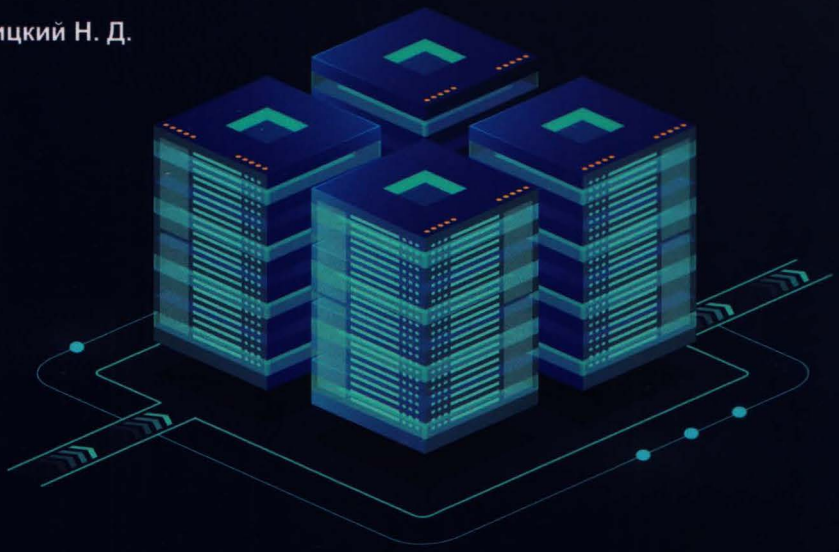
Левицкий Н. Д.



УДАЛЕННЫЙ СЕРВЕР
СВОИМИ РУКАМИ

От азов создания до практической работы

- **Полный стек** технологий для настройки и развертывания выделенного сервера
- Создайте свой **zoom!**
- **Собственный** выделенный сервер на базе Linux
- Создайте свой **независимый** хостинг!

НиТ издательство

Левицкий Н. Д.

УДАЛЕННЫЙ СЕРВЕР
СВОИМИ РУКАМИ
От азов создания до практической работы

"Наука и Техника"

Санкт-Петербург

УДК 004.42
ББК 32.973

Левицкий Н. Д.

Удаленный сервер своими руками. От азов создания до практической работы.— СПб.: Наука и Техника, 2021. — 400 с., ил.

ISBN 978-5-94387-568-7

Эта книга поможет вам самостоятельно освоить полный цикл создания своего собственного выделенного сервера: от установки операционной системы (для удобства это будет Linux) на сервер до настройки и администрирования. Будет подробно рассказано: как выполнить первоначальную настройку сервера; как использовать командную строку; вы узнаете как настраивать сетевые интерфейсы сервера; поговорим о маршрутизации и настройке брандмауэра; как пользоваться удаленным входом в систему по ssh; как настаивать файловый сервер FTP; что такое DHCP-сервер и как подключаться к Windows-инфраструктуре; как повысить производительность сервера и многое другое.

Отдельно внимание будет уделено вопросам обеспечения безопасности вашего сервера: локальная безопасность, защита от сетевых атак, шифрование данных, защита сетевых служб и т.д.

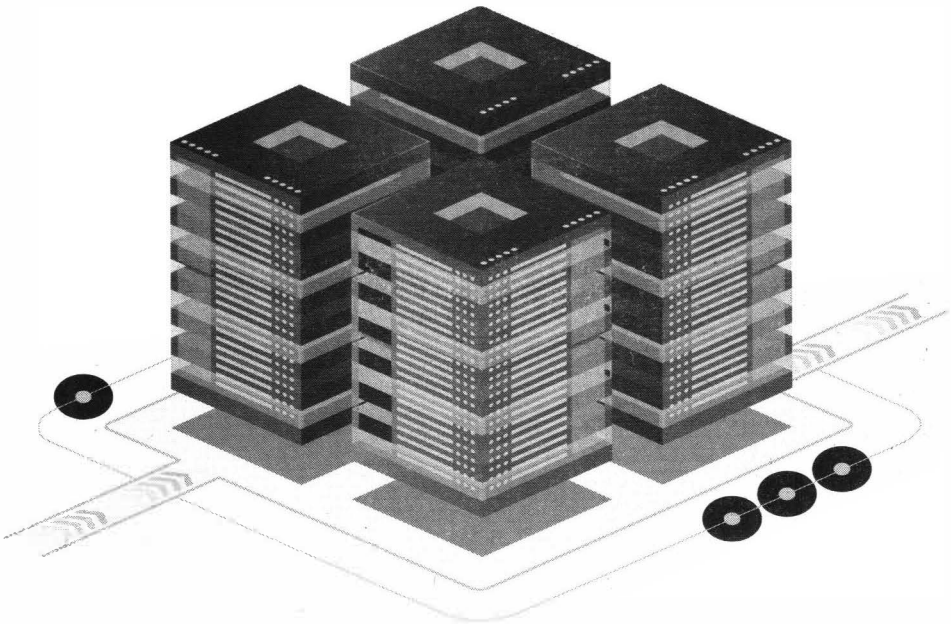
Также будет рассмотрена настройка веб-сервера и его администрирование: подключение ssl-сертификата, выбор и установка панели управления сервером; настройка почты на веб-сервере и т.п. Отдельно будет рассказано как создавать свой zoom — сервер видеоконференций!

Книга будет полезна как опытным, так и начинающим администраторам, поскольку изложение будет вестись от самых азов до более сложных тем, и каждый сможет найти то, что ему нужно!

ISBN 978-5-94387-568-7

Контактные телефоны издательства:
(812) 412 70 26
Официальный сайт: www.nit.com.ru

Содержание

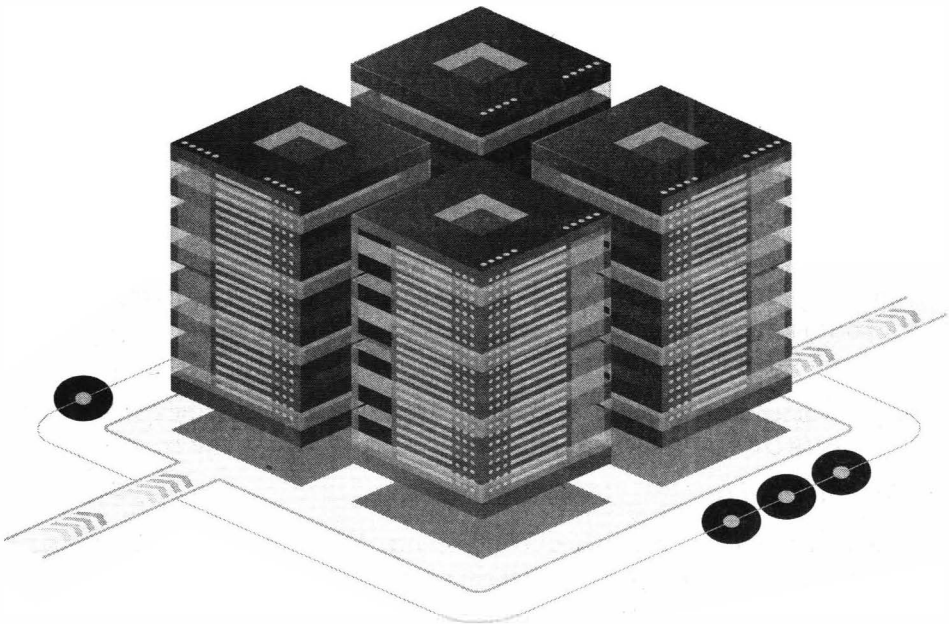


Часть I.

Устанавливаем и настраиваем Linux на сервере

Глава 1.

Установка системы на сервер



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.

1.1. 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.

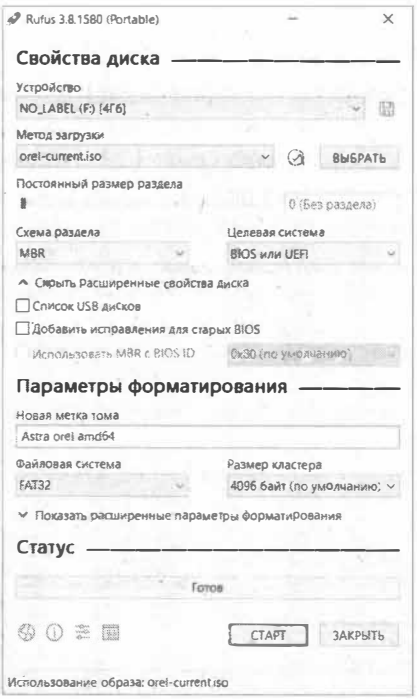


Рис. 1.1. Запись ISO образа с Linux на флешку

1.2. 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.

1.3. 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. 1.2). Если вы не успеете это сделать, то Ubuntu будет запущена в режиме LiveCD на английском языке – не волнуйтесь язык легко изменить при установке системы.

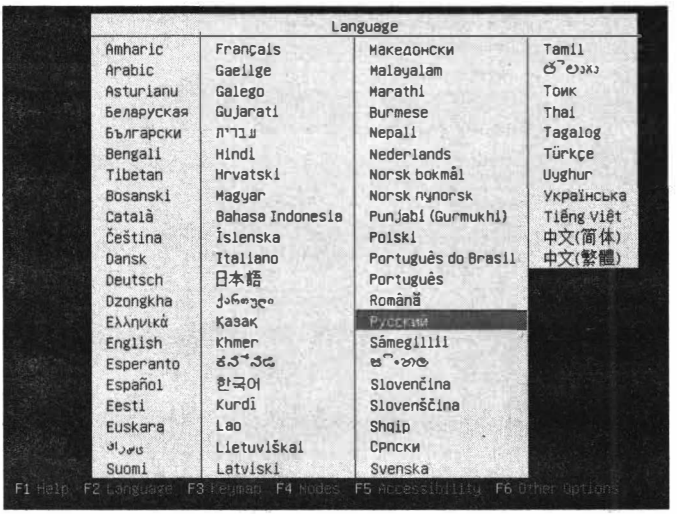


Рис. 1.2. Выбор языка при установке Ubuntu

Если вы успели нажать любую клавишу, вы увидите меню загрузчика (рис. 1.3). Назначение команд загрузчика понятно и в особых комментариях не

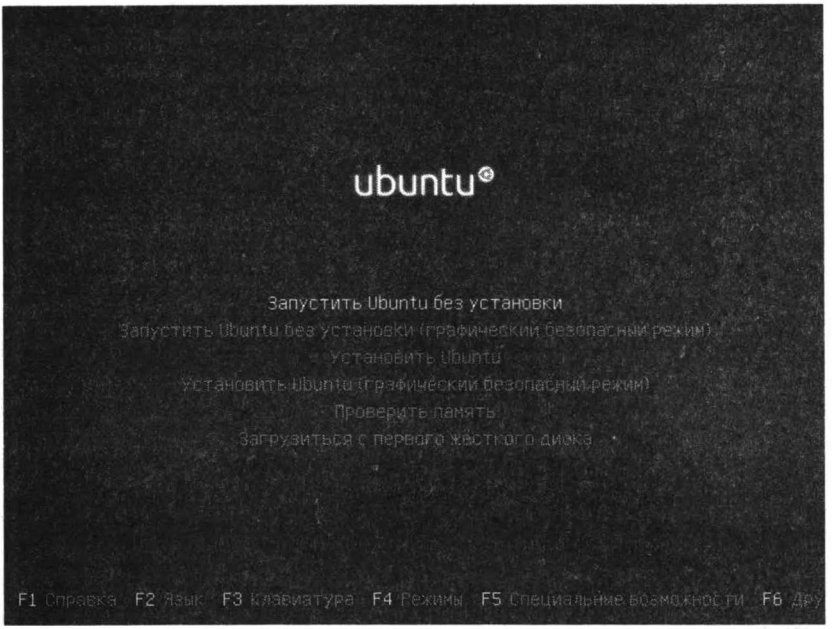


Рис. 1.3. Меню загрузчика Ubuntu

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. 1.4. 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. 1.5). Для запуска установки дважды щелкните по значку **Установить Ubuntu 20.04** на рабочем столе.

Рис. 1.5. Ubuntu в режиме LiveCD

1.4. 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

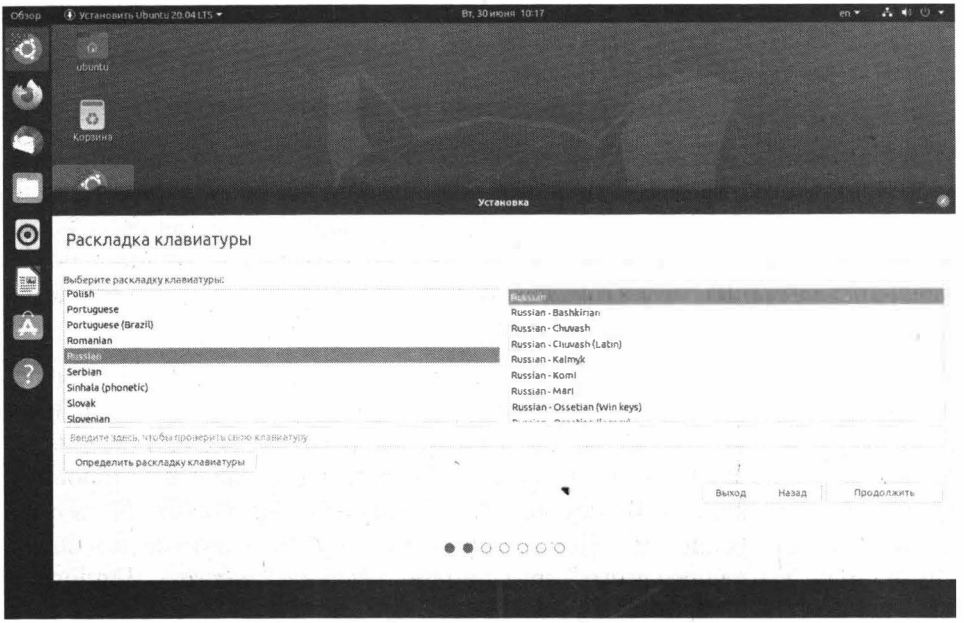


Рис. 1.6. Параметры клавиатуры (Ubuntu)

Ubuntu, в некоторых – нет. Например, после нажатия **Продолжить** в обоих случаях инсталляторы предложат выбрать раскладку клавиатуры (рис. 1.6, 1.7). 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. 1.7. Параметры клавиатуры (Astra Linux)

1.5. Разметка диска

1.5.1. 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), 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.

1.5.2. 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.

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.

1.5.3. 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.

1.5.4. 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.

1.5.5. Ручная разметка в Ubuntu

Если вы не планируете использовать целевой компьютер для других операционных систем, можете выбрать вариант **Стереть диск и установить Ubuntu** (рис. 1.8). Идеальный выбор для нового домашнего/персонального компьютера.

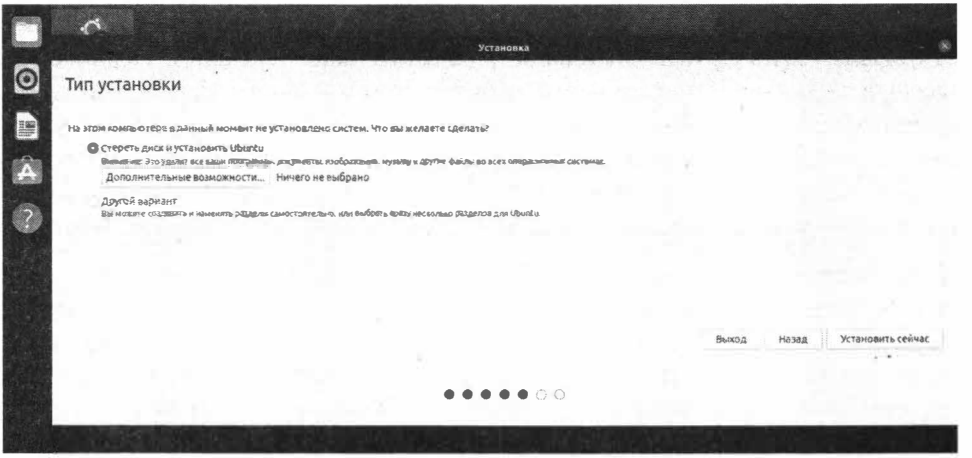


Рис. 1.8. Тип установки

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. 1.9 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.

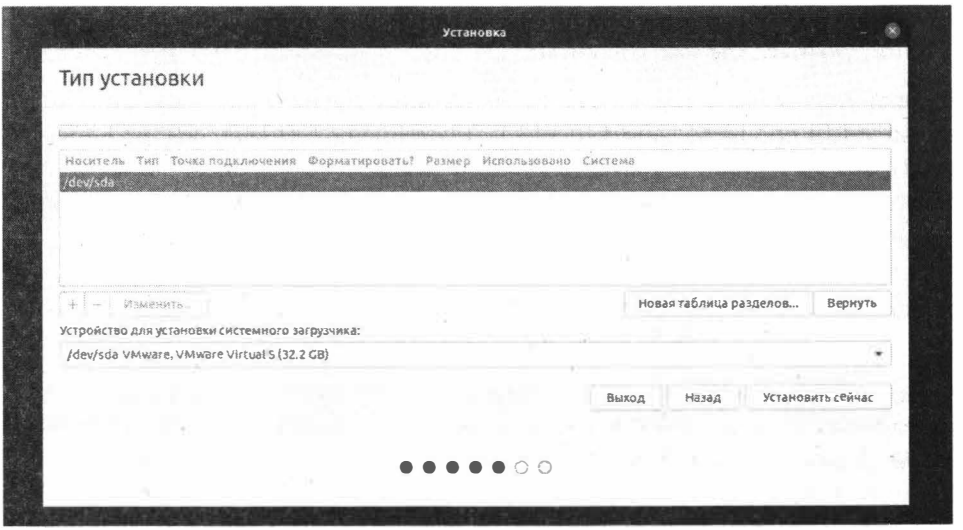


Рис. 1.9. Начало ручной разметки диска

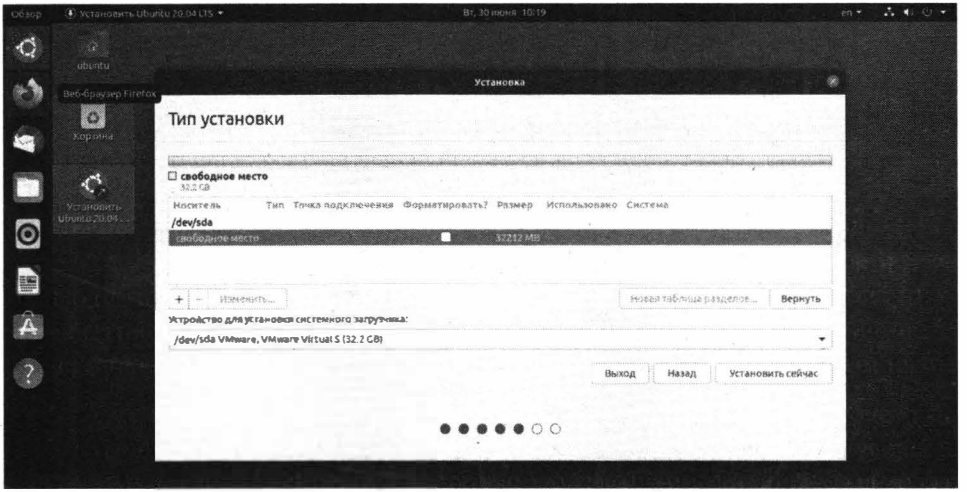


Рис. 1.10. После создания таблицы разделов

После этого вы увидите, что у вас появилось свободное место (рис. 1.10). Нажмите кнопку + для создания нового раздела. Введите размер создаваемого раздела, выберите точку монтирования и файловую систему (рис. 1.11).

