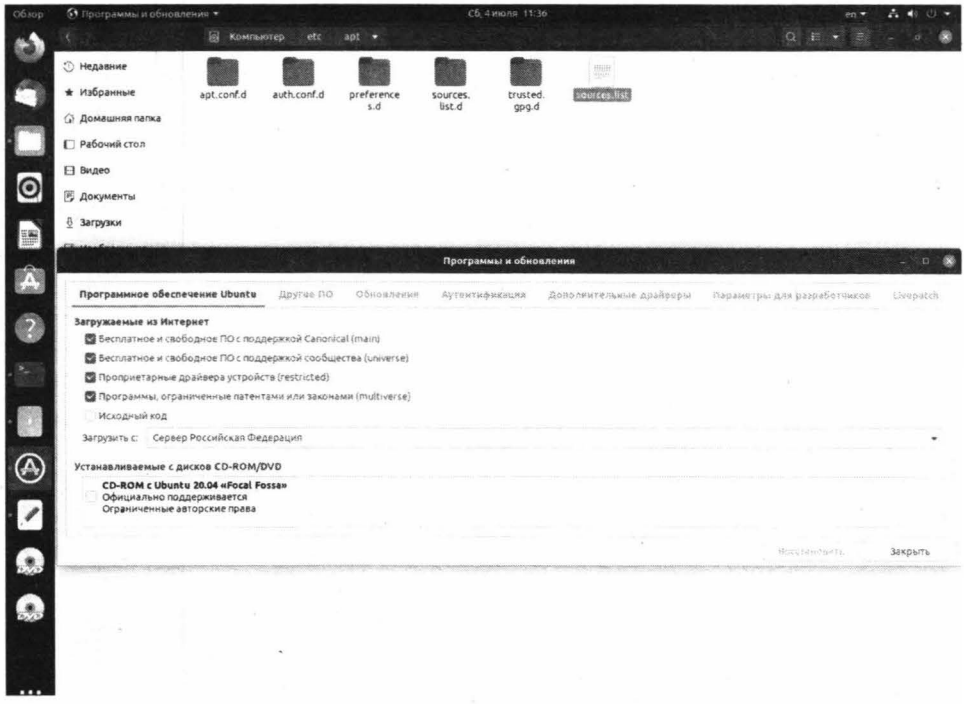


Рис. 10.2. Программы и обновления

Формат вызова команды **apt** следующий:

```
sudo apt [опции] команды [пакет]
```

В качестве примера приведу команду установки пакета **mc**:

```
sudo apt install synaptic
```

Обратите внимание, что я запускаю команду от имени root. Если запустить команду `apt` через команду sudo (sudo apt install synaptic), нужно, чтобы пользователь, который запускает sudo, был внесен в файл /etc/sudoers (см. гл. 17).

По умолчанию пользователь, которого вы создаете при установке Debian, не вносится в этот файл, поэтому нужно или получать права root командой **su**, или добавить пользователя в файл /etc/sudoers.

Основные команды `apt` приведены в таблице 10.1.

Таблица 10.1. Основные команды менеджера пакетов apt-get

Команда	Описание
install <список пакетов>	Устанавливает пакеты из списка. Элементы списка разделяются пробелами
remove <список пакетов>	Удаляет пакеты из списка. Элементы списка разделяются пробелами
purge <список пакетов>	Удаляет не только пакеты, но их конфигурационные файлы. Это означает, что если вы установили какую-нибудь программу, настроили ее, а потом удалили командой apt remove, то конфигурационный файл этой программы останется в системе. Если вы теперь установите эту программу снова, то можно будет использовать предыдущий конфигурационный файл, так как он не был удален
check	Поиск нарушенных зависимостей

clean	Очищает локальное хранилище полученных пакетов. При установке, пакеты из репозитария загружаются в каталог var/cache/apt/archive. При интенсивной установке программного обеспечения в этом каталоге накапливается довольно много пакетов, поэтому очистка хранилища помогает сэкономить дисковое пространство.
upgrade [список пакетов]	Обновляет указанные пакеты, если пакеты не заданы, обновляет все пакеты, требующие обновления
full-upgrade	Обновляет всю систему
update	Синхронизирует внутреннюю базу данных о пакетах с источниками пакетов, которые описаны в /etc/apt/sources.list
autoremove	Когда вы устанавливаете пакет, то часто устанавливаются дополнительные пакеты, являющиеся его зависимостями. Если теперь вы удалите этот пакет, то зависимости останутся в системе. Команда apt autoremove удаляет эти зависимости, но только те, которые не нужны другим установленным пакетам
list	Выводит список пакетов, соответствующих какому-то критерию. Примеры приведены далее. Вывести список установленных в системе пакетов: apt list --installed Вывести список пакетов, которые требуют обновления (у которых вышла новая версия): apt list --upgradable Вывести список всех пакетов доступных для вашей системы: apt list --all-versions

show <пакет>	Выводит информацию о пакете
search <слово>	Данная команда выполняет поиск указанного слова в названии пакетов и в описании пакетов. Поддерживаются регулярные выражения
edit-sources	Открывает файл /etc/apt/sources.list в текстовом редакторе для редактирования, после сохранения изменений и закрытия редактора, выполняет проверку файла на предмет ошибок. В случае наличия ошибок, выводит предложение на повторное редактирование файла, чтобы исправить ошибки

10.5. Графические средства установки программ

Установка программ из пакетов имеет свои недостатки. Самый главный из них – вам нужно знать имя пакета, в котором находится нужная вам программа. Если вы следуете какому-то руководству, то особых проблем нет – открыл терминал, ввел команду установки нужного пакета и все. Когда вы знаете имя пакета или команду установки, все хорошо. Но если таких познаний у вас нет, тогда вам нужно или найти имя пакета в Интернете или же воспользоваться графическими средствами для установки программ.

В состав дистрибутива Astra Linux входит, на наш взгляд, один из самых удобных графических менеджеров пакетов – Synaptic. Вы найдете его в программной группе **Системные**. Использовать Synaptic очень просто (рис. 10.3). Слева отображаются группы пакетов и фильтры (под группами пакетов), позволяющие отфильтровать список пакетов по разделам, состоянию, архитектуре и т.д. В правой верхней части находится список пакетов в выделенной группе. Если выделить любой из пакетов, в область ниже будет загружено его описание.

Установленные пакеты отмечаются зеленым квадратиком. У неустановленных квадратик неокрашенный. Чтобы установить пакет, щелкните на нем правой кнопкой мыши и выберите команду **Отметить для установки** (рис. 10.3). Значок пакета будет изменен на квадратик с желтой стрелкой. Можете выбрать другие пакеты, которые вы хотите установить. Когда будете готовы, нажмите кнопку **Применить** для установки пакетов (рис. 10.4).

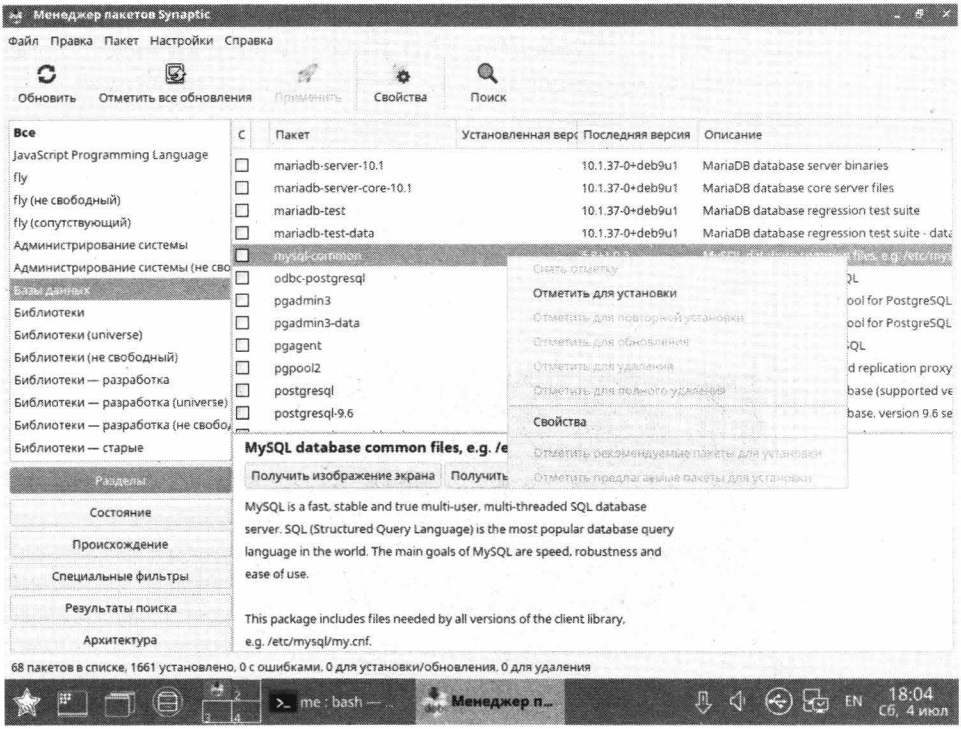


Рис. 10.3. Менеджер пакетов Synaptic

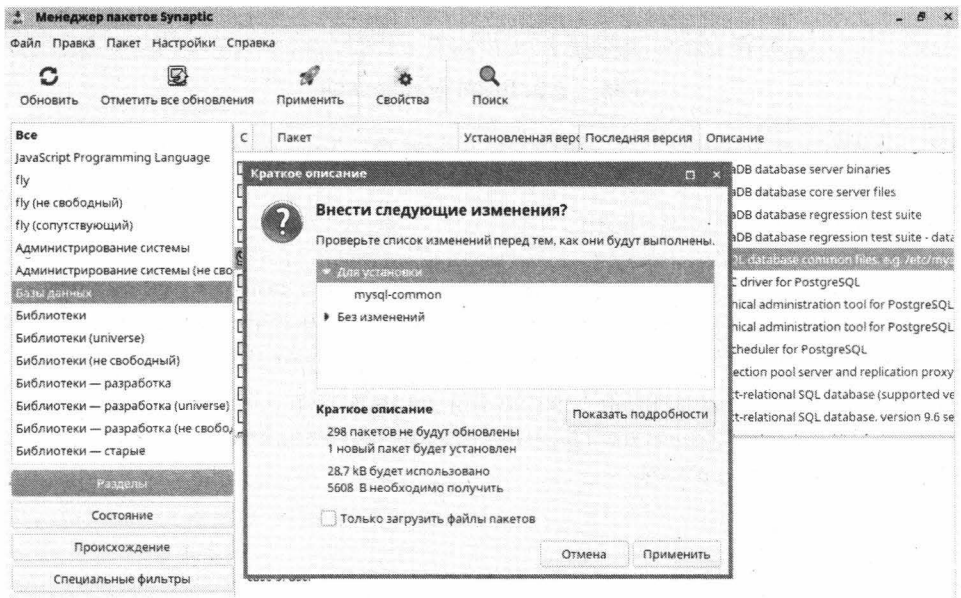


Рис. 10.4. Применение изменений

Менеджер пакетов покажет, какой размер будет загружен из Интернета и сколько пакетов будет установлено. Дождитесь установки пакета (рис. 10.5). После этого вы можете использовать установленную программу.

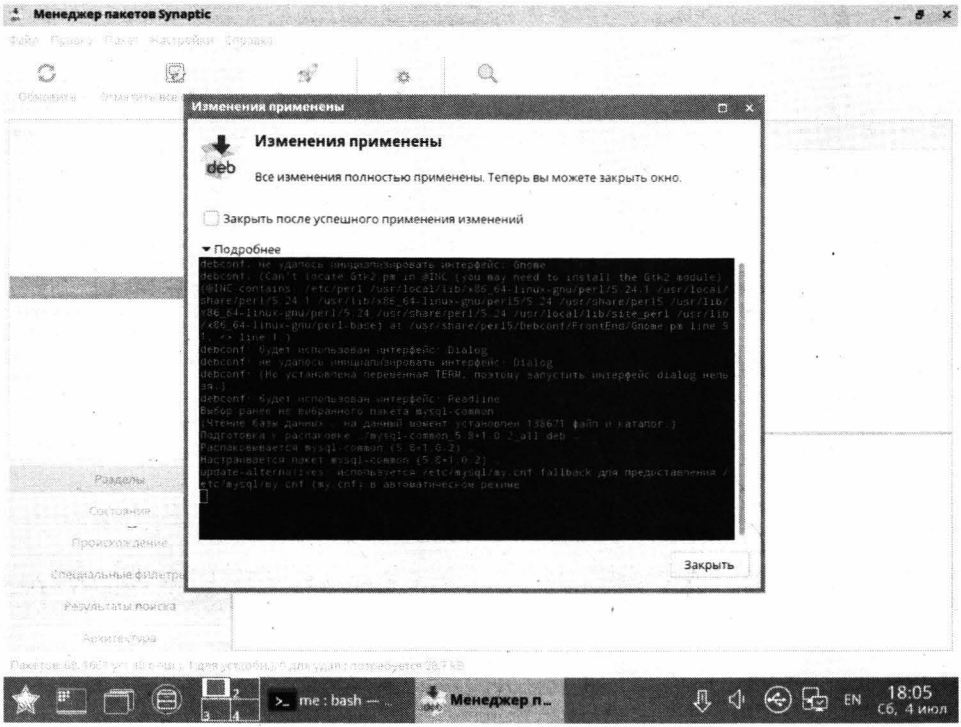


Рис. 10.5. Пакет установлен

В Ubuntu менеджер Synaptic по умолчанию недоступен. Для его установки нужно ввести команду:

```
sudo apt install synaptic
```

Однако в Ubuntu обычному пользователю больше понравится использовать Ubuntu Software или центр программного обеспечения. Все доступное ПО разбито на группы, вы можете выбрать программу и установить ее.

На вкладке **Установленные** находится список уже установленных программ. Напротив каждой программы есть кнопка **Удалить**, использующаяся для удаления программы.

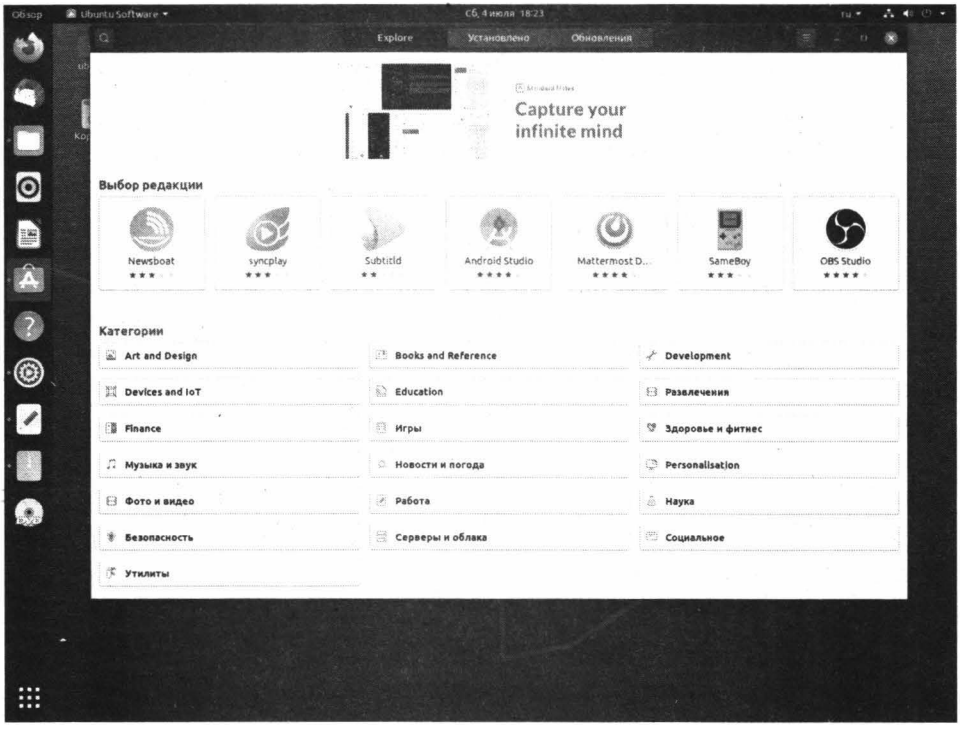


Рис. 10.6. Ubuntu Software

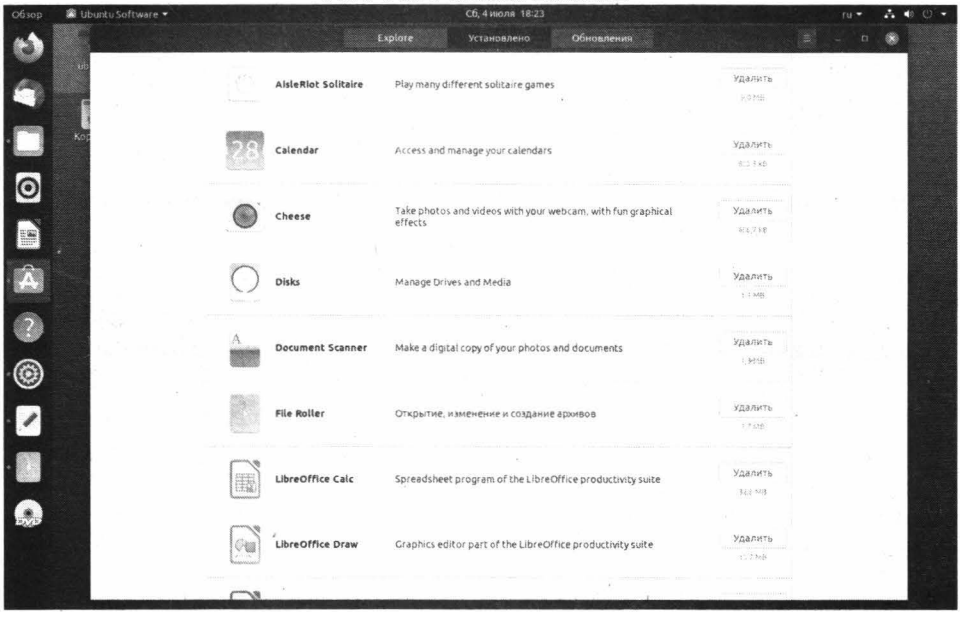


Рис. 10.7. Установленные программы

10.6. Снапы

Ранее уже было сказано, зачем используют снапы (snaps). Представим вполне реальную ситуацию. Пользователь устанавливает самую новую версию Ubuntu, пусть это будет версия 20.04. После установки пользователь не обновляет систему. Его все устраивает, и он спокойно работает некоторое время, скажем, год или полтора. Потом он хочет установить новую версию браузера или какого-то другого приложения и не может этого сделать, поскольку его дистрибутив устарел. 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 по ряду причин, да и это опасно: система может быть разрушена из-за нарушения связей между программами и библиотеками.

Есть и другая ситуация – когда нужно установить приложение, пакет которого конфликтует с уже установленным пакетом. Просто устанавливаемое приложение использует библиотеки, которые не могут «ужиться» с теми, которые уже установлены.

Проблемы настолько частые, то были предложены снапы. Пакет содержит саму программу, а также различные вспомогательные файлы – документацию, ресурсы (картинки, например), файлы локализации, какие-то сценарии и т.д. Но пакет не содержит всего, что нужно для работы этой программы в системе. Например, если программе для работы нужна библиотека **lib**, то просто в пакете «прописывается» зависимость – нужно установить такой-то пакет для работы этого пакета. 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, это порождает уже описанные ранее проблемы.

Снап – это решение всей головной боли, как пользователя, так и разработчика приложения. Снап можно считать таким пакетом, в котором содержится не только программа, но и все необходимые для ее работы библиотеки. Получается, что все, что нужно для работы программы содержится в снапе. Да, это неэкономно, но, согласитесь, жестким диском размером в 1 Тб сегодня никого не удивишь, а головной боли будет меньше. Вы просто установите программу и будете ее использовать, а не два дня пытаться решить проблемы, возникшие при установке ее пакета.

Также не нужно бояться, что система превратится в мусорку. Снапы устанавливаются в отдельной папке в виде защищенного от записи образа. Из-

менения, которые должен внести снап в файловую систему (например, в каталог /lib) будут внесены в виртуальной файловой системе. Таким образом, установка снапа никак не повлияет на работу основной системы и на другие снапы. Снапы решают проблемы с совместимостью версий разных приложений.

Для начала работы со снапами в старых версиях Ubuntu (например, в 16.04, в более ранних версиях они не поддерживаются вообще) нужно сначала установить пакет snapd. В современных версиях данный пакет уже установлен.

Для поиска доступных снапов введите команду:

```
snap find <название>
```

Например

```
snap find hello-world
```

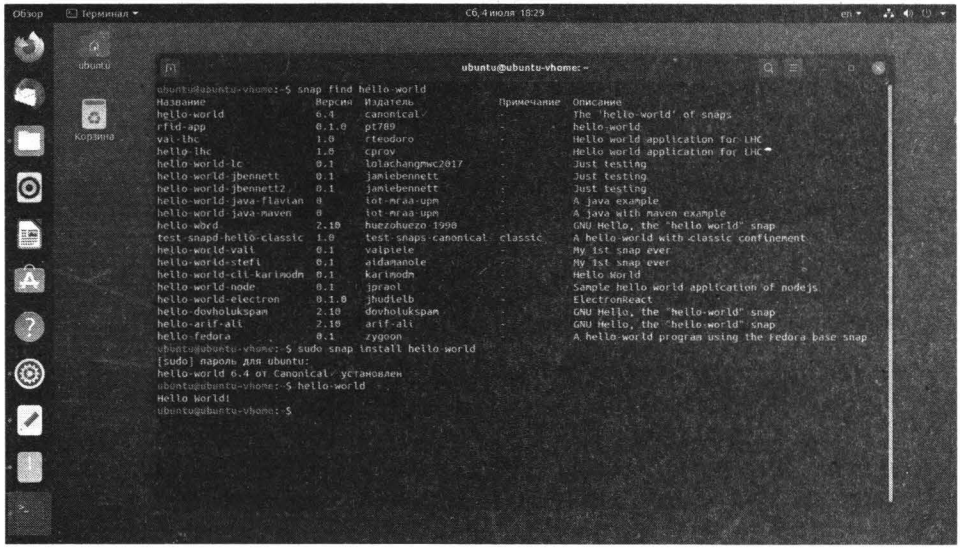


Рис. 10.8. Поиск, установка и запуск снапа

Вывод этой команды изображен на рис. 10.8. В первой колонке приводится название снапа. Пусть мы хотим установить снап hello-world:

```
sudo snap install hello-world
```

Как видите, команда установки снапа аналогична команде установки пакета, только вместо **apt** используется утилита snap.

После установки снапа можно запустить имеющуюся в нем программу:

```
hello-world
Hello, world!
```

Просмотреть установленные снапы можно командой **snap list** (рис. 10.9).

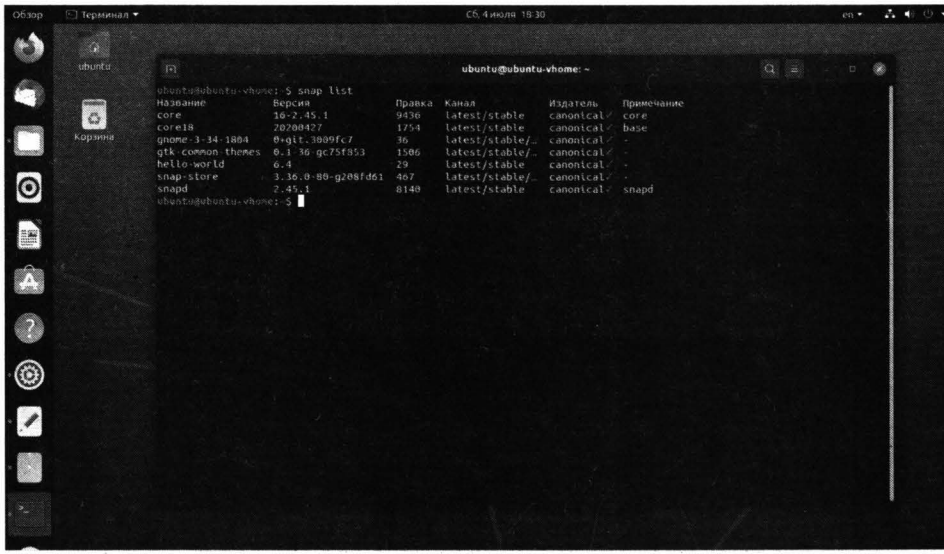


Рис. 10.9. Команда snap list

Как и пакеты, снапы можно обновить. Например:

```
sudo snap refresh hello-world
```

Данная команда обновит снап hello-world, если для него доступны обновления. Это очень удобно – вы обновляете не только приложения, но и все необходимые для его работы библиотеки.

Для обновления всех снапов используется другая команда:

```
sudo snap resresh
```

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

10.7. Ошибка при выполнении apt: Unable to acquire the dpkg lock /var/lib/dpkg/lock

К сожалению, время от времени при работе с Linux возникают различные ошибки. Одна из наиболее часто возникающих выглядит так: **Не удалось получить доступ к файлу блокировки /var/lib/dpkg/lock-frontend - open (11: Ресурс временно недоступен)**. Ошибка возникает при попытке установить пакет в Ubuntu с помощью команды apt install.

Сообщение об ошибке может немного отличаться в зависимости от различных условий. Например, могут появляться следующие ошибки:

```
E: Could not get lock /var/lib/dpkg/lock - open (11: Resource
temporarily unavailable)
E: Unable to lock the administration directory (/var/lib/
dpkg/), is another process using it?
E: Could not get lock /var/lib/apt/lists/lock - open (11:
Resource temporarily unavailable)
E: Unable to lock directory /var/lib/apt/lists/
E: Could not get lock /var/lib/dpkg/lock - open (11: Resource
temporarily unavailable)
E: Unable to lock the administration directory (/var/lib/
dpkg/), is another process using it?
```

Данные ошибки появляются, когда программа apt не может получить доступ к файлу блокировки /var/lib/dpkg/lock*. Данный файл используется, чтобы запретить одновременное выполнение операций, связанных с управлением пакетами в системе, так как при одновременном изменении данных о пакетах будет нарушена целостность «пакетной базы».

Обычно существует две основные причины появления, описанных ошибок:

- В данный момент уже выполняется экземпляр программы apt
- Предыдущий вызов apt завершился некорректно

Сначала нужно проверить, что уже не запущен другой экземпляр программы apt-get (apt). Выполним следующую команду, чтобы проверить есть ли **apt** в списке запущенных процессов:

```
ps aux | grep -i apt
```

Вывод может быть таким:

```
ubuntu 8425 0.0 0.0 79516 3752 pts/1 S+ 10:31 0:00 sudo
aptt install inkscape
ubuntu 8456 0.0 0.0 38892 944 pts/0 S+ 10:32 0:00 grep
--color=auto -i apt
```

В первой строке мы видим, что уже есть работающий экземпляр программы **apt**, который имеет PID (идентификатор) 8425. Вторая строка относится к нашей команде **grep**, которую мы запустили с аргументом **apt**, поэтому она вывела саму себя. Итак, нас интересует только первая строка.

Если вы уверены, что не запускали программу **apt** сами, или она не запущена в фоновом режиме, например, выполняется автоматическое обновление системы, то нужно принудительно завершить ее выполнение. Для этого воспользуемся командой kill 9. Команде нужно указать числовой идентификатор процесса. В нашем случае это 8425:

```
sudo kill -9 8425
```

После выполнения данной команды, процесс с идентификатором 8425 завершится.

Если первый способ вам не помог, то нужно удалить все файлы блокировки. Для этого выполняем команды:

```
sudo rm /var/lib/apt/lists/lock
sudo rm /var/cache/apt/archives/lock
sudo rm /var/lib/dpkg/lock
sudo rm /var/lib/dpkg/lock-frontend
```

Если при выполнении каких-нибудь из этих команд появится сообщение: `rm: невозможно удалить '/var/../lock': Нет такого файла или каталога`, это нормально, не обращайте на него внимания.

После этого нужно выполнить переконфигурацию пакетов:

```
sudo dpkg --configure -a
```

10.8. Невозможно найти определенный пакет

При работе над книгой мы столкнулись со следующей ситуацией. Когда вы вводите команду установки пакета, например, ту же `sudo apt install`

synaptic, то менеджер пакетов сообщает вам, что такой пакет не найден. Проблема оказалась в том, что список пакетов почему-то не синхронизировался с сервером ru.ubuntu.com. Для ее решения запустите update-manager, нажмите кнопку **Настройки** и настройте менеджер пакетов на использование основного сервера, как показано на рис. 10.10. Не факт, что такая проблема проявится у вас, но если так произойдет, то вы знаете, что с этим делать.

В следующей главе мы поговорим о популярных программах, которые, возможно, вам захочется установить.

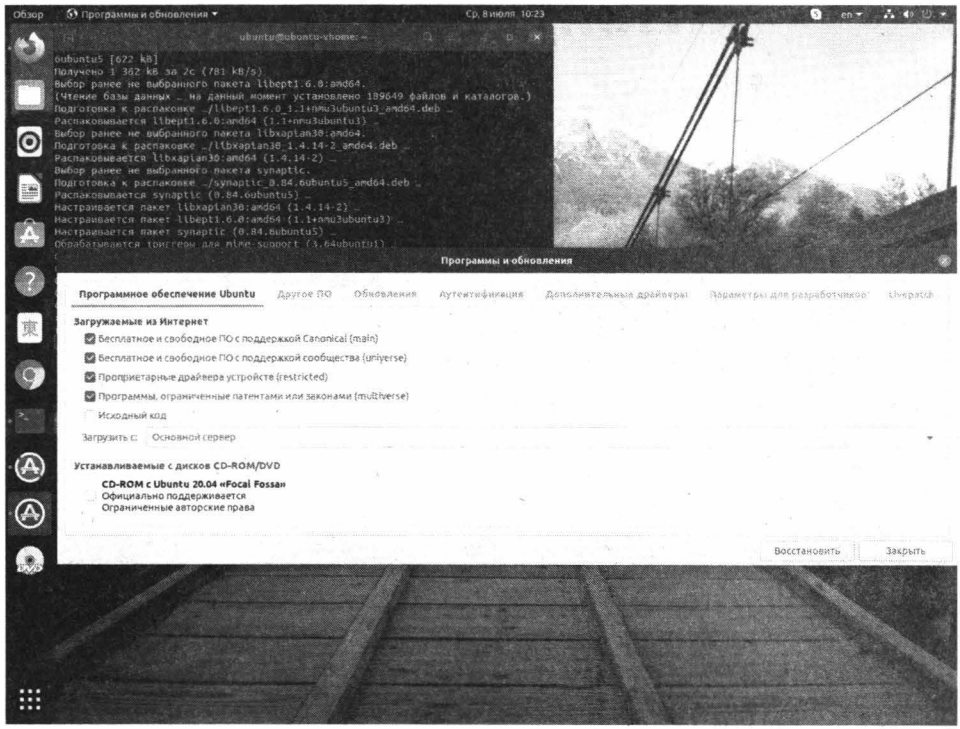
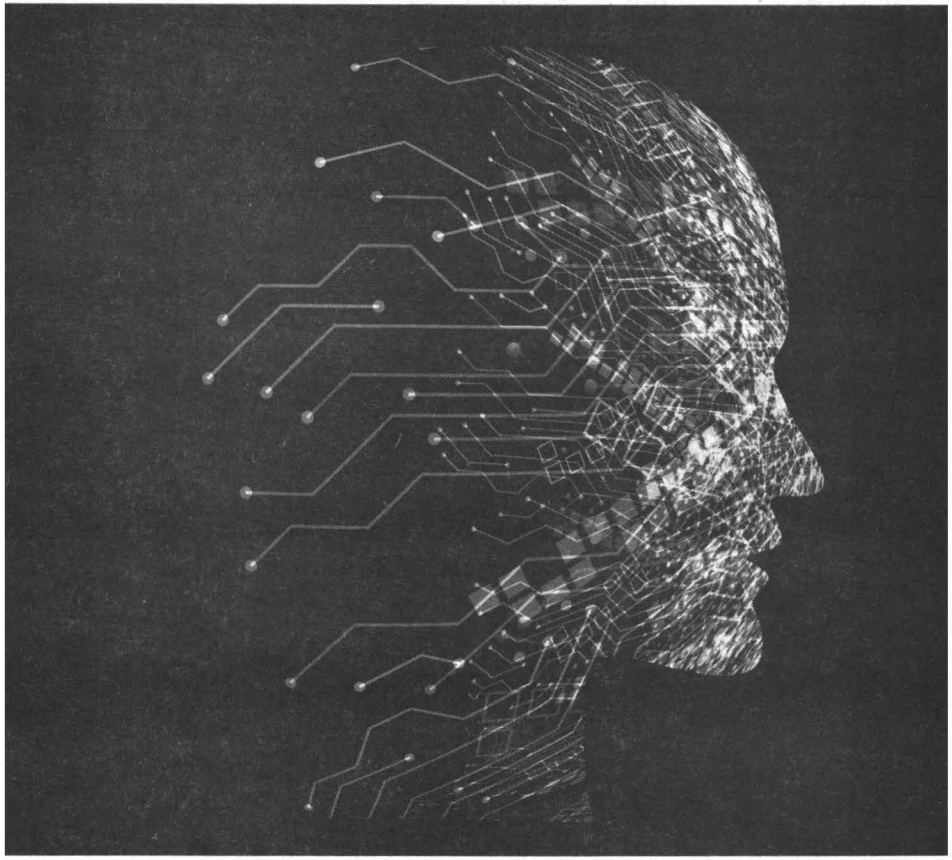


Рис. 10.10. Использование основного сервера пакетов

Глава 11.

Популярные Linux-программы



11.1. Офисные пакеты

Когда-то в Linux было многообразие офисных пакетов. Со временем все эти пакеты были вытеснены одним – LibreOffice, в котором содержатся все необходимые аналоги программ из MS Office:

- LibreOffice Writer – текстовый процессор, аналог MS Word;
- LibreOffice Calc – электронная таблица, аналог MS Excel;
- LibreOffice Impress – презентации, аналог MS PowerPoint;
- LibreOffice Draw – векторный графический редактор для создания блок-схем и диаграмм, аналог MS Visio;
- LibreOffice Base – база данных, аналог MS Access.

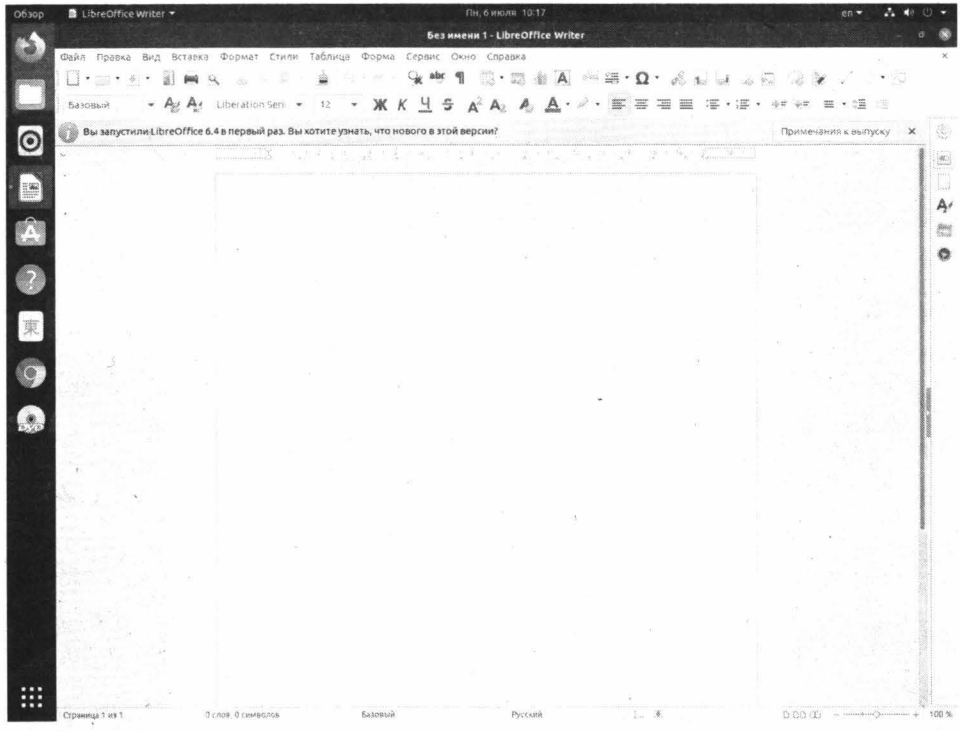


Рис. 11.1. Текстовый процессор LibreOffice Writer

Данные программы не просто выполняют те же функции, что и программы из пакета MS Office, но и поддерживают форматы MS Office. Да, интерфейс программ несколько специфичен (напоминает старые версии MS Office), но при желании можно разобраться. К тому же есть версия LibreOffice для Windows. Офисный пакет полностью бесплатный, поэтому есть неплохая возможность сэкономить, если установить его еще и на Windows-машины.

Если же какого-то функционала MS Office будет не хватать, тогда не забывайте об онлайн-версии Office 365, которую можно использовать прямо в браузере.

11.2. Графические текстовые редакторы

Для редактирования обычного текста (без форматирования), например, сценариев командной оболочки или конфигурационных файлов, можно обойтись стандартным текстовым редактором. Для редактирования файлов конфигурации нужны права root, поэтому запускать редактор нужно из терминала так:

```
sudo gedit <имя_файла>
```

Рис. 11.2. Текстовый редактор gedit

Штатный текстовый редактор обладает возможностью поиска и замены текста, нумерует строки, показывает позицию курсора. Для редактирования текстовых файлов конфигурации – то, что нужно.

Если возможностей этого редактора вам окажется мало, используя Ubuntu Software, вы всегда сможете установить редактор Atom (рис. 11.3). Это профессиональный текстовый редактор для программистов и благодаря системе плагинов, его возможности практически безграничны (рис. 11.4).

Рис. 11.3. Установка текстового редактора Atom

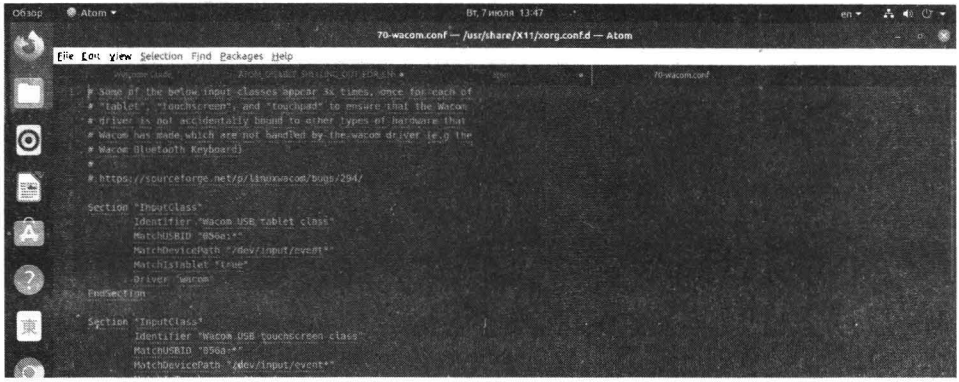


Рис. 11.4. Текстовый редактор Atom

11.3. Консольные текстовые редакторы

Работая в графическом режиме, вам вряд ли захочется использовать консольные текстовые редакторы. Но при администрировании виртуального сервера выбора у вас не будет.

Виртуальные серверы Linux не оснащены каким-либо графическим интерфейсом, поэтому их администрирование осуществляется через консоль – или посредством Web-консоли, встроенной в панель управления, или же по SSH. Часто у администратора возникает потребность отредактировать какой-то файл конфигурации сервера. Проблем с этим, как правило, никаких нет – запускаешь предпочитаемый текстовый редактор, открываешь файл, редактируешь и сохраняешь. Однако редактирование некоторых файлов конфигурации, в частности /etc/sudoers, осуществляется только посредством специальных утилит (в данном случае – **visudo** или **crontab** – при редактировании расписания планировщика), которые запускают текстовый редактор по умолчанию. Таковым редактором является редактор **vi**, перекочевавший в современные дистрибутивы Linux с 1970-ых годов и его нельзя назвать удобным. В этой заметке будут рассмотрены некоторые текстовые редакторы, и будет показано, как по умолчанию установить понравившийся редактор, чтобы он вызывался при редактировании некоторых специальных файлов конфигурации, которые нельзя редактировать вручную.

Самый удобный — редактор **nano** (раньше он назывался **pico** и входил в состав почтового клиента pine). Редактор nano изображен на рис. 11.5.

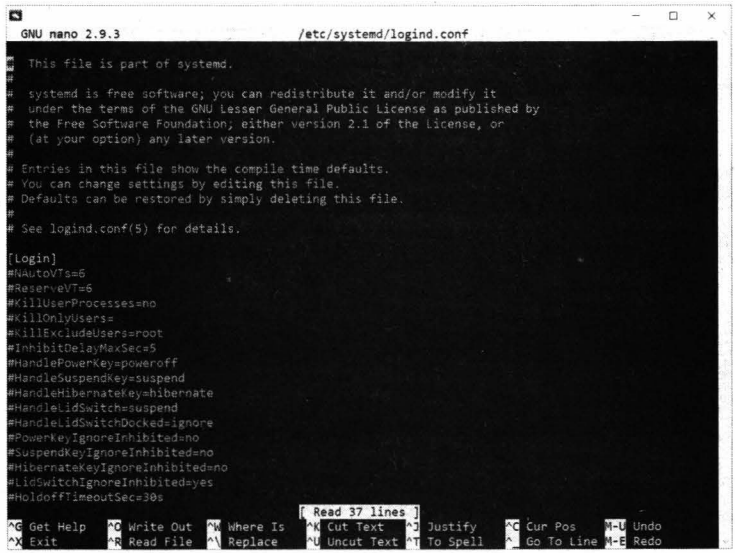


Рис. 11.5. Текстовый редактор nano

Внизу (под текстом) есть подсказка по комбинациям клавиш для управления редактором. Символ ^ означает <Ctrl>. То есть 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** (в Linux его нет). Он похож на nano, но подсказки выводятся до текста (вверху экрана), а не после него, но идея та же. Также довольно удобен редактор **joe**. Скажем так, nano будет удобнее, он поддерживает подсветку синтаксиса, внизу есть панель с подсказками, но это дело привычки.

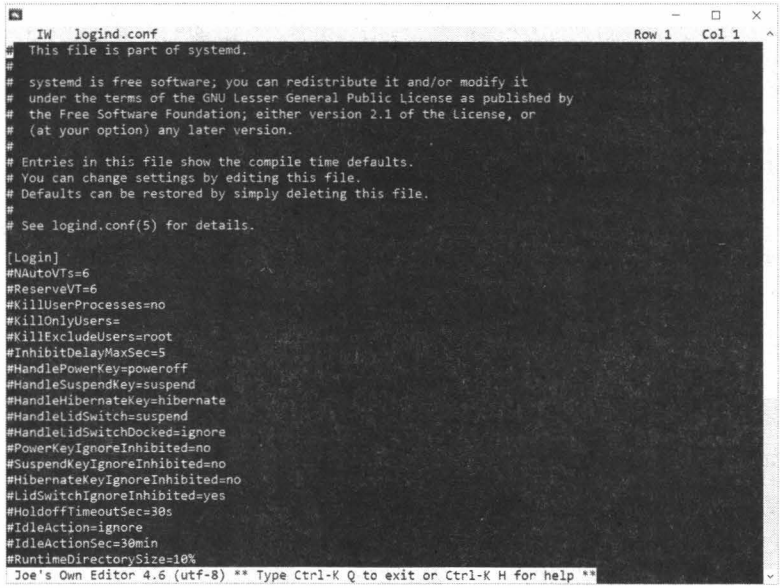


Рис. 11.6. Редактор joe

В пакет **mc** (файловый менеджер) входит довольно удобный редактор mcedit, который запускается при нажатии клавиши <F4> в mc (рис. 11.7). При желании вы можете запустить редактор отдельно:

```
mcedit <имя файла>
```

Классический синий фон, подсказки функциональных клавиш внизу и т.д. Редактор не менее удобен, чем nano.

Кстати, редакторы **joe**, **nano** и **ee** запускаются аналогично:

```
joe <имя файла>
nano <имя файла>
ee <имя файла>
```

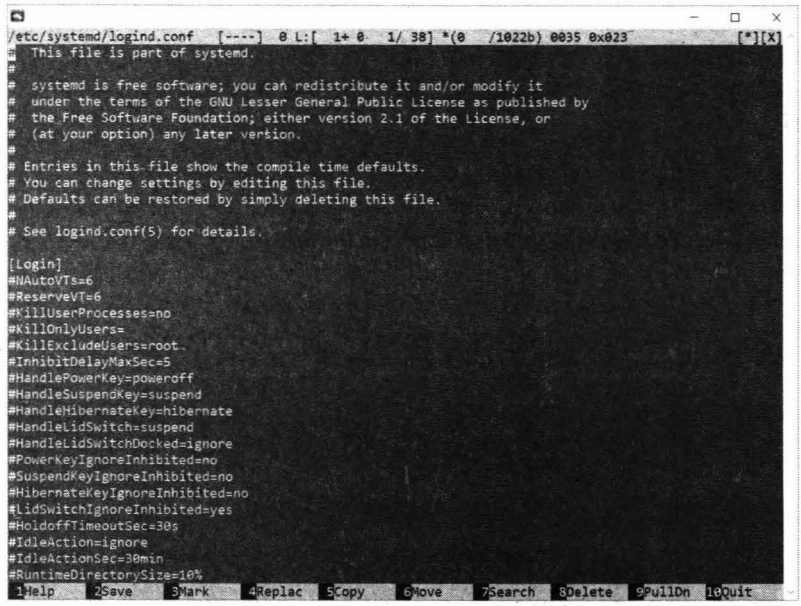


Рис. 11.7. Редактор mcedit

Выбор конкретного редактора зависит от ваших личных предпочтений. Но в любом случае, каждый из представленных редакторов будет удобнее, чем стандартный vi.

Некоторые утилиты, например, crontab, visudo вызывают текстовый редактор по умолчанию для редактирования тех или иных данных. В этом случае будет вызван vi, который, как было отмечено, неудобен. Чтобы вызывался нужный вам редактор, его нужно сделать редактором по умолчанию. Для этого нужно установить переменную окружения EDITOR:

```
which nano
/bin/nano
export EDITOR=/bin/nano
```

Первая команда (which nano) сообщает путь к выбранному редактору. Далее этот путь нам нужно указать в качестве значения переменной EDITOR.

Вот только помните, что при следующем входе в систему переменная EDITOR будет установлена по умолчанию. Чтобы этого не произошло, нужно отредактировать файл .bashrc того пользователя, от имени которого будете редактировать конфигурационные файлы. В случае с root это будет файл /root/.bashrc:

```
cd ~
nano .bashrc
```

В этот файл нужно добавить команду:

```
export EDITOR=/bin/nano
```

Сохраните файл. Теперь нужно выйти из системы (команда exit) и снова войти (по SSH или через Web-консоль). После этого запустите любую команду, вызывающую стандартный текстовый редактор, например, crontab -e. Если вы увидели выбранный вами текстовый редактор, значит, все прошло нормально (рис. 11.8). В противном случае вы где-то допустили ошибку.

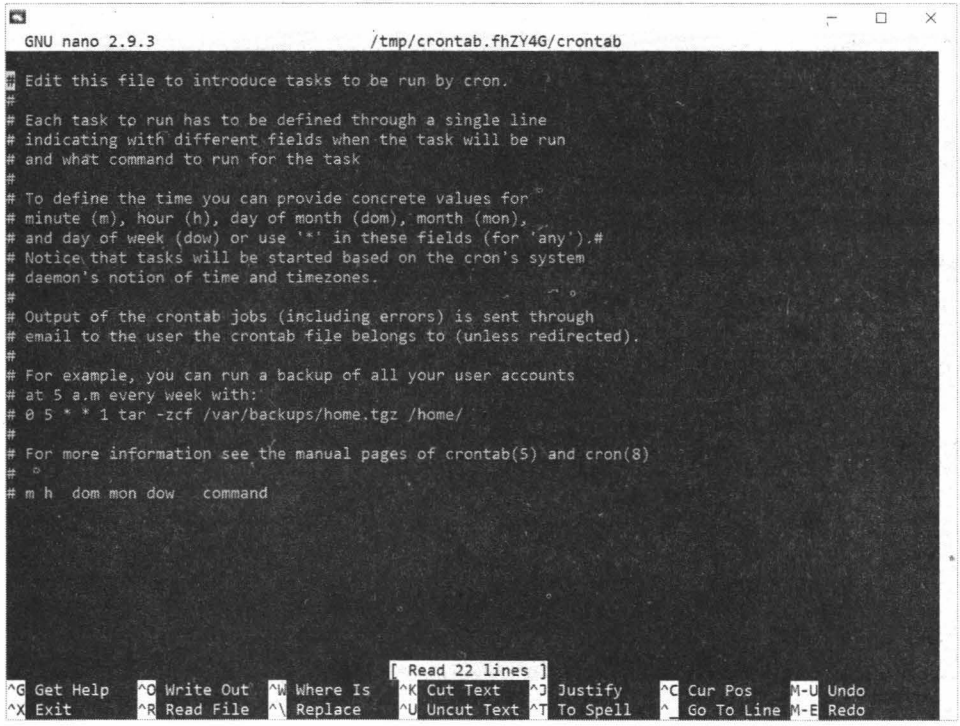


Рис. 11.8. При редактировании расписания (команда crontab –e) открылся nano. Настройка успешна

11.4. Программы для работы с Интернетом

Основные программы для работы с Интернетом установлены по умолчанию. К ним относятся:

- Браузер Firefox;
- Почтовый клиент Thunderbird;
- Torrent-клиент Transmission.

При желании вы можете доустановить FileZilla (FTP-клиент) и sFTP Client (клиент обмена файлами по протоколу SSH). Установка этих программ осуществляется посредством **Ubuntu Software** и предельно проста – все, что вам нужно сделать – нажать кнопку **Установить**.

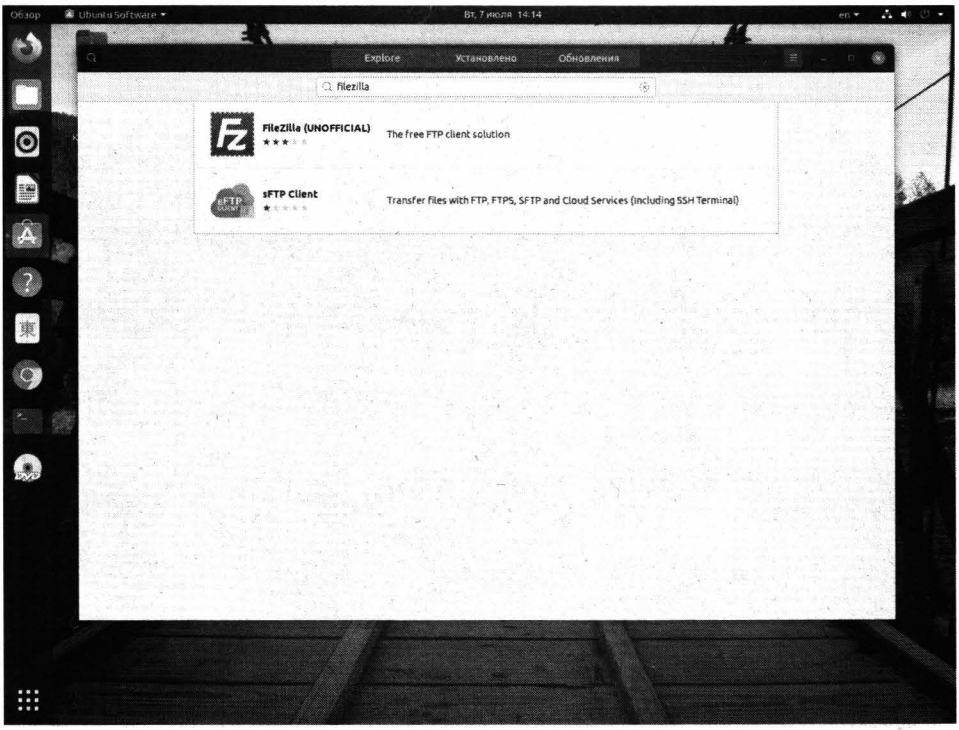
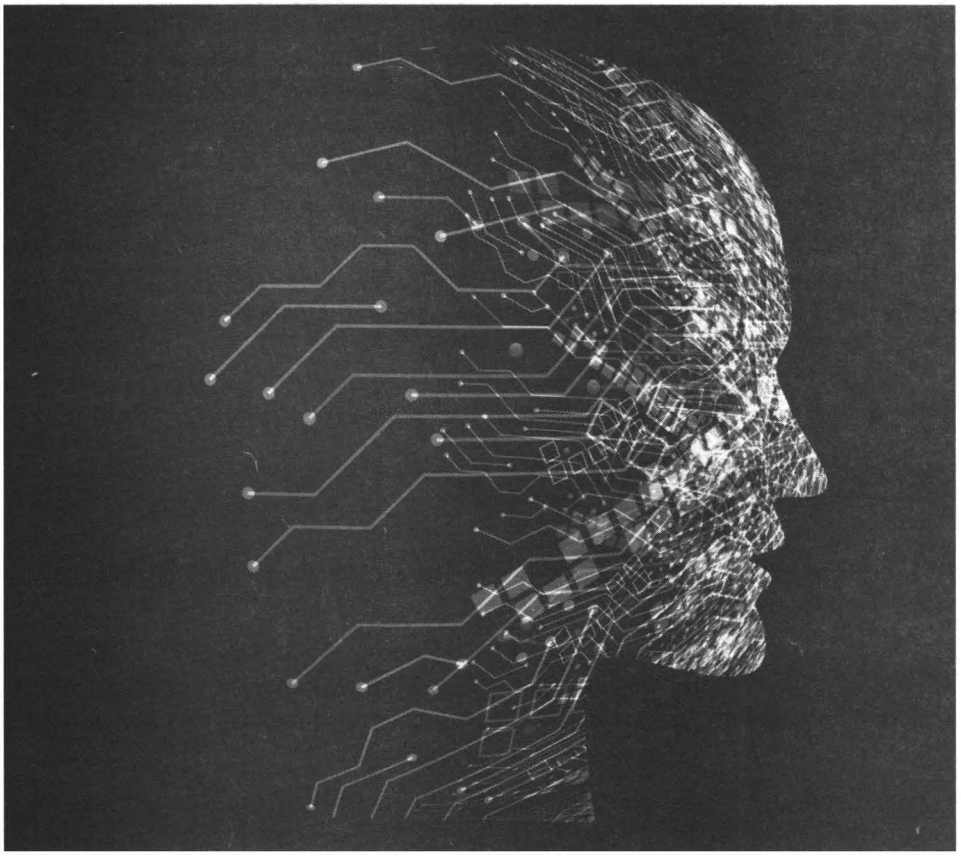


Рис. 11.9. Ubuntu Software

Глава 12.

Запуск Windows-приложений в Linux



Wine - это программа с открытым исходным кодом, которая позволяет запускать Windows-приложения в среде Linux и MacOS. Можно сказать, что это слой совместимости между операционной системой и Windows-программами. Вызовы процедур из библиотек Windows подменяются на системные вызовы Linux и с помощью этого появляется возможность выполнять Windows-программы в Linux.

Обратите внимание Wine: не виртуальная машина, в которую устанавливается Windows (которую, по-хорошему, нужно лицензировать). Это платформа запуска Windows-приложений, которой не нужна сама Windows! Соответственно и лицензировать ничего не нужно.

Платформа Wine постоянно развивается, постоянно выходят новые версии, в которых больше поддерживаемых функций Windows, исправлены многие ошибки, добавляется поддержка новых возможностей. Стабильные релизы Wine выходят приблизительно раз в год, полтора. Но корректирующие, тестовые релизы есть постоянно, даже по несколько раз в месяц.

За последнее время Wine очень сильно продвинулся в плане запуска игр. Благодаря библиотеке DXVK уже можно играть даже многие современные игры Windows без потери производительности. Дальше будет рассмотрена установка wine Ubuntu 20.04. Установим последнюю версию из официальных репозиториев, а также воспользуемся PPA.

12.1. Установка из официального репозитария

Для установки из официального репозитария просто введите команду:

```
sudo apt install wine
```

Установка займет много времени, поскольку нужно будет загрузить много вспомогательных пакетов (рис. 12.1). Можете выпить чашку кофе или позвонить другу, в общем, займите себя чем-нибудь.

Если вы увидите сообщение, что пакет **wine** не найден, тогда выполните команду:

```
sudo update-manager
```

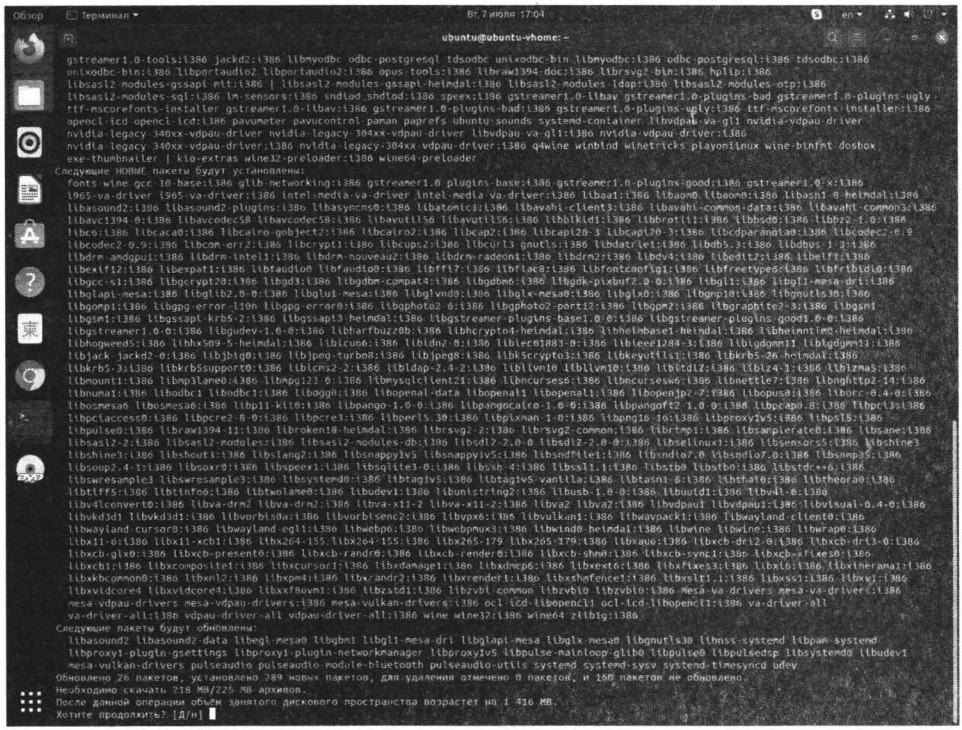


Рис. 12.1. Установка wine

В появившемся окне вместо сервера в Российской федерации выберите **Основной сервер**. Как показывает практика, ru.ubuntu.com почему-то время от времени сбоит и не всегда удается загрузить с него пакеты. После этого введите команду:

```
sudo apt update
```

После этого повторите попытку установки wine.

12.2. Установка из PPA

В репозиториях Ubuntu содержится далеко не самая последняя версия. На момент написания этих строк в репозитории focal (версия 20.04) содержится версия 5.0 (рис. 12.2).

А из репозитория разработчика можно установить версию 5.12 (рис. 12.3). При этом вам не нужно устанавливать тонну ненужного вам программного обеспечения и компилировать программу из исходного кода. Все, что вам нужно – это добавить новый репозитария и установить из него wine.

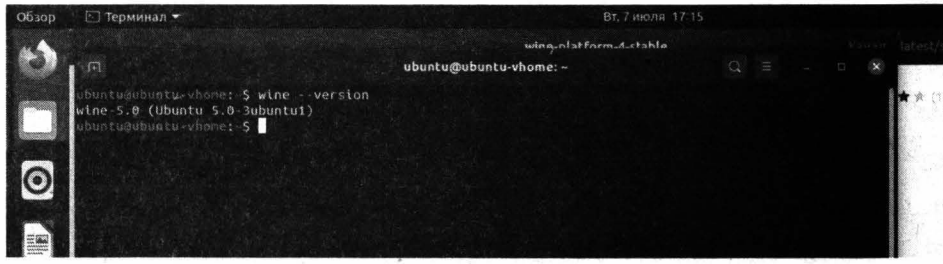


Рис. 12.2. Версия wine, установленная из репозитария

Рис. 12.3. Содержимое репозитария

Wine требует поддержку 32-битной архитектуры для установки, поэтому добавим нужную архитектуру командой:

```
$ sudo dpkg --add-architecture i386
```

Загрузим ключ репозитария и добавим его в APT с помощью следующей команды:

```
$ wget -O - https://dl.winehq.org/wine-builds/winehq.key |
sudo apt-key add -
```

Добавим репозитария focal main – он содержит пакеты для Ubuntu 20.04, если у вас другая версия замените **focal** на название версии Ubuntu:

```
$ sudo add-apt-repository 'deb https://dl.winehq.org/wine-
builds/ubuntu/ focal main'
```

Обновим список пакетов:

```
$ sudo apt update
```

Вывод должен быть, как показано на рис. 12.4. Убедитесь, что в списке есть репозитарий для focal:

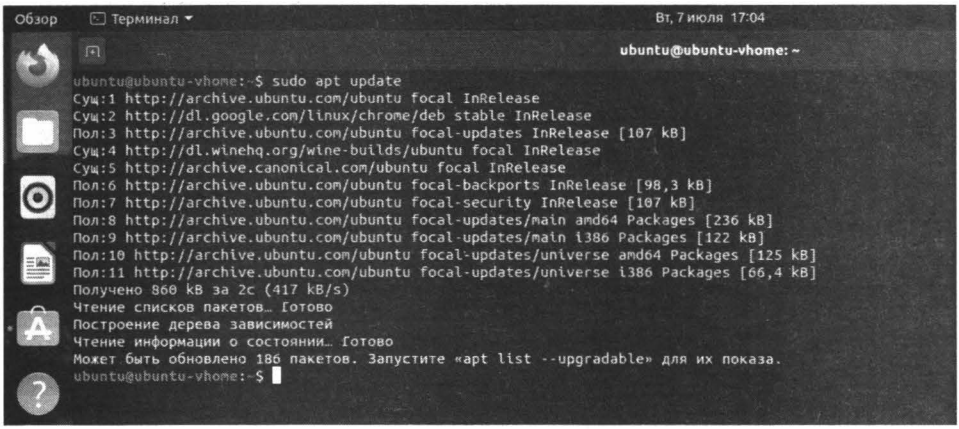


Рис. 12.4. Обновление списка пакетов

Теперь установим wine:

```
$ sudo apt install --install-recommends winehq-stable
```

Осталось дождаться установки.

12.3. Настройка после установки

Сразу после установки введите команду:

```
$ winecfg
```

Запустится конфигуратор, основное назначение которого – создать служебные каталоги, в том числе каталог для диска **C:**, на который будут устанавливаться Windows-программы. В окне конфигуратора выберите версию Windows. По умолчанию используется Windows 7, но учитывая, что современные программы постепенно отказываются от ее поддержки, рекомендуется выбрать Windows 10.

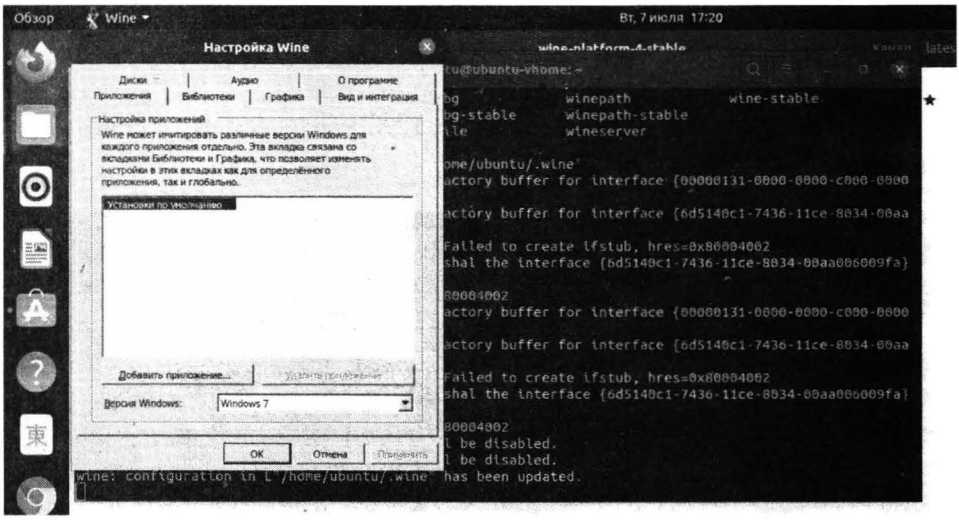


Рис. 12.5. Установка версии Windows

На вкладке **Диски** содержит список дисков, которые будут доступны Windows-приложениями. По умолчанию содержимое диска C: будет находиться в каталоге ~/.wine/drive_c. Остальные параметры можете не изменять – вернетесь к ним, когда у вас появится такая необходимость.

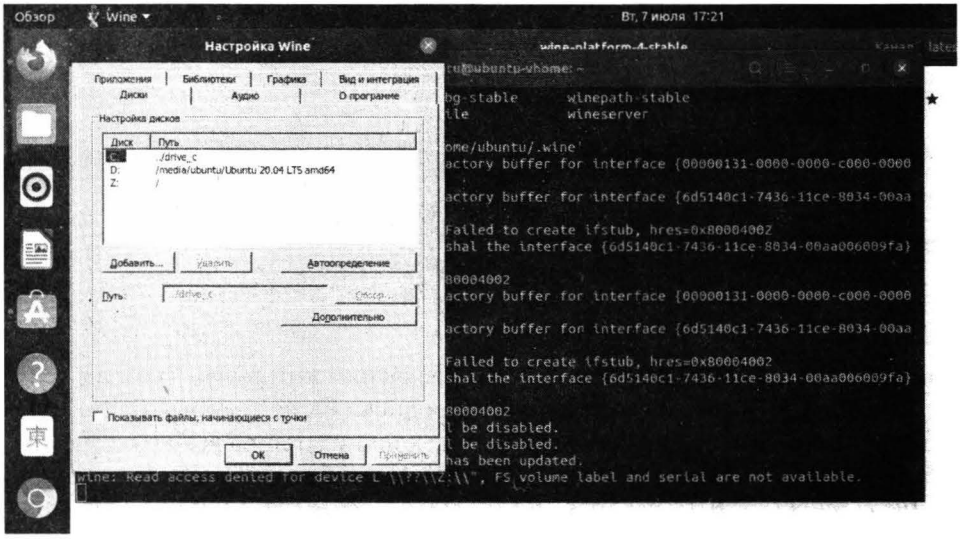


Рис. 12.6. Вкладка Диски

12.4. Установка и запуск Windows-программы

Далее будет показан процесс установки и запуска Windows-программы. Мы будем устанавливать популярных двухпанельный файловый менеджер Total Commander – помимо всего прочего он пригодится вам для просмотра файловой системы, которую «видят" устанавливаемые Windows-приложения.

Скачайте установочный 64-битный exe-файл с официального сайта (https://www.ghisler.com/). Он будет помещен в каталог **Загрузки**. Перейдите в этот каталог, используя файловый менеджер Ubuntu, щелкните правой кнопкой мыши в любой свободной области и выберите команду **Открыть в терминале**. Так вы откроете терминал, в котором уже будет выбран текущий каталог Загрузки – так быстрее.

Введите команду:

```
$ wine tcmd951x64
```

Запустится программа установки Total Commander (рис. 12.7). Произведите установку, как обычно. По завершении установки вы получите соответствующее сообщение.

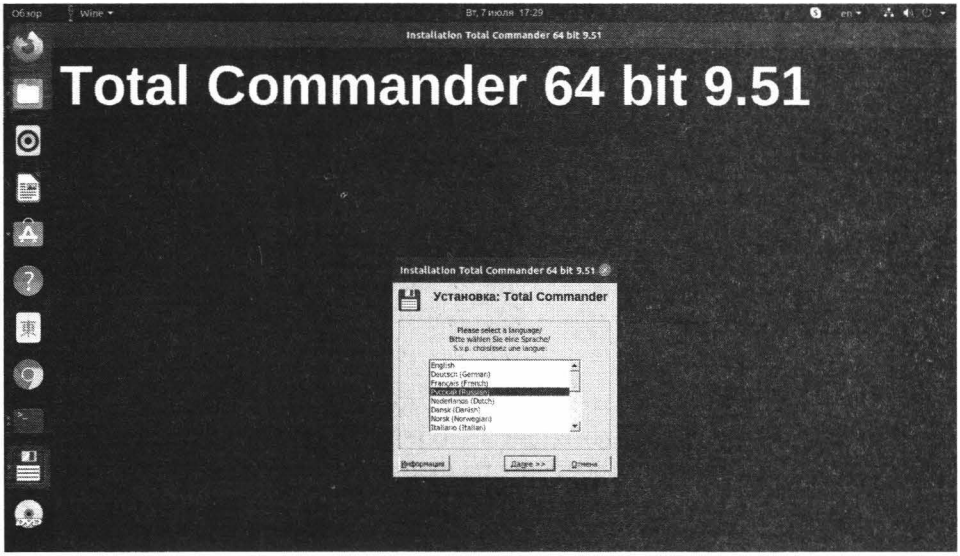


Рис. 12.7. Программа установки Total Commander

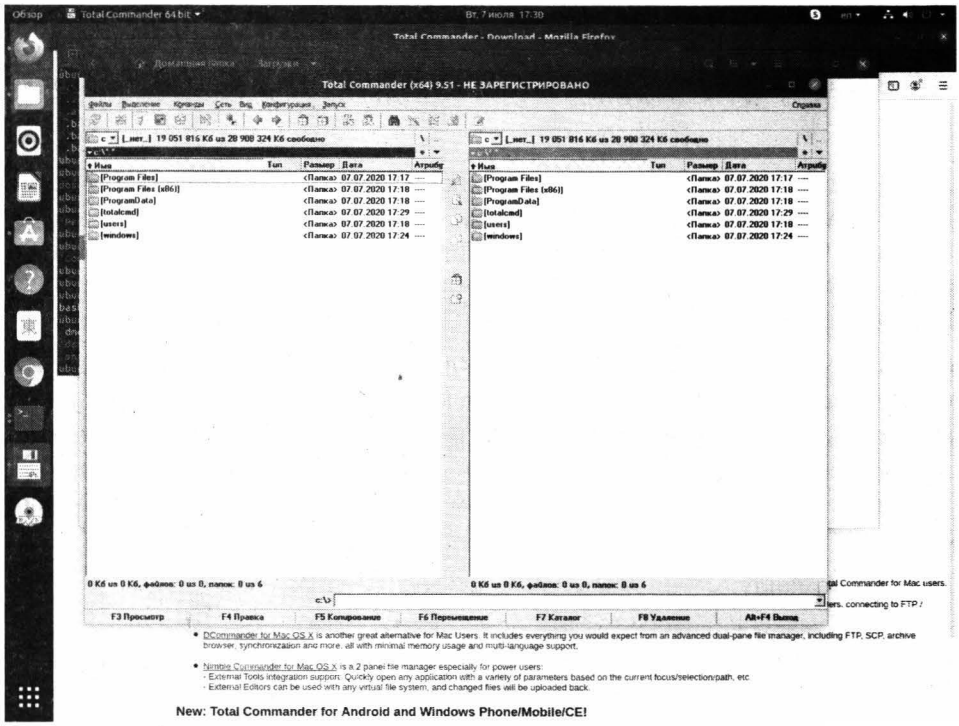


Рис. 12.8. Total Commander запущен в Linux

Далее откройте экран **Приложения** и вы найдете там кнопку для запуска Total Commander. Прошу заметить: это все-таки Windows-приложение, а запустить вы его можете как обычное «родное» приложение для Linux.