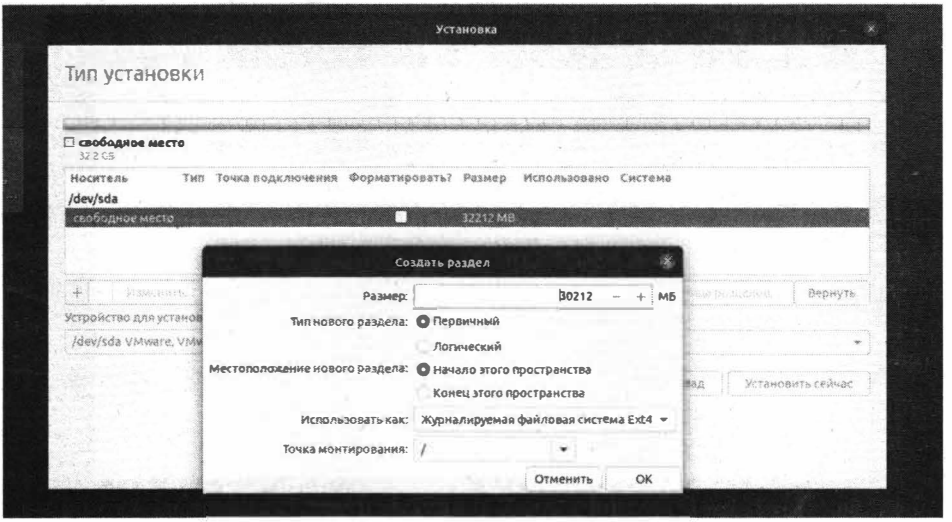


Рис. 1.11. Создание нового раздела

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. 1.12.

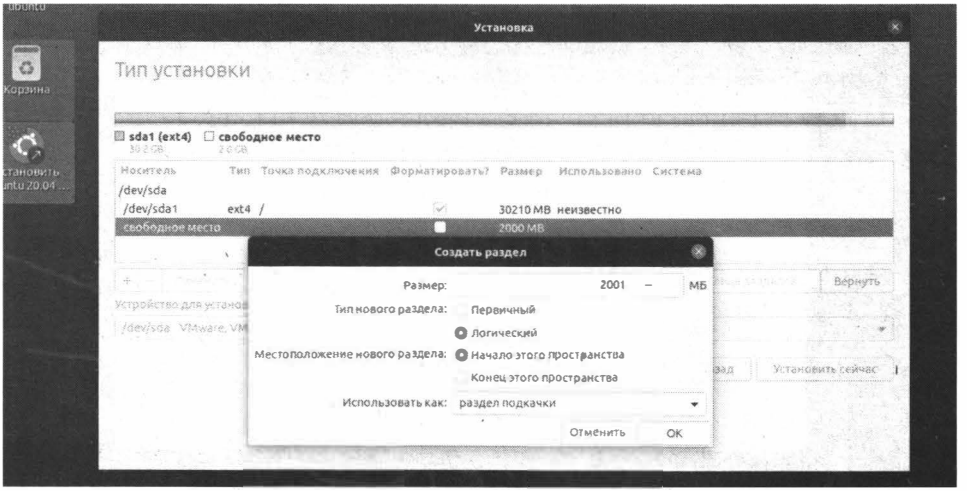


Рис. 1.12. Создание раздела подкачки

Созданная схема разметки изображена на рис. 1.13. Нажмите кнопку **Установить сейчас** для начала установки системы.

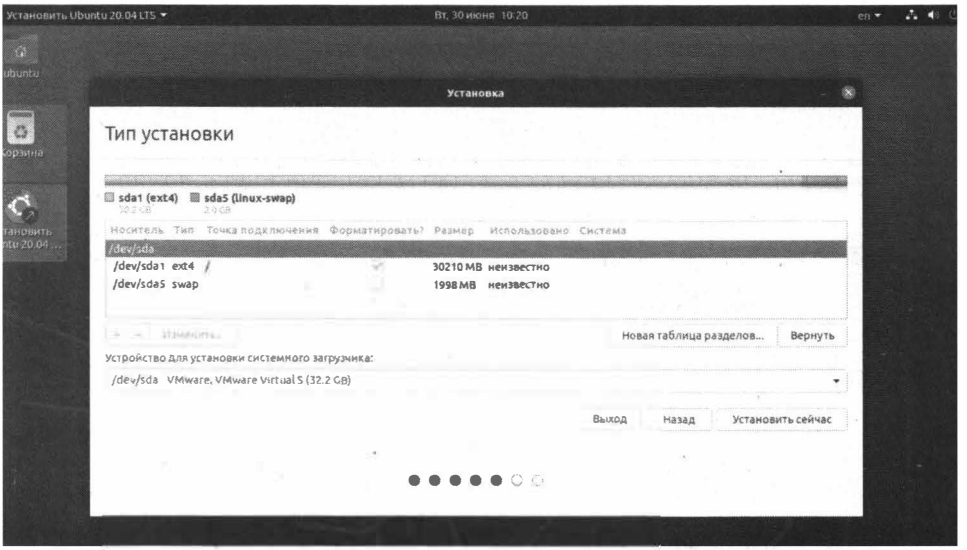


Рис. 1.13. Разметка завершена

1.5.6. 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!

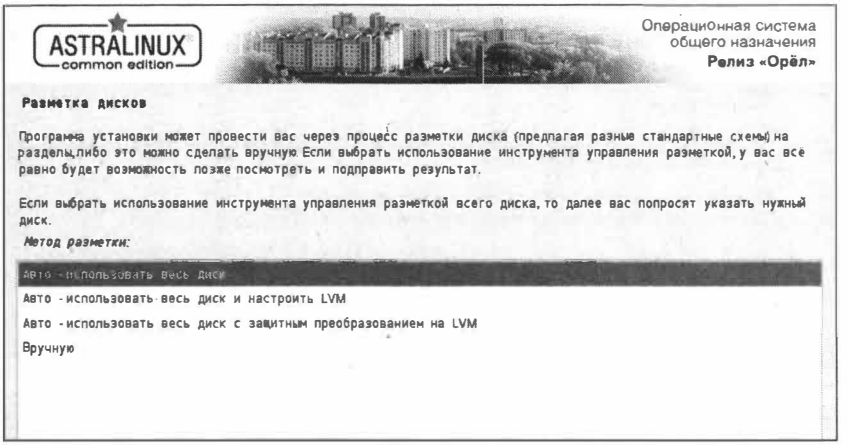


Рис. 1.14. Разметка в Astra Linux

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. 1.14)

В противном случае выбираем **Вручную**. Далее выбираем ваш жесткий диск и нажимаем кнопку **Продолжить** (рис. 1.15).

Рис. 1.15. Выберите жесткий диск и нажмите кнопку "Продолжить"

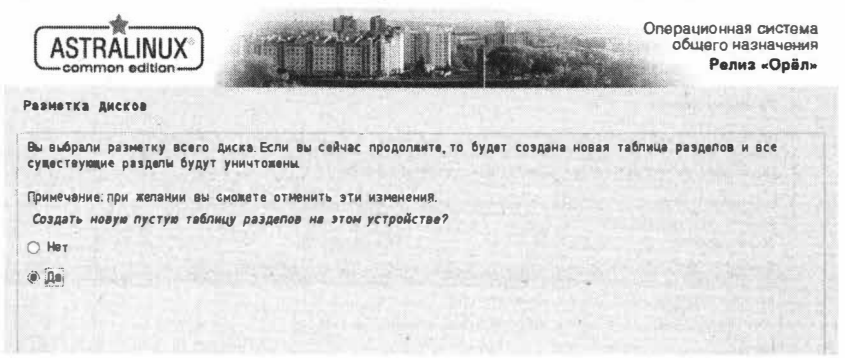


Рис. 1.16. Новая таблица разделов

Создайте новую таблицу разделов (рис. 1.16). Затем выделите свободное место (как показано на рис. 1.17) и снова нажмите кнопку **Продолжить**.

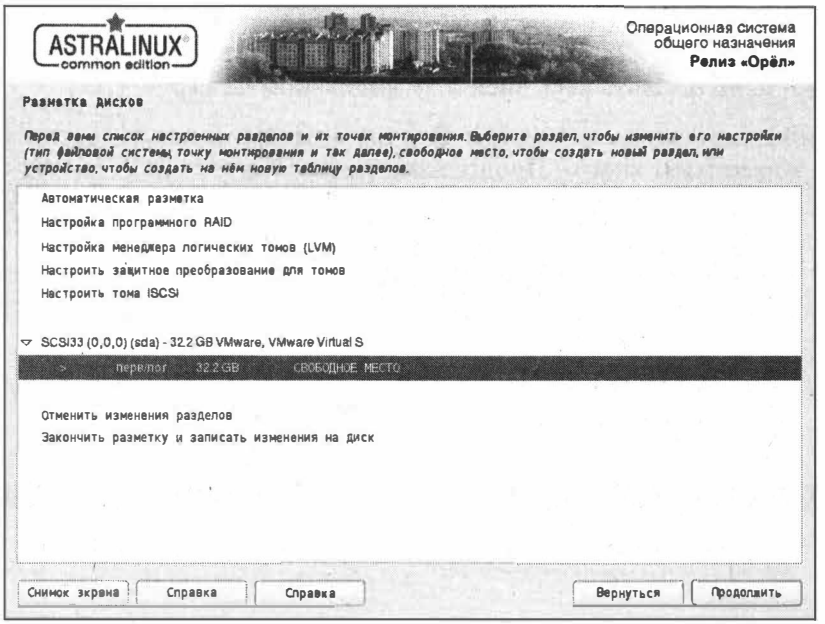


Рис. 1.17. Выберите свободное место и нажмите кнопку "Продолжить"

В появившемся меню (рис. 1.18) выберите команду **Создать новый раздел** и снова нажмите сами знаете какую кнопку. Нужно будет ввести размер раздела (рис. 1.19), выбрать его тип (первичный/логический), расположение (начало/конец), далее появится возможность установить другие параметры раздела (рис. 1.20). Вы можете изменить точку монтирования, параметры монтирования, метку и т.д. Параметр **Использовать как** позволяет изменить файловую систему раздела.

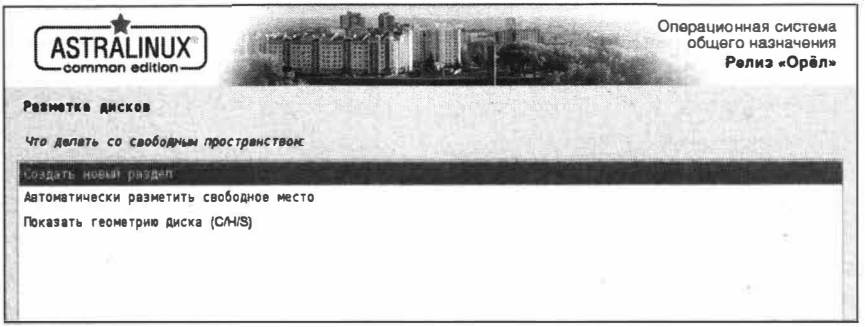


Рис. 1.18. Выберите команду "Создать новый раздел"

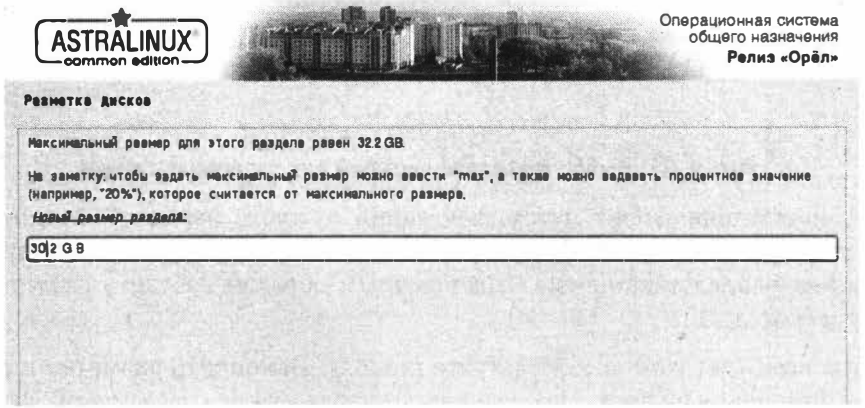


Рис. 1.19. Размер создаваемого раздела

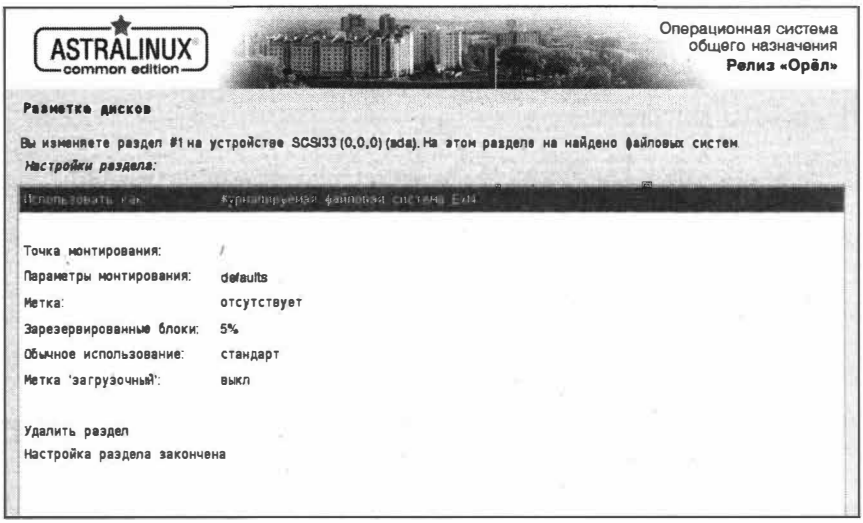


Рис. 1.20. Настройка раздела

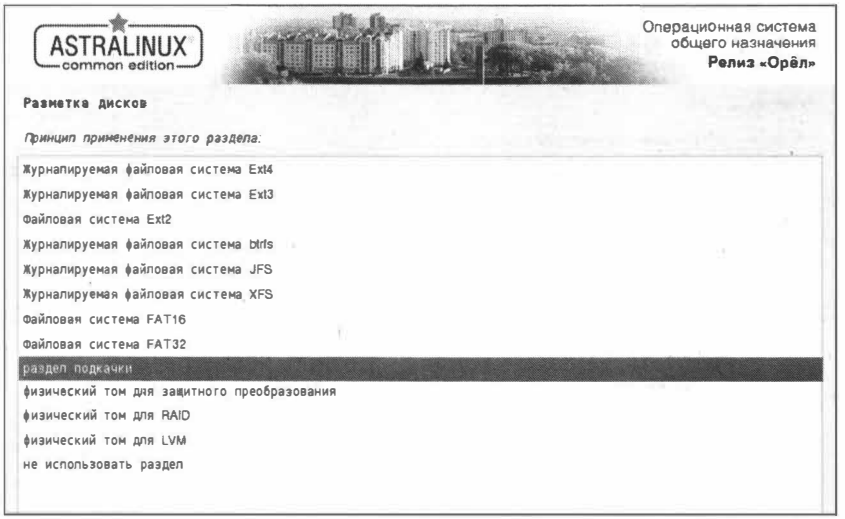


Рис. 1.21. Выбор файловой системы для раздела подкачки

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. 1.21).

Когда все будет готово, используйте команду **Закончить разметку и записать изменения на диск**. На следующем экране нужно согласиться с форматированием разделов (рис. 1.23). Средство разметки диска очень неудобное, но и к нему можно привыкнуть.

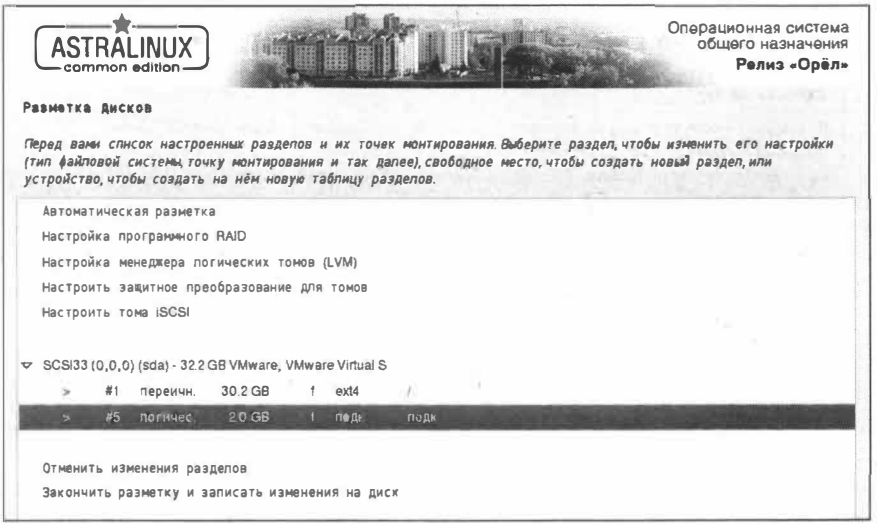


Рис. 1.22. Готовая разметка

Рис. 1.23. Подтверждаем изменения

1.6. 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. 1.24) 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-**

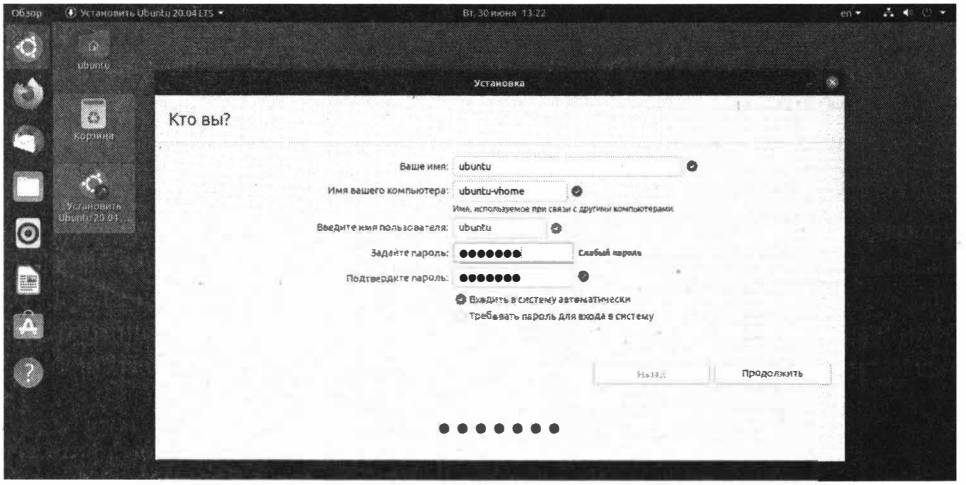


Рис. 1.24. Создание пользователя в Ubuntu

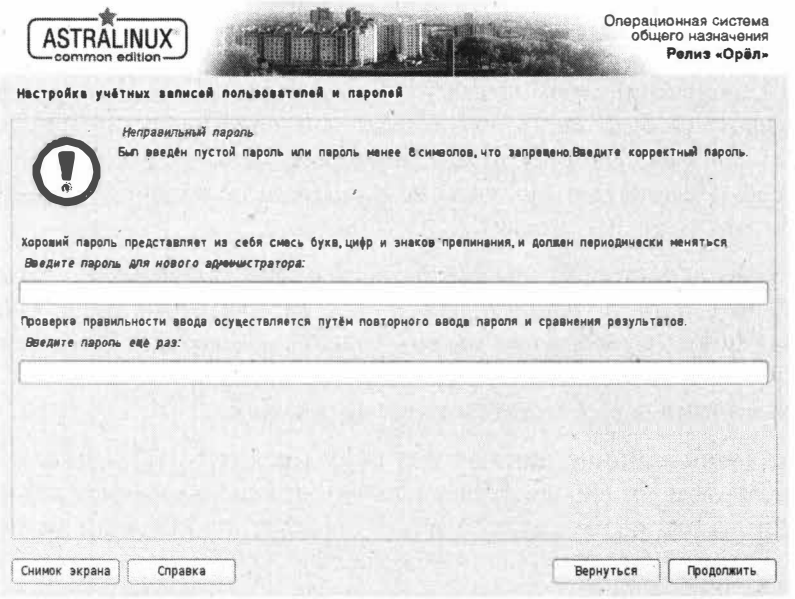


Рис. 1.25. Простые пароли в Astra Linux использовать нельзя

скую сессию (рис. 1.26). В следующей главе будет показано, как включить/выключить автовход уже после установки системы.

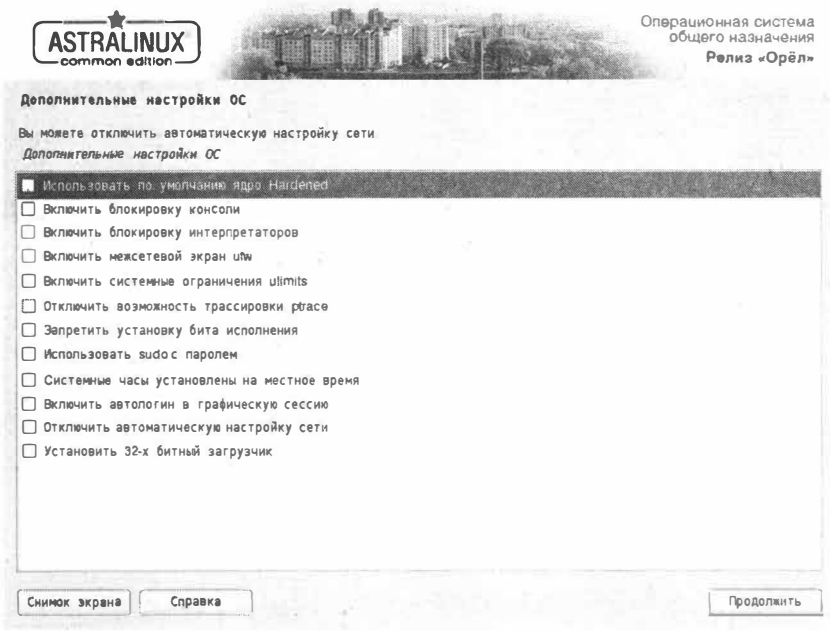


Рис. 1.26. Включение автовхода в Astra Linux

1.7. Параметры программного обеспечения

В Astra Linux позволяют выбрать наборы программного обеспечения (рис. 1.27). 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. 1.28), 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.

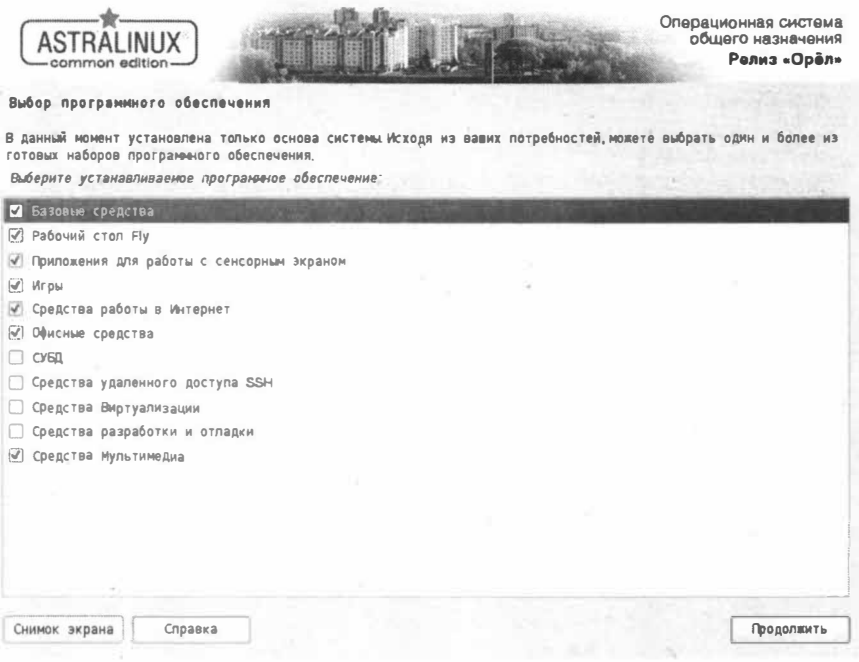


Рис. 1.27. Доступные по умолчанию наборы ПО

Рис. 1.28. Настройки ПО в Ubuntu

Сразу после установки Ubuntu предложит вам доустановить необходимые программы (рис. 1.29). Вы можете сделать это сразу, а можете по мере необходимости.

Рис. 1.29. Предложение установить некоторые программы

1.8. Установка загрузчика

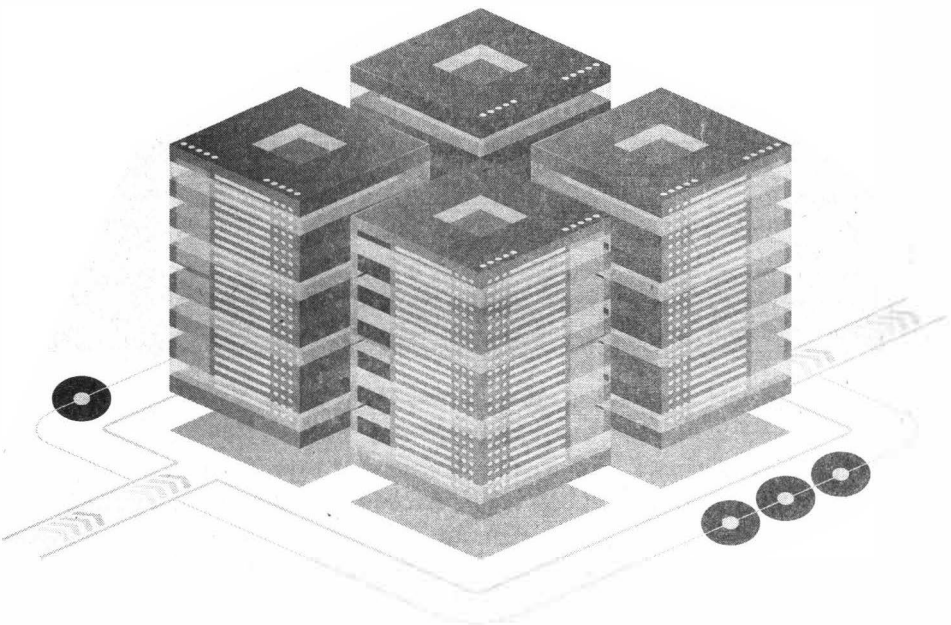
Ubuntu устанавливается проще. По сути, самое сложное – это разметка диска. Но в Astra Linux нужно еще выбрать, куда устанавливать загрузчик Linux. Обычно его нужно установить в /dev/sda. Но если у вас есть уже другая операционная система и вы хотите загружать Linux не с помощью родного загрузчика GRUB2, 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. 1.30. Выбор устройства для установки загрузчика

Глава 2.

Вход в систему сервера



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

2.1. 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. 2.1 изображен вход в Astra Linux. На рис. 2.2 изображен графический вход в Astra Linux.

Рис. 2.1. Вход в систему (Astra Linux)

Давайте разберемся, что есть что. Самая первая строка (рис. 2.1) - 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 #

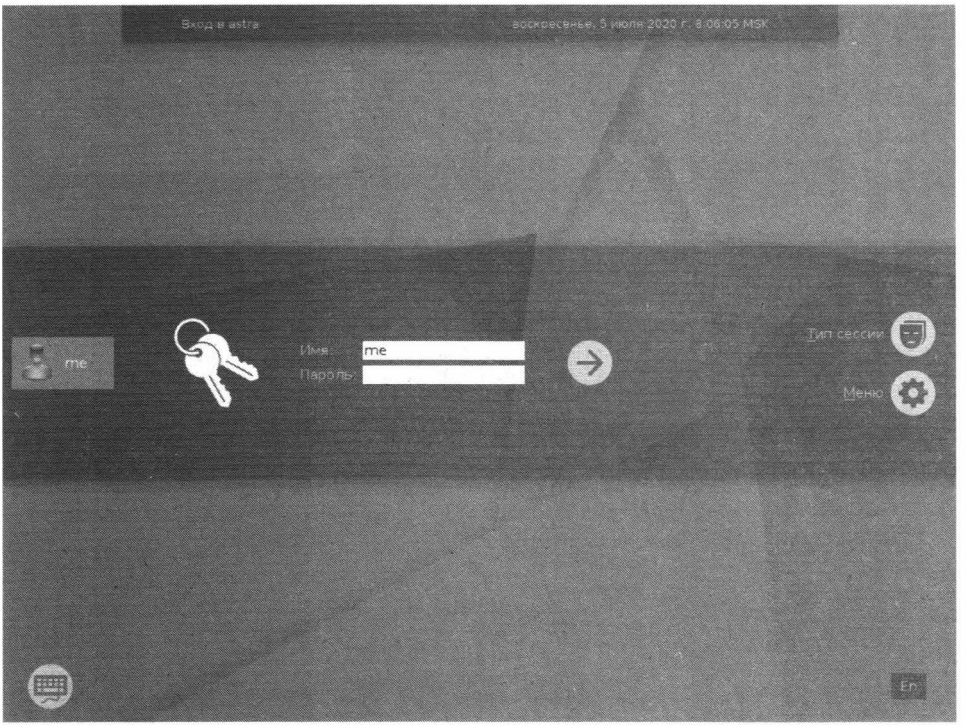


Рис. 2.2. Графический вход (Astra Linux)

(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-

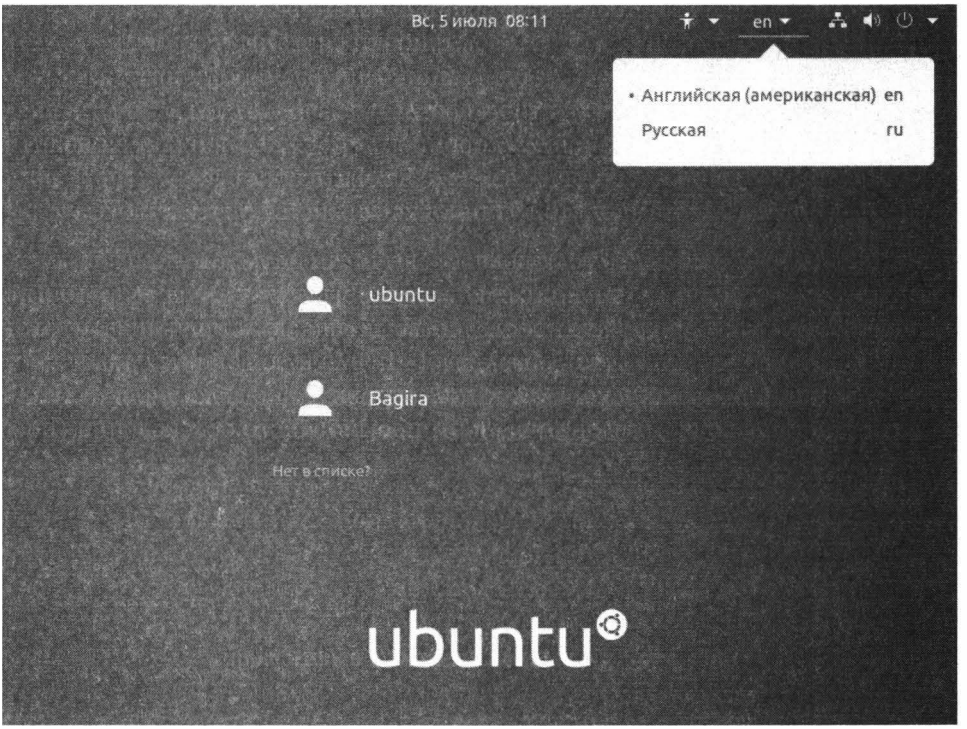


Рис. 2.3. Вход в систему (Ubuntu 20.04)

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. 2.3).

В Astra Linux можно выбрать тип сессии (рис. 2.4), а также открыть меню (рис. 2.5), 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.

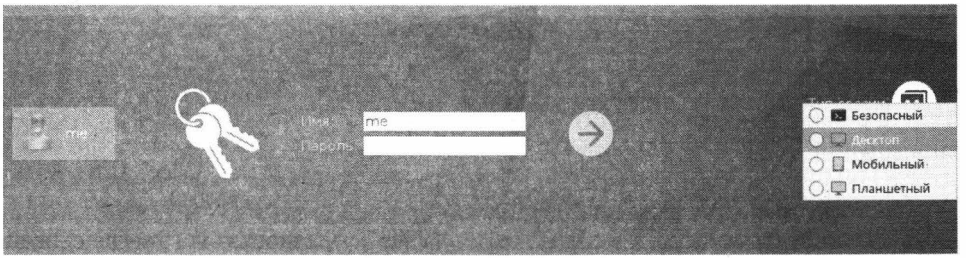


Рис. 2.4. Выбор типа сеанса (Astra Linux)

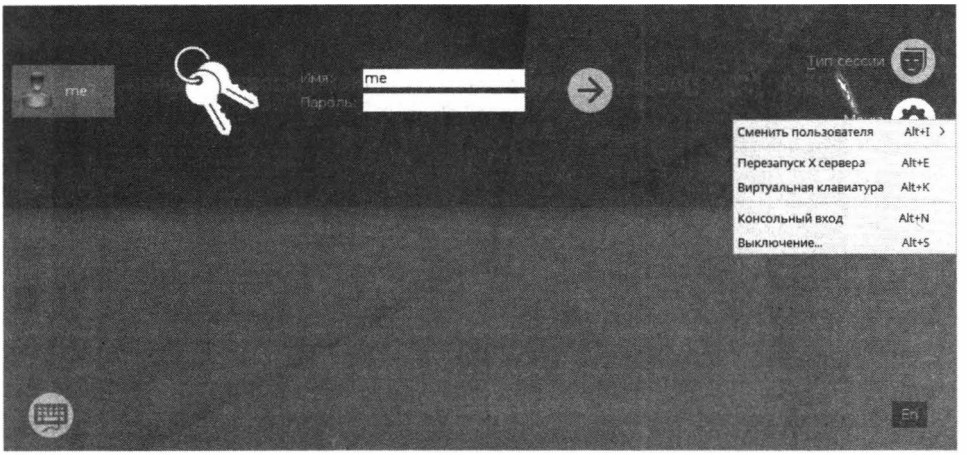


Рис. 2.5. Меню входа в систему (Astra Linux)

2.2. 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.

2.2.1. 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. 2.6). В самом низу этой панели есть кнопка, позволяющая открыть экран Приложения (рис. 2.7).

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.