Запустите программу (рис. 12.8). Далее вы можете пользоваться нею без всяких ограничений.

12.5. Список игр и других приложений, работающих через Wine

Что делать, если не получается запустить приложение в Linux? Во-первых, попробуйте изменить версию Windows. В некоторых случаях это может помочь. Во-вторых, поищите решение на форумах: скорее всего, вы не единственный, кто столкнулся с такой проблемой и наверняка решение уже есть. В-третьих, поищите нужные настройки в базе данных приложений. На официальном сайте Wine ведется база данных программ и игр, которые

можно запустить через Wine: Wine Application Database (AppDB) – https://appdb.winehq.org.

12.6. Использование отдельных префиксов

Некоторые программы должны запускаться внутри своей среды, то есть должны быть изолированы от других приложений. Для этого им нужен отдельный префикс (отдельная директория среды, в которой они будут работать).

Префикс задается переменной WINEPREFIX. Посмотрим, как все это реализовать на практике. Сначала создадим новый префикс. Выполняем команду:

```
WINEPREFIX="/home/ubuntu/.wine/" winecfg
```

Теперь выполняем саму программу и указываем для нее новый префикс:

```
WINEPREFIX="/home/ubuntu/.wine/" wine /путь/к/файлу/setup.
exe
```

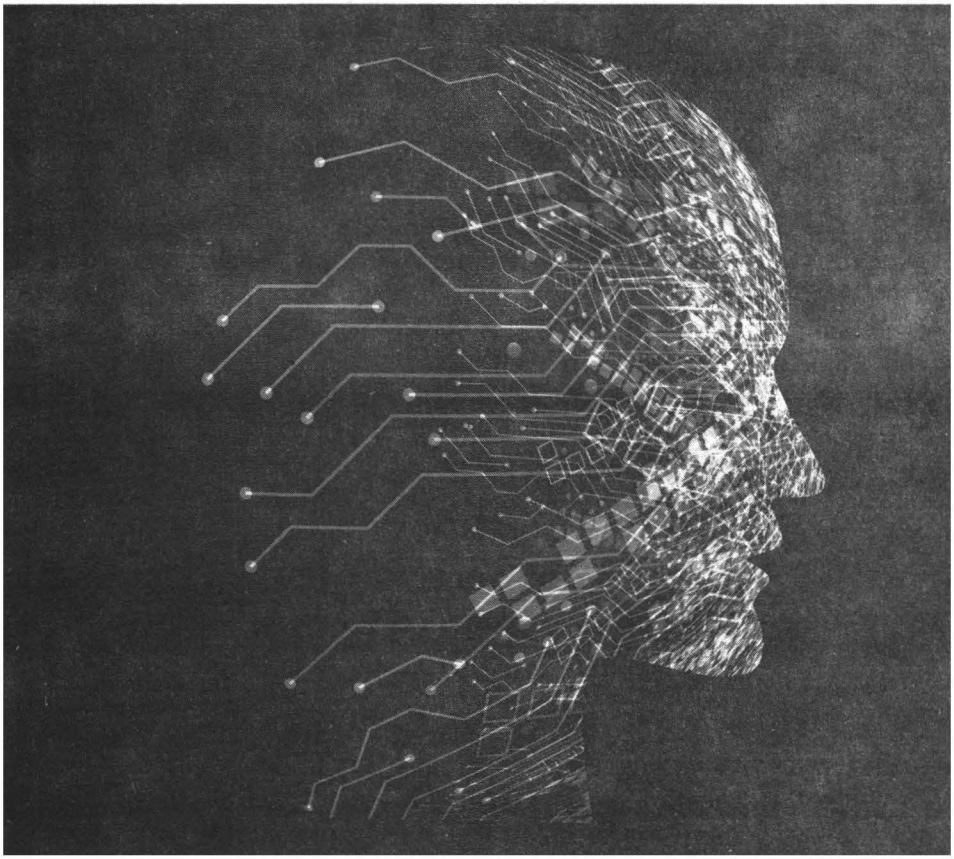
Часть III.

Локальное администрирование

В этой части книги вы узнаете об управлении файловыми системами, загрузке операционной системы, а также об основных процессах и группах пользователей в Linux

Глава 13.

Файловая система



13.1 Какие файловые системы поддерживает Linux

Операционная система Linux поддерживает очень много операционных систем. Но самое главное - это модульный принцип организации ядра Linux. Даже если кто-то сегодня создаст файловую систему, о которой еще вчера никто ничего не знал, то ему достаточно создать модуль ядра и Linux будет поддерживать его операционную систему.

Родным для Linux является семейство файловых систем ext*. Самая древняя файловая система Linux называлась ext (сегодня вы вряд ли с ней столкнетесь), затем появились ext2, ext3 и ext4. Еще в 2010 году ходили слухи о ext5, но ее так и не создали.

Файловые системы ext3 и ext4 являются журналируемыми, то есть они ведут «журналы» своей работы, что позволяет произвести восстановление информации в случае сбоя. Журналы работают так: перед осуществлением операции файловая система записывает в журнал эту операцию, а после выполнения операции - удаляет запись из журнала. Если после занесения информации в журнал произошел сбой (например, отключение электричества), то после его устранения (подача электричества) файловая система выполнит все действия, которые она не успела выполнить. Конечно, это не панацея и резервные копии никто не отменял. Но все же лучше, чем ничего.

Однако не во всех дистрибутивах ext4 используется по умолчанию. Linux также поддерживает и другие файловые системы: XFS, ReiserFS, BtrFS, ZFS, JFS. Вы можете встретить дистрибутивы, в которых по умолчанию используется одна из этих файловых систем. У каждой из этих файловых систем есть свои отличия:

- **JFS** (Journaled File System) - 64-битная журналируемая файловая система созданная IBM, распространяется по лицензии GPL и благодаря этому факту она оказалась в Linux. Обладает высокой производительностью, но у нее маленький размер блока (от 512 байт до 4 Кб), поэтому на сервере данных ее можно использовать с большим успехом, но не на

рабочих станциях, на которых производится обработка видео в реальном времени, так как размер блока для этих задач будет маловат. В отличие от ext3, в которую поддержка журнала была добавлена (по сути, ext3 - это то же самое, что и ext2, но с журналом), JFS была изначально журналируемой. Максимальный размер тома - 32 Пб, максимальный размер файла - 4 Пб.

- **ReiserFS** - самая экономная файловая система, поскольку позволяет хранить в одном блоке несколько файлов. В других файловых системах файл должен занимать как минимум 1 блок и получается, что если размер файла меньше размера блока, то «остаток» просто не используется. Когда в системе много небольших файлов, дисковое пространство используется очень нерационально. В ReiserFS все иначе. Если размер блока, скажем 4 Кб, то в него могут поместиться несколько файлов общим размером 4 Кб, а не только один, например, два файла по 2 Кб. Максимальный размер тома и файла зависят от версии ReiserFS и разрядности системы.

- **XFS** - высокопроизводительная (до 7 Гбайт/с) файловая система, разработанная Silicon Graphics. Изначально была рассчитана на большие размеры накопителей (более 2 Тб) и большие размеры файлов. Очень хорошо проявила себя при работе с файлами большого размера. Размер блока у этой файловой системы - от 512 байт до 64 Кбайта. Выделяет место экстентами (Extent — указатель на начало и число последовательных блоков). В экстентах выделяется место для хранения файлов, а также экстентами хранятся свободные блоки. Именно эта файловая система используется по умолчанию в современных версиях Fedora Server,

- **Btrfs (B-tree FS, «Better FS» или «Butter FS»)** - файловая система, разработанная специально для Linux, и основанная на структурах Б-деревьев. Работает по принципу «копирование при записи» (copy-on-write). Создана компанией Oracle Corporation в 2007 году, распространяется по лицензии GPL. Изначально планировалась как конкурент популярной файловой системе ZFS.

- **ZFS** (Zettabyte File System) - файловая система, созданная в Sun Microsystems для операционной системы Solaris. Позже она появилась и в Linux. Ее особенность - полный контроль над физическими и логическими носителями.

Кроме перечисленных выше файловых систем Linux поддерживает еще и файловые системы Windows - FAT, FAT32, NTFS. Также поддерживаются всевозможные сменные носители вроде оптических дисков, флешки, внеш-

ние жесткие диски и, соответственно, файловые системы на этих сменных носителях. На внешних жестких дисках используются файловые системы FAT32 или NTFS, если вы специально не переформатировали их в файловую систему Linux. На оптических дисках может использоваться UDF, ISO 9660, Joliet и подобные.

13.2. Какую файловую систему выбрать?

Файловых систем довольно много, так какую из них выбрать для вашей системы? На рабочих станциях и серверах общего назначения я бы использовал ext4 или ReiserFS. Последняя особенно хороша, если у вас много мелких файлов - тогда дисковое пространство будет использоваться более рационально.

На сервере баз данных лучше использовать JFS - тогда прирост производительности вам гарантирован. А вот XFS для некоторых видов серверов подходит так себе, однако, это не помешало разработчикам Fedora Server использовать эту файловую систему по умолчанию в своем дистрибутиве (начиная с версии 22). Видимо, повлияла высокая производительность и ориентация на большие объемы накопителей и файлов.

Очень неплохой является файловая система ZFS, особенно она хороша при управлении различными дисковыми устройствами. Вы можете создать пул и добавить в него несколько дисковых устройств. Этим ZFS чем-то похожа на LVM - менеджер логических томов, который мы рассмотрим в следующей главе.

Не спешите с выбором файловой системы именно сейчас. В конце этой главы вы найдете подробную информацию об ext4. Может, в ней вы найдете ответы на все ваши вопросы и остановите свой выбор на ext4.

Выбор файловой системы нужно производить даже не на основе характеристик самой файловой системы, а исходя из назначения компьютера. Например, для рабочей станции с головой хватит ext4. С сервером сложнее - нужно учитывать, какой это будет сервер. Как уже отмечалось, для сервера БД - JFS, для хостинга (где хранятся файлы других пользователей) - ReiserFS, если нужно работать с большими объемами данных - XFS.

13.3. Что нужно знать о файловой системе Linux

13.3.1. Имена файлов и каталогов

Нужно помнить следующие правила именования файлов и каталогов в Linux:

- Linux чувствительна к регистру символов, то есть файлы Document.txt и document.TXT - это разные документы.
- В Linux нет понятия «расширение» файла. Если в Windows мы привыкли, что последние символы после последней точки (обычно от 1 до 4 символов) называются расширением. В Linux такого понятия нет. Если кто-то и употребляет термин «расширение» в Linux, то это только для того, чтобы бывшим Windows-пользователям (которыми являемся, по сути, все мы) было понятнее, что имеется в виду.
- Максимальная длина имени файла - 254 символа.
- Имя может содержать любые символы (в том числе и кириллицу), кроме / \ ? < > * «|.
- Разделение элементов пути осуществляется с помощью символа /, а не \, как в Windows. В Windows мы привыкли к путям вида C:\Users\John, в Linux используется прямой слэш: /home/john.
- Если имя файла начинается с точки, он считается скрытым. Пример: .htaccess.

13.3.2. Файлы устройств

Уникальность файловой системы Linux в том, что для каждого устройства в Linux создается собственный файл в каталоге /dev. Загляните в каталог /dev - в нем вы найдете множество файлов для всех устройств вашей системы. Вот примеры некоторых файлов устройств:

- /dev/sda - первый жесткий диск, как правило, подключенный к первому SATA-контроллеру.
- /dev/sda1 - первый раздел на первом жестком диске. Нумерация разделов жестких дисков в Linux начинается с 1.
- /dev/mouse - файл устройства мыши.
- /dev/cpu - файл устройства процессора;
- /dev/cdrom - ваш CD/DVD-привод;

- /dev/random - файл устройства-генератора случайных чисел;
- /dev/tty1 - первая консоль (терминал).

Файлы устройств бывают двух типов: символьные, обмен информацией с которыми осуществляется посимвольно, и блочные, обмен информацией с которыми осуществляется блоками данных. Пример символьного устройства - /dev/ttyS0 - последовательный (COM) порт, пример блочного устройства - /dev/sda1 - раздел жесткого диска.

13.3.3. Корневая файловая система и основные подкаталоги первого уровня

Самое большое отличие, к которому придется вам привыкнуть - это наличие корневой файловой системы. Вспомните, как Windows управляет жесткими дисками. Представим, что у нас есть жесткий диск с двумя логическими дисками (разделами). Первый будет в Windows называться C:, а второй - D:. У каждого из этих логических дисков будет свой корневой каталог - C:\ и D:\.

В Linux все иначе. Представьте, что мы разбили жесткий диск /dev/sda на два раздела (как и в случае с Windows). Первый будет называться /dev/sda1 (Windows бы его назвала C:), а второй - /dev/sda2 (в Windows он был бы D:).

Мы установили Linux на первый раздел /dev/sda1. Точка монтирования этого раздела будет /, что соответствует корневой файловой системе. Второй раздел вообще никак не будет отображаться, пока вы его не *подмонтируете*. Подмонтировать можно к любому каталогу. Например, вы можете подмонтировать раздел /dev/sda2 к каталогу /home и тогда домашние каталоги пользователей будут храниться физически на другом разделе. Точка монтирования - это каталог, через который осуществляется доступ к другому разделу. Правильнее сказать даже к другой файловой системе, которая физически может находиться на другом разделе, на другом жестком диске, на внешнем жестком диске, флешке и т.д.

Корневая файловая система содержит стандартные каталоги. У каждого каталога есть свое предназначение, например, в каталоге /bin хранятся стандартные программы, в каталоге /home - домашние каталоги пользователей, в каталоге /tmp - временные файлы и т.д. Назначение стандартных каталогов приведено в таблице 13.1.

Таблица 13.1. Назначение стандартных каталогов корневой файловой системы Linux

Каталог	Описание
/	Каталог корневой файловой системы
/bin	Содержит стандартные утилиты (cat, ls, cp и т.д.)
/boot	Содержит конфигурационный файл загрузчика и некоторые модули загрузчика
/dev	Содержит файлы устройств
/etc	В этом каталоге находятся конфигурационные файлы системы и программ. Содержимое этого каталога будет рассмотрено в главе 9
/home	Здесь хранятся домашние каталоги пользователей
/lib	Содержит библиотеки и модули
/lost+found	В этом каталоге хранятся восстановленные после некорректного размонтирования файловой системы файлы
/misc, /opt	Опциональные каталоги, могут содержать все, что угодно. Некоторые программы могут устанавливаться в каталог /opt
/media	Некоторые дистрибутивы монтируют сменные устройства (оптические диски, флешки) к подкаталогам этого каталога
/mnt	Содержит точки монтирования. Как правило, здесь хранятся стационарные точки монтирования, которые обычно описываются в файле /etc/fstab
/proc	Каталог псевдофайловой системы procfs, см. гл. 8
/root	Каталог пользователя root
/sbin	Содержит системные утилиты. Запускать эти утилиты имеет право только пользователь root.
/tmp	Содержит временные файлы
/usr	Может содержать много чего - пользовательские программы (несистемные программы), документацию, исходные коды ядра и т.д.
/var	Содержит постоянно изменяющиеся данные системы - почтовые ящики, очереди печати, блокировки (locks) и т.д.

13.4. Ссылки

Ссылки позволяют одному и тому же файлу существовать в системе под разными именами. Ссылки бывают жесткими и символические. Сейчас разберемся в чем разница. Если на файл указывает хотя бы одна жесткая ссылка, вы не сможете его удалить. Количество ссылок на файл можно узнать командой ls -l. Что касается символических ссылок, то вы можете удалить файл, если на него указывает хоть 100 символических ссылок. После этого они будут «оборваны» - ссылки, как файлы, останутся на жестком диске, но они будут указывать на несуществующий файл.

У жестких ссылок есть одно ограничение. Они не могут указывать на файл, находящийся за пределами файловой системы. Представим, что каталог /tmp находится физически на одном и том же разделе, что и /. Тогда вы сможете создать ссылки на файлы, которые находятся в каталоге /tmp. Но если /tmp - это точка монтирования, к которой подмонтирован другой раздел, вы не сможете создать жесткие ссылки.

Для создания ссылок используется команда **ln**:

```
ln [-s] файл ссылка
```

Если параметр -s не указан, то будет создана **жесткая** *ссылка* на *файл*. Если параметр -s указан, то будет создана **символическая** ссылка.

13.5. Права доступа

13.5.1. Общие положения

В Linux, как и в любой многопользовательской системе, есть понятия владельца файла и прав доступа. Владелец - это пользователь, которому принадлежит файл. В большинстве случаев - это пользователь, создавший файл.

Права доступа определяют, кто и что может сделать с файлом. Права доступа файла может изменять владелец файла или пользователь root. Владелец может назначить, например, кто имеет право читать и изменять файл. Владелец также может «подарить» файл другому пользователю. После этого владельцем станет уже другой пользователь.

Права доступа у пользователя root максимальные, а это означает, что он может изменить владельца любого файла (вы можете создать файл, а root может сделать владельцем любого другого пользователя) и изменить права

доступа любого файла. Пользователь root может удалить и изменить любой файл, может создать файл в любой папке и т.д. С одной стороны, это хорошо, но если злоумышленник завладеет паролем root, то хорошего в этой ситуации мало.

Права доступа в Linux по умолчанию настроены так, что пользователь владеет только своим домашним каталогом /home/<имя_пользователя>. Поэтому создавать файлы и выполнять другие операции по работе с файлами (удаление, редактирование, копирование и т.д.) пользователь может только в этом каталоге и то при условии, что файлы принадлежат ему.

Если в домашнем каталоге пользователя root создал файл, пользователь не сможет удалить или изменить его, поскольку он не является его владельцем. Сможет ли он прочитать этот файл, зависит от прав доступа к файлу (о них мы поговорим позже).

Остальные файлы, которые находятся за пределами домашнего каталога, пользователь может только просмотреть и то, если это не запрещено правами доступа. Например, файл /etc/passwd пользователь может просмотреть, а /etc/shadow - нет. Также пользователь не может создать файлы в корневой файловой системе или в любом другом каталоге, который ему не принадлежит, если иное не установлено правами доступа к этому каталогу.

13.5.2. Смена владельца файла

Команда **chown** используется для изменения владельца файла/каталога. Формат такой:

```
chown <пользователь> <файл/каталог>
```

Здесь пользователь - это новый владелец файла. Чтобы подарить другому пользователю файл, вы должны быть или его владельцем, или пользователем root.

13.5.3. Определение прав доступа

Для изменения прав доступа используется команда **chmod**. Для изменения прав доступа вы должны быть владельцем файла/каталога или же пользователем root. Формат команды следующий:

```
chmod <права> <файл/каталог>
```

Права доступа состоят из трех наборов: для владельца, для группы владельца и для прочих пользователей. Первый набор задает возможности владельца файла, второй - для группы пользователей, в которую входит владелец и третий - для всех остальных пользователей.

В наборе может быть три права - чтение (r), запись (w) и выполнение (x). Для файла право на выполнение означает возможность запустить этот файл на выполнение (обычно используется для программ и сценариев). Право выполнение для каталога - возможность просматривать этот каталог.

Права доступа в наборе определяются четко в определенном порядке и могут быть представлены, как символьном, так и числовом виде (в двоичной или восьмеричной системе). Рассмотрим несколько наборов прав доступа:

- 100 - только чтение
- 110 - чтение и запись
- 101 - чтение и выполнение
- 111 - чтение, запись, выполнение

Учитывая, что права доступа задаются для владельца, группы и остальных пользователей, полный набор прав доступа может выглядеть так:

```
111 100 000
```

В этом наборе мы предоставляем полный доступ (в том числе и выполнение) владельцу, группе владельца разрешено только чтение, остальным пользователям доступ к файлу/каталогу вообще запрещен.

В двоичной системе права доступа мало кто записывает. В основном их преобразуют в восьмеричную систему. Если вы забыли, то вам поможет следующая таблица (табл. 13.2).

Таблица 13.2. Преобразование из двоичной в восьмеричную систему

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

Если вы видите право доступа 666, то никакой дьявольщины в нем нет, это всего лишь полный доступ к обычному файлу (не к программе и не к сценарию). Для каталога полные права доступа выглядят как 777 - чтение, изменение и просмотр каталога для владельца, группы и прочих пользователей.

Просмотреть текущие права доступа можно командой `ls -l <файл/каталог>`, например:

```
# ls -l config
-rw-r--r--. 1 root root 110375 янв 2 08:28 config
```

Как мы видим, задано три набора rw-, r--, r--. Выходит, владельцу разрешена запись и чтение файла, остальным пользователям (группа и прочие) - только чтение. В восьмеричной системе этот набор прав доступа выглядит как 644.

Первый символ (в нашем случае это -) является признаком каталога. Если бы мы выводили права доступа каталога, то вместо - здесь был бы символ **d**. Для файла выводится просто «-».

Символьный способ задания прав доступа немного проще, но лично я предпочитаю числовой. Рассмотрим, как использовать символьный:

```
# chmod +x config
```

Просмотрим опять права доступа:

```
# ls -l config

-rwxr-xr-x. 1 root root 110375 sep 2 08:28 config
```

Как видите, право выполнение было добавлено во все три набора прав доступа.

13.5.4. Специальные права доступа

В Linux есть еще специальные права доступа SUID (Set User ID root) и SGID (Set Group ID root), позволяющие обычным пользователям запускать программы, которые требуют для своей работы прав root.

В современных дистрибутивах Linux вам придется изменять эти права доступа чрезвычайно редко (может быть даже вообще никогда), но вам нужно знать, как их изменить. Например, если программу /usr/sbin/program вы

хотите разрешить запускать с правами root обычным пользователям, установите права доступа так:

```
# chmod u+s /usr/sbin/program
```

Использование SUID - плохое решение с точки зрения безопасности. Правильнее использовать команду sudo, если какому-то пользователю будут нужны права root (см. гл. 14).

13.6. Атрибуты файла

В Linux кроме прав доступа есть еще и атрибуты файла, подобно атрибутам файла в других операционных системах. Изменить атрибуты файла можно командой **chattr**:

```
chattr +/-<атрибуты> <файл>
```

Просмотреть установленные атрибуты можно командой lsattr:

```
lsattr <файл>
```

Некоторые полезные атрибуты файлов приведены в таблице 13.3.

Таблица 13.3. Полезные атрибуты файлов

Атрибут	Описание
i	Запрещает изменение, переименование и удаление файла. Этот атрибут можно установить для критических конфигурационных файлов или для каких-либо других критических данных. Установить (как и сбросить) этот атрибут может только пользователь root или процесс с возможностью CAP_LINUX_IMMUTABLE. Другими словами, сбросить этот атрибут просто так нельзя - нужны только права root
u	При удалении файла с установленным атрибутом u его содержимое хранится на жестком диске, что позволяет легко восстановить файл
c	Файл будет сжиматься. Можно установить этот атрибут для больших файлов, содержащих несжатые данные. Доступ к сжатым файлам будет медленнее, чем к обычным, поэтому плохое решение устанавливать этот атрибут для файлов базы данных. Этот атрибут нельзя устанавливать для файлов, уже содержащих сжатые данные - архивы, JPEG-фото, MP3/MP4-файлы и т.д. Этим вы не только не уменьшите его размер, но и замедлите производительность

S	Данные, записываемые в файл, сразу будут сброшены на диск. Аналогично выполнению команды sync сразу после каждой операции записи в файл
s	Прямо противоположен атрибуту u. После удаления файла, принадлежащие ему блоки будут обнулены и восстановить их уже не получится

Пример установки атрибута:

```
# chattr +i config
```

Пример сброса атрибута:

```
# chattr -i config
```

13.7. Поиск файлов

Для поиска файлов вы можете использовать команды **which**, **locate** и **find**. Первая используется только для поиска программ. Она позволяет определить, в каком каталоге находится исполнимый файл той или иной программы, например:

```
# which pppd
/sbin/pppd
```

Данную программу очень удобно использовать администратору, когда нужно вычислить месторасположение программы, например, чтобы указать точный путь к программе в каком-то сценарии или конфигурационном файле.

Команда **locate** позволяет произвести быстрый поиск файла. Однако команда locate будет работать не во всех дистрибутивах, а только там, где доступен updatedb, который и формирует базу данных, по которой производит поиск команда locate. Если файл будет на диске, но его не будет в базе данных, то locate его не найдет - вот в чем основной недостаток этой команды. Для обновления базы данных нужно ввести команду updatedb (или дождаться, пока планировщик обновит базу данных).

Преимущество команды locate в том, что поиск файла производится практически мгновенно, особенно по сравнению с командой find. Однако если файла не будет в базе данных (файл был создан после обновления базы данных locate), команда locate его не найдет.

Конфигурация updatedb хранится в файле /etc/updatedb.conf (листинг 13.1).

Листинг 13.1. Файл /etc/updatedb.conf

```
PRUNE_BIND_MOUNTS = "yes"
PRUNEFS = "9p afs anon_inodefs auto autofs bdev binfmt_misc
cgroup cifs coda configfs cpuset debugfs devpts ecryptfs exofs
fuse fuse.sshfs fusectl gfs gfs2 hugetlbfs inotifyfs iso9660
jffs2 lustre mqueue ncpfs nfs nfs4 nfsd pipefs proc ramfs
rootfs rpc_pipefs securityfs selinuxfs sfs sockfs sysfs tmpfs
ubifs udf usbfs"
PRUNENAMES = ".git .hg .svn"
PRUNEPATHS = "/afs /media /mnt /net /sfs /tmp /udev /var/
cache/ccache /var/lib/yum/yumdb /var/spool/cups /var/spool/
squid /var/tmp"
```

Если параметр PRUNE_BIND_MOUNTS равен yes, файловые системы, смонтированные в режиме **bind** не исследуются при помощи updatedb. Параметр PRUNEFS задает типы файловых систем, которые не будут исследоваться updatedb. Аналогично, параметры PRUNENAMES и PRUNEPATHS задают имена файлов (у нас заданы «расширения») и пути (каталоги).

В листинге 13.1 приведен пример файла update.conf по умолчанию из Fedora Server. Вы можете отредактировать его под свои нужды, например, закомментировать параметр PRUNENAMES, отредактировать параметр PRUNEPATHS и т.д.

Теперь перейдем к третьей и самой универсальной команде поиска - **find**. Формат вызова следующий:

```
$ find список_поиска выражение
```

Полное описание команды **find** вы найдете в справочной системе (команда man mount), а мы рассмотрим несколько примеров.

```
$ find / -name test.txt
```

Мы ищем все файлы с именем test.txt, начиная с корневого каталога /. Если нужно найти все текстовые файлы (*.txt), начиная с корневого каталога, тогда команда будет такой:

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

Следующая команда ищет только пустые файлы (параметр -empty):

```
$ find . -empty
```

Если нужно задать размер файла, тогда можем указать размер явно. В следующем примере мы задаем размер файла - от 500 до 700 Мб:

```
$ find ~ -size +500M -size -700M
```

Команда **find** может не только находить файлы, но и выполнять действие для каждого найденного файла. В следующем примере мы находим все старые резервные копии («расширение» .bak) и удаляем их:

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

Поиск с помощью find занимает немало времени - ведь нет никакой базы данных. Команде find нужно «пройтись» по всем каталогам и проверить в них наличие искомых файлов.

13.8. Монтирование файловых систем

13.8.1. Монтируем файловые системы вручную

Ранее было сказано, что такое точка монтирования - это каталог, через который происходит доступ к файловой системе, физически размещенной на другом носителе (другом разделе жесткого диска, флешке, оптическом диске или даже на другом компьютере).

Для монтирования файловой системы используется команда **mount**, для размонтирования - **umount**:

```
# mount [опции] <имя устройства> <точка монтирования>
# umount <имя устройства или точка монтирования>
```

Для монтирования файловой системы нужны права root, поэтому команды mount и umount нужно вводить с правами root.

Представим, что мы подключили флешку. Если у вас один жесткий диск (/dev/sda), то флешке будет назначено имя /dev/sdb, если жестких дисков два, то флешке будет назначено следующее имя - /dev/sdc и т.д.

На одном носителе (это качается жестких дисков, флешек и подобных носителей) может быть несколько разделов, которым назначаются номера, нумерация начинается с единицы. Поэтому вы не можете подмонтировать все устройство /dev/sdc. Вы должны указать номер раздела.

Подмонтируем нашу флешку (пусть это будет устройство /dev/sdc и на нем будет всего один раздел с номером 1):

```
# mount /dev/sdc1 /mnt/usb
```

Каталог /mnt/usb - это и есть точка монтирования. Точка монтирования должна существовать до вызова команды mount, то есть вы не можете подмонтировать файловую систему к несуществующему каталогу.

После этого вы можете обращаться к файлам и каталогам на флешке через каталог /mnt/usb:

```
# ls /mnt/usb
```

Итак, последовательность действий такая: создание точки монтирования (один раз), монтирование файловой системы, работа с файловой системой и размонтирование. Размонтирование осуществляется командой umount. В качестве параметра команды umount нужно передать или название точки монтирования или имя устройства:

```
# mkdir /mnt/usb
# mount /dev/sdc1 /mnt/usb
# cp test.txt /mnt/usb
# umount /mnt/usb
```

Очень важно размонтировать файловую систему, особенно это касается внешних файловых систем. При завершении работы система автоматически размонтирует все смонтированные файловые системы.

Думаю, в общих чертах операция монтирования должна быть понятной. Теперь поговорим о параметрах команды mount. Самый часто используемый параметр - это параметр -t, позволяющий задать тип монтируемой файловой системы. Обычно команда mount сама в состоянии распознать тип файловой системы, но в некоторых случаях ей нужно указать его вручную. Вот наиболее распространенные типы файловых систем:

- vfat - файловая система Windows (FAT/FAT32);
- ntfs - файловая система Windows NT;
- ntfs-3g - драйвер ntfs-3g для чтения и записи NTFS (рекомендуется);
- ext2/ext3/ext4 - различные версии файловой системы Linux;
- iso9660 - файловая система оптического диска CD/DVD;
- udf - иногда Windows форматирует оптический диск как UDF;
- reiserfs - файловая система ReiserFS;
- smbfs - файловая система Samba;
- nfs - сетевая файловая система.

Например, в случае с NTFS рекомендуется использовать драйвер ntfs-3g:

```
# mount -t ntfs-3g /dev/sdc1 /mnt/usb
```

Параметр -r позволяет смонтировать файловую систему в режиме «только чтение», параметр -w монтирует файловую систему в режиме «чтение/запись», но обычно в этом режиме файловая система монтируется по умолчанию, поэтому в нем нет необходимости.

Параметр -a монтирует все файловые системы, перечисленные в файле /etc/fstab, за исключением тех, для которых указана опция noauto.

13.8.2. Имена устройств

Интерфейсов жестких дисков довольно много - IDE (ATA/PATA), SATA (Serial ATA), SCSI, USB. Раньше жесткие диски с интерфейсом IDE назывались в Linux /dev/hd? (? - буква, которая зависит от того, как подключен жесткий диск). Жесткие диски с интерфейсом SATA и SCSI назывались /dev/sd? (? - буква диска, соответствующая его порядковому номеру при подключении к интерфейсу).

Сейчас даже IDE-диски называются /dev/sd? (как и SATA/SCSI), что сначала вносило некую путаницу. Но жесткие диски с интерфейсом IDE вышли из моды и практически не используются. Мода на SCSI-диски также практически закончилась, поскольку SATA-диски такие же быстрые, как и SCSI - некоторые обеспечивают такую же производительность, как и SCSI - в некоторых случаях чуть меньше, в некоторых - даже больше. Так что SCSI уже можно списывать со счета - если вам достался сервер со SCSI-диском, отказываться от него не стоит, а вот новый сервер будет поставляться или с интерфейсом SATA или с интерфейсом SAS.

Интерфейс SAS (Serial Attached SCSI) обратно совместим с интерфейсом SATA и позволяет последовательно подключать SATA-диски и обеспечивает пропускную способность в 6 Гбит/с. Если вы купите сервер с интерфейсом SAS, то в большинстве случаев он будет оснащен высокопроизводительными SATA-дисками, а не SCSI-дисками. Поэтому никакой путаницы уже нет.

Что же касается USB-дисков (флешки и внешние жесткие диски), то они также получают обозначение /dev/sd?.

Оптические диски (приводы CD/DVD) в большинстве случаев называются /dev/sr?, где ? - номер привода, нумерация начинается с 0.

Чтобы узнать, какие жесткие диски и оптические приводы установлены в вашем компьютере, введите команды (рис. 13.1):

```
ls /dev/sd?
ls /dev/sr0
```

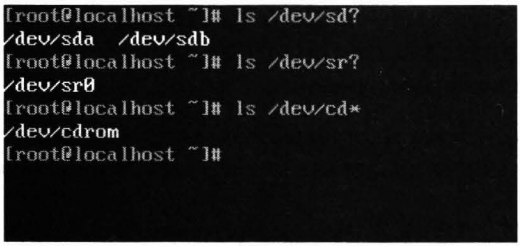


Рис. 13.1. Жесткие и оптические диски

Если у вас один оптический диск, можно смело использовать ссылку /dev/cdrom. Из рис. 13.1 видно, что у нас установлено два жестких диска - /dev/sda и /dev/sdb, а также один оптический привод /dev/sr0.

Для монтирования не достаточно указать имя всего устройства, нужно уточнить номер раздела. Когда разделов много, вы можете забыть (или не знать), какой именно раздел вам нужен. Вы можете использовать команду fdisk: запустите fdisk <имя устройства>, а затем введите команду **p** для вывода таблицы разделов и команду **q** для выхода из fdisk.

Посмотрите рис. 13.2. Из него становится понятно, что на нашем жестком диске есть два раздела - /dev/sda1 и /dev/sdb2.

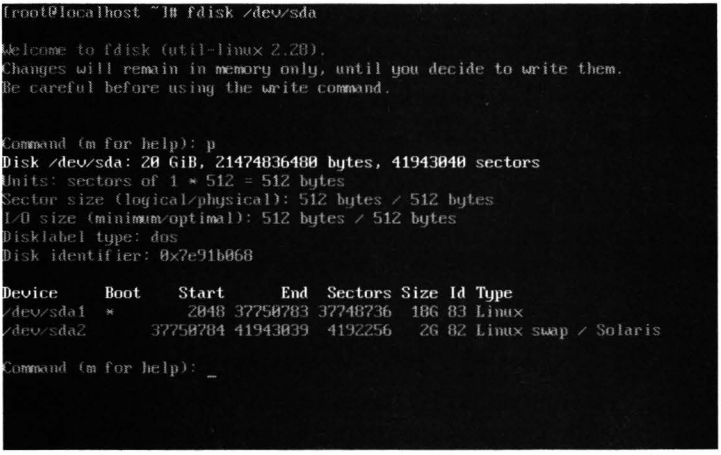


Рис. 13.2. Программа fdisk

Кроме коротких имен вроде /dev/sd? в современных дистрибутивах часто используются идентификаторы UUID. Загляните в файл /etc/fstab и в нем в большинстве случаев вместо привычных имен /dev/sd? вы обнаружите вот такие «страшные» имена:

```
# В Fedora, Debian, Ubuntu
UUID=2f149af9-3bff-44bd-d16s-ff98s9a7116d / ext4 defaults 0  1

# openSUSE
/dev/disk/by-id/dm-name-suse-server-root / ext4  defaults 1 1
```

Преимущество идентификаторов UUID в том, что они не изменяются, если вы иначе подключите жесткий диск. Представим, что у вас есть два жестких диска - /dev/sda и /dev/sdb. На первый вы установили Linux (в раздел /dev/sda1), а второй используете для хранения данных (на нем всего один раздел /dev/sdb1, который монтируется, как /home). Вы отключили оба диска, а затем, подключая, перепутали их местами. В итоге второй диск стал диском /dev/sda, а первый - /sdb. При загрузке может случится конфуз, точнее, система вообще не загрузится. Даже если вы выберете в BIOS SETUP загрузку со второго жесткого диска, тоже ничего хорошего не выйдет. С UUID достаточно выбрать загрузку со второго жесткого диска, и система будет загружена.

Вы можете использовать обычные стандартные имена, а можете использовать UUID-идентификаторы. Узнать, какие UUID-идентификаторы соответствуют каким обычным именам, можно с помощью команды:

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

Вывод этой команды представлен на рис. 13.3.

Рис. 13.3. Соответствие UUID-идентификаторов обычным именам

13.8.3. Монтируем файловые системы при загрузке

Вводить команды mount при каждой загрузке не очень хочется, поэтому проще «прописать» файловые системы в файле /etc/fstab, чтобы система смонтировала их при загрузке.

Формат файла /etc/fstab следующий:

```
устройство точка тип опции флаг_копирования флаг_проверки
```

Первое поле - это устройство, которое будет монтироваться к точке монтированию - второе поле. Вы можете использовать, как обычные имена, так и UUID. Третье поле - тип файловой системы. Четвертое поле - параметры файловой системы (табл. 13.4), последние два поля - это флаг резервной копии и флаг проверки. Первый флаг определяет, будет ли файловая система заархивирована командой dump при создании резервной копии (1 - будет, 0 - нет). Второй флаг определяет, будет ли файловая система проверяться программой **fsck** на наличие ошибок (1, 2 - будет, 0 - нет). Проверка производится, если достигнуто максимальное число попыток монтирования для файловой системы или если файловая система была размонтирована некорректно. Для корневой файловой системы это поле должно содержать 1, для остальных файловых систем - 2.

Таблица 13.4. Параметры файловой системы

Опция	Описание
defaults	Параметры по умолчанию
user	Разрешает обычному пользователю монтировать/размонтировать данную файловую систему
nouser	Запрещает обычному пользователю монтировать/размонтировать данную файловую систему. Смонтировать эту ФС может только root. Используется по умолчанию
auto	ФС будет монтироваться при загрузке. Используется по умолчанию, поэтому указывать не обязательно
noauto	ФС не будет монтироваться при загрузке системы
exec	Разрешает запуск исполнимых файлов на данной ФС. Используется по умолчанию. Для Windows-файловых систем (vfat, ntfs) рекомендуется использовать опцию noexec
noexec	Запрещает запуск исполнимых файлов на данной ФС
rw	ФС будет монтироваться в режиме «только чтение»

rw	ФС будет монтироваться в режиме «чтение запуск». По умолчанию для файловых систем, которые поддерживают запись
utf8	Для преобразования имен файлов будет использоваться кодировка UTF8
data	Задает режим работы журнала (см. ниже)

13.8.4. Автоматическое монтирование файловых систем

В современных дистрибутивах Linux сменные носители вроде USB-дисков и оптических дисков монтируются автоматически:

- Debian, Ubuntu, Fedora, CentOS - монтирование производится к каталогу /media/<метка_устройства>. В качестве метки может использоваться или метка, установленная при форматировании, или серийный номер устройства, если метка не устанавливалась.
- openSUSE - монтирование будет производиться к каталогу /var/run/media/<имя пользователя>/<метка>.

За автоматическое монтирование отвечает демон automount, который вы можете отключить, если автоматическое монтирование вам не нужно.

13.9. Работа с журналом

Существует три режима работы журналируемой файловой системы ext3/ext4: **journal**, **ordered** и **writeback**. По умолчанию используется режим ordered - оптимальный баланс между производительностью и надежностью. В этом режиме в журнал будет заноситься информация только об изменении метаданных.

Самый медленный режим journal. В этом режиме в журнал записывается максимум информации, которая понадобится при восстановлении в случае сбоя. Режим очень медленный и использовать его следует только, если безопасность для вас важнее, чем производительность.

Самый быстрый режим writeback, но в нем, по сути, журнал не будет использоваться и у вас не будет никакой защиты, например, от той же перезагрузки.

Режим работы журнала задается параметром **data**, например:

```
/dev/sdb1 /home ext4 data=journal 1 2
```

13.10. Преимущества файловой системы ext4

Поговорим о преимуществах файловой системы ext4. Возможно, она вас полностью устроит и вам не придется искать другую файловую систему для своего компьютера.

Впервые файловая система ext4 появилась в ядре версии 2.6.28. По сравнению с ext3, максимальный размер раздела был увеличен до 1 эксбибайта (1024 петабайтов), а максимальный размер файла составляет 2 Тб. По производительности новая файловая система ext4 превзошла файловые системы ext3, Reiserfs, XFS и Btrfs (в некоторых операциях).

Так, ext4 опередила знаменитую XFS в тесте на случайную запись. Файловая система Btrfs провалила этот тест с огромным «отрывом» от лидеров - XFS и ext4. Производительность ext4 была примерно такой же, как у XFS, но все-таки немного выше, чем у XFS. В Интернете вы найдете множество тестов производительностей - просмотрите их, если вам интересно.

Основной недостаток ext3 заключается в ее методе выделения места на диске. Ее способ выделения дискового пространства не отличается производительностью, а сама файловая система эффективна для небольших файлов, но никак не подходит для хранения огромных файлов. В ext4 для более эффективной организации данных используются экстенты. Экстент - это непрерывная область носителя информации. К тому же ext4 откладывает выделение дискового пространства до последнего момента, что еще более увеличивает производительность.

Файловая система ext3 может содержать максимум 32 000 каталогов, в ext4 количество каталогов не ограничено.

В журнале ext4 тоже произошли изменения - в журнале ext4 используются контрольные суммы, что повышает надежность ext4 по сравнению с ext3.

Выходит, по сравнению с ext3, у ext4 есть следующие преимущества:

- Улучшена производительность - производительность почти достигла XFS, а в некоторых тестах даже превышает ее.
- Улучшена надежность - используются контрольные суммы журналов.
- Улучшена масштабируемость - увеличен размер раздела, размер файла и поддерживается неограниченное количество каталогов.

13.11. Специальные операции с файловой системой

13.11.1. Монтирование NTFS-разделов

Не думаю, что на сервере вам придется монтировать NTFS-разделы, но ситуации бывают разные. Для монтирования NTFS-раздела используется модуль ntfs-3g, который в большинстве случаев уже установлен по умолчанию. Если он не установлен, для его установку введите команду (замените yum на имя вашего менеджера пакетов):

```
# yum install ntfs-3g
```

Команда монтирования NTFS-раздела выглядит так:

```
# mount -t ntfs-3g раздел точка_монтирования
```

Например, вам кто-то принес флешку, отформатированную как NTFS. Для ее монтирования введите команду (измените только имя устройства и точку монтирования):

```
# mount -t ntfs-3g /dev/sdb1 /mnt/usb
```

Модуль ntfs-3g выполняет монтирование в режиме чтение/запись, поэтому вы при желании можете произвести запись на NTFS-раздел.

13.11.2. Создание файла подкачки

При нерациональном планировании дискового пространства может возникнуть ситуация, когда раздела подкачки стало мало или вы вообще его не создали. Что делать? Повторная разметка диска требует времени, а выключать сервер нельзя. Сервер тормозит, поскольку ему не хватает виртуальной памяти, а ждать от начальства подписи на дополнительные модули оперативной памяти придется еще неделю. А за это время пользователи вас окончательно достанут своими жалобами.

Выход есть. Он заключается в создании файла подкачки на жестком диске. Такой файл подкачки будет работать чуть медленнее, чем раздел подкачки, но это лучше, чем вообще ничего. Хотя, если у вас SSD-диск, никакой разницы в производительности практически не будет.

Первым делом нужно создать файл нужного размера. Следующая команда создает в корне файловой системы файл swap01 размером 1 Гб:

```
# dd if=/dev/zero of=/swap01 bs=1k count=1048576
```

После этого нужно создать область подкачки в этом файле:

```
# mkswap /swap01 1048576
```

Наконец, чтобы система «увидела» файл подкачки, его нужно активировать:

```
# swapon /swap01
```

Чтобы не вводить эту команду после каждой перезагрузке сервера, нужно обеспечить ее автоматический запуск.

13.11.3. Файлы с файловой системой

Только что было показано, как создать файл произвольного размера, а потом использовать его в качестве файла подкачки. При желании этот файл можно отформатировать, как вам угодно. Даже можно создать в нем файловую систему.

Рассмотрим небольшой пример. Давайте опять создадим пустой файл размером 1 Гб:

```
# dd if=/dev/zero of=/root/fs01 bs=1k count=1048576
```

После этого нужно создать файловую систему в этом файле:

```
# mkfs.ext3 -F /root/fs01
```

Чтобы не заморачиваться, я создал самую обычную файловую систему ext3. После этого файл с файловой системой можно подмонтировать и использовать как обычных сменный носитель, то есть записывать на него файлы:

```
# mkdir /mnt/fs01
# mount -t ext3 -o loop /root/fs01 /mnt/fs01
```

После того, как закончите работу с файлом, его нужно размонтировать:

```
# umount /mnt/fs01
```

Зачем это вам нужно - знаете только вы. В конце-концов, можно использовать или зашифрованную файловую систему или просто архив. Но для общего развития это очень важно, особенно, если вы надумаете создавать собственный дистрибутив Linux.

13.11.4. Создание и монтирование ISO-образов

Все мы знаем утилиты, позволяющие в Windows подмонтировать ISO-образ диска. В Linux все подобные операции делаются с помощью штатных средств и никакие дополнительные программы не нужны.

Представим, что нам нужно создать образ диска. Если диск вставлен в привод, для создания его ISO-образа выполните команду:

```
$ dd if=/dev/cdrom of=~/dvd.iso
```

Здесь, /dev/cdrom - имя устройства (в Linux это имя соответствует любому оптическому приводу - CD или DVD), а dvd.iso - файл образа.

Иногда ставится другая задача: есть папка, по которой нужно создать ISO-образ. То есть у нас диска, но есть файлы, которые нужно записать на диск, но прежде вы хотите создать его ISO-образ.

Пусть у нас есть папка ~/dvd и нужно создать ISO-образ, содержащий все файлы из этой папки. Файл образа будет опять называться ~/dvd.iso. Для этого используйте команду mkisofs:

```
$ mkisofs -r -jcharset utf8 -o ~/dvd.iso ~/dvd
```

Чтобы проверить, что образ был создан корректно, его нужно подмонтировать к нашей файловой системе:

```
# mkdir /mnt/iso-image
# mount -o loop -t iso9660 dvd.iso /mnt/iso-image
```

Здесь все просто: опция -o loop означает, что будет монтироваться обычный файл, а не файл устройства, опция -t задает тип файловой системе, далее следуют название файла и название папки, к которой будет выполнено монтирование.

13.12. Файлы конфигурации Linux

13.12.1. Содержимое каталога /etc

Думаю, среди читателей этой книги, нет таких, которые бы не знали о реестре Windows. В реестре Windows хранятся настройки самой системы и практически всех программ (исключения составляют лишь устаревшие программы, хранящие свои параметры в ini-файлах).

Так вот, в Linux есть свое подобие реестра - это каталог /etc. И вы можете быть уверены, что в /etc находятся все настройки системы и всех программ. Да, некоторые программы также хранят персональные настройки в домаш-

них каталогах пользователя, но глобальные настройки обычно хранятся в каталоге /etc.

Преимущество каталога /etc перед реестром - в том, что для его редактирования вам не нужен какой-то особый редактор. В каталоге /etc просто хранится набор текстовых конфигурационных файлов. Вы можете просмотреть или отредактировать файл с помощью любого текстового редактора. Также можно легко скопировать файлы/каталоги, например, перед изменением. Единственный недостаток - вам нужно знать формат каждого файла, но обычно с этим проблем никаких нет, поскольку файлы снабжены подробными комментариями, путь и на английском языке.

13.12.2. Конфигурационные файлы

Чтобы эффективно настраивать систему, нужно ориентироваться в содержимом конфигурационного каталога. Если загляните в /etc, то кроме подкаталогов в нем вы обнаружите и файлы, которые находятся непосредственно в /etc. Описание этих файлов приведено в таблице 13.5. Содержимое каталога может отличаться в зависимости от используемого дистрибутива и установленного программного обеспечения. Но общая картина будет примерно одинаковой во всех дистрибутивах.

Таблица 13.5. Файлы из каталога /etc

Файл	Описание
DIR_COLORS	Конфигурация цвета для утилиты ls. Здесь вы можете указать, каким цветом будут выводиться каталоги, файлы, ссылки и т.д.
GREP_COLORS	Конфигурация цвета для утилиты grep
adjtime	Содержит параметры для корректировки аппаратных часов
aliases	База данных псевдонимов для почтовых агентов (MTA)
at.deny	Содержит пользователем, которым запрещено использовать планировщик at. Дополнительная информация доступна в man at.
anacrontab	Конфигурация (таблица расписания) планировщика anacron

bash.bashrc	Глобальный файл конфигурации оболочки bash. Локальные файлы конфигурации находятся в домашнем каталоге каждого пользователя
bind.keys	Содержит ключи для DNS-сервера bind9. Настройка DNS-сервера рассматривается в главе 23
bindresvport.blacklist	Содержит список номеров портов в диапазоне от 600 до 1024, которые не могут быть использованы bindresvport, который обычно вызывается IRC-службами. По умолчанию запрещены порты 623, 631, 636, 664, 774, 921, 993 и 995, которые используются различными сетевыми службами
cron.deny	Список пользователей, которым запрещено использовать cron
crontab	Таблица расписания демона crond
crony*	Конфигурация NTP (сервер времени)
csh.cshrc	Файл конфигурации для оболочки C Shell
csh.login	Глобальный файл конфигурации для C Shell. Определяет поведения во время регистрации пользователя (login) в системе. На самом деле - это сценарий, который запрещено редактировать вручную. Он должен изменяться только время обновления системы. Вместо этого лучше редактировать /etc/csh.login.local, если вам нужно внести изменения в настройки вашего локального окружения
crypttab	Содержит информацию о зашифрованных томах
defaultdomain	Содержит доменное имя для сервисов NIS и NIS+
dhclient.conf	Файл конфигурации для DHCP-клиента
dhclient6.conf	Файл конфигурации для DHCP-клиента, версия IPv6
dhcpd.conf	Файл конфигурации для DHCP-сервера
dhcpd6.conf	Файл конфигурации для DHCP-сервера, версия IPv6
dialogrc	Конфигурационный файл для пакета dialog, определяет цветовую схему диалоговых интерфейсов, построенных с помощью пакета dialog
dnsmasq.conf	Конфигурационный файл для dnsmasq (DNS-маскарадинга)

dracut.conf	Содержит параметры dracut - средства, которое формирует initramfs
drirc	Конфигурационный файл для репозитария DRI CVS
environment	Файл используется PAM-модулем pam_env. Содержит переменные окружения, описанные в виде пар КЛЮЧ=ЗНАЧЕНИЕ, по одной паре в одной строке
esd.conf	Параметры EsounD (Enlightened Sound Daemon), который используется для смешивания вместе некоторых цифровых аудио потоков для проигрывания на одиночном устройстве
ethers	Содержит 48-битные Ethernet-адреса и соответствующие им IP-адреса или имена узлов. Может использоваться некоторыми сетевыми службами для разрешения MAC-адресов в IP-адреса
exports	Содержит список экспортируемых файловых систем
fedora-release	Информация о релизе Fedora
filesystems	Информационный файл, содержит некоторые характеристики и атрибуты файловых систем. В нем нет ничего интересного
fstab	Содержит список файловых систем, которые будут монтироваться автоматически при загрузке системы
ftpusers	Список пользователей, которые НЕ могут войти в систему по FTP. Среди них вы найдете пользователя root и многие другие системные учетные записи, которые используются для сетевых сервисов и обычно обладают повышенными привилегиями
group	Содержит группы пользователей
host.conf	Задает порядок разрешения доменных имен
hostname	Содержит доменное имя узла

hosts	Ранее использовался для разрешения IP-адресов в доменные имена. Сейчас для этого используется система DNS, но вы все равно можете определить в нем некоторые IP-адреса, если ваша сеть не использует DNS или же вам нужно переопределить разрешение для определенного IP-адреса. Обе ситуации в наше время настолько редки, что похожи на что-то из области технической фантастики
hosts.*	Файл конфигурации локальной сети
hushlogins	Если существует файл file ~/.hushlogin или /etc/hushlogins, осуществляется «тихий» вход (это отключает проверку e-mail и вывод последнего времени входа и сообщения дня (Message of Day)). Если существует файл /var/log/lastlog, то выводится время последнего входа в систему
idmapd.conf	Конфигурация демона idmapd
idn.conf	Файл конфигурации для idnkit
idnalias.conf	Псевдонимы кодировок для idnkit
inputrc	Позволяет задавать обработку отображения символов в специальных ситуациях. Используется редко
issue	Приглашение, которое выводится при входе в систему (ссылка на /usr/lib/issue)
issue.net	Приглашение, которое выводится при сетевом входе в систему (ссылка на /usr/lib/issue.net)
krb5.conf	Параметры Kerberos
ld.so.cache	Хэш-версия файла ld.so.conf. Создается утилитой ldconfig
ld.so.conf	Настройка динамического связывания во время выполнения
lesskey	Задает параметры преобразования некоторых символов/клавиш, вы не будете редактировать этот файл
libao.conf	Конфигурация библиотеки libao (обычно здесь задается аудио-драйвер по умолчанию)
libaudit.conf	Конфигурация библиотеки libaudit

login.defs	Контрольные определения для пакета shadow. Например, здесь можно задать количество неудачных попыток входа в систему и многие другие параметры
logrotate.conf	Задает параметры ротации журналов
machine-id	Содержит идентификатор машины
mail.rc	Файл конфигурации программы mail
manpath.config	Этот файл используется пакетом man-db для настройки путей man и cat
mime.types	Содержит список MIME-типов и соответствующих им расширений файлов
mke2fs.conf	Файл конфигурации программы mke2fs
motd	Содержит сообщение дня (Message of the Day). В зависимости от настроек системы может выводиться при входе в систему
mtools.conf	Данный файл является частью пакета mtools.conf
netconfig	Файл конфигурации сети. Теперь используется только с кодом TI-RPC в библиотеке libtirc
netgroup	Содержит описание конфигурации сетевых групп
networks	Статическая информация о сетевых именах
nfsmount.conf	Файл конфигурации монтирования NFS
nscd.conf	Конфигурационный файл для nscd (Name Service Cache)
nsswitch.conf	Параметры NSS (Network Service Switch)
ntp.conf	Файл конфигурации сервера времени ntpd
os-release	Содержит информацию о релизе: номер версии, кодовое имя и т.д. (ссылка на /usr/lib/os-release)
passwd	База данных паролей. Подробно будет рассмотрен в главе 17
permissions	Этот файл используется программой chkstat и косвенно некоторыми RPM-скриптами для проверки или установки прав и режимов файлов и каталогов при установке

printcap	Этот файл автоматически генерируется cupsd, его не нужно редактировать. Все изменения, внесенные в этот файл, будут потеряны.
profile	Не изменяйте этот файл во избежание потери изменений во время очередного обновления системы. Если вам нужно изменить этот файл, отредактируйте /etc/profile.local, чтобы установить ваши локальные настройки, например, глобальные псевдонимы, переменные VISUAL и EDITOR и т.д.
protocols	Список IP-протоколов
python3start	Startup-сценарий Python 3 для сохранения истории интерпретатора и автодополнения имен
pythonstart	То же, что и python3start, но для старых версий Python
raw	Определяет параметры привязки raw-устройств к блочным устройствам
rc.splash	Сценарий, определяющий внешний вид индикатора начальной загрузки
resolv.conf	Конфигурационный файл для системы разрешения имен
rpc	Список протоколов удаленного вызова процедур (RPC)
rsyncd.conf	Файл конфигурации для rsyncd
rsyncd.secrets	Пароли rsyncd
screenrc	Параметры программы screen (менеджер экрана с эмуляцией терминала VT100/ANSI)
securetty	Содержит имена устройств терминалов (tty), на которых разрешает вход в систему пользователю root
services	Список служб (сервисов)
shadow	Пароли из файла /etc/passwd физически хранятся в /etc/shadow. Поэтому фактически в /etc/passwd хранится список пользователей, а пароли этих пользователей находятся в /etc/shadow, доступ к которому ограничен
shells	Список установленных в системе интерпретаторов

slp.conf	Файл конфигурации OpenSLP SPI
smartd.conf	Файл конфигурации демона smartd
sudoers	Позволяет определить, кому можно использовать команду sudo (см. гл. 17)
suspend.conf	Некоторые параметры питания
sysctl.conf	Файл конфигурации sysctl. Кроме этого файла sysctl также читает параметры из файлов /etc/sysctl.d/*.conf, /run/sysctl.d/*.conf и некоторых других
ttytype	Содержит список терминалов и определяет их тип
usb_ modeswitch	Параметры для пакета usb_modeswitch
vconsole.conf	Конфигурационный файл для виртуальной консоли
virc	Файл конфигурации текстового редактора vi
wgetrc	Параметры программы wget
xattr.conf	Задает, как обработать расширенные атрибуты при копировании между файлами

13.12.3. Подкаталоги с конфигурационными файлами

Далее мы «пройдемся» по подкаталогам каталога /etc с целью выяснить, что находится в каждом из них (табл. 13.6). Содержимое вашего каталога /etc может отличаться в зависимости от дистрибутива и уже установленных программ. Вполне вероятно, что у вас не будет некоторых каталогов, представленных в таблице 13.6, но будут некоторые другие каталоги, назначение которых можно найти или в справочной системе Linux или в Интернете.

Таблица 13.6. Подкаталоги каталога /etc

Каталог	Описание
NetworkManager	Содержит файл конфигурации Network Manager и файлы конфигураций сетевых соединений в некоторых дистрибутивах

PackageKit	PackageKit - это открытый и свободный набор приложений для обеспечения высокоуровнего интерфейса для различных менеджеров пакетов. Этот каталог содержит файлы конфигурации PackageKit
X11	Параметры графического интерфейса X11 (X Window)
abrt	Конфигурация abrtd - демона автоматических отчетов о сбоях
alternatives	Каталог альтернатив по умолчанию. Файл является частью подсистемы update-alternatives, которая обслуживает символические ссылки, определяющие команды, файлы и каталоги, используемые по умолчанию
audit и audisp	В этих каталогах находятся конфигурационные файлы демона аудита - auditd и его диспетчера событий (audit event dispatcher). Основной конфигурационный файл - /etc/audit/auditd.conf. В нем задается поведение демона и некоторые настройки, например, расположение журнала по умолчанию /var/log/audit/audit.log
apt	Конфигурация менеджера пакетов apt
bash_comletion.d	Параметры автодополнения командной строки для оболочки bash
binfmt.d	Демон binfmt.d настраивает дополнительные двоичные файлы для выполнения во время загрузки. В этом каталоге находятся его конфигурационные файлы
cockpit	Конфигурация панели управления Cockpit
cifs-utils	Пакет cifs-utils содержит средство монтирования ресурсов общего доступа SMB/CIFS в Linux. В этом каталоге находятся конфигурационные файлы пакета cifs-utils

crond.d	Содержит файлы, которые загружаются в память одновременно с файлами из каталога /var/spool/cron. После этого демон cron загружает содержимое файла /etc/crontab и начинает его обработку
cron.daily, cron.hourly, cron.mounthly, cron.weekly	Содержат сценарии, которые будут выполнены демоном cron, соответственно, ежедневно, ежечасно, ежемесячно и еженедельно
cups	Содержит параметры конфигурации системы печати CUPS (Common Unix Printing System)
cupshelpers	В пакете python-cupshelpers содержатся модули Python, которые помогают создавать приложения и утилиты с использованием Python-интерфейса к CUPS. В этом каталоге находятся параметры этого пакета
crypto-policies	Конфигурация крипто-политик
dbus-1	Содержит файлы конфигурации демона dbus-daemon
default	Содержит некоторые параметры по умолчанию, например, параметры загрузчика GRUB. Подробнее об этом каталоге мы поговорим в главе 15
depmod.d	depmod - программа для создания файла modules.dep и map-файла. В этом каталоге находятся ее файлы конфигурации. Как администратор сервера, вы можете о них просто забыть, они вам не пригодятся
dhcp	Конфигурация DHCP-сервера
dnf	Конфигурация менеджера пакетов dnf
dracut.conf.d	dracut заменяет mkinitrd для создания загрузочной файловой системы в оперативной памяти (ramdisk). Здесь находятся конфигурационные файлы dracut
firewalld	Конфигурация брандмауэра

fonts	Содержит конфигурационные файлы подсистемы шрифтов. В частности, файл /etc/fonts/fonts.conf описывает каталоги со шрифтами, каталоги с кэшем шрифтов, а также описывает аналоги шрифтов
gnupg	Содержит конфигурационный файл GnuPg
grub.d	Содержит файлы, относящиеся к загрузчику GRUB (гл. 15). Вам не нужно редактировать эти файлы, они предназначены для служебных целей
init.d	Содержит сценарии системы инициализации
iproute2	Параметры подсистемы маршрутизации. Например, могут использоваться для IP-балансировки, то есть объединения нескольких Интернет-каналов в один
iscsi	Параметры iSCSI
java	Параметры Java
joe	Содержит конфигурационные файлы текстового редактора joe
jvm, jvm-common	Параметры виртуальной машины Java
ld.so.conf.d	Содержит файлы с расширением conf, которые используются для поиска разделяемых библиотек
libnl	Конфигурационные файлы библиотеки libnl
logrotate.d	Конфигурация средства ротации журналов
mc	Файлы конфигурации файлового менеджера Midnight Commander
mcelog	Программа mcelog позволяет расшифровать аппаратные ошибки. Настраивается посредством конфигурационных файлов в каталоге /etc/mcelog

modprobe.d	modprobe — программа для добавления и удаления модулей из ядра Linux. Соответственно в одноименном подкаталоге каталога /etc находятся ее конфигурационные файлы
modules-load.d	Содержит параметры некоторых модулей ядра. По умолчанию в этом каталоге пусто
openldap	Содержит конфигурационные файлы сервера каталогов OpenLDAP
opt	Файлы конфигурации для /opt/
pam.d	Параметры конфигурации модулей аутентификации PAM (Pluggable Authentication Modules)
pkcs11	Параметры программы pkcs11, используемой для управления объектами данных, которые находятся на зашифрованных устройствах PKCS#11 (Cryptoki)
pki	Содержит список GPG-ключей
plymouth	Plymouth — свободный графический экран загрузки для Linux. Этот каталог содержит его конфигурационные файлы
pm	Содержит конфигурационные файлы пакета pm-utils. Пакет pm-utils - это инфраструктура управления питанием нового поколения
postfix	Конфигурационные файлы почтового агента Postfix
ppp	Файлы конфигурации протокола PPP
pptp.d	Файлы конфигурации демона pptpd (протокол PPTP)
products.d	Относится к системе миграции между версиями дистрибутива openSUSE. Дополнительная информация может быть найдена по ссылке http://doc.opensuse.org/products/draft/SLES/SLES-deployment_sd_draft/cha.update.sle.html

pulse	PulseAudio - это звуковой сервер для POSIX-систем. Его основное назначение - смешивать звуковые потоки от разных приложений, что позволяет нескольким потокам воспроизводиться одновременно. Здесь находятся конфигурационные файлы PulseAudio
rc.d	Ссылка на каталог init.d
rsyslog.d	Конфигурация демона протоколирования rsyslogd
rpm	Различные параметры системы управления пакетами RPM. Обычно файлы из этого каталога не требуют изменения. Дополнительную информацию можно получить по адресу http://wiki.opennet.ru/RPM
samba	Конфигурационные файлы Samba, будут рассмотрены в главе 25
sasl2	Параметры SASL (Simple Authentication and Security Layer)
security	Еще один параметр конфигурации, относящийся к модулям аутентификации PAM
selinux	Содержит файлы конфигурации системы безопасности SELinux
skel	При создании новой учетной записи пользователя создается его домашний каталог в каталоге /home, при этом в созданный домашний каталог пользователя копируются файлы из каталога /skel. Все помещенные в этот каталог файлы будут скопированы в созданный домашний каталог
ssh	Содержит файлы конфигурации SSH-клиента и SSH-сервера
ssl	Файлы конфигурации OpenSSL

sudoers.d	Кроме файла /etc/sudoers, настройки sudo могут определяться содержимым файлов из каталога /etc/sudoers.d
sysconfig	Содержит конфигурационные файлы всей системы. В этом каталоге очень много различных конфигурационных файлов. Например, в каталоге /etc/sysconfig/network вы найдете конфигурационные файлы сетевых интерфейсов. А в файле clock хранится выбранный при установке часовой пояс
sysctl.d	sysctl.d настраивает параметры ядра при загрузке, здесь находятся его конфигурационные параметры
systemd	Конфигурационные файлы демона systemd
udev	Файлы конфигурации и файлы правил менеджера устройств udev
wpa_supplicant	Конфигурационные файлы пакета wpa_supplicant (обеспечивает поддержку WEP, WPA и WPA2)
xdg	xdg-open - это независимый пользовательский инструмент для настройки приложений рабочего стола по умолчанию. В этом каталоге находятся его конфигурационные файлы. Например, в каталоге /etc/xdg/autostart находятся все программы, которые могут быть запущены автоматически. Однако запускаются лишь те, которым разрешен запуск в определенной сессии
xinetd.d	Содержит дополнительные файлы конфигурации суперсервера xinetd (в современных дистрибутивах не используется)
xml	Конфигурационные файлы библиотеки libxml

13.13. Псевдофайловые системы

Псевдофайловые системы **sysfs** (каталог /sys) и **proc** (каталог /proc) используются для настройки системы и получения различной информации о системе и процессах. Свое название псевдофайловые системы получили из-

за того, что они работают на уровне виртуальной файловой системы. В итоге оба эти средства (назовем их так) для конечных пользователей выглядят как обычная файловая система - вы можете зайти как в каталог /sys, так и в каталог /proc. В обоих этих каталогах будут файлы, вы можете просмотреть эти файлы и даже изменить их содержимое.

Содержимое многих файлов псевдофайловой системы /proc формируется «на лету». Обратите внимание на размер любого файла в каталоге /proc - он равен нулю, но если открыть файл, то информация в нем будет. Например, в файле /proc/version находится информация о версии Linux.

Монтирование файловых систем **sysfs** и **proc** осуществляется или в сценариях инициализации системы или через /etc/fstab. В последнем случае записи монтирования псевдофайловых систем выглядят так:

```
sysfs     /sys       sysfs    defaults        0 0
proc      /proc      proc     defaults        0 0
```

13.13.1. Псевдофайловая система sysfs

Файловая система sysfs (каталог /sys) предоставляет пользователю информацию о ядре Linux, об имеющихся в системе устройствах и драйверах этих устройств. На рис. 13.4 представлено содержание каталога /sys. В нем вы найдете следующие подкаталоги:

- **block** - содержит каталоги для всех блочных устройств, которые есть в вашей системе в настоящее время. Здесь под устройством подразумевается наличие физического устройства и его драйвера. Если вы подключите внешний жесткий диск, то в каталоге /sys/devices появится новое устройство, но в каталоге /sys/block оно появится только, если в системе есть драйвер для работы с этим устройством или же драйвер (модуль) встроен в само ядро;
- **bus** - здесь находится список шин, которые поддерживает ваше ядро. Заглянув в этот каталог, вы обнаружите подкаталоги pci, pci_express, scsi и т.д. В каждом из этих каталогов будут подкаталоги devices и drivers. В первом находится информация об устройствах, подключенных к данной шине, во втором - информация о драйверах устройств;
- **class** - позволяет понять, как устройства формируются в классы. Для каждого класса есть отдельный подкаталог в каталоге class;
- **devices** - содержит дерево устройств ядра, точнее структуру файлов и каталогов, которая полностью соответствует внутреннему дереву устройств ядра;

- **firmware** - содержит интерфейсы, предназначенные для просмотра и манипулирования firmware-специфичными объектами и их параметрами;
- **fs** - информация о файловых системах, которые поддерживает ваше ядро;
- **kernel** - общая информация о ядре;
- **module** - здесь вы найдете подкаталоги для каждого загруженного модуля ядра. Имя подкаталога соответствует имени модуля. В каждом из подкаталогов модулей вы найдете подкаталог parameters, содержащий специфичные для модуля параметры;
- **power** - позволяет управлять параметрами питания, а также переводить систему из одного состояния питания в другое. Далее будет показано несколько примеров.

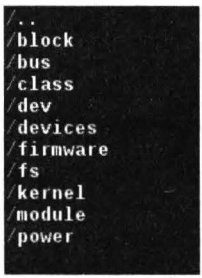


Рис. 13.4. Содержание каталога /sys

Довольно интересен с практической точки зрения каталог /sys/power. В файле state находится состояние питание. Изменив должным образом содержимое этого файла, можно изменить состояние питания. Например, вот как можно перевести систему в состояние «Suspend to RAM», когда питание процессора отключается, но питание на память подается, благодаря чему ее содержимое не уничтожается:

```
$ sudo echo -n mem > /sys/power/state
```

При желании можно отправить систему в состояние «Suspend to Disk», когда содержимое памяти будет записано на жесткий диск, после чего питание будет отключено:

```
$ sudo echo -n disk > /sys/power/state
```

13.13.2. Псевдофайловая система proc

Файловая система **proc** позволяет отправлять информацию ядру, модулям и процессам. Вы можете не только получать информацию о процессах, но

изменять параметры ядра и системы «на лету». Эта файловая система интересна тем, что позволяет изменять такие параметры ядра, которые невозможно изменить другим способом, к тому же вносимые изменения вступают в силу сразу же.

Некоторые файлы в /proc доступны только для чтения - вы можете только просмотреть их. А некоторые вы можете изменять, и эти изменения сразу же отразятся на работе системы. Просмотреть файлы из /proc можно любой программой для просмотра файлов, проще всего в консоли использовать команду **cat**:

```
cat /proc/<название файла>
```

Записать информацию в файл можно с помощью команды echo, как уже было показано выше:

```
sudo echo «информация» > /proc/<название файла>
```

В каталоге /proc очень много файлов и рассмотреть все мы не сможем. Прежде, чем мы приступим к самым интересным с моей точки зрения файлам, нужно понять, что означают каталоги с числами. Эти каталоги содержат информацию о запущенных процессах.

Итак, самые полезные информационные файлы:

- /proc/cmdline - содержит параметры, переданные ядру при загрузке;
- /proc/cpuinfo - содержит информацию о процессоре, откройте этот файл, думаю, вам будет интересно. Кроме общей информации о процессоре вроде модели и частоты здесь выводится точная частота, размер кэша и псевдорейтинг производительности, выраженный в BogoMIPS. Значение BogoMIPS показывает «сколько миллионов раз в секунду компьютер может абсолютно ничего не делать». Способ измерения производительности пусть и не самый удачный, но от него до сих пор не отказались, а «на дворе» уже 3-я версия ядра;
- /proc/devices - список устройств;
- /proc/filesystems - полный список поддерживаемых вашим ядром файловых систем;
- /proc/interrupts - информация по прерываниям;
- /proc/ioports - информация о портах ввода/вывода;
- /proc/meminfo - полная информация об использовании оперативной памяти. Как по мне, вывод этого файла более понятен и удобен, чем вывод команды free;

- /proc/mounts - содержит список подмонтированных файловых систем;
- /proc/modules - список загруженных модулей и их параметры;
- /proc/swaps - содержит список активных разделов и файлов подкачки;
- /proc/version - здесь находится версия ядра.