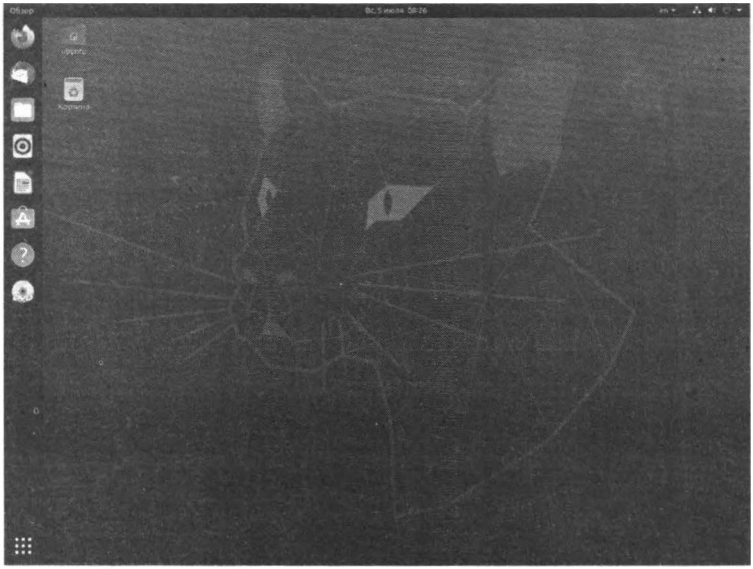


Рис. 2.6. Рабочий стол Ubuntu 20.04

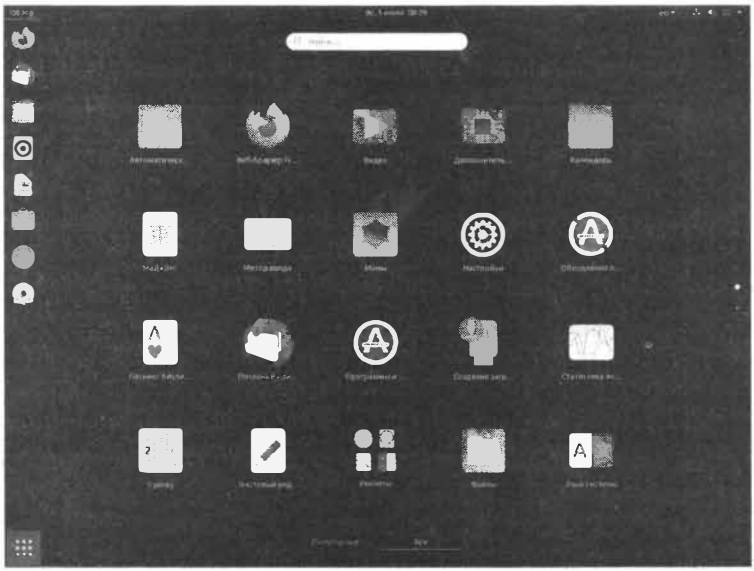


Рис. 2.7. Приложения

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. 2.8 показано, что запущены браузер, файловый менеджер и терминал (средство для ввода команд).

Рис. 2.8. 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. 2.9).

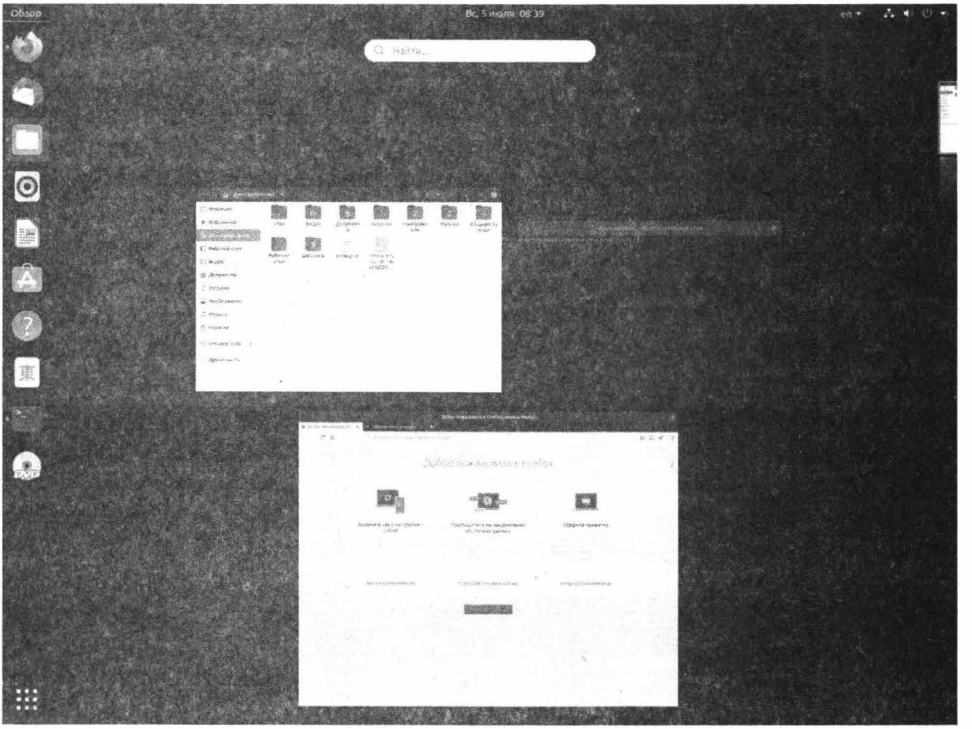


Рис. 2.9. Экран Обзор

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. 2.10). 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.

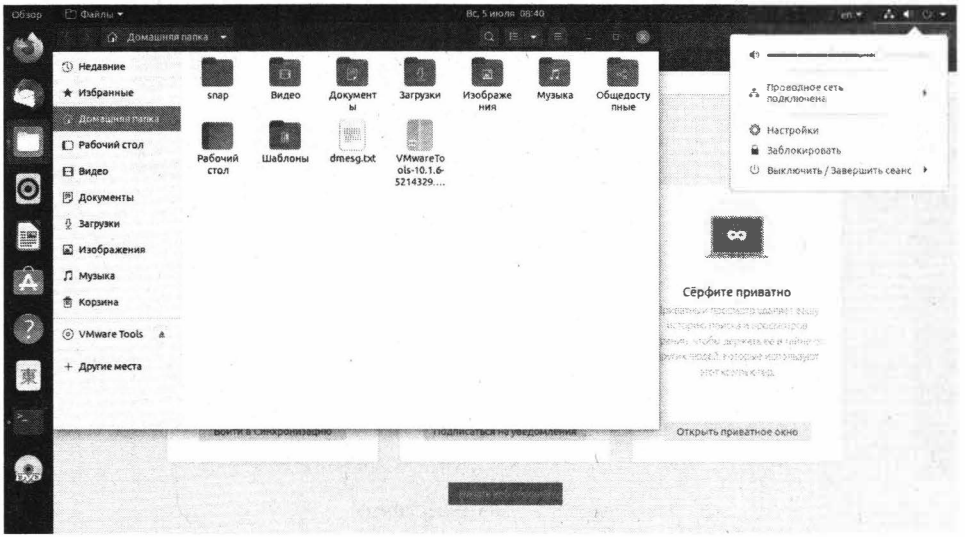


Рис. 2.10. Системное меню

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. 2.11). 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.

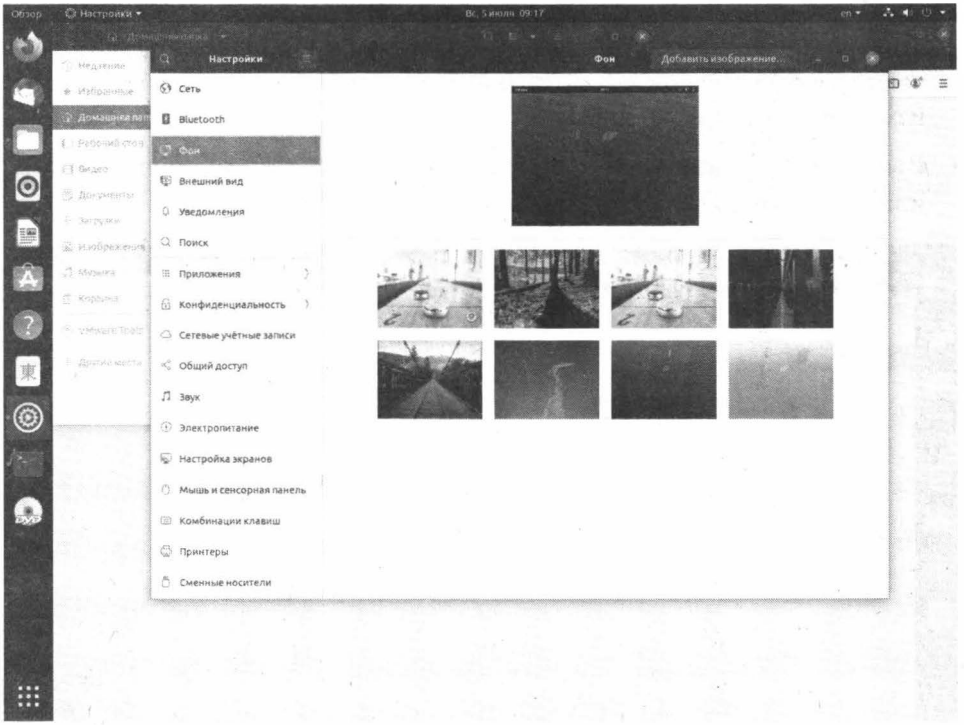


Рис. 2.11. Раздел "Фон"

Рис. 2.12. Раздел "Внешний вид"

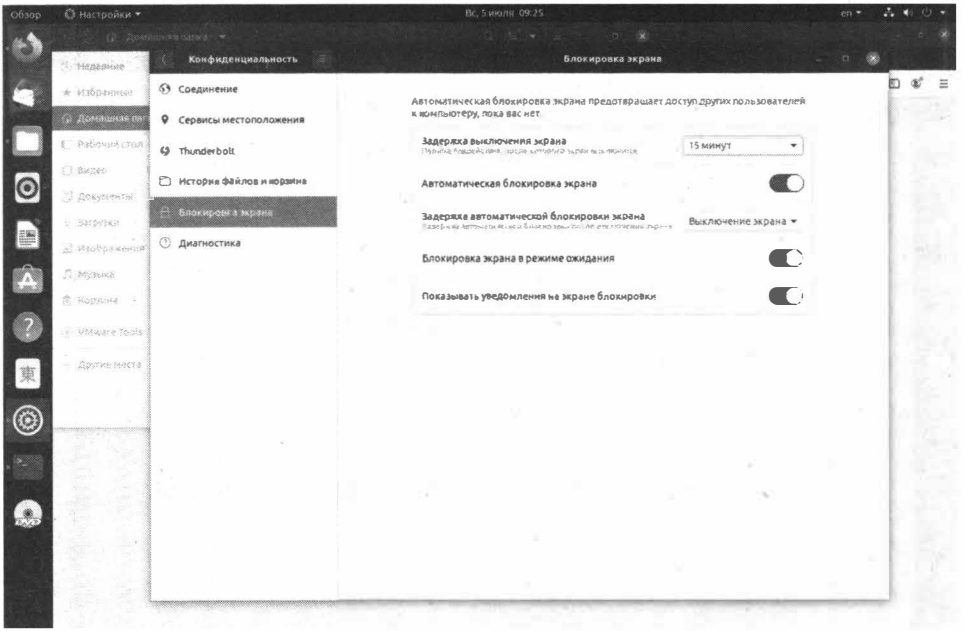


Рис. 2.13. Раздел "Блокировка экрана"

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. 2.15).

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.

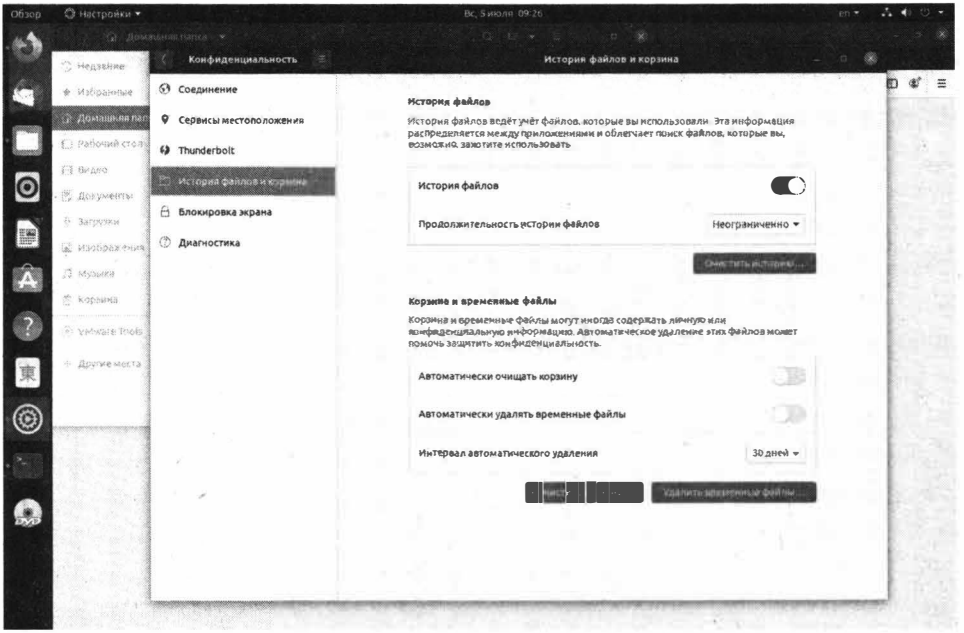


Рис. 2.14. История файлов и корзина

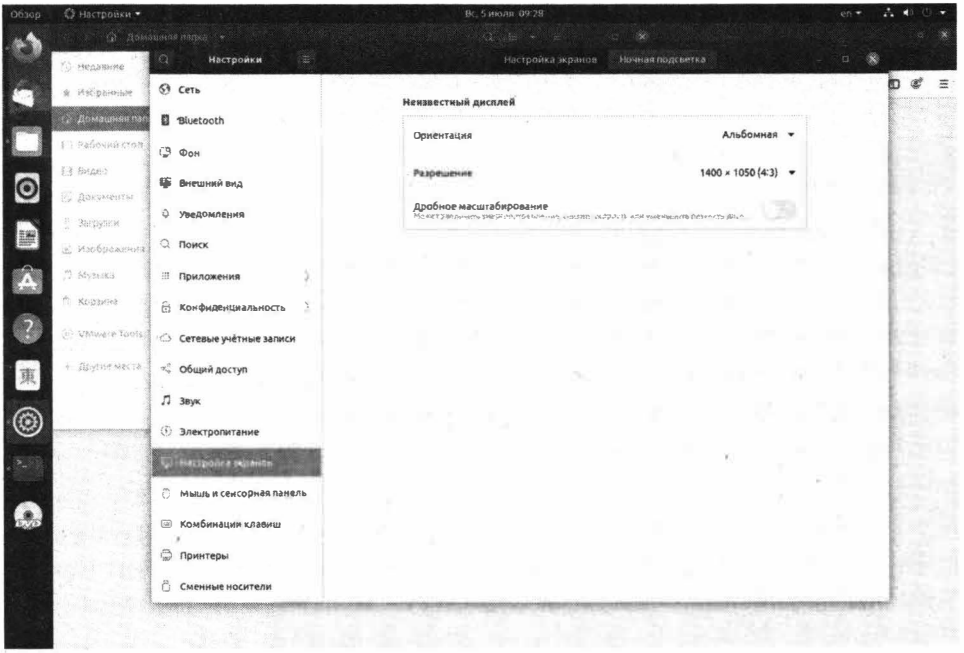


Рис. 2.15. Настройка экранов

2.2.2. 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. 2.16). Бывшие Windows-пользователи оценят такое решение – для них такой интерфейс будет привычнее интерфейса Ubuntu.

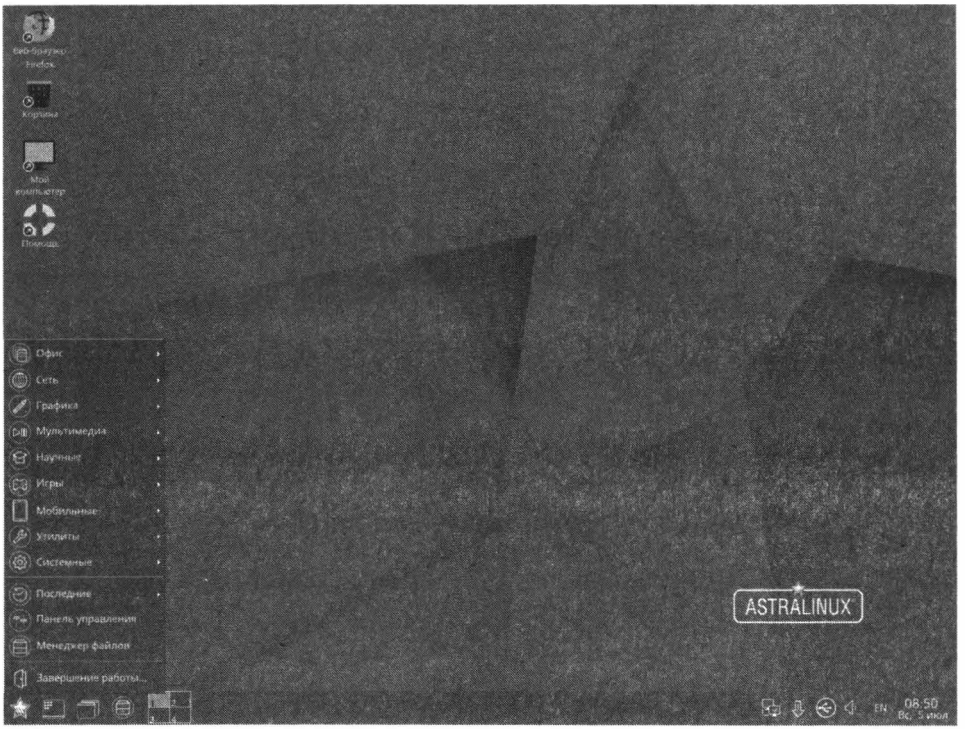


Рис. 2.16. Рабочий стол и главное меню Astra Linux

Рассмотрим содержимое панели задач. **Первая** кнопка открывает главное меню, изображенное на рис. 2.16. **Вторая** кнопка – сворачивает все окна, предоставляя доступ к рабочему столу. **Третья** используется для переключения между окнами – когда вам нужно быстро найти запущенное окно (рис. 2.17).