Используя /proc можно не только получить информацию о системе, но и изменить ее. Например, в файлах /proc/sys/kernel/hostname и /proc/sys/kernel/domainname содержится информация об имени компьютера и домена. Вы можете не только просмотреть, но и изменить содержимое этих файлов, изменив, соответственно, имя узла и имя домена. Хотя практика изменения доменных имен через /proc/sys практикуется не часто, никто не мешает вам это сделать:

```
sudo echo "server" > /proc/sys/kernel/hostname
sudo echo "example.com" > /proc/sys/kernel/domainname
```

Файл /proc/sys/kernel/ctrl-alt-del позволяет регулировать тип перезагрузки системы при нажатии комбинации клавиш Ctrl + Alt + Del. По умолчанию в этом файле содержится значение **0**, что означает так называемую «мягкую перезагрузку» (soft reboot). Если же вы внесете в этот файл значение **1**, то при нажатии Ctrl + Alt + Del эффект будет такой же, как при нажатии кнопки **Reset** на корпусе компьютера:

```
sudo echo "1" > /proc/sys/kernel/ctrl-alt-del
```

Файл /proc/sys/kernel/printk позволяет задать, какие сообщения ядра будут выведены на консоль, а какие - попадут в журнал демона syslog. По умолчанию в этом файле содержатся значения 4 4 1 7. Сообщения с приоритетом 4 и ниже (первая четверка) будут выводиться на консоль. Вторая четверка - это уровень приоритета по умолчанию. Если для сообщения не задан уровень приоритета, то считается, что его приоритет будет равен 4.

Третье значение определяет номер самого максимального приоритета. Последнее значение - это уровень приоритета по умолчанию для первого значения.

В большинстве случаев изменяют только первое значение, позволяющее определить, будет ли сообщения с указанным уровнем приоритета выводиться на консоль или нет. Остальные параметры оставляют без изменения.

В файле /proc/sys/net/core/netdev_max_backlog содержится максимальное число пакетов в очереди. Значение по умолчанию - 1000.

Файл /proc/sys/fs/file-max позволяет изменить максимальное количество заголовков файлов, которое может быть одновременно открыто. Другими словами, этот файл задает, сколько одновременно может быть открыто файлов. Значение по умолчанию для ядра 3.16 и файловой системы btrfs - 73054.

Чтобы сохранить внесенные «на лету» изменения, и чтобы их не пришлось снова вводить при следующей перезагрузке сервера, нужно отредактировать файл /etc/sysctl.conf. Представим, что вы изменили значение из файла /proc/sys/fs/file-max. Тогда в файл /etc/sysctl.conf нужно добавить строку:

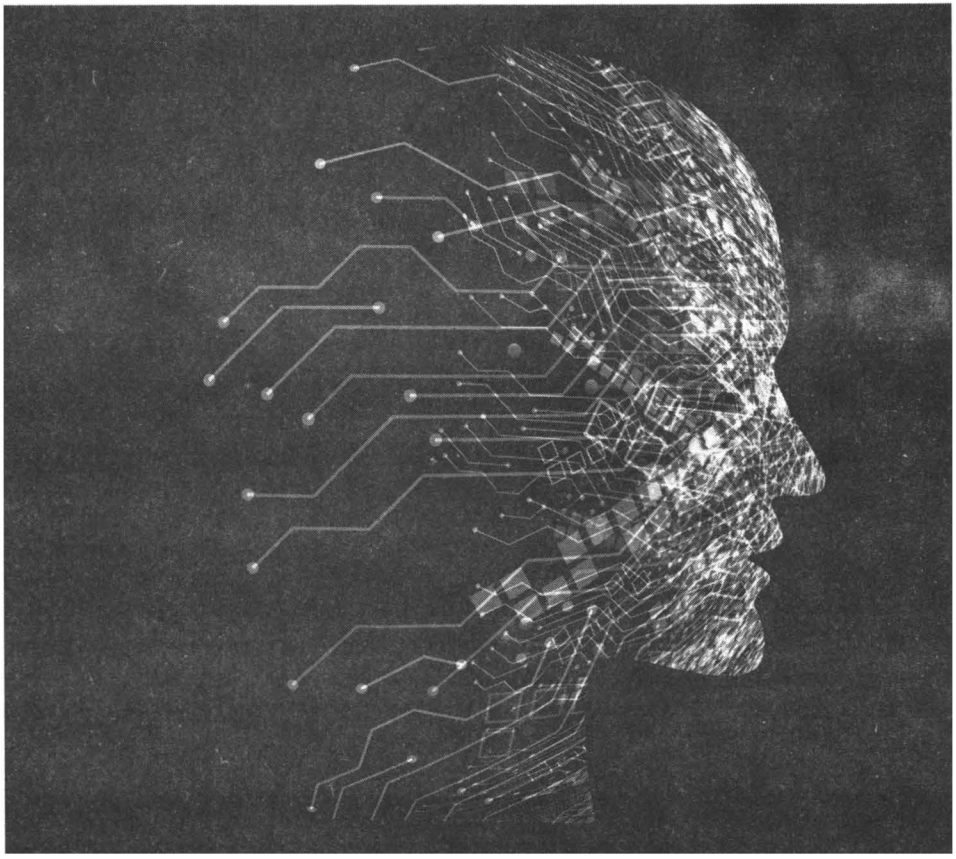
```
fs.file-max = 16 384
```

Принцип прост: /proc/sys/ отбрасывается совсем, а в оставшейся строке все слеши заменяются точками. Само же значение указывается через знак равенства. Если нужно указать несколько значений, то они указываются через пробел.

О файловой системе в Linux можно написать отдельную книгу, которая будет не меньше, чем ту, которую вы держите в руках. Поэтому в этой главе мы рассмотрели только самое необходимое.

Глава 14.

Управление хранилищем



В этой главе мы рассмотрим довольно таки важные вещи – подключение нового жесткого диска и его разметка в классическом варианте – без всяких менеджеров томов, затем мы рассмотрим LVM и то, как можно расширить пространство группы томов, например, когда вы увеличили дисковое пространство виртуального сервера.

14.1. Подключение нового жесткого диска и его разметка

Классической программой для разметки жесткого диска в Linux и других операционных системах является программа **fdisk**. Конечно, в той же Windows программа fdisk совсем другая, но названия программ совпадают.

Рассмотрим пример использования этой программы. Представим, что мы подключили новый жесткий диск и нам нужно «ввести» его в эксплуатацию. Тренироваться лучше всего в виртуальной машине, особенно, если вы в первый раз осуществляете разметку диска.

Формат вызова **fdisk** такой:

```
# fdisk <устройство>
```

Да, команду fdisk нужно запускать с правами root. Далее мы будем считать, что новым является устройство /dev/sdb:

```
# fdisk /dev/sdb
```

Посмотрите на рис. 14.1. Я запустил программу fdisk для нового и неразмеченного жесткого диска. Программа сообщила мне, что:

1. Все изменения хранятся только в памяти и не переносятся на жесткий диск до тех пор, пока вы их не запишите.
2. Устройство не содержало таблицы разделов и была создана таблица разделов DOS (по умолчанию).

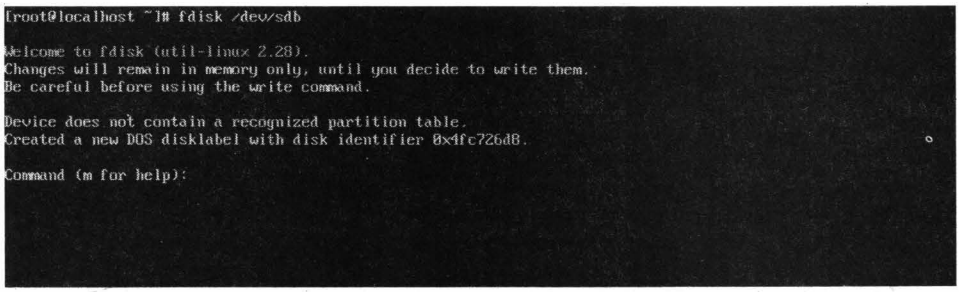


Рис. 14.1. Запуск fdisk для нового жесткого диска

Первым делом ознакомимся со списком команд `fdisk`. Введите команду m для получения справки. Список команд в последних версиях `fdisk` разбит на группы (рис. 14.2). В таблице 14.1 приведен список команд `fdisk`.

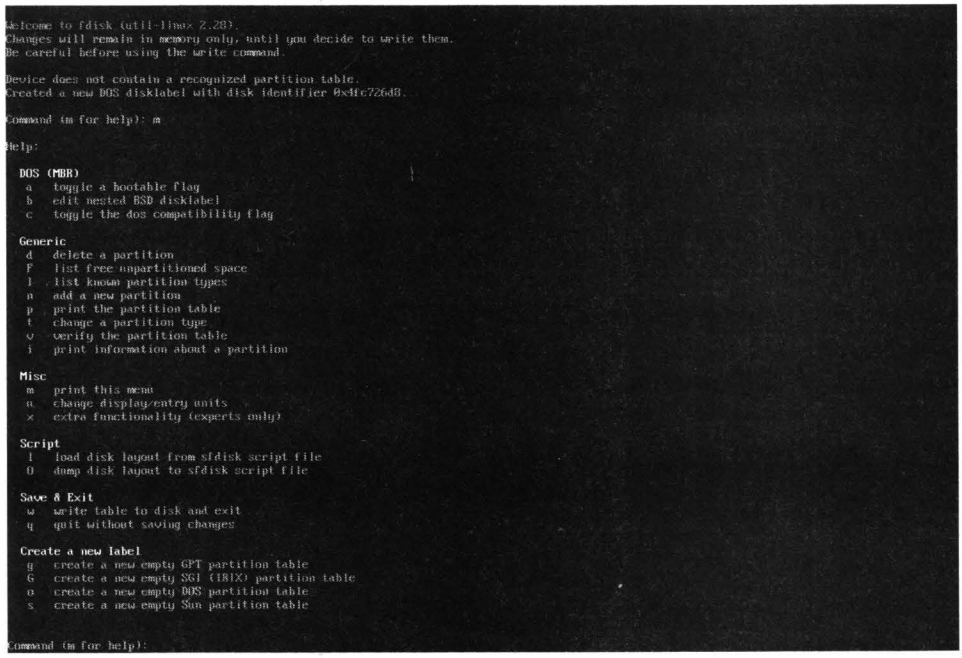


Рис. 14.2. Список команд fdisk

Таблица 14.1. Команды программы fdisk

Команда	Описание
a	Сделать раздел активным. Данный флаг был нужен для старых версий Windows, которые не могли загружаться с неактивных разделов. Сейчас эта команда попросту не нужна, а в мире Linux - тем более
b	Редактировать вложенную BSD-метку
c	Применить флаг совместимости с DOS
d	Удалить раздел
l	Вывести известные типы разделов
n	Добавить новый раздел
p	Вывод таблицы разделов
t	Изменить тип раздела
v	Проверить таблицу разделов
m	Вывод справки
u	Изменить единицы измерения
x	Дополнительная функциональность (только для экспертов)
w	Записать таблицу разделов на диск и выйти
q	Выход без сохранения изменении
g	Создать новую пустую таблицу разделов GPT
G	Создать новую пустую таблицу разделов SGI (для ОС IRIX)
o	Создать новую пустую таблицу разделов DOS
s	Создать новую пустую таблицу разделов Sun

Наша задача - создать раздел (или несколько разделов, здесь решать вам) и подмонтировать его (их) к корневой файловой системе.

Первым делом выведем таблицу разделов командой p (рис. 14.3). Как видно из рис. 14.3, таблица разделов пуста, а размер нашего жесткого диска всего 60 Гб.

```
Command (m for help): p
Disk /dev/sdb: 60 GiB, 64424509440 bytes, 125829120 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: 0x4fc726d8

Command (m for help):
```

Рис. 14.3. Пустая таблица разделов

Наш диск довольно скромного размера, поэтому программа создала таблицу разделов DOS. Для больших жестких дисков лучше создать таблицу разделов GPT. Если программа неправильно выбрала тип таблицы разделов или вы хотите изменить его принудительно, введите команду g. Посмотрите на рис. 14.4: я изменил тип таблицы разделов, а затем опять отобразил таблицу разделов. Она по-прежнему пуста, но обратите внимание на ее тип - теперь у нас таблица разделов GPT.

> Примечание. Таблица разделов GPT (GUID Partition Table) является частью стандарта EFI (Extensible Firmware Interface) - стандарта, который был предложен компанией Intel на смену стандарта BIOS. Таблица GPT использует современную систему адресации логических блоков (LBA), а не старую систему CHS (цилиндр-головка-сектор). Но самое главное - это размер раздела. В GPT можно создать раздел размером до 9.4 Збайт (9.4×10^{21} байт), а в MBR - максимальный размер раздела всего 2.2 Тб (2.2×10^{21} байт).

```
Command (m for help): g
Created a new GPT disklabel (GUID: 299E41E1-4AB9-42E6-911F-C39BC06E7DE3).

Command (m for help): p
Disk /dev/sdb: 60 GiB, 64424509440 bytes, 125829120 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: gpt
Disk identifier: 299E41E1-4AB9-42E6-911F-C39BC06E7DE3

Command (m for help):
```

Рис. 14.4. Изменение таблицы разделов

Настало время создать раздел. Введите команду n. Программа попросит вас ввести (рис. 14.5):

- Номер раздела - это первый раздел, поэтому введите 1. В принципе, когда вы будете создавать второй раздел, программа автоматически предложит вам ввести номер 2.

- Первый сектор раздела. Просто нажмите **Enter** - программа автоматически предложит правильный вариант.
- Последний сектор раздела. Если вы хотите создать раздел на весь жесткий диск (то есть использовать все доступное пространство), тогда просто нажмите **Enter**. В противном случае укажите размер раздела. Проще всего это сделать, используя модификаторы +M и +G, например, для создания раздела размером 30 Гб укажите +30G. Если у вас очень большой жесткий диск, где пространство измеряется терабайтами, используйте модификатор T, например, +1T.

```
Command (m for help): n
Partition number (1-128, default 1): 1
First sector (2048-125829086, default 2048):
Last sector, +sectors or +size{K,M,G,T,P} (2048-125829086, default 125829086):

Created a new partition 1 of type 'Linux filesystem' and of size 60 GiB.

Command (m for help):
```

Рис. 14.5. Создание нового раздела

Программа сообщит вам, что один раздел создан. Также будет сообщен размер раздела. По умолчанию тип раздела - файловая система Linux (Linux filesystem). Если вы хотите изменить тип раздела, введите команду:

```
t <номер раздела>
```

Например:

```
t 1
```

Далее нужно или ввести код типа раздела или ввести команду **L** для вывода подсказки (рис. 14.6). Проблема вся в том, что нет способа постраничного просмотра типов разделов, а все они не помещаются на одном экране. Поэтому все равно придется обращаться к документации. Однако в большинстве случаев изменять тип раздела не нужно, поскольку при создании раздела он создается уже нужного типа. Исключение может возникнуть разве что для раздела подкачки (Linux swap). Его код - 82.

```
68 MidnightBSD data             0505E456-237C-11E1-B4B3-E89A8F7FC3A7
69 MidnightBSD boot             0505E45E-237C-11E1-B4B3-E89A8F7FC3A7
70 MidnightBSD swap             0505E45B-237C-11E1-B4B3-E89A8F7FC3A7
71 MidnightBSD UFS              0394EF8B-237E-11E1-B4B3-E89A8F7FC3A7
72 MidnightBSD ZFS              0505E45D-237C-11E1-B4B3-E89A8F7FC3A7
73 MidnightBSD Vinum            0505E45C-237C-11E1-B4B3-E89A8F7FC3A7
74 Ceph Journal                 45B0969E-9B03-4F30-B4C6-B4B80CEFF106
75 Ceph Encrypted Journal       45B0969E-9B03-4F30-B4C6-5EC00CEFF106
76 Ceph OSD                     4FBD7E29-9D25-41B8-AFD0-062C0CEFF05D
77 Ceph crypt OSD               4FBD7E29-9D25-41B8-AFD0-5EC00CEFF05D
78 Ceph disk in creation        89C57F98-2FE5-4DC0-89C1-F3AD0CEFF2BE
79 Ceph crypt disk in creation  89C57F98-2FE5-4DC0-89C1-5EC00CEFF2BE
80 OpenBSD data                 824CC7A0-36A8-11E3-890A-952519AD3F61
81 QNX6 file system             CEF5A9AD-73BC-4601-89F3-CDEEEEE321A1
82 Plan 9 partition             C91818F9-8025-47AF-89D2-F030D7000C2C

Hex code (type L to list all codes):
```

Рис. 14.6. Подсказка по типу раздела

Если вы передумали менять тип раздела, просто нажмите **Enter**. Теперь введите p для просмотра нашей таблицы разделов. Всегда просматривайте таблицу перед ее записью. После этого введите команду w для сохранения изменений и выхода из программы (рис. 14.7).

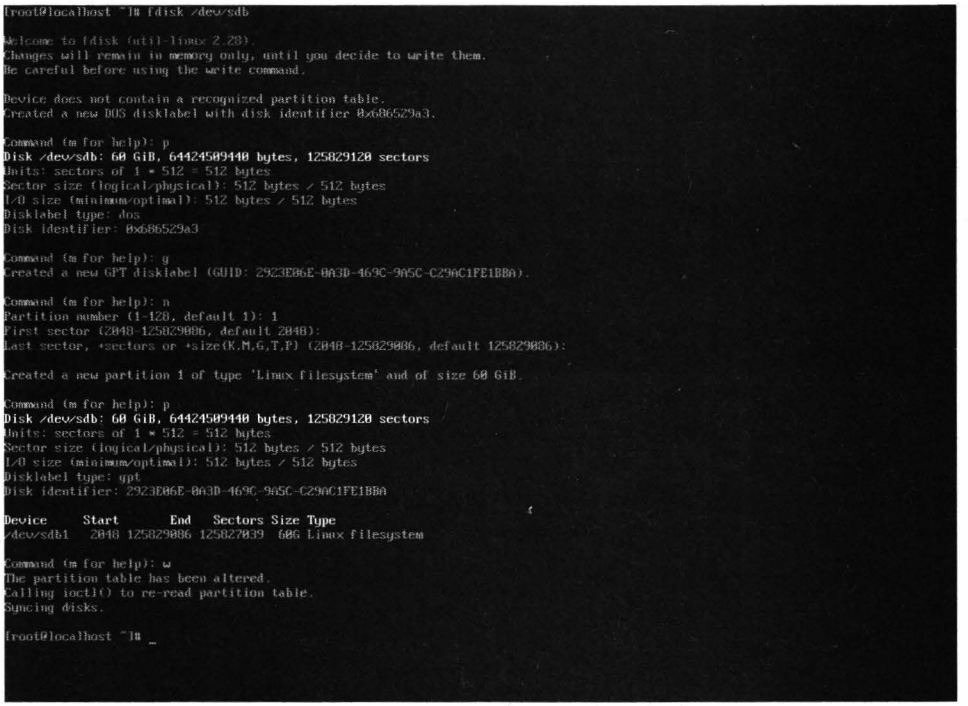


Рис. 14.7. Весь сеанс разметки диска: от создания таблицы разделов до записи изменений

Создать раздел мало. Нужно еще создать файловую систему. Не будем ничего выдумывать и создадим стандартную файловую систему ext4 командой mkfs.ext4:

```
# mkfs.ext4 /dev/sdb1
```

Результат выполнения этой команды приведен на рис. 14.8.

После этого нужно создать точку монтирования для нового раздела и подмонтировать раздел (название точки монтирования можете изменить по своему усмотрению):

```
# mkdir /mnt/sdb1
# mount /dev/sdb1 /mnt/sdb1
# ls /mnt/sdb1
```

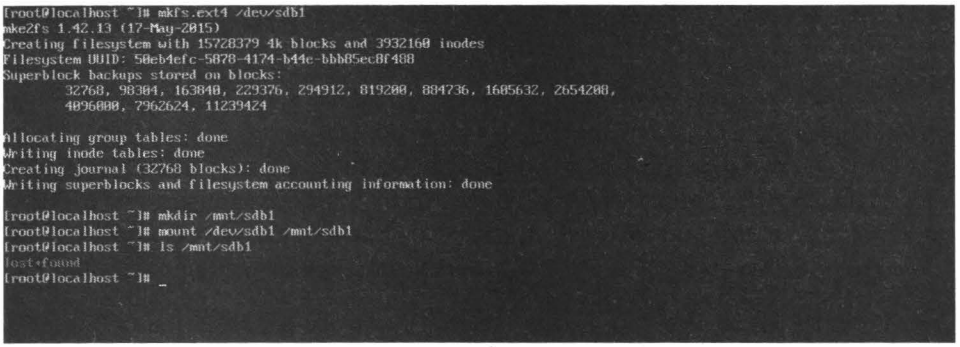


Рис. 14.8. Создание файловой системы и монтирование жесткого диска

Почти все готово. Осталось только добавить запись в /etc/fstab для автоматического монтирования созданного раздела:

```
/dev/sdb1 /mnt/sdb1 ext4 defaults    1 1
```

Вот теперь можно приступать к использованию нового жесткого диска.

14.2. Менеджер логических томов

14.2.1. Введение в LVM

Впервые менеджер логических томов (LVM, Logical Volume Manager) появился в ядре 2.4 (его первая версия LVM 1), но более активно он стал применяться только в дистрибутивах с ядром 2.6 (уже вторая версия LVM 2).

Некоторые современные дистрибутивы используют LVM по умолчанию создают пулы LVM, в некоторых же LVM даже не установлен по умолчанию. Чтобы понять, нужен ли вам LVM, нужно разобраться, что это такое. В этой главе все будет изложено максимально доступно, так что бояться использовать LVM не стоит. В тоже время, если вам необходимы точные академические определения и дополнительная информация, обратитесь к Linux LVM Howto (http://tldp.org/HOWTO/LVM-HOWTO/), в котором много технических подробностей (если они вам нужны) и не совсем все сразу понятно.

LVM - это дополнительный уровень абстракции над аппаратными средствами, позволяющий собрать вместе несколько дисков в один логический диск, а затем разбить его так, как вам хочется.

Примеров использования LVM множество. Самый простой из них - объединение нескольких небольших дисков в один диск большего размера. Например, вам досталось даром (или почти даром) несколько SSD-дисков не-

большого размера, скажем, по 128 Гб, и вы хотите объединить эти 2-3 диска, чтобы получить один большой диск 256-384 Гб.

Второй пример тоже часто распространен. Представим, что система у вас установлена на небольшом диске, пусть даже на том же SSD-диске размером 80 Гб. Для Linux такой объем вполне достаточен, но рано или поздно свободное место закончится (все зависит от файлов, с которыми вы работаете). Вы покупаете диск большего размера, скажем, на 500 Гб или даже на 1 Тб. Но система уже установлена и переустанавливать ее не хочется. Что делать? Здесь вам поможет LVM.

Первый приведенный мною пример (объединение трех дисков во время установки) слишком тривиален и с ним справится любой, даже самый начинающий пользователь - просто во время установки нужно выбрать LVM (если, конечно, дистрибутив его поддерживает) и выбрать диски, которые вы объединяете в группу (пул).

Второй пример более сложный только за счет того, что система уже установлена, и мы договорились, что переустанавливать ее не будем. Поэтому он и заслуживает рассмотрения в этой главе, а дополнительную информацию вы сможете найти в Linux VVM Howto, ссылка на который была приведена ранее.

14.2.2. Уровни абстракции LVM

Прежде, чем перейти к рассмотрению практической стороны вопроса, нужно разобраться с тремя уровнями абстракции, с которыми вам придется столкнуться в LVM. Вот эти уровни:

1. **PV** (Physical Volume) – физические тома. Это могут быть разделы или целые, еще не размеченные диски.
2. **VG** (Volume Group) – группа томов. Физические тома объединяются в группу и создается единый диск, который вы можете разбить так, как вам хочется.
3. **LV** (Logical Volume) - логический раздел. Это раздел нашего единого диска (VG), который вы можете отформатировать в любую файловую систему и использовать так, как вам хочется, как обычный раздел обычного жесткого диска.

Прежде, чем мы продолжим, вы должны знать об одном недостатке LVM. Тома LVM не поддерживаются загрузчиком GRUB. Поэтому если вы используете этот устаревший загрузчик, вам нужно создать отдельный раздел /boot за пределами LVM. Грубо говоря, если у вас есть диск /dev/sda,

раздел /dev/sda1 должен монтироваться к /boot. В него будет установлен загрузчик. Размер этого раздела должен быть небольшой, примерно 100 Мб (вполне будет достаточно).

Что же касается GRUB2, то он нормально загружается с LVM и никакой дополнительный раздел создавать не нужно.

14.2.3. Немного практики

Первым делом вам нужно установить пакет lvm2, если он еще не установлен.

```
# apt-get install lvm2
# yum install lvm2
```

Команда установки этого пакета для Fedora/CentOS приведена на всякий случай, так как в большинстве случаев этот пакет в этих дистрибутивах установлен по умолчанию.

Будем считать, что система сейчас установлена на /dev/sda1, который подмонтирован как /. Больше никаких разделов на этом диске не создано - для упрощения примера. Мы подключили второй жесткий диск, который пока еще не разбит. Имя этого диска - /dev/sdb.

Если на втором диске не созданы разделы, то создавать их и не нужно. Вы можете сдать все устройство сразу физическим томом (PV). Если на нем есть разделы - не беда, вы можете добавить в группу томов все разделы поочередно.

Тратить время на создание разделов не хочется, тем более, что это и не нужно, поэтому создаем PV на все устройство /dev/sdb. Для этого используется команда pvcreate:

```
# pvcreate /dev/sdb
Physical volume "/dev/sdb" successfully created
```

Теперь нужно создать группу томов с помощью команды vgcreate. Данной команде нужно передать имя группы (пусть это будет my_vg) и указать физическое устройство:

```
# vgcreate my_vg /dev/sdb
Volume group «vg0» successfully created
```

Создаем отдельные логические тома (команда lvcreate) для раздела подкачки (swap) и разделов /home, /tmp и /var. Параметр -L задает размер раздела,

например, -L30G задает размер 30 Гб. Параметр -n задает имя логического тома:

```
# lvcreate -n swap -L8G my_vg
# lvcreate -n home -L500G my_vg
# lvcreate -n var -L30G my_vg
# lvcreate -n tmp -L5G my_vg
```

Последний параметр - это имя нашей группы. При желании можно создать отдельный раздел и для /usr, но, как правило, программное обеспечение (а оно в основном устанавливается в /usr) в Linux много места не занимает. В общем, смотрите сами - никто не мешает ввести еще одну команду lvcreate. Тем более что у нас еще осталось место (если учитывать, что у нас жесткий диск на 1 ТБ).

Теперь разберемся, что и где мы создали. Просмотреть информацию по физическим томам, группам томов и логическим разделам можно с помощью команд pvdisplay, vgdisplay и lvdisplay соответственно.

Созданные нами разделы будут храниться в папке /dev/my_vg/. В этом каталоге вы найдете файлы home, tmp, var (правда, это будут ссылки, а не файлы, но суть от этого не меняется).

Когда созданы логические тома, можно создать на них файловые системы (отформатировать их). Вы можете использовать любую файловую систему, я предпочитаю ext4:

```
# mkfs.ext4 -L var /dev/my_vg/var
# mkfs.ext4 -L home /dev/my_vg/home
# mkfs.ext4 -L tmp /dev/my_vg/tmp
# mkswap -L swap /dev/my_vg/swap
# swapon /dev/my_vg/swap
```

Первые три команды создают файловую систему ext4 на устройствах /dev/my_vg/var, /dev/my_vg/home и /dev/my_vg/tmp. Последние две создают раздел подкачки и активируют его.

Настало время заняться перемещением данных. Суть в следующем - нужно подмонтировать поочередно новые тома и скопировать в них содержимое /home и /var:

```
# mkdir /mnt/home
# mkdir /mnt/var

# mount /dev/my_vg/home /mnt/home
# mount /dev/my_vg/var /mnt/var
```

```
# cp -a /home/* /mnt/home
# cp -a /var/* /mnt/var

# umount /mnt/home
# umount /mnt/var
```

В папку /tmp копировать ничего не нужно. Нужно только изменить права доступа:

```
# mkdir /mnt/tmp
# mount /dev/my_vg/tmp /mnt/tmp
# chmod -R a+rwx /mnt/tmp
# umount /tmp
```

Почти все. Теперь нужно добавить в /etc/fstab записи, монтирующие файловые системы /home, /var, /tmp и указать в нем раздел подкачки:

```
/dev/mapper/my_vg-home    /home  ext4 relatime      1 1
/dev/mapper/my_vg-home    /var         ext4 relatime      1 1
/dev/mapper/my_vg-tmp     /tmp         ext4 noatime       0 2
/dev/mapper/my_vg-swap    none         swap sw            0 0
```

Все готово. Осталось только ввести команду reboot, чтобы система перезагрузилась. Корневая файловая система осталась на старом жестком диске (как и /usr), а каталоги, которые занимают больше всего места, были перемещены на логические разделы LVM.

14.3. Расширение LVM-пространства

Предположим, что вы расширили дисковое пространство виртуального сервера. Однако одного расширения в панели управления недостаточно – чтобы система увидела изменения, нужно произвести определенные действия. В принципе, задачу расширения пространства сервера можно было решить иначе, например, добавить еще один виртуальный жесткий диск, а дальше или использовать классический способ (раздел 14.1) или же добавить диск в группу томов (раздел 14.2) – все зависит от того, как настроен операционная система виртуального сервера. Но случилось то, что случилось – вы уже расширили жесткий диск, а вернуть ресурсы в пул, как правило, нельзя. Поэтому рассмотрим процедуру расширения виртуального диска.

Первым делом, посмотрим, сколько сейчас дискового пространства доступно. Для этого используется уже известная команда df с параметром –h, чтобы вывод был в удобочитаемом формате:

```
df -h
```

```
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/vgroup1-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. 14.9. Команда df –h

На данный момент общий размер группы томов /dev/mapper/vgroup1-root составляет 19 Гб. Однако мы расширили диск и теперь нам нужно расширить размер этой группы томов до полного размера диска.

Чтобы система увидели новый объем жесткого диска, нужно пересканировать аппаратную конфигурацию. Для этого мы будем использовать следующую команду:

```
echo 1 > /sys/block/sda/device/rescan
```

Запустите утилиту **parted** (используется для работы с разделами диска):

```
parted
```

Введите команду p для просмотра имеющихся разделов (рис. 14.10). Запомните номер раздела, который мы будем расширять (2) и новый размер диска (42.9GB).

```
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
Disk 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. 14.10. Текущая таблица разделов

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.

```
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
Disk 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. 14.11. Изменение размера раздела

Введите команду **quit** для выхода из parted. Parted сделал свою работу. Осталось сообщить ядру об изменениях размера:

```
pvresize /dev/sda2
  Physical volume "/dev/sda2" changed
  1 physical volume(s) resized / 0 physical volume(s) not
resized
```

Расширим логический том:

```
lvextend -r -l +100%FREE /dev/mapper/vgroup1-root
```

По окончанию работ введем df –h, чтобы убедиться, что дисковое пространство расширилось.

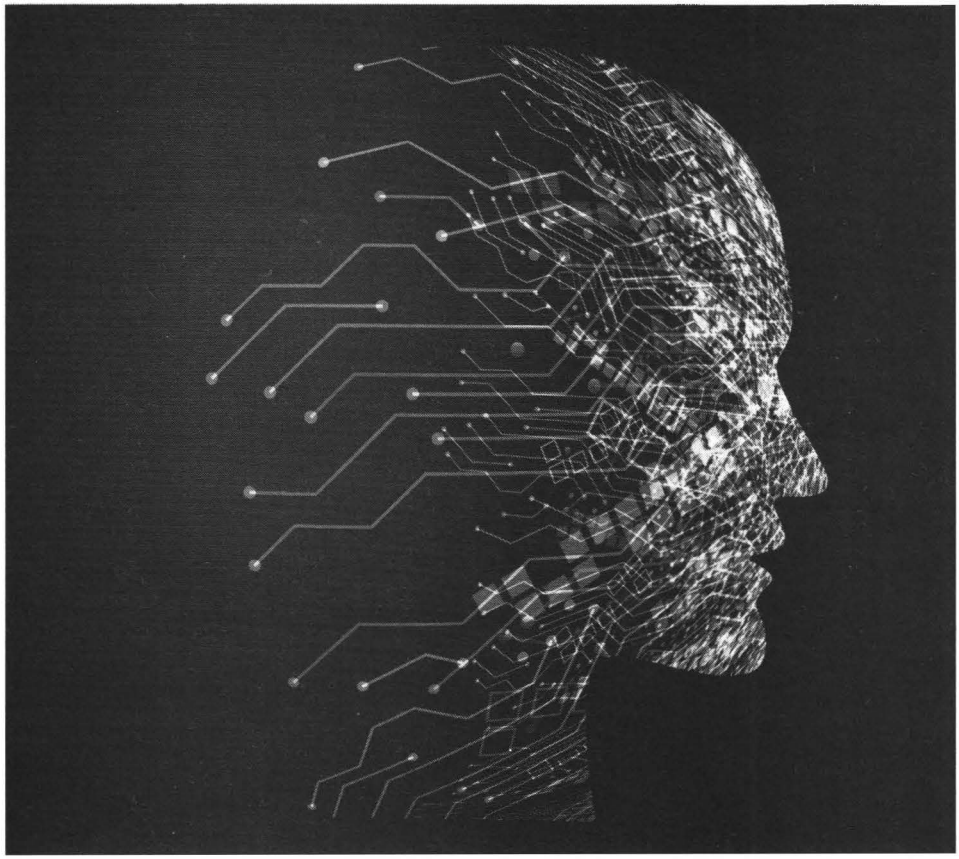
Посмотрите на рис. 14.12. На нем результат выполнения команд **pvresize**, **lvextend** и **df**. Последний вывод сообщает там, что размер группы томов vgroup1-root теперь составляет 41 Гб. Мы успешно расширили том до нового размера.

```
root@ubuntu1804:~# pvresize /dev/sda2
  Physical volume "/dev/sda2" changed
  1 physical volume(s) resized / 0 physical volume(s) not resized
root@ubuntu1804:~# lvextend -r -l +100%FREE /dev/mapper/vgroup1-root
  Size of logical volume vgroup1/root changed from 18.11 GiB (4637 extents) to <38.07 GiB (9745 exte
nts).
  Logical volume vgroup1/root successfully resized.
meta-data=/dev/mapper/vgroup1-root isize=512    agcount=12, agsize=400640 blks
         =                       sectsz=512   attr=2, projid32bit=1
         =                       crc=1        finobt=1 spinodes=0 rmapbt=0
         =                       reflink=0
data     =                       bsize=4096   blocks=4748288, imaxpct=25
         =                       sunit=0      swidth=0 blks
naming   =version 2              bsize=4096   ascii-ci=0 ftype=1
log      =internal               bsize=4096   blocks=2560, version=2
         =                       sectsz=512   sunit=0 blks, lazy-count=1
realtime =none                   extsz=4096   blocks=0, rtextents=0
data blocks changed from 4748288 to 9978880
root@ubuntu1804:~# df -H
Filesystem                Size  Used Avail Use% Mounted on
udev                      1.6G     0  1.6G   0% /dev
tmpfs                     315M  4.8M  311M   2% /run
/dev/mapper/vgroup1-root   41G  2.3G   39G   6% /
tmpfs                     1.6G     0  1.6G   0% /dev/shm
tmpfs                     5.3M     0  5.3M   0% /run/lock
tmpfs                     1.6G     0  1.6G   0% /sys/fs/cgroup
/dev/sda1                 967M  147M  754M  17% /boot
tmpfs                     315M     0  315M   0% /run/user/0
root@ubuntu1804:~#
```

Рис. 14.12. Том расширен

Глава 15.

Управление загрузкой ОС



Загрузчики Linux

Существует несколько загрузчиков Linux. На сегодняшний день основным загрузчиком является GRUB2, который устанавливается по умолчанию во всех современных дистрибутивах Linux.

Одним из самых «древних» загрузчиков является LILO (LInux LOader). Этот загрузчик давно уже не используется и ему на смену пришел загрузчик GRUB (GRand Unified Bootloader). GRUB является более гибким загрузчиком и «понимает» много разных файловых систем, в том числе FAT/FAT32, ext2, ext3, ReiserFS, XFS, BSDFS.

На смену GRUB пришел загрузчик GRUB2. Его отличия - очень запутанный и неудобный файл конфигурации, но время не стоит на месте, и если вы хотите использовать последние новинки в мире файловых систем Linux, а именно файловую систему ext4 и загрузку с LVM, вы должны использовать GRUB2. Также GRUB2 поддерживает UEFI, но это пригодится вам, только если вы - счастливый обладатель жесткого диска объема 2 Тб или больше.

Собственно, GRUB2 сейчас устанавливается во всех современных дистрибутивах, и нет смысла возвращаться на GRUB. Если возникнет необходимость, вы можете вернуться на обычный GRUB, установив пакет grub-legacy, но такая возможность есть только для платформы x86.

15.2. Загрузчик GRUB2

15.2.1. Конфигурационные файлы

В каталоге /etc/grub.d хранятся шаблоны, определяющие настройки GRUB2. Также некоторые его параметры хранятся в файле /etc/default/grub. По шаблонам из /etc/grub.d и файлу /etc/default/grub программой /usr/sbin/grub-mkconfig создается рабочий конфигурационный файл /boot/grub/grub.cfg, который по задумке разработчиков GRUB2 вы не должны редактировать вручную.

Поэтому есть две стратегии настройки GRUB2. Первая заключается в непосредственном редактировании файла /boot/grub/grub.cfg. Загрузчику

GRUB2 все равно, кто или что отредактирует этот файл - или вы или программа grub-mkconfig. Вторая заключается в редактировании файлов из каталога /etc/grub.d и файла /etc/default/grub. После чего вы будете должны ввести команду grub-mkconfig для создания файла /boot/grub/grub.cfg по заданным вами настройкам.

Чтобы решить, какая из стратегий для вас лучше, нужно знать формат и содержимое всех этих файлов. Начнем с основного файла конфигурации, который сложнее и длиннее файла конфигурации обычного GRUB (см. лист. 15.1).

Листинг 15.1. Файл конфигурации /boot/grub/grub.cfg

```
#
# Не редактируйте этот файл вручную!
#
# Он автоматически генерируется программой grub-mkconfig по
шаблонам
# из /etc/grub.d и настройкам из /etc/default/grub
#

### НАЧАЛО файла /etc/grub.d/00_header ###
if [ -s $prefix/grubenv ]; then
  load_env
fi
# Загрузочная метка по умолчанию
set default=»0»
if [ "${prev_saved_entry}" ]; then
  set saved_entry="${prev_saved_entry}"
  save_env saved_entry
  set prev_saved_entry=
  save_env prev_saved_entry
  set boot_once=true
fi

function savedefault {
  if [ -z "${boot_once}" ]; then
    saved_entry="${chosen}"
    save_env saved_entry
  fi
}

function load_video {
  insmod vbe
  insmod vga
```

```
  insmod video_bochs
  insmod video_cirrus
}

insmod part_msdos
insmod ext2
# Корневое устройство
set root='(hd0,msdos1)'
search --no-floppy --fs-uuid --set=root b7300e54-fff5-4f31-8002-
bea43c64f344
if loadfont /usr/share/grub/unicode.pf2 ; then
  set gfxmode=640x480
  load_video
  insmod gfxterm
  insmod part_msdos
  insmod ext2
  set root='(hd0,msdos1)'
  search --no-floppy --fs-uuid --set=root b7300e54-fff5-4f31-
8002-bea43c64f344
  set locale_dir=($root)/boot/grub/locale
  set lang=ru_RU
  insmod gettext
fi
terminal_output gfxterm
set timeout=5
### КОНЕЦ файла /etc/grub.d/00_header ###

### НАЧАЛО файла /etc/grub.d/05_debian_theme ###
insmod part_msdos
insmod ext2
# Корневое устройство
set root='(hd0,msdos1)'
search --no-floppy --fs-uuid --set=root b7300e54-fff5-4f31-8002-
bea43c64f344
insmod png
if background_image /usr/share/images/desktop-base/joy-grub.
png; then
  set color_normal=white/black
  set color_highlight=black/white
else
  set menu_color_normal=cyan/blue
  set menu_color_highlight=white/blue
fi
### КОНЕЦ файла /etc/grub.d/05_debian_theme ###

### НАЧАЛО файла /etc/grub.d/10_linux ###
```

```
# Содержит главную загрузочную метку. Далее мы ее рассмотрим
подробнее
menuentry 'Debian GNU/Linux, Linux 3.2.0-4-amd64' --class
debian --class gnu-linux --class gnu --class os {
   load_video
   insmod gzio
   insmod part_msdos
   insmod ext2
   set root='(hd0,msdos1)'
   search --no-floppy --fs-uuid --set=root b7300e54-fff5-4f31-
8002-bea43c64f344
   echo 'Загружается Linux 4.2.0-4-amd64 ...'
   linux   /boot/vmlinuz-4.2.0-4-amd64 root=UUID=b7300e54-fff5-
4f31-8002-bea43c64f344 ro initrd=/install/gtk/initrd.gz quiet
   echo 'Загружается начальный ramdisk ...'
   initrd  /boot/initrd.img-4.2.0-4-amd64
}
menuentry 'Debian GNU/Linux, Linux 4.2.0-4-amd64 (recovery
mode)' --class debian --class gnu-linux --class gnu --class os
{
   load_video
   insmod gzio
   insmod part_msdos
   insmod ext2
   set root='(hd0,msdos1)'
   search --no-floppy --fs-uuid --set=root b7300e54-fff5-4f31-
8002-bea43c64f344
   echo 'Загружается Linux 4.2.0-4-amd64 ...'
   linux   /boot/vmlinuz-4.2.0-4-amd64 root=UUID=b7300e54-fff5-
4f31-8002-bea43c64f344 ro single initrd=/install/gtk/initrd.gz
   echo 'Загружается начальный ramdisk ...'
   initrd  /boot/initrd.img-4.2.0-4-amd64
}
### КОНЕЦ /etc/grub.d/10_linux ###

### НАЧАЛО /etc/grub.d/20_linux_xen ###
### КОНЕЦ /etc/grub.d/20_linux_xen ###

### НАЧАЛО /etc/grub.d/20_memtest86+ ###
# Метка для memtest86 - программы для проверки памяти
menuentry "Memory test (memtest86+)" {
   insmod part_msdos
   insmod ext2
   set root='(hd0,msdos1)'
   search --no-floppy --fs-uuid --set=root b7300e54-fff5-4f31-
8002-bea43c64f344
```

```
    linux16 /boot/memtest86+.bin
}
menuentry "Memory test (memtest86+, serial console 115200)" {
    insmod part_msdos
    insmod ext2
    set root='(hd0,msdos1)'
    search --no-floppy --fs-uuid --set=root b7300e54-fff5-4f31-
8002-bea43c64f344
    linux16 /boot/memtest86+.bin console=ttyS0,115200n8
}
menuentry "Memory test (memtest86+, experimental multiboot)" {
    insmod part_msdos
    insmod ext2
    set root='(hd0,msdos1)'
    search --no-floppy --fs-uuid --set=root b7300e54-fff5-4f31-
8002-bea43c64f344
    multiboot /boot/memtest86+_multiboot.bin
}
menuentry "Memory test (memtest86+, serial console 115200,
experimental multiboot)" {
    insmod part_msdos
    insmod ext2
    set root='(hd0,msdos1)'
    search --no-floppy --fs-uuid --set=root b7300e54-fff5-4f31-
8002-bea43c64f344
    multiboot /boot/memtest86+_multiboot.bin
console=ttyS0,115200n8
}
### КОНЕЦ /etc/grub.d/20_memtest86+ ###

# Далее этот файл я немного сократил, поскольку дальше в нем
нет ничего интересного
```

Файл огромный и его синтаксис напоминает синтаксис bash-сценариев. Если вы просмотрели этот конфигурационный файл, то вы уже догадались, что делает программа grub-mkconfig: она собирает воедино все файлы из каталога /etc/grub.d (кстати, в листинге 15.1 перечислена большая часть из этих файлов) и вносит в общий конфигурационный файл из /etc/default/grub.

Основная запись из всего листинга 15.1 - это запись menuentry. Именно в таких записях описываются элементы меню загрузчика GRUB.

```
menuentry 'Debian GNU/Linux, Linux 4.2.0-4-amd64' --class
debian --class gnu-linux --class gnu --class os {
```

```
  load_video
  insmod gzio
  insmod part_msdos
  insmod ext2
  set root='(hd0,msdos1)'
  search --no-floppy --fs-uuid --set=root b7300e54-fff5-4f31-
8002-bea43c64f344
  echo 'Loading Linux 4.2.0-4-amd64 ...'
  linux  /boot/vmlinuz-4.2.0-4-amd64 root=UUID=b7300e54-
fff5-4f31-8002-bea43c64f344 ro initrd=/install/gtk/initrd.gz
quiet
  echo 'Loading initial ramdisk ...'
  initrd /boot/initrd.img-4.2.0-4-amd64
}
```

В одинарных кавычках после menuentry указывается название загрузочной метки. Далее идут параметры, которые вообще можно не указывать и от этого Debian загружаться не перестанет. В фигурных скобках - основная конфигурация. Директива load_video - это ни что иное, как вызов функции load_video, которая также описана в этом файле конфигурации. Функция вставляет некоторые модули (команда insmod), необходимые для работы графического режима. Обратите внимание, что команды insmod загружают не модули ядра Linux, а модули GRUB2, которые находятся в каталоге /boot/grub.

Внутри {} можно использовать команду echo для обозначения различных этапов загрузки, что и сделано в нашем примере. Можете отказаться от echo, на загрузку это никак не повлияет.

Основные команды - это **linux** и **initrd**. Первая указывает путь к ядру Linux и задает параметры ядра. В нашем случае параметр ядра root указывает устройство, на котором находится корневая файловая система. Устройство указано в виде UUID. UUID-имена очень удобны. Представим, что у вас есть один жесткий диск SATA и два контроллера. Если вы подключите его ко второму контроллеру, обычное имя изменится (например, было /dev/sda, а стало /dev/sdb), а UUID-имя - нет. При желании, вы можете указать имя в старом формате, например, root=/dev/sda1. Параметр ядра ro задает монтирование корневой файловой системы в режиме «только чтение» (это нормально, позже она будет перемонтирована), initrd - задает файл RamDisk, а последний параметр ядра quiet задает «тихую» загрузку ядра, при которой будут выводиться только самые важные сообщения.

Команда **initrd** задает путь к файлу initrd.

Теперь рассмотрим файл /etc/default/grub (листинг 15.2)

Листинг 15.2. Файл /etc/default/grub

```
# После редактирования этого файла запустите команду 'update-
grub' для
# обновления файла /boot/grub/grub.cfg.
# Для получения полной информации об этом файле введите
команду
#   info -f grub -n 'Simple configuration'

# Загрузочный элемент (menuentry по умолчанию)
GRUB_DEFAULT=0
# Таймаут
GRUB_TIMEOUT=5
# Задает название дистрибутива, не изменяйте эту строку
GRUB_DISTRIBUTOR=`lsb_release -i -s 2> /dev/null || echo
Debian`
# Параметры ядра Linux по умолчанию
GRUB_CMDLINE_LINUX_DEFAULT="quiet"
# Еще одна строка для задания параметров ядра
GRUB_CMDLINE_LINUX="initrd=/install/gtk/initrd.gz"

# Раскомментируйте эту строку, если вы хотите отключить
графический режим
#GRUB_TERMINAL=console

# Разрешение в графическом режиме
# Вы можете использовать только те режимы, которые ваша
видеокарта
# поддерживает через VBE. Просмотреть список
# таких режимов можно с помощью команды `vbeinfo'
#GRUB_GFXMODE=640x480

# Расскомментируйте эту строку, если вы не хотите
# использовать UUID-имена устройств
#GRUB_DISABLE_LINUX_UUID=true

# Расскомментируйте, если хотите запретить генерирование
меток восстановления
#GRUB_DISABLE_RECOVERY=»true»

# Расскомментируйте, если хотите получить гудок при
загрузке GRUB
#GRUB_INIT_TUNE=»480 440 1»
```

Как видите, параметры из файла /etc/default/grub понятны и не нуждаются в особых комментариях. Но в нем мы узнали о еще одной команде - update-grub. Так какую из них использовать - update-grub или grub-mkconfig?

На самом деле это почти одна и та же команда. Дело в том, что команда grub-mkconfig по умолчанию выводит конфигурацию GRUB2 на экран, поэтому, чтобы она записалась в файла /boot/grub/grub.cfg, запускать ее нужно так:

```
sudo grub-mkconfig > /boot/grub/grub.cfg
```

Или же вы можете ввести команду update-grub, которая сделает то же самое. Другими словами, команда update-grub - это сценарий, который вызывает только что приведенную команду. Как по мне, то использовать команду update-grub удобнее.

Так какую стратегию GRUB2 использовать? Редактировать шаблоны и параметры или сразу конфигурационный файл? Если вы работаете за компьютером в гордом одиночестве и нет и не предвидится других администраторов, тогда можете выбрать ту стратегию, которая вам больше нравится.

Если же есть или планируются другие администраторы, то нужно редактировать шаблоны и параметры вместо редактирования конфигурационного файла вручную. Дело в том, что если вы внесете изменения непосредственно в конфигурационный файл, а потом другой администратор захочет изменить какой-то незначительный параметр, например, добавить гудок при загрузке GRUB2, то команда update-grub перезапишет все сделанные вами изменения.

15.2.2. Выбор метки по умолчанию

Как правило, даже если у вас установлена одна только Linux, у вас будет несколько загрузочных меток (несколько записей menuentry). Выбрать метку по умолчанию можно с помощью параметра GRUB_DEFAULT. Нумерация меток начинается с **0**, то есть первой метке соответствует значение **0**.

После того, как вы установите другой номер метки по умолчанию нужно ввести команду update-grub и перезагрузить систему.

Другими словами, последовательность такая: редактируем файл /etc/default/grub, изменяем значение параметра GRUB_DEFAULT и вводим команду update-grub.

15.2.3. Загрузка Windows

Упор в этой книге делается на серверы и прочее производственное применение Linux, однако вдруг вы захотите установить Linux на домашний компьютер, скорее всего, ей придется «сожительствовать» с Windows.

Чтобы добавить возможность загрузки Windows из GRUB2, нужно отредактировать файл /etc/grub.d/40_custom и добавить в него следующие строки:

```
menuentry "Windows (on /dev/sda1)" {
        insmod part_msdos
        insmod ntfs
        set root='(hd0,msdos1)'
        search --no-floppy --fs-uuid --set UUID
        drivemap -s (hd0) ${root}
        chainloader +1
}
```

Здесь Windows находится на /dev/sda1 и адрес этого раздела указывается в set root. Значение UUID нужно заменить на UUID вашего Windows-раздела. О том, как «вычислить» UUID раздела, было сказано в главе 13. Остальные параметры можете оставить без изменения (еще не забудьте исправить hd0 в drivemap, если Windows установлена не на первом жестком диске).

15.2.4. Пароль загрузчика GRUB2

Загрузчик GRUB позволял только установить пароль - или общий или на загрузку определенной метки. Загрузчик GRUB2 более гибкий в этом плане, поскольку вы можете настроить не только пароли, но и логины. Также есть минимальная система разграничения прав доступа.

Итак, в GRUB2 есть суперпользователь, который может редактировать загрузочные метки. Существует возможность восстановить пароль root путем передачи ядру параметра init. Но для этого нужно отредактировать конфигурацию GRUB2. Если вы установите пароль суперпользователя, то изменить конфигурацию загрузчика вы сможете только после ввода этого пароля.

Также в GRUB2 есть обычные пользователи, которые имеют право только выбирать загрузочную метку. Они не имеют права редактировать конфигурацию загрузчика. В принципе, можно обойтись одним паролем супер-

пользователя, но при желании GRUB2 может довольно гибко разграничить права пользователей.

Давайте сначала добавим пароль суперпользователя. Для этого в файл /etc/grub.d/00_header добавьте строки:

```
set superusers="main_admin"
password main_admin 123456789
```

Первая команда задает суперпользователя main_admin, а вторая - задает для него пароль. Старайтесь избегать общепринятых имен вроде admin, root и т.д. Так у злоумышленника, который хочет изменить конфигурацию GRUB2 будет две неизвестных.

Пароль пока в незашифрованном виде и это не очень хорошо. Поскольку если загрузиться с LiveCD или LiveUSB, то его можно будет увидеть. Позже я покажу, как зашифровать пароль.

Обычные пользователи задаются инструкцией **password**, например:

```
password me 12345
```

По сути main_admin - тоже был бы обычным пользователем, если бы не инструкция set superusers, которая делает его суперпользователем.

Представим, что у нас есть следующие строки:

```
set superusers="main_admin"
password main_admin 123456789
password me 12345
```

Пользователь main_admin может загружать операционные системы и редактировать конфигурацию GRUB2. Пользователь me может только загружать операционные системы.

Если вы хотите, чтобы определенные метки могли загружать только определенные пользователи, добавьте к menuentry параметр --users:

```
menuentry "Windows" --users me {
        insmod part_msdos
        insmod ntfs
        set root='(hd0,msdos1)'
        search --no-floppy --fs-uuid --set UUID
        drivemap -s (hd0) ${root}
        chainloader +1
}
```

Теперь зашифруем пароль. Введите команду:

```
grub-mkpasswd-pbkdf2
```

Программа запросит пароль, зашифрует его и выведет на экран его кэш. Вы увидите что-то подобное:

```
grub.pbkdf2.sha512.10000.9290F727ED06C38BA4549EF7DE25CF5642659
211B7FC076F2D27080136.887CFF169EA83D5235D8004742AA7D6187A41E31
87DF0CE14E256D85ED97A979080136.887CFF169EA8335235D8004242AA7D6
187A41E3187DF0CE14E256D85ED97A97357AAA8FF0A3871AB9EEFF458392F4
62F495487387F685B7472FC6C29E293F0A0
```

Данный хэш нужно указать вместо пароля пользователя. Однако вместо инструкции passwd нужно использовать password_pbkdf2. Например:

```
password_pbkdf2 me grub.pbkdf2.sha512.10000.9290F727ED06C38BA4
549EF7DE25CF5642659211B7FC076F2D27080136.887CFF169EA83D5235D80
04742AA7D6187A41E3187DF0CE14E256D85ED97A979080136.887CFF169EA8
335235D8004242AA7D6187A41E3187DF0CE14E256D85ED97A97357AAA8FF0A
3871AB9EEFF458392F462F495487387F685B7472FC6C29E293F0A0
```

После изменения файлов из каталога /etc/grub.d не забудьте ввести команду upgrade-grub для обновления основного файла конфигурации.

15.2.5. Установка загрузчика

Команда установки загрузчика такая же, как и в случае с GRUB:

```
# /sbin/grub-install <устройство>
```

Например:

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

Поскольку GRUB2 у вас уже установлен, вряд ли вам придется вводить эту команду. Исключение может поставить разве что переустановка Windows, которая перезапишет загрузочный сектор своим загрузчиком. Поэтому вам придется загрузиться с LiveCD, выполнить chroot для вашей старой корневой системы и ввести команду grub-install для установки загрузчика GRUB2.

15.3. Система инициализации

После своей загрузки ядро передает управление системе инициализации. Цель этой системы - выполнить дальнейшую инициализацию системы. Самая главная задача системы инициализации - запуск и управление системными службами.

Служба (сервис, демон) - специальная программа, выполняющаяся в фоновом режиме и предоставляющая определенные услуги (или, как говорят, сервис - отсюда и второе название). Что превращает обычный компьютер, скажем, в FTP-сервер? Правильно, запущенная служба FTP - тот же ProFTPD или подобная. Вы можете установить программу ProFTPD и настроить ее на автоматический запуск системой инициализации. Тогда при каждой загрузке наш компьютер будет превращаться в FTP-сервер. Аналогично и с другими сервисами - достаточно установить определенную программу, чтобы превратить компьютер в веб-сервер или почтовый сервер. Но стоит вам отключить ее и компьютер уже прекращает предоставлять обеспечиваемые программой услуги, следовательно, превращается в самый обычный компьютер.

В мире Linux существовало очень много разных систем инициализации – init, upstart, init-ng. Все их рассматривать уже нет смысла, поскольку в современных дистрибутивах используется современная система инициализации systemd.

systemd — подсистема инициализации и управления службами в Linux, фактически вытеснившая в 2010-е годы традиционную подсистему **init**. Основная особенность — интенсивное распараллеливание запуска служб в процессе загрузки системы, что позволяет существенно ускорить запуск операционной системы. Основная единица управления — модуль, одним из типов модулей являются «службы» — аналог демонов — наборы процессов, запускаемые и управляемые средствами подсистемы и изолируемые контрольными группами.

15.3.1. Принцип работы

Система инициализации **systemd** используется во многих современных дистрибутивах, в частности в Fedora, Ubuntu, CentOS и openSUSE. На данный момент - это самая быстрая система инициализации.

Давайте подумаем, как можно ускорить запуск Linux? Можно пойти по пути upstart - параллельно запускать службы. Но параллельный запуск -

не всегда хорошо. Нужно учитывать зависимости служб. Например, сервис d-bus нужен многим другим сервисам. Пока сервис d-bus не будет запущен, нельзя запускать сервисы, которые от него зависят.

Если сначала запускать основные сервисы и ждать, пока они будут запущены, а потом уже запускать службы, которые от них зависимы, особого выигрыша в производительности по сравнению с init вы не увидите. Но если сервис d-bus (или любой другой, от которого зависят какие-то другие сервисы) запускается долго, то все остальные службы будут ждать его.

Как обойти это ограничение? При своем запуске службы проверяют, запущена ли необходимая им служба, по наличию файла сокета. Например, в случае с d-bus - это файл /var/run/dbus/system_bus_socket. Если мы создадим сокеты для всех служб, то мы можем запускать их параллельно, особо не беспокоясь, что произойдет сбой какой-то службы при запуске из-за отсутствия службы, от которой они зависят. Даже если несколько служб, которым нужен сервис d-bus, запустятся раньше, чем сам сервис d-bus: ничего страшного. Каждая из этих служб отправит в сокет (главное, что он уже открыт!) сообщение, которое обработает сервис d-bus после того, как он запустится. Вот и все.

Но это не единственное "ухищрение", посредством которого осуществляется ускорение запуска компьютера, инициализацию которого производит systemd. Эта система инициализации запускает только необходимые сервисы. Остальные же будут запущены по мере необходимости. Концепция отложенного запуска используется и в других операционных системах - например, в Mac OS X (там система инициализации называется launchd) и в Windows (концепция отложенного запуска служб). Так что решение не очень новое, но зато проверенное.

Основными функциями **systemd** являются:

- **Активация на основании сокетов** - система инициализации systemd прослушивает сокеты всех системных служб. Сокеты передаются системным службам сразу после запуска сервисов. Благодаря этому осуществляется параллельный запуск сервисов. Также это позволяет перезапускать сервисы без потери любых отправленных им сообщений, то есть пока сервис перезапускается, отправленные ему сообщения накапливаются, и он сможет их обработать после того, как будет запущен.
- **Активация на основании устройств** - systemd может запустить определенные службы, когда станет доступным определенный тип оборудования. Например, вы подключили Bluetooth-адаптер, может быть запущен сервис bluetooth.

- **Активация на основании d-bus** - служба инициализации может запустить сервисы, которые используют d-bus для межпроцессного взаимодействия, например, когда клиентское приложение попытается связаться с системной службой.
- **Активация на основании путей** - systemd может запустить службу, если изменится содержание каталога.
- **Управление точками монтирования и автоматическим монтированием** - система инициализации отслеживает и управляет точками монтирования и автоматического монтирования.
- **Снимки системных состояний** - благодаря этой возможности systemd может сохранить состояние всех модулей и восстановить предыдущее состояние системы.
- **Параллелизация** - systemd запускает системные службы параллельно благодаря активации на основании сокетов. Параллельная активация существенно сокращает время загрузки системы.
- **Обратная совместимость с SysV** - поддерживаются сценарии инициализации SysV, что упрощает переход на systemd. Однако все устанавливаемые в современных дистрибутивах пакеты служб уже адаптированы под systemd, поэтому не нужно надеяться, что во время установки пакета какого-то сервиса будут установлены SysV-сценарии. Будут созданы файлы, необходимые для запуска сервиса посредством systemd.

15.3.2. Конфигурационные файлы systemd

Обилие различных конфигурационных файлов systemd может ввести в ступор даже бывалого линуксоида, не говоря уже о пользователе, который впервые видит systemd. Когда я впервые познакомился с systemd, у меня было только одно желание - снести ее и установить вместо нее init. Но мы это не будем делать. Чтобы разобраться со всеми файлами, нужно понимать, как работает эта система.

В systemd используется концепция модулей (юнитов). Существующие типы модулей описаны в таблице 15.1.

Таблица 15.1. Типы модулей системы инициализации systemd

Тип	Описание
service	Служба (сервис, демон), которую нужно запустить. Пример имени модуля: network.service. Изначально systemd поддерживала сценарии SysV (чтобы управлять сервисами можно было, как при использовании init), но в последнее время в каталоге /etc/init.d систем, которые используют systemd практически пусто (или вообще пусто), а управление сервисами осуществляется только посредством systemd
target	Цель. Используется для группировки модулей других типов. В systemd нет уровней запуска, вместо них используются цели. Например, цель multi-user.target описывает, какие модули должны быть запущены в многопользовательском режиме
snapshot	Снимок. Используется для сохранения состояния systemd. Снимки могут использоваться для перевода системы из одного состояния в другое, например, в состояние сна и пробуждение
mount	Точка монтирования. Представляет точку монтирования. Система инициализации systemd контролирует все точки монтирования. При использовании systemd файл /etc/fstab уже не главный, хотя все еще может использоваться для определения точек монтирования
automount	Автоматическая точка монтирования. Используется для монтирования сменных носителей - флешек, внешних жестких дисков, оптических дисков и т.д.
socket	Сокет. Представляет сокет, находящийся в файловой системе или в Интернете. Поддерживаются сокеты AF_INET, AF_INET6, AF_UNIX. Реализация довольно интересная. Например, если сервису service1.servive соответствует сокет service1.socket, то при попытке установки соединения с service1.socket будет запущен service1.servive

device	Устройство. Представляет устройство в дереве устройств. Работает вместе с udev: если устройство описано в виде правила udev, то его можно представить в systemd в виде модуля device
path	Файл или каталог, созданный где-то в файловой системе
scope	Процесс, который создан извне
slice	Управляет системными процессами. Представляет собой группу иерархически организованных модулей
swap	Представляет область подкачки (раздел подкачки) или файл подкачки (свопа)
timer	Представляет собой таймер системы инициализации systemd

Модули хранятся в следующих каталогах:

- /etc/systemd/system/ - обладает самым высоким приоритетом. Здесь содержатся модули, которые созданы и управляются системным администратором.
- /run/systemd/system/ - модули, созданные во время выполнения. Приоритет этого каталога ниже, чем каталога /etc/systemd/system/, но выше, чем у /usr/lib/systemd/system.
- /usr/lib/systemd/system/ - модули, которые установлены из пакетов.

Типичный файл модуля типа service приведен в листинге 15.3.

Листинг 15.3. Типичный файл модуля типа service

```
[Unit]
Description=Daemon to detect crashing apps
After=syslog.target

[Service]
ExecStart=/usr/sbin/abrtd
Type=forking

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

В секции **Unit** содержится общая информация о сервисе. Эта секция есть и в других модулях, а не только в сервисах.

Секция **Service** содержит информацию о сервисе. Параметр ExecStart описывает команду, которую нужно запустить. Параметр Type указывает, как сервис будет уведомлять systemd об окончании запуска.

Секция **Install** содержит информацию о цели, в которой должен запускаться сервис. В нашем видно, что сервис будет запущен при активации цели multi-user.target.

Вы можете использовать эту "болванку" для написания собственного сервиса, который потом нужно поместить в файл /etc/systemd/system/имя_сервиса.service. После этого нужно перезапустить саму systemd, чтобы она узнала о новом сервисе:

```
# systemctl daemon-reload
```

15.3.3. Цели

Теперь поговорим о целях. Файлы целей *.target предназначены для группировки вместе других юнитов systemd через цепочку зависимостей. Так, модуль цели graphical.target, который используется для запуска графического сеанса, запускает системные службы GDM (файл gdm.service) и Accounts Service (accounts-daemon.service), а также активирует цель multi-user.target. В свою очередь, цель multi-user.target запускает другие системные службы, например, D-Bus (dbus.service) и активирует другие цели вроде basic.target.

В systemd имеются предопределенные цели, которые напоминают стандартный набор уровней запуска.

Некоторые цели называются runlevelN.target, чтобы упростить переход бывших пользователей **init** на **systemd**, а именно:

- poweroff.target (runlevel0.target) – завершение работы и отключение системы;
- rescue.target (runlevel1.target) – однопользовательский режим, среда восстановления;
- multi-user.target (runlevel2.target, runlevel3.target, runlevel4.target) – многопользовательский режим, без графического интерфейса;
- graphical.target (runlevel5.target) – многопользовательский режим с графическим интерфейсом

- reboot.target (runlevel6.target) – завершение работы и перезагрузка системы

Управление службами осуществляется с помощью программы systemctl. Подробнее о службах мы поговорим в следующем разделе, а пока разберемся, как использовать systemctl для завершения работы системы:

- systemctl halt – останавливает систему;
- systemctl poweroff – выключает систему;
- systemctl reboot – перезагружает систему.

Многим пользователям будет удобнее использовать старые команды **halt**, **poweroff** и **reboot**. Но все же теперь вы знаете, что есть альтернативные способы завершения работы.

15.4. Управление сервисами при использовании systemd

При использовании системы инициализации systemd управление службами осуществляется посредством программы systemctl. Команда systemctl используется для разных целей, поэтому в таблице 15.2 представлены не все ее параметры, а только те, которые имеют отношение к сервисам.

Таблица 15.2. Параметры программы systemctl

Параметр	Описание
start <имя.service>	Запускает сервис
stop <имя.service>	Останавливает сервис
restart <имя.service>	Перезапускает сервис
try-restart <имя.service>	Перезапуск сервиса только, если он запущен

reload <имя.service>	Перезагружает конфигурацию сервиса
status <имя.service>	Отображает подробное состояние сервиса
is-active <имя.service>	Отображает только строку active (сервис запущен) или inactive (остановлен)
list-units --type service --all	Выводит состояние всех сервисов
enable <имя.service>	Включает сервис (обеспечивает его автоматический запуск)
disable <имя.service>	Отключает сервис (сервис не будет автоматически запускаться при запуске системы)
reenable <имя.service>	Деактивирует сервис и сразу его использует
list-unit-files --type service	Выводит список всех сервисов и сообщает, какие из них активированы, а какие - нет

Примеры:

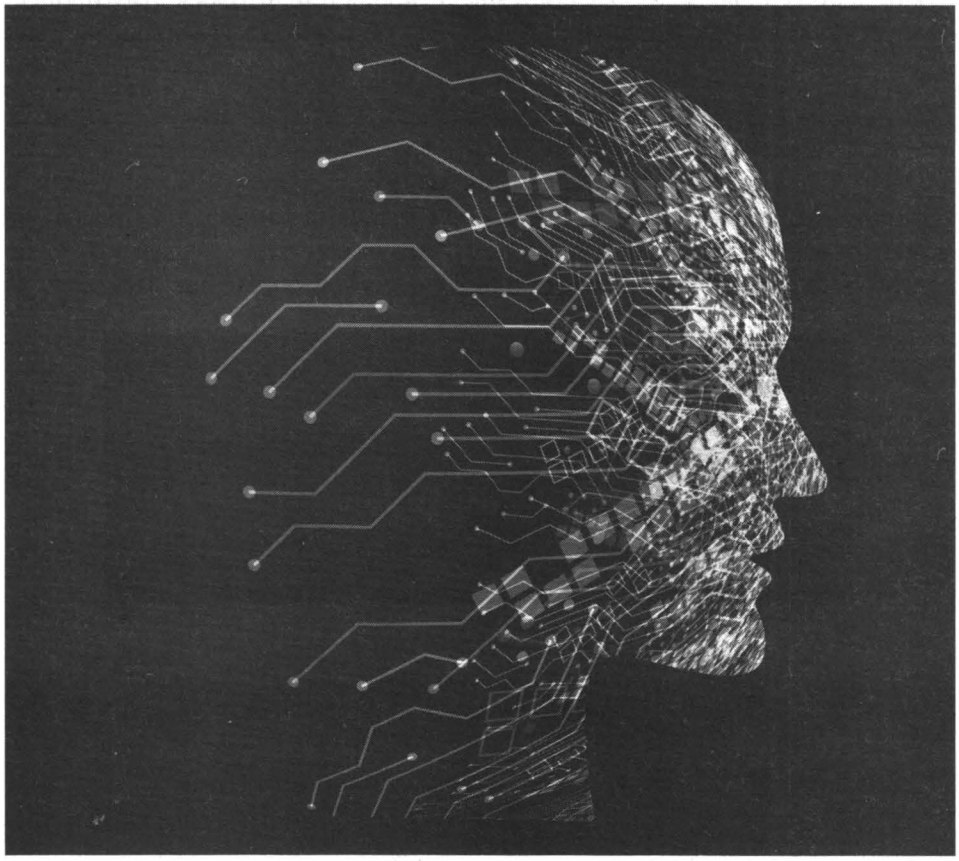
```
# systemctl start httpd.service
# systemctl stop httpd
```

Первая команда запускает сервис httpd (веб-сервер), вторая - останавливает. Обратите внимание, что ".service" можно не указывать.

Бывалые пользователи Linux сразу заметят удобства. Ранее, чтобы отключить службу на определенном уровне запуска, нужно было удалить ее символическую ссылку из определенного каталога. Аналогично, чтобы служба запускалась на определенном уровне запуска (например, в графическом режиме), нужно было создать символическую ссылку. Сейчас всего этого нет, а есть только команды **enable** и **disable**, что гораздо удобнее.

Глава 16.

Управление процессами



16.1. Команды ps, nice и kill

16.1.1. Получение информации о процессе

Современные операционные системы устроены так, что каждому процессу присваивается уникальный номер - PID (Process ID, ИД процесса), используя который можно управлять процессом, например, можно завершить процесс или изменить его приоритет.

Узнать PID можно с помощью команды `ps`. Команда `ps`, введенная без параметров, просто показывает список процессов, запущенных на текущем терминале. Видно, что сейчас запущен `bash` и сама команда `ps` (правда, на момент завершения вывода процесс с ID 975 уже не будет существовать, но на момент самого вывода такой процесс существовал), см. рис. 16.1.

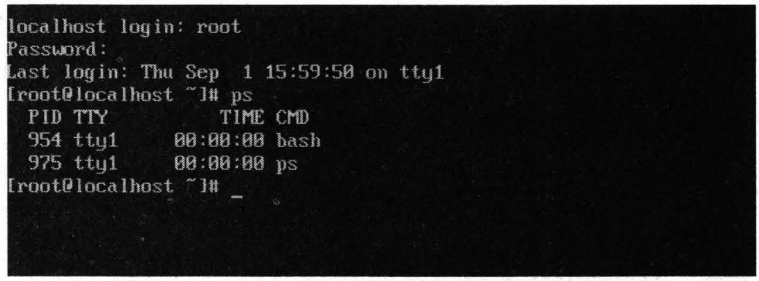


Рис. 16.1. Команда `ps`

Параметр -a позволяет вывести список всех процессов пользователя. Посмотрите на рис. 16.2 на консоли `tty3` запущена программа `nano`, на консоли `tty1` - программа `ps`.

Если нужно вывести процессы какого-то определенного пользователя, тогда используйте параметр -u:

```
$ ps -u root
```

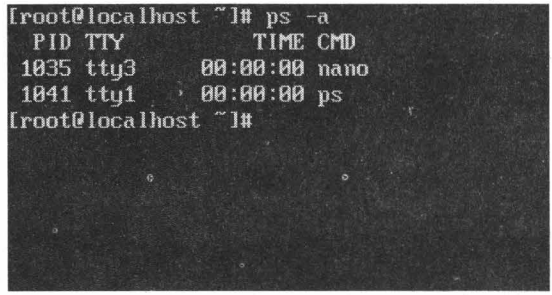


Рис. 16.2. Команда `ps -a`

Вывести абсолютно все процессы можно с помощью опции -A. Обратите внимание на регистр опции! Поскольку в системе процессов будет очень много, лучше перенаправить вывод программы на команду `less` для более удобного просмотра (рис. 16.3):

```
$ ps -A | less
```

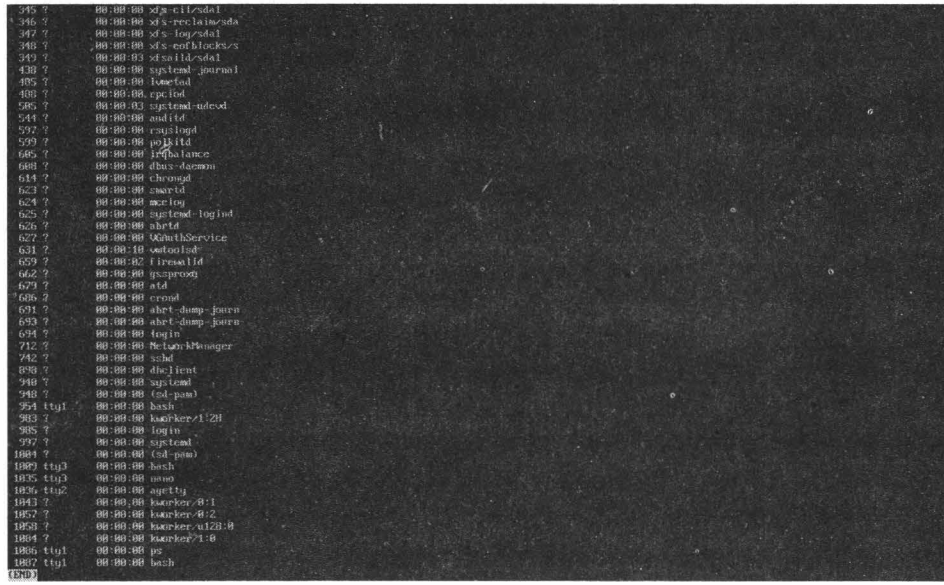


Рис. 16.3. Команда `ps -A | less`

Примечание. Для выхода из программы `less` нажмите q на клавиатуре. Листать вывод можно стрелками вверх и вниз.

Команда `ps` сортирует процессы по `PID`. Колонка `TTY` - это терминал, к которому привязан процесс. Если в этой колонке вы видите знак ?, значит, процесс не привязан ни к одному из терминалов. Как правило, это системные процессы-службы. Они запускаются без привязки к терминалу. Чтобы отобразить только процессы без привязки к терминалу, используется опция -x (рис. 16.4).

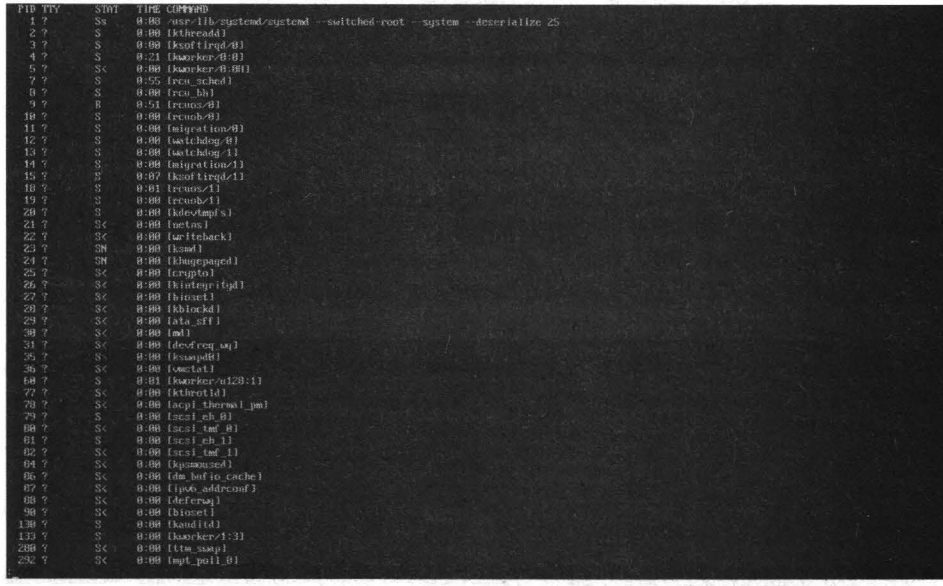


Рис. 16.4. Команда `ps -x | less`

Колонка `STAT` - это состояние, в котором находится процесс. Возможные значения для этой колонки приведены в таблице 16.1. Обратите внимание: колонка `STAT` есть только, когда программа запущена с параметром -x. Если программа запущена с другими параметрами, например, -A, вместо нее будет колонка `TIME`, которая сообщает занимаемое процессом процессорное время.

Таблица 16.1. Возможные состояния процесса

Состояние	Описание
D	Процесс в непрерывном сне (как правило, ожидает ввода/вывода)
R	Выполняется в данный момент

S	Ожидание (то есть процесс «спит» менее 20 секунд, после чего он переходит или в состояние R или в состояние D)
T	Процесс остановлен
t	То же, что и T, но причина остановки - останов отладчиком
W	Процесс в свопинге (подкачке)
X	Процесс мертв, вы его никогда не увидите
Z	Процесс-зомби - он уже завершен, но не «похоронен» его родителем, то есть процесс-родитель еще не считал код завершения

У команды `ps` есть несколько синтаксисов установки параметров. Мы использовали BSD-синтаксис. Например, для вывода всех процессов в стандартном синтаксисе используется команда -e, а в нашем случае - -A. Вы вольны использовать любой синтаксис, но в случае с BSD-синтаксисом программа `ps` выводит дополнительное состояние процесса (работает подобно программе `ps` в системе BSD). Дополнительное состояние процесса описано в таблице 16.2.

Таблица 16.2. Дополнительное состояние процесса (BSD-синтаксис)

Состояние	Описание
<	Высокий приоритет
N	Низкий приоритет
L	У процесса есть страницы, заблокированные в памяти
s	Это лидер сессии
l	Процесс является многопотоковым
+	Находится на первом плане в группе процессов

Последняя колонка вывода `ps` - это `CMD`. Она содержит команду, которой был запущен процесс. Не просто название исполнимого файла, но путь (если он был указан в команде) и переданные программе параметры.

Если программа была запущена без указания полного пути к исполнимому файлу, и вы хотите знать, где он находится, введите команду which, например:

```
$ which nano
/bin/nano
```

Если вам нужно узнать PID определенного процесса, но вам не хочется просматривать длинный список системных процессов, используйте команду grep, как фильтр. Например, следующая команда позволит нам узнать PID процесса sshd (это SSH-сервер):

```
# ps -A | grep sshd
```

Если такой процесс не запущен, вывод будет пуст. Или же вы получите вывод вроде этого:

```
929 ?  00:00:00 sshd
```

16.1.2. Изменение приоритета процесса

Когда мы знаем PID процесса, мы можем изменить его приоритет. В некоторых случаях полезно изменить приоритет процесса. Например, можно повысить приоритет процесса, выполняющего резервное копирование, чтобы программа успела за ночь создать все необходимые резервные копии, и чтобы этот процесс утром потом не мешал нормальной работе сервера.

Запустить программу с определенным приоритетом можно командой nice:

```
# nice -n <приоритет> команда аргументы
```

Здесь приоритет задается от -20 (максимальный приоритет) до 19 (минимальный). Если процесс уже был запущен, и вы не можете его прерывать, но повысить приоритет нужно, используйте команду renice:

```
# renice -n <приоритет> -p PID
```

16.1.3. Аварийное завершение процесса

Если процесс завис и его нельзя завершить, как обычно, тогда для его аварийного завершения используется команда kill. Формат вызова этой команды следующий:

```
$ kill [опции] PID
```

Конечно, перед этим нужно узнать PID процесса. На рис. 16.5 изображена команда `kill` в действии: сначала я вывел список процессов, чтобы узнать PID процесса nano (1191), затем я ввел команду `kill 1191`, чтобы «убить» этот процесс. Наконец, я вывел список процессов еще раз, чтобы убедиться, что процесс nano завершен.

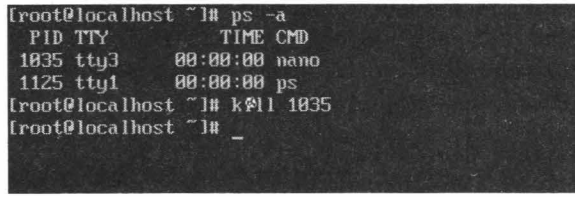


Рис. 16.5. Использование команды `kill`

Используя параметры программы, можно по-разному завершить процесс. Самый эффективный сигнал `9` (KILL) - означает аварийное завершение процесса. Программа не может игнорировать или как-либо обработать этот процесс.

Если нужно попытаться корректно завершить работу программы, ей отправляют сигнал `15` (TERM), означающий, что программа должна освободить все занятые ресурсы, сохранить все данные. Вот только если программа зависла и не отвечает на запросы пользователя, этот сигнал мало чем поможет, но попытаться стоит.

Сигнал `19` (STOP) позволяет временно приостановить работу программы, а сигнал `18` (CONT) - возобновить приостановленный ранее процесс.

Для сетевых служб полезен сигнал `1` (HUP), означающий, что процесс должен перезапуститься и перечитать файл конфигурации. Полезно, когда вы изменили файл конфигурации и хотите, чтобы демон был перезапущен (хотя для этого правильнее использовать команду `service`). Обычная программа при получении сигнала 1 завершает работу.

Пример отправки сигнала:

```
$ kill -9 1035
```

Если вам лень получать PID процесса, можно завершить его и по имени, используя команду `killall`, например:

```
$ killall nano
```

Вот только если в вашей системе есть два процесса с именем **nano**, например, один на консоли `tty2`, а другой - на tty4, то будут завершены оба процесса. Если это то, что вам нужно, используйте `killall`, в противном случае

лучше использовать команду `kill` для завершения именно того процесса, который можно завершить.

Еще есть команда `xkill`, позволяющая «убить» программу, имеющую графический интерфейс. Такие программы можно завершить и командой **kill**, но программа xkill предоставляет графический метод завершения. После ввода этой команды указатель мыши примет вид черепа. Для завершения программы нужно щелкнуть по ее окну.

16.2. Команда top

Как было отмечено ранее, программа `ps` по умолчанию сортирует процессы по колонке PID, а не по колонке `TIME`. Конечно, можно использовать различные параметры программы, чтобы добиться нужного нам вывода, но все равно программа не будет показывать ситуацию в реальном времени. Если же вам нужно знать, что происходит с вашими процессами в реальном времени, вам нужно использовать программу `top` (рис. 16.6).

Назначение колонок программы описано в таблице 16.3.

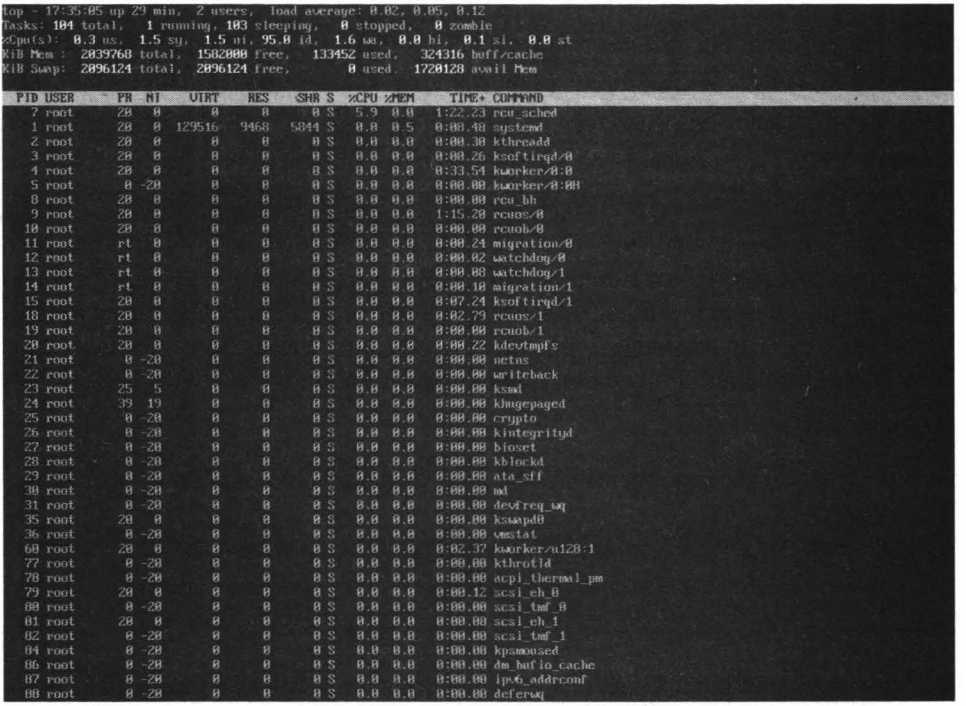


Рис. 16.6. Команда top

Таблица 16.3. Колонки программы top

Колонка	Описание
PID	PID процесса
USER	Владелец процесса (пользователь, запустивший программу)
PR	Приоритет процесса
NI	Значение nice (см. ранее)
VIRT	Виртуальная память, которая используется процессом
RES	Размер процесса, который не перемещается в область подкачки
SHR	Разделяемая память, используемая процессом
S	Состояние процесса (см. табл. 16.1)
%CPU	Процессорное время, занимаемое процессом в данный момент
%MEM	Память, используемая процессом
TIME+	Процессорное время, которое было потрачено с момента запуска процесса
COMMAND	Команда запуска процесса

При просмотре списка программы `top` вы можете управлять сортировкой процессов с помощью нажатия клавиши F, которая изменяет колонку, по которой сортируется список процессов (рис. 16.7). По умолчанию сортировка выполняется по колонке `%CPU`.

Нажатие клавиши <U> показывает только процессы определенного пользователя. После нажатия <U> нужно будет ввести имя пользователя, процессы которого вы хотите просмотреть, или нажать **Enter**, чтобы просмотреть процессы всех пользователей.

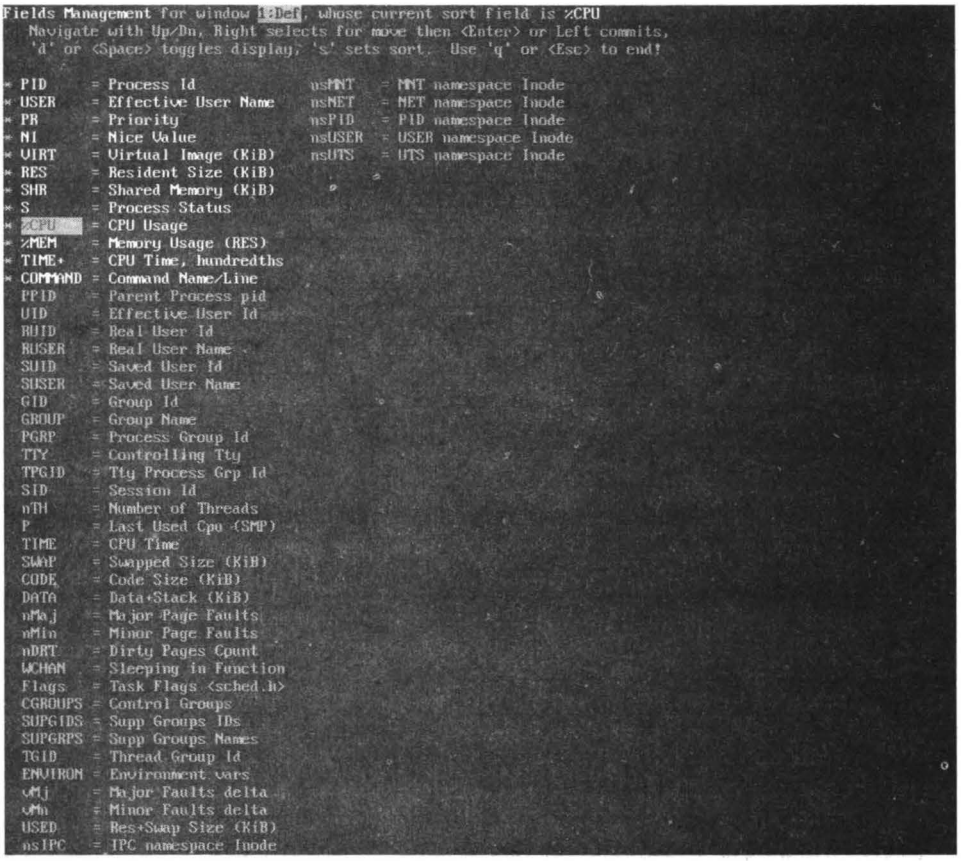


Рис. 16.7. Выбор колонки, по которой осуществляется сортировка

16.3. Информация об использовании памяти и дискового пространства

Хотя управление памятью и дисковым пространством не совсем относится к управлению процессами, но эти самые процессы активно «поедают», как память, так и дисковое пространство, поэтому иногда полезно знать, как просмотреть информацию об использовании памяти (команда `free`) и дискового пространства (команда `df`), см. рис. 16.8.

Программа `free` выводит информацию об использовании памяти (`Mem`) и подкачки (`Swap`). Колонка `total` - это общее количество памяти в килобайтах, `used` - использованное количество памяти (тоже в килобайтах), `free` - свободно памяти, `shared` - разделяемая память, `buff/cache` - размер кэша, `available` - общий объем доступной памяти.

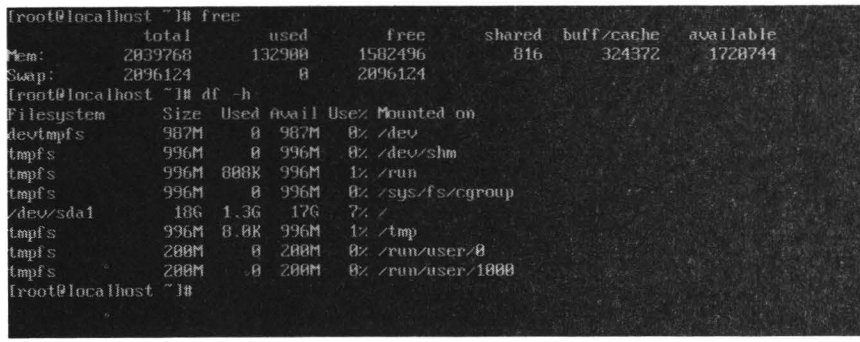


Рис. 16.8. Команды `free` и `df -H`

Параметр -H команды `df` означает вывод информации об объеме в удобных для восприятия человеком единицах, то есть в мегабайтах и гигабайтах.

Обратите внимание на значение `buff/cache` в выводе команды `free`. Оно показывает сколько памяти задействовано под буфер ввода/вывода и кэш. В нашем случае (рис. 16.8) - примерно 323 Мб. На реальном сервере это значение будет гораздо выше. Немного освободить память можно, очистив кэш. Для этого введите команду:

```
sync; echo 3 > /proc/sys/vm/drop_caches
```

Сначала мы командой `sync` сбрасываем содержимое буферов на диск, а затем уничтожаем кэш. Если просмотреть затем информацию об использовании памяти, то вы увидите, что размер кэша был уменьшен почти в три раза (рис. 16.9). Однако помните, что эта команда может негативно отразиться на стабильности системы и на скорости ее работы. Не всегда очистка кэша таким вот варварским образом - это хорошо.

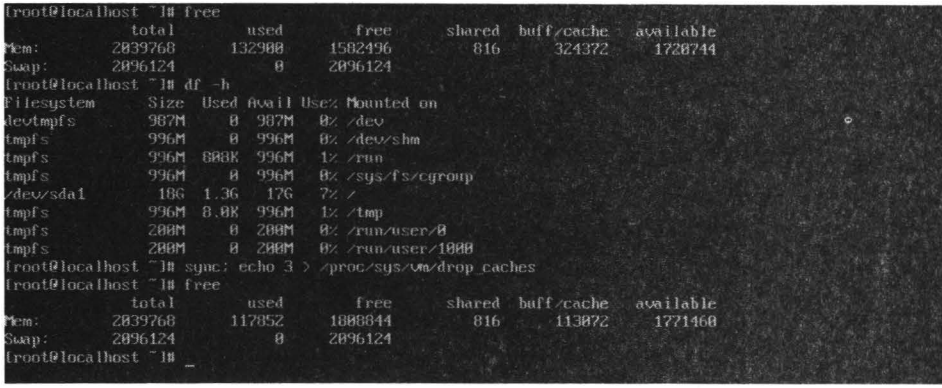


Рис. 16.9. До и после ввода команды `sync; echo 3 > /proc/sys/vm/drop_caches`

16.4. Команда fuser

Команда `fuser` позволяет узнать, какой процесс открыл тот или иной ресурс, например, файл или сетевой порт. Примеры использования программы:

```
fuser -va 23/tcp
fuser -va /chroot/etc/resolv.conf
```

В первом случае мы получим идентификатор процесса, открывшего TCP-порт 23, во втором - идентификатор процесса, открывшего файл /chroot/etc/resolv.conf. Что делать далее - решать вам, например, можно «убить» этот процесс командой `kill`.

16.5. Планировщики заданий

16.5.1. Планировщик cron

Планировщик заданий нужен для периодического выполнения каких-либо заданий. Задания могут быть самыми разнообразными - очистка временного каталога для экономии места, очистка кэша, обновление баз антивируса, запущенного на почтовом сервере и т.д. Никаких ограничений нет - вы можете написать на bash небольшой сценарий с необходимыми вам действиями, а затем настроить планировщик для его периодического выполнения.

Самый древний планировщик заданий - **cron**. Он появился еще во времена первых версий UNIX. Но примечательно то, что в современных дистрибутивах используются его модифицированная версия, которая настраивается практически так же. Так, в современных версиях дистрибутивов openSUSE и Fedora используется демон cronie, который является потомком того самого демона cron.

Не будем вникать в тонкости новой версии, а просто разберемся, как ее настроить. Как и в том самом cron есть файл /etc/crontab - это таблица расписания планировщика задач. Формат записей в этом файле следующий:

```
минуты (0-59) часы (0-23) день (1-31) месяц (1-12) день_недели
(0-6, 0 — Вс) команда
```

В листинге 16.1 приведен пример этого файла по умолчанию.

Листинг 16.1. Пример файла /etc/crontab

```
SHELL=/bin/bash
PATH=/usr/bin:/usr/sbin:/sbin:/bin:/usr/lib/news/bin
MAILTO=root

-*.15 * * * *  root test -x /usr/lib/cron/run-crons && /usr/
lib/cron/run-crons >/dev/null 2>&1
```

> Примечание. Просмотреть (даже просмотреть, не говоря уже о модификации!) файл /etc/crontab может только пользователь root. Поэтому сначала нужно получить права root, а затем уже что-либо делать с файлом /etc/crontab.

Переменная SHELL задает путь к оболочке, PATH - путь поиска программ, а MAILTO определяет имя пользователя, которому будет отправлен отчет о выполнении расписания. В таблице расписания всего одна запись - она проверяет наличие сценариев в каталогах cron.hourly, cron.daily, cron.weekly и cron.monthly и их выполнение. Об этом мы поговорим чуть позже.

Представим, что вам нужно выполнять какой-то сценарий каждый день в 8:30. Для этого в /etc/crontab нужно добавить строку:

```
30 8  *  *  *  /usr/bin/script arguments
```

Однако планировщик предлагает более удобный способ изменения таблицы расписания. Представим, что у вас есть команда, которую нужно выполнять периодически. Создайте файл myscript со следующим содержимым:

```
#/bin/bash
/путь/ myscript аргументы
```

Сохраните файл и установите право выполнения для этого файла:

```
chmod +x myscript
```

Только что вы создали простейший сценарий, который просто выполняет вашу команду с заданными аргументами. При желании, необходимости и знании bash (информацию по этой оболочке вы без особых проблем найдете в Интернете) вы можете усовершенствовать этот сценарий.

После того, как сценарий создан, его нужно поместить в один из каталогов:

- /etc/cron.hourly - содержит сценарии, которые будут выполняться каждый час;

- /etc/cron.daily - содержит сценарии, который будут выполняться ежедневно;
- /etc/cron.weekly - сюда нужно поместить сценарии, которые будут выполняться еженедельно;
- /etc/cron.monthly - содержит сценарии, которые будут выполнены раз в месяц.

Просто поместите сценарий в один из этих каталогов и положитесь на **cron** - далее вашего вмешательства не требуется.

Также существует возможность создать отдельное расписание для каждого пользователя. Для этого используется команда `crontab`. Однако такая возможность на современных серверах (когда пользователи не работают с терминалом сервера непосредственно) используется довольно редко. При этом в файл /etc/cron.deny заносятся пользователи, которым запрещено использовать планировщик cron. Если вам нужно отредактировать таблицу расписания для каждого пользователя, используйте команду `crontab`. В большинстве случаев команда будет такой:

```
crontab -e -u <имя_пользователя>
```

После этого откроется текстовый редактор (определенный в переменной окружения EDITOR) для редактирования таблицы расписания указанного пользователя. Синтаксис такой же, как для общесистемной таблицы расписания.

16.5.2. Планировщик anacron

Планировщик **anacron** - еще один форк старого-доброго планировщика cron. Ситуация с anacron такая: когда стало понятно, что cron устарел, начали появляться альтернативные планировщики, подобные ему. У каждого из них были свои преимущества и недостатки, но некоторые разработчики дистрибутивов остановили свой выбор на планировщике anacron. Однако прошло время и сейчас этот планировщик практически не используется. В основном используется планировщик cronie, который настраивается практически так же, как и cron, что и было показано ранее.

Конечно, вам могут встретиться дистрибутивы, в которых по каким-то причинам используется anacron. Как правило, это устаревшие версии дистрибу-

тивов. Если же вы обнаружили anacron в современной версии дистрибутива, то, скорее всего, это личное предпочтение разработчиков дистрибутивов.

Основное «визуальное» отличие этого планировщика - наличие файла /etc/anacrontab вместо просто /etc/crontab. Формат записей также другой:

```
Период    Задержка        ID          Команда
```

Пример таблицы расписания anacrontab:

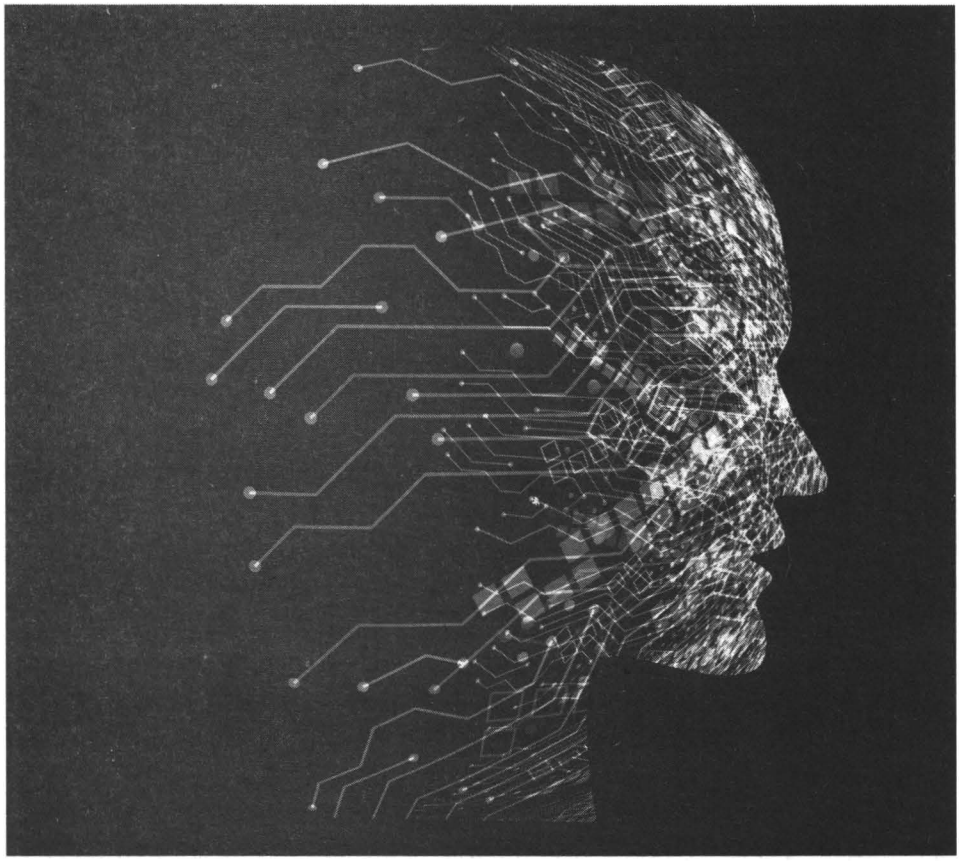
```
1  5    cron.daily       run-parts /etc/cron.daily
7  10   cron.weekly      run-parts /etc/cron.weekly
30 75   cron.monthly     run-parts /etc/cron.monthly
```

Принцип прост: как и в предыдущем случае, вам нужно поместить ваш сценарий в один из каталогов /etc/cron.daily, /etc/cron.weekly или /etc/cron.monthly (каталога cron.hourly для этого планировщика не было предусмотрено).

По сути, как только вы увидели каталоги cron.daily, cron.weekly или cron.monthly, можете сразу помещать в них свои сценарии, особо не разбираясь, какой планировщик установлен. Самое главное, чтобы служба cron была включена.

Глава 17.

Пользователи и группы



17.1. Введение в учетные записи Linux

Операционная система Linux поддерживает регистрацию и одновременную работу множества пользователей. Обратите внимание: именно одновременную работу. Раньше, еще во времена UNIX, были компьютеры, к которым подключалось несколько мониторов и клавиатур. Каждый комплект монитор + клавиатура назывался терминалом и представлял собой отдельное рабочее место пользователя. Пользователь входил в систему, а его рабочее место в системе отображалось как ttyN, где N – номер рабочего места.

Сегодня такие компьютеры уже более не востребованы, их вытеснили персональные компьютеры, которые и стали называться персональными, поскольку предполагают подключение только одного рабочего места. Мониторов можно подключить несколько, а устройство ввода – клавиатура будет одна. Но даже на таких компьютерах возможна одновременная работа нескольких пользователей. Например, вы можете войти в систему как обычно – посредством графического интерфейса. Другие пользователи смогут войти через ssh или FTP. И все вы будете работать с системой одновременно. SSH-пользователи смогут выполнять команды и получать результат их выполнения, FTP-пользователи – обмениваться с вашим компьютером файлами.

Все учетные записи можно разделить на три вида: учетные записи обычных пользователей, учетные записи системных служб и учетная запись root. С учетными записями обычных пользователей все ясно – они имеют право входить в систему разными способами (если тот или иной способ не запрещен настройками системы), для них определен домашний каталог (обычно /home/<имя_пользователя>), пароль и командная оболочка (как правило, в последнее время используется /bin/bash).

Права обычных учетных записей:

- Право на вход в систему - по умолчанию обычный пользователь может войти в систему самыми разными способами, если это не ограничено настройками системы (например, модулями PAM). Пользователь может войти локально - через консоль или в графическом режиме через дисплей-менеджер вроде gdm. Также никто не запрещает (опять-таки по умолчанию) удаленный вход, например, по SSH или FTP, если на ком-

пьютере, в который осуществляется вход, установлены соответствующие службы.

- Право на запуск программ, не требующих для своего выполнения прав root - как правило, такие программы находятся в каталогах /bin и /usr/bin. А вот из каталога /sbin запустить программу может только суперпользователь. Программы, действие которых распространяется на всю систему, например, программы изменения сетевых интерфейсов, программы разметки диска находятся в каталоге /sbin (super-bin). Чтобы запустить эти программы, пользователю нужно получить полномочия root. О том, как это сделать, будет сказано в следующем разделе.
- Обычный пользователь может создавать, удалять, читать, изменять, запускать, устанавливать права и выполнять другие операции над файлами, которые находятся в его домашнем каталоге. Как правило, это каталог /home/<имя_пользователя>. Хотя администратор может назначить пользователю любой другой каталог, хоть /users/bagira, как правило, этого никто не делает. Каталог /home может находиться физически на одном разделе, что и корневая файловая система, а может находиться и на другом разделе и даже на другом диске. На крупных серверах, как правило, под /home отводят целый диск или даже создают RAID-массивы дисков.
- Право на чтение файлов – обычный пользователь может читать большую часть файлов за пределами домашнего каталога. Исключения разве что составляют домашние каталоги других пользователей (если эти другие пользователи явно не разрешили этому пользователю читать их файлы) и некоторые файлы/каталоги в /etc. Например, файл /etc/passwd могут читать все пользователи, а вот файл /etc/shadow - только root.
- Пользователь не имеет право вносить изменения в конфигурацию всей системы, то есть устанавливать программы, изменять глобальные настройки устройств, параметры ядра, параметры загрузчика и т.д.
- Пользователь имеет право изменить свои пользовательские параметры, например, обои рабочего стола, некоторые переменные окружения, которые будут влиять только на его работу и т.д.
- Право на изменение своего пароля, но обычный пользователь не имеет право изменять пароль других пользователей.

Учетные записи системных служб не имеют право входить в систему. Для них не задан ни пароль, ни домашний каталог, а в качестве оболочки ис-

пользуется /bin/true или /bin/false – чтобы пользователь, используя учетную запись службы, не мог войти в систему через консоль. От имени таких учетных записей выполняются различные службы, например, от имени пользователя www-data выполняется веб-сервер, gdm – учетная запись для GNOME Display Manager и т.д.

Пользователь root – пользователь с максимальными правами, он может делать все:

- Право на изменение любого файла – root может читать, записывать, удалять любые файлы, в том числе и файлы в домашних каталогах других пользователей.
- Право на изменение конфигурации системы – пользователь root может изменять конфигурацию систему посредством редактирования файлов в каталоге /etc, /proc, запуска конфигураторов системы.
- Право на запуск любых программ – root может запустить любую программу, в каком бы каталоге она ни находилась.
- Право на создание, удаление, изменение (в том числе изменение пароля) других учетных записей.
- Право на установку и удаление программ.

Власть пользователя root неограниченна. Так было до определенного момента, пока не появились системы принудительного контроля доступа вроде, которые могут даже ограничить самого root. Вот только беда SELinux, LIDS, Tomoyo и другие подобные системы по умолчанию неактивны или даже не установлены, поэтому пока их не активировать пользователь root будет все равно самым главным.

> Примечание. Напомним, что когда у вас привилегии пользователя root, приглашение командной строки заканчивается символом #, а когда вы работаете как обычный пользователь - $.

17.2. Получение полномочий root

Самый простой способ получить права root – это войти как root. То есть при входе в систему вы указываете имя пользователя root и пароль, указанный при установке. Проблема в том, что не во всех дистрибутивах этот трюк сработает. Учитывая всю опасность, которую несет использование учетной за-

писи root, во многих дистрибутивах учетная запись root отключена, а вход как root ограничен самыми разными способами. Например, в том же дистрибутиве Ubuntu учетная запись root попросту отключена. Включать учетную запись root не рекомендуется – ведь злоумышленник знает, что учетка root есть везде и ему не придется угадывать имя пользователя, останется только подобрать пароль root. А так вы исключается даже саму возможность входа как root. Тем более права root можно получить другим образом, о котором мы поговорим далее. А пока рассмотрим способы блокировки входа (кроме отключения учетной записи) root.

Часто вход как root ограничен на уровне менеджера дисплея. В дистрибутивах, где используется графическая среда KDE и менеджер экрана KDM (K Display Manager) нужно отредактировать файл /etc/alternatives/kdm4-config. В нем нужно найти директиву AllowRootLogin и присвоить ей значение true, чем вы разрешите вход как root в графическом режиме.

При использовании GDM (Gnome Display Manager) вход как root ограничен не столько конфигурацией самого GDM, а столько PAM-модулями. Нужно открыть /etc/pam.d/gdm-password и найти строку, отвечающую за запрет входа, как root (рис. 17.1). Она может выглядеть, например, так:

```
auth required pam_succeed_if.so user != root quiet
```

Рис. 17.1. Файл /etc/pam.d/gdm-password

Некоторые дистрибутивы разрешают заходить, как root даже в графическом режиме. Просто они отображают предупреждение о том, что работать, как root небезопасно. Пример такого дистрибутива – CentOS.

Настоятельно рекомендуется работать в системе как обычный пользователь, а максимальные права получать только тогда, когда они вам действительно нужны. Например, когда понадобится запустить какую-то программу, требующую права root. При этом вам особо ничего не придется делать, кроме как ввести пароль.

Рассмотрим пример. Вы установили Ubuntu, при установке создали учетную запись ubuntu и задали пароль. По умолчанию инсталлятор создает первую учетную запись так, что она вносится в файл sudoers. В этом файле указываются все учетные записи, имеющие право выполнять административные задачи. Когда вы попытаетесь выполнить одну из таких задач, например, добавить нового пользователя, графический интерфейс автоматически запросит у вас ваш пароль. Это будет не пароль root, а ваш пароль, то есть пароль пользователя ubuntu (рис. 17.2). Ему разрешено выполнять административные задачи, просто система пытается убедиться, что вы – это вы, а не некто, кто оказался за вашим компьютером во время вашего отсутствия.

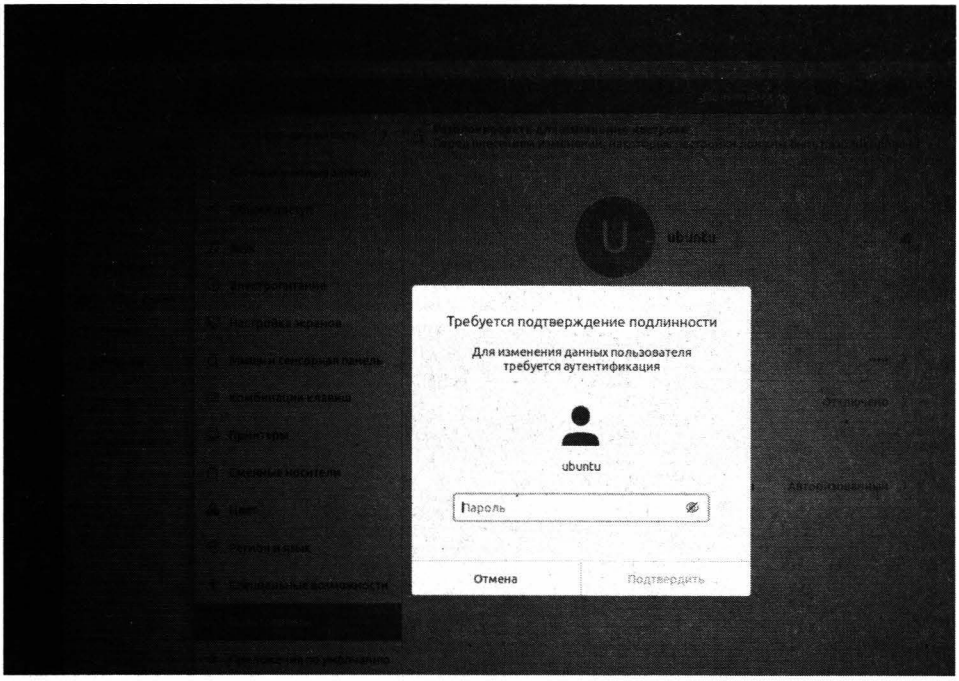


Рис. 17.2. Ввод пользователя

Когда же вам нужна командная строка с правами root, не обязательно даже переключаться в консоль. Достаточно открыть терминал и ввести команду **su**. Она запросит вас ввести пароль root. После ввода пароля вы получаете терминал с правами root. Это означает, что все команды, которые вы будете вводить после ввода команды su и успешной аутентификации, будут выполняться с правами root.

> Примечание. Если учетная запись root отключена, то вы не сможете использовать команду sudo, так как она предполагает ввод пароля root, а вы его не знаете.

Когда вы - единственный администратор, команда su - идеальный вариант. Но когда администраторов несколько, команда su - не выход, поскольку пароль root нужно будет сообщить всем остальным администраторам. Если потом возникнет нестандартная ситуация, выяснить, кто виноват будет сложнее.

На этот случай у пользователя root могут быть доверенные лица. Это может быть помощник администратора, его заместитель - называйте, как хотите. Есть лица, которым разрешено получать права root. Такие лица вносятся в файл /etc/sudoers.

Редактировать файл /etc/sudoers можно только через команду visudo (рис. 17.3):

```
export EDITOR=nano
sudo visudo
```

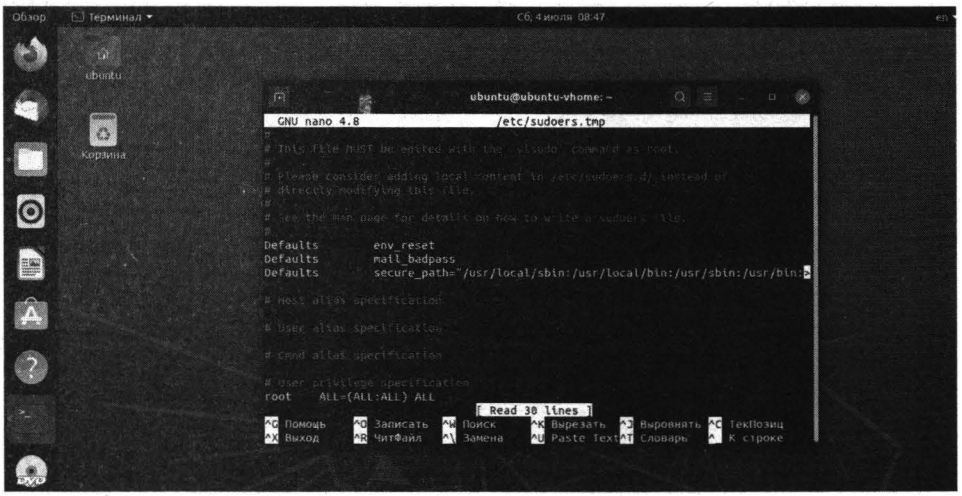


Рис. 17.3. Редактирование файла /etc/sudoers

Первая команда устанавливает переменную окружения EDITOR, задающую удобный текстовый редактор, который будет использован для /etc/sudoers. Вторая команда вызывает утилиту для редактирования файла /etc/sudoers.

Представим, что у нас есть пользователь bagira, которому нужно разрешить делать все, что можно пользователю root. Для этого нужно добавить в /etc/sudoers запись вида:

```
bagira ALL=(ALL:ALL) ALL
```

Можно также добавить запись:

```
%sudo  ALL=(ALL:ALL) ALL
```

Она означает, что членам группы sudo можно делать все, что можно делать пользователю root. Тогда всех администраторов-помощников нужно добавить в группу sudo (далее будет показано, как это сделать).

Сохраните файл и выйдите из редактора. Войдите как пользователь, которому вы предоставили право sudo. В нашем случае - это пользователь bagira. Далее введите команду, которая требует прав root через команду sudo:

```
sudo <команда>
```

Например:

```
sudo aptitude
```

Обратите внимание: **система запрашивает пароль пользователя, а не пароль root**. Пользователь указывает свой пароль, а система знает, что ему разрешено получать права root. В итоге наши помощники не знают пароль root и смогут выполнять определенные действия с правами root под *своим именем*.

Посмотрите на рис. 17.4. Хотя пользователь внесен в sudoers, ему отказано в доступе при попытке запуска программы, требующей максимальные права. Системе нужно указать, что вы явно хотите запустить такую программу и предоставить максимальные права. Для этого нужно использовать команду sudo, а в качестве аргумента – передать ей команду, которую вы хотите запустить с максимальными правами:

```
sudo visudo
```

Контролировать получение прав sudo можно командой[1]:

```
# tail /var/log/auth.log | grep sudo
```

[1] В некоторых дистрибутивах журнал аутентификации называется secure, а не auth.log

Рис. 17.4. Нюанс запуска программы с максимальными правами

Посмотрите на рис. 17.5. 4 июля 2020 года в 8:56 пользователь ubuntu пытался выполнить команду sudo для выполнения команды tail /var/log/secure. То есть в журнале отображаются не только попытки использования sudo, но и журналируются даже вводимые пользователями команды.

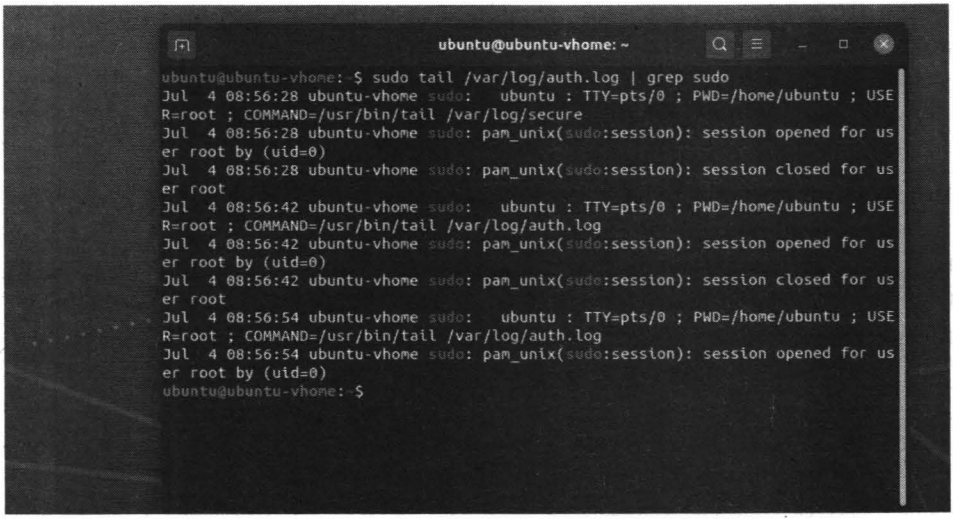


Рис. 17.5. Журнал аутентификации secure

Если вам нужно получить некоторый аналог команды **su**, чтобы вы могли вводить сразу неограниченное количество команд с максимальными правами без приставки sudo, используйте следующий трюк – запустите с максимальными правами оболочку bash. Все команды, вводимые в этой оболочке, будут выполнены с максимальными правами:

```
sudo bash
```

Закрыть такой сеанс можно командой **exit**.

Прежде, чем перейти к следующему разделу, разберемся, как включить учетную запись root в Ubuntu. Для этого нужно просто задать пароль:

```
sudo passwd root
```

Сначала нужно ввести ваш пароль, затем новый пароль для root, после этого – подтвердить пароль. После этого вы сможете войти в систему как root в консоли. Для входа в графическом режиме нужно редактировать файл, относящиеся к PAM, как было сказано ранее. Отметим, что активация пользователя root – занятие небезопасное, гораздо правильнее использовать команду sudo. На личном компьютере еще такое мероприятие допускается, но на сервере – такое делать воспрещено.

17.3. Управление учетными записями пользователей

17.3.1. Создание учетной записи пользователя

Создать новую учетную запись пользователя можно командой **adduser** или **useradd**. Чаще всего используется именно первая команда, вторая используется гораздо реже.

В большинстве случаев adduser просто добавляет в файл /etc/passwd учетную запись пользователя. В дистрибутивах Debian и Ubuntu команда adduser запрашивает контактную информацию (полное имя пользователя, номера телефонов и т.д.), а также сразу устанавливает пароль пользователя (рис. 17.6).

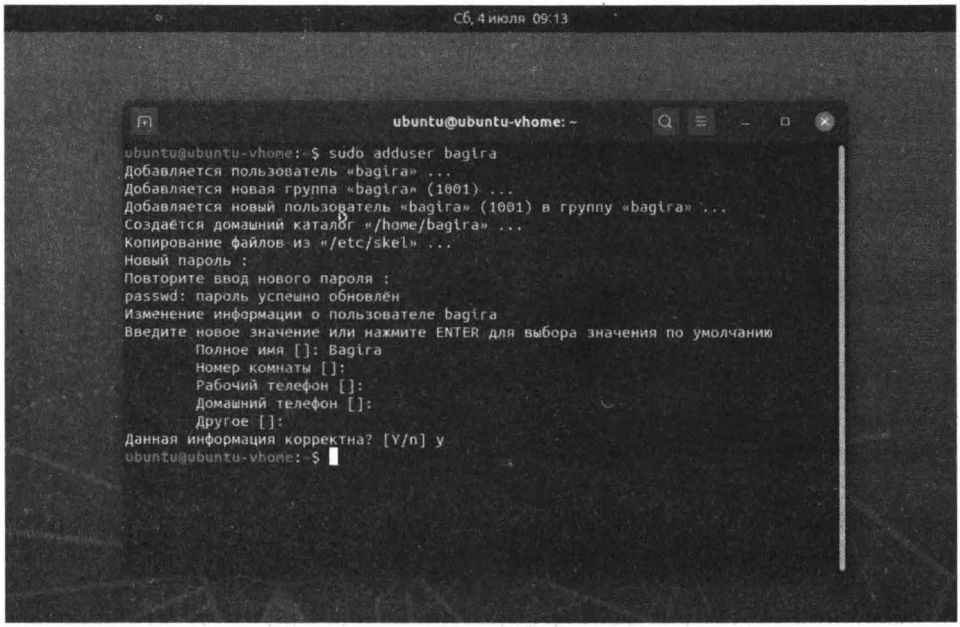


Рис. 17.6. Создание нового пользователя в Ubuntu 20.04

Если в вашем дистрибутиве команда **adduser** не запросила пароль пользователя, а просто добавила его учетную запись, тогда вам нужно еще ввести команду passwd <имя пользователя> для установки его пароля, иначе пользователь не сможет войти в систему.

Итак, в одних дистрибутивах достаточно команды:

```
# adduser <имя>
```

В других же нужно ввести две команды:

```
# adduser <имя>
# passwd <пароль>
```

Напомню, что для добавления учетной записи пользователя нужны права root, который пользователь может получить через su или sudo, если он внесен в /etc/sudoers и ему разрешена операция добавления пользователя.

17.3.2. Файлы /etc/passwd и /etc/shadow

При добавлении учетной записи происходят следующие действия (вкратце):

- Добавляется запись в файл /etc/passwd - это небольшая база данных о пользователях в текстовом формате. Этот файл могут просмотреть все пользователи.
- Если при создании учетной записи утилита запрашивает пароль, то он будет внесен в файл /etc/shadow. Пароли в этом файле хранятся в зашифрованном виде, а доступ имеет только root. Команда `passwd <имя>`, изменяющая пароль пользователя, вносит изменения как раз в этот файл.
- Создается домашний каталог /home/<имя> и в него копируется содержимое каталога /etc/skel.
- Создается почтовый ящик пользователя в каталоге /var/spool/mail.
- Владельцем каталога /home/<имя> и всех файлов и каталогов в нем назначается создаваемый пользователь.

Рассмотрим формат файла /etc/passwd:

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

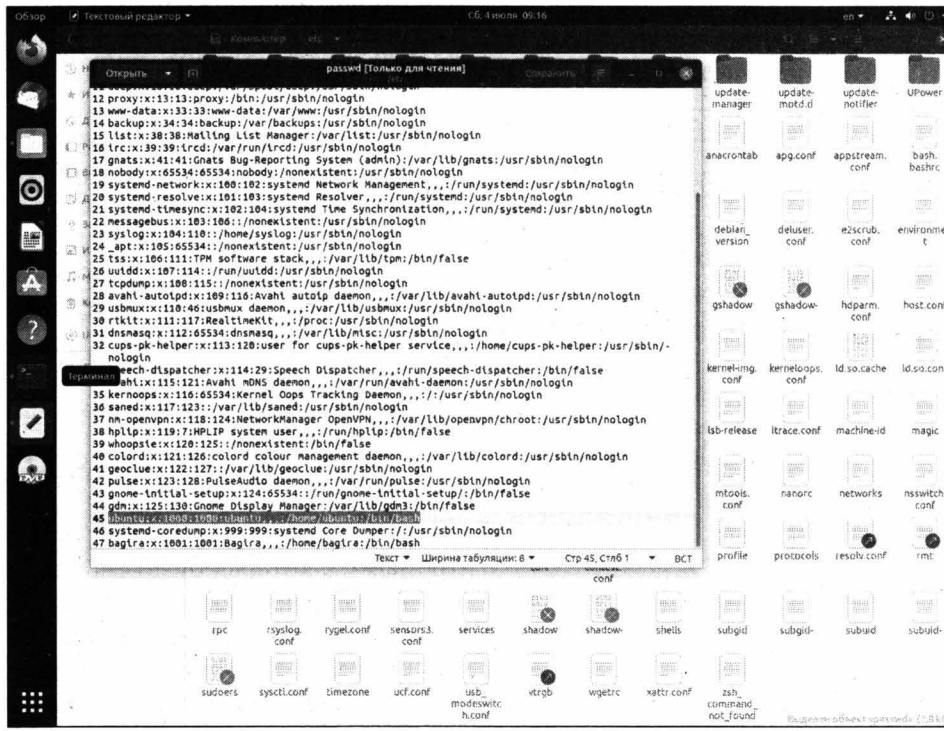


Рис. 17.7. Файл /etc/passwd

Вот фрагмент этого файла:

```
ubuntu:x:1000:1000:ubuntu:/home/ubuntu:/bin/bash
bagira:x:1001:1001::/home/bagira:/bin/bash
```

Формат файла /etc/passwd приведен в таблице 17.1.

Таблица 17.1. Формат файла /etc/passwd

Номер поля	Название	Описание
1	Имя пользователя	Имя, использующееся при входе в систему
2	Пароль	Поскольку пароль пользователя хранится в файле /etc/shadow, то в файле /etc/passwd вместо пароля просто указывается символ 'x'
3	UID	Идентификатор пользователя
4	GID	Идентификатор группы пользователя
5	Полное имя пользователя	Устанавливается администратором и ни на что не влияет. В крупных организациях имя пользователя помогает установить контакт с пользователем. Это поле может также содержать номер телефона, номер комнаты и прочую информацию, которую запрашивает adduser при создании пользователя
6	Домашний каталог пользователя	Обычно это /home/<имя_пользователя>
7	Оболочка	Программа, которая будет запущена при входе пользователя в систему из консоли (для графического режима это поле не имеет значения). Список доступных оболочек хранится в файле /etc/shells

В файле /etc/shadow полей больше, чем в /etc/passwd. Как и в случае с /etc/passwd, поля разделяются двоеточиями:

1. Имя пользователя. Совпадает с именем пользователя в файле /etc/passwd.
2. Зашифрованный пароль. Позже мы поговорим о том, как распознать алгоритм шифрования, которым был зашифрован пароль.
3. Количество дней (с 1 января 1970 года), когда пароль был сменен в последний раз.
4. Число дней до смены пароля. Если в этом поле 0, то пароль может быть сменен в любой момент.
5. Количество дней, после которых пароль должен быть сменен. Обычно здесь значение 999999, которое показывает, что пользователь может никогда не менять свой пароль.
6. Число дней, в течение которых пользователь получает предупреждение о необходимости изменить пароль. Обычно такие предупреждения пользователь получает за неделю (7 дней) до часа «X».
7. Число дней после окончания действия пароля, когда еще пользователь может работать со старым паролем. Если после этого срока пользователь не сменит пароль, учетная запись будет заблокирована.
8. Число дней, начиная с 1 января 1970, после которых пароль будет заблокирован
9. Не используется.

Обычно последние три поля не используются. По зашифрованному паролю можно понять, какой алгоритм шифрования использует система. Посмотрите на начало зашифрованного пароля:

- 1 – MD5. Ранее часто использовался, сейчас чаще используется SHA-512, поскольку в MD5 обнаружились математические уязвимости.
- 2, $2a$ – Blowfish. Чаще используется в FreeBSD/OpenBSD, чем в Linux;
- 5 – SHA-256;
- 6 – SHA-512. Используется в современных дистрибутивах.

Форматы файлов /etc/passwd и /etc/shadow были приведены «для общего развития», чтобы вы понимали, что происходит. Модифицировать учетную запись пользователя правильнее с помощью команды usermod, а не с помощью редактирования файла /etc/passwd. Конечно, можно внести небольшие изменения, например, изменить полное имя пользователя. А вот для изменения остальных параметров, например, домашнего каталога, правильнее использовать usermod, чтобы потом не делать много ручной работы.

17.3.3. Изменение и удаление учетных записей

Как было отмечено, ранее для модификации учетной записи пользователя нужно использовать команду **usermod**, но прежде поговорим об изменении пароля, так как изменение пароля - это тоже, по сути, изменение учетной записи.

Для установки и изменения пароля пользователя используется команда **passwd**:

```
# passwd <имя>
```

Если пользователь хочет изменить собственный пароль, то указывать имя не нужно:

```
$ passwd
```

А вот теперь можно приступить к рассмотрению команды **usermod**. Формат вызова этой команды следующий:

```
# usermod [параметры] учетная_запись
```

Параметры команды usermod описаны в таблице 17.2.

Таблица 17.2. Параметры команды usermod

Параметр	Описание
-a, -append	Добавляет пользователя в дополнительную группу. Используется только с параметром -G
-c, --comment комментарий	Добавляет комментарий для учетной записи пользователя

-d, --home каталог	Задает новый домашний каталог пользователя. Если указать параметр -m, то текущий домашний каталог пользователя будет перенесен в новый домашний каталог, который будет создан, если не существует
-e, --expiredate дата	Указывает дату устаревания учетной записи пользователя. По достижению этой даты учетная запись пользователя будет заблокирована. Дата указывается в формате ГГГГ-ММ-ДД. Если дату не указывать, то устаревание учетной записи будет отключено
-f, --inactive дни	После указанного числа, которые пройдут после устаревания пароля, учетная запись будет блокирована. Значение -1 означает, что эта возможность не используется, а 0 - запись будет заблокирована сразу же после устаревания пароля
-g, --gid группа	Указывает имя или GID первичной группы пользователя. Группа с таким именем/GID должна существовать. Все файлы в домашнем каталоге пользователя, которые принадлежали бывшей первичной группе, теперь будут принадлежать новой группе
-G, --groups группа1[, группа2,..., группаN]	Список дополнительных групп, в которых находится пользователь. Перечисление групп осуществляется через запятую без дополнительных пробелов. Например, -G group1,group2
-l, --login новое_имя	Изменяет имя пользователя на новое_имя.
-L, --lock	Блокирует учетную запись пользователя. Нельзя использовать этот параметр с -p или -I
-m, --move-home	Перемещает домашний каталог. Используется вместе с параметром -d

-o, --non-unique	При использовании с -u позволяет указать не уникальный UID (идентификатор пользователя)
-p, --password пароль	Шифрованное значение пароля, которое возвращает функция crypt. Использовать этот параметр не рекомендуется, поскольку другие пользователи увидят незашифрованный пароль в списке процессов
-R, -root chroot	Выполняет изменения в каталоге chroot и использует файлы конфигурации из этого каталога
-s, --shell оболочка	Задает оболочку для пользователя. Если оболочка не указана, то будет использована оболочка по умолчанию
-u, --uid UID	Задает новый UID пользователя, который должен быть уникальным
-U, --unlock	Разблокирует учетную запись пользователя
-Z, --selinux-user SEUSER	Новый пользователь SELinux для пользовательского входа

Рассмотрим несколько примеров:

```
# usermod -d /home/new_home -m ubuntu
# usermod -L bagira
# usermod -G admins,sudo mark
```

Первая команда задает новый каталог для пользователя ubuntu. Теперь он будет называться /home/new_home. Старые файлы (из каталога /home/ubuntu) будут перемещены в новый домашний каталог.

Вторая команда блокирует учетную запись пользователя bagira. Третья команда вносит пользователя mark в группы admins и sudo.

Теперь рассмотрим команду userdel (см. табл. 17.3):

```
# userdel [параметры] пользователь
```

Таблица 17.3. Параметры команды userdel

Параметр	Описание
-f, --force	Удаляет учетную запись, даже если пользователь работает в системе. Также будет удален домашний каталог и почтовый ящик, даже если другой пользователь использует тот же домашний каталог. Если в файле /etc/login.defs параметр USERGROUPS_ENAB равен yes, то будет удалена и первичная группа пользователя, даже если она является первичной и для другого пользователя. Довольно опасный параметр, который может привести систему в нерабочее состояние
-r, --remove	Удаляет домашний каталог пользователя и почтовый ящик. Файлы этого пользователя, созданные на других файловых системах, нужно искать и удалять вручную
-R, --root chroot	Выполняет изменения в каталоге chroot и использует файлы конфигурации из этого каталога
-Z, --selinux-user	Удаляет все пользовательские сопоставления SELinux для учетной записи пользователя

Пример удаления учетной записи ubuntu, домашний каталог и почтовый ящик также будут удалены:

```
# userdel -r ubuntu
```

17.3.4. Группы пользователей

Для более простого управления пользователями их можно объединять в группы. Например, можно задать ограничения ресурсов для группы пользователей. Тогда они будут распространяться на всех пользователей, входящих в группу и вам не придется их устанавливать для каждого пользователя отдельно.

Но, прежде чем устанавливать права для группы, нужно эту группу создать. Добавить группу можно командой groupadd, однако ничего плохого не случится, если вы просто отредактируете файл /etc/group (не groups, а именно group!) и добавите группу вручную. При добавлении группы следите, чтобы

ID группы был уникальным. Если же вы не хотите ни за чем следить, тогда просто введите команду **groupadd**:

```
# groupadd [параметры] имя_группы
```

С параметрами команды groupadd можно ознакомиться в справочной системе - man groupadd.

17.4. Графические конфигураторы

Управлять учетными записями пользователя можно и с помощью графических конфигураторов, что будет привычнее для бывших Windows-пользователей. В Ubuntu для управления учетными записями пользователей нужно перейти в окно **Настройки**, далее – в раздел **Пользователи** (рис. 17.8). Для управления другими учетными записями нажмите кнопку **Разблокировать**, введите свой пароль, после чего вы сможете управлять другими учетными записями – изменять их пароли, создавать и удалять учетные записи и т.д.

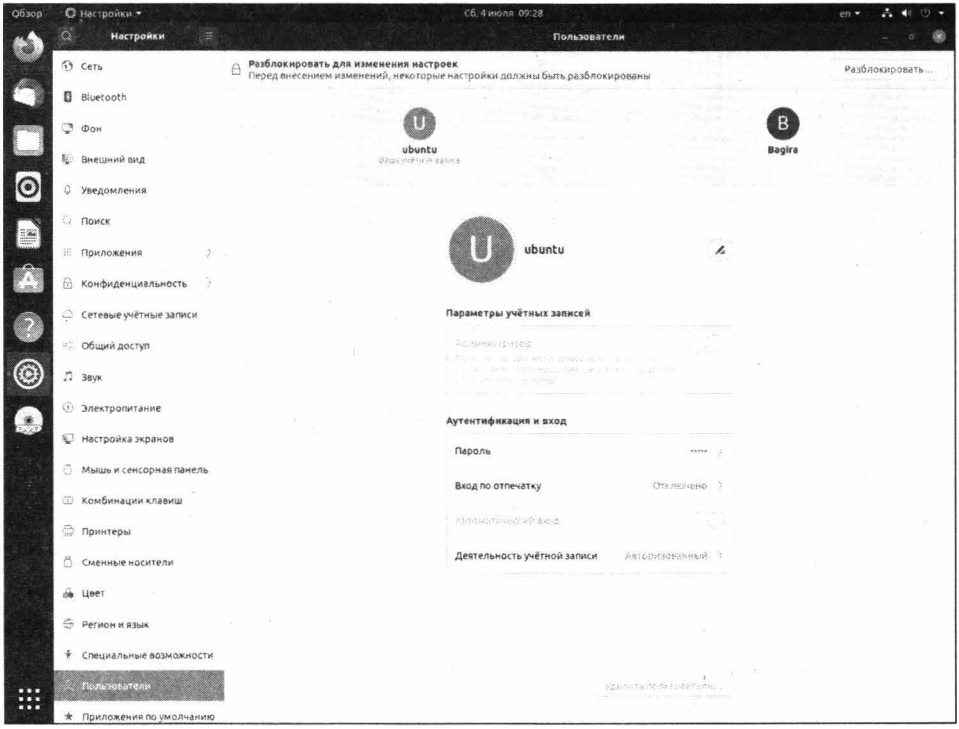


Рис. 17.8. Графический конфигуратор в Ubuntu

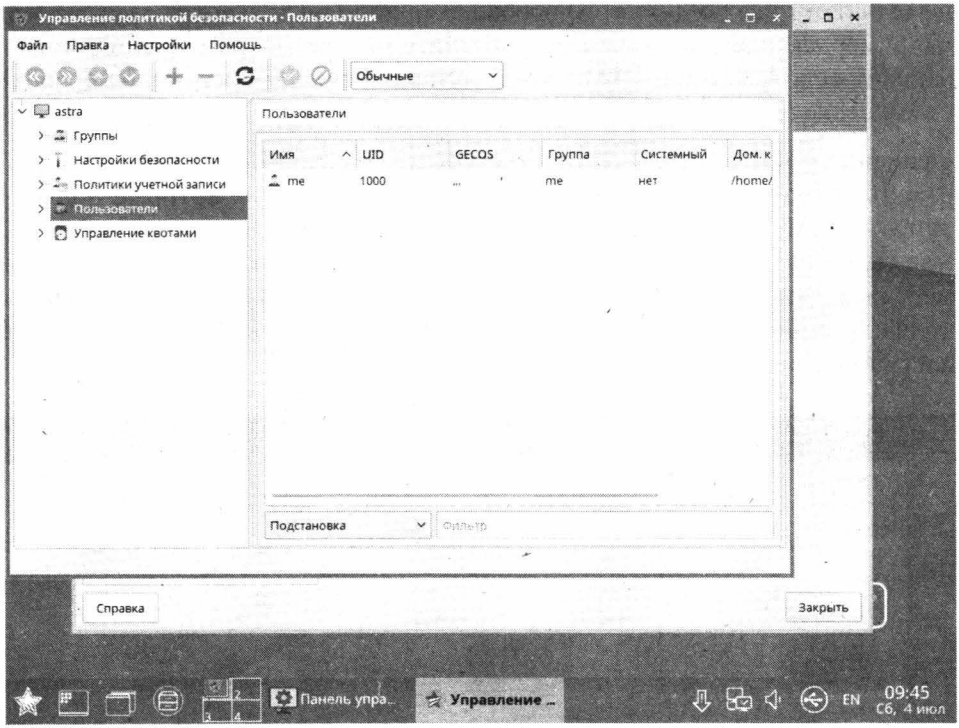


Рис. 17.9. Управление пользователями в Astra Linux

В Astra Linux нужно открыть панель управления, а далее перейти в раздел **Безопасность**, выбрать **Политика безопасности** и в открывшемся окне перейти в раздел **Пользователи** (рис. 17.9). Немного запутано, но как есть.

17.5. Модули PAM

Подключаемые модули аутентификации PAM (Pluggable Authentication Modules) предоставляют администраторам дополнительные методы подтверждения подлинности пользователя. Модули PAM - это не новинка в мире Linux. Они были разработаны очень давно, но до сих пор есть даже в самых современных дистрибутивах Linux, поскольку заменить их, по сути, нечем.

Модули PAM позволяют использовать несколько схем аутентификации. Большинство приложений, которые нуждаются в проверке подлинности пользователя, используют PAM. Модули PAM позволяют реализовать альтернативную аутентификацию, например, по отпечаткам пальцев или

по сетчатке глаз, но для этого необходимо дополнительное оборудование, например, сканер отпечатков. В этой книге мы рассмотрим традиционный вариант использования PAM - когда аутентификация происходит посредством ввода пароля с клавиатуры.

Основной файл конфигурации называется /etc/pam.conf. В каталоге /etc/pam.d/ находится конфигурация для разных сервисов, которые поддерживают PAM, например, в /etc/pam.d/sshd находится конфигурация PAM-модулей для SSH, в /etc/pam.d/gdm-passwd – конфигурация пароля для менеджера дисплея GDM и т.д. В каталоге /etc/security также есть файлы конфигурации, относящиеся к PAM, например, файл access.conf управляет доступом в систему.

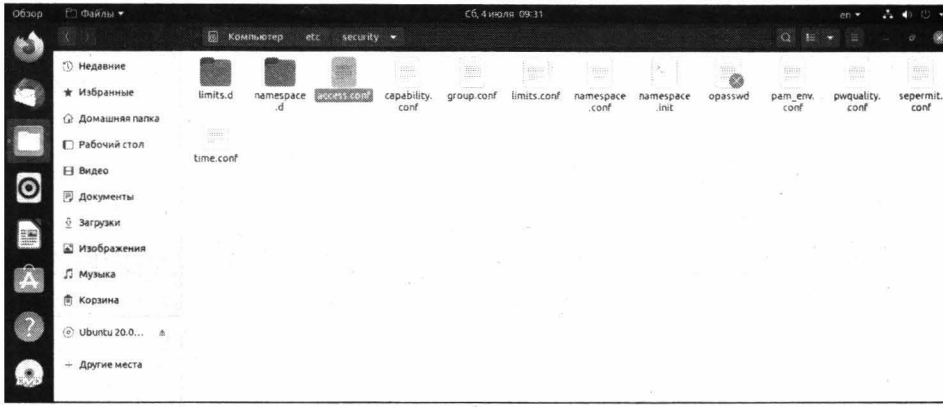


Рис. 17.10. Содержимое каталога /etc/security

Безопасность вашей системы зависит от используемых вами модулей. Модули хранятся в каталоге /lib/security или /lib64/security (для 64-битных систем), однако некоторые дистрибутивы не следуют этому стандарту. К примеру, в некоторых системах модули можно найти в каталоге /usr/lib/security. При желании можно написать и собственные модули, но для начала следует разобраться с уже имеющимися. Ниже приведен список наиболее часто используемых модулей. Больше информации по каждому из них можно получить, набрав man модуль, к примеру, man pam_pwcheck. Обратите внимание, что нет «стандартного списка» модулей. Их состав варьируется от дистрибутива к дистрибутиву.

- **pam_access** - разрешает или запрещает доступ, в зависимости от IP-адреса, имени пользователя, имени хоста или доменного имени и т.п. По умолчанию, правила доступа определены в файле /etc/security/access.conf. Когда пользователь входит, проверяются правила доступа до пер-

вого совпадения, и делается решение, разрешить или запретить доступ. Также смотри модуль pam_time - там другие ограничения.