Четвертая кнопка открывает файловый менеджер (рис. 2.18). 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

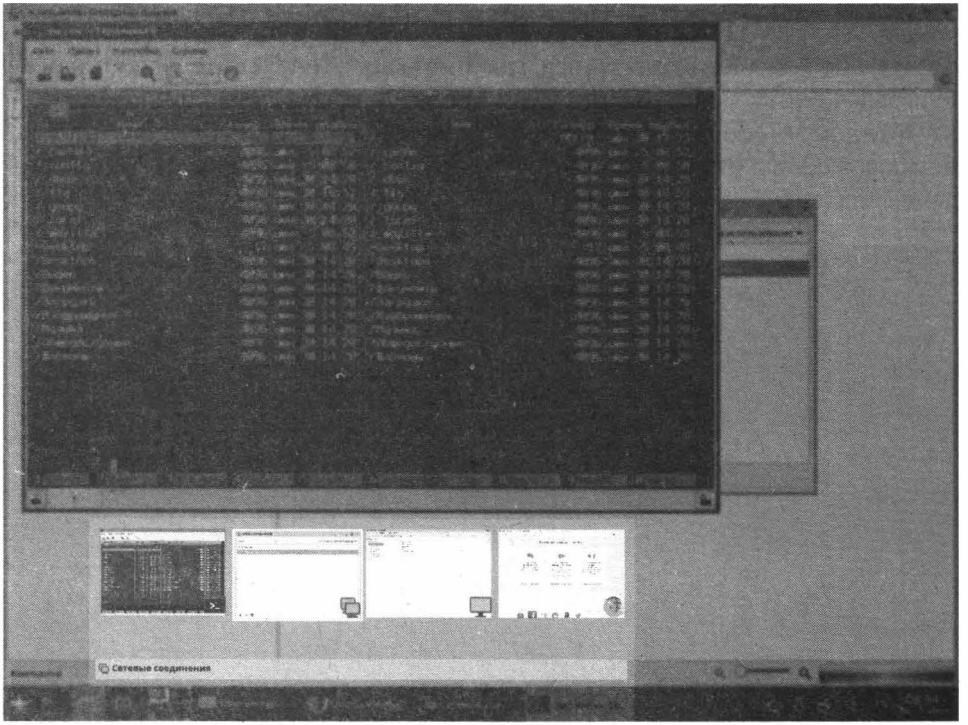


Рис. 2.17. Переключение между окнами в Astra Linux

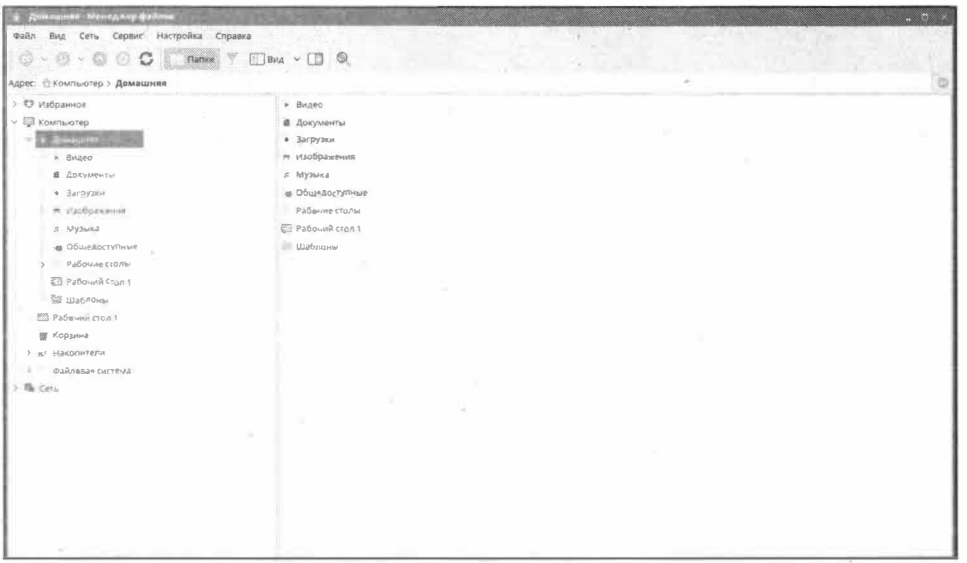


Рис. 2.18. Файловый менеджер Astra Linux

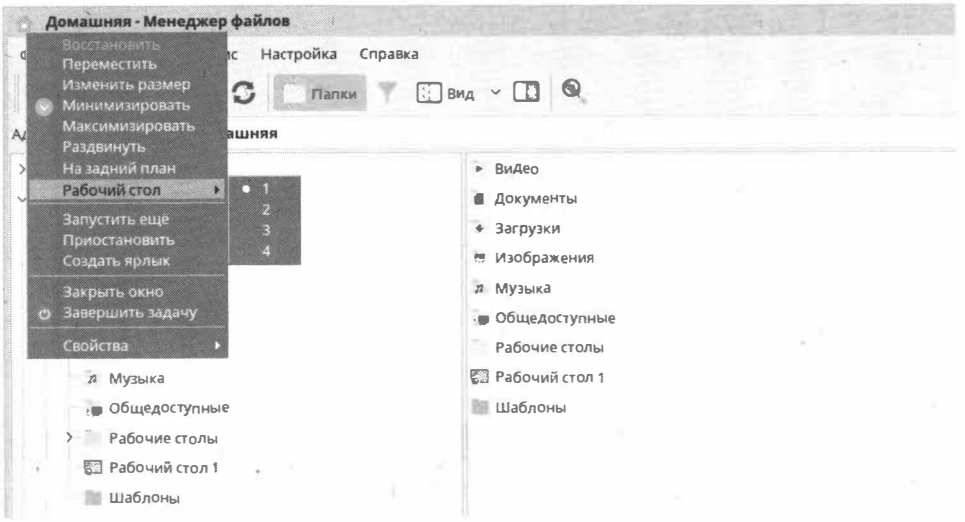


Рис. 2.19. Перемещения окна на другой рабочий стол

стол щелкните правой кнопкой мыши по заголовку окна и выберите меню **Рабочий стол**, далее выберите номер рабочего стола (рис. 2.19).

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. 2.20).

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

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

Рис. 2.20. 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 6.04.

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. 2.21).

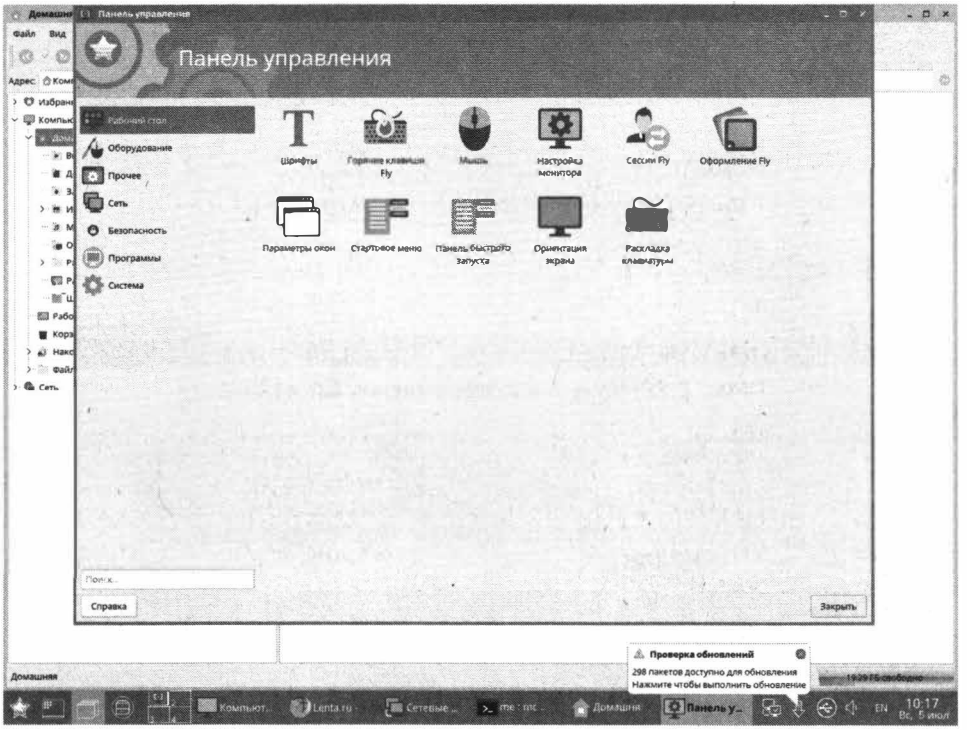


Рис. 2.21. Панель управления Astra Linux

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

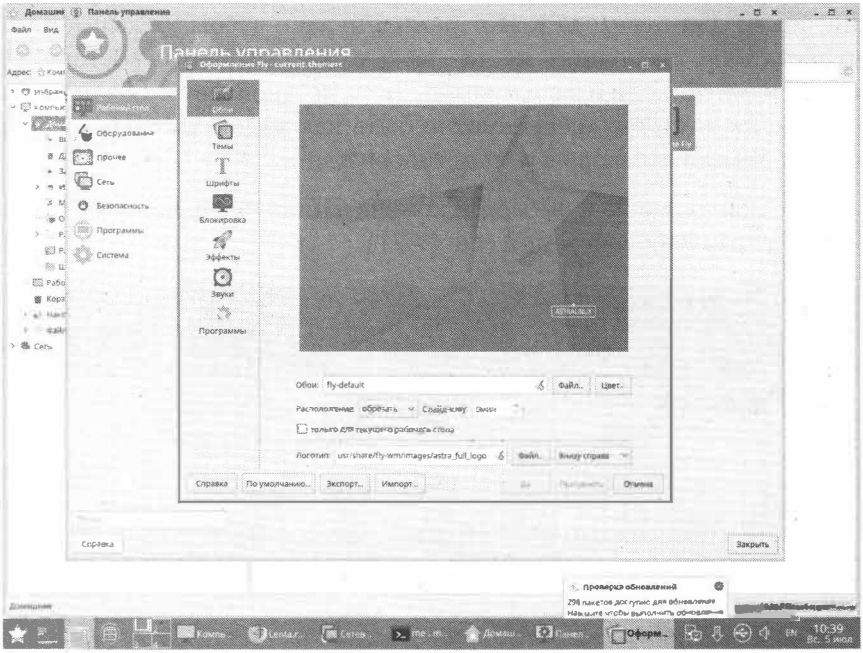


Рис. 2.22. Изменение оформления Astra Linux

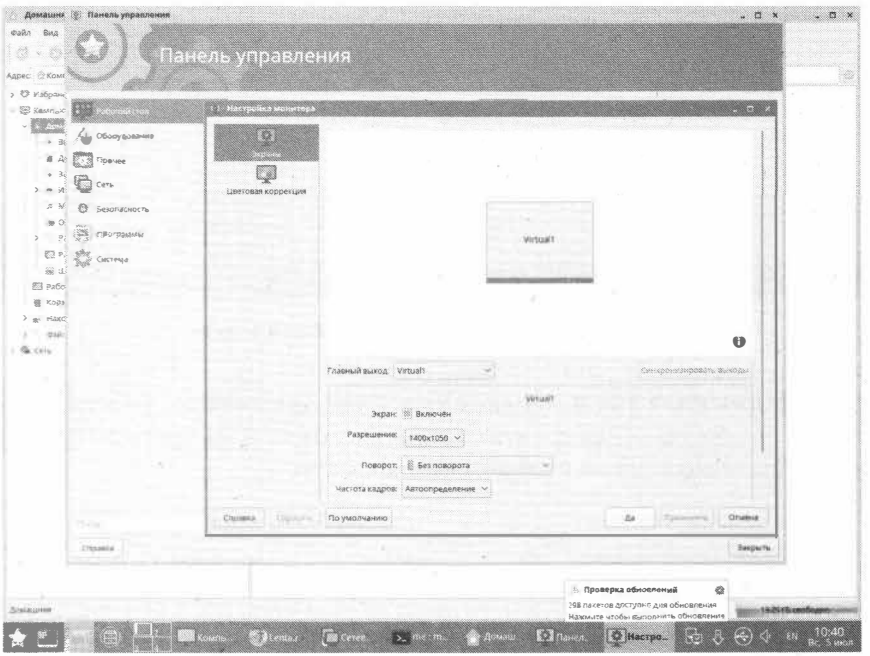


Рис. 2.23. Выбор разрешения монитора

Раздел **Настройка монитора** позволяет выбрать разрешение монитора (рис. 2.23).

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.

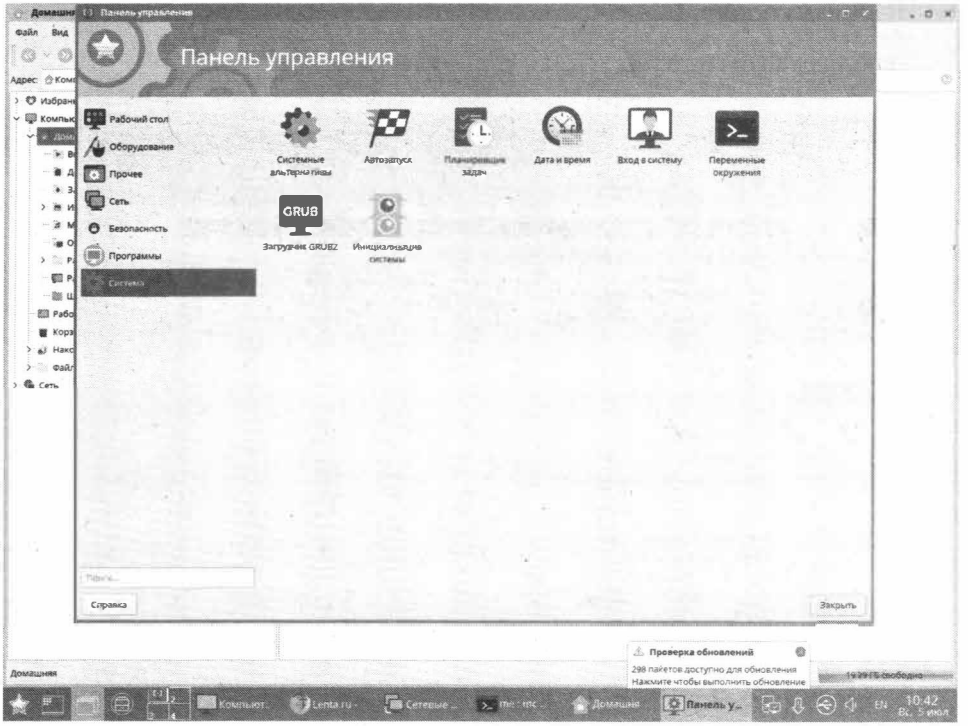


Рис. 2.24. Раздел Система

2.3. 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. 2.25);
6. Нажмите кнопку **Да**;
7. Перезагрузите систему.

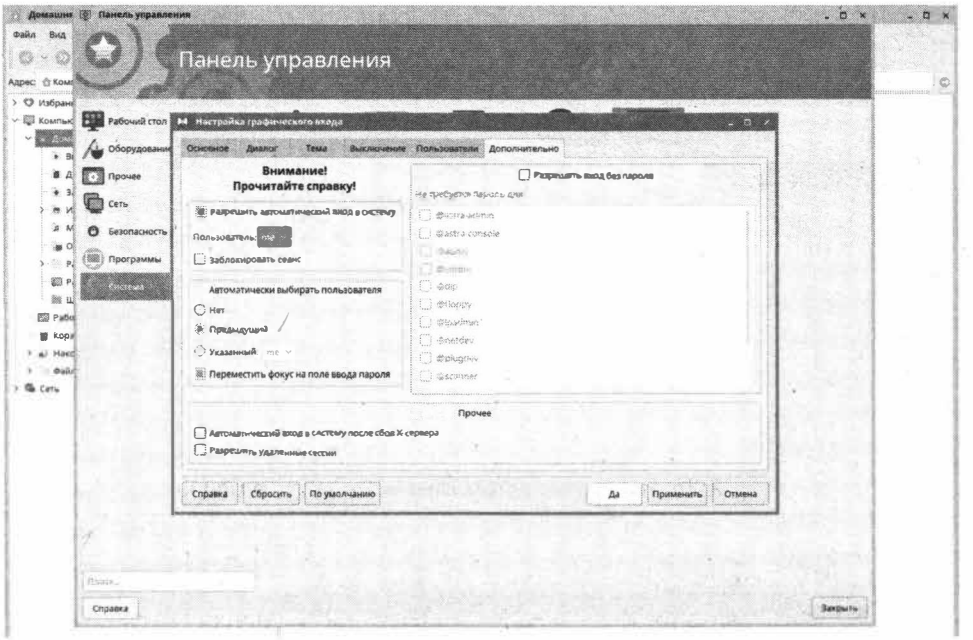


Рис. 2.25. Настройка автоматического входа в Astra Linux

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.

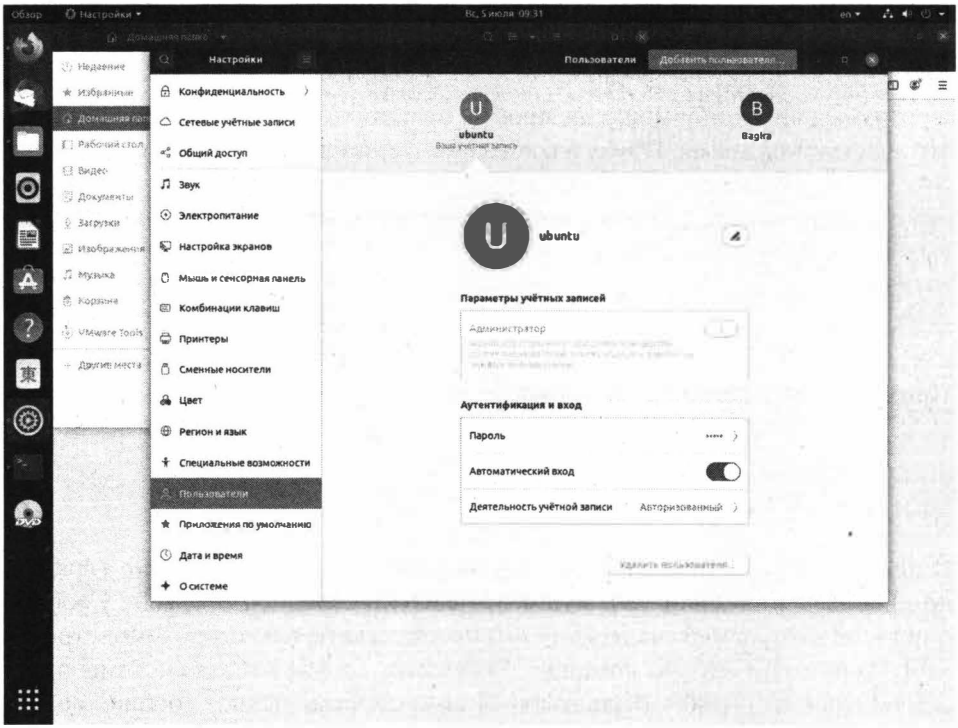


Рис. 2.26. Настройка автоматического входа в Ubuntu

2.4. 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.