- **pam_cracklib** и **pam_pwcheck** - предоставляют функции проверки прочности пароля (проверки на легкость угадывания или повторяемость). У пользователя спрашивают пароль, и если он проходит предустановленные правила и считается прочным, тогда нужно ввести его еще раз для проверки правильности ввода.
- **pam_deny** - безусловно запрещает доступ. Этот модуль можно использовать для блокирования пользователей, как политику по умолчанию. См. также pam_permit.
- **pam_echo** - выводит предустановленное текстовое сообщение. См. также pam_motd.
- **pam_env** - позволяет присвоение значений переменным окружения. Правила по умолчанию берутся из файла /etc/security/pam_env.conf.
- **pam_exec** - вызывает внешнюю программу.
- **pam_lastlog** - выводит дату и время последнего входа в систему.
- **pam_limits** - устанавливает ограничения на системные ресурсы, используемые пользователем. Ограничения по умолчанию берутся из файла /etc/security/limits.conf.
- **pam_listfile** - разрешает или запрещает сервис в зависимости от значений в файле. К примеру, если вы хотите открыть FTP-доступ лишь для некоторых пользователей, перечень которых указан в файле /etc/ftpusers_ok, нужно добавить строку auth required pam_listfile.so item=user sense=allow file=/etc/ftpusers_ok onerr=fail в файл /etc/pam.d/ftpd. См. также модуль pam_nologin.
- **pam_mail** - сообщает пользователю о наличии свежей электронной почты.
- **pam_mkhomedir** - создает домашний каталог пользователя, если он не существует на локальной машине. Таким образом, можно использовать централизованную авторизацию (к примеру, в NIS или LDAP) и создавать домашние каталоги лишь при необходимости.
- **pam_motd** - выводит «сообщение дня». См. также модуль pam_echo.
- **pam_nologin** - запрещает доступ, когда существует файл /etc/nologin.
- **pam_permit** - безусловно разрешает доступ - очень небезопасно! См. также модуль pam_deny.

- **pam_rootok** - разрешает доступ для пользователя root без дополнительных проверок. Обычно этот модуль используется в /etc/pam.d/su, чтобы пользователь root мог войти под любым другим пользователем даже без ввода пароля. Файл должен содержать следующие строки (обратите внимание на вторую строку, см. модуль pam_wheel):
  - auth sufficient pam_rootok.so
  - auth required pam_wheel.so
  - auth required pam_unix.so
- **pam_succeed_if** - проверяет некоторые характеристики учетной записи, к примеру, принадлежность к определенной группе, значение UID и т.п.
- **pam_time** - запрещает доступ к службе в зависимости от дня недели и времени дня. По умолчанию правила берутся из файла /etc/security/time.conf. Однако, запрет накладывается лишь на момент входа в систему. Способа принудительно заставить пользователя выйти из системы нет.
- **pam_umask** - устанавлиает маску создания файлов.
- **pam_unix** или **pam_unix2** - классическая аутентификация в UNIX-стиле, основана на файлах /etc/passwd и /etc/shadow. См. также модуль pam_userdb.
- **pam_userdb** - аутентифицирует пользователя с помощью базы данных. См. также модуль pam_unix.
- **pam_warn** - заносит название службы, номер терминала, пользователя и другие данные в системный журнал. Модуль можно использовать везде, он не влияет на процесс аутентификации.
- **pam_wheel** - позволяет root-доступ лишь для членов группы wheel. Часто этот модуль используется для su, чтобы лишь избранные пользователи могли пользоваться этой программой. Пример использования можно найти в описании модуля pam_rootok.

Если книга не посвящена отдельно PAM, лучше всего рассматривать PAM на отдельных примерах.

17.5.1. Ограничиваем доступ к системе по IP-адресу

Файл /etc/security/access.conf используется модулем pam_access.so, чтобы определить, каким пользователям позволено входить в систему и с каких IP-адресов.

Если открыть файл access.conf, то в нем будет достаточно много различных примеров, которые хорошо прокомментированы. Если вы знаете английский язык, то не составит особого труда во всем разобраться самостоятельно.

Формат этого файла следующий:

```
разрешения : пользователи : источники
```

Разрешение может начинаться с символа «+» (доступ разрешен) или «-» (доступ запрещен). Если нужно указать несколько пользователей, то их имена разделяют пробелом. Если нужно сделать исключение для некоторых пользователей, то перед их именами указывают служебное слово EXCEPT.

Третье поле может содержать список из одного или более имен консолей (tty) - для несетевого доступа к системе, имен узлов (для сетевого доступа), доменных имен (начинаются с «.»), IP-адресов узлов, IP-адресов сетей (заканчиваются «.»). Также можно указать все источники (ALL), ни один из источников (NONE) или только локальные источники (LOCAL).

Теперь несколько примеров:

```
-:ALL EXCEPT root:tty1
```

Первая консоль - это только консоль root. Другим пользователям запрещено ее занимать. Мы запрещаем доступ (-) всем пользователям (ALL) кроме (EXCEPT) пользователя root на консоли tty1.

Следующий пример - разрешение регистрации как root с определенных IP-адресов:

```
+ : root : 192.168.1.1 192.168.1.4 192.168.1.9
+ : root : 127.0.0.1
```

Если нужно разрешить регистрацию root со всей подсети 192.168.1.0, тогда укажите адрес этой подсети, указав точку вместо 0:

```
+ : root : 192.168.1.
```

Самый жесткий пример - запрещаем root вообще входить в систему:

```
- : root : ALL
```

> Примечание. Обратите внимание, что комментарии в этом файле начинаются с #, если вы хотите использовать один из примеров, приведенных в файле, убедитесь, что вы раскомментировали нужную вам строку.

Чуть выше мы разрешили вход пользователя root с определенных IP-адресов. К сожалению, одного только редактирования access.conf будет недостаточно. Нужно еще отредактировать соответствующие файлы в /etc/pam.d. Нас интересует регистрация по SSH (telnet уже не используется, поэтому вы будете регистрироваться по SSH) и обычная регистрация в системе. Поэтому нам нужно отредактировать файлы /etc/pam.d/sshd и /etc/pam.d/system-auth. В этих файлы вам нужно добавить строчку:

```
account   required  /lib64/security/pam_access.so
```

Если у вас 32-разрядная система, тогда нужно добавить немного другую строку:

```
account   required  /lib/security/pam_access.so
```

17.5.2. Ограничиваем время входа в систему

Безопасностью системы лучше управлять, когда вы бодрствуете. Поэтому имеет смысл разрешить регистрацию только в это время, например, с 8:00 до 19:00 (вдруг, кто-то немного задержится на работе).

Откройте файл /etc/security/time.conf и добавьте в него строку:

```
login;tty* & !ttyp*; !root & admin & ; !Al0800-1900
```

Здесь мы разрешаем пользователям регистрироваться только с 8:00 по 19:00. На пользователей root и admin это правило не распространяется. Также в файле time.conf вы найдете еще несколько примеров.

Как и в случае с предыдущим файлом, вам нужно изменить файлы /etc/pam.d/sshd и /etc/pam.d/system-auth, в которые нужно добавить строку:

```
account   required  /lib64/security/pam_time.so
```

или строку (для 32-разрядной системы):

```
account   required  /lib/security/pam_time.so
```

17.5.3. Ограничение системных ресурсов с помощью PAM

С помощью PAM-модулей можно ограничить системные ресурсы, что полезно для защиты системы от DoS-атаки. Принцип DoS-атаки заключается в том, что злоумышленник узурпирует все ресурсы системы, в результате обычным пользователям ничего не остается. Ограничив системные ресурсы, вы можете смягчить последствия DoS-атаки на ваш сервер. Конечно,

полной защиты этот способ не даст, но все равно, ваш сервер будет продолжать работать, хоть и медленно. Все же - это лучше, чем ничего.

Ограничить системные ресурсы можно с помощью /etc/security/limits.conf. Формат записей в этом файле такой:

```
домен      тип      ресурс          значение
```

В качестве домена указывается или **имя пользователя**, или **имя группы пользователей** (@имя). Также можно указать звездочку (*), если ограничение должно распространяться на всех пользователей.

Ограничения бывают мягкими (soft) и жесткими (hard). Мягкое ограничение можно незначительно превысить, жесткое превысить нельзя.

Возможные значения третьего поля задают тип ограничиваемого ресурса и представлены в таблице 17.4.

Таблица 17.4. Ресурсы, которые можно ограничить с помощью limits.conf

Элемент	Описание
core	Позволяет ограничить размер файла ядра (в килобайтах)
cpu	Задает максимальное процессорное время (в минутах)
data	Определяет максимальный размер сегмента данных (в килобайтах)
fsize	Позволяет указать максимальный размер файла (в килобайтах)
maxlogins	Определяет максимальное количество параллельных регистраций пользователя. По умолчанию пользователю разрешается войти неограниченное количество раз разными способами - по SSH, FTP, с разных консолей и т.д.
nofile	Задает максимальное число одновременно открытых файлов
nproc	Определяет число процессов, которые может запустить пользователь
priority	Задает приоритет, с которым будут выполняться процессы пользователя или группы.
stack	Максимальный размер стека (в килобайтах)

Последнее поле определяет значение лимита. Теперь несколько примеров:

```
*     hard    maxlogins    3
@ssh_users  hard    nproc       5
@ssh_users  hard    fsize       24576
```

В первом случае мы ограничиваем число одновременных регистраций пользователей до 3 (консоль, X11, если есть и SSH - этого более чем достаточно). Во втором пользователям из группы ssh_users мы разрешаем запускать не более 5 процессов одновременно. Также SSH-пользователям не разрешается создавать файлы размером более 24 Мб.

Обратите внимание: здесь мы просто задает лимит на максимальный размер файла. В принципе, 24 Мб этого вполне достаточно даже для хранения фотографий с зеркальной камеры и больших документов Word, содержащих изображения и другие объемные объекты. А видео и файлы большего размера пусть пользователи хранят или на своих компьютерах или входят иным способом, например, по FTP, где можно более качественно ограничить операции с файлами.

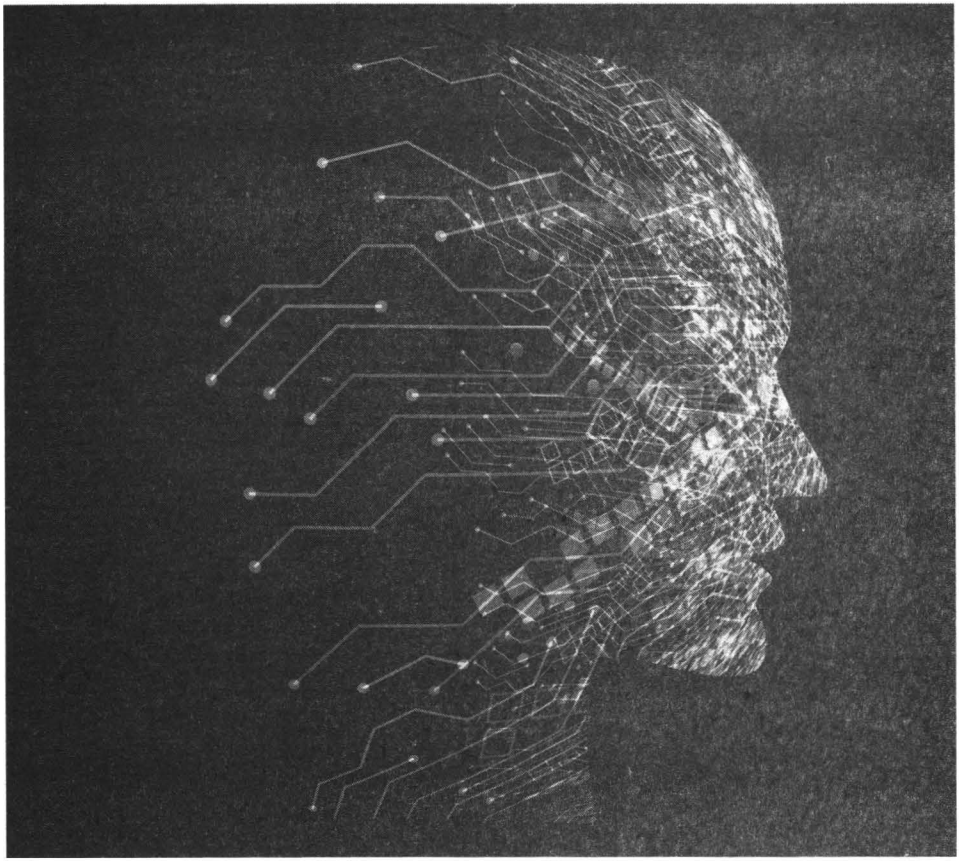
После редактирования /etc/security/limits.conf никакие другие файлы редактировать не нужно. Но описанные вами изменения будут действовать для новых сеансов пользователей, поэтому желательно перезагрузить систему, чтобы изменения действовали сразу для всех пользователей.

Если вам нужна дополнительная информация о PAM, предлагаем ознакомиться с официальной документацией, доступной по адресу:

https://mirrors.edge.kernel.org/pub/linux/libs/pam/

Глава 18.

Его величество Ядро



18.1. Что такое ядро

В этой главе мы поговорим о ядре Linux. Ядро Linux - это и есть основная программа, которую запускает загрузчик при выборе загрузочной метке. У вас (даже если установлен всего один дистрибутив) может быть установлено несколько ядер. Например, вы можете установить дистрибутив, в нем будет одна версия ядра, затем установить более новую версию ядра вручную. При загрузке вы сможете выбрать нужную вам версию ядра. На рис. 18.1 показано меню загрузчика Astra Linux, где пользователю предлагается выбрать одну из версий ядра – **generic** или **hardened**. Вторая версия повышает общую защищенность системы от взлома. Так, hardened-ядро умеет блокировать массу потенциально опасных операций, а компилятор hardened-gcc

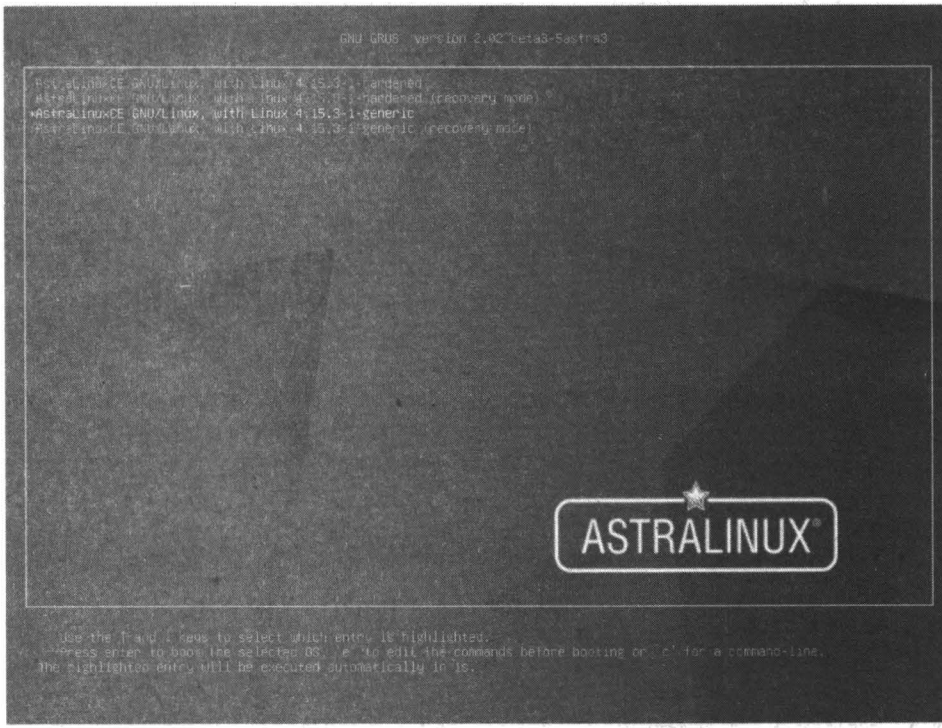


Рис. 18.1. Выбор ядра при загрузке системы

позволяет защитить компилируемые им программы от взлома типовыми методами вроде переполнение буфера. Грубо говоря, если у вас стоит «дырявая» версия программы X, и ее пытается взломать хакер, то в обычной системе у него это получится, а в hardened — не получится, да еще и в лог запись пойдет.

На данный момент последней стабильной версией ядра является версия 5.7.6 (от 24 июня 2020 года). Однако не нужно думать, что в вашем дистрибутиве будет самая последняя версия. Как правило, в вашем дистрибутиве будет именно та версия ядра, которая была стабильной на момент выпуска дистрибутива.

Номер версии ядра выводится при входе в систему (если вы входите в консоли) или его можно узнать командой uname -a (рис. 18.2). На рис. 18.2 показано, что в Ubuntu 20.04 используется версия ядра 5.4.0 – даже не 5.7.0.

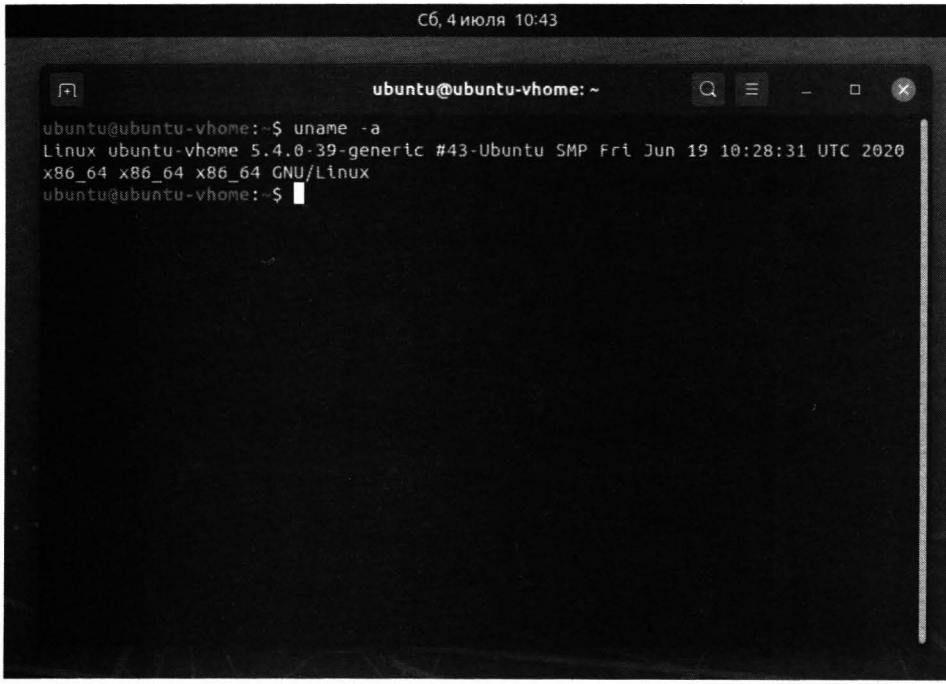


Рис. 18.2. Команда uname –a

При загрузке ядро выводит сообщения, просмотреть которые можно командой:

```
dmesg | less
```

Обычно сообщения ядра понятны сами по себе без особых комментариев. Например, в самом начале выводится версия ядра и переданные ядру параметры:

```
[    0.000000] Linux version 5.4.0-39-generic
(buildd@lcy01-amd64-016) (gcc version 9.3.0 (Ubuntu
9.3.0-10ubuntu2)) #43-Ubuntu SMP Fri Jun 19 10:28:31 UTC
2020 (Ubuntu 5.4.0-39.43-generic 5.4.41)

[    0.000000] Command line: BOOT_IMAGE=/boot/vmlinuz-
5.4.0-39-generic root=UUID=05fd3e4e-1605-478c-8db9-
b6df62a01ad3 ro quiet splash
```

Карта физической памяти:

```
[    0.000000] BIOS-e820: [mem 0x0000000000000000-
0x000000000009efff] usable
[    0.000000] BIOS-e820: [mem 0x000000000009f000-
0x000000000009ffff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000000ca000-
0x00000000000cbfff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000000dc000-
0x00000000000fffff] reserved
[    0.000000] BIOS-e820: [mem 0x0000000000100000-
0x000000001fedffff] usable
[    0.000000] BIOS-e820: [mem 0x000000001fee0000-
0x000000001fefefff] ACPI data
[    0.000000] BIOS-e820: [mem 0x000000001feff000-0x000000001fefffff]
ACPI NVS
[    0.000000] BIOS-e820: [mem 0x000000001ff00000-0x000000001fffffff]
usable
[    0.000000] BIOS-e820: [mem 0x00000000e0000000-0x00000000efffffff]
reserved
[    0.000000] BIOS-e820: [mem 0x00000000fec00000-
0x00000000fec0ffff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000fee00000-
0x00000000fee00fff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000fffe0000-0x00000000ffffffff]
reserved
```

Сообщит, что найдена SMP-таблица (значит, наша машина является многопроцессорной или хотя бы содержит процессор с несколькими ядрами):

```
[    0.000000] found SMP MP-table at [mem 0x000f6bf0-
0x000f6bff] mapped at [ffff8800000f6bf0]
```

Выведет информацию о процессоре:

```
[    0.694470] smpboot: CPU0: Intel(R) Core(TM) i5-7200U
CPU @ 2.50GHz (family: 0x6, model: 0x8e, stepping: 0x9)
...
[    0.715491] smp: Brought up 1 node, 2 CPUs
```

Частоту процессора и рейтинг в «попугаях»:

```
[    0.000001] tsc: Detected 2711.997 MHz processor
[    0.715491] smpboot: Total of 2 processors activated
(10847.98 BogoMIPS)
```

BogoMIPS - это псевдорейтинг, который показывает, сколько миллионов пустых операций процессор может выполнить за секунду. Судить о производительности по этому рейтингу можно только косвенно.

Исследуйте вывод ядра самостоятельно. Уверен, в нем вы найдете много интересного, в том числе и о своей системе.

Поскольку ядро - это программа, то ней можно передать параметры, влияющие на поведение ядра. О параметрах ядра мы поговорим в следующем разделе, а сейчас рассмотрим несколько не менее интересных моментах.

18.2. Параметры ядра

Первым делом разберемся, как передать ядру параметры. Первым делом в меню загрузчика нужно выбрать загрузочную запись, которую вы хотите отредактировать (рис. 18.1). Далее нужно нажать e для редактирования конфигурации загрузчика (рис. 18.3). Среди строк конфигурации GRUB2 найдите строку, которая начинается с со слова «linux». Это и есть строка загрузки ядра Linux. Сразу после «linux» указывается путь к ядру, а все, что после него - это и есть параметры ядра. Отредактируйте имеющиеся параметры или добавьте новые параметры в конец строки.

В главе 15 было показано, как сохранить конфигурацию GRUB2 (после перезагрузки внесенные вами изменения будут потеряны, поэтому их нужно внести в конфигурационные файлы GRUB2), а пока нажмите Ctrl + x для загрузки с отредактированной вами конфигурацией.

Самые часто используемые параметры ядра описаны в таблице 18.1.

Рис. 18.3. Редактирование параметров ядра

Таблица 18.1. Часто используемые параметры ядра Linux

Параметр	Описание
Параметры корневой файловой системы	
root=устройство	Указывает устройство, содержащее корневую файловую систему. Вы можете указать, как короткое имя устройства (/dev/sda1), так и UUID устройства
rootfstype=тип	Тип корневой файловой системы. Обычно в нем нет необходимости, поскольку ядро само определяет тип ФС

rootwait	Ядро будет ждать появления устройства с корневой файловой системой. Вы можете держать ядро на жестком диске, а корневую файловую систему на USB-диске. С точки зрения производительности может и не очень хорошо, но зато неплохо с точки зрения безопасности, когда весь жесткий диск с данными можно быстро извлечь и удалиться
rootdelay=N	Подождать N секунд перед монтированием файловой системы
ro	Монтирует корневую ФС в режиме «только чтение». После проверки утилитой fcsk корневая ФС будет перемонтирована в режим rw
rw	Монтирует корневую ФС в режим «чтение/запись». Но тогда вам нужно отредактировать сценарии инициализации системы и удалить из них вызов утилиты fsck, поскольку проверять ФС в режиме rw этой утилитой нельзя
Параметры, связанные с аппаратными средствами	
noscsi	Отключает поддержку SCSI
nousb	Отключает поддержку USB
nopcmcia	Отключает поддержку PCMCIA-карт
noapic	Отключает поддержку APIC. Полезен, если вы увидите при загрузке сообщение об ошибке, связанной с APIC
nodmraid	Отключает программные RAID-массивы, созданные на уровне BIOS
mem=xxxxM	Задает точный размер оперативной памяти, например, mem=4096M
vga=режим	Задает VGA-режим. Можно попросить ядро позволить вам выбрать один из режимов, передайте ядру такой параметр - vga=ask
Параметры управления питанием	

noapm	Отключает расширенное управление питанием APM
acpi=off	Отключает ACPI (Advanced Configuration and Power Interface)
pci=noacpi	Отключить APCI для PCI
Общесистемные параметры	
init=программа	Позволяет указать систему инициализации
reboot=тип	Указывает тип перезагрузки: warm (теплая) или cold (холодная)
single	Однопользовательский режим, который может использоваться для восстановления системы в случае сбоя
quiet	Отключает большинство сообщений ядра при загрузке системы
boot_delay=N	Задает задержку в N секунд перед выводом следующего сообщения ядра. Система будет загружаться очень медленно, лучше дождаться загрузки, выполнить команду dmesg и в спокойной обстановке прочитать все сообщения

18.3. Обновление ядра до версии 5.7

Любой Linux-пользователь может скачать исходный код ядра и откомпилировать его. Должен отметить, что в последнее время компиляция ядра используется очень и очень редко. Хотя бы потому, что в репозитариях современных дистрибутивов уже есть образы ядра, в которых активированы те или иные функции. Чаще всего переходят на более новую версию ядра для обеспечения поддержки тех или иных устройств, повышения производительности системы или же для устранения каких-либо ошибок, например, в версии 5.4 драйвер звуковой карты может работать как-то не так, зато в версии 5.7 все будет хорошо. Дабы не дожидаться, пока выйдет новая версия дистрибутива (а ждать придется, как минимум полгода), то проще установить новую версию ядра самостоятельно.

Это можно сделать, как уже было отмечено, двумя способами. Первый – компиляция ядра из исходного кода. Метод так себе:

- Вам нужно установить тонны программного обеспечения, необходимого для разработчиков. Это ПО понадобится вам только один раз, потом его придется или удалять или оно будет просто занимать много места на жестком диске. А если у вас небольшой SSD-диск, то проблема свободного места еще более актуальна для вас.
- Процесс компиляции требует наличия большого количества свободного места на диске, которого у вас может и не быть.
- Процесс компиляции требовательный к ресурсам и занимает много времени. В принципе, можно оставить компьютер на ночь и утром получить готовое ядро. Здесь уже на ваше усмотрение.

Второй способ заключается в установке уже откомпилированного ядра. Все, что вам нужно сделать – это скачать и установить несколько deb-пакетов. Способ, который подойдет большинству пользователей – он быстрый, не требует много места на диске, а результат будет тем же.

Рассмотрим, как установить ядро 5.7.0 в Ubuntu и других Debian-ориентированных системах (Mint, Astra Linux и др). Существует две основных ветки ядра – generic (для всех систем) и lowlatency (для систем, где нужна низкая задержка, например, для записи аудио, для работы с видео). Вам нужно загрузить следующие пакеты:

linux-headers-5.7.0-xxxxxx_all.deb

linux-headers-5.7.0-xxx-generic(/lowlatency)_xxx_amd64.deb

linux-modules-5.7.0-xxx-generic(/lowlatency)_xxx_amd64.deb

linux-image-xxx-5.7.0-xxx-generic(/lowlatency)_xxx_amd64.deb

В именах пакетов исправьте **generic** на **lowlatency**, если вам это нужно. Команды для загрузки будут такими:

```
cd /tmp/

wget -c https://kernel.ubuntu.com/~kernel-ppa/mainline/
v5.7/linux-headers-5.7.0-050700_5.7.0-050700.202005312130_
all.deb
```

```
wget -c https://kernel.ubuntu.com/~kernel-ppa/
mainline/v5.7/linux-headers-5.7.0-050700-gener
ic_5.7.0-050700.202005312130_amd64.deb

wget -c https://kernel.ubuntu.com/~kernel-ppa/
mainline/v5.7/linux-image-unsigned-5.7.0-050700-gener
ic_5.7.0-050700.202005312130_amd64.deb

wget -c https://kernel.ubuntu.com/~kernel-ppa/
mainline/v5.7/linux-modules-5.7.0-050700-gener
ic_5.7.0-050700.202005312130_amd64.deb
```

Мы загружаем все эти пакеты во временный каталог. После чего нужно их установить:

```
sudo dpkg -i *.deb
```

После установки загруженные deb-файлы можно удалить для экономии места:

```
rm *.deb
```

Как только пакеты будут установлены, перезагрузите систему. При перезагрузке выберите новую версию ядра в меню загрузчика GRUB2.

Если что-то пошло не так, удалить версию 5.7.0 можно командой:

```
sudo dpkg --purge linux-image-unsigned-5.7.0-050700-generic
```

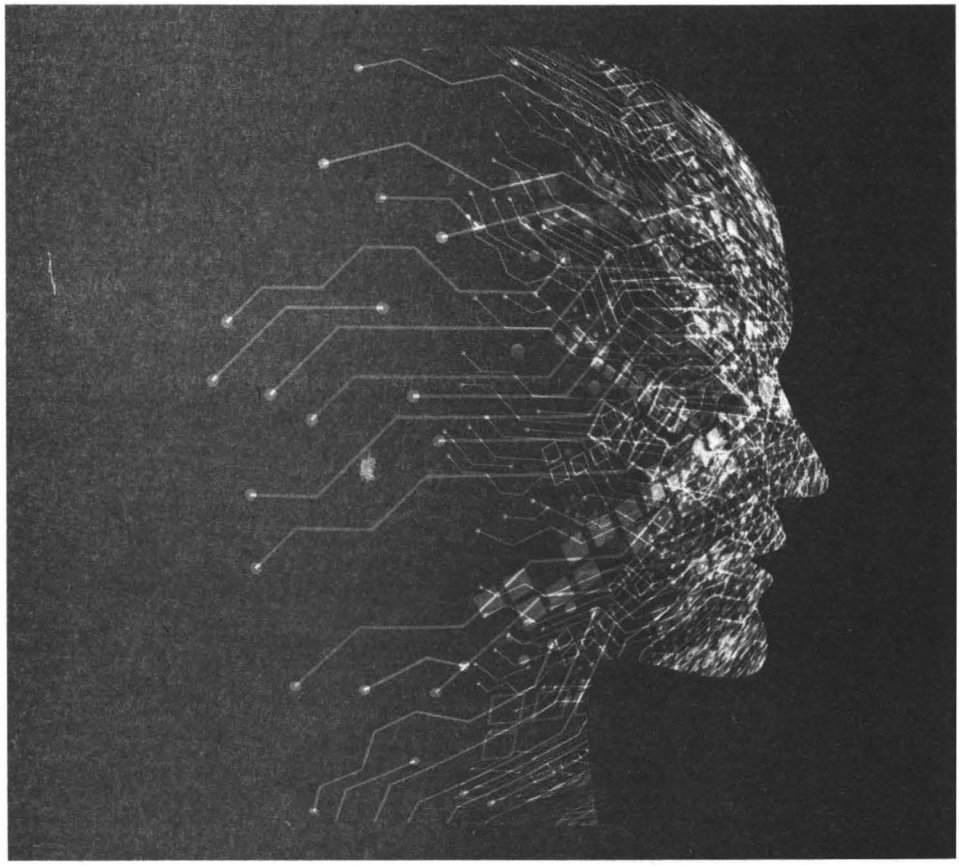
Часть IV.

Сервер для локальной сети

В этой части книги мы затрагиваем вопросы администрирования Linux-сервера в локальной сети: будет показано, как настроить серверы DNS, SSH, DHCP, FTP, поговорим об интеграции сервера в Windows-сеть, о безопасности сервера, а также настроим брандмауэр и научимся защищать сервер от сетевых атак.

Глава 19.

Маршрутизация и настройка брандмауэра



19.1. Просмотр таблицы маршрутизации

Отправляемые данные, как вы знаете, не отправляются целиком - они разбиваются на меньшие части - пакеты. Маршрутизация - это процесс перенаправления пакета по различным сетям вплоть до места назначения.

В сетях TCP/IP информация маршрутизации хранится в виде таблицы маршрутизации. Таблица маршрутизации содержит ряд простых правил. Например, если пакет отправляется в **сеть 1** он должен быть отправлен на маршрутизатор М1, если пакет отправляется в **сеть 2** то его нужно отправить на маршрутизатор М2. Пакет, отправляемый на любую другую сеть (не 1 и не 1), нужно отправить на маршрутизатор М (шлюз по умолчанию). Получив пакеты, маршрутизаторы М1, М1 и М сами знают, что с ними делать. Возможно, передать дальше другому маршрутизатору, а может отправить узлу-получателя пакета. Все зависит от такой же таблицы маршрутизации, но уже на тех маршрутизаторах. Что будут делать маршрутизаторы с полученными пакетами - это их дело, наше дело - доставить пакеты до этих маршрутизаторов.

Таблица маршрутизации ядра Linux хранит данные о маршрутизации. Каждая строка в этой таблице содержит несколько параметров - адрес сети назначения, маска сети, флаги, интерфейс и т.д.

Ядро при отправке пакета исследует таблицу машрутизации: оно определяет, в какую сеть направляется пакет - если она есть в таблице маршрутизации, то пакет отправляется через заданный в таблице интерфейс. Если нужной сети в таблице нет, пакет отправляется или на шлюз по умолчанию или же (если он не задан), отправителю пакета передается ICMP-сообщение «Network Unreachable» (сеть недоступна).

Просмотреть таблицу маршрутизации ядра можно или командой `netstat -rn` или командой `route`:

```
# netstat -rn
# route
```

Вывод команд немного отличается, как видно из рис. 19.1. Здесь видно, что шлюз по умолчанию - 192.168.52.2. По сути, это самая простая таблица маршрутизации, какая только может быть.

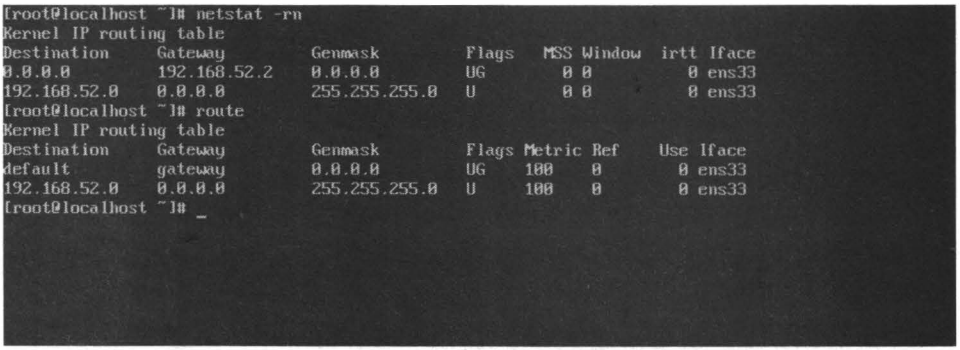


Рис. 19.1. Команды `netstat -rn` и `route`

Разберемся, что содержится в столбцах таблицы маршрутизации. Столбец Destination хранит адрес сети назначения, Gateway - шлюз (маршрутизатор), которому нужно отправить пакеты, чтобы они достигли сеть из колонки Destination.

Столбец Genmask содержит маску сети, а Flags - флаги маршрута. Флаги могут быть следующими:

- U — маршрут активен;
- H — маршрут для хоста (H - host), а не для сети;
- G — флаг шлюза (G- gateway);
- D — динамический маршрут, который был установлен демоном маршрутизации;
- M — маршрут, который был модифицирован демоном маршрутизации;
- C — запись кэширована;
- ! — запрещенный маршрут.

Колонка MSS (Maximum Segment Size) содержит значение MSS - максимальный размер сегмента для TCP-соединений по этому маршруту. Столбец Window показывает размер окна по умолчанию для TCP-соединений по этому маршруту. Столбец **irtt** — это начальное время RTT. В большинстве сетей время RTT не нужно изменять, но в некоторых медленных сетях время RTT можно увеличить, чтобы избежать лишних повторений пакетов.

Система отправляет пакет и ждет некоторое время (RTT) от получателя подтверждения получения. Если подтверждение получения не было, тогда система отправляет пакет еще раз. В медленных сетях подтверждение получения может не успеть дойти до отправителя пакета, поэтому RTT увеличивают (это можно сделать с помощью команды route). Столбец **Iface** задает интерфейс, используемый для отправки пакета.

В выводе команды route вместо столбцов MSS и Windows есть колонки Metric и Ref. Первая содержит метрику, то есть расстояние до маршрутизатора в хопах (переходах): один хоп - это один маршрутизатор. Столбец Ref содержит - это количество ссылок на маршрут. Ядром Linux этот параметр не учитывается.

19.2. Изменение и сохранение таблицы маршрутизации

Для редактирования таблицы маршрутизации используется команда **route**. Записи в таблице маршрутизации бывают статическими (добавляются командой route) или динамическими (добавляются по мере работы системы демоном маршрутизации).

Вот как можно добавить маршрут по умолчанию командой route:

```
# route add default gw 192.168.1.1 eth0
```

Думаю, эта команда понятна: 192.168.1.1 - это новый шлюз по умолчанию, а пакеты к нему будут отправляться через интерфейс eth0.

При перезагрузке таблица маршрутизации очищается, поэтому она формируется вручную, то есть командами route, а не каким-либо демоном маршрутизации, то эти команды нужно добавить в конфигурационные файлы сети или сценарии инициализации системы, чтобы внесенные вами изменения не были потеряны.

Шлюз по умолчанию можно сохранить в файле /etc/sysconfig/networking-scripts/ifcfg-имя_интерфейса (см. гл. 6). В листинге 19.1 приводится пример этого файла.

Листинг 19.1. Файл /etc/sysconfig/networking-scripts/ifcfg-имя_интерфейса

```
DEVICE=eth0
HWADDR=00:0C:29:E8:F0:C4
TYPE=Ethernet
ONBOOT=yes
NETMASK=255.255.255.0
IPADDR=192.168.1.101
GATEWAY=192.168.1.1
NETWORK=192.168.1.0
BROADCAST=192.168.1.255
BOOTPROTO=none
NM_CONTROLLED=no
```

Параметр GATEWAY позволяет указать IP-адрес шлюза. Но обычно вам не придется редактировать файлы настройки сетевых интерфейсов обычных компьютеров, поскольку такая общая информация как IP-адрес шлюза задается DHCP-сервером примерно так:

```
subnet 192.168.1.0 netmask 255.255.255.0 {
    # Список маршрутизаторов (через пробел)
       option routers                  192.168.1.1;
    ...
```

Теперь рассмотрим общий формат вызова команды route:

```
# route [операция] [тип] адресат gw шлюз [метрика] [dev
интерфейс]
```

Параметр `операция` может принимать значения **add** и **del**. Первый добавляет маршрут, а второй - удаляет. Второй параметр (`тип`) необязательный - он позволяет задать тип маршрута: default (маршрут по умолчанию), -net (маршрут к сети), -host (маршрут к узлу).

Третий параметр, `адресат`, содержит адрес сети, если вы задаете маршрут к сети, или адрес узла, если добавляется маршрут к узлу). Если вы задаете маршрут по умолчанию, то этот параметр вообще не нужно указывать.

Параметр `шлюз` задает IP-адрес шлюза. Можно также указать и доменное имя, но обычно указывается IP-адрес. Параметр `метрика` задает число переходов (маршрутизаторов) на пути к адресату. Параметр `dev` нужно указывать, если в системе установлено несколько сетевых интерфейсов и нужно

указать, через какой именно сетевой интерфейс нужно отправить пакеты. Оба последних параметра не являются обязательными.

Рассмотрим несколько примеров использования команды route:

```
# route add -net 192.168.16.0 netmask 255.255.255.0 dev
ens33
# route add -net 192.168.16.0 netmask 255.255.255.0 gw
192.168.16.1
# route add -net 10.100.0.0 netmask 255.0.0.0 reject
# route del 10.100.0.0
```

Первая команда добавляет маршрут к сети 192.168.16.0 через устройство ens33. Как видите, мы не указываем IP-адрес шлюза, а просто указали имя интерфейса - все пакеты, адресованные сети 192.168.16.0, будут отправлены через интерфейс ens33.

Вторая команда добавляет маршрут к сети 192.168.16.0 через шлюз 192.168.16.1. Все пакеты, адресованные этой сети, будут отправлены маршрутизатору с IP-адресом 192.168.16.1. В этом случае сетевой интерфейс указывать не обязательно.

Третья команда задает запрещающий маршрут. Отправка пакетов в сеть 10.100.0.0 запрещена (параметр reject). Последняя команда удаляет ранее заданный запрещающий маршрут.

Удаление маршрутов в Linux заслуживает отдельного разговора. Команда удаления маршрута выглядит так:

```
# route del IP-адрес
```

В Linux нет параметра -f (как в FreeBSD), который позволяет удалить сразу все маршруты (то есть очистить таблицу маршрутизации), поэтому вам придется ввести ряд команд route.

Как уже было отмечено, таблица маршрутизации очищается при перезагрузке. Чтобы этого не произошло, маршруты нужно определить в файлах конфигурации сети.

В Debian/Astra Linux/старых версиях Ubuntu настройка статических маршрутов производится в файле /etc/network/interfaces. Этот файл был описан в главе 6. Просто добавьте в секцию настройки интерфейса команду up и укажите команду route, которую нужно выполнить. Допустим, для настройки маршрутов вы ввели три команды:

```
route add -net 192.168.1.0 netmask 255.255.255.0 gw
192.168.17.254 eth0
```

```
oute add -net 192.168.14.0 netmask 255.255.255.0 gw
92.168.17.254 eth0
oute add -net 192.168.22.0 netmask 255.255.255.0 gw
92.168.17.254 eth0
```

'огда в файл /etc/network/interfaces нужно добавить следующие строки:

```
p route add -net 192.168.1.0 netmask 255.255.255.0 gw
92.168.17.254 eth0
p route add -net 192.168.12.0 netmask 255.255.255.0 gw
92.168.17.254 eth0
p route add -net 192.168.21.0 netmask 255.255.255.0 gw
92.168.17.254 eth0
```

3 этом примере пакеты ко всем трем сетям направляются на шлюз 92.168.17.254, а он уже сам решает, что с ними делать. Прежде, чем мы ассмотрим настройку брандмауэра, нужно отметить, как превратить ком-ьютер в шлюз. Для этого нужно разрешить пересылку пакетов протокола Pv4 (IPv4 forwarding). Для этого добавьте значение 1 в файл /proc/sys/ et/ipv4/ip_forward:

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

Конечно, после перезапуска это значение будет потеряно. Чтобы его сохра-ить, добавьте в файл /etc/sysctl.conf следующую строку:

```
et.ipv4.ip_forward=0
```

Итак, ваш компьютер теперь превращен в шлюз. Но сам по себе перена-равлять пакеты он не будет. Нужно задать ряд правил перенаправления акетов. Для этого мы будем использовать брандмауэр iptables, который меется во всех современных дистрибутивах. Именно о нем и пойдет речь следующем разделе. Забегая наперед, скажу, что iptables можно использо-ать не только для настройки шлюза, но и просто для защиты сервера или абочей станции от нежелательных подключений.

3 новых версиях Ubuntu (начиная с версии 18.04) настройка маршрутов существляется через файл /etc/netplan/01-netcfg.yaml. Откройте этот айл:

```
udo nano /etc/netplan/01-netcfg.yaml
```

3 нем вы найдете конфигурацию по умолчанию, которая может быть при-ерно такой:

```
# This file describes the network interfaces available on
your system
# For more information, see netplan(5).
network:
  version: 2
  renderer: networkd
  ethernets:
    eno1:
      dhcp4: yes
```

Добавьте следующие строки:

```
      routes:
      - to: 192.168.44.0/24
        via: 192.168.0.1
```

Данная конфигурация означает, что маршрут к сети 192.168.44.0/24 (маска 255.255.255.0) будет проходить через маршрутизатор 192.168.0.1. Полная конфигурация будет выглядеть так:

```
# This file describes the network interfaces available on
your system
# For more information, see netplan(5).
network:
  version: 2
  renderer: networkd
  ethernets:
    eno1:
      dhcp4: true
      routes:
      - to: 192.168.44.0/24
        via: 192.168.0.1
```

Обратите внимание, что YAML-файл очень требователен к отступам и разметке. Убедитесь, что оператор «routes» находится на расстоянии двух пробелов от имени интерфейса (в нашем случае eno1), к которому вы применяете маршрут.

Сохраните изменения в файле и примените их посредством команды:

```
sudo netplan apply
```

Проверим, все ли хорошо:

```
ip route show
```

Убедитесь, что вы видите эту строку в выводе предыдущей команды:

```
192.168.44.0/24 via 192.168.0.1 dev eno1 proto static
```

Если вы не видите данный статический маршрут в выводе, значит, что-то не так с вашей конфигурацией. Вернитесь к YAML-файлу и проверьте разметку/отступы. Вы также можете проверить конфигурацию командой:

```
sudo netplan try
```

19.3. Настройка брандмауэра iptables

Брандмауэр **iptables** на данный момент считается устаревшим, но до сих пор используется даже в самых новых версиях дистрибутивах. От него пытаются отказаться последние лет 5, но до сих пор этого не произошло и, на наш взгляд, в ближайшие лет 5 он будет все еще актуален. В следующем разделе мы рассмотрим современный браундмауэр ifw, который, судя по всему, вытеснит iptables в будущем, хотя на данный момент он даже не устанавливается по умолчанию.

19.3.1. Преобразование сетевого адреса

Что такое брандмауэр, я надеюсь, объяснять не нужно. А вот о преобразовании сетевого адреса (NAT, Network Address Translation) поговорить стоит, чтобы не было лишних вопросов. Пространство IP-адресов не безразмерное. Рано или поздно IPv4-адреса закончатся. Именно поэтому и был разработан протокол IPv6. Учитывая тенденции роста Интернета, думаю, что переход на IPv6 состоится в ближайшие годы, но пока будем говорить исключительно о IPv4, поскольку новый протокол IPv6 пока практически не используется.

Если бы реальные IP-адреса раздавались бы всем желающим, они бы уже давно закончились. Поэтому обычно при подключении к Интернету клиенту выдается всего один IP-адрес, даже если у вас целая организация. Этот один реальный IP-адрес, как правило, используется на шлюзе. А во внутренней сети используются локальные IP-адреса: 10.*.*.* (сеть класса A), 172.16.*.*–172.31.*.* (класс B) и 192.168.*.* (сеть класса C). Эти IP-адреса не могут пройти через маршрутизатор Интернета - как только маршрути-

затор увидит локальный IP-адрес, пакет сразу же будет отброшен. Но зато такие IP-адреса вполне подойдут для использования в локальных сетях. Данные IP-адреса уникальны только в пределах вашей организации. В соседней организации могут использоваться такие IP-адреса, как и у вас, но поскольку между компьютерами организаций нет взаимодействия, в этом нет ничего страшного.

Когда компьютер с локальным IP-адресом 192.168.1.102 отправляет пакет компьютеру, обладающему реальным IP-адресом, например, веб-серверу www.example.com, наш шлюз должен выполнить NAT. То есть он должен перезаписать локальный IP-адрес так, чтобы он выглядел как реальный. Поскольку в нашем распоряжении только один реальный IP-адрес (IP-адрес шлюза), то веб-сервер www.example.com будет «думать», что запрос пришел от нашего шлюза, и он ничего не будет подозревать о компьютерах, которые находятся за шлюзом. Веб-сервер обработает запрос и отправит пакет. Наш шлюз получит этот пакет (с ответом) и отправит его компьютеру 192.168.1.102, перезаписав поле отправителя так, что компьютер 192.168.1.102 будет «думать», что ответ пришел непосредственно от www.example.com.

19.3.2. Цепочки и правила

При настройке брандмауэра **iptables** очень важно разобраться с цепочками и правилами. Ведь основная задача брандмауэра - это фильтрация пакетов, проходящих через сетевой интерфейс. Когда брандмауэр получает пакет, он анализирует его и затем принимает решение - принять его или уничтожить. Брандмауэр может выполнять и более сложные действия, но обычно достаточно этих двух.

Обработка пакета заключается в его прохождении по цепочке правил. Каждое правило состоит из условия и действия. Если пакет соответствует условие, то выполняется указанное действие. Действие также называется целью. Если пакет не соответствует условию правила, то он передается следующему правилу. Если же пакет не соответствует ни одному из правил цепочки, выполняется действие по умолчанию.

Цепочки правил существует не сами по себе, а собираются в таблицы. Таблица **filter** является основной таблицей, отвечает за фильтрацию пакетов, в ней содержатся правила фильтрации пакетов. Таблица nat используется при преобразовании сетевого адреса. Есть еще одна таблица - mangle, кото-

рая используется, если нужно произвести специальные действия над пакетом, отличные от фильтрации и NAT.

В состав каждой таблицы входят три цепочки: INPUT, OUTPUT и FORWARD. Первая используется для входящих пакетов, вторая - для исходящих, а третья для пересылаемых пакетов. При желании вы можете создать собственную таблицу, но в ней все равно будут цепочки INPUT, OUTPUT и FORWARD.

Теперь поговорим о действиях над пакетом. Действие ACCEPT принимает пакет, DROP - уничтожает пакет. Действие MASQUERADE позволяет скрыть IP-адрес пакета. Если же в качестве действия указано имя цепочки, то пакет будет отправлен для обработки в указанную цепочку.

Рассмотрим процесс обработки входящего пакета. Сначала пакет поступает в цепочку PREROUTING таблицы **mangle**. После чего, если он не был уничтожен правилами таблицы mangle, он попадает в цепочку PREROUTING таблицы **nat**. Здесь правила проверяют, нужно ли модифицировать назначение пакета или нет. После этого пакет направляется или в цепочку FORWARD (если его нужно передать дальше - другому компьютеру) или в цепочку INPUT (если пакет адресован этому компьютеру).

Если пакет был адресован этому компьютеру, то он передается правилам цепочки INPUT таблицы **mangle** и **filter**. Если эти правила не уничтожили пакет, то он отправляется приложение, например, веб-серверу. Приложение получает данные, обрабатывает их и отправляет ответ. Ответ приложения преобразуется в пакет, который сначала обрабатывается цепочкой OUTPUT таблиц mangle, nat и filter, а затем отправляется в цепочку POSTROUTING и обрабатывается правилами таблиц **mangle** и **nat**. Если после всего этого пакет еще жив, он становится исходящим пакетом и отправляется в сеть.

19.3.3. Команда iptables

Для управления правилами брандмауэра используется команда iptables. У этой команды очень и очень много различных параметров. В этой книге не будет полного руководства по iptables, потому что есть документация (man iptables), доступная каждому пользователю. Вместо переписывания руководства другими словами мы остановимся на практическом применении iptables. Сначала рассмотрим часто используемые параметры (чтобы вы понимали, что происходит), а затем настроим шлюз для небольшой организации.

Чтобы добавить правило в цепочку, нужно использовать параметр -A:

```
# iptables -A <цепочка> <правило>
```

По умолчанию правило будет добавлено в таблицу **filter**. Если нужно указать другую таблицу, то для этого используется параметр -t:

```
# iptables -t <таблица> -A <цепочка> <правило>
```

Параметр -P позволяет задать действие по умолчанию:

```
# iptables -P INPUT DROP
# iptables -P FORWARD ACCEPT
# iptables -P OUTPUT DROP
```

В таблице 19.1 указываются возможные действия, которые вы можете использовать с iptables.

Таблица 19.1. Возможные действия над пакетами

Действие	Описание
ACCEPT	Принимает пакет и передает его дальше - в следующую цепочку
DROP	Уничтожает пакет, как будто бы его никогда не было
REJECT	Уничтожает пакет, а отправителю пакета сообщается об этом с помощью ICMP-сообщения. Параметр --reject-with позволяет уточнить тип ICMP-сообщения: icmp-host-unreachable — узел недоступен; icmp-net-unreachable — сеть недоступна; icmp-port-unreachable — порт недоступен; icmp-proto-unreachable — протокол недоступен.
LOG	Протоколирует информацию о пакете в протокол. Полезно из соображений отладки, когда вы настраиваете шлюз
RETURN	Возвращает пакет в цепочку, откуда он прибыл. Использовать не рекомендуется, поскольку возможны зацикливания - вы можете легко попасть в бесконечный цикл, если будете использовать это действие

SNAT	Выполняет подмену IP-адреса источника (Source NAT, SNAT). Данное действие используется в цепочках POSTROUTING и OUTPUT таблицы nat
DNAT	Выполняет подмену IP-адреса получателя (Destination NAT, DNAT). Поддерживается только в цепочке POSTROUTING таблицы nat
MASQUERADE	Используется в таблице POSTROUTING таблицы nat. Чем-то похоже на SNAT, но используется при работе с динамическими IP-адресами, когда возможна «потеря» интерфейса при изменении IP-адреса

Прежде, чем мы начнем создавать наш собственный шлюз, нужно рассмотреть параметры, касающиеся фильтрации пакетов (табл. 19.2).

Таблица 19.2. Параметры, касающиеся фильтрации пакетов

Параметр	Описание
--source	Указывает источник пакета. Вы можете указывать, как доменное имя, так и IP-адрес или даже целую подсеть, например, 192.168.5.0/255.255.255.0
--destination	Указывает получателя пакета. Аналогично, можно указывать, как доменное имя, так и IP-адрес
--protocol (-p)	Позволяет указать протокол. Вы можете использовать идентификаторы, определенные в файле /etc/protocols
--source-port (--sport)	Указывает порт отправителя. Опцию можно использовать вместе с параметром -p
--destination-port (--dport)	Указывает порт получателя. Можно использовать вместе с параметром -p

--state	Позволяет указать состояние пакета. Фильтр по состоянию возможен только при загрузке модуля state (-m state). Возможны следующие состояния пакета: NEW (новое соединение), ESTABLISHED (установленное соединение), RELATED (связанные с соединением пакеты), INVALID (неопознанные пакеты)
--in-interface (-i)	Указывает входящий интерфейс.
--out-interface (-o)	Указывает исходящий интерфейс

В таблице 19.2 указаны далеко не все параметры, но для организации собственного шлюза представленных параметров будет более чем достаточно.

19.3.4. Практический пример

Сейчас рассмотрим практический пример - настройку шлюза небольшой сети с нуля на базе дистрибутива Debian. У нас есть следующие вводные данные:

- Внешняя сеть - интерфейс eth0, реальный IP-адрес 88.99.88.99 (IP-адрес приведен для примера).
- Внутренняя - интерфейс eth1, IP-адрес сети 192.168.1.0/24 (у вас, скорее всего, будет другой адрес).

Простым «перебросом» пакетов мы не ограничимся, а постараемся решить попутные задачи, а именно:

- Запретить доступ сотрудников к некоторым сайтам. Администрации доступ к этим сайтам будет разрешен;
- Запретить ICQ для сотрудников, но не для администрации;
- Запретить некоторые нежелательные приложения;
- Открыть доступ к порту 80 извне для доступа к будущему веб-сайту. Все, что вам останется сделать - это установить nginx (или Apache) и ваш сайт будет виден из Интернета;
- Открыть доступ к шлюзу извне для его администрирования по SSH;
- Открыть порты 25 и 110, необходимые для работы будущего почтового сервера;

- Администратору (его IP-адрес 192.168.1.10) открываем все, что ему будет нужно.

Создайте каталог /etc/firewall, в котором мы будем хранить все необходимое для организации нашего шлюза. Сейчас создайте в ней два файла - admins.txt и black.txt. В первый нужно внести IP-адреса администрации (по одному IP-адресу в одной строке) - им будут доступны все сайты. Это может быть сотрудники IT-отдела, директор, его зам и т.д. - в общем, все вышестоящее начальство. В файл black.txt нужно внести сайты, доступ к которым нужно ограничить (тоже по одному сайту в одной строке).

```
# mkdir /etc/firewall
# touch /etc/firewall/admins.txt
# touch /etc/firewall/black.txt
```

После этого можно приступить, собственно, к настройке шлюза. Первым делом нам нужно отключить Network Manager и настроить наши интерфейсы статически. На сервере (шлюзе) Network Manager не нужен, а сетевые интерфейсы можно легко настроить с помощью /etc/network/interfaces.

Сначала введите команду:

```
# runlevel
```

Вы узнаете текущий уровень запуска:

```
# runlevel
N 2
```

Далее переходим в каталог /etc/rcX.d, где X - это уровень запуска и удаляем ссылку на network-manager. Можно также использовать команду:

```
# update-rc.d -f network-manager remove
```

Далее остановите Network Manager:

```
# /etc/init.d/network-manager stop
```

После этого отредактируйте файл /etc/network/interfaces и сконфигурируйте сетевые интерфейсы (лист. 19.2).

Листинг 19.2. Файл /etc/network/interfaces

```
# Локальный интерфейс lo
auto lo
iface lo inet loopback

# Внешняя сеть
auto eth0
iface eth0 inet static
address 88.99.88.99
netmask 255.255.255.0
network 88.99.88.99
dns-nameservers 8.8.8.8

# Внутренняя сеть
auto eth1
iface eth1 inet static
address 192.168.1.1
netmask 255.255.255.0
network 192.168.1.0
broadcast 192.168.1.255
```

Перезапустим сеть:

```
# service networking restart
```

Теперь создайте каталог /etc/firewall и поместите туда файл firewall.sh, который будет содержать команды, превращающий наш обычный компьютер в шлюз:

```
# touch /etc/firewall/firewall.sh
# chmod +x /etc/firewall/firewall.sh
# nano /etc/firewall/firewall.sh
```

Код файла firewall.sh приведен в листинге 19.3.

Листинг 19.3. Файл firewall.sh

```
#!/bin/bash

# Определяем некоторые переменные, чтобы облегчить редактирование
# конфигурации в будущем
```

```
# Внешний IP-адрес и внешний интерфейс
WAN_IP1=88.99.88.99
WAN_IFACE1=eth0

# Внутренний адрес сервера и внутренний интерфейс
LAN_IP=192.168.1.1
LAN_IFACE=eth1

# Внутренняя сеть
LOCAL_NET=192.168.1.0/24

# loopback
LO_IFACE=lo
LO_IP=127.0.0.1

# Путь к iptables
ip=/sbin/iptables

# Черный список
blacklist=( $(cat "/etc/firewall/black.txt") )
admins=( $(cat "/etc/firewall/admins.txt") )

# Очищаем все таблицы iptables
$ip -F -t filter
$ip -F -t nat
$ip -F -t mangle
$ip -F

# Правила по умолчанию: запрещаем все, что явно не разрешено
$ip -P INPUT DROP
$ip -P OUTPUT DROP
$ip -P FORWARD DROP

# Превращаем компьютер в шлюз, на всякий случай, если мы
# забыли сделать
# это раньше
echo 1 > /proc/sys/net/ipv4/ip_forward

# Включаем NAT, чтобы локальные узлы могли получать доступ к
Интернету
$ip -t nat -A POSTROUTING -o $WAN_IFACE1 -s $LOCAL_NET ! -d
$LOCAL_NET -j SNAT --to-source $WAN_IP1

# Отбрасываем INVALID-пакеты
$ip -A INPUT   -m state --state INVALID -j DROP
$ip -A FORWARD -m state --state INVALID -j DROP
```

```
# Разрешаем трафик через loopback
$ip -A INPUT -p all -i $LO_IFACE -j ACCEPT
$ip -A OUTPUT -p all -o $LO_IFACE -j ACCEPT

# Разрешаем трафик через внутренний адаптер
# Между сервером (шлюзом) и локальной сетью разрешаем все
$ip -A INPUT -p all -i $LAN_IFACE -s $LOCAL_NET --match state
--state NEW,ESTABLISHED -j ACCEPT

# Разрешаем исходящие новые и уже установленные соединения
# в внутреннюю сеть с адаптера локальной сети
$ip -A OUTPUT -p all -o $LAN_IFACE -d $LOCAL_NET --match state
--state NEW,ESTABLISHED -j ACCEPT

# Разрешаем новые и уже установленные соединения извне (с внешней сети)
# к портам 80 (веб-сервер) и 22 (ssh):

$ip -A INPUT -p tcp -i $WAN_IFACE1 -m multiport --dports
80,22,25,110 --match state --state NEW,ESTABLISHED -j ACCEPT
$ip -A OUTPUT -p tcp -o $WAN_IFACE1 -m multiport --sports
80,22,25,110 --match state --state ESTABLISHED,RELATED -j
ACCEPT

# Разрешаем выход с сервера во внешнюю сеть, но только на
# определенные порты
# Разрешаем порты 80 (HTTP), 443 (SSL) и 53 (DNS)
$ip -A INPUT -i $WAN_IFACE1 -p tcp -m multiport --sports
80,53,443 -j ACCEPT
$ip -A OUTPUT -o $WAN_IFACE1 -p tcp -m multiport --dports
80,53,443 -j ACCEPT
$ip -A INPUT -i $WAN_IFACE1 -p udp -m multiport --sports 53 -j
ACCEPT
$ip -A OUTPUT -o $WAN_IFACE1 -p udp -m multiport --dports 53
-j ACCEPT

# Открываем админу (192.168.1.10) доступ ко всему, что будет нужно
# Просмотрите список портов и откройте то, что вам будет нужно
# tcp

$ip -A FORWARD -p tcp -s 192.168.1.10 ! -d $LOCAL_NET -m
multiport --dports 80,53,443,22,25,110,5190 -j ACCEPT
$ip -A FORWARD -p tcp -d 192.168.1.10 ! -s $LOCAL_NET -m
multiport --sports 80,53,443,22,25,110,5190 -j ACCEPT

# udp
```

```
$ip -A FORWARD -p udp -s 192.168.1.10 ! -d $LOCAL_NET -m
multiport --dports 53 -j ACCEPT
$ip -A FORWARD -p udp -d 192.168.1.10 ! -s $LOCAL_NET -m
multiport --sports 53 -j ACCEPT

# Резрешаем ICQ только администраторам
i=0
for i in "${admins[@]}"
do
  $ip -A FORWARD -p tcp -d $i --sport 5190 -j ACCEPT
  $ip -A FORWARD -p tcp -s $i --dport 5190 -j ACCEPT
done

# Резрешаем избранным (список admins) доступ к сайтам из
# черного списка
j=0
for j in "${blacklist[@]}"
do
 i=0
 for i in "${admins[@]}"
 do
 $ip -A FORWARD -d $i -s $j -j ACCEPT
 done
done

# Всем остальным запрещаем доступ к сайтам из списка blacklist
i=0
for i in "${blacklist[@]}"
do
$ip -A FORWARD -s $i -j DROP
done


# Разрешаем транзит некоторых пакетов (80, 443 и 53)
$ip -A FORWARD -p tcp -s $LOCAL_NET ! -d $LOCAL_NET -m
multiport --dports 80,53,443 -j ACCEPT
$ip -A FORWARD -p tcp -d $LOCAL_NET ! -s $LOCAL_NET -m
multiport --sports 80,53,443 -j ACCEPT
$ip -A FORWARD -p udp -s $LOCAL_NET ! -d $LOCAL_NET -m
multiport --dports 53 -j ACCEPT
$ip -A FORWARD -p udp -d $LOCAL_NET ! -s $LOCAL_NET -m
multiport --sports 53 -j ACCEPT
```

После этого нужно обеспечить автоматический запуск нашего сценария /etc/firewall/firewall.sh.

19.4. Настройка брандмауэра ufw

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 – UTF (Uncomplicated Firewall) – несложный файрвол.

19.4.1. Проверяем состояние брандмауэра

Первым делом нужно убедиться, что пакет **ufw** вообще установлен или установить его, если это не так:

```
sudo apt install ufw
```

Как показано на следующей иллюстрации, уже установлена последняя версия пакета ufw.

Рис. 19.2. Установлена самая новая версия ufw

Теперь посмотрим состояние брандмауэра:

```
# ufw status verbose
```

По умолчанию фильтр пакетов выключен, поэтому вы получите сообщение

```
Состояние: неактивен
```

Не нужно спешить включать файрвол: сначала его нужно настроить. Ведь если порт 22 окажется по умолчанию недоступен, то вы потеряете доступ к своему серверу, если администрируете его удаленно. Об этом нужно помнить!