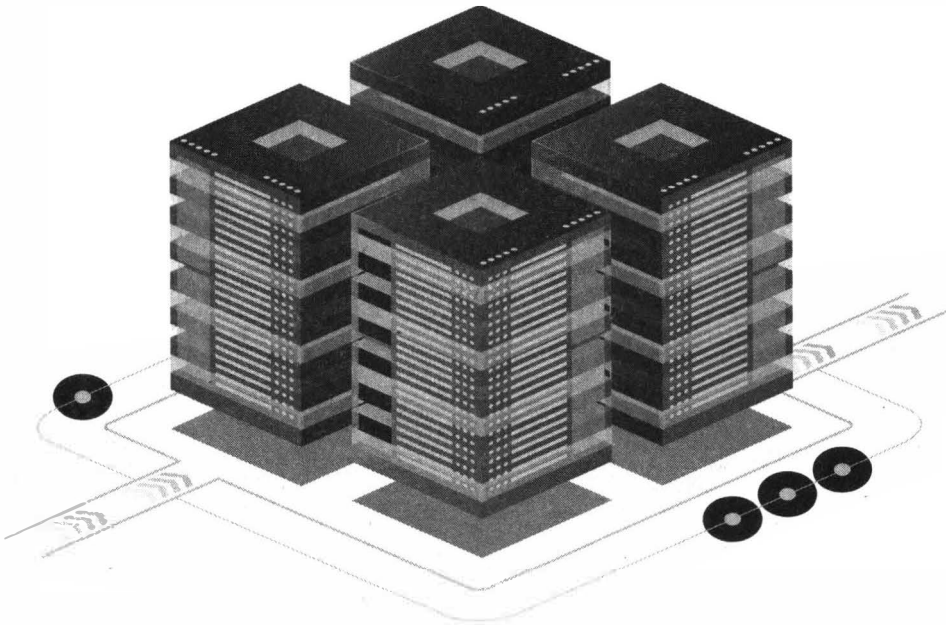
Дополнительные параметры shutdown приведены в таблице 2.1.

Таблица 2.1. 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 3.

Обновления и тюнинг интерфейса



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.

3.1. 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.

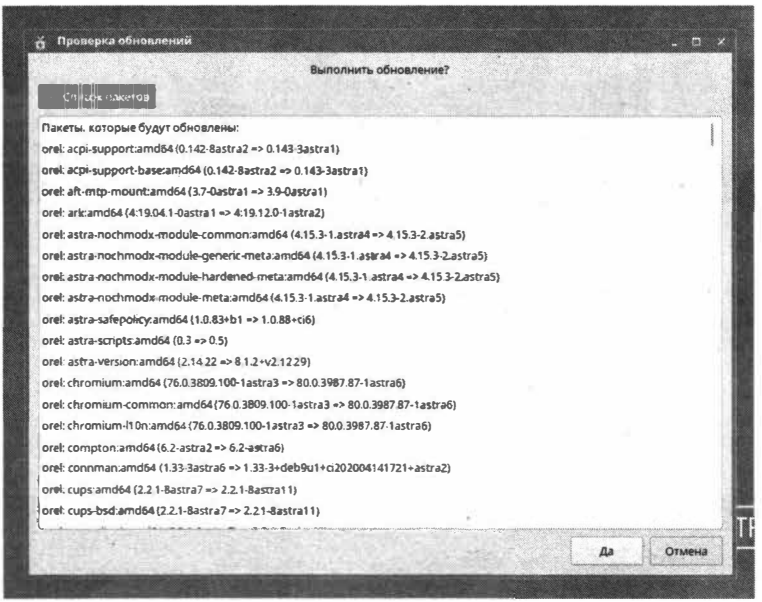


Рис. 3.1. Обновление системы в Astra Linux

В Ubuntu нажмите Alt + F2, в появившемся окне введите команду **update-manager**. Откроется окно менеджера обновлений (рис. 3.2). 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. 3.2. Она означает, что обновления уже установлены и для их применения нужно перезагрузить компьютер.

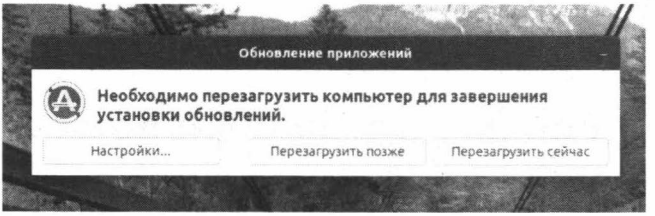


Рис. 3.2. Менеджер обновлений в Ubuntu

Нажмите кнопку **Настройки** (рис. 3.3), чтобы настроить периодичность проверки обновлений. Если вы не хотите, чтобы система автоматически проверяла наличие обновлений, из списка **Автоматически проверять наличие обновлений** выберите **Никогда**.

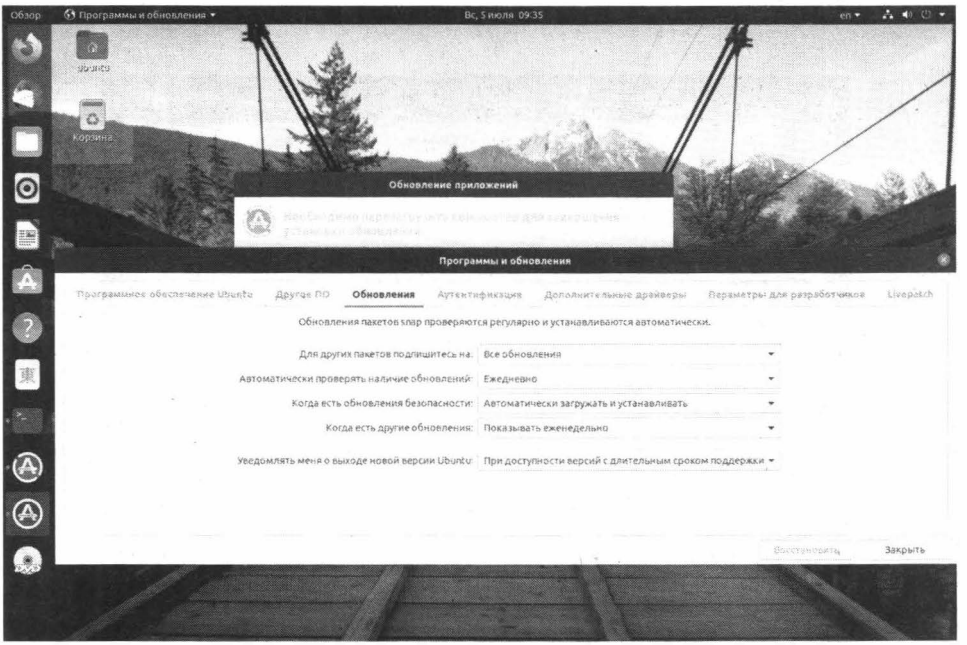


Рис. 3.3. Параметры обновлений по умолчанию

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

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

3.2. Настройка 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.

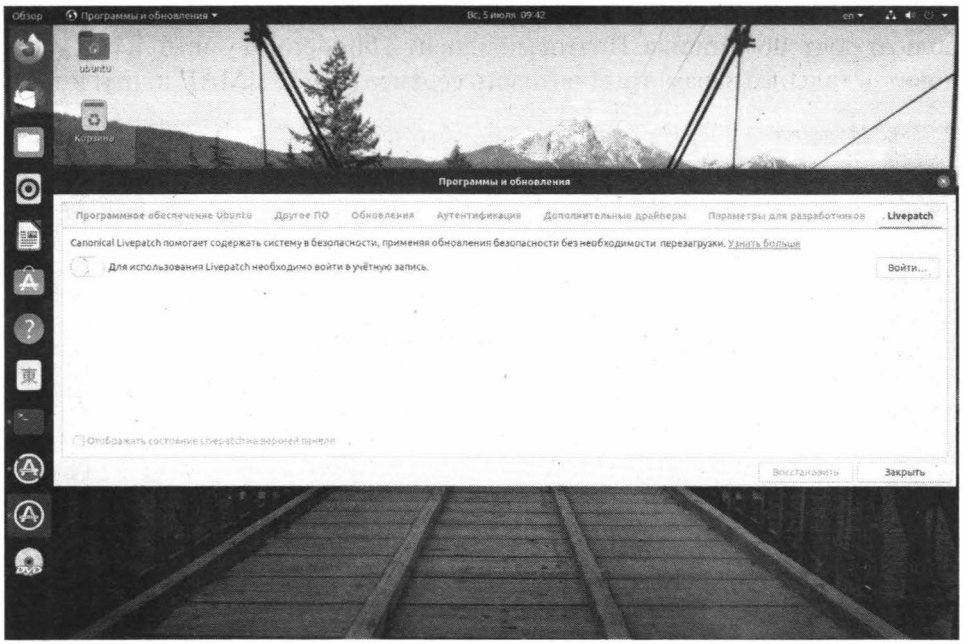


Рис. 3.4. Активация Livepatch

3.3. 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**.

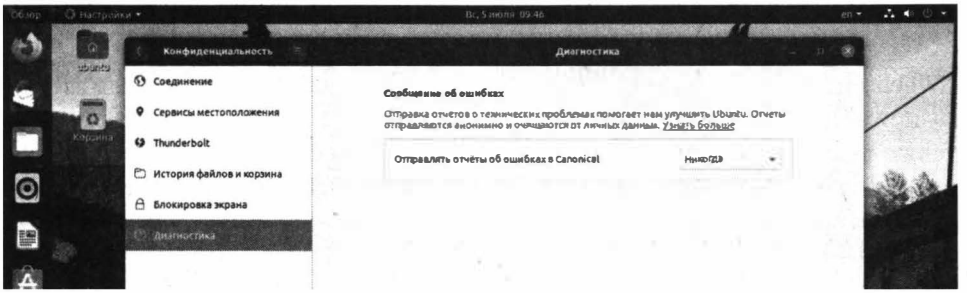


Рис. 3.5. Отключаем уведомления об ошибках

3.4. 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.).

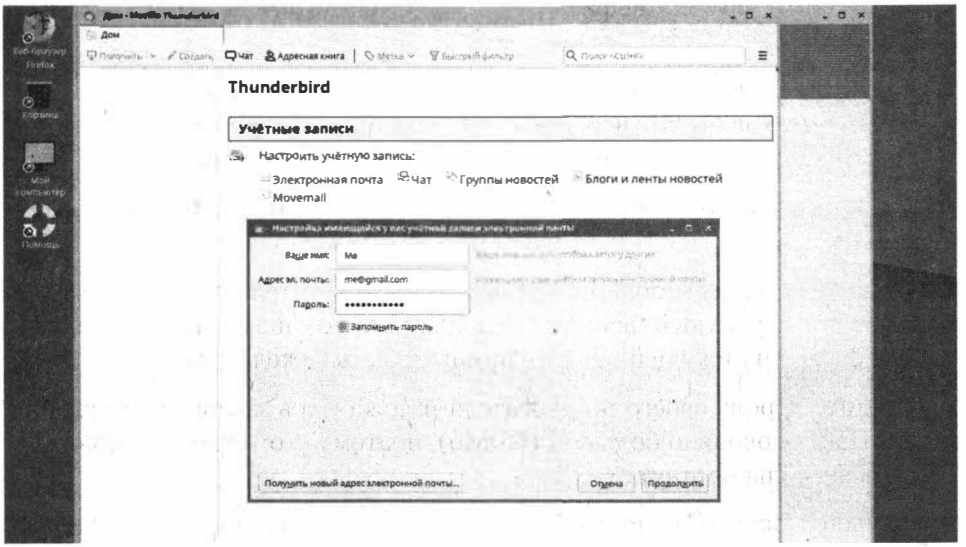


Рис. 3.6. Почтовый агент Thunderbird в Astra Linux

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!

3.5. 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.

3.6. 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. 3.14. Проигрыватель VLC

3.7. 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
```

3.8. 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. 3.15).

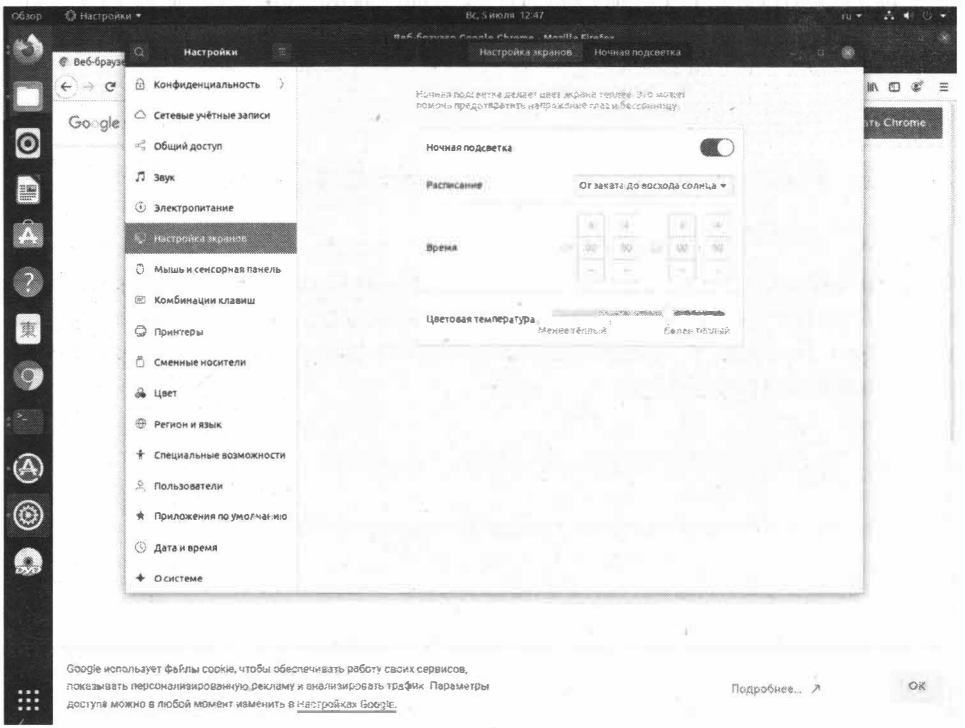


Рис. 3.15. Активация ночного режима

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.

3.9. Установка wine для запуска Windows-приложений

Установите Wine – средство, обеспечивающее запуск Windows-приложений. Для его установки введите команду:

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

3.10. 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
```

3.11. 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. 3.16):

```
$ 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.

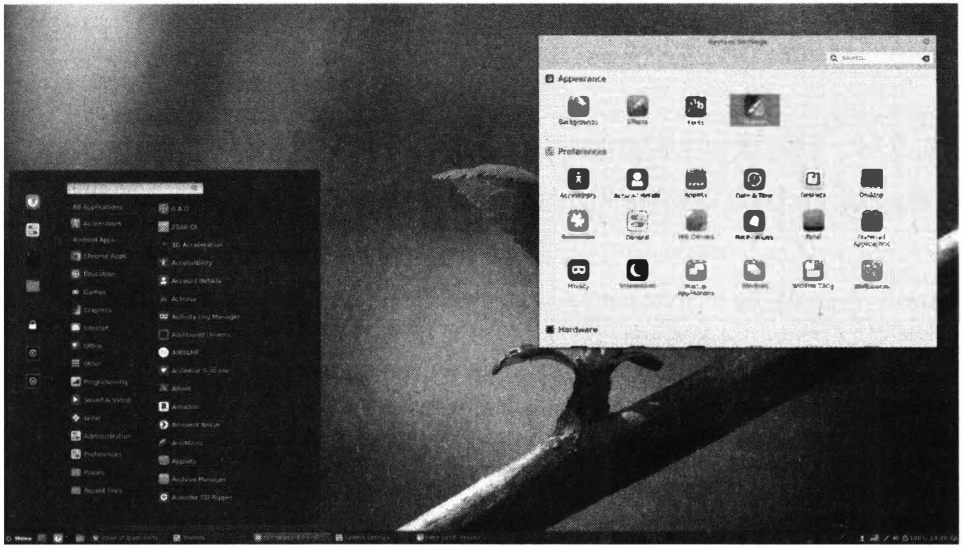


Рис. 3.16. Графическая среда Cinnanmon

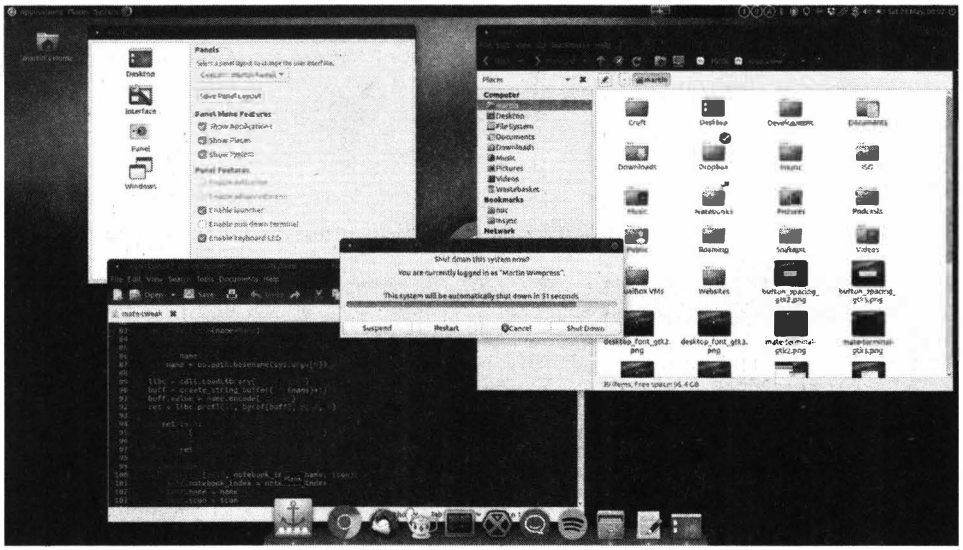


Рис. 3.17. Графическая среда MATE

3.12. 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
```

3.13. 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. 3.18). Кстати, только с ее помощью можно изменить тему оформления в

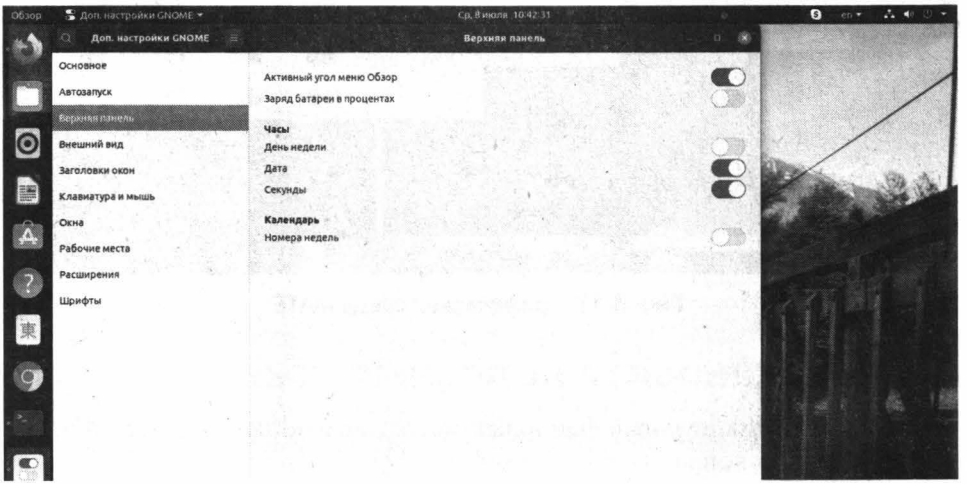


Рис. 3.18. Приложение Gnome Tweaks

Ubuntu на любую другую. Перейдите в раздел **Внешний вид** и выберите другую тему оформления, как показано на рис. 3.19. На рис. 3.20 показано, что тема изменена. Далее будет показан пример наиболее популярного «тюнинга» Ubuntu – установка темы оформления в стиле macOS.

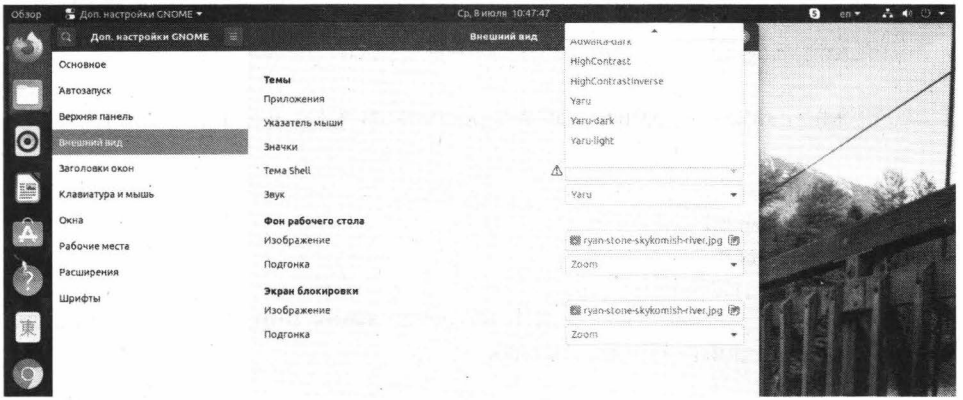


Рис. 3.19. Изменение темы оформления

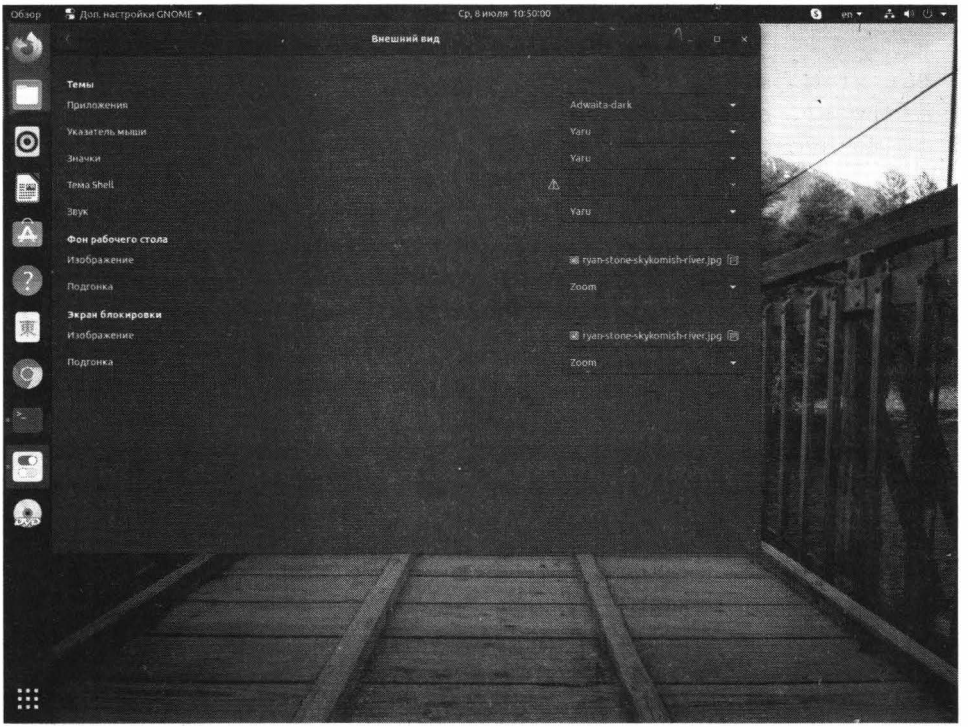


Рис. 3.20. Тема оформления изменена

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. 3.21 показан процесс распаковки необходимых архивов, а на рис. 3.22 – настройки, сделанные в Gnome Tweaks. У вас должно получиться так, как показано на рис. 3.22. 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/**

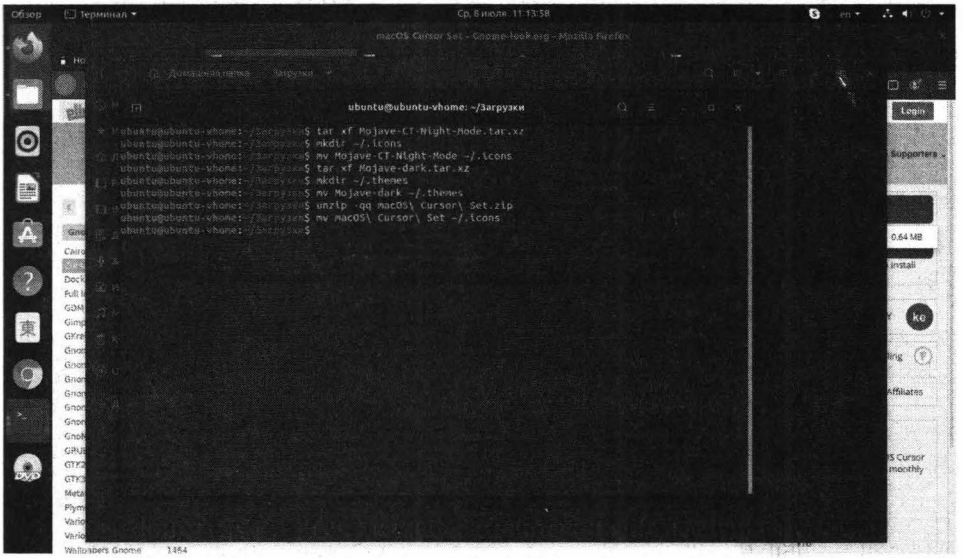


Рис. 3.21. Распаковка ресурсов

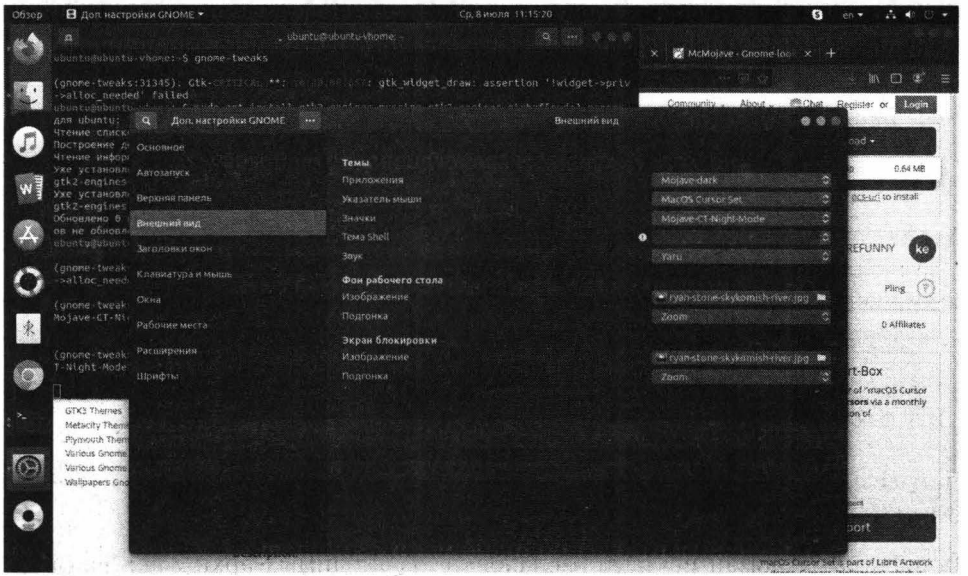


Рис. 3.22. Настройки в Gnome Tweaks

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`

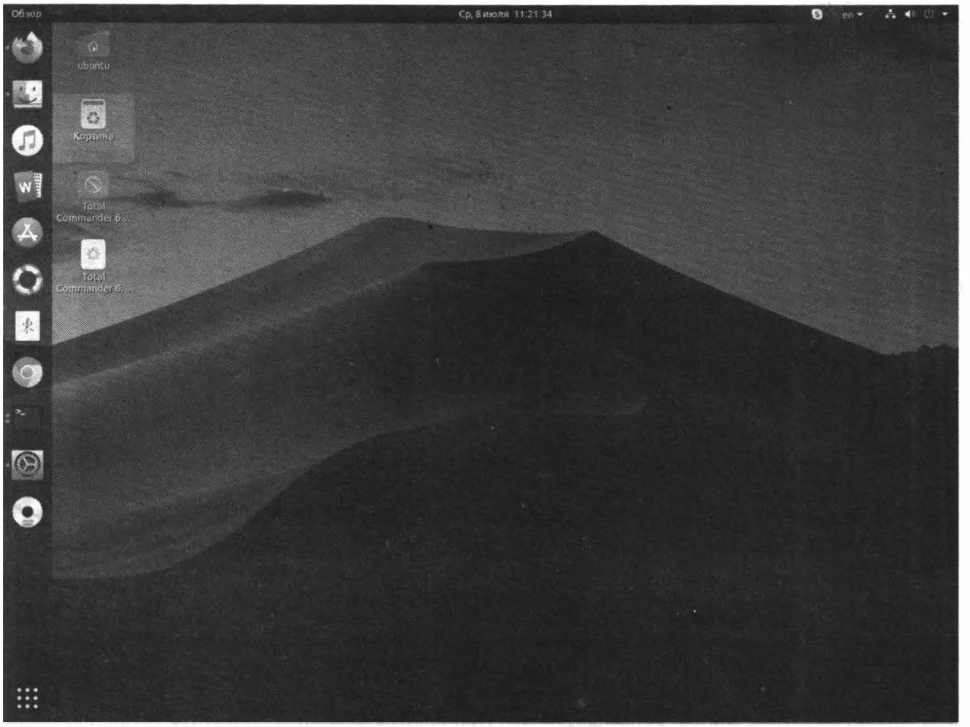


Рис. 3.23. Как будет выглядеть ваша Ubuntu после установки фонового изображения

Запустите новую панель задач:

```
plank
```

В нижней части экрана вы увидите ту самую панель (рис. 3.24).

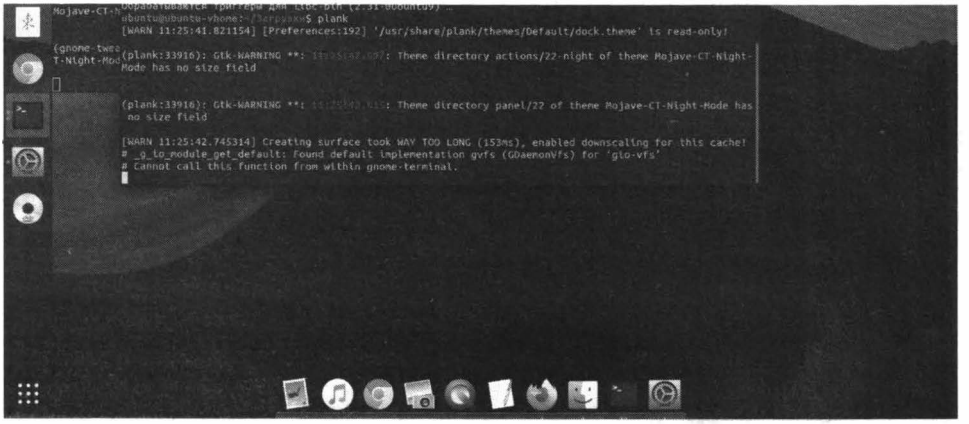


Рис. 3.24. Панель задач в стиле macOS

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. 3.25).