19.4.2. Базовая настройка

По умолчанию брандмауэр запрещает все входящие соединения и разрешает все исходящие. Такая политика идеальная с точки зрения безопасности (дальше вы поймете почему) – ведь если кто-то (и вы в том числе) захочет к нему подключиться, что 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
```

Данные два правила как раз и задают политику по умолчанию – запрещаются все входящие соединения и разрешаются все исходящие.

Итак, все входящие соединения запрещены. Чтобы к серверу 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, нужно явно указать номер порта:

```
ufw allow 3333
```

После разрешения **ssh** (это главное, чтобы сейчас файрвол нам не разорвал соединение) можно включить **ufw** командой:

```
ufw enable
```

Вы увидите сообщение о том, что межсетевой экран включен и будет запускаться при загрузке системы.

Посмотрите на следующий скриншот (рис. 19.3).

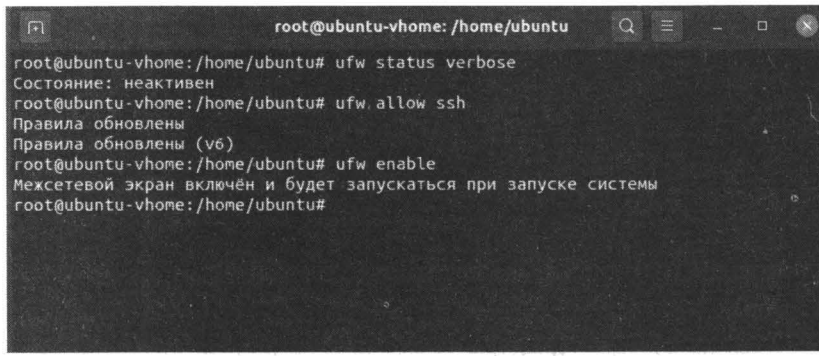


Рис. 19.3. Процесс настройки ufw

Посмотрим, что произошло. Сначала мы разрешили ssh, на что получили ответ, что правила обновлены:

```
Правила обновлены
Правила обновлены (v6)
```

Затем мы включаем файрвол и получаем 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 фильтра пакетов.

19.4.3. Создаем правила для других приложений

Теперь нужно разрешить работу других приложений. Как правило, нужно разрешить службу http (веб-сервер), ftp 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
```

19.4.4. Разрешаем IP-адреса

Ufw позволяет разрешить определенному IP-адресу доступ ко всем портам сервера, например:

```
ufw allow from 46.229.220.16
```

Если нужно разрешить доступ конкретному IP-адресу только к определенному 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 администратора динамический IP), можно так:

```
ufw allow from 123.45.67.89/24 to any port 22
```

19.4.5. 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
```

19.4.6. Удаление/сброс правил

Сбросить все правила можно командой:

```
ufw reset
```

Но убедитесь, что на момент ввода этой команды вы отключили файрвол, иначе вы потеряете доступ по ssh.

Удалить конкретное правило можно по номеру. Сначала введите следующую команду, чтобы узнать номер правила:

```
ufw status numbered
```

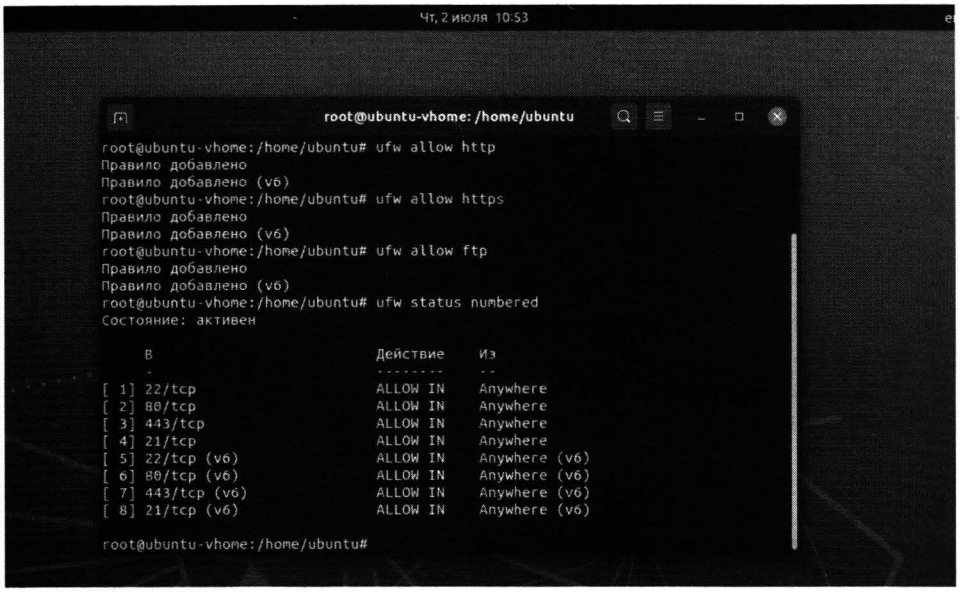


Рис. 19.4. Список правил

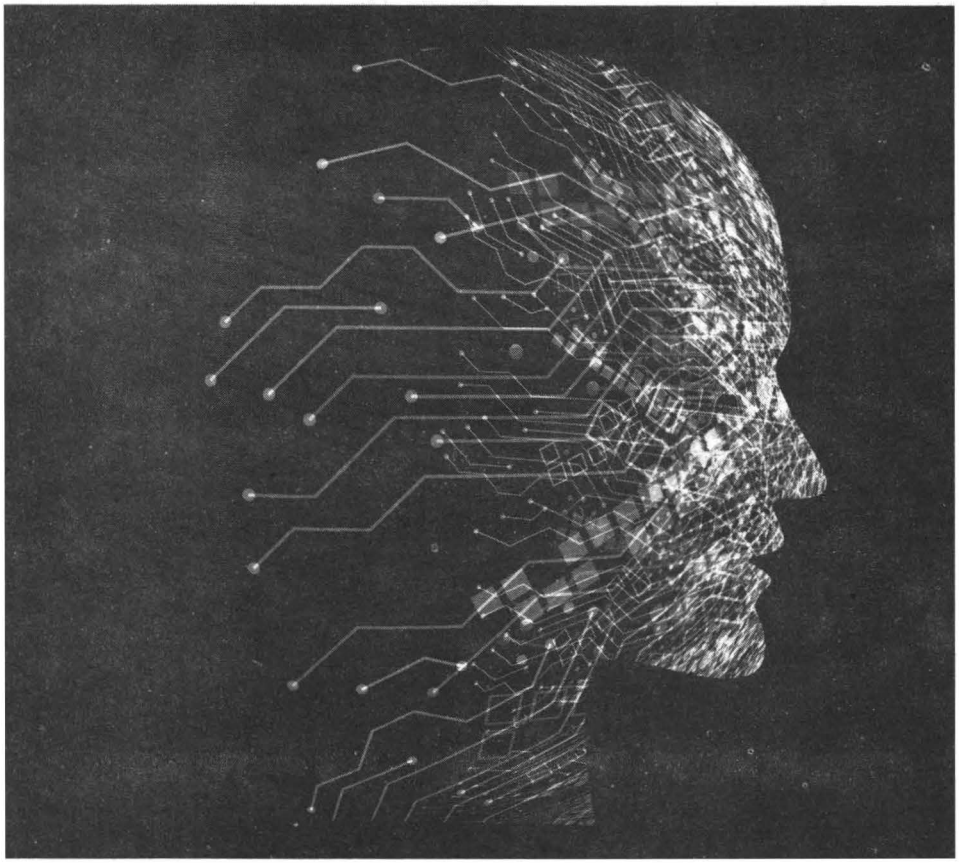
19.4.7. Отключение файрвола

Отключить **ufw** можно командой ufw disable. Отключение может понадобиться, если какие-то из правил работают неправильно и нужно временно отключить файрвол, чтобы разрешить работу тех или иных сервисов

Если вы ранее использовали **iptables**, то наверняка заметили, что синтаксис ufw гораздо проще. Если же до этого вам не приходилось настраивать брандмауэр, то ufw – оптимальное решение, с которым не составит труда разобраться даже начинающему администратору.

Глава 20.

Удаленный вход в систему по SSH



20.1. Протокол SSH

Особенностью SSH-соединения является шифрование всех передаваемых по нему данных. Если злоумышленник перехватит SSH-трафик, он не узнает, ни логин, ни пароль, ни команды, которые вы передавали на сервер.

В Linux используется свободная реализация протокола SSH - OpenSSH. Данная реализация была определена рабочей группой IETF. Не беспокойтесь: OpenSSH так же безопасен, как и SSH. Далее мы будем говорить SSH, а подразумевать именно OpenSSH.

Для шифрования передаваемых данных SSH-соединение может использовать различные алгоритмы, например, BlowFish, 3DES (Data Encryption Standard), IDEA (International Data Encryption Algorithm) и RSA (Rivest-Shamir-Adelman algorithm). Алгоритм определяется настройками SSH-сервера.

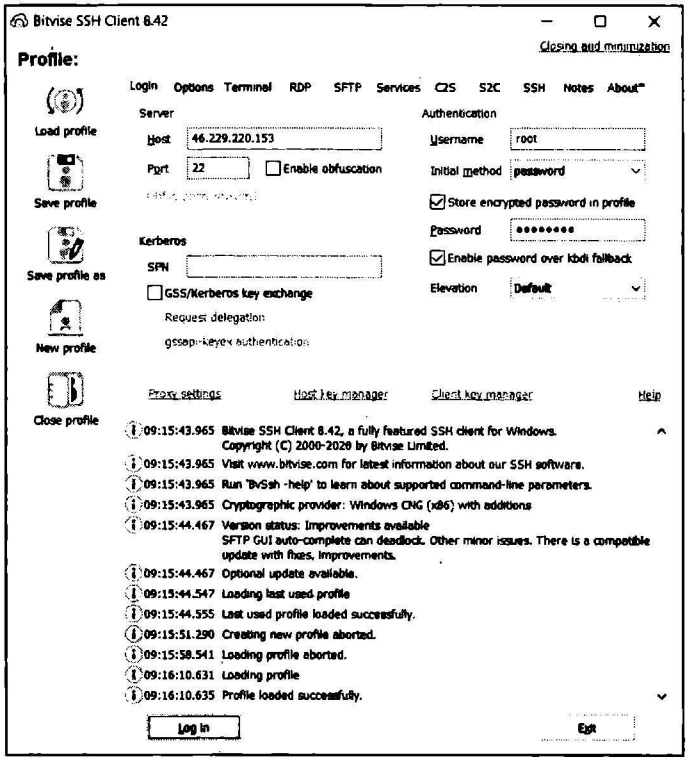


Рис. 20.1. Приложение Bitvise SSH Client

Чтобы удаленные пользователи могли подключиться к вашему серверу, на нем нужно установить демон sshd. Как правило, этот демон содержится в пакете openssh-server. Клиент SSH, то есть программа, с помощью которой удаленные пользователи будут подключаться к вашему SSH-серверу, содержится в пакете openssh-client. Итак, на компьютеры, с которых вы планируете подключаться к серверу, нужно установить пакеты openssh-client.

Что делать, если у вас не Linux, а подключаться к SSH-серверу все равно нужно? Не беспокойтесь! SSH-клиенты созданы для самых разных операционных систем. Один из самых хороших клиентов для Windows – Bitvise SSH Client. Он бесплатный, поддерживает создание и загрузку профилей соединений, что позволяет удаленно администрировать несколько серверов, поддерживает не только передачу команд, но и файлов (по протоколу sFTP). Для iOS можно использовать приложение WebSSH, а для Android – SSH Client. Все эти приложения бесплатны.

20.2. SSH-клиент

Программа ssh является SSH-клиентом и содержится в пакете openssh-client. Как правило, этот пакет установлен по умолчанию и вам не нужно его устанавливать вручную.

Формат вызова программы ssh следующий:

```
ssh [параметры] <адрес_удаленного_компьютера>
```

Параметры можно не указывать, но раз они есть, знать о них вы обязаны (табл. 20.1). Хочу предупредить сразу: опций у программы много и в таблице 20.1 будут представлены только самые полезные или наоборот, бесполезные (чтобы привлечь к ним ваше внимание) опции.

Таблица 20.1. Некоторые параметры программы ssh

Параметр	Описание
-1	Заставляет клиент использовать первую версию протокола SSH. Можно использовать только при подключении к очень старым серверам

-2	Клиент будет использовать только вторую версию протокола SSH. Это означает, что если вы с этим параметром попытаетесь подключиться к старому серверу, у вас ничего не выйдет. Как правило, ssh автоматически определяет версию протокола и в параметрах -1 и -2 нет смысла
-4	Клиент будет использовать только IPv4-адреса
-6	Клиент будет использовать только IPv6-адреса
-A	Включает перенаправление соединения агента аутентификации. Данный параметр можно включить отдельно для каждого узла в конфигурационном файле. Используйте перенаправление агента с осторожностью. Данная возможность является потенциально уязвимой
-a	Отключает перенаправление соединения агента аутентификации
-b адрес	Использует адрес на локальной машине, как адрес источника соединения. Полезная опция, если у вас установлено несколько IP-адресов
-C	Запрашивает сжатие всех данных (в том числе stdin, stdout, stderr и данные X11). Уровень сжатия устанавливается опцией CompressionLevel для протокола версии 1. Сжатие данных полезно на модемных линиях и других медленных соединениях, но в быстрых сетях оно только вызовет лишние задержки
-c	Задает список алгоритмов шифрования, разделенных запятыми в порядке предпочтения. Вы можете указать алгоритмы blowfish, des или 3des. Данная опция используется только для второй версии протокола SSH. Для первой версии протокола можно указать лишь один предпочитаемый протокол
-f	Переводит ssh в фоновый режим. Полезно, когда вы запускаете X11-программу (графическую программу) по SSH

-l имя	Позволяет указать имя пользователя, под которым вы будете регистрироваться на SSH-сервере. Указывать не обязательно, поскольку сервер и так попросит вас ввести логин
-p порт	Указывает порт SSH-сервера, отличный от используемого по умолчанию
-q	«Тихий» режим. В нем отображаются только фатальные ошибки. Все, что не важно, выводиться не будет
-X	Включает перенаправление X11. Полезный параметр при запуске графических программ
-x	Отключает перенаправление X11
-v	Подробный режим - антипод для -q
-V	Выводит номер версии и выходит

Использовать **ssh** очень просто. В следующем примере я подключаюсь как root к узлу **server**:

```
$ ssh -l root server
```

20.3. Настройка SSH-сервера

Теперь приступим к настройке SSH-сервера. Для его установки нужно установить пакет `openssh-server`. После этого нужно запустить сервер командой:

```
sudo systemtl start ssh
```

Иногда сервис называется sshd, тогда команда будет иной:

```
sudo systemtl start sshd
```

Сервер сразу же готов к работе и его вполне безопасно можно использовать с параметрами по умолчанию.

Конфигурационный файл SSH-сервера называется `/etc/ssh/sshd_config`. Пример файла конфигурации sshd_config приведен в листинге 20.1 (с комментариями на русском языке).

Листинг 20.1. Пример файла конфигурации sshd_config

```
# Какой порт будет использоваться для SSH-сервера.
# Порт по умолчанию - 22
# Теоретически, номер порта можно изменить, но на практике
# в этом нет необходимости. Защита с помощью изменения номера
# порта - дело неблагодарное, поскольку есть сканеры портов,
# которые могу легко вычислить номер порта
Port 22
# Какие адреса мы будем слушать. Чтобы sshd работал на всех
# интерфейсах, закомментируйте директиву ListenAddress
#ListenAddress ::
#ListenAddress 0.0.0.0

# Номер версии протокола
Protocol 2
# Ключи для протокола версии 2
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_dsa_key
HostKey /etc/ssh/ssh_host_ecdsa_key

# Для улучшения безопасности включаем разделение привилегий
UsePrivilegeSeparation yes

# Время жизни (в секундах) и размер ключа сервера версии 1
KeyRegenerationInterval 3600
ServerKeyBits 768

# Журналирование
SyslogFacility AUTH
LogLevel INFO

# Аутентификация:
LoginGraceTime 120
# Если нужно запретить вход как root по ssh
# (это не запрещает команду
# su), выключите этот параметр (установите значение no).
PermitRootLogin yes
StrictModes yes

# Максимальное количество попыток аутентификации
#MaxAuthTries 3

# Использование RSA (yes)
RSAAuthentication yes
# Аутентификация с открытым ключом
```

```
PubkeyAuthentication yes
#AuthorizedKeysFile %h/.ssh/authorized_keys

# Отключаем устаревшую .rhosts-аутентификацию
# Файлы ~/.rhosts and ~/.shosts читаться не будут:
IgnoreRhosts yes
RhostsRSAAuthentication no
HostbasedAuthentication no

# Запрещаем (значение no) пустые пароли
PermitEmptyPasswords no

# Не используем аутентификацию вызов-ответ
ChallengeResponseAuthentication no

# Параметры Kerberos
#KerberosAuthentication no
#KerberosGetAFSToken no
#KerberosOrLocalPasswd yes
#KerberosTicketCleanup yes

# Параметры GSSAPI
#GSSAPIAuthentication no
#GSSAPICleanupCredentials yes

# X11-форвардинг
X11Forwarding yes
X11DisplayOffset 10
# Выводить сообщение дня (можете отключить)
PrintMotd yes
# Выводить время последнего входа
PrintLastLog yes
# Использовать постоянные TCP-соединения
TCPKeepAlive yes
#UseLogin no

#MaxStartups 10:30:60
# Баннер
#Banner /etc/issue.net

# Разрешать клиенту передавать локальные переменные окружения
AcceptEnv LANG LC_*

# Параметры подсистемы sftp
Subsystem sftp /usr/lib/openssh/sftp-server
```

```
# Использовать ли модули PAM?
UsePAM yes
```

Конфигурация по умолчанию вполне пригодна для использования. Я выделил несколько опций, которые вам, возможно, захочется изменить. Остальные параметры вряд ли вы будете изменять.

20.4. Защищенное копирование файлов

Иногда возникает необходимость скопировать важные файлы по защищенному каналу. Заморачиваться с отправкой их почтой с использованием асинхронного шифрования как-то не хочется, да и почтой можно передать только файлы небольших размеров. Для защищенного копирования файлов используется утилита **scp**.

Рассмотрим несколько примеров использования этой команды:

```
$ scp user@example.com:file.txt /home/ubuntu
```

Данная команда копирует файл file.txt с удаленного SSH-сервера example.com в каталог /home/ubuntu. Вход на сервер будет выполнен от имени пользователя user. Пароль будет запрошен при запуске программы. Файл file.txt должен находиться в домашнем каталоге пользователя user.

Аналогично, можно скопировать некоторый локальный файл в некоторый удаленный каталог на SSH-сервере:

```
$ scp file.txt user@examle.com:/some/remote/directory
```

Теперь усложним задачу. Пусть нам нужно скопировать локальный каталог dir1 и все его содержимое (опция -r) в каталог /home/ubuntu/dir2 удаленного сервера example.com. Команда будет такой:

```
$ scp -r dir1 user@example.com:/home/ubuntu/dir2
```

А вот совсем сложный пример. Мы копируем файл /dir/file.txt, находящийся на сервере host1, в каталог /some/directory сервера host2. На обоих компьютерах должен быть запущен SSH-сервер. Команда выглядит так:

```
$ scp user@host1:/dir/file.txt user@host2:/some/directory/
```

20.5. Оптимизация SSH

SSH-сервер работает довольно быстро, но иногда администраторам приходится столкнуться с тормозами особенно при входе пользователя. Ускорить вход пользователя поможет отключение использования DNS. Добавьте в файл конфигурации sshd_conf директиву:

```
UseDNS no
```

В этом случае IP-адреса не будут разрешаться в доменные имена, что существенно повысит производительность, поскольку не будет необходимости обращаться к DNS-серверу и ждать от него ответа.

Также нужно использовать постоянные TCP-соединения:

```
TCPKeepAlive yes
```

Еще можно отказаться от вывода сообщения дня, но на вход пользователя это особо не повлияет:

```
PrintMotd no
```

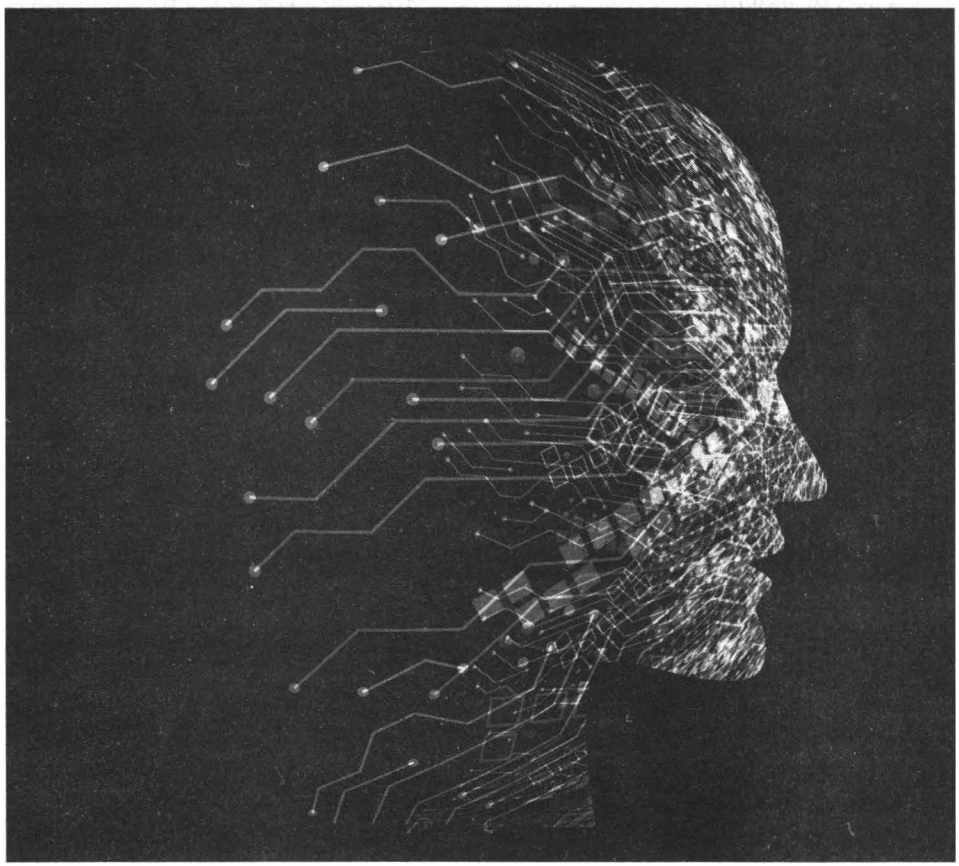
Серьезное «торможение» (порой на несколько секунд) могут добавить PAM-модули. Но отключать PAM-модули полностью не нужно! Достаточно из файлов /etc/pam.d/login и /etc/pam.d/sshd нужно удалить строки:

```
sessionoptional  pam_motd.so   motd=/run/motd.dynamic
noupdate
session   optional  pam_motd.so
```

После этого регистрация на вашем SSH-сервере будет происходить мгновенно.

Глава 21.

Общие вопросы администрирования веб-сервера



Прежде, чем приступить к разработке своего Интернет-магазина, нужно проделать много подготовительной работы – выбрать сервер, доменное имя, SSL-сертификат, настроить программное обеспечение сервера и установить саму CMS (Content Management System, система управления контентом). Только после этого можно начать продавать что-либо, то есть занять сервер полезной работы.

21.1. Выбор доменного имени

Как яхту назовешь, так она и поплывет. Конечно, все зависит от личных предпочтений руководства магазина, уже существующего названия торговой марки и других факторов. Как технический специалист, могу порекомендовать выбирать доменное имя второго уровня и желательно в домене .com (<название>.com), поскольку вы создаете именно Интернет-магазин, а не сайт общественной или правительственной организации.

С точки зрения SEO доменное имя второго уровня лучше, чем третьего, например, sales.example.org. А домен верхнего уровня (TLD) .com идеально подходит для коммерческих ресурсов.

Купить доменное имя можно у регистрации доменных имен, например, на reg.ru, nic.ru и т.д. Если у вас сервер будет не физический, а виртуальный, совсем не обязательно покупать доменное имя у облачного провайдера. Хотя, если вам удобнее, вы можете купить и сервер, и домен у одной организации.

21.2. Выбор типа сервера

Для вашего Интернет-магазина нужен собственный сервер, просто хостинг не подойдет по причинам низкой производительности и невозможности гибкого управления программным обеспечением. Хотя некоторые провайдеры предоставляют уже готовый хостинг с ПО для организации магазина, но как будет работать такой магазин – никому не понятно. Я же рекомендую не тратить свое время на подобные продукты (исключение SaaS-услуга

с Интернет-магазином, например, Shopify, но это уже совсем другой уровень и если вы выбрали такую, то можете дальше эту книгу не читать и купить книгу по Shopify).

Итак, нужен собственный сервер. Возможны три варианта:

1. Физический сервер
2. Виртуальный сервер (VDS/VPS)
3. Выделенный (dedicated) сервер

От первого варианта прошу вас отказаться. Во-первых, хороший сервер стоит несколько тысяч евро. Это капитальные затраты, которые можно заменить операционными посредством использования виртуального сервера. Физический сервер можно использовать, если он уже был ранее куплен до вас.

Во-вторых, физические серверы требуют существенных затрат на их обслуживание:

1. Нужно производить ремонт, модернизацию, чистку. В случае с виртуальным сервером – все это ложится на плечи провайдера. Вам ничего не нужно делать и никаких затрат.
2. Нужно обеспечить резервное питание. Это не всегда возможно, поскольку часто альтернативным источником выступает дизель-генератор, а его можно установить не везде (правила пожарной безопасности и уровень шума мешают установить его в обычном офисном здании).
3. Необходимо вторая Интернет-линия и роутер (маршрутизатор) с возможностью переключения между каналами. Роутер стоит недорого, а вот за второй Интернет-канал вам придется платить каждый месяц, даже если вы его не используете. Хотя этот пункт менее проблематичен, чем второй.
4. Необходимо обеспечить систему кондиционирования летом во избежание перегрева сервера.

Виртуальный сервер ничем по управлению не отличается от физического – вы можете установить любое программное обеспечение и настроить сервер так, как вам нужно. Зато у виртуального есть масса преимуществ:

- Перед установкой какого-то расширения или внесений существенных изменений в работу CMS или самого сервера, вы можете сделать сни-

мок (снапшот). В случае если что-то пойдет не так, восстановить «все, как было» можно за считанные секунды. Это основное преимущество использования виртуализации. На продакшн-серверах данная функциональность «must have» – пользователи вашего магазина не должны ждать, пока администратор сделает откат при неудачной настройке.

- Модернизация сервера занимает считанные минуты. Например, вы можете докупить дополнительные ядра процессора и дополнительные гигабайты оперативной памяти за считанные минуты. Более того, когда дополнительные ресурсы вам будут не нужны, вы можете вернуть их в пул и платить меньше. Например, на время сезонных распродаж (Новый год, Рождество), вы можете добавить дополнительные ресурсы, а после праздников – вернуть их в пул.

Выделенный (dedicated) сервер – это то же самое, что и физический сервер, но установленный в дата-центр облачного провайдера. Вы покупаете сервер и помещаете его на хранение в дата-центр. Провайдер обеспечивает резервирование Интернет-соединения, питания и охрану сервера. Остальные операции, например, модернизация сервера, его ремонт – осуществляются за дополнительную плату. При этом вы не можете сделать снимок сервера, вы не можете доустановить, а потом – вернуть ресурсы сервера. Следующая таблица позволяет сравнить эти три вида серверов.

Таблица 21.1. Сравнительная таблица типов серверов

Тип сервера	Преимущества	Недостатки
Физический	Настоящий компьютер из плат и проводов Сразу доступны все ресурсы сервера Более высокая производительность	Значительно дороже при покупке и содержании Всю сумму платим сразу Требуется администратор именно сервера Необходимо платить за colocation или обеспечить самому надлежащие условия для работы сервера Сложность модернизации

Виртуальный	Гораздо дешевле физического сервера Простота обслуживания, не нужен отдельный администратор в штате Обеспечение работоспособности - не ваша проблема Быстрое клонирование сервера Быстрое создание «снимка» сервера, что позволяет восстановить сервер за считанные секунды Простая модернизация сервера Оплата только за используемые ресурсы, возможность быстро изменить конфигурацию сервера В стоимость уже входит IP-адрес и Интернет канал	Производительность немного ниже, чем у физического сервера Нельзя увидеть/пощупать физически
Выделенный	Все преимущества физического сервера Услуги colocation уже входят в тариф Не нужно беспокоиться о том, что сервер может сломаться Физический сервер, который вы получаете сразу, а оплачиваете - по мере использования	Дороже виртуального сервера Невозможно создать снапшот, как в случае с виртуальным сервером, нужно использовать традиционные решения резервного копирования

Арендовать виртуальный сервер в большинстве случаев выгоднее, проще и удобнее, чем связываться с физическим сервером. Сегодня, по сути, физи-

ческое оборудование имеет смысл приобретать, если вы сами планируете предоставлять виртуальные серверы в аренду. Во всех остальных случаях можно обойтись виртуальным сервером.

21.3. Выбор облачного провайдера

Очень важно не допустить ошибку при выборе облачного провайдера. Такие ошибки могут очень дорого обойтись – вы потеряете деньги, время, репутацию. Поэтому мы подготовили список, который необходимо уточнить у облачного провайдера перед тем, как воспользоваться их услугами:

- **Уровень сертификации по Tier** – узнайте, присвоен ли центру обмена данных уровень по UTI. С Tier I и Tier II нечего связываться. Tier I – все равно, что разместить серверы у себя в офисе, поскольку данный уровень не предполагает резервирование электропитания. Уровень доступности 99.671%, то есть 28.8 часов простоя в год. Самый надежный и доступный по деньгам – Tier III. Резервируются все инженерные системы, обеспечиваются возможности ремонта и модернизации без остановки сервисов. Tier III (99.98% доступности или 1.6 часа простоя) предполагает постройку второго ЦОД внутри того же здания - ведь все нужно дублировать, в том числе СКС, электричество, систему охлаждения, у всего серверного оборудования должны быть независимые подключения к нескольким источникам питания и т.д. В то же время стоимость будет гораздо ниже, чем у Tier IV (99.99% доступности).
- **Наличие необходимых лицензий ФСТЭК** – помните, что вы будете хранить на данном сервере персональные данные своих клиентов, поэтому наличие лицензий ФСТЭК – необходимое условие.
- **Физическое размещение серверов** – ФЗ-152 требует, чтобы персональные данных граждан РФ хранились только в пределах РФ, поэтому физическое местоположение играет роль. Как бы вам ни хотелось купить сервер в США, ничего не выйдет.
- **Были ли раньше аварии на площадке** - были ли на площадке выбранного вами провайдера аварии и если были, то каковы их причины и какие меры были приняты. Нужные сведения можно легко найти в Интернете, равно, как и отзывы довольных и не очень клиентов.
- **Наличие тестового режима** - самый хороший способ протестировать, подходит ли вам данная площадка или нет. Узнайте у провайдера, есть ли тестовый режим и как ним можно воспользоваться.

- **Чем является виртуальное ядро** - облачные провайдеры измеряют процессорную мощность своих серверов в виртуальных ядрах (vCPU). Одно vCPU может равняться одному физическому ядру процессора, а может быть и четвертью (1/4) ядра. Этот вопрос нужно уточнить заранее. Покупая виртуальную машину с процессором на 4 ядра, получите ли вы 4 ядра или всего одно ядро (если 1vCPU = 0.25 одного ядра)?

- **Базовая скорость Интернет-канала** – трафик, как правило, безлимитный, а вот скорость доступа к Интернету может быть разная. Некоторые провайдеры предоставляют серверы с низкоскоростным Интернет-соединением 10 Мбит/с, а за более скоростной доступ нужно доплачивать. Некоторые сразу предоставляют полноценный канал 100 Мбит/с без каких-либо доплат. Протестировать Интернет-канал можно командой `wget -O - https://raw.github.com/sivel/speedtest-cli/master/speedtest.py | python`. Вы увидите скорость upload и download. В идеале она должна быть примерно одинаковой и при соединении 100 Мбит/с у вас должен быть результат не ниже 76 Мбит/с при прохождении теста.

- **Скорость работы дисковой подсистемы** – многие провайдеры предоставляют серверы с SSD-дисками. Получите тестовый доступ и подключитесь к серверу по ssh. Введите команду `dd if=/dev/zero of=temp bs=1M count=2048`. Вы увидите реальную скорость обмена данными с диском. Если провайдер заявляет, что у вас SSD, а скорость обмена данными меньше 200 Мбайт/с, ищите другого провайдера.

- **Периодичность и стоимость резервного копирования** - уточните, есть ли сервис резервного копирования или резервирование данных придется осуществлять своими силами, то есть покупать еще один виртуальный накопитель, устанавливать и настраивать программное обеспечение для бэкапа и т.д. Если сервис есть, то нужно уточнить, сколько он стоит, чтобы вы могли планировать свои месячные затраты на содержание виртуальной инфраструктуры. Автор этой книги, работая с cloudways.com, был удивлен узнать, что поддержка снапшотов есть не для всех типов серверов. Для серверов DigitOcean возможности создания снапшотов нет. Лучше узнать это до того момента, как вам понадобится возможность создания снапшота.

- **Тарификация** - первое, что нужно уточнить - единицу тарификации - минута, час, день и т.д. Что произойдет, если вы выключите сервер на некоторое время? Будет ли такой простой бесплатным или будет тарифицироваться только хранение информации? Что будет, если вы измените конфигурацию сервера в меньшую сторону? Как и когда это отразится стоимости сервера? Допусти, вы заказали сервер с 16 Гб оперативной па-

мяти, а после некоторого времени вы решили, что 16 - это много и будет достаточно 12 Гб. В 11:00 вы уменьшаете размер оперативной памяти. Когда это отразится на тарификации? Моментально, через час или в начале следующего дня?

- **Скрытые платежи** – узнайте, за что еще вам придется платить. Например, придется ли доплачивать за панель управления сервером, резервное копирование и т.д. Постарайтесь по максимум просчитать, сколько будет стоить содержание виртуальной инфраструктуры в месяц, чтобы в конце месяца это не было для вас неприятным сюрпризом.
- **Способы подключения к серверу** – как можно подключиться к арендованному серверу? Обычно предоставляется SSH-доступ, но может быть еще и доступ через веб-консоль управления сервером, что будет особенно полезным, если вы при настройке брандмауэра случайно закроете сами себе SSH-доступ – такое бывает.
- **Служба поддержки** – узнайте график и условия работы службы поддержки. Какие услуги саппорт оказывает платно, а какие – нет. Например, если нет веб-консоли, а вы заблокируете сами себя и обратитесь в саппорт, будет ли настройка брандмауэра платной или вам помогут бесплатно?

21.4. Выбор конфигурации сервера

Конфигурацию сервера нужно подбирать, исходя из выполняемых ним функций и установленного на сервере ПО. На первых порах вам хватит 4 процессорных ядра и 8 Гб оперативной памяти. Далее будьте готовы к расширению ресурсов. Да, 8 ядер и 32 Гб «оперативки». Если подобная конфигурация сервера – для вас дорого, тогда стоит остановиться прямо сейчас и обратить внимание на SaaS-решения – возможно, они окажутся дешевле.

Будьте готовы, что в реальных условиях вам понадобится как минимум 16 Гб памяти и 4-6 ядер. Соответственно, данные операционные расходы нужно будет закладывать в работу магазина.

21.5. Переезд с хостинга на сервер

Сейчас рассмотрим практический пример. Пусть у вас есть хостинг и на нем есть сайт. Вы хотите перенести сайт на собственный веб-сервер (физический или виртуальный сервер – без разницы).

21.5.1. Этапы переноса

Перенос сайта на VPS состоит со следующих этапов:

1. Копирование файлов на локальный компьютер.
2. Экспорт базы данных.
3. Установка веб-сервера, СУБД и другого ПО на виртуальный сервер (ВС).
4. Настройка ПО на VPS.
5. Загрузка файлов с локальной системы на ВС.
6. Редактирование конфигурации движка (CMS)
7. Импорт базы данных на ВС.
8. Перенос домена.

21.5.2. Копирование файлов сайта на локальный компьютер

Подключитесь к виртуальному хостингу по FTP. Лучше всего для этого использовать FileZilla, поскольку этот FTP-клиент хорошо работает с большим количеством файлов. Перейдите в каталог, содержащий файлы сайта. Как правило, это каталог public_html. Если вы раньше администрировали сайт, то наверняка знаете, как называется этот каталог. Если возникли сложности, обратитесь в саппорт хостинг-провайдера.

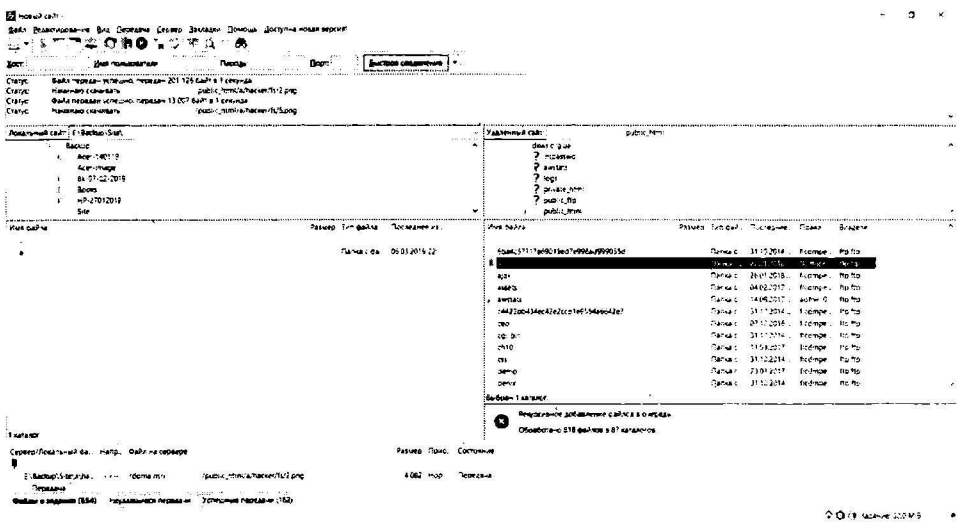


Рис. 21.1. Копирование файлов на локальный компьютер

21.5.3. Экспорт базы данных на локальный компьютер

Войдите в панель управления старого хостинга. В ней часто есть ссылка на phpMyAdmin –это приложение используется для работы с БД.

Далее действия будут такими:

1. Выберите базу данных, которую нужно перенести.
2. Откройте вкладку **Экспорт**.
3. Выберите метод экспорта **Обычный**.
4. Нажмите кнопку **Вперед**.
5. Сохраните дамп.

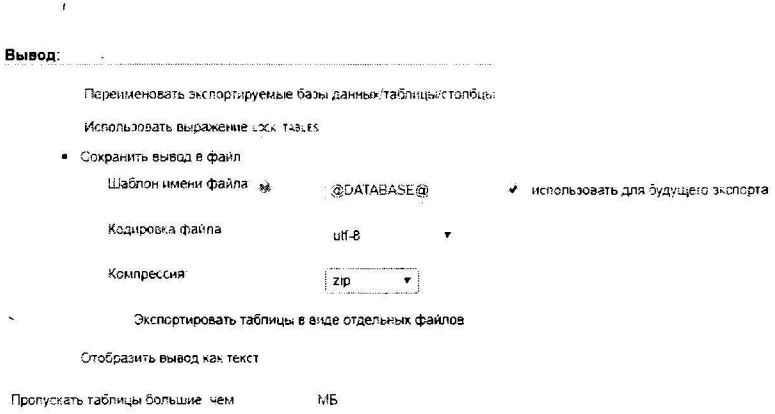


Рис. 21.2. Экспорт базы данных

21.5.4. Установка веб-сервера, СУБД и другого ПО на VPS

Подключитесь к серверу по **ssh**. После этого первым делом обновим список пакетов и сами пакеты:

```
sudo apt update
sudo apt upgrade
```

Далее установим Apache (веб-сервер) и файловый менеджер mc (нужен для упрощения перемещения по файловой системе и редактирования файлов):

```
sudo apt install apache2 mc
```

Перейдите в директорию /etc/apache2/sites-enabled и откройте файл 00-default.conf. Мы не будем создавать отдельные web-серверы для каждого сайта. Будем считать, что у нас есть один сайт и его конфигурация как раз будет храниться в 00-default.conf. Реальный пример конфигурации приведен на следующем рисунке.

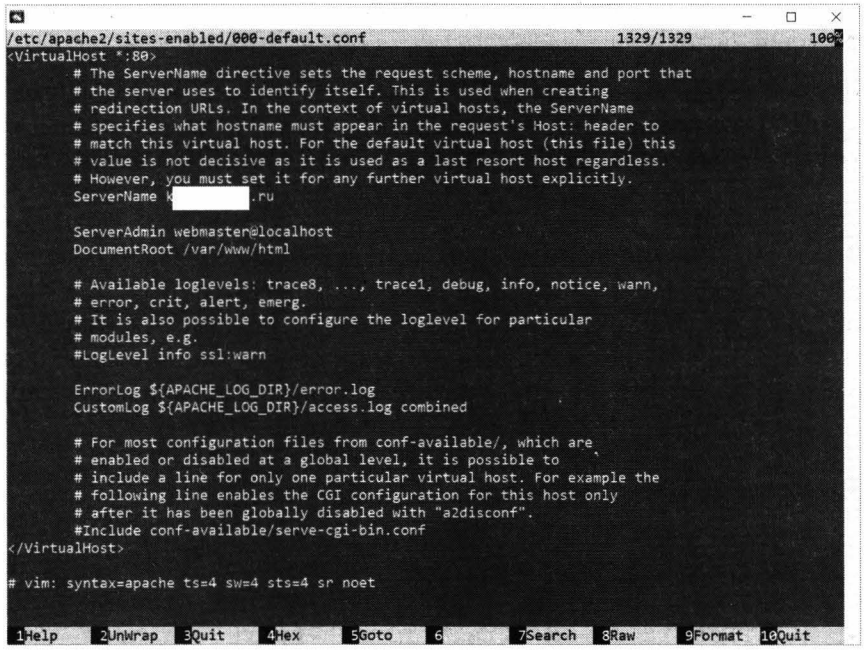


Рис. 21.3. Конфигурация сервера: имя сервера затерто (поскольку производилась настройка реального сервера и его имя скрыто в интересах клиента)

Основные директивы:

- ServerName – здесь нужно указать домен;
- DocumentRoot – указывает, где будут храниться документы сайта, мы используем каталог по умолчанию /var/www/html
- ServerAdmin – можно указать адрес электронной почты администратора сервера.

Запустим web-сервер:

```
sudo systemctl start apache2.service
```

Установим СУБД MySQL, команды:

```
sudo apt install mysql-server mysql-client
sudo mysql_secure_installation
```

Первая команда устанавливает необходимые пакеты – сервер и клиент. Вторая запускает настройку так называемой безопасной инсталляции, в ходе которой будет произведено:

- Установка пароля для MySQL-пользователя root. Обратите внимание, что этот пользователь и системный пользователь root – две разные сущности, поэтому постарайтесь, чтобы и пароли у них были разные.
- Удаление тестовой БД.
- Запрет подключения к серверу баз данных извне, только с локального узла. Это означает, что к СУБД сможет подключиться ПО, работающее только на этом VPS, а не все желающие. Не беспокойтесь: к вашему сайту смогут подключаться все пользователи, просто они не смогут напрямую подключаться к БД, что нежелательно с точки зрения безопасности.

После этого нужно ввести команду:

```
mysql -u root -p
CREATE DATABASE site;
CREATE USER siteuser@localhost IDENTIFIED BY '123456789';
GRANT ALL PRIVILEGES ON site.* TO siteuser@localhost
IDENTIFIED BY 'password';
FLUSH PRIVILEGES;
exit
```

Здесь запускается клиент и от имени MySQL-пользователя root выполняются запросы. Первый запрос создает базу данных site, второй – создает пользователя siteuser, от имени которого CMS будет обращаться к СУБД. Третий запрос – предоставление полномочий пользователю siteuser ко всем таблицам базы данных **site**. Естественно, вместо '123456789' укажите какой-то сложный пароль.

Последний запрос осуществляет применение полномочий, а команда **exit** – выход из клиента mysql.

Для установки PHP и всех необходимых (в большинстве случаев) пакетов используется вот такая длинная команда:

```
sudo apt install php php-cli openssl php-curl php-gd php-
mcrypt php-xml php-intl php-zip php-mbstring php-soap php-
mysql php-json libapache2-mod-php php-xsl composer
```

Она установит последнюю версию интерпретатора, доступную для вашего дистрибутива. Например, для Ubuntu 16.04 – это будет 7.0, для 18.04 – 7.2.

Узнать версию можно командой:

```
php -v
```

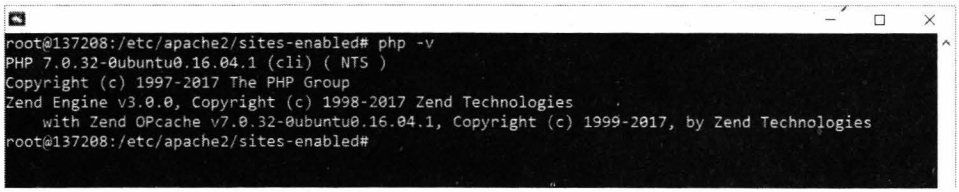


Рис. 21.4. Версия интерпретатора

Установим максимальный размер памяти для сценария. Откройте файл конфигурации (X – версия):

```
sudo mcedit /etc/php/7.X/apache2/php.ini
```

В нем нужно изменить лимит памяти и сразу сохраниться:

```
memory_limit = 512M
```

Добавим необходимые модули Apache (введите команду):

```
a2enmod rewrite
```

Также, чтобы нормально работали SEF URL некоторых CMS нужно открыть /etc/apache2/sites-enabled/000-default.conf и добавить в секцию VirtualHost строки:

```
<Directory /var/www/html/magento_test>
Options Indexes FollowSymLinks MultiViews
AllowOverride All
</Directory>
```

Все, можно повторно перезапускать Apache.

21.5.5. Загрузка файлов с локальной системы на VPS

Загрузите ваши файлы в каталог /var/www/html или любой другой, который вы указали в DocumentRoot. Если для подключения по SSH вы ис-

пользуете Bitvise SSH Client, загрузить файлы можно в окне Bitvise SFTP. Используйте команду контекстного меню **Upload**.

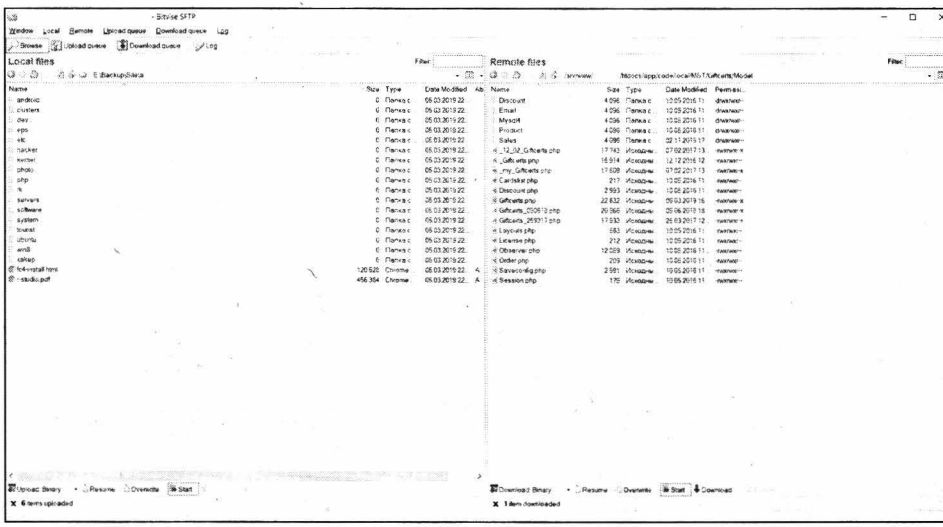


Рис. 21.5. Bitvise SFTP

После этого перейдите в каталог /var/www/html (или любой другой, указанный в конфигурации):

```
sudo find . -exec chown www-data:www-data {} \;
```

Команда делает владельцем всех файлов и каталогов пользователя www-data, от имени которого работает веб-сервер (сейчас владелец root и пользователь www-data не сможет перезаписать файлы, созданные суперпользователем, в результате движок сайта не сможет записать кэш и другую информацию).

21.5.6. Редактирование конфигурации движка сайта

В файловом менеджере **mc** откройте для редактирования (кнопка F4) файл конфигурации движка. Его название и расположения зависит от используемой CMS (например, в случае WordPress – это wp-config.php). В нем нужно «прописать» новые параметры для доступа к БД:

- Узел – localhost
- Имя пользователя – siteuser
- Пароль – 123456789

- База данных – site

Конечно, у вас будут другие, свои значения, которые вы указали при настройке MySQL.

21.5.7. Импорт базы данных на VPS

Здесь все просто. Загрузите на сервер (по SSH) дамп базы данных, полученный при экспорте. Пусть он называется db.sql. Если он сжатый, то сначала его нужно распаковать:

```
unzip db.sql.zip
```

Теперь импортируем его в БД:

```
mysql -u siteuser -p site < db.sql
```

Разберемся что есть что: опция –u задает пользователя БД (siteuser), опция –p говорит о том, что нужно будет спросить пароль этого пользователя, site – это БД, а db.sql – импортируемая БД.

21.5.8. Перенос доменного имени

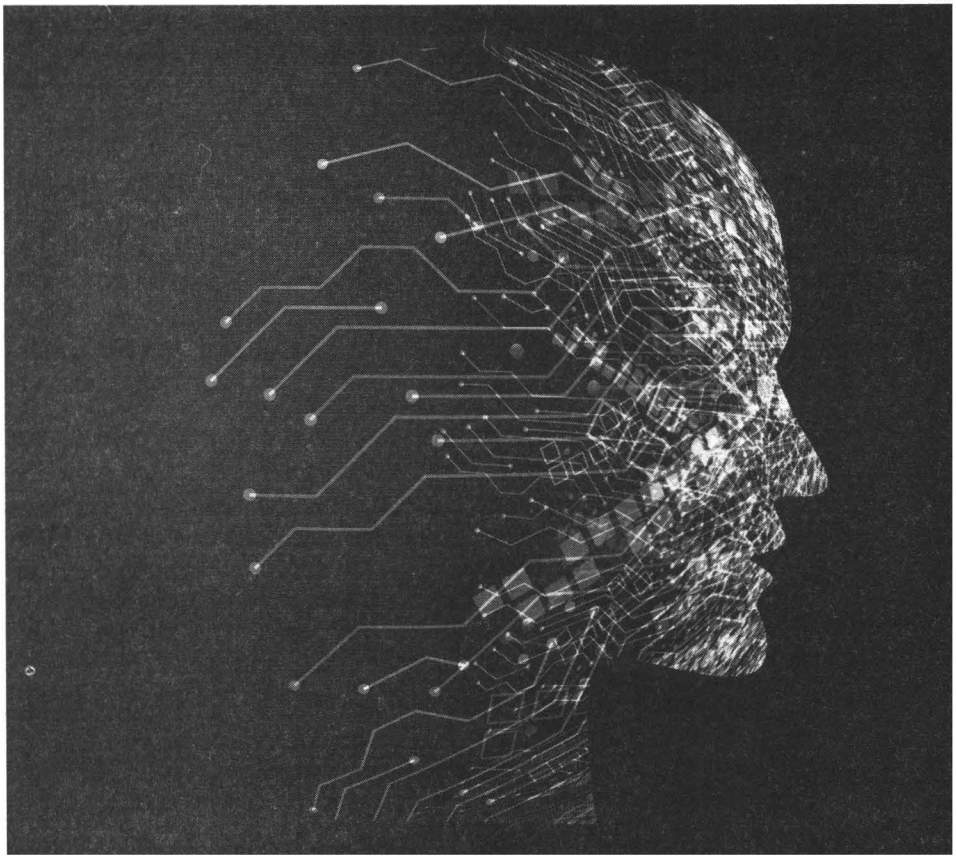
Осталось самое малое – открыть панель управления доменным именем. Если вы покупали домен вместе с хостингом, скорее всего, это будет панель управления хостингом. Если вы покупали домен у регистратора (например, на reg.ru), зайдите в личный кабинет. В настройках домена у вас сейчас прописаны NS-записи – удалите их. Вместо этого создайте A-запись, указывающую на IP-адрес вашего VPS.

> Внимание! Многие после этого совершают одну большую ошибку, а именно – «забывают» пароль от старого хостинга или удаляют аккаунт. Если домен зарегистрирован непосредственно у регистратора доменного имени (nic.ru, reg.ru и т.д.), старый аккаунт можно удалить. А вот если вы покупали домен вместе с хостингом, то управление доменом будет производиться через панель управления старым хостингом. Да, за хостинг платить не нужно, но раз в году необходимо оплачивать продление доменного имени. Или же обратитесь в службу поддержки старого хостинга – они помогут перенести домен в другое место. Лучше всего перенести домен к регистратору доменных имен, а не к новому хостеру – так не придется проделывать эту процедуру еще раз при следующем «переезде».

На этом все. Процедура переезда с виртуального хостинга на сервер завершена.

Глава 22.

Файловый сервер FTP



22.1. Выбор FTP-сервера

Многообразие выбора с одной стороны – хорошо, с другой – порождает проблему выбора. Перед установкой FTP-сервера нужно определиться, какой именно лучше всего подойдет для поставленной вами задачи. А выбрать есть из чего: древний и проверенный временем **wu-ftpd**, «легковесный» **vsftpd**, сверхкомпактный **pure-ftpd** или же универсальный солдат **proftpd**.

Последний отлично подойдет в случае, когда вы еще не знаете, в каком направлении будет развиваться ваш проект. Посредством ProFTPD можно легко реализовать как анонимный FTP-сервер, используемый для загрузки вашего программного обеспечения другими пользователями, так и полноценный FTP-сервер хостинг-провайдера, который сможет с легкостью обслуживать несколько тысяч клиентов и поддерживает базу данных MySQL для хранения учетных данных этих клиентов.

Когда же вы не строите грандиозные планы, вам не нужны тысячи клиентов, а нужен защищенный FTP-сервер, который будет использоваться пользователями для загрузки вашего программного обеспечения – используйте vsftpd (Very Secure FTP Daemon). Не зря он называется Very Secure, что в переводе с английского означает, как очень безопасный. Вы просто устанавливаете и можете не беспокоиться, что кто-то его взломает, потому что у вас не хватило времени на его настройку. Настроек минимум, но и функционала тоже. Для гостевого доступа – идеальное решение. Не зря разработчики дистрибутивов Linux доверяют vsftpd и размещают образы дистрибутивов на FTP-серверах, работающих под управлением vsftpd. Нужно отметить, что vsftpd поддерживает, как гостевые, так и обычные (неанонимные) учетные записи пользователей, но он не позволяет одновременную регистрацию обычных и анонимных пользователей. Другими словами, он у вас будет работать или в режиме обычного FTP-сервера (у каждого пользователя будет свой логин и пароль) или же в режиме анонимного сервера (все будут использовать в качестве логина **anonymous**, в качестве пароля – свой e-mail). Для кого-то – это серьезный недостаток, а кто-то даже не поймет, в чем дело (не всем, кому нужен обычный доступ, использует анонимный доступ и наоборот).

Когда-то стандартом де-факто был wu-ftpd. Старый и проверенный временем сервер. На данный момент он полностью вытеснен ProFTPD и по ряду причин рекомендуется использовать именно ProFTPD.

Для небольших проектов можно использовать pure-ftpd – это простой сервер, который вообще не нужно настраивать, но на производственных (читайте – реальных) серверах его использовать не рекомендуется.

22.2. Универсальный солдат - ProFTPD

22.2.1. Установка и управление сервером

Данный сервер устанавливается, как и любой другой – путем установки соответствующего пакета, который в данном случае называется proftpd. Пакет имеется практически во всех дистрибутивах Linux и называется везде одинаково, что упрощает его установку (не нужно загружать его исходники, выполнять компиляцию, нужно только подставить имя пакета в команду установки). Для установки нужно ввести одну из команд в зависимости от дистрибутива Linux:

```
sudo apt install proftpd              # Ubuntu, Debian
sudo dnf install proftpd              # Fedora, CentOS
```

Конфигурация сервера хранится в каталоге /etc/proftpd. Основной конфигурационный файл называется proftpd.conf и будет рассмотрен в следующем разделе. В некоторых дистрибутивах файл proftpd.conf находится в каталоге /etc, то есть полное его имя - /etc/proftpd.conf

Для управления сервером (для запуска, перезапуска, останова) используются следующие команды:

```
sudo systemctl start proftpd
sudo systemctl restart proftpd
sudo systemctl stop proftpd
sudo systemctl status proftpd
```

В старых дистрибутивах используется команда service вместо systemctl:

```
# service proftpd start
# service proftpd restart
# service proftpd stop
# service proftpd status
```

Назначение команд такое же, как и в случае с другими серверами – запуск, перезапуск, останов сервера и запрос его состояния (status).

> Внимание! По умолчанию сервер настраивается на автоматический запуск, поэтому не перезагружайте операционную систему после установки сервера или, наоборот, не устанавливайте FTP-сервер, если планируете перезапуск ОС. Ненастроенный FTP-сервер может представлять «дыру» в безопасности всей системы. Сначала настраиваем – потом запускаем.

22.2.2. Редактируем конфигурацию сервера

Конфигурационный файл proftpd.conf довольно прост для понимания, но компактным его не назовешь. В нем достаточно много комментариев и опытны пользователи, владеющие английским языком, без особых проблем могут разобраться с ним самостоятельно. Однако, если вы не относите себя к опытным пользователям, либо London is the capital of Great Britain – это все, что вы можете сказать на английском, просмотрите листинг 22.1 и свертесьсверьтесь с вашим файлом конфигурации. В листинге 22.1 представлен перевод файла конфигурации с дополнительными комментариями. В зависимости от версии ProFTPD и вашего дистрибутива, файл конфигурации может немного отличаться от приведенного (разумеется, кроме комментариев – они и так будут отличаться).

Листинг 22.1. Файл /etc/proftpd.conf

```
# Основной файл конфигурации ProFTPD

# название сервера
ServerName      «ProFTPD»
# Тип сервера - автономный. Не изменяйте это значение
ServerType      standalone
# Это сервер по умолчанию
DefaultServer     on

# Используем стандартный порт FTP-сервера - 21.
Port      21

# Диапазон портов брандмауэра для FTP-команды PASV
# (пассивные порты)
PassivePorts    40000 40999
```

```
# Уровень отладки - от 0 до 9
# по умолчанию - 0
DebugLevel      0

# SystemLog - задаем файл журнала
SystemLog    /var/log/proftpd/proftpd.log

# По умолчанию мы не используем IPv6. Если нужна поддержка
# IPv6, установите значение on для этого параметра
UseIPv6      off

# Обычно значение 022 - отличный выбор и не изменяйте его
# за исключением случаев, когда вы понимаете, что делаете
Umask        022

# Для предотвращения DoS-атак установите максимальное число
# дочерних процессов до 30. Если вам нужно более 30
# одновременных
# соединений, просто увеличьте это значение. Этот параметр
# работает только в # standalone-режиме. В inetd-режиме
# количество одновременных соединений ограничивает
# суперсервер xinetd
# Значение по умолчанию - 30, но я его сделал меньше - 20.
# В вашем случае
# нужно экспериментировать и смотреть, сколько
# одновременных пользователей
# могут работать с сервером
MaxInstances     20

# Учетные записи пользователя и группы, от имени которых
будет работать сервер
User      ftp
Group       ftp

# Формат журналирования
LogFormat default  "%h %l %u %t \"%r\" %s %b"
LogFormat auth   "%v [%P] %h %t \"%r\" %s"
LogFormat write    "%h %l %u %t \"%r\" %s %b"

# ------------------------------
# Глобальные параметры описываются в секции Global
# ------------------------------
<Global>

    # ------------------------------
```

```
# Вход на сервер
# ------------------------------

# Поскольку DeferWelcome равно on, то приветствие
# будет отображено после
# аутентификации, а не до нее (off)
ServerIdent    on "FTP server ready"
DeferWelcome     on
# Директива DisplayConnect задает текстовый файл, который
# будет отображен сразу после подключения пользователя,
# но до его входа
#DisplayConnect       /etc/proftpd/msg
# Директива DisplayLogin указывает текстовый файл,
# который будет показан
# когда пользователь зайдет на сервер
#DisplayLogin    /etc/proftpd/msg


<IfModule mod_ident.c>
  # Отключаем Ident-запросы (RFC1413)
  IdentLookups   off
</IfModule>


# Если директива UseFtpUsers включена (on), то ProFTPD
# при подключении
# пользователя будет искать его имя в файле /etc/ftpusers.
# Если его там
# нет, тогда сервер откажет в подключении
   UseFtpUsers   off
# Подключение на основании /etc/shell. При включенной
# директиве
# будет требоваться «правильная» оболочка. Если к серверу
# будут
# подключаться не только Linux/Unix-клиенты, эту директиву
# нужно выключить (off)
RequireValidShell off

# Максимальное число времени в секундах, которые разрешается
# клиенту потратить на аутентификацию
TimeoutLogin   60

# Максимальное число попыток входа
MaxLoginAttempts 3
```

```
  # Максимальное число клиентов на один узел. Позже мы
  # поговорим об этой
  # директиве подробнее
  #MaxClientsPerHost    none
  # Максимальное число соединений для одного пользователя
  #MaxClientsPerUser    1 "Only one connection at a time."

  # ------------------------------
  # Аутентификация
  # ------------------------------

  ### PAM-аутентификация
  # Включите PAM-аутентификацию, если она вам нужна (если
  # хотите
  # контролировать вход систему по FTP через PAM)
  AuthPAM     off

  # измененный AuthPAMConfig-файл
  AuthPAMConfig     proftpd
  ### PAM-аутентификация

  # Задает альтернативный файл паролей. Если нужно, чтобы
# информация об учетных записях бралась из /etc/passwd,
# укажите его здесь, а еще лучше - закомментируйте следующие
# две опции
AuthUserFile   /etc/proftpd/auth/passwd
# Задает альтернативный файл групп
  AuthGroupFile    /etc/group

  ### порядок модулей аутентификации.
# сначала аутентификация будет производиться
# средствами самой ОС, а потом - через файл, заданный
# в AuthFile
  AuthOrder    mod_auth_unix.c mod_auth_file.c
  # AuthOrder    mod_auth_file.c
# Если нужна аутентификация PAM, то порядок должен быть такой:
# AuthOrder mod_auth_pam.c* mod_auth_unix.c

  # ------------------------------
  # После входа пользователя (логина)
  # ------------------------------
  # Задает файл,который будет отображен после входа на сервер
  DisplayLogin   welcome.msg
  # Задает файл, который будет отображаться при изменении
  #  каталога
```

```
DisplayChdir   .message
# Если off, запрещает перезаписывать существующие файлы
AllowOverride     off

# Тайм-аут простоя: если пользователь не проявляет
# активности, соединение будет закрыто
TimeoutIdle       600
# Тайм-аут начала передачи: если пользователь вошел и не
# начал
# передачу за 900 секунд (по умолчанию), соединение будет
# разорвано
TimeoutNoTransfer      900
# Замирание во время передачи. Подробнее этот параметр мы
# обсудим позже
TimeoutStalled    300
# Максимальная продолжительность сессии с момента
# аутентификации
TimeoutSession    3600

# ------------------------------
# Сеанс пользователя
# ------------------------------

# Задает корневую файловую систему пользователя
# Это очень важный параметр и о нем мы поговорим отдельно
DefaultRoot     ~ web,!users

# Регулярное выражение, задающее параметры командной строки,
# которые
# будут блокироваться
DenyFilter              \*.*/
# Позволяет указать, что именно будет выводиться при
# листинге
# каталога. Обычно не нужно изменять этот параметр
ListOptions             «-A +R» strict
# Включает/выключает glob()-функциональность
UseGlobbing             off

# Показывать (on) или нет (off) символические ссылки
ShowSymlinks            on
# Рекомендуется выключить этот параметр (off), чтобы сервер
# показывал локальное время, а не GMT
TimesGMT                off

# ------------------------------
# Загрузка и выгрузка файлов
```

```
  # ------------------------------

  # Можно ли перезаписывать существующие файлы или нет
  AllowOverwrite  off
  # Можно ли клиентам продолжать загрузку (on) или нет (off)
  # Для удобства пользователей рекомендую включить этот
  # параметр
AllowRetrieveRestart     on
  # Для более безопасной загрузки включите (on) этот параметр
  HiddenStores     on
  # Включает автоматическое удаление частично загруженных
  # файлов
  DeleteAbortedStores   on
  #AllowStoreRestart    off

  # ------------------------------
  # Параметры протоколирования. Смело все оставляйте как
  # есть
  # ------------------------------

  WtmpLog   off
  TransferLog    /var/log/proftpd/xferlog

  # Записываем все попытки входа
  ExtendedLog    /var/log/proftpd/auth.log AUTH auth

  # Протоколирование доступа к файлам/каталогам
  ExtendedLog    /var/log/proftpd/access.log WRITE,READ
write

  # Параноидальный уровень протоколирования....
  ExtendedLog    /var/log/proftpd/paranoid.log ALL default

  # SQLLogFile
  #SQLLogFile    /var/log/proftpd/SQL.log
</Global>

### Конец глобальных параметров ###

# Запрещаем использование CHMOD
<Limit SITE_CHMOD>
  DenyAll
</Limit>
```

```
#####
# Включаем другие конфигурационные файлы
#Include       /etc/proftpd/conf.d/*.conf

#####

# ------------------------------
# Настройки анонимного доступа
# ------------------------------
# Базовая анонимная конфигурация, загрузка файлов
# на сервер запрещена
# Анонимным пользователям можно только скачивать файлы с
# сервера
# Если вам не нужен анонимный вход, просто удалите секцию
# <Anonymous>

<Anonymous ~ftp>
   # Limit LOGIN
   #<Limit LOGIN>
   #  Order Allow,Deny
   #  Allow from .examples.net,113.141.114.1
   #  Deny from All
   #</Limit>

   # Ограничиваем WRITE везде, запрещаем запись полностью
   <Limit WRITE>
      DenyAll
   </Limit>

   # LoginPasswordPrompt -- будем ли отображать приветствие
   # или нет
   LoginPasswordPrompt off

   # DirFakeMode -- прячем настоящие разрешения файлов/
   # каталогов
   DirFakeMode 0640

   # DirFakeUser -- прячем настоящих владельцев файлов/
   # каталогов
   DirFakeUser On

   # DirFakeGroup -- скрываем настоящую группу файла/
   # каталога
   DirFakeGroup On
```

```
# Для анонимного входа можно использовать как имя
# anonymous, так и ftp
UserAlias   anonymous ftp

# Максимальное число одновременных анонимных
# пользователей
MaxClients    10
# Максимальный размер получаемого файла
#MaxRetrieveFileSize  512 Mb

# Ограничиваем скорость передачи данных до 255 Кайт/с
#TransferRate APPE,RETR,STOR,STOU 255

# Файл 'welcome.msg' будет отображаться при входе, а файл
# '.message' при
# каждом новом изменении каталога
DisplayLogin     welcome.msg
DisplayChdir     .message

# Далее следует закомментированная секция Directory,
# позволяющая указать
# параметры каталога. В данном случае ограничивается
# доступ к каталогу
# pub. Получить доступ к нему могут только сети
# .examples.net и с IP
# 113.141.114.1
#<Directory pub>
#  <Limit ALL>
#  Order Allow,Deny
#  Allow from .examples.net,113.141.114.1
#  Deny from All
#  </Limit>
#</Directory>

# Следующая секция содержит параметры каталога uploads,
# который обычно
# используется для загрузки файлов анонимными пользователями
# на сервер.
# Если вам нужна такая возможность, раскомментируйте эту
# секцию
# Мы запретили чтение этого каталога, но разрешили
# загрузку в него файлов
#<Directory uploads/*>
```

```
  #  <Limit READ>
     DenyAll
  #  </Limit>
  #  <Limit STOR>
     AllowAll
  #  </Limit>
  #</Directory>
</Anonymous>
```

22.2.3. Обеспечение безопасности FTP-сервера

Ограничение доступа к системным файлам

Поскольку файл конфигурации ProFTPD содержит много самых разных параметров, есть вероятность настроить его неправильно, что приведет к снижению безопасности сервера. В этом разделе мы рассмотрим некоторые параметры, способные ухудшить безопасность сервера. Начнем с директивы DefaultRoot. Для нее нужно задать значение ~:

```
DefaultRoot ~
```

После этого для каждого пользователя его домашний каталог станет его корневым каталогом, то есть пользователь не сможет выйти за пределы его домашнего каталога и прочитать ваши системные конфигурационные файлы.

Любые изменения, произведенные пользователем в его домашнем каталоге, никак не отразятся на системе. Он волен делать все, что ему заблагорассудится – он может загружать файлы в свой домашний каталог, скачивать файлы на свой компьютер, удалять файлы и т.д. Но поскольку доступ к системным файлам закрыт, навредить системе он не сможет. Однако есть исключения: можно навредить системе и без доступа к ее системным файлам.

Ограничение количества регистраций пользователя. Защита от DOS-атаки

По умолчанию с одного и того же IP-адреса могут регистрироваться (логиниться) неограниченное количество пользователей. Никакие ограничения не задаются, поскольку теоретически с одного и того же IP-адреса могут входить несколько пользователей, например, есть какой-то не очень большой провайдер, у него есть только один «белый» IP-адрес, который используют все его пользователи. Мы можем ограничить количество клиентов, ко-

торые могут войти на сервер с одного IP-адреса. В корпоративной сети все просто – один клиент – один IP-адрес, поэтому можем написать так:

```
MaxClientsPerHost       1
```

Если же к FTP-серверу разрешено подключаться пользователям Интернета, то нужно помнить как раз о тех самых небольших провайдерах. Вспомните об университетах, библиотеках и о прочих местах, где предоставляется доступ к Интернету, но, как правило, в таких местах никто не заботится об уникальности IP-адреса и все работают через один IP-адрес шлюза. Получается, что к вашему серверу могут подключиться несколько пользователей из этой сети и у всех них будет одинаковых IP-адрес, хотя сами пользователи будут разные. Обычно такие ситуации - редкие, поэтому можно ограничиться 2-3 клиентами. Но бывают исключения - у вас может быть очень популярный сервер, например, с развлекательным контентом или же вы знаете, что где-то есть другая сеть, практически все пользователи которой будут подключаться к вашему серверу. Здесь решать только вам. Пока установим ограничение на уровне 3 пользователей с одного IP-адреса:

```
MaxClientsPerHost       3
```

Директива **MaxInstances** задает максимальное число одновременных клиентов. Чтобы предотвратить DoS-атаку, не устанавливайте большие значения для этого параметра:

```
MaxInstances    20
```

Опасность этого параметра в том, что более 20 пользователей не смогут работать одновременно. Представим, что вы уже превратились в популярного хостинг-провайдера и у вас уже есть более 500 клиентов. Вероятность, что в бизнес-время более 20 из 500 человек захотят внести изменения в свои сайты, довольно высока. Здесь нужно исходить из поставленных задач. Возможно, придется поднять ограничение до 50 (10% от всего количества пользователей). В любом случае, наверняка у вас будет служба поддержки, в которую будут обращаться пользователи. Если возникнут проблемы, тогда вы всегда сможете поднять этот лимит.

> Внимание! Каждый процесс proftpd занимает около 2.5 Мб, следовательно, 100 процессов займут всего 250 Мб. Как видите, память расходуется экономно. Но нужно помнить о загрузке процессора. При закачке одного файла один процесс proftpd занимает от 10 до 30% процессорного времени одного ядра. Вот и считаем, что если даже один процесс расходует 10% процессорного времени одного ядра, всего 10 процессов «сожрут» одно ядро процессора. 40 одновременных процессов окажут ощутимое влияние даже на четырехядерный

процессор. Вот поэтому в настройках по умолчанию и рекомендует не превышать значение 30 для этого параметра.

Директива MaxClientsPerUser задает, сколько соединений может создать один пользователь. Чтобы один пользователь не залогинился 30 раз и не захватил все 30 процессов, рекомендуется ограничить это значение до **1**:

```
MaxClientsPerUser  1 "Only one connection at a time."
```

Первый параметр этой директивы - число соединений. Второй - сообщение об ошибке, которое будет выведено.

Настройки для медленных соединений

Если у клиента медленное или нестабильное соединение, бывает так, что он может начать передачу файла, но потом связь может оборваться. Можно задать тайм-аут, определяющий, сколько нужно ждать в такой ситуации до разъединения. Медленные и нестабильные соединения уходят в прошлое, поэтому можно понизить время ожидания с 5 минут (300 секунд) до 2 минут:

```
TimeoutStalled    120
```

Это делается специально, чтобы процесс proftpd завершился как можно быстрее и не занимал драгоценное процессорное время.

Максимальная продолжительность сессии с момента аутентификации задается директивой TimeoutSession. По умолчанию сессия составляет 1 час, чего вполне хватит даже для загрузки больших файлов. Например, при относительно низкой скорости загрузки в 1 Мбайт/с файл размером 1 Гб загрузится примерно за 1024 секунды. То есть за одну такую сессию пользователь сможет загрузить три таких файла. Если есть необходимость загрузки больших объемов данных, этот параметр можно увеличить.

Настройки для анонимных пользователей

В некоторых случаях нужно разрешить работу анонимных пользователей. Сервер ProFTPD может одновременно работать, как в обычном, так и в автономном режиме. Приведенная далее конфигурация разрешает загрузку файлов в каталог uploads, который пользователи не могут прочитать - это

делается, чтобы один анонимный пользователь не удалил файлы, загруженные другими пользователями:

```
<Directory uploads/*>
  <Limit READ>
   DenyAll
   </Limit>
   <Limit STOR>
   AllowAll
   </Limit>
</Directory>
```

Доступ на запись разрешен всем (AllowAll), но рекомендуется ограничивать его по IP-адресу, например, разрешить запись только пользователям корпоративной сети:

```
  <Directory uploads/*>
   <Limit READ>
    DenyAll
   </Limit>
   <Limit STOR>
    DenyAll
    Allow from 10.1.1.
   </Limit>
</Directory>
```

В глобальной секции можно ограничить доступ к серверу только пользователям локальной сети. Повторюсь - только, если у вас корпоративный сервер. Это можно сделать путем ограничения операции LOGIN, например:

```
<Limit LOGIN>
  Order deny, allow
  DenyAll
  Allow from 10.10.1.
</Limit>
```

Аутентификация пользователей

Сервер ProFTPD поддерживает аутентификацию как посредством операционной системы Linux (то есть проверкой логина и пароля занимается сама Linux), так и посредством различных плагинов. В следующем разделе будет показано, как реализовать аутентификацию пользователей через таблицу MySQL, что пригодится при большом количестве FTP-пользователей.

В большинстве случаев вам будет удобнее использовать для аутентификации саму операционную систему Linux:

```
AuthOrder   mod_auth_unix.c
```

Если пользователей немного и все они зарегистрированы через /etc/passwd, это неплохой вариант. При желании можно использовать аутентификацию PAM:

```
AuthOrder mod_auth_pam.c* mod_auth_unix.c
```

Если же пользователей очень много (несколько сотен или тысяч), удобнее для аутентификации использовать MySQL. В этом случае учетные записи FTP-пользователей будут храниться не в /etc/passwd, а в отдельной таблице MySQL. Настройка этого способа аутентификации будет рассмотрена в следующем разделе, поскольку все не так просто, как кажется на первый взгляд.

Ускорить аутентификацию может отключение следующих директив:

```
IdentLookups   off
UseReverseDns off
```

Первая строка отключает Ident-запросы (давно уже не используются), вторая - запрещает разрешать IP-адреса пользователей в доменные имена. При входе пользователя на сервер его IP-адрес автоматически преобразовывается в доменное имя. Мы отключили данный функционал, чтобы аутентификация проходила быстрее. При желании вы всегда сможете разрешить IP-адрес в доменное имя вручную, если вам это будет нужно.

Директива TimeoutLogin задает, сколько времени можно потратить пользователю на аутентификацию. По умолчанию - 60 секунд. Сейчас никто не вводит пароль вручную, поэтому нет смысла ждать 60 секунд. FTP-клиент вводит пароль моментально. Зато такая настройка может «подвесить» сервер, например, злоумышленник подключается к серверу под множеством учетных записей и заставляет сервер ждать 60 секунд для каждой из них. Так что меняем это значение на 10. Десять секунд вполне достаточно на предоставление пароля.

```
TimeoutLogin   10
```

Как все будет готово, запустим сервер:

```
sudo systemctl start proftpd
```

После запуска просмотрим его состояние:

```
sudo systemctl status proftpd
```

Далее попробуем подключиться к серверу:

```
$ ftp <IP-адрес сервера>
```

22.2.4. Аутентификация с помощью MySQL

При большом количестве пользователей хранить учетные записи удобнее в таблице MySQL. Во-первых, удобнее управлять таблицей в базе данных, чем редактировать /etc/passwd. С помощью простых SQL-запросов можно легко деактивировать пользователей, которые были зарегистрированы в определенный период или соответствующих другим критериям. При использовании системной аутентификации файл /etc/passwd придется редактировать вручную, да еще и каждую запись отдельно. Во-вторых, FTP-пользователи (которые обычно являются вашими клиентами) не будут смешаны с системными пользователями (с персоналом компании), что есть хорошо – вы всегда сможете понять, кто есть кто.

Для реализации аутентификации через MySQL нужно установить плагин proftpd-mysql, который обеспечит аутентификацию через MySQL. Как правило, это делается путем установки соответствующего пакета (пакет proftpd-mysql).

Далее в область глобальных параметров конфигурационного файла proftpd.conf нужно добавить строки:

```
SQLAuthTypes            Plaintext
SQLAuthenticate         users
SQLConnectInfo          ftpusers@localhost:3306 ftp
password
SQLUserInfo `users` `username` `password` `uid` `gid`
`homedir` `shell`
```

Первая строка определяет тип аутентификации. Мы будем хранить в БД пароли в открытом виде (в незашифрованном виде), поэтому используем тип аутентификации Paintext. Можно, конечно, использовать опцию Backend и шифровать пароли с помощью MySQL-функции PASSWORD().

Директива SQLAuthenticate указывает, кого мы будем аутентифицировать - пользователей.

Директива SQLConnectInfo задает параметры подключения к MySQL-серверу. Здесь:

- **ftpusers** - название базы данных, которая будет содержать информацию о пользователях;
- **localhost** - имя MySQL-сервера;

- **3306** - порт сервера;
- **ftp** - имя MySQL-пользователя (его еще нужно создать!);
- **password** - пароль MySQL-пользователя.

Директива SQLUserInfo задает имя и структуру таблицы с информацией о пользователях. Здесь 'users' - имя таблицы, остальные поля задают, соответственно, имя пользователя, пароль, UID, GID, домашний каталог и оболочку.

Далее с помощью phpMyAdmin или клиента mysql создайте в базе данных ftpusers следующую таблицу:

```
CREATE TABLE `users ` (
  `uuid` int(11) NOT NULL auto_increment,
  `username` varchar(32) NOT NULL,
  `password` varchar(128) NOT NULL,
  `uid` int(11) NOT NULL,
  `gid` int(11) NOT NULL,
  `homedir` varchar(50) NOT NULL,
  `shell` varchar(20) NOT NULL,
  `last_login` int(15) NOT NULL,
  `login_count` int(15) NOT NULL,
  `last_err_login` int(15) NOT NULL,
  `err_login_count` int(15) NOT NULL,
  PRIMARY KEY  (`uuid`)
) ENGINE=MyISAM;
```

Собственно, вот и все. Можно приступить к заполнению этой таблицы. Если вы желаете хранить пароли в зашифрованном виде, используйте тип аутентификации Backend и пароли в БД вносите через MySQL-функцию PASSWORD()[1].

22.3. Очень безопасный vsftpd

Сервер vsftpd не только очень безопасный, но и очень компактный. Функционала меньше, настраивать его придется меньше времени, меньше вероятность допустить ошибку при настройке.

[1] http://dev.mysql.com/doc/refman/5.0/en/encryption-functions.html#function_password

Устанавливается этот сервер посредством установки одноименного пакета, который входит в состав практически всех современных дистрибутивов Linux.

Конфигурационный файл vsftpd называется /etc/vsftpd.conf. В листинге 22.2 приводится вполне рабочая конфигурация сервера vsftpd. Напомним, что vsftpd не поддерживает одновременную регистрацию анонимных и обычных пользователей. Если вам нужно, чтобы на сервере регистрировались, как анонимные, так и обычные пользователи, вам нужно использовать proftpd.

Параметр anonymous_enable позволяет включить/выключить поддержку анонимных пользователей, а параметр local_enable - локальных. Не пытайтесь включить оба параметра – у вас ничего не выйдет. Включите один из них (см. далее).

Как правило, vsftpd используется для загрузки файлов анонимными пользователями, поэтому данное ограничение не является существенным.

Листинг 22.2. Конфигурационный файл /etc/vsftpd.conf

```
# Разрешаем только анонимных пользователей
anonymous_enable=YES

# Запрещаем загрузку файлов анонимными пользователями
anon_upload_enable=NO

# Можно ли анонимному пользователю создавать свои каталоги?
anon_mkdir_write_enable=NO

# Разрешить ли операцию записи (не только сохранение файла, но
и удаление, и
# переименование) анонимному пользователю?
anon_other_write_enable=NO

# Запрещаем регистрацию обычных пользователей
# Не пытайтесь установить оба параметра в YES - такая
конфигурация работать
# не будет
local_enable=NO

# Из соображения безопасности не изменяйте этот параметр!
chroot_local_user=YES
```

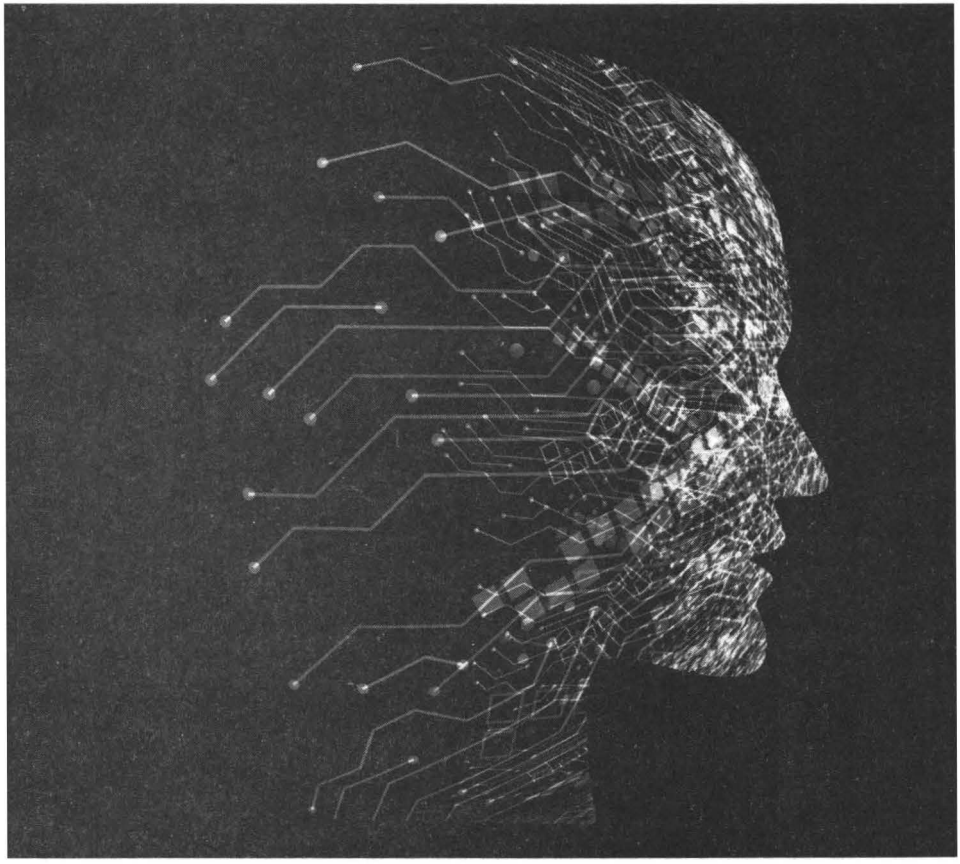
```
# Максимальная скорость передачи данных (в байтах/сек.)
# 0 - без ограничения
local_max_rate=7200

# Разрешена ли запись в каталог?
write_enable=NO

# Выводить ли сообщения при смене директории?
dirmessage_enable=YES

# Строка, которая будет показана при входе пользователя
ftpd_banner="Welcome to FTP service."

# Включить регистрацию событий?
xferlog_enable=YES

# Протоколировать все активные FTP-соединения?
log_ftp_protocol=NO

# Разрешать ли соединения только на порт 20 (ftp data)?
connect_from_port_20=YES

# Таймаут сессии
idle_session_timeout=600

# Таймаут передачи данных
data_connection_timeout=120

# Предоставлять вход через PAM
pam_service_name=vsftpd

# Для автономной работы (как standalone в proftpd) для
следующего
# параметра нужно установить значение YES
listen=YES
```

После того, как вы отредактируете файл конфигурации, сохраните его и запустите сервер:

```
sudo systemctl start vsftpd
```

Глава 23.

Доменная система имен



23.1. Разнообразие DNS-серверов

Все мы знаем, что такое система доменных имен (DNS, Domain Name System). Система DNS отвечает за разрешение числовых IP-адресов в символьные доменные имена. Например, всем известному сайту www.mail.ru соответствует IP-адрес 217.69.139.202 (это один из его IP-адресов). В браузере вы можете ввести или символьное имя (www.mail.ru) или этот IP-адрес. Машинам проще работать с числами, человеку - с символьными именами.

Рассмотрим процесс разрешения доменного имени в IP-адрес. Когда вы вводите адрес узла в браузер, браузер обращается к резолверу - это часть операционной системы, которая отвечает за разрешение доменных имен. Если доменное имя есть в локальном кэше резолвера, он сразу же возвращает его браузеру. Браузер, получив IP-адрес узла, отправляет ему запрос, например, GET / с целью получения корневой страницы (которая обычно называется index.*).

Если в кэше резолвера нет доменного имени, тогда он обращается к DNS-серверу, который указан в настройках сети или же получен от DHCP-сервера во время автоматической настройки сетевого интерфейса. Как правило, это DNS-сервер провайдера. DNS-сервер провайдера проверяет свой кэш. Если в нем будет нужное доменное имя, он отправляет соответствующий ему IP-адрес резолверу. Если же нет, тогда DNS-сервер обращается к DNS-серверу корневого домена .ru. Скорее всего, в его кэше будет нужный IP-адрес, если же нет, тогда будет произведено обращение к DNS-серверу домена mail.ru с

целью получить IP-адрес узла www. Полученный IP-адрес будет по цепочке возвращен узлу, который запросил разрешение доменного имени.

Как видите, схема разрешения доменного имени является рекурсивной, а наш запрос - рекурсивным.

Существуют различные виды DNS-серверов. Даже если у вашей организации нет своего домена или же вы предпочитаете, чтобы делегированием домена занимался ваш регистратор или Интернет-провайдер, вы все равно можете установить кэширующий DNS-сервер. При этом рекурсивными запросами будут заниматься DNS-серверы провайдера, а вашему серверу нужно только кэшировать результаты запросов - так вы ускорите скорость разрешения доменных имен - странички начнут открываться быстрее, а ваши пользователи будут довольны. Зачем нужен кэширующий DNS-сервер, если в какой системе есть кэш резолвера? В кэше резолвера находятся только те имена, к которым обращались вы. В кэше кэширующего DNS-сервера находятся все имена, к которым обращались все пользователи вашей сети. Следовательно, чем больше пользователей в вашей сети, тем больше будет эффективность от кэширующего DNS-сервера.

Итак, вы уже познакомились с одним типом DNS-сервера - кэширующим. Есть и обычный сервер DNS - он занимается тем, что и должен заниматься - хранит информацию о вашей доменной зоне. Также он называется первичным DNS-сервером. На помощь первичному настраивают вторичный DNS-сервер - он будет обрабатывать DNS-запросы, когда первичному серверу стало «плохо». Далее в этой главе будет показано, как настроить вторичный DNS-сервер.

Настройку DNS-сервера мы начнем с самого простого варианта - с кэширующего DNS-сервера.

23.2. Настройка кэширующего DNS-сервера Unbound

Для настройки кэширующего DNS-сервера раньше было принято использовать BIND9 - тот же пакет, который используется для настройки полноценно DNS-сервера. Но сейчас вместо него принято использовать пакет unbound.

Сервер Unbound распространяется под лицензией BSD, обладает модульной структурой и может работать, как в рекурсивном, так и кэширующем

режиме. Мы же будем использовать Unbound сугубо в кэширующем режиме.

Основной файл конфигурации называется /etc/unbound/unbound.conf. По умолчанию он практически пуст, а полный пример со всеми возможными опциями можно найти в каталоге /usr/share/doc/unbound/examples.

В листинге 23.1 представлен листинг /etc/unbound/unbound.conf, настраивающий Unbound на работу в кэширующем режиме.

Листинг 23.1. Файл /etc/unbound/unbound.conf

```
server:
# Порт, на котором наш сервер будет «слушать» запросы
port: 53
# Описываем интерфейсы, на которых мы будем слушать запросы
# 192.168.1.1 - сервер нашей локальной сети, на котором
установлен Unbound
interface: 127.0.0.1
interface: 192.168.1.1
# Исходящий интерфейс (WAN)
outgoing-interface: xxx.xx.xx.xx
#  Сеть, которой разрешен доступ к нашему серверу
access-control: 192.168.1.0/24 allow
# Разрешаем IPv4 TCP/UDP, запрещаем IPv6
do-ip4: yes
do-ip6: no
do-udp: yes
do-tcp: yes
# Пользователь, от имени которого будет запускаться сервер
username: unbound
# Указываем файл журнала и отключаем использование syslog
logfile: "unbound.log"
use-syslog: no
# Путь к PID-файлу
pidfile: "/var/run/local_unbound.pid"
# Скрываем версию софта
hide-version: yes
# Уровень журналирования - 0 (только ошибки)
verbosity: 0
# Следующая строка настраивает Unbound на осуществление
криптографической
# валидации DNSSEC, используя корневой ключ
```

```
auto-trust-anchor-file: "/var/lib/unbound/root.key"
```

Теперь проверим конфигурацию сервера:

```
# unbound-checkconf
```

Если ошибок нет, вы получите сообщение:

```
unbound-checkconf: no errors in /etc/unbound/unbound.conf
```

Осталось только перезапустить Unbound:

```
# service unbound restart
```

Осталось только настроить DHCP-сервер, чтобы он сообщал всем локальным узлам ваш новый IP-адрес DNS-сервера. В нашем случае - это 192.168.1.1.

23.3. Настройка кэширующего сервера на базе bind

Ради контраста сейчас мы попытаемся настроить кэширующий сервер на базе пакета bind (в некоторых дистрибутивах - bind9). Первым делом установим сам bind:

```
# apt-get install bind9
```

> Примечание. BIND9 - это уникальный сервер. Его пакет называется bind9, каталог с конфигурационными файлами - /etc/bind, а название конфигурационного файла - named.conf. Процесс (исполнимый файл) называется named. К такому многообразию имен придется привыкнуть. Главное понимать, что это одно и то же.

Раньше основной конфигурационный файл /etc/bind/named.conf был довольно большим, если не огромным. Сейчас в нем только три строчки (лист. 23.2).

Листинг 23.2. Файл /etc/bind/named.conf по умолчанию

```
include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.local";
include "/etc/bind/named.conf.default-zones";
```

Общие параметры теперь вынесены в файл /etc/bind/named.conf.options. Локальные зоны описываются в named.conf.local, а зоны по умолчанию - в named.conf.default-zones. Нас в данный момент интересует файл named.conf.default-zones, в котором описаны зоны по умолчанию. Проследите, чтобы в этом файле были описаны зоны, представленные в листинге 23.3.

Листинг 23.3. Файл /etc/bind/named.conf.default-zones

```
// 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";
};
```

Как видите, в этом конфигурационном файле описывается зона корневых серверов и локальная зона localhost, которая отвечает за преобразование имени localhost в IP-адрес 127.0.0.1 и обратно.

Теперь рассмотрим файл /etc/bind/named.conf.options (лист. 23.4).

Листинг 23.4. Файл /etc/bind/named.conf.options

```
options {
   directory "/var/cache/bind";

   // Здесь описываются forward-серверы

   // forwarders {
   //   0.0.0.0;
   // };

   dnssec-validation auto;

   auth-nxdomain no;    # conform to RFC1035
   listen-on-v6 { any; };
};
```

Все, что нужно - это добавить IP-адрес DNS-сервера провайдера в блок forwarders. Именно они будут выполнять всю грязную работу по разрешению доменных имен, а наш сервер будет только кэшировать результаты запроса.

Перед блоком forwarders можно указать параметр **forward**, который может принимать значение **only** или **first**. В первом случае наш сервер вообще не будет предпринимать попыток обработать запрос самостоятельно. Во втором случае сервер предпримет попытку обработать запрос самостоятельно, если не получит ответ от серверов, описанных в блоке forwarders. Второе значение более предпочтительно:

```
forward first;
forwarders {
  8.8.8.8;
  8.8.8.4;
};
```

Вот и все. Перезапустите сервер:

```
# service bind9 restart
```

Просмотрите файл журнала:

```
# tail /var/log/daemon.log
```

Вы должны увидеть сообщение, что сервер запущен вроде этого:

```
Jun 29 09:41:42 debian named[6921]: running
```

Проверим, работает ли наш DNS-сервер. В /etc/resolv.conf на DNS-сервере добавьте строку:

```
nameserver 127.0.0.1
```

Если вы по каким-то причинам не отключили NetworkManager, то он при следующей перезагрузке перезапишет этот файл. Понятно, что NetworkManager получит IP-адрес DNS-сервера от DHCP-сервера, но пока вы еще не настроили DHCP-сервер, тогда можете запретить изменение файла /etc/resolv.conf:

```
# chattr +i /etc/resolv.conf
```

После этого перезапустите сеть или компьютер. После этого введите команду:

```
nslookup mail.ru
```

Вывод будет таким:

```
Server:127.0.0.1
Address:  127.0.0.1#53

Non-authoritative answer:
Name:     mail.ru
Address:  217.69.139.202
Name:     mail.ru
Address:  217.69.139.200
Name:     mail.ru
Address:  94.100.180.201
Name:     mail.ru
Address:  94.100.180.200
```

Как видите, ответ пришел от нашего сервера 127.0.0.1. Мы убедились, что наш DNS-сервер работает, значит, можно настроить DHCP-сервер, чтобы он «раздавал» всем вашим клиентам IP-адрес только что настроенного DNS-сервера. Конечно, прописывать в настройках DHCP-сервера нужно не IP-адрес 127.0.0.1, а IP-адрес сервера, который вы можете получить командой ifconfig, запущенной на DNS-сервере.

23.4. Настройка полноценного DNS-сервера

Настройка полноценного DNS-сервера сложнее настройки кэширующего DNS-сервера. Ведь нам придется настроить одну или несколько зон, которые будет обслуживать наш DNS-сервер. Домен - это не зона. Зона - это часть домена, которая управляется определенным DNS-сервером. В случае с небольшими доменами, то зона = домен. Но иногда бывает так, что часть поддоменов одного домена обслуживается одним DNS-сервером, а другая часть - другим DNS-сервером. Так вот эта часть, которая обслуживается определенным DNS-сервером, и есть зона.

Представим, что у нас есть домен example.com, который представляет сеть 192.168.1.0. В конфигурацию BIND нужно добавить следующие строки:

```
zone "example.com"  {
   type master;
   file "example.com";
   notify no;
};

zone "0.1.168.192.in-addr.arpa" {
   type master;
   file "192.168.1.0";
   notify yes;
}
```

Данные строки нужно добавить или в файл /etc/bind/named.conf.local или прямо в файл /etc/bind/named.conf после тех строк, которые в нем уже есть.

Файл example.com, описанный в первом блоке **zone**, содержит конфигурацию прямого преобразования, то есть используется для преобразования доменных имен в IP-адреса. Содержимое этого файла приведено в лист. 23.5.

Листинг 23.5. Файл /etc/bind/example.com

```
@  IN SOA  ns.example.com.      admin.example.com. (
      1    ; серийный номер
      60480     ; обновление каждые 60480 секунд
      86400     ; повтор каждые 86400 секунд
```

```
     2419200; время хранения информации - 672 часа
     86400    ; TTL записи
)
     IN NS    ns.example.com.
     IN A   192.168.1.1
     IN MX    100  mail.example.com.
www    IN CNAME  ns.example.com.
mail   IN A   192.168.1.3
ftp    IN A   192.168.1.2
localhost.  IN A 127.0.0.1
```

Теперь разберемся, что есть что. Первым делом запомните «правило точки». Если в конце доменного имени ставится точка (посмотрите - ns.example.com.), то сервер не будет дописывать имя домена example.com к имени. Если же имя указано без точки, к нему будем дописано имя домена example.com.

Запись SOA описывает начало полномочий. Первое имя после SOA - это имя данного компьютера - ns.example.com. Затем указывается электронный адрес администратора сервера. Символ @ зарезервирован, поэтому первая точка считается @. Выходит, адрес администратора - admin@example.com. Оставшаяся часть записи SOA прокомментирована в листинге.

Запись NS задает имя DNS-сервера имен, в нашем случае это ns.example.com, запись A задает его IP-адрес.

Запись MX задает адрес и приоритет (100) почтового сервера. У вас может быть несколько почтовых серверов, тогда вы можете указать несколько записей MX с разным приоритетом. Чем ниже значение приоритета, тем выше приоритет сервера, например:

```
IN MX 100 mail1.example.com.
IN MX 200 mail2.example.com.
```

Далее записи CNAME создают псевдоним для имени www. Это означает, что веб-сервер тоже запущен на этом компьютере и когда кто-то укажет имя www.example.com запрос придет к узлу с IP-адресом 192.168.1.1.

Запись A используется для преобразования доменного имени в соответствующий ему IP-адрес. Как видите, мы задаем IP-адреса для компьютеров с именами mail, ftp и localhost.

Неужели в сети нет больше компьютеров? Остальные компьютеры, как правило, являются обычными рабочими станциями и назначением IP-адресов им занимается DHCP-сервер. Обычно не нужно разрешать имена этих ком-

пьютеров в IP-адреса, так как никаких соединений с ними устанавливать не планируется.

Теперь рассмотрим файл обратного преобразования - /etc/bind9/192.168.1.0 (лист. 23.6).

Листинг 23.6. Файл /etc/bind9/192.168.1.0

```
@  IN SOA  ns.example.com. admin.example.com. (
      1     ; серийный номер
      60480      ; обновление каждые 60480 секунд
      86400      ; повтор каждые 86400 секунд
      2419200 ; время хранения информации - 672 часа
      86400      ; TTL записи
)
@  IN NS ns.example.com
1  IN PTR  ns.example.com
2  IN PTR  ftp.example.com
3  IN PTR  mail.example.com
$GENERATE 5-104 $ PTR   ip-192-168-1-$.example.com
```

Из листинга 23.6 видно, что IP-адрес 192.168.1.1 принадлежит узлу ns.example.com, 192.168.1.2 - узлу ftp.example.com и адрес 192.168.1.3 - узлу mail.example.com. Последняя запись не обязательна. Она говорит, что узлам с IP-адресами от 192.168.1.5 до 192.168.1.104 будут соответствовать имена ip-192-168-1-N-example.com, где N - последнее число IP-адреса. Данная запись нужна только, если вы заботитесь о преобразовании IP-адресов, выданных вашим DHCP-сервером, в доменные имена.

В этом файле вы можете не указывать IP-адреса полностью, но если вы это делаете, то их нужно указывать в обратном порядке, например:

```
1.1.168.192  IN     PTR    ns.example.com
```

Точки в конце доменного имени также не нужны. Вот собственно и все. Почти все. Мы еще не позаботились о защите вашего сервера. Для настройки удаленного управления сервером нужно подготовить блоки key и controls. Проще всего это сделать с помощью команды:

```
# /usr/sbin/rndc-confgen > remote.conf
```

Далее скопируйте содержимое файла remote.conf в самое начало файла named.conf или в самое начало файла named.conf.options - в зависимости от ваших предпочтений (где вы предпочитаете хранить конфигурацию).

Комментарии из remote.conf, которые также будут сгенерированы утилитой, можно не копировать. Вот что нужно скопировать (ключ у вас будет другим):

```
key "rndc-key" {
      algorithm hmac-md5;
      secret «sJZkjPskmF9KPkQBwaUtfQ==»;
};
controls {
default-key "rndc-key";
default-server 127.0.0.1;
default-port 953;
};
```

Также в файл named.conf.options нужно добавить блок allow-query (внутри блока **options**):

```
allow-query {
192.168.1.0/24;
localhost;
}
```

Здесь мы разрешаем обращаться к нашему DNS-серверу только пользователям внутренней локальной сети и узлу localhost.

Также можно обновить файл корневых серверов (это рекомендуется делать периодически):

```
# wget ftp://ftp.internic.net/domain/named.root
# cp named.root /etc/bind/db.root
```

Вот теперь действительно все и можно перезапустить сервер:

```
# service bind9 restart
```

23.5. Настройка вторичного DNS-сервера

В крупных сетях или там, где важна отказоустойчивость, например, в сетях провайдера, важно настроить вторичный DNS-сервер, который будет обслуживать запросы клиентов в случае отказа первичного сервера.

Вторичный сервер настраивается, как и первичный, вот только тип зоны задается как подчиненная (slave), а в блоке masters указываются первичные DNS-серверы (в нашем случае только один):

```
zone "example.com"  {
   type slave;
   file "example.com";
   masters { 192.168.1.1; };
};
```

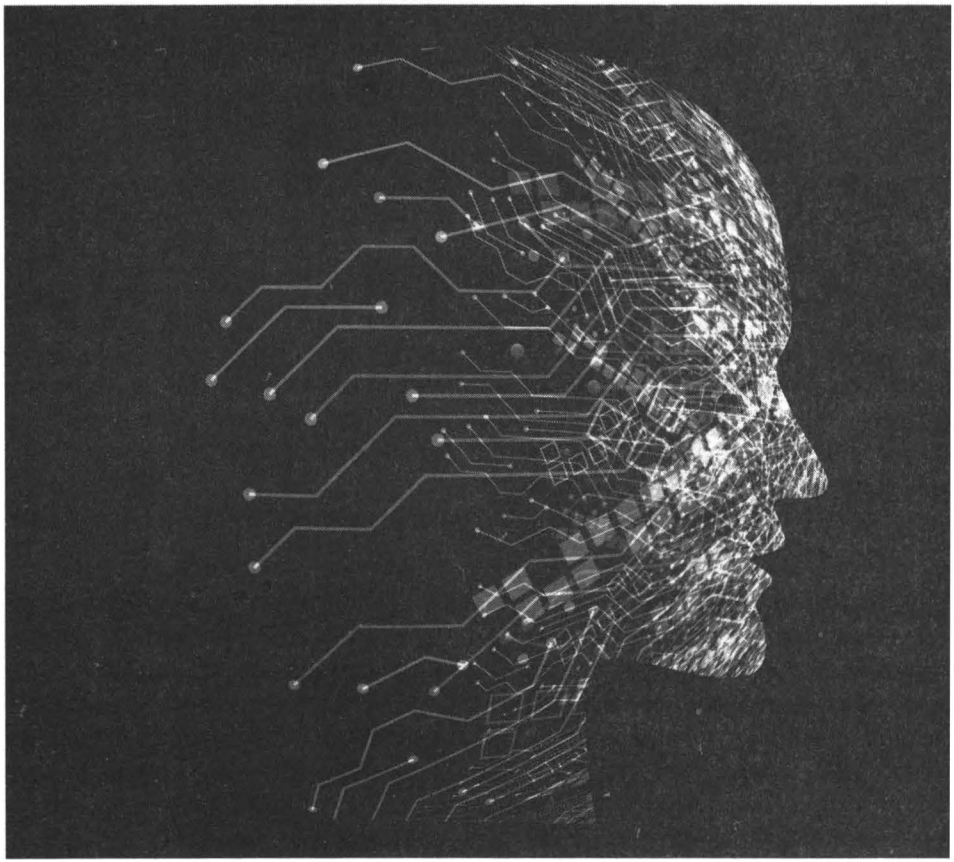
На первичном сервере в блоке options нужно добавить блок allow-transfer, в котором указывают IP-адрес вторичного DNS-сервера:

```
options {
...
allow-transfer { 192.168.1.2; };
}
```

Вот теперь действительно все.

Глава 24.

DHCP-сервер



24.1. Настраивать DHCP-сервер или нет?

В небольшой сети, как правило, в DHCP-сервере необходимости особой нет, поскольку имеется сетевое устройство вроде Wi-Fi-маршрутизатора, «на борту» которого работает небольшой DHCP-сервер, возможностей которого вполне достаточно для обслуживания такой сети.

Однако в сетях среднего и большого размера, как правило, используются DHCP-серверы. Конфигурация таких сетей более сложна и обычно содержит подсети: то есть нужно назначать IP-адреса не только для основной сети, но и для нескольких подсетей. Встроенные DHCP-серверы беспроводных маршрутизаторов (да и такие простые устройства в крупных сетях редко используются) не обладают необходимым функционалом, поэтому без полноценного DHCP-сервера будет сложно.

24.2. Принцип работы протокола DHCP

Протокол DHCP (Dynamic Host Configuration Protocol) используется для автоматической настройки узлов сети. DHCP-сервер, работающий в сети, автоматически настраивает все остальные компьютеры и сетевые устройства (например, сетевые принтеры, хранилища данных и т.д.). Компьютеру передается следующая информация: IP-адрес, маска сети, IP-адрес шлюза, IP-адреса DNS-серверов

При необходимости DHCP-сервер может предоставлять и другую информацию (например, адреса WINS-серверов, если это необходимо) и выполнять некоторые защитные функции, например, сервер может не предоставить IP-адрес устройству, если его MAC-адрес не находится в списке разрешенных, другими словами, DHCP-сервер может назначать IP-адреса только тем устройствам, чьи MAC-адреса «прописаны» в его конфигурационном файла. Такая защита легко обходится даже опытными пользователями, не говоря уже о профессионалах, но все же лучше, чем ничего.

DHCP-сервер можно настроить так, чтобы он предоставлял компьютеру с определенным MAC-адресом один и тот же IP-адрес. Эту функцию очень

удобно использовать для серверов, у которых должны быть постоянные IP-адреса.

Лет 20 назад DHCP-серверы использовались только в крупных сетях, где были сотни узлов. В небольших сетях на несколько десятков узлов IP-адреса назначались вручную. Но позже все осознали преимущества DHCP и теперь уже сложно найти сеть, в которой не использовался бы DHCP. Даже в небольших домашних сетях есть DHCP-сервер, который, как правило, запущен на маршрутизаторе, предоставляющем доступ к Интернету. Однако функционал таких маршрутизаторах обычно оставляет желать лучшего, поэтому в сетях среднего размера, как уже отмечалось, рекомендуется настраивать отдельный DHCP-сервер, работающий на стационарном компьютере под управлением Windows Server или Linux.

Далее в этой главе приводится описание настройки DHCP-сервера, установить который можно посредством установки пакета dhcp (CentOS), dhcp-server или isc-dhcp-server (последние версии Ubuntu). Название пакета может отличаться в зависимости от дистрибутива и его версии.

24.3. Редактирование конфигурации DHCP

Конфигурация DHCP хранится в двух конфигурационных файлах – dhcpd.conf и dhcpd6.conf. Оба файла находятся в каталоге /etc. В некоторых дистрибутивах эти конфигурационные файлы следует поискать в каталогах /etc/dhcp или /etc/dhcpd.

Как вы уже догадались, первый файл используется для протокола IPv4, второй – для IPv6. Обычно второй не используется, а конфигурация хранится в первом файле.

Директивы файла конфигурации не чувствительны к регистру символов, поэтому вы можете написать директиву, как ddns-update-style ad-hoc или как DDNS-UPDATE-STYLE AD-HOC. Разницы нет. Однако использовать верхний регистр не принято.

Комментарии в файле dhcpd.conf начинаются с решетки, например:

```
# Комментарий
```

Сервер DHCP может предоставлять IP-адреса нескольким подсетям. Каждая из подсетей описывается в виде блочной директивы **subnet**. Параметры конфигурации, описанные за пределами директивы subnet, применяются

ко всем подсетям - это глобальные параметры. А вот параметра конфигурации, описанные внутри директивы subnet, применяются только к конкретной подсети.

Обычно в самом начале файла конфигурации задаются следующие директивы:

```
option domain-name "company.com"
option domain-name-servers ns1.company.com ns2.company.com
```

Если ваш DHCP-сервер обслуживает несколько подсетей и у каждой из них есть собственное доменное имя, тогда опции domain-name и domain-name-servers задаются внутри блочной директивы subnet.

Директивы default-lease-time и max-lease-time задают время аренды IP-адреса по умолчанию и максимальное время аренды. IP-адрес выделяется DHCP-сервером не навсегда, а только на определенное время. По истечению данного времени IP-адрес возвращается в пул адресов, а компьютеру назначается другой IP-адрес из пула свободных адресов. Вполне вероятно, что компьютеру опять будет назначен тот же адрес, например, если в настройках DHCP-сервера указано, что компьютеру с определенным MAC-адресом должен быть назначен определенный IP-адрес.

> Примечание. База данных аренды IP-адресов находится в файле /var/lib/dhcp/dhcpd.leases

Пример установки этих директив (значения указываются в секундах):

```
default-lease-time 28800;      # 8 часов
max-lease-time 86400;          # 24 часа
```

Очень важной является директива ddns-update-style, задающая стиль обновления DNS: непосредственное (ad-hoc) или предварительное взаимодействие DHCP-DNS (interim). Разработчики протокола DHCP рекомендуют использовать второй стиль:

```
ddns-update-style interim;
```

Теперь рассмотрим блочную директиву **subnet**, которая описывает параметры определенной подсети. В примере ниже описываются параметры для подсети 192.168.0.0 с маской сети 255.255.255.0 (сеть класса C):

```
subnet 192.168.0.0 netmask 255.255.255.0 {
    # Список маршрутизаторов (через пробел)
        option routers                  192.168.0.1;
  # Маска подсети
        option subnet-mask              255.255.255.0;
    # Широковещательный адрес
        option broadcast-address        192.168.0.255;
    # Доменное имя
        option domain-name              "company.com
    # IP-адрес/имя DNS-сервера, если не задано в глобальных
    # параметрах
        option domain-name-servers      192.168.0.1;
    # Сервер времени (NTP)
        option ntp-servers              192.168.0.1;
    # IP-адрес сервера NetBIOS и его тип узла (если нужно)
        option netbios-name-servers     192.168.0.1;
        option netbios-node-type        8;
    # Диапазон IP-адресов, выделяемый клиентам сети
        range   192.168.0.101   192.168.0.200;
}
```

На данный момент в директиве **subnet** представлены практически все опции. Но на практике набор опций у вас будет другим. Серверы NetBIOS используется не всеми, как и серверы времени (NTP). Параметры DNS (имя и IP-адреса DNS-серверов) обычно выносят в область глобальных параметров, поскольку они в большинстве случаев одинаковы для всех подсетей.

Если из нашей директивы subnet удалить все ненужное, она будет выглядеть так:

```
subnet 192.168.0.0 netmask 255.255.255.0 {
        option routers                  192.168.0.1;
        option subnet-mask              255.255.255.0;
        range   192.168.0.101   192.168.0.200;
}
```

Мы оставили только список маршрутизаторов, маску подсети и диапазон IP-адресов, из которого будут выделяться IP-адреса.

В листинге 24.1 приводится пример файла dhcpd.conf для небольшой сети с несложной топологией.

Листинг 24.1. Пример файла dhcpd.conf для простой сети

```
ddns-update-style interim;

# Расположение базы данных с арендой IP-адресов
lease-file-name "/var/lib/dhcpd/dhcpd.leases";

# Данный сервер является официальным DHCP-сервером для
локальной сети
authoritative;

# Доменное имя и имена DNS-серверов
option domain-name                  "company.com";
option domain-name-servers           ns1.company.com ns2.
company.com

# Время аренды
default-lease-time                   86400;   # 24 часа
max-lease-time                       172800;  # 48 часов

subnet 192.168.0.0 netmask 255.255.255.0 {
        option routers          192.168.0.1;
        option subnet-mask      255.255.255.0;
        range           192.168.0.101   192.168.0.200;
}
```

Мы удалили из dhcpd.conf все лишнее и наш файл получился довольно компактным (сравните его с файлом по умолчанию, где в качестве примера приводятся чуть ли не все мыслимые и немыслимые параметры).

24.4. DHCP-сервер в больших сетях

Далее мы рассмотрим пример более сложной сети с несколькими подсетями – ради чего, собственно, и есть смысл настраивать DHCP-сервер вручную, а не использовать встроенный в маршрутизатор DHCP. Общая сеть должна быть описана в директиве shared-network, а все подсети должны быть описаны директивами **subnet** внутри директивы shared-network. Пример приведен в листинге 24.2.

Листинг 24.2. Пример файла dhcpd.conf для сложной сети

```
shared-network my_bignet {

# Доменное имя и имена DNS-серверов
option domain-name                  "company.com";
option domain-name-servers            ns1.company.com ns2.
company.com
# Шлюз по умолчанию
     option routers                   192.168.0.1;

# Подсети 192.168.1.0 и 192.168.2.0

     subnet 192.168.0.0 netmask 255.255.252.0 {
         range 192.168.0.101 192.168.0.200;
     }
     subnet 192.168.1.0 netmask 255.255.252.0 {
         range 192.168.1.101 192.168.1.200;
     }
}
```

Если для подсетей 192.168.0.0 и 192.168.1.0 нужно указать различные параметры, например, разные шлюзы или разные имена DNS-серверов, то соответствующие параметры нужно указать в директиве subnet для определенной подсети.

24.5. Статические IP-адреса. Директива host

Иногда нужно привязать некоторые IP-адреса к MAC-адресам. Это полезно, если в вашей сети есть несколько компьютеров, IP-адреса которых не должны изменяться (как правило, это серверы сети и некоторые специальные компьютеры вроде компьютера администратора). Статические IP-адреса описываются с помощью директивы **host**:

```
host server {
  option host-name "server";
  option routers 192.168.1.1;
  hardware ethernet 00:FF:FB:69:DC:E5;
  fixed-address 192.168.1.99;
}
```

В данном случае если к сети подключится компьютер с MAC-адресом 00:FF:FB:69:DC:E5, ему будет назначен IP-адрес 192.168.1.99, имя узла server и шлюз по умолчанию 192.168.1.1.

Директиву **host** нужно поместить в одну из директив subnet, к которой принадлежит выделяемый IP-адрес, например:

```
subnet 192.168.1.0 netmask 255.255.255.0 {
        option routers          192.168.1.1;
        option subnet-mask      255.255.255.0;
        range           192.168.1.101   192.168.1.200;

host server {
     option host-name "server";
     option routers 192.168.1.1;
     hardware ethernet 00:FF:FB:69:DC:E5;
     fixed-address 192.168.1.99;
}

}
```

Управлять DHCP-сервером можно с помощью команды **service**. Следующие команды позволяют запустить, перезапустить или остановить сервер:

```
$ sudo systemctl start isc-dhcp-server
$ sudo systemctl enable isc-dhcp-server
$ sudo systemctl enable isc-dhcp-server
```

или (в старых дистрибутивах):

```
# service dhcpd start
# service dhcpd restart
# service dhcpd stop
```

Обратите внимание, как называется сервис DHCP-сервиса. В современных версиях Ubuntu он называется isc-dhcp-server. В старых версиях Ubuntu и других дистрибутивов нужный сервис назывался dhcpd.

24.6. Настройка DHCP-клиента в Ubuntu

В Ubuntu 16.04 вы можете настроить интерфейс в файле конфигурации /etc/network/interfaces.

```
$ sudo nano /etc/network/interfaces
```

Добавьте эти строчки:

```
auto  eth0
iface eth0 inet dhcp
```

Сохраните файл и перезапустите сетевой сервис (или перезагрузите систему).

```
$ sudo systemctl restart networking
```

В Ubuntu 18.04 и более новых сетевое управление контролируется программой Netplan. Вам нужно отредактировать соответствующий файл, например, в каталоге /etc/netplan/

```
$ sudo vim /etc/netplan/01-netcfg.yaml
```

Затем включите dhcp4 под конкретным интерфейсом, например, под ethernet, ens0, и закомментируйте статические настройки, связанные с IP:

```
network:
  version: 2
  renderer: networkd
  ethernets:
    ens0:
      dhcp4: yes
```

Сохраните изменения и выполните следующую команду, чтобы применить изменения:

```
$ sudo netplan apply
```

Для получения дополнительной информации смотрите справочные страницы dhcpd и dhcpd.conf.

```
$ man dhcpd
$ man dhcpd.conf
```

В Windows все гораздо проще - нужно включить автоматическое назначение IP-адреса, как показано на рис. 24.1. Как правило, по умолчанию все и так уже настроено и никаких дополнительных действий предпринимать не нужно.

На этом все. Мы рассмотрели процесс настройки сервера и клиента DHCP.

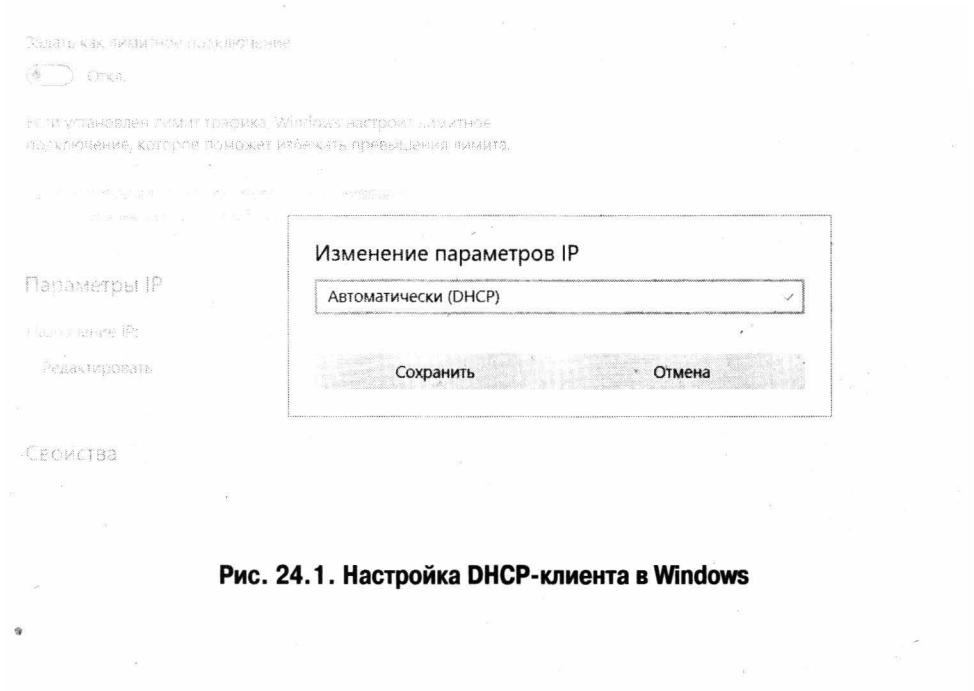
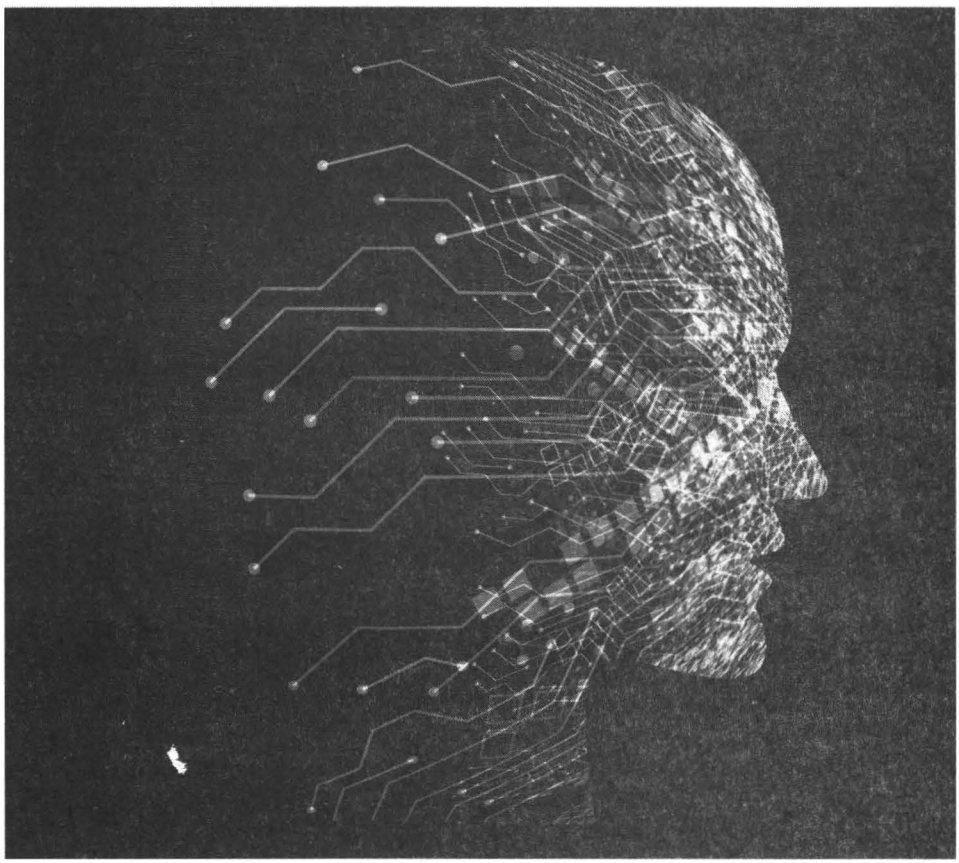


Рис. 24.1. Настройка DHCP-клиента в Windows

Глава 25.

Подключаем Linux к Windows-инфраструктуре



25.1. Знакомство с Samba

В современной компьютерной сети взаимодействуют самые различные устройства, управляемые самыми разными операционными системами: стационарные компьютеры и ноутбуки под управлением Windows, Linux и MacOS, смартфоны под управлением Linux, Android и iOS и т.д.

Samba - это сервис, позволяющий Linux-машине интегрироваться в Windows-сеть и стать ее полноценным участником. С помощью Samba вы можете использовать ресурсы Windows-сети, предоставлять ресурсы Windows-машинам и даже выступать в роли контроллера Active Directory.

Конечно, если честно, Samba в качестве контроллера Active Directory используется довольно редко. Чаще требуется настройка иного плана - *включение Linux-сервера в состав домена ActiveDirectory*, что и будет рассмотрено в этой главе.

После произведенной настройки наш сервер под управлением Linux сможет стать полноценным сервером домена ActiveDirectory и предоставлять другим компьютерам домена ресурсы, например, принтеры или дисковые ресурсы.

25.2. Установка необходимого программного обеспечения

Для реализации поставленной задачи нам нужно установить Samba, Kerberos и Winbind. Конечно, если вам не нужно интегрировать сервер в домен AD, то будет достаточно одного пакета Samba, но такие конфигурации встречаются редко, а интегрироваться в домен AD без Kerberos и Winbind невозможно.

Итак, установим необходимые пакеты (на примере Debian/Ubuntu):

```
sudo apt-get install install samba krb5-user winbind
```

Первый пакет (samba) позволяет стать членом домена и предоставлять/использовать ресурсы. Второй пакет необходим для работы протокола Kerberos, который используется для аутентификации в Windows. Без третьего пакета нельзя использовать учетную запись пользователя из AD.

Далее мы будем использовать следующие параметры для настройки:

- Имя домена - MY.COMPANY
- Имя контроллера ActiveDirectory - dc.my.company
- IP-адрес контроллера домена - 192.168.1.2
- Имя Linux-сервера - linux

25.3. Подготовительная настройка

Первым делом на компьютере linux нужно настроить DNS и синхронизацию времени. Откройте файл /etc/resolv.conf и добавьте в него следующие строки:

```
domain my.company
search my.company
nameserver 192.168.1.2
```

Нужно запретить редактировать этот файл, чтобы его не изменил NetworkManager:

```
sudo chattr +i /etc/resolv.conf
```

Также нужно отредактировать /etc/hosts:

```
127.0.0.1 localhost
127.0.1.1 linux.my.company      linux
```

Также убедитесь, что файл /etc/hostname содержит правильное имя узла (linux). На этом настройка DNS закончена, а после нее нужно настроить

синхронизацию времени, что очень важно для нормальной работы нашего сервера в AD.

Для автоматической синхронизации времени используется демон **ntpd**. Установим его:

```
sudo apt-get install ntp
```

После этого нужно отредактировать файл /etc/ntp.conf и добавить в него строку:

```
server dc.my.company
```

Теперь перезапускаем сеть и **ntpd**:

```
service networking restart
service ntp restart
```

25.4. Настройка Kerberos

Теперь мы можем приступить к настройке нашей связки Kerberos + Samba + Winbind. Первым делом настроим Kerberos. Нужно отредактировать файл /etc/krb5.conf и привести его к виду, представленному в листинге 25.1. Строки, выделенные жирным, необходимо заполнить своими данными.

Листинг 25.1. Файл /etc/krb5.conf

```
[libdefaults]
  default_realm = MY.COMPANY
  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]
  LAB.LOCAL = {
    kdc = dc
    admin_server = dc
    default_domain = MY.COMPANY
  }

[domain_realm]
  .lab.local = MY.COMPANY
  lab.local = MY.COMPANY
[login]
  krb4_convert = false
  krb4_get_tickets = false
```

Теперь проверим, работает ли Kerberos и можем ли мы аутентифицироваться в домене:

```
kinit пользователь@MY.COMPANY
```

Здесь нужно указать имя пользователя, зарегистрированного в домене MY.COMPANY. Имя домена нужно писать заглавными буквами.

Если аутентификация прошла успешно, и вы не получили сообщения об ошибке, значит вы все сделали правильно. В результате вам будет назначен тикет Kerberos. Просмотреть все тикеты можно командой klist, а уничтожить все тикеты - командой kdestroy.

25.5. Настройка Samba

Следующий шаг - это настройка Samba. Вам нужно отредактировать файл /etc/samba/smb.conf, в котором прописываются параметры входа в домен, а также предоставляемые нашим сервером ресурсы.

Самое главное в файле /etc/samba/smb.conf - это секция global, где описываются основные параметры Samba. Остальные секции, как правило,

описывают предоставляемые ресурсы. В листинге 25.2 будут приведены секции **global** и **public**. В последней описываются общий каталог, который будет доступен всем пользователям. Дополнительную информацию о настройке Samba можно получить по адресу https://help.ubuntu.com/community/Samba. Сейчас главное интегрировать Samba в домен AD, а описать дополнительные ресурсы и настроить к ним права доступа, думаю, вы сможете самостоятельно.

Листинг 25.2. Пример файла конфигурации /etc/samba/smb.conf для интеграции сервера в домен AD

```
[global]
   # Имя рабочей группы и домена нужно указывать заглавными
буквами
   workgroup = MY
   realm = MY.COMPANY

   # Авторизация через AD
   security = ADS
   encrypt passwords = true

   # Отключаем прокси DNS
   dns proxy = no

   # Ускоряем работу Samba
   socket options = TCP_NODELAY

   # Если не хотите, чтобы Samba стала контроллером домена,
установите
   # следующие параметры таким образом
   domain master = no
   local master = no
   preferred master = no
   os level = 0
   domain logons = no

   # Отключаем поддержку принтеров
   load printers = no
   show add printer wizard = no
   printcap name = /dev/null
   disable spoolss = yes

[public]
```

```
# Комментарий
comment = Public Directory
# путь
path = /var/samba
# не только чтение, но и запись
read only = no
# явно разрешаем запись
writable = yes
# разрешаем гостевой доступ
guest ok = yes
# разрешить просмотр содержимого каталога
browseable = yes
```

Сохраните файл конфигурации и введите команду **testparm**, которая проверит файл конфигурации Samba и сообщит, нет ли в нем ошибок:

```
# testparm
Load smb config files from /etc/samba/smb.conf
Loaded services file OK.
Server role: ROLE_DOMAIN_MEMBER
...
```

Как видите, файл в порядке, а роль сервера - член домена (ROLE_DOMAIN_MEMBER). Все, как нам и нужно. Теперь запустим Samba:

```
sudo service smbd start
```

Попробуем подключиться к нашему домену как пользователь user:

```
# net ads join -U user -D MY
Enter user's password:
Using short domain name — MY
Joined 'linux' to realm 'my.company'
```

Если при подключении к домену вы увидите сообщение «DNS update failed!», попробуйте первым делом перезагрузить свой компьютер (linux), прежде чем разбираться, почему не обновляется DNS.

25.6. Настройка Winbind

Если вам нужно только добавить свой сервер в состав домена AD, то на этом вся настройка закончена, но если вам нужно еще и взаимодействовать с пользователями домена, например, настраивать SMB-шары с разграничением доступа, вам нужен Winbind. Демон Winbind используется для связи локальной системы управления пользователями Linux с системой управления пользователями AD. Другими словами, Winbind нужен, если вы хотите видеть пользователей домена AD на локальной машине. Благодаря этому вы можете назначать пользователей домена владельцами папок и файлов на вашем компьютере и выполнять любые другие операции, связанные с правами доступа.

Для настройки Winbind используется опять файл конфигурации /etc/samba/smb.conf. В секцию **global** добавьте следующие строки:

```
# Диапазоны идентификаторов для виртуальных пользователей и
групп
idmap uid = 10000 - 40000
idmap gid = 10000 - 40000

# Не изменяйте эти строки
winbind enum groups = yes
winbind enum users = yes
winbind use default domain = yes
template shell = /bin/bash
winbind refresh tickets = yes
```

После этого нужно перезапустить Samba и Winbind, но сделать это нужно в определенной последовательности:

```
sudo service winbind stop
sudo service smbd restart
sudo service winbind start
```

После этого Winbind уже будет работать, но он все еще не интегрирован. Для интеграции Winbind в Linux. Откройте файл /etc/nsswitch.conf и найдите в нем строчки:

```
passwd:     compat
group:      compat
```

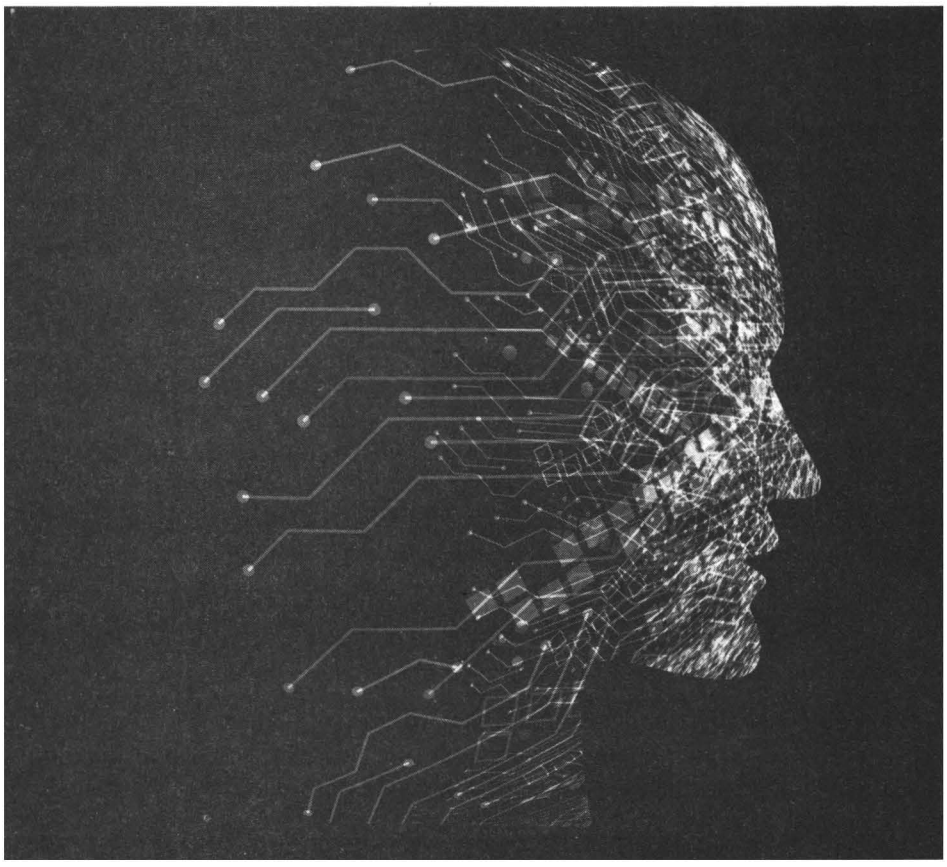
Эти строки нужно изменить так:

```
passwd:     compat winbind
group:      compat winbind
```

Раз мы уже настраиваем интеграцию с сетью Microsoft, то не грех и компьютер перезагрузить - по примеру Microsoft. После этого ваш сервер linux станет полноценным членом AD-домена MY.COMPANY.

Глава 26.

Резервное копирование



26.1. Средства резервного копирования

Резервное копирование - важнейшая задача администратора. Администратору могут простить многое, но не потерю данных. Именно поэтому важно разработать план резервного копирования и следовать ему.

При разработке вашего плана резервного копирования нужно предусмотреть методы защиты от:

- отказа жесткого диска;
- случайного удаления файлов;
- повреждения содержимого файлов;
- полного уничтожения компьютера (например, при пожаре).

Подумаем, как мы можем защититься от этих проблем. От отказа жесткого диска нас может спасти использование RAID-массивов. Как правило, при использовании RAID-массива проблема выхода жесткого диска из строя решается полностью.

От случайного удаления файлов или повреждения их содержимого могут помочь снимки (snaphosts) файловых систем. LVM2 помогает создать снимки и использовать их для подобного рода восстановления (см. гл. 14).

От полного уничтожения компьютера, например, при пожаре, защититься сложнее. Ведь при пожаре могут быть повреждены и резервные копии. Именно поэтому нет смысла делать резервные копии на второй жесткий диск, который установлен в вашем сервере. Резервные копии нужно делать или на внешний жесткий диск, который будет храниться в безопасном месте, в идеале - в пожаростойком сейфе и в другом помещении. Также можно хранить резервные копии на удаленной машине, которая находится в другом кабинете или вообще в другом здании. Это снижает вероятность того, что пожаром будут повреждены обе машины.

Существуют полностью автоматические средства резервного копирования вроде Amanda или Bacula, но когда есть несколько серверов (а не целый парк), их использование нецелесообразно, поскольку можно легко обойтись стандартными средствами Linux, что и будет показано в этой главе.

26.2. Разработка плана резервного копирования для веб-сервера

Также нужно разработать план резервного копирования сервера. Я предлагаю довольно простой план: сначала создается эталонный диск, который поможет при полном «падении» сервера. Резервные копии данных будем делать каждый день. Теоретически, делать резервные копии можно и чаще, но, как правило, резервное копирование производится ежедневно - в начале или в конце дня. Каждая следующая резервная копия будет перезаписывать предыдущую. Решение, может, и не очень хорошее, но довольно эффективное, если предполагается круглосуточная (или около того) работа службы поддержки.

Представим, что мы будем делать резервные копии в 4 часа утра. Как по мне, это оптимальное время для создания резервной копии, поскольку «совы» уже спят, а «жаворонки» еще не проснулись.

Скажем, в 7 часов утра еще не проснувшийся «жаворонок» нечаянно удаляет файл index.php. Он отправляет администратору сервера запрос с просьбой восстановить этот файл из резервной копии. Поскольку она была создана в 4 утра, то этот файл будет в составе резервной копии, и администратор сможет его восстановить.

Если пользователь еще что-то «начудит», то до 4 часов утра и у пользователя, и у администратора есть достаточно времени, чтобы связаться и произвести восстановление.

Если круглосуточной техподдержки не планируется, ситуация немного меняется. Представим, что администратор работает до 19:00, а пользователь удаляет свой файл в 22:00, а в 4:00 будет создана новая резервная копия, в которой уже нет файла index.php. Что делать? Тогда нужно изменить наше расписание. Ранее мы договорились, что будем делать резервные копии на другие машины или на внешний жесткий диск. Достаточно купить несколько (минимум два) внешних жестких диска и использовать их поочередно. Хотя можно обойтись и одним. Просто нужно изменить расписание.

Например, рабочий день администратора начинается в 9:00, резервная копия создается в 10:00. В 19:00 администратор уходит домой. Пользователю не спится и в 22:00 он удаляет файл. Он отправляет запрос администратору и у администратора до создания резервной копии есть час, чтобы восстановить файл.

Такая схема очень неплохая, но вы никогда не задумывались, почему резервные копии создаются ночью или рано утром? Правильно, чтобы снизить нагрузку на сервер. Представим, что у вас есть 100 пользователей и каждому вы предоставили 10 Гб места на диске. Если каждый из пользователей будет использовать все выделенное ему дисковое пространство, то нужно заархивировать 1 Тб данных, а это довольно много. Поэтому сервер может немного «подтормаживать» и пользователи будут жаловаться на то, что сайты открываются не так быстро, как хотелось.

Поэтому можно купить два внешних жестких диска и чередовать их. Один используем по четным числам, другой - по нечетным. Используя такую схему, можно настроить планировщик на создание резервной копии на 4 часа утра, как было предложено ранее, но при этом резервная копия будет создаваться во время минимальной нагрузки на сервер. И овцы целы, и волки сыты.

Все, что нужно сделать администратору - это подмонтировать нужный внешний диск до того, как он пойдет домой. В некоторых организациях используют 7 разных жестких дисков - на каждый день недели, тогда можно восстановить состояние файла, скажем, за прошлую среду. Такой способ очень затратный, но зато наиболее эффективный.

Что же касается внешних жестких дисков, то на сегодняшний день USB-диски размером 12 Тб, так что для резервного копирования среднестатистического сервера должно быть достаточно.

Также можно копировать резервные копии на внешние машины для лучшей сохранности - чем больше резервных копий, тем лучше.

26.3. Разработка сценария резервного копирования

Представим, что у нас есть веб-сервер хостинг-провайдера и нужно обеспечить его резервное копирование. Повторюсь, я буду использовать только стандартные средства, чтобы показать, насколько гибким может быть Linux. Что же касается уже готовых комплексов резервного копирования, то у каждого из них есть своя документация и множество статей в Интернете на русском языке. Поэтому не вижу смысла переписывать руководство еще раз другими словами.

Итак, приступим. Первым делом настройте сервер, чтобы он работал. Убедитесь, что настроили все и вам больше не придется вносить изменения в конфигурацию сервера. Будем считать, что пользовательские данные будут, как обычно, храниться в каталоге /home. Базы данных пользователей, как обычно, хранятся в /var/lib/mysql/<имя>. Для упрощения нашей конфигурации будем считать, что одному пользователю будет принадлежать только одна база данных, которая совпадает с именем пользователя. Например, есть сайт www.example.com, он принадлежит пользователю examplecom, его HTML-код хранится в каталоге /home/examplecom/public_html, а база данных - в каталоге /var/lib/mysql/examplecom.

Сразу рекомендую настраивать RAID-массивы, чтобы была возможность горячей замены, и вы могли на лету восстанавливать работоспособность сервера в случае выхода из строя одного из жестких дисков.

После этого используйте инструмент клонирования системы (или Clonezilla или Remastersys Backup, если у вас Debian/Ubuntu) для создания эталонного диска. Созданный образ запишите на болванку и положите где-то подальше от вашего сервера - в сейф или вообще унесите себе домой, если правила компании разрешают делать это. При создании эталонного диска каталог /home включать в состав образа не нужно.

Если с сервером произойдет катастрофа, которая уничтожит все его диски, то ни RAID, ни что-либо еще не спасет его. Зато вы сможете взять другой сервер и произвести установку с эталонного диска. При этом вам уже не придется тратить время на его настройку. Итак, можно считать, что мы обезопасили сервер от полного фиаско.

Теперь нужно позаботиться о создании резервной копии данных. Нам нужно копировать подкаталоги каталога /home и /var/lib/mysql. В первом хранятся HTML-код (и вспомогательные файлы) сайтов пользователей, а во

втором - база данных. Разработаем первую версию нашего сценария backup. Сначала нужно создать сам сценарий:

```
# cd /bin
# touch backup
# chmod +x backup
# nano backup
```

Затем введите следующий код:

```
#!/bin/bash

rm /mnt/backup/*

cd /home

for dn in `ls /home`; do
  echo "Creating backup for $dn"
  tar -czf /mnt/backup/$dn.tar.gz $dn
done
```

Разберемся, что и к чему. Первым делом мы удаляем старые резервные копии (команда rm). Программа **tar** с параметром **c** создает новый архив, но если файл архива существует, он не будет перезаписан. Чтобы добавить файлы в уже существующий архив можно использовать параметр **u**, но его нельзя использовать вместе с параметром с, поэтому нужно проверять существует ли файл архива. Если нет, тогда нужно создать архив (параметр **c**), если да, тогда нужно модифицировать архив (параметр **u**).

Далее мы создаем архив для каждого домашнего каталога пользователя. Например, если в /home у вас есть два подкаталога – user1 и user2, то будут созданы архивы user1.tar.gz и user2.tar.gz. Заметьте: мы не делаем резервную копию всего /home. Тогда мы получим один большой архив, с которым будет очень неудобно работать (да и можем превысить ограничение на максимальный размер файла - все зависит от объемов данных, с которыми вы работаете), поэтому проще и правильнее создать несколько меньших архивов.

Цикл `for` проходится по всем элементам вывода команды `ls /home`. В этом каталоге (/home) обычно находятся только каталоги (файлов в нем нет, только в его подкаталогах), то нам достаточно просто вывести его содержимое командой `ls`, а затем передать каждую строку вывода команде `tar`.

Опции `czf` означают, что нам нужно создать архив (c) в файле (параметр **f**) и при этом его сжать. Команда tar может создавать архивы и на ленточных устройствах, поэтому важно задать параметр f. Второй параметр - это имя архива, в нашем случае это будет /mnt/backup/<имя_пользователя>.tar.gz. Третий параметр - это имя каталога, который мы будем архивировать.

Каталог /mnt/backup должен существовать и к нему должен быть подмонтирован внешний жесткий диск. Об этом наш сценарий не заботиться, но при желании никто не помешает вам добавить строку вроде этой:

```
mount /dev/sdc1 /mnt/backup
```

Эта строку нужно поместить до команды **rm**, а в самой команде нужно изменить имя устройства (в нашем случае это /dev/sdc1).

Если вы решились управлять монтированием через сценарий, то в конце сценария нужно добавить строку размонтирования диска:

```
umount /mnt/backup
```

Также мы можем скопировать по SSH созданные архивы на удаленную машину. Тогда у вас будет две резервных копии, что повышает надежность нашего плана резервного копирования. После команды tar добавьте команду:

```
scp $dn.tar.gz user@examle.com:/backups
```

Данная команда скопирует файл <имя_пользователя>.tar.gz на SSH-сервер example.com в каталог /backups. Вот только есть один момент: программа scp запрашивает пароль пользователя для копирования файлов на удаленный сервер. Чтобы этого не происходило (тогда вам нужно присутствовать в 4 утра перед компьютером и вводить пароль), нужно настроить аутентификацию с помощью ключей. Пока не будем на этом останавливаться, а рассмотрим измененную версию нашего сценария:

```
#!/bin/bash

mount /dev/sdc1 /mnt/backup

rm /mnt/backup/*

cd /home
```

```
for dn in `ls /home`; do
  echo "Creating backup for $dn"
  tar -czf /mnt/backup/$dn.tar.gz $dn
scp $dn.tar.gz user@examle.com:/backups
done

# Сбрасываем буферы на диск (не обязательно)
sync
# Размонтируем файловую систему
umount /mnt/backup
```

Если вам нужно копировать только содержимое каталога /home, то у вас есть уже рабочая версия сценария. Однако у нас еще есть каталог /var/lib/mysql. Однако код цикла для создания резервной копии будет отличен, поскольку в каталоге /var/lib/mysql кроме подкаталогов с файлами баз данных есть еще и обычные файлы, которые нам не нужно никуда копировать. Поэтому в цикле нужно проверить, является ли элемент, который вывела программа **ls**, каталогом и заархивировать только его:

```
echo "Database backup..."
cd /var/lib/mysql
for dn in `ls /var/lib/mysql`; do
  test -d && "$dn" && tar -czf /mnt/backup/db/$dn.tar.gz
$dn
done
```

Обратите внимание: архивы создаются в каталоге /mnt/backup/db: если имя базы данных у нас совпадает с именем пользователя, то ничего хорошего из затеи хранить архив в каталоге /mnt/backup у нас не выйдет. Поэтому мы должны создать каталог **db** заранее и хранить в нем резервные копии баз данных.

Кроме домашних каталогов и базы данных нужно еще сделать резервную копию почтовых ящиков пользователей, которые обычно хранятся в каталоге /var/mail. Здесь все просто - каждому пользователю соответствует файл почтового ящика, поэтому можем использовать код, подобный созданию резервной копии домашнего каталога, только резервные копии будем хранить в каталоге /mnt/backup/mail:

```
cd /var/mail

for dn in `ls /var/mail`; do
  echo "Creating backup for $dn [mail]"
```

```
   tar -czf /mnt/backup/mail/$dn.tar.gz $dn
# Если нужно, раскомментируйте строку
# scp $dn.tar.gz user@examle.com:/backups/mail
done
```

Теперь интегрируем наши два сценария в один общий (листинг 26.1)

Листинг 26.1. Сценарий /bin/backup

```
#!/bin/bash

mount /dev/sdc1 /mnt/backup

# Рекурсивное удаление всего из /mnt/backup
rm -r /mnt/backup/*
# Создаем каталог для хранения базы данных
mkdir /mnt/backup/db
# Каталог для почтовых ящиков
mkdir /mnt/backup/mail

cd /home

# Копируем домашние каталоги
for dn in `ls /home`; do
   echo "Creating backup for $dn"
   tar -czf /mnt/backup/$dn.tar.gz $dn
done

# Копируем базы данных
echo "Database backup..."
cd /var/lib/mysql
for dn in `ls /var/lib/mysql`; do
   test -d && "$dn" && tar -czf /mnt/backup/db/$dn.tar.gz $dn
done

cd /var/mail

for dn in `ls /var/mail`; do
   echo "Creating backup for $dn [mail]"
   tar -czf /mnt/backup/mail/$dn.tar.gz $dn
done

# Если нужно хранить резервные копии на удаленной машине
# scp -r /mnt/backup user@example.com:/backups
```

```
# Сбрасываем буферы на диск (не обязательно)
sync
# Размонтируем файловую систему
umount /mnt/backup
```

Наш сценарий готов полностью. Осталось только настроить SSH-сервер на аутентификацию без пароля. Для этого на клиенте (то есть на нашем сервере, резервное копирование которого вы будете производить, далее SSH-клиент) введите команду:

```
# ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
```

Эта команда создает пару приватного/публичного ключей. Обратите внимание, что когда программа попросит вас ввести ключевую фразу, вводить ничего не нужно - вы же не хотите вводить пароль?

Теперь на SSH-сервере (на компьютере, на который вы будете отправлять резервные копии) откройте файл конфигурации sshd. Обычно это /etc/ssh/sshd_config. Добавьте в него строки:

```
RSAAuthentication yes
PubkeyAuthentication yes
AuthorizedKeysFile %h/.ssh/authorized_keys
```

Обычно они в нем уже есть, но их нужно раскомментировать. Строка «%h/.ssh/authorized_keys» означает, что ключи будут храниться в домашнем каталоге пользователя (%h), подкаталог .ssh, файл authorized_keys. Если вы этого файла там нет, вы можете его создать самостоятельно.

Скопируем наш файл публичный ключ на SSH-сервер (команду вводим на SSH-клиенте):

```
scp /root/.ssh/id_rsa.pub root@example.com:/root/
```

На SSH-сервере вводим следующую команду, помещающую ваш ключ в файл authorized_keys:

```
cat /root/id_rsa.pub >> /root/.ssh/authorized_keys
```

Перезапустите SSH-сервер:

```
# service ssh restart
```

Теперь переходим на SSH-клиент и попробуем скопировать файл на сервер. На этот раз команда ssh не должна запрашивать пароль:

```
scp /root/.ssh/id_rsa.pub root@example.com:/root/
```

Все, наш SSH-сервер настроен. Не забудьте на нем создать каталог /backups и можно приступать к тестированию сценария резервного копирования.

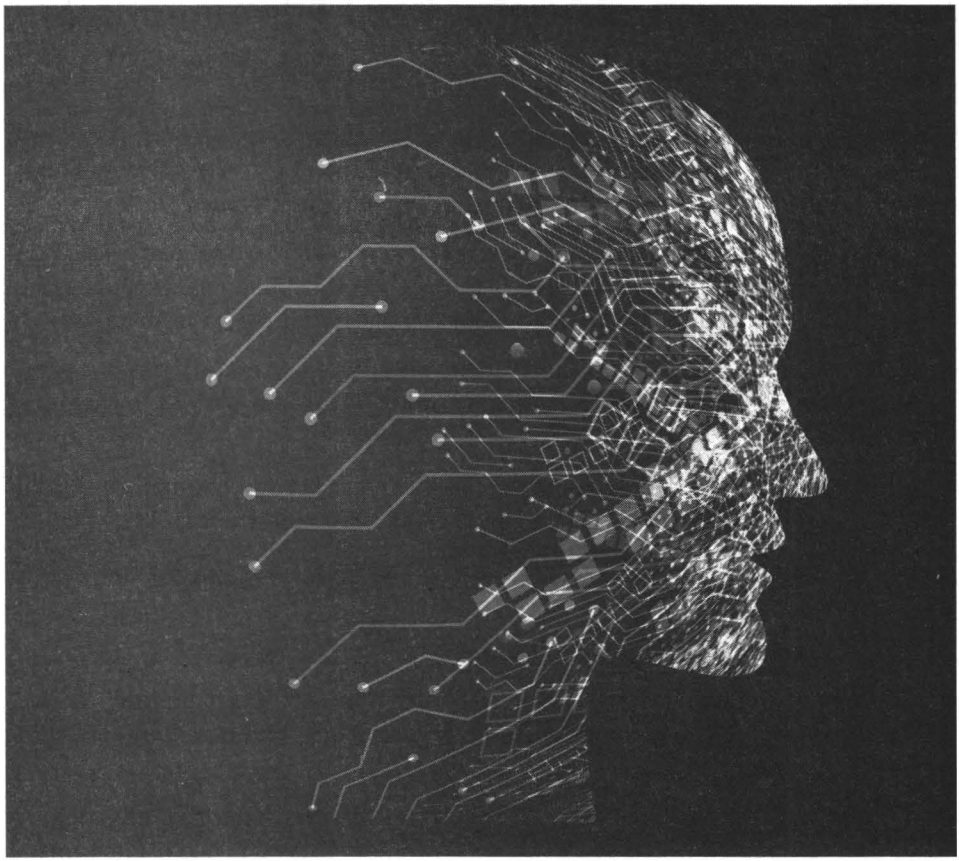
Когда вы убедитесь, что все работает, как нужно, добавьте вызов сценария /bin/backup на вашем веб-сервере в таблицу расписания планировщика (см. гл. 16).

Теперь на секундочку задумайтесь, что мы только что создали. Мы создали систему резервного копирования с ротацией резервных копий и возможностью хранения резервных копий на удаленной машине. И все это - стандартными средствами.

Если у вас есть несколько серверов, вы можете скопировать сценарий **backup** на каждый сервер (возможно, придется модифицировать пути, в зависимости от настроек и специфики сервера) и настроить его на автоматический запуск. Единственный мой совет - настраивайте резервное копирование каждого сервера на разное время, например, с промежутком в полчаса или даже час, если информации много. Если вы создаете резервную копию первого сервера в 4 утра, второго создавайте в 4:30, третьего в 5:00 и т.д.

Глава 27.

Обеспечение безопасности



Обеспечение безопасности сервера – вопрос достаточно емкий. По сути, можно написать отдельную книгу – и такие книги есть. При обеспечении безопасности любого объекта нужно ответить на три вопроса:

- Что нужно защитить? Определите, что именно вы хотите защитить. Возможно, это хранимые данные, а может сетевой сервис (например, вы не хотите, чтобы неавторизированные пользователи ним пользовались).
- От чего вы защищаетесь? Что больше всего вас беспокоит? Утечка конфиденциальных данных? Потеря данных? Потеря дохода, вызванная техническим сбоем?
- От кого вы защищаетесь? От действий неквалифицированных пользователей или от действий хакеров-профи? Меры безопасности будут существенно отличаться в этих двух случаях.

Далее мы рассмотрим следующие вопросы:

- **Локальная безопасность** – здесь мы поговорим об обеспечении безопасности сервера внутри предприятия, чтобы с ним, родимым, ничего плохого не случилось.
- **Защита от сетевых атак** – сетевые атаки могут быть как внутренними (злоумышленник находится во внутренней сети), так и внешними. В главе 19 мы рассматривали настройку брандмауэра, в этой главе мы

поговорим о том, как настроить брандмауэр, чтобы защитить сервер от сетевых атак. Здесь же будет рассмотрена защита от брутфорса паролей SSH и предотвращение сканирования портов сервера.

- **Защита сетевых служб** – отдельный вопрос в области обеспечения безопасности – защита сетевых служб. Для общей защиты всех сетевых сервисов мы будем использовать популярное решение fail2ban.
- **Шифрование данных** – будет рассмотрено шифрование файловой системы посредством eCryptfs. Если вас интересует защита данных как таковая, то eCruptfs – достойное решение. Если же вам нужна защита персональных данных в рамках ФЗ-152, то придется поискать другое решение, поскольку eCryptfs не поддерживает ГОСТ-алгоритмы шифрования и для проверяющих такая защита окажется недостаточной, хотя алгоритм AES, используемый в eCryptfs, ничем не хуже.
- **Настройка собственного VPN-сервера** – если файловая система eCryptfs защищает данные, которые хранятся на жестком диске, то VPN-сервер будет защищать данные, передаваемые по сети.

27.1. Локальная безопасность сервера

Локальная безопасность сервера начинается с организации серверной комнаты – отдельного помещения, в котором будут находиться серверы и другое сетевое оборудование. При этом должны соблюдаться следующие требования к серверному помещению:

- Организация пропускного режима, например, электронный замок с чипкартой, чтобы было видно, кто входил в помещение. Такая система позволяет ограничить доступ сотрудников в отдельные помещения предприятия.
- Организация видеонаблюдения не только за входной дверью, но и внутри помещения, в том числе за консолью сервера, чтобы было видно, кто и что делал в помещении.
- Обеспечение оптимальной температуры для работы оборудования, как правило, это 18 градусов.
- Обеспечение отдельного помещения для IT-отдела. Запомните: серверная – это серверная, человек должен там находиться, если возникла какая-то проблема, которую невозможно решить удаленно (из другого

кабинета). В серверной не должен находиться IT-персонал и наоборот – сервер не должен находиться в IT-отделе.

- Установка пожарной сигнализации.
- Установка охранной сигнализации с датчиками движения, открытия дверей, разбития стекла.
- Установка решеток на окна и дополнительных (неэлектронных) замков.
- Резервирование Интернет-канала и электросети. С Интернет-каналом проще – достаточно хотя бы подключения к другому провайдеру. Линии двух провайдеров, скорее всего, будут проходить по одному и тому же участку и в случае стихийного бедствия будут обе повреждены. Но, по крайней мере, вы хотя бы будете застрахованы от проблем у какого-то из провайдеров. Бывают всякие технические сбои, а пока основной соединение «отремонтируют», вы будете пользоваться резервным. С электричеством сложнее. Самый простой и дешевый вариант – дизель-генераторы, но их не всегда можно установить. Источники бесперебойного питания, если они нужны не только для корректного завершения работы, то обойдутся довольно дорого. Цена ИБП уровня предприятия легко достигает нескольких тысяч долларов и это без батарей.

Если на вашем предприятии готовились документы по защите персональных данных, то все эти требования должны быть учтены в документе, именуемом Модель угроз. В нем приводятся актуальные угрозы и способы их устранения.

Как видите, много чего нужно сделать до того, как установить пломбы на корпус сервера и пароль на BIOS. Организация серверного помещения – дело не дешевое, поэтому многие организации предпочитают размещать сервер в дата-центре (colocation), где все эти условия уже созданы. Так выходит дешевле. Еще дешевле будет, если сервер не покупать вообще, а ограничиться виртуальным сервером. Виртуальный сервер начальной конфигурации обойдется даже дешевле услуги colocation физического сервера. Поэтому если вы только начинаете планировать инфраструктуру предприятия, задумайтесь о виртуальных серверах, как средстве экономии средств предприятия.

Вот, а теперь настало время поговорить о том, что многие считают локальной безопасностью:

- Установите пломбы на корпус сервера (серверов) и пароль на вход в BIOS. Этим вы немного обезопаситесь от злоумышленника, который загрузится с LiveUSB/LiveCD для получения доступа к данным сервера

в обход операционной системы. При желании, конечно, можно сорвать пломбы и сбросить настройки BIOS, но это не останется незамеченным. Пароль нужно устанавливать не на загрузку, а на вход в BIOS, иначе при перезагрузке сервера вам нужно будет лично присутствовать при его загрузке для ввода пароля.

- Установите пароль загрузчика GRUB2 (гл. 15). Аналогично, пароль нужно устанавливать не на загрузку, а на изменение параметров GRUB2, чтобы никто не смог подменить систему инициализации и обойти проверку пароля.

О втором случае хотелось бы поговорить отдельно. Параметр ядра **init** позволяет задать систему инициализации. В качестве значения init вы можете указать любую другую программу, и она будет выполнена с максимальными правами (поскольку ядро запускает систему инициализации с максимальными правами). Проведем небольшой эксперимент с дистрибутивом Astra Linux. Перезагрузите систему и в меню загрузчика нажмите e для редактирования параметров ядра. Добавьте параметр init=/bin/bash, как показано на рис. 27.1.

Рис. 27.1. Передача параметров ядра

После этого нажмите F10 и дождитесь завершения загрузки. После этого вы увидите приглашение командной строки (ведь мы запустили командный интерпретатор вместо системы инициализации) с решеткой, что свидетельствует о максимальных правах (рис. 27.2). Все, теперь вы можете делать с системой все, что вам захочется.

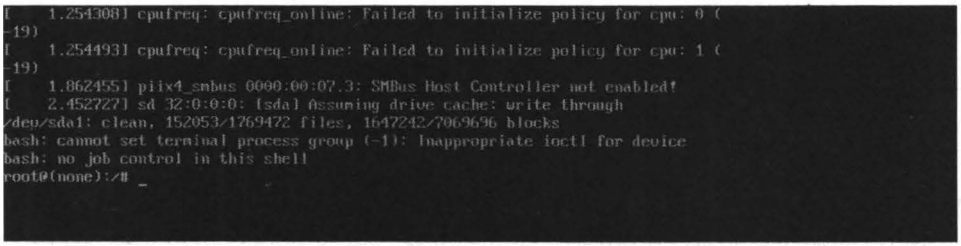


Рис. 27.2. Получены максимальные права

Некоторые дистрибутивы блокируют запуск в подобном режиме – предлагают ввести пароль root, но Astra Linux к таким не относится, что является существенным недостатком, особенно для дистрибутива, который позиционирует себя как дистрибутив «особого назначения».

Защититься поможет только установка пароля загрузчика, как было показано в главе 15. Данный пароль попросту не позволит редактировать параметры ядра и подменить систему инициализации.

27.2. Защита от сетевых атак

Существует очень много различных атак сети. Среди них можно выделить три основных вида: DoS-атаки, атаки, целью которых является получение доступа к сети, и разведывательные атаки, целью которых является получение информации о сети. В этом разделе мы поговорим о DoS-атаках, их разновидностях, обнаружении и защите сети от этих DoS-атак средствами маршрутизатора. При этом маршрутизатор может быть как программным (компьютер под управлением Linux и должным образом настроенным ПО) или же программным («коробочка» от Cisco или другого вендора)

27.2.1. DoS- и DDoS-атаки

DoS-атака, или атака на отказ в обслуживании (Denial of Service - DoS), является наиболее распространенной в информационном мире атакой. С другой стороны, DoS-атака является наиболее молодой. О DoS-атаках всерьез заговорили только в 1999 году, когда были «завалены» Web-сайты из-

вестных корпораций (Amazon, Yahoo, CNN, eBay, E-Trade и др.). Хотя сама техника DoS-атак появилась в 1996 году, но до 1999 на нее особо никто не обращал внимания.

Позже появились распределенные DoS-атаки (Distributed Denial of Service, DDoS). В этом случае атака производится не одним узлом, а несколькими, что усложняет пресечение атаки и обнаружение источника атаки.

Что же такое DoS-атака? Ее цель – захватить все ресурсы компьютера-жертвы, чтобы другие пользователи не смогли использовать этот компьютер. К ресурсам относятся: память, процессорное время, дисковое пространство, сетевые ресурсы и др.

Рассмотрим наиболее распространенные виды DoS-атак:

- Smurf - злоумышленником отправляются широковещательные echo-пакеты (протокол ICMP), в заголовках которых в качестве источника указывается адрес жертвы, в результате все системы, получившие ping-запрос, «заваливают» жертву echo-ответами.
- ICMP-flood - похожа на Smurf, но отправляет ICMP-запросы напрямую на узел-жертву, без использования широковещательного адреса.
- UDP-flood - отправка на узел-мишень огромного количества UDP-пакетов, что приводит к «связыванию» сетевых ресурсов
- TCP-flood - аналогична предыдущей, но используются TCP-пакеты
- TCP SYN-flood - одна из самых интересных атак. О ней в этом номере мы уже говорили, я позволю себе напомнить вам ее принцип. Злоумышленник отправляет на открытый порт много SYN-пакетов с недостижимым адресом источника. Атакуемый маршрутизатор должен ответить пакетом <SYN, ACK>, но ведь узел, указанный в качестве источника недоступен, поэтому трехступенчатая схема установления TCP-соединения не завершается. Поскольку таких SYN-пакетов очень много, лимит на количество открытых соединений очень быстро превышается, и жертва отказывается принимать запросы на установление соединения от обычных пользователей сети. В результате страдают обычные пользователи сети, которые не могут подключиться к серверу.

Далее мы поговорим, как обнаружить DoS-атаку и как защититься от нее средствами маршрутизатора. Для защиты от DoS мы будем использовать перехват TCP-соединений (TCP Intercept), пакетный фильтр, NBAR и ограничение потока ICMP-пакетов. Нужно отметить, что далеко не всегда

можно уберечься от DoS-атаки, но при правильной настройке всегда можно свести старания злоумышленника на нет.

27.2.2. Обнаружение атаки

Вот основные симптомы DoS-атаки:

- Огромное количество ARP-запросов;
- Огромное количество записей в вашей NAT/PAT-таблице;
- Повышенное использование памяти маршрутизатора;
- Повышенное использование процессорного времени маршрутизатора.

Напомню, что мы рассматриваем управляемые маршрутизаторы, оснащенные собственной операционной системой. Для обнаружения симптомов DoS-атаки вам нужно подключиться к своему маршрутизатору, и используя диагностические утилиты операционной системы маршрутизатора, выяснить, имеет ли место DoS-атака. Например, в Cisco IOS просмотреть использование процессора можно с помощью команды show processes cpu. В Linux можно использовать команду top или аналогичные альтернативные команды.

27.2.3. Специальные настройки ядра

Используя специальные настройки ядра, можно предотвратить DDoS-атаку. Для этого откройте файл /etc/sysctl.conf и добавьте в него следующие строки:

```
kernel.printk = 4 4 1 7
kernel.panic = 10
kernel.sysrq = 0
kernel.shmmax = 4294967296
kernel.shmall = 4194304
kernel.core_uses_pid = 1
kernel.msgmnb = 65536
kernel.msgmax = 65536
```

```
vm.swappiness = 20
vm.dirty_ratio = 80
vm.dirty_background_ratio = 5
fs.file-max = 2097152
net.core.netdev_max_backlog = 262144
net.core.rmem_default = 31457280
net.core.rmem_max = 67108864
net.core.wmem_default = 31457280
net.core.wmem_max = 67108864
net.core.somaxconn = 65535
net.core.optmem_max = 25165824
net.ipv4.neigh.default.gc_thresh1 = 4096
net.ipv4.neigh.default.gc_thresh2 = 8192
net.ipv4.neigh.default.gc_thresh3 = 16384
net.ipv4.neigh.default.gc_interval = 5
net.ipv4.neigh.default.gc_stale_time = 120
net.netfilter.nf_conntrack_max = 10000000
net.netfilter.nf_conntrack_tcp_loose = 0
net.netfilter.nf_conntrack_tcp_timeout_established = 1800
net.netfilter.nf_conntrack_tcp_timeout_close = 10
net.netfilter.nf_conntrack_tcp_timeout_close_wait = 10
net.netfilter.nf_conntrack_tcp_timeout_fin_wait = 20
net.netfilter.nf_conntrack_tcp_timeout_last_ack = 20
net.netfilter.nf_conntrack_tcp_timeout_syn_recv = 20
net.netfilter.nf_conntrack_tcp_timeout_syn_sent = 20
net.netfilter.nf_conntrack_tcp_timeout_time_wait = 10
net.ipv4.tcp_slow_start_after_idle = 0
net.ipv4.ip_local_port_range = 1024 65000
net.ipv4.ip_no_pmtu_disc = 1
net.ipv4.route.flush = 1
net.ipv4.route.max_size = 8048576
net.ipv4.icmp_echo_ignore_broadcasts = 1
net.ipv4.icmp_ignore_bogus_error_responses = 1
net.ipv4.tcp_congestion_control = htcp
net.ipv4.tcp_mem = 65536 131072 262144
net.ipv4.udp_mem = 65536 131072 262144
net.ipv4.tcp_rmem = 4096 87380 33554432
net.ipv4.udp_rmem_min = 16384
net.ipv4.tcp_wmem = 4096 87380 33554432
net.ipv4.udp_wmem_min = 16384
net.ipv4.tcp_max_tw_buckets = 1440000
net.ipv4.tcp_tw_recycle = 0
net.ipv4.tcp_tw_reuse = 1
net.ipv4.tcp_max_orphans = 400000
net.ipv4.tcp_window_scaling = 1
```

```
net.ipv4.tcp_rfc1337 = 1
net.ipv4.tcp_syncookies = 1
net.ipv4.tcp_synack_retries = 1
net.ipv4.tcp_syn_retries = 2
net.ipv4.tcp_max_syn_backlog = 16384
net.ipv4.tcp_timestamps = 1
net.ipv4.tcp_sack = 1
net.ipv4.tcp_fack = 1
net.ipv4.tcp_ecn = 2
net.ipv4.tcp_fin_timeout = 10
net.ipv4.tcp_keepalive_time = 600
net.ipv4.tcp_keepalive_intvl = 60
net.ipv4.tcp_keepalive_probes = 10
net.ipv4.tcp_no_metrics_save = 1
net.ipv4.ip_forward = 0
net.ipv4.conf.all.accept_redirects = 0
net.ipv4.conf.all.send_redirects = 0
net.ipv4.conf.all.accept_source_route = 0
net.ipv4.conf.all.rp_filter = 1
```

Все эти строки помогут повысить производительность вашего сервера во время DDoS-атаки – он дольше продержится и у вас будет дополнительное время на определение источника атаки и его блокировку. Чтобы данные изменения вступили в силу, выполните команду sudo sysctl –p.

27.2.4. Блокируем все подозрительное

Предотвратить DDoS-атаку могут и определенные правила брандмауэра. Мы будем использовать iptables, поскольку он более гибкий в настройке и предоставляет те возможности, которых нет в других продуктах.

Блокируем неправильные пакеты:

```
iptables -t mangle -A PREROUTING -m conntrack --ctstate
INVALID -j DROP
```

Блокируем новые не-SYN пакеты:

```
iptables -t mangle -A PREROUTING -p tcp ! --syn -m
conntrack --ctstate NEW -j DROP
```

Данное правило блокирует все пакеты, которые являются новыми (не принадлежат уже установленному соединению) и не используют флаг SYN.

Блокируем пакеты с неправильным значением MSS:

```
iptables -t mangle -A PREROUTING -p tcp -m conntrack
--ctstate NEW -m tcpmss ! --mss 536:65535 -j DROP
```

Такие пакеты выглядят подозрительно, поэтому лучше их заблокировать, чем проглядеть атаку.

Блокируем пакеты с поддельными TCP-флагами:

```
iptables -t mangle -A PREROUTING -p tcp --tcp-flags
FIN,SYN,RST,PSH,ACK,URG NONE -j DROP
iptables -t mangle -A PREROUTING -p tcp --tcp-flags FIN,SYN
FIN,SYN -j DROP
iptables -t mangle -A PREROUTING -p tcp --tcp-flags SYN,RST
SYN,RST -j DROP
iptables -t mangle -A PREROUTING -p tcp --tcp-flags FIN,RST
FIN,RST -j DROP
iptables -t mangle -A PREROUTING -p tcp --tcp-flags FIN,ACK
FIN -j DROP
iptables -t mangle -A PREROUTING -p tcp --tcp-flags ACK,URG
URG -j DROP
iptables -t mangle -A PREROUTING -p tcp --tcp-flags ACK,FIN
FIN -j DROP
iptables -t mangle -A PREROUTING -p tcp --tcp-flags ACK,PSH
PSH -j DROP
iptables -t mangle -A PREROUTING -p tcp --tcp-flags ALL ALL
-j DROP
iptables -t mangle -A PREROUTING -p tcp --tcp-flags ALL NONE
-j DROP
iptables -t mangle -A PREROUTING -p tcp --tcp-flags ALL
FIN,PSH,URG -j DROP
iptables -t mangle -A PREROUTING -p tcp --tcp-flags ALL
SYN,FIN,PSH,URG -j DROP
iptables -t mangle -A PREROUTING -p tcp --tcp-flags ALL
SYN,RST,ACK,FIN,URG -j DROP
```

Приведенный выше набор правил блокирует пакеты, использующих фиктивные флаги TCP, т.е. флаги TCP, которые нормальные пакеты не будут использовать.

27.2.5. Блокируем пакеты из-под приватных подсетей (спуфинг)

Правила будут такими:

```
iptables -t mangle -A PREROUTING -s 224.0.0.0/3 -j DROP
iptables -t mangle -A PREROUTING -s 169.254.0.0/16 -j DROP
iptables -t mangle -A PREROUTING -s 172.16.0.0/12 -j DROP
iptables -t mangle -A PREROUTING -s 192.0.2.0/24 -j DROP
iptables -t mangle -A PREROUTING -s 192.168.0.0/16 -j DROP
iptables -t mangle -A PREROUTING -s 10.0.0.0/8 -j DROP
iptables -t mangle -A PREROUTING -s 0.0.0.0/8 -j DROP
iptables -t mangle -A PREROUTING -s 240.0.0.0/5 -j DROP
iptables -t mangle -A PREROUTING -s 127.0.0.0/8 ! -i lo -j
DROP
```

Эти правила блокируют поддельные пакеты, исходящие из частных (локальных) подсетей. На общедоступном сетевом интерфейсе обычно не должно быть пакетов с локальных подсетей.

Данные правила предполагают, что ваш интерфейс петли (loopback) использует адрес IP 127.0.0.0/8.

27.2.6. Дополнительные правила

Также не будут лишними следующими правила:

```
iptables -t mangle -A PREROUTING -p icmp -j DROP
iptables -A INPUT -p tcp -m connlimit --connlimit-above 80
-j REJECT --reject-with tcp-reset
```

Первое правило удаляет все ICMP-пакеты. Как правило, ICMP используется только для ping'a – чтобы проверить, «жив» ли узел или нет. Обычно это вам не нужно (есть и так много средств мониторинга, позволяющих убедиться, что с узлом все хорошо), а сторонним узлам и пользователям и подавно не нужно ничего знать о вашем узле. Поэтому от ICMP можно смело отказаться.

Второе правило позволяет предотвратить атаки соединения. Он отклоняет соединения от хостов, которые уже установили более 80 соединений. Если

у вас возникнут какие-либо проблемы с легитимными узлами, вам нужно поднять лимит на количество установленных TCP-соединений.

Следующие правила ограничивают число новых соединений TCP, которые клиент может установить за секунду. Они полезны против атак на соединения, но не годятся против SYN-флуда, поскольку обычно используется бесконечное количество разных поддельных IP-адресов источника.

```
iptables -A INPUT -p tcp -m conntrack --ctstate NEW -m
limit --limit 60/s --limit-burst 20 -j ACCEPT
iptables -A INPUT -p tcp -m conntrack --ctstate NEW -j DROP
```

Следующее правило блокирует фрагментированные пакеты:

```
iptables -t mangle -A PREROUTING -f -j DROP
```

А эти правила ограничивают входящие TCP RST пакеты, чтобы избежать TCP RST-наводнения:

```
iptables -A INPUT -p tcp --tcp-flags RST RST -m limit
--limit 2/s --limit-burst 2 -j ACCEPT
iptables -A INPUT -p tcp --tcp-flags RST RST -j DROP
```

27.2.7. Полный список анти-DDoS правил

Собирая все вместе, приводим полный список правил, позволяющих существенно защитить ваш сервер от всякого рода DDoS-атак:

```
### 1: Избавляемся от неправильных пакетов ###
/sbin/iptables -t mangle -A PREROUTING -m conntrack --ctstate
INVALID -j DROP

### 2: Удаляем новые TCP-пакеты без флага SYN ###
/sbin/iptables -t mangle -A PREROUTING -p tcp ! --syn -m
conntrack --ctstate NEW -j DROP

### 3: Удаляем пакеты с подозрительным значением MSS ###
/sbin/iptables -t mangle -A PREROUTING -p tcp -m conntrack
--ctstate NEW -m tcpmss ! --mss 536:65535 -j DROP

### 4: Блокируем пакеты с фиктивными TCP-флагами ###
```

```
/sbin/iptables -t mangle -A PREROUTING -p tcp --tcp-flags
FIN,SYN,RST,PSH,ACK,URG NONE -j DROP
/sbin/iptables -t mangle -A PREROUTING -p tcp --tcp-flags
FIN,SYN FIN,SYN -j DROP
/sbin/iptables -t mangle -A PREROUTING -p tcp --tcp-flags
SYN,RST SYN,RST -j DROP
/sbin/iptables -t mangle -A PREROUTING -p tcp --tcp-flags
FIN,RST FIN,RST -j DROP
/sbin/iptables -t mangle -A PREROUTING -p tcp --tcp-flags
FIN,ACK FIN -j DROP
/sbin/iptables -t mangle -A PREROUTING -p tcp --tcp-flags
ACK,URG URG -j DROP
/sbin/iptables -t mangle -A PREROUTING -p tcp --tcp-flags
ACK,FIN FIN -j DROP
/sbin/iptables -t mangle -A PREROUTING -p tcp --tcp-flags
ACK,PSH PSH -j DROP
/sbin/iptables -t mangle -A PREROUTING -p tcp --tcp-flags ALL
ALL -j DROP
/sbin/iptables -t mangle -A PREROUTING -p tcp --tcp-flags ALL
NONE -j DROP
/sbin/iptables -t mangle -A PREROUTING -p tcp --tcp-flags ALL
FIN,PSH,URG -j DROP
/sbin/iptables -t mangle -A PREROUTING -p tcp --tcp-flags ALL
SYN,FIN,PSH,URG -j DROP
/sbin/iptables -t mangle -A PREROUTING -p tcp --tcp-flags ALL
SYN,RST,ACK,FIN,URG -j DROP

### 5: Блокируем спуфинг-пакеты ###
/sbin/iptables -t mangle -A PREROUTING -s 224.0.0.0/3 -j DROP
/sbin/iptables -t mangle -A PREROUTING -s 169.254.0.0/16 -j
DROP
/sbin/iptables -t mangle -A PREROUTING -s 172.16.0.0/12 -j
DROP
/sbin/iptables -t mangle -A PREROUTING -s 192.0.2.0/24 -j DROP
/sbin/iptables -t mangle -A PREROUTING -s 192.168.0.0/16 -j
DROP
/sbin/iptables -t mangle -A PREROUTING -s 10.0.0.0/8 -j DROP
/sbin/iptables -t mangle -A PREROUTING -s 0.0.0.0/8 -j DROP
/sbin/iptables -t mangle -A PREROUTING -s 240.0.0.0/5 -j
DROP
/sbin/iptables -t mangle -A PREROUTING -s 127.0.0.0/8 ! -i
lo -j DROP

### 6: Блокируем ICMP ###
/sbin/iptables -t mangle -A PREROUTING -p icmp -j DROP
```

```
### 7: Удаляем фрагментированные пакеты ###
/sbin/iptables -t mangle -A PREROUTING -f -j DROP

### 8: Ограничиваем соединения по IP ###
/sbin/iptables -A INPUT -p tcp -m connlimit --connlimit-
above 111 -j REJECT --reject-with tcp-reset

### 9: Ограничиваем RST-пакеты ###
/sbin/iptables -A INPUT -p tcp --tcp-flags RST RST -m limit
--limit 2/s --limit-burst 2 -j ACCEPT
/sbin/iptables -A INPUT -p tcp --tcp-flags RST RST -j DROP

### 10: Ограничиваем число TCP-соединений в секунду с
одного IP ###
/sbin/iptables -A INPUT -p tcp -m conntrack --ctstate NEW
-m limit --limit 60/s --limit-burst 20 -j ACCEPT
/sbin/iptables -A INPUT -p tcp -m conntrack --ctstate NEW
-j DROP
```

27.2.8. Защита от брутфорса SSH

С помощью правил **iptables** можно защититься от брутфорса (перебора пароля) SSH:

```
/sbin/iptables -A INPUT -p tcp --dport ssh -m conntrack
--ctstate NEW -m recent --set
/sbin/iptables -A INPUT -p tcp --dport ssh -m conntrack
--ctstate NEW -m recent --update --seconds 60 --hitcount 10 -j
DROP
```

27.2.9. Запрет сканирования портов

Также **iptables** позволяет защититься от сканирования портов:

```
/sbin/iptables -N port-scanning
/sbin/iptables -A port-scanning -p tcp --tcp-flags
SYN,ACK,FIN,RST RST -m limit --limit 1/s --limit-burst 2 -j
RETURN
/sbin/iptables -A port-scanning -j DROP
```

27.2.10. Определение источника атаки

Определить источник DoS-атаки обычно довольно сложно, но возможно. Очень важно определить именно источник атаки, а не другие скомпрометированные узлы, которые использовались при атаке. Например, в случае со Smurf-атакой мнимыми источниками будут все узлы сети, которые «завалили» ваш сервер или маршрутизатор ping-пакетами, но ведь кто-то отправил же такой ping-пакет! Это и есть настоящий источник атаки. С другими видами DoS-атак ситуация примерно такая же. Сетевой червь может инфицировать целые подсети, которые по команде злоумышленника все вместе будут отправлять TCP/UDP-пакеты на ваш сервер. Опять-таки есть первоисточник атаки.

Что делать, если вы уже подверглись DoS-атаке? Первым делом все силы нужно направить на то, чтобы такая атака не повторилась. Да, нежелательный трафик вы уже обратно не вернете, и его придется оплатить. Тут уже ничего не поделать.

Нужно сообщить своему провайдеру об атаке - вместе вам будет проще отследить источник атаки. Провайдер, в свою очередь, может обратиться к вышестоящему провайдеру с целью обнаружения реального источника атаки. Помните, что любой пользователь, который подключается к Интернету, подписывает договор о том, что он не будет причинять вреда другим пользователям. Может рядовой пользователь такой договор и не подписывает, зато его подписывают провайдеры. То есть в крайнем случае можно подать в суд на провайдера, в сети которого находился злоумышленник. Защититься от DoS-атаки поможет правильно продуманная политика безопасности сети, правильно настроенные маршрутизаторы, брандмауэры, прокси-серверы (если все это используется в вашей сети). Довольно эффективным способом защиты от DoS-атаки является переключение на резервный канал (о котором не знает злоумышленник) в самом начале атаки - да, злоумышленника вы не накажете, но зато уменьшите материальный ущерб.

В любом случае о каждой атаке нужно официально (вплоть до письменного уведомления) оповещать своего провайдера, ведь ваша безопасность и в его интересах.

Защита сетевых служб

Основное правило при защите сетевых служб следующее: отключайте неиспользуемые сетевые службы. Нет ничего хуже включенной, но не настро-

енной сетевой службы. Это просто черная дыра в безопасности вашего сервера.

Пока службы отключены, они не представляют угрозу безопасности. Однако, включая их, вам, по следующим причинам, следует проявлять бдительность:

- По умолчанию файрвол не включен, поэтому если служба прослушивает все сетевые интерфейсы, они, фактически, становятся общедоступными.
- У некоторых служб нет учетных данных для аутентификации, они дают вам их установить при первом использовании. У некоторых есть стандартные учетные записи, их логины и пароли широко известны. Проверьте, чтобы пароли доступа к службам были заданы, либо изменены на те, которые знаете только вы.
- Многие службы работают под суперпользователем, с полными административными привилегиями, поэтому несанкционированный доступ к ним или дыры в системе их безопасности обычно приводят к серьезным последствиям.

Для каждой сетевой службы есть свои рекомендации по обеспечению безопасности, и вы без проблем найдете их в Интернете. Кроме выполнения этих рекомендаций настоятельно рекомендуется установить средство fail2ban, позволяющее защитить сразу несколько сетевых служб и даже защититься от DDoS-атак, которым удалось просочиться сквозь наш грозный брандмауэр, который мы настроили в прошлом разделе.

Для установки fail2ban введите команду:

```
sudo apt install fail2ban
```

Основной конфигурационный файл находится по пути /etc/fail2ban/jail.conf. Однако, его не рекомендуется менять и для настройки используют подключаемые файлы из каталога /etc/fail2ban/jail.d. Настройки по умолчанию хранятся в файле /etc/fail2ban/jail.d/default.conf. Рассмотрим секцию DEFAULT:

```
[DEFAULT]
maxretry = 4
findtime = 480
bantime = 720
action = firewallcmd-ipset
```

```
ignoreip = 127.0.0.1/8
```

где:

- **maxretry** — количество действий, которые разрешено совершить до бана;
- **findtime** — время в секундах, в течение которого учитывается maxretry;
- **bantime** — время, на которое будет блокироваться IP-адрес;
- **action** — действия, которое будет выполняться, если Fail2ban обнаружит активность, соответствующую критериям поиска;
- **ignoreip** — игнорировать защиту, если запросы приходят с перечисленных адресов.

В данном примере, если в течение 8 минут (480) будет найдено 5 строк (maxretry = 4), содержащих критерий фильтра, Fail2ban заблокирует IP-адрес, с которого идет подключение на 12 минут (720);

В секции [DEFAULT] хранятся общие настройки для всех правил. Каждую из настроек можно переопределить при конфигурировании самого правила.

Для настройки новых правил нужно создать новый конфигурационный файл в каталоге /etc/fail2ban/jail.d. Например, создадим файл ssh.conf, в котором будут правила для защиты сервиса ssh:

```
[ssh]
enabled = true
port    = ssh
filter = sshd
action = iptables[name=sshd, port=ssh, protocol=tcp]
   sendmail-whois[name=ssh, dest=****@gmail.com,
sender=fail2ban@myhost.ru]
logpath = /var/log/auth.log
maxretry = 3
bantime = 600
```

Внимание! Следите за тем, чтобы путь к файлу хранения логов (logpath) был указан корректно.

Примечание. Вы можете хранить конфигурацию защиты отдельного сервиса в отдельном файле, а можете собрать все приведенные далее секции в один файл – services.conf. Как вам будет удобнее.

Если выполнено более 3 неудачных попыток подключения к серверу через основные порты SSH, то IP-адрес, с которого выполнялась авторизация, попадет в бан на 10 минут (600 секунд). Правило запрета будет добавлено в **iptables**. В то же время владелец сервера получит уведомление на e-mail, указанный в значении переменной **dest**, о том, что указанный IP был заблокирован за попытку получения несанкционированного доступа по протоколу SSH. Также в сообщении будет указана WHOIS информация о заблокированном IP.

Для защиты от DDoS-атаки на SSH можно создать секцию:

```
[ssh-ddos]
enabled = true
port    = ssh
filter = sshd-ddos
logpath = /var/log/auth.log
maxretry = 2
```

Ниже представлены примеры настроек конфигурационного файла для защиты почтового сервера **postfix**.

```
[postfix]
enabled = true
port    = smtp,ssmtp,submission
action = iptables[name=Postfix-smtp, port=smtp,
protocol=tcp]
filter = postfix
logpath = /var/log/mail.log
bantime = 86400
maxretry = 3
findtime = 3600
ignoreip = 127.0.0.1
```

Для защиты веб-сервера Apache можно использовать следующие настройки Fail2ban:

```
[apache]
enabled = true
port    = http,https
filter = apache-auth
logpath = /var/log/apache2/error.log
```

```
maxretry = 3

[apache-multiport]
enabled = true
port    = http,https
filter  = apache-auth
logpath = /var/log/apache2/error.log
maxretry = 3

[apache-noscript]
enabled = true
port    = http,https
filter = apache-noscript
logpath = /var/log/apache2/error.log
maxretry = 3
```

Как вы уже могли заметить, в используемых выше секциях конфигурации отсутствуют значения параметра **action**. В этом случае при обнаружении атаки на сервис Apache программа Fail2ban будет выполнять действие, определенное в секции [DEFAULT], а именно action = iptables-multiport. Это значит, что атакующий IP-адрес будет заблокирован в iptables при помощи так называемого модуля multiports. Модуль **multiports** позволяет настроить правило сразу для диапазонов портов.

Для защиты FTP-сервера vsftpd с помощью Fail2ban можно использовать следующие параметры:

```
[vsftpd]
enabled = true
port    = ftp,ftp-data,ftps,ftps-data
filter = vsftpd
logpath = /var/log/vsftpd.log
action = iptables[name=VSFTPD, port=21, protocol=tcp]
bantime = 600
maxretry = 3
findtime = 1800
```

Не забудьте о необходимости перезапуска Fail2ban после каждого редактирования конфигурационного файла:

```
sudo systemctl restart fail2ban
```

27.4. Шифрование данных

В этом разделе мы поговорим о криптографической файловой системе eCryptfs и зашифруем каталог с данными (/opt/data).

Первым делом нужно установить утилиты eCryptfs. На данный момент у меня Debian, поэтому для их установки буду использовать apt-get:

```
sudo apt install ecryptfs-utils
```

Чтобы зашифровать каталог, нужно его подмонтировать, указав тип файловой системы ecryptfs:

```
sudo mount -t ecryptfs /opt/data /opt/data
```

Вывод будет таким (жирным выделено то, что нужно ввести вам):

```
Passphrase: <секретная фраза>
Select cipher:
 1) aes: blocksize = 16; min keysize = 16; max keysize = 32
(not loaded)
 2) blowfish: blocksize = 16; min keysize = 16; max keysize =
56 (not loaded)
 3) des3_ede: blocksize = 8; min keysize = 24; max keysize =
24 (not loaded)
 4) twofish: blocksize = 16; min keysize = 16; max keysize = 32
(not loaded)
 5) cast6: blocksize = 16; min keysize = 16; max keysize = 32
(not loaded)
 6) cast5: blocksize = 8; min keysize = 5; max keysize = 16
(not loaded)
Selection [aes]: просто нажмите Enter
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
```

Теперь разберемся, что произошло. Мы согласились на использование алгоритма по умолчанию (AES). Если считаете, что другой алгоритм лучше, можете выбрать его. Также мы отказались от шифрования имен файлов (Enable filename encryption): если что-то случится с зашифрованным каталогом, то разобраться, где и какой файл, будет сложно.

На данный момент /opt/data зашифрован. Осталось самое главное — проверить, а зашифрован ли на самом деле каталог? Попробуем скопировать в него любой файл из незашифрованной файловой системы:

```
cp /etc/motd /optta
```

Размонтируем зашифрованный каталог:

```
sudo umount /opt/data
```

Теперь пробуем прочитать /opt/data/motd:

```
cat /opt/data/motd
```

Если вы увидите всякого рода иероглифы и абракадабру, значит, шифрование работает.

27.5. Настройка VPN-сервера

Представим, что у нас есть торговая организация, представители которой «рассекают» по всей стране. Им часто приходится пользоваться публичными Wi-Fi сетями (например, в ресторанах и отелях) для передачи данных

в главный офис. Кто и как настраивал такие публичные сети – непонятно. Перехватывают ли они передаваемые пользователями данные – тоже непонятно, но риск такой есть и его нужно минимизировать. Для этого мы настроим VPN-сервер, подключившись к которому, пользователи смогут безопасно передавать данные в главный офис, не опасаясь их перехвата. При использовании VPN-сервера весь трафик будет проходить в зашифрованном виде, и вы можете не опасаться за его перехват на участке «точка-доступа => провайдер => офис».

Конечно, можно использовать один из множества уже готовых VPN-сервисов, и вы также получите шифрование трафика. Но здесь есть ряд «но»:

- Собственный VPN-сервер подконтролен вам и только вам. Что происходит с данными, которые вы передаете через сторонние сервисы – неизвестно. Да, трафик будет зашифрован, но внутри VPN-соединения данные на VPN-сервер будут отправлены в незашифрованном виде. Может, VPN-сервис хранит все передаваемые данные и потом они окажутся в ненужных руках? Некоторые VPN-сервисы прямо заявляют об этом, что, мол, ведется лог и по первому запросу правоохранительных органов вся активность пользователя будет передана им. Вы то ничего не делаете противозаконного, но беспокоит сам факт протоколирования всего, что вы передаете. А вдруг произойдет утечка данных?
- Ограничения по скорости – у VPN-сервиса очень много пользователей и они вынуждены ограничивать скорость соединения. Ваш собственный VPN будет работать явно быстрее.
- Стоимость – хороший VPN-сервис стоит 20-30$ в месяц, у вас в штате 30-50 торговых представителей. Экономию в месяц можете подчитать самостоятельно.
- Стабильность – 8 мая 2020 года перестал работать популярный сервис SecurityKISS. Представьте, что вам в срочном порядке нужно найти новый сервис, настроить 30-50 аккаунтов и т.д. В собственном сервере можно быть уверенным. Что же касается обеспечения его бесперебойной работы, то для начала хватит хорошего ИБП, а если нужна доступность 24/7, то никто не мешает арендовать виртуальный сервер и настроить его как VPN. В месяц это обойдется 2000-3000 рублей, что гораздо дешевле 600 – 1500$ затрат на VPN-сервис.

Далее будет описана настройка VPN в Ubuntu. Если вы будете арендовать виртуальный сервер, то на них часто используются старые версии Ubuntu

вроде 16.04, 17.04 – ничего страшного. Самая последняя версия вам не нужна, к тому же из ее репозитариев удалили много полезного ПО – разработчики готовятся переходить на снапы, поэтому настраивать VPN в 20.04 вам будет некомфортно.

27.5.1. Создание всех необходимых сертификатов и ключей

Создание VPN-сервера требует некоторого времени, хотя ничего сложного нет. Установите программное обеспечение:

```
sudo apt install openvpn easy-rsa
```

Первый пакет - это сам OpenVPN, а второй - easy-rsa - пакет, позволяющий построить собственный сервер сертификации.

Далее нужно настроить центр сертификации. OpenVPN использует TLS/SSL, поэтому нам нужны сертификаты для шифрования трафика между сервером и клиентом. Чтобы не покупать сертификаты (мы же хотим максимально сэкономить!), мы создадим собственный центр сертификации.

Скопируйте каталог easy-rds в домашний каталог командой make-cadir:

```
make-cadir ~/openvpn-ca
cd ~/openvpn-ca
```

Далее нужно редактировать файл **vars**, воспользуемся для этого редактором **nano**:

```
nano vars
```

Найдите следующие переменные и задайте свои значения. Данные переменные используются при создании сертификатов:

```
export KEY_COUNTRY="RU"
export KEY_PROVINCE=" "
export KEY_CITY="Moscow"
export KEY_ORG="ABC Co"
export KEY_EMAIL="admin@abc.com"
export KEY_OU=»SomeWorkgroup»
```

Также найдите и отредактируйте переменную KEY_NAME:

```
export KEY_NAME=»server»
```

Для простоты можно использовать просто «server» (или любую другую строку, но запомните, какую именно). Если вы будете использовать название, отличное от «server», тогда вам придется изменить некоторые команды, в которых встречается это название.

Приступим 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 удаляет имеющиеся ключи, а вторая запускает процесс создания ключа и сертификата корневого центра сертификации. Все значения уже указаны в файле vars, поэтому вам нужно будет только нажимать **Enter** для подтверждения выбора. На данный момент у нас есть собственный центр сертификации, который мы будем использовать для создания сертификата, ключа и файлов шифрования для сервера.

Приступим к созданию сертификата и ключа для сервера. Для создания ключей для сервера введите команду (замените server на свое значение, если вы его изменили в **vars**):

```
./build-key-server server
```

Процесс создания ключей очень прост - нажимайте **Enter** в ответ на предлагаемые значения. Значение challenge password задавать не нужно. В конце процесса нужно два раза ввести y - для подписи и для подтверждения создания сертификата (сертификат выдается на 10 лет, так что в ближайшие 10 лет вас оставят в покое и вам не нужно будет его обновлять):

```
Certificate is to be certified until Jul  10 11:14:12 2030
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 создать остальные файлы:

```
./build-dh
openvpn --genkey --secret keys/ta.key
```

Команда build-dh создает ключи протокола Диффи-Халлмана, вторая команда - генерирует подпись HMAC. В зависимости от мощности вашего сервера, эти команды могут работать несколько минут каждая. Так что сервер не завис, он работает.

Осталось создать сертификат и ключи для клиента и мы сможем наконец-то приступить к настройке самого сервера. Создать данные ключи можно, как на клиенте (а потом подписать полученный ключ центром сертификации на сервере), так и на сервере. Проще это делать на сервере.

Сейчас мы создадим ключ и сертификат для **одного** клиента. Клиентов у нас несколько, поэтому данный процесс нужно будет повторить для каждого из них. Можете оформить последовательность действий в bash-сценарий для экономии времени.

Команда build-key используется 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
```

27.5.2. Настройка сервера

Когда все сертификаты и ключи сгенерированы, можно приступить к настройке сервера. Первым делом скопируйте сгенерированные ранее файлы из каталога openvpn-ca/keys в /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 /etc/openvpn/server.conf.

После того, как распакуете шаблон файла конфигурации можно приступить к его редактированию. Откройте /etc/openvpn/server.conf в любимом текстовом редакторе.

Далее приведен фрагмент этого файла. Внимательно читайте комментарии:

```
# Раскомментируйте эту строку
tls-auth ta.key 0
# Установите key-direction в 0
key-direction 0
# Расскомментируйте 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 раскомм.
# 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. Разрешите пересылать трафик, если вы этого еще не сделали. Откройте файл sysctl.conf:

```
sudo mcedit /etc/sysctl.conf
```

Раскомментируйте строчку:

```
net.ipv4.ip_forward=1
```

Чтобы изменения вступили в силу, введите команду:

```
sudo sysctl -p
```

Нам осталось только настроить брандмауэр и можно запускать VPN-сервер. Будем считать, что используется брандмауэр UFW (в современных дистрибутивах используется вместо iptables). Вы должны знать имя публичного интерфейса, пусть это будет ens33 - для примера, в вашем случае имя публичного интерфейса будет отличаться. Выяснить чего можно командой:

```
ip route | grep default
```

Данное название нужно добавить в файл /etc/ufw/before.rules. В самое начало этого файла нужно добавить строки (также укажите IP-адрес и маску вашей подсети):

```
# START OPENVPN RULES
# NAT table rules
*nat
:POSTROUTING ACCEPT [0:0]
# Allow traffic from OpenVPN client to ens33
-A POSTROUTING -s 192.168.0.0/24 -o ens33 -j MASQUERADE
COMMIT
# END OPENVPN RULES
```

Вместо ens33 нужно указать имя вашего публичного интерфейса. Теперь откройте файл nano /etc/default/ufw и найдите 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 используете 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 2020-07-10 13:16 0:05
EDT; 25s ago
```

Если все нормально, тогда обеспечим автоматический запуск сервера:

```
sudo systemctl enable openvpn@server
```

Теперь готовимся встречать клиентов. Прежде, чем клиенты смогут подключиться, нужно позаботиться об инфраструктуре настройки клиентов. Создадим каталог для хранения файлов:

```
mkdir -p ~/clients/files
chmod 700 ~/clients/files
```

Такие права доступа нужны, поскольку данный каталог будет содержать ключи клиентов.

Далее установим базовую конфигурацию:

```
cd /usr/share/doc/openvpn/examples/sample-config-files/
cp client.conf ~/clients/base.conf
```

Откройте файл ~/clients/base.conf. В нем нужно сделать несколько изменений:

```
# Укажите IP-адрес сервера и порт (1193 для UDP или 443
# для TCP)
remote IP-адрес порт
# Укажите протокол udp или tcp
proto протокол
# Раскомментируйте директивы
user nobody
```

```
group nogroup
# Найдите директивы ca, cert и key. Закомментируйте их
#ca ca.crt
#cert client.crt
#key client.key
# Добавьте параметры cipher и auth так, как они описаны
# в server.conf
cipher AES-128-CBC
auth SHA256
# Установите key-direction в 1
key-direction 1
```

Теперь создадим сценарий генерации файлов конфигурации (листинг 27.1):

```
cd ~/clients
touch make_config
chmod +x make_config
mcedit make_config
```

Листинг 27.1. 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 ~/clients/files вы найдете файл user1.ovpn.

27.5.3. Подключаем клиентов

Передайте файлы конфигурации (.ovpn) клиентам. Можете отправить по электронной почте вместе со следующей инструкцией.

Сначала рассмотрим настройку клиента в Linux. Установите 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
```

Если в вашем дистрибутиве нет файла /etc/openvpn/update-resolv-conf, то делать ничего не нужно!

Теперь подключитесь к VPN-серверу:

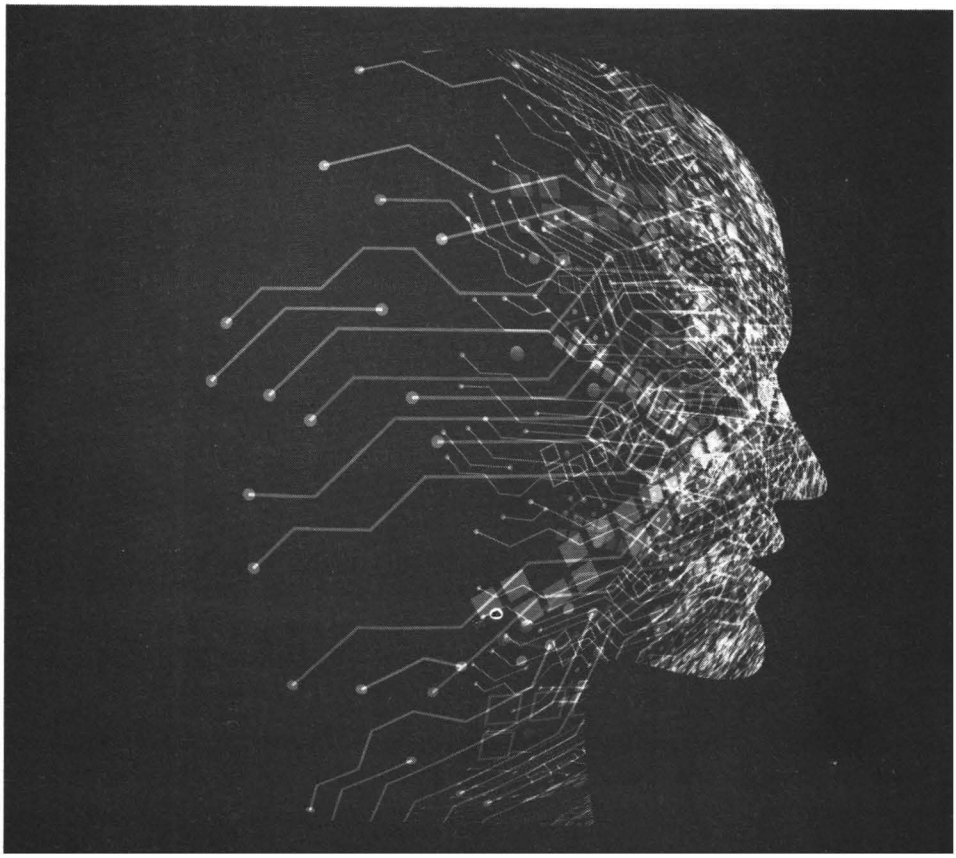
```
sudo openvpn --config user1.ovpn
```

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 странички проекта https://openvpn.net/index.php/open-source/downloads.html.

После запуска OpenVPN он должен автоматически увидеть ваш профиль. Щелкните на пиктограмме клиента на панели быстрого запуска правой кнопкой мыши и выберите команду **Подключиться**. Все достаточно просто.

Приложение 1.

Командный интерпретатор bash



В этом приложении мы рассмотрим командный интерпретатор **bash**. Это больше, чем просто средство для ввода команд. У него есть свои параметры и свои команды. Да, именно команды. На bash можно написать сценарии (подобно тому, как в MS DOS создавались BAT-файлы). Возможности bash подобны языку программирования – есть управляющие структуры, циклы и т.д. Обо всем этом мы поговорим в данной главе.

ПР1.1. Настройка bash

bash – это самая популярная 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.

Интерпретатор bash – не единственная оболочка, доступная в вашей системе. Список доступных оболочек доступен в файле /etc/shells. В большинстве случаев по умолчанию используется bash.

Файл /etc/profile содержит глобальные настройки bash, он влияет на всю систему – на каждую запущенную оболочку. Обычно /etc/profile не нуждается в изменении, а при необходимости изменить параметры bash редактируют один из файлов:

- ~/.bash_profile – обрабатывается при каждом входе в систему;
- ~/.bashrc – обрабатывается при каждом запуске дочерней оболочки;
- ~/.bash_logout – обрабатывается при выходе из системы.

Файл ~/.bash_profile часто не существует, а если и существует, то в нем есть всего одна строка:

```
source ~/.bashrc
```

Данная строка означает, что нужно прочитать файл .bashrc. Поэтому будем считать основным конфигурационным файлом файл .bashrc. Но помните, что он влияет на оболочку текущего пользователя (такой файл находится в домашнем каталоге каждого пользователя, не забываем: «~» означает домашний каталог). Если же вдруг понадобится задать параметры, которые повлияют на всех пользователей, то нужно редактировать файл /etc/profile.

Файл .bash_history (тоже находится в домашнем каталоге) содержит историю команд, введенных пользователем. Здесь вы можете просмотреть свои же команды, которые вы накануне вводили.

Конфигурация bash хранится в файле .bashrc. Обычно в этом файле задаются псевдонимы команд, определяется внешний вид приглашения командной строки, задаются значения переменных окружения.

Псевдонимы команд задаются с помощью команды alias, вот несколько примеров:

```
alias h='fc -l'
alias ll='ls -laFo'
alias l='ls -l'
alias g='egrep -i'
```

Псевдонимы работают просто: при вводе команды l на самом деле будет выполнена команда ls -l.

Теперь рассмотрим пример изменения приглашения командной строки. Глобальная переменная PS1 отвечает за внешний вид командной строки. По умолчанию командная строка имеет формат:

```
пользователь@компьютер:рабочий_каталог
```

Значение PS1 при этом будет:

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

В табл. ПР1.1 приведены допустимые модификаторы командной строки.

Таблица ПР1.1. Модификаторы командной строки

Модификатор	Описание
\a	ASCII-символ звонка (код 07). Не рекомендуется его использовать — очень скоро начнет раздражать
\d	Дата в формате «день недели, месяц, число»
\h	Имя компьютера до первой точки
\H	Полное имя компьютера
\j	Количество задач, запущенных в оболочке в данное время
\l	Название терминала
\n	Символ новой строки
\r	Возврат каретки
\s	Название оболочки
\t	Время в 24-часовом формате (ЧЧ: ММ: СС)
\T	Время в 12-часовом формате (ЧЧ: ММ: СС)
\@	Время в 12-часовом формате (AM/PM)
\u	Имя пользователя
\v	Версия bash (сокращенный вариант)
\V	Версия bash (полная версия: номер релиза, номер патча)
\w	Текущий каталог (полный путь)
\W	Текущий каталог (только название каталога, без пути)
\!	Номер команды в истории
\#	Системный номер команды

\$	Если UID пользователя равен 0, будет выведен символ #, иначе — символ $
\\	Обратный слэш
$( )	Подстановка внешней команды

Вот пример альтернативного приглашения командной строки:

```
PS1='[\u@\h] $(date +%m/%d/%y) \$'
```

Вид приглашения будет такой:

```
[denis@hosting] 08/04/19 $
```

В квадратных скобках выводится имя пользователя и имя компьютера, затем используется конструкция $() для подстановки даты в нужном нам формате и символ приглашения, который изменяется в зависимости от идентификатора пользователя.

Установить переменную окружения можно с помощью команды export, что будет показано позже.

ПР1.2. Зачем нужны сценарии 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 нам нужно записать все эти три файла на DVD с помощью любой программы для прожига DVD.

Если выполнять данную операцию раз в месяц (или хотя бы раз в неделю), то ничего страшного. Но представьте, что вам нужно делать это каждый день или даже несколько раз в день? Думаю, такая рутинная работа вам быстро

надоест. А ведь можно написать сценарий, который сам будет создавать резервные копии и записывать их на DVD! Все, что вам нужно, — это вставить чистый DVD перед запуском сценария.

Можно пойти и иным путем: написать сценарий, который будет делать резервные копии системных каталогов и записывать их на другой раздел жесткого диска. Ведь не секрет, что резервные копии делаются не только на случай сбоя системы, но и для защиты от некорректного изменения данных пользователем. Помню, удалил важную тему форума и попросил своего хостинг-провайдера сделать откат. Я 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.

ПР1.3. Сценарий Привет, мир!

По традиции напишем первый сценарий, выводящий всем известную фразу: Hello, world!. Для редактирования сценариев вы можете использовать любимый текстовый редактор, например, **nano** или **ee** (листинг ПР1.1).

Листинг ПР1.1. 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 **#** и **!** окажется пробел, то данная директива не сработает, поскольку будет воспринята как обычный комментарий. Комментарии начинаются, как вы уже догадались, с решетки:

```
# Комментарий
```

Вторая строка — это оператор **echo**, выводящий нашу строку. Сохраните сценарий под именем hello и введите команду:

```
$ chmod +x hello
```

Для запуска сценария введите команду:

```
./hello
```

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
```

ПР1.4. Переменные в 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=firma.ru
echo $ADDRESS
```

При объявлении переменной знак доллара не ставится, но он обязателен при использовании переменной. Также 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'
```

Чтение значения переменной с клавиатуры осуществляется с помощью инструкции **read**. При этом указывать символ доллара не нужно. Вторая 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:

- BASH — полный путь до исполняемого файла командной оболочки bssh;
- BASH_VERSION — версия bash;
- HOME — домашний каталог пользователя, который запустил сценарий;
- HOSTNAME — имя компьютера;
- RANDOM — случайное число в диапазоне от 0 до 32 767;
- OSTYPE — тип операционной системы;
- PWD — текущий каталог;
- PS1 — строка приглашения;
- UID — ID пользователя, который запустил сценарий;
- USER — имя пользователя.

Для установки собственной переменной окружения используется команда **export**:

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

ПР1.5. Передаем параметры сценарию

Очень часто сценариям нужно передавать различные параметры, например, режим работы или имя 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 ПР1.2).

Листинг ПР1.2. Получение параметров сценария

```
# Сценарий должен вызываться так:
# имя_сценария параметр

# Анализируем первый параметр
case «$1» in
  start)
    # Действия при получении параметра start
    echo «Запускаемся...»
    ;;
  stop)
    # Действия при получении параметра stop
    echo «Завершаем работу...»
    ;;
*)
    # Действия в остальных случаях
    # Выводим подсказку о том, как нужно использовать
сценарий, и
    # завершаем работу сценария
    echo «Использовать: $0 {start|stop }»
    exit 1
    ;;
esac
```

Думаю, приведенных комментариев достаточно, поэтому подробно рассматривать работу сценария из листинга ПР1.2 не будем.

ПР1.6. Обработка массивов

Интерпретатор **bash** позволяет использовать массивы. Массивы объявляются подобно переменным.

Вот пример объявления массива:

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

Выводим элемент массива:

```
echo $ARRAY[0]
```

ПР1.7. Циклы *for* и *while*

Как и в любом языке программирования, в bash можно использовать циклы. Мы рассмотрим циклы **for** и **while**, хотя вообще 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. нам нужно вывести 1, 2 и 3, но с помощью while, а не for:

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

ПР1.8. Условные операторы

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

```
if условие_1 then
    команды_1
elif условие_2 then
    команды_2
...
elif условие_N then
    команды_N
```

```
else
    команды_N+1
fi
```

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

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

```
# Переменная N = 10
[ N==10 ]

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

Операции сравнения указываются не с помощью привычных знаков > или <, а с помощью следующих выражений:

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

Применять данные 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 исполняемым.

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

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

Значение указанной переменной по очереди сравнивается с приведенными значениями (значение_1, ..., значение_N). Если есть совпадение, то будут выполнены команды, соответствующие значению. Если совпадений нет, то будут выполнены команды по умолчанию. Пример использования **case** был приведен в листинге ПР1.2.

ПР1.9. Функции в bash

В bash можно использовать функции. Синтаксис объявления функции следующий:

```
имя() { список; }
```

Вот пример объявления и использования функции:

```
list_txt()
{
echo "Text files in current directory:"
ls *.txt
}
```

ПР1.10. Практические примеры сценариев

Проверка прав пользователя

Для сценариев, требующих полномочий root, сначала нужно проверить, какой пользователь запустил сценарий. UID пользователя root всегда равен 0. Проверка, является ли пользователь, запустивший сценарий, пользователем root, может выглядеть так, как показано в листинге ПР1.3.

Листинг ПР1.3. Проверка полномочий пользователя

```
#!/bin/bash

ROOT_UID=0

if [ "$UID" -eq "$ROOT_UID" ]
then
  echo «Root»
else
  echo «Обычный пользователь»
fi

exit 0
```

Проверка свободного дискового пространства с уведомлением по e-mail

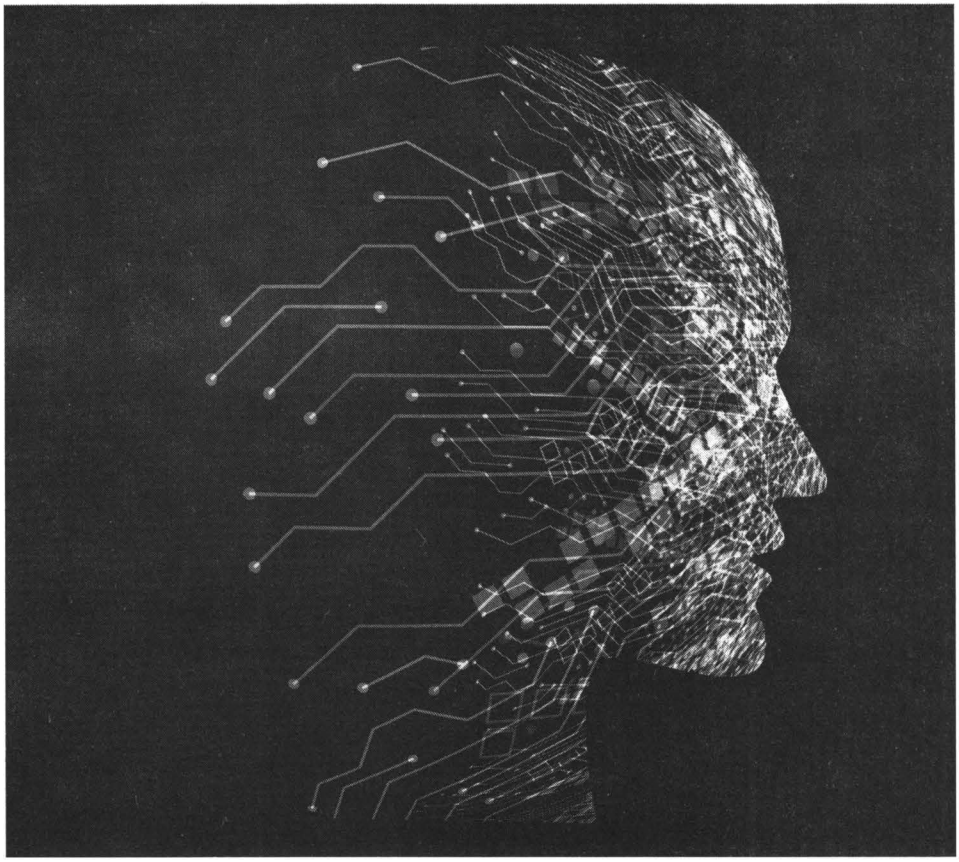
Следующий сценарий проверяет свободное дисковое пространство сервера и отправляет уведомление по e-mail администратору, если осталось меньше 2000 Мбайт дискового пространства. Сценарий целесообразно запускать через планировщик заданий (например, через cron) с заданной периодичностью (например, каждый час). Код сценария приведен в листинге ПР1.4.

Листинг ПР1.4. Проверка свободного дискового пространства

```
#!/bin/bash
# В переменную freespace будет записано свободное
пространство
# на контролируемом диске - /dev/sda1 (в Мбайт)
freespace=`df -m | grep "/dev/sda1" | awk '{print $4}'`

# Если места на диске < 2000 Mb, то отправляем письмо
администратору
if [ $freespace -lt 2000 ];
then
echo «На сервера меньше < 2000 MB» | mail -s «Свободное
место заканчивается!» admin@firma.ru
fi
```

Приложение 2.

Сетевая файловая система NFS



ПР2.1. Вкратце о NFS и установка необходимых пакетов

Сетевая файловая система (NFS, Network File System) - довольно старый и распространеный способ предоставления доступа к общим файлам. NFS позволяет монтировать файловые системы, которые физически находятся на других компьютерах в вашей локальной сети.

Принцип работы NFS прост: на сервере устанавливается демон nfsd, который экспортирует файловые системы. NFS-клиент монтирует экспортируемые файловые системы. После монтирования с удаленной файловой системой можно работать, как с локальной - для пользователя ничего не изменится, разве что скорость доступа к файлам будет чуть ниже, поскольку данные ведь передаются по сети.

В вашей сети может быть несколько NFS-серверов и несколько NFS-клиентов. По сути, каждый NFS-клиент может выступать и в роли NFS-сервера, но обычно NFS используется иначе. Выделяется один NFS-сервер, который будет экспортировать общую файловую систему, которая может использоваться для хранения общих файлов - довольно частый сценарий.

Чтобы установить NFS-сервер и/или NFS-клиент произведите поиск пакета по слову 'nfs'. Далее прочитайте описания пакетов и установите необходимые вам пакеты. Названия пакетов могут отличаться в зависимости от дистрибутива. Например, в openSUSE на сервере нужно установить следующие пакеты:

- nfs-kernel-server - содержит NFS-сервер;
- nfswatch - утилита мониторинга NFS-трафика (можно не устанавливать);
- quota-nfs - система дисковых квот для NFS.

На клиенте нужно установить только пакет nfs-client.

ПР2.2. Файл /etc/exports

В файле /etc/exports прописываются экспортируемые файловые системы. Эти экспортируемые файловые системы могут монтировать клиенты. Общий формат записи в файле exports следующий:

```
файловая_система [компьютер] (опции)
```

Вот типичный пример:

```
/home/public (rw)
/home/templates (ro)
```

В данном случае каталог /home/public сервера будет доступен NFS-клиентам для чтения и записи (rw), а каталог /home/templates - только для чтения (ro). При желании можно еще и указать узлы, которым будет доступен экспортируемый каталог, например:

```
/home/share1 192.168.1.100 (rw)
/home/share2 host.example.com (rw)
/home/share3 *.example.com (rw)
```

Здесь первый каталог (/home/share) будет доступен только узлу с IP-адресом 192.168.1.100, второй каталог - только узлу с доменным именем

host.example.com, а третий - всем узлам из домена example.com. Также можно указать целую IP-сеть и ее маску, например:

```
/home/share4 192.168.1.0/28 (rw)
```

В данном случае получить доступ к /home/share4 смогут лишь первые 16 IP-адресов - от 192.168.1.0 до 192.168.1.15. Компьютеры с IP-адресами 192.168.1.16 и выше не смогут использовать эту файловую систему.

Обычно можно обойтись только опциями ro, rw и noaccess. Последняя используется для запрещения доступа к файловой системе. Например, вы хотите запретить узлу 192.168.1.5 доступ к файловой системе:

```
/home/share4 192.168.1.5 (noaccess).
```

Остальные опции NFS приведены в таблице ПР2.1.

Таблица ПР2.1. Опции экспорта файловой системы в NFS

Опция	Описание
all_squash	Преобразует идентификаторы групп и пользователей в анонимные
insecure	Разрешает запросы с любых портов, см. secure
link_absolute	Не изменяет символические ссылки (по умолчанию)
link_relative	Абсолютные ссылки будут преобразованы в относительные
root_squash	Преобразует все запросы от root в запросы от анонимного пользователя
no_root_squash	Разрешает доступ к файловой системе от имени root. Не рекомендуется
secure	Принимает запросы на монтирование файловой системы только с портов с номерами < 1024. Такие порты может создавать только root, поэтому такое соединение считается безопасным. Используется по умолчанию

Колисниченко Денис Николаевич

LINUX

Полное руководство
по работе и администрированию

Группа подготовки издания:

Зав. редакцией компьютерной литературы: *М. В. Финков*

Редактор: *Е. В. Финков*

Корректор: *А. В. Громова*

12+

ООО «Наука и Техника»
Лицензия №000350 от 23 декабря 1999 года.
192029, г. Санкт-Петербург, пр.Обуховской обороны, д. 107.
Подписано в печать 07.09.2020. Формат 70x100 1/16.
Бумага офсетная. Печать офсетная. Объем 30 п. л.
Тираж 1350. Заказ 7713.

Отпечатано с готовых файлов заказчика
в АО «Первая Образцовая типография»,
филиал «УЛЬЯНОВСКИЙ ДОМ ПЕЧАТИ»
432980, Россия, г. Ульяновск, ул. Гончарова, 14

LINUX

Полное руководство по работе и администрированию

Linux в наше время весьма популярен как у обычных пользователей, так и у крупных корпораций – таких как Microsoft, IBM и т.д. Эта книга содержит в себе как теоретические, так и практические материалы, т.е. теория и практика будут объединены в одно целое - не будет отдельных больших и скучных теоретических глав.

Книгу можно разделить на четыре части. В первой части мы поговорим об установке системы, рассмотрим вход и завершение работы, выполним кое-какие действия по настройке системы, рассмотрим основы командной строки;

Вторая часть посвящена настройкам Интернета, установке программного обеспечения, и обзору популярных программ для Linux – вы узнаете не только, как устанавливать программы, но и какую программу установить, что не менее важно. В третьей части будет подробно рассмотрено локальное администрирование в Linux: управление файловыми системами; загрузка операционной системы; системные процессы и основные группы пользователей;

Ну и, конечно, какой Linux без сервера? Четвертая часть посвящена вопросам администрирования Linux-сервера в локальной сети – будет показано, как настроить серверы DNS, SSH, DHCP, FTP; поговорим об интеграции сервера в Windows-сеть; о безопасности сервера; а также настроим брандмауэр и научимся защищать сервер от сетевых атак.

Книга будет полезна для любого уровня читателей – как для тех, кто только заинтересовался Линуксом, так и для тех, кто хочет расширить свои навыки использования этой операционной системы. Каждый найдет здесь для себя что-то полезное и востребованное! Важно, что одним из дистрибутивов (наряду с Ubuntu), на котором показывается работа в Linux, выбран российский Astra Linux, сертифицированный и рекомендованный к использованию на территории России.

ISBN 978-5-94387-608-0

9 78- 5- 94387- 608- 0

Издательство "Наука и Техника"
г. Санкт-Петербург

Для заказа книг:
(812) 412-70-26
e-mail: nitmail@nit.com.ru
www.nit.com.ru

НиТ издательство

www.nit.com.ru