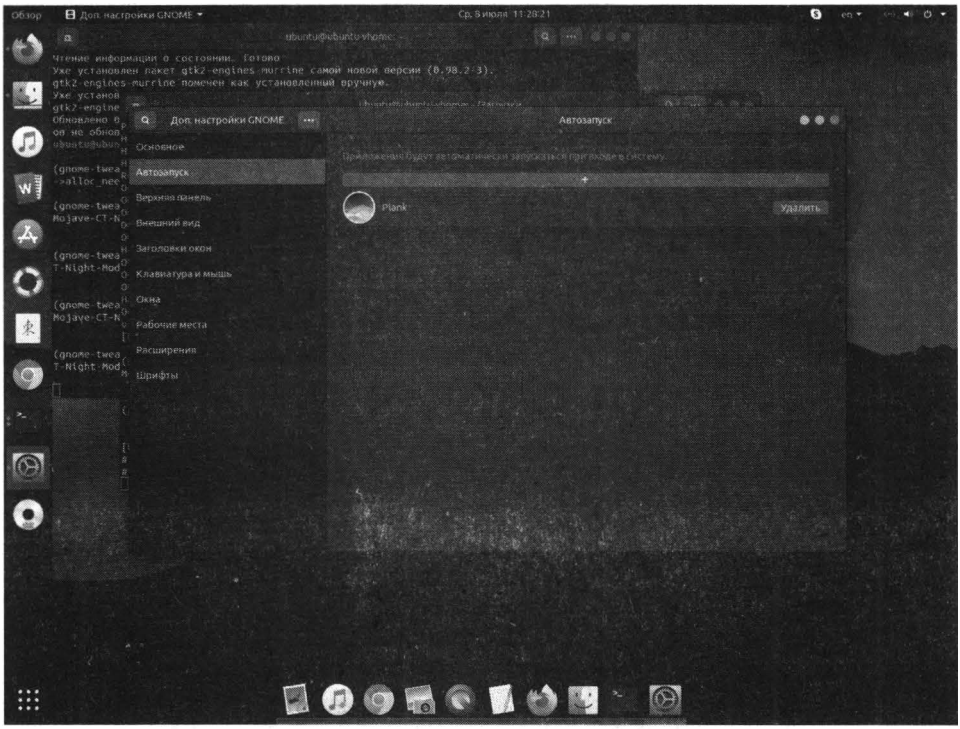


Рис. 3.25. Автоматический запуск панели задач Plank

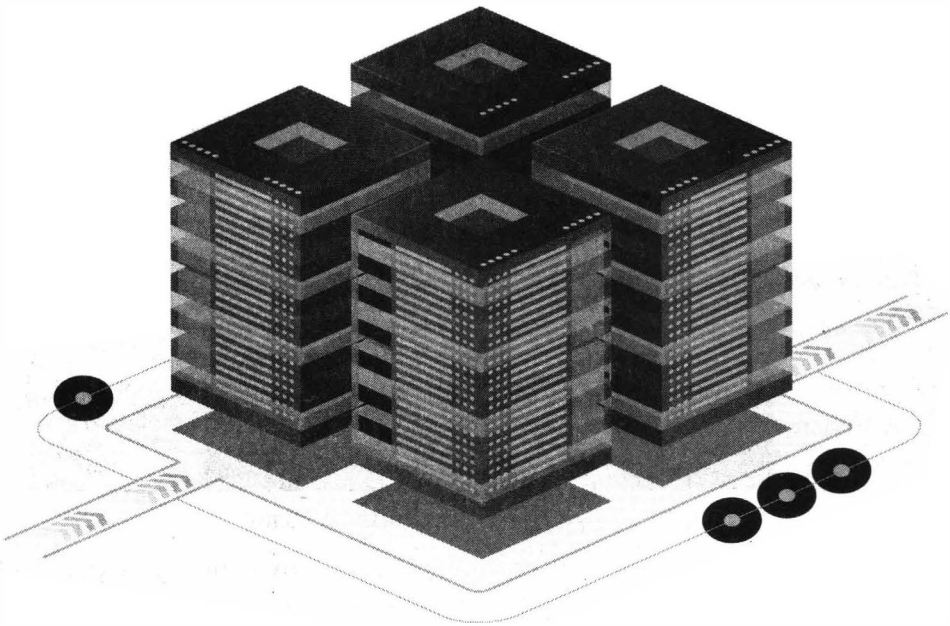
Осталось удалить стандартную панель задач. Для этого введите команду:

```
$ 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 4.

Командная строка и ее использование на сервере



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

4.1. 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. 4.1.

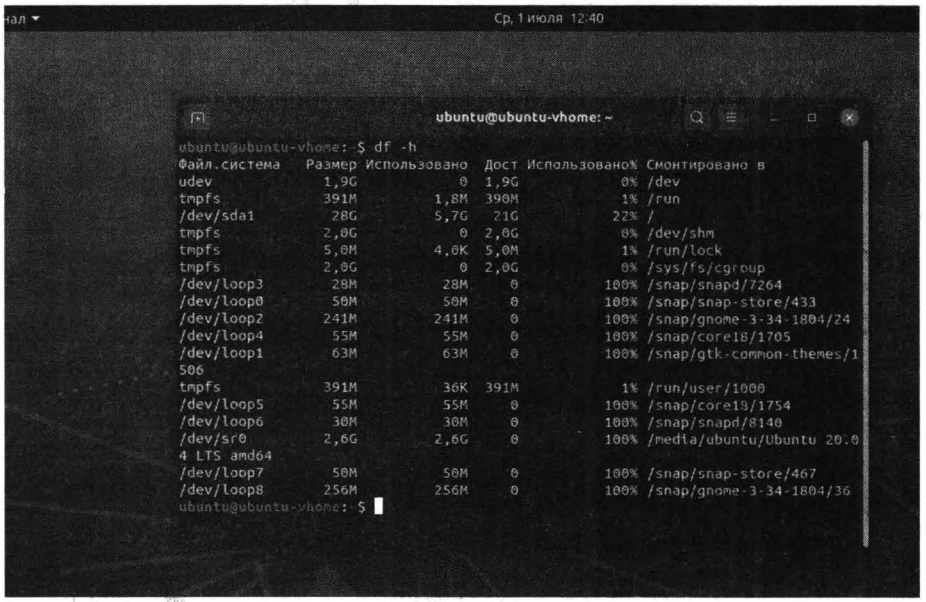


Рис. 4.1. Приложение "Терминал" (Ubuntu)

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. 4.2. 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.

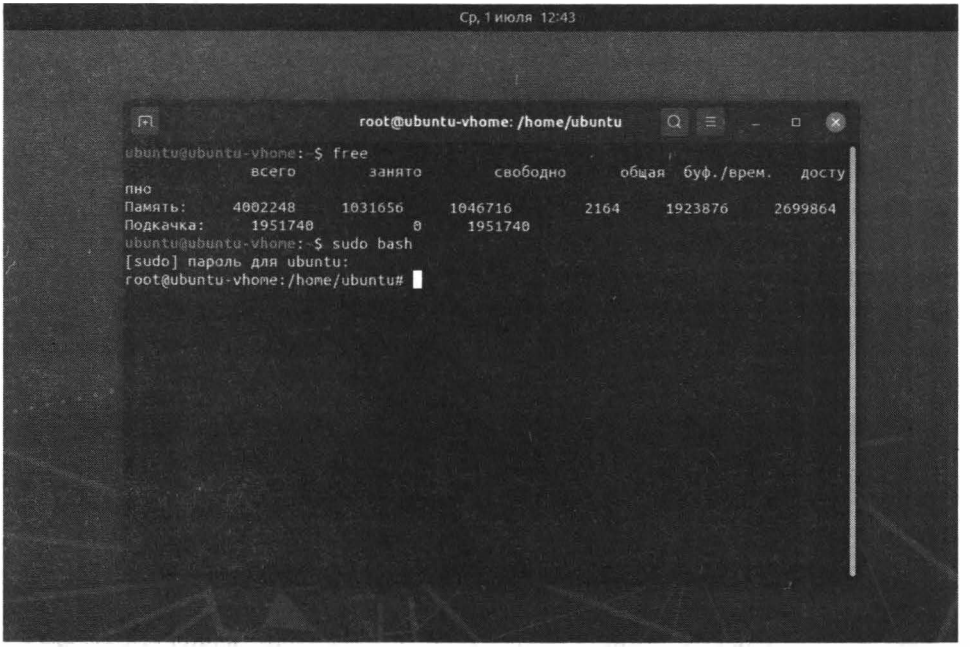


Рис. 4.2. Разные формы приглашения командной строки

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.

4.2. 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!

4.3. 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.

4.4. 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.

4.5. Команды для работы с файлами и каталогами

4.5.1. 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. 4.1).

Таблица 4.1. 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 4.1 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.

> Примечание. Обратите внимание, что в таблице 4.1 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
```

4.5.2. Команды для работы с каталогами

Аналогично командам для работы с файлами, команды для работы с каталогами представлены в таблице 4.2.

Таблица 4.2. 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. 4.3). А общий формат вызова этой команды таков:

```
ls [параметры] [каталог]
```

Таблица 4.3. 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 4.3 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.

4.6. 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.

4.6.1. Команды для работы с устройствами и драйверами

В таблице 4.4 приведены некоторые команды, которые помогут вам обнаружить аппаратную проблему - проблему с устройством, либо с его драйвером.

Таблица 4.4. 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

4.6.2. Команды настройки сетевых интерфейсов

Подробно настройка сети будет рассматриваться в следующей главе, а пока рассмотрим таблицу 4.5, в которой представлен короткий список команд, которые вам могут пригодиться при настройке сети.

Таблица 4.5. 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

4.6.3. Программы тестирования и настройки жесткого диска

Команды для тестирования и настройки жесткого диска, подобно ранее приведенным командам, также представлены в виде таблицы - табл. 4.6.

Таблица 4.6. 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

4.7. Команды обработки текста

4.7.1. 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
```

4.7.2. 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

4.7.3. 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. 4.7.

Таблица 4.7. 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

4.7.4. 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. 4.8.

Таблица 4.8. 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

4.7.5. 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. 4.9.

Таблица 4.9. 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 4.10.

Таблица 4.10. 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

4.7.6. 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.

4.7.7. 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
```

4.7.8. 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. 4.11.

Таблица 4.11. 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)

4.7.9. 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
```

4.7.10. 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
```

4.7.11. 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.

4.7.12. Команда unexpand

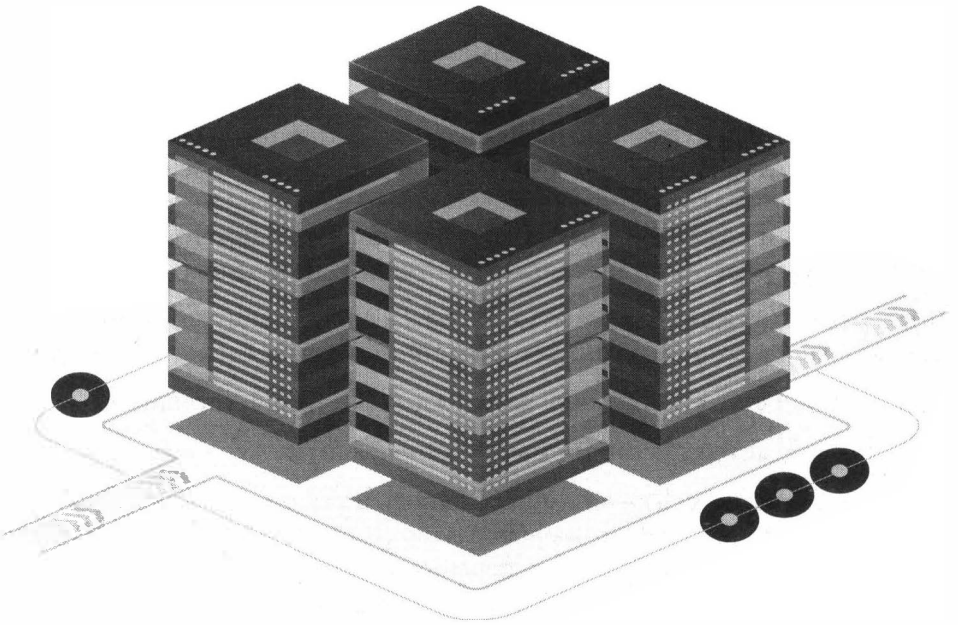
Заменяет последовательные пробелы символами табуляции. По умолчанию 8 пробелов заменяются одним символом табуляции. Количество пробелов можно задать с помощью параметра -t n (где n — количество пробелов).

Синтаксис вызова:

```
unexpand [параметры] файл
```

Глава 5.

Настройка сетевых интерфейсов



5.1. 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. 5.1), вы без проблем сможете обжать кабель:

Таблица 5.1. 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.

5.2. 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. 5.1). Выберите команду **Параметры соединения**, чтобы открыть конфигуратор сети.

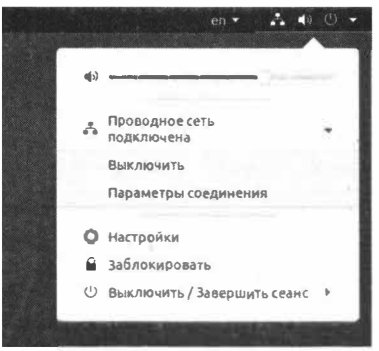


Рис. 5.1. Системное меню

В появившемся окне **Сеть** нажмите кнопку с изображением шестеренки напротив строки **Подключено** – 1000 Мбит/с (рис. 5.2).

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**).

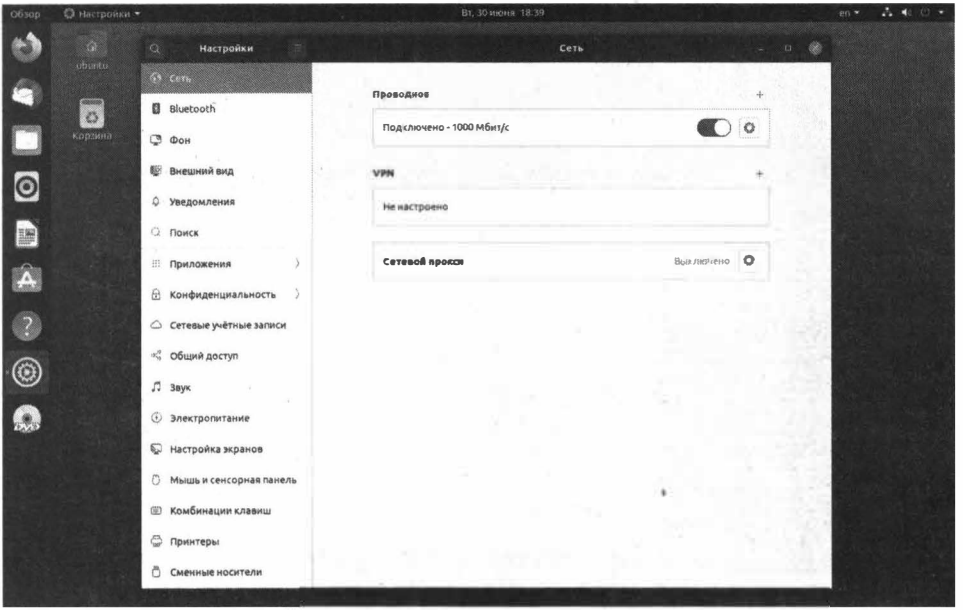


Рис. 5.2. Окно "Сеть"

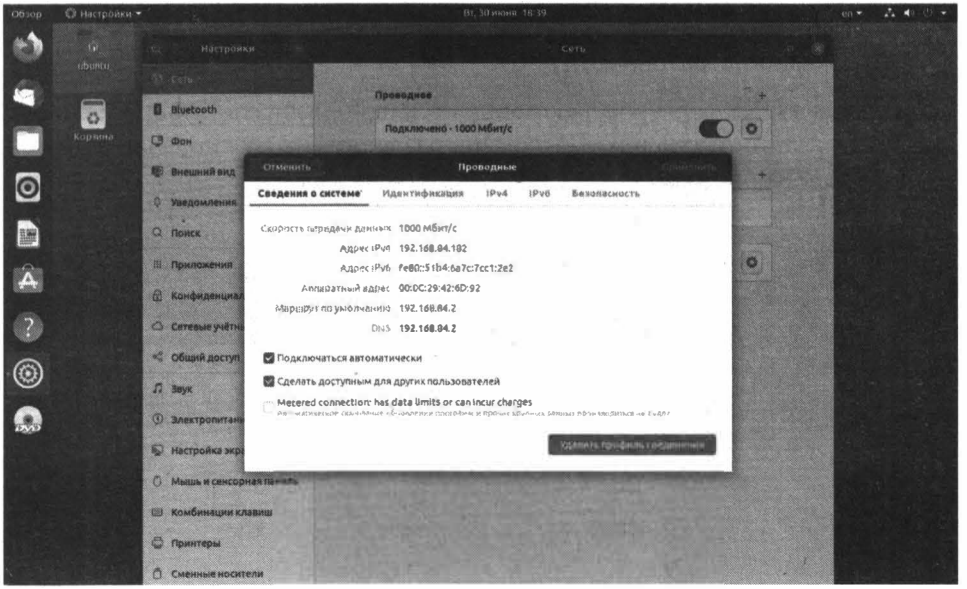


Рис. 5.3. Сведения о системе

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

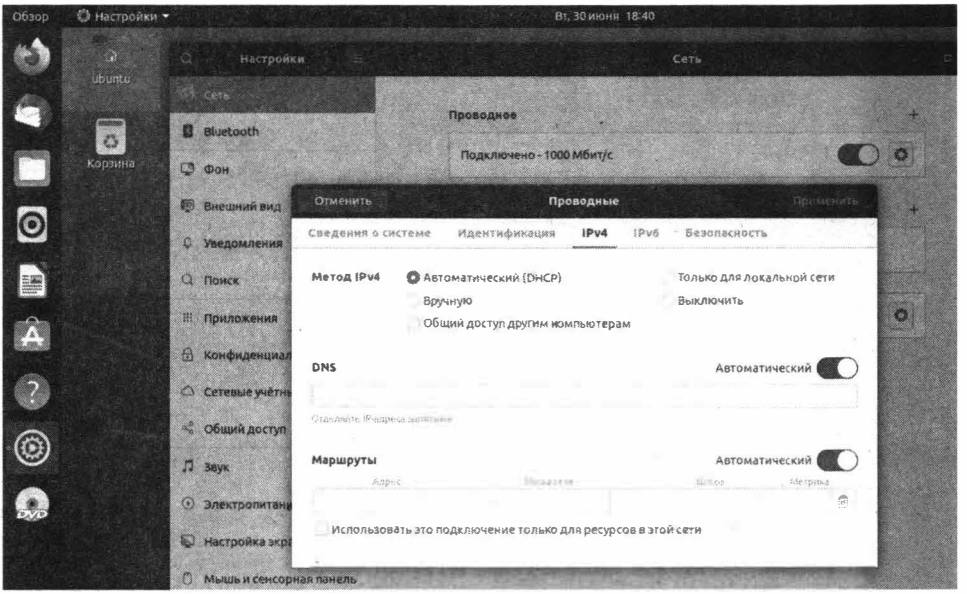


Рис. 5.4. Используем DHCP

5.4). Для ручной настройки выберите **Вручную** и укажите (рис. 5.5):

- 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. 5.5.

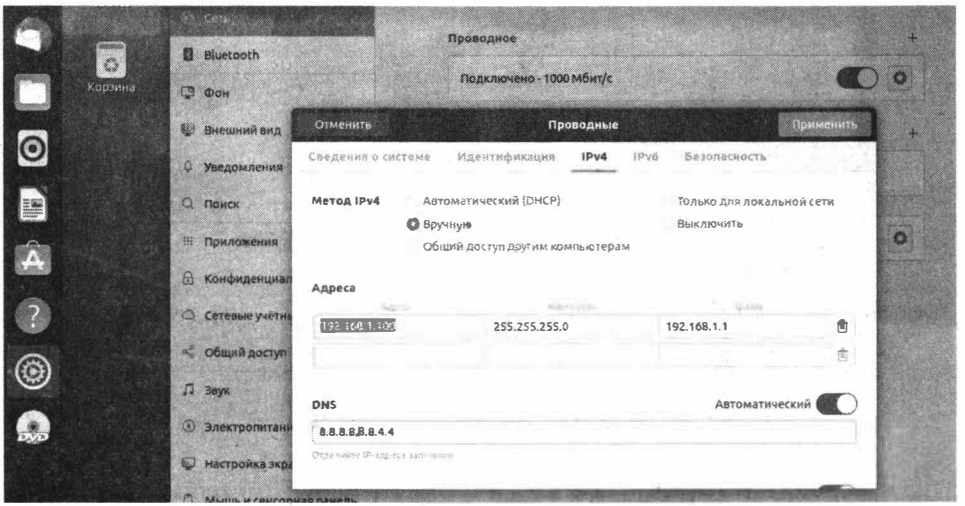


Рис. 5.5. Ручная настройка

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. 5.6). В появившемся окне выберите сетевое соединение и нажмите кнопку с изображением шестеренки (рис. 5.7).

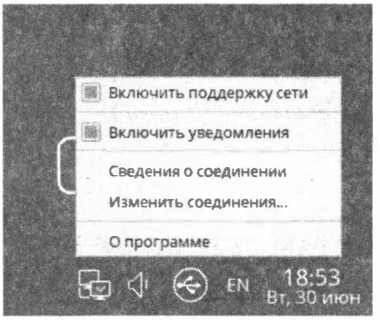


Рис. 5.6. Вызов конфигуратора сети в Astra Linux

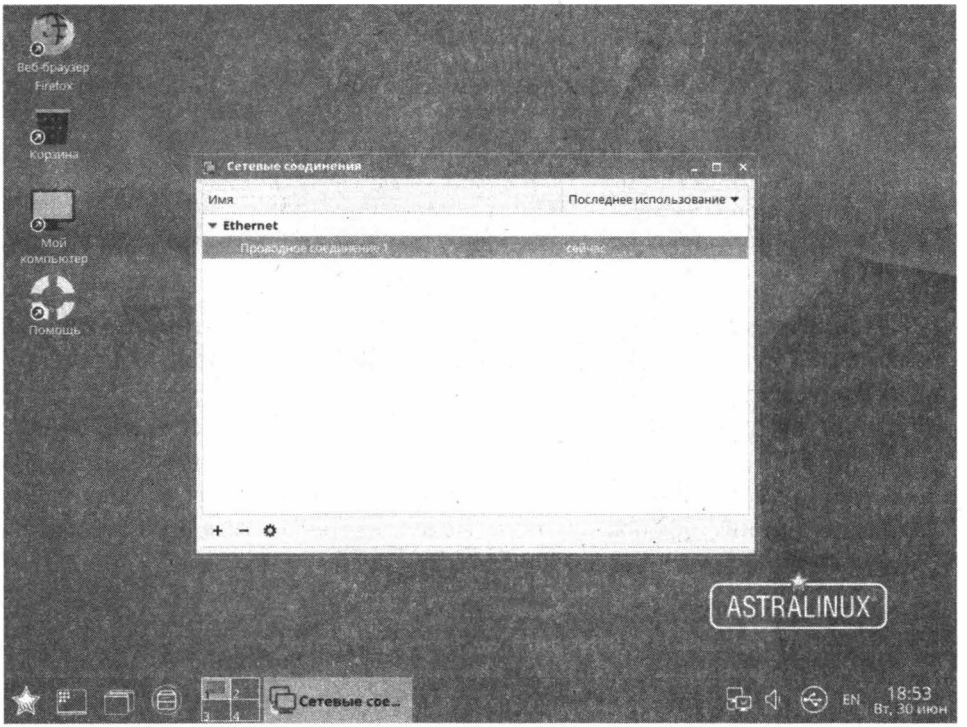


Рис. 5.7. Конфигуратор сети

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**.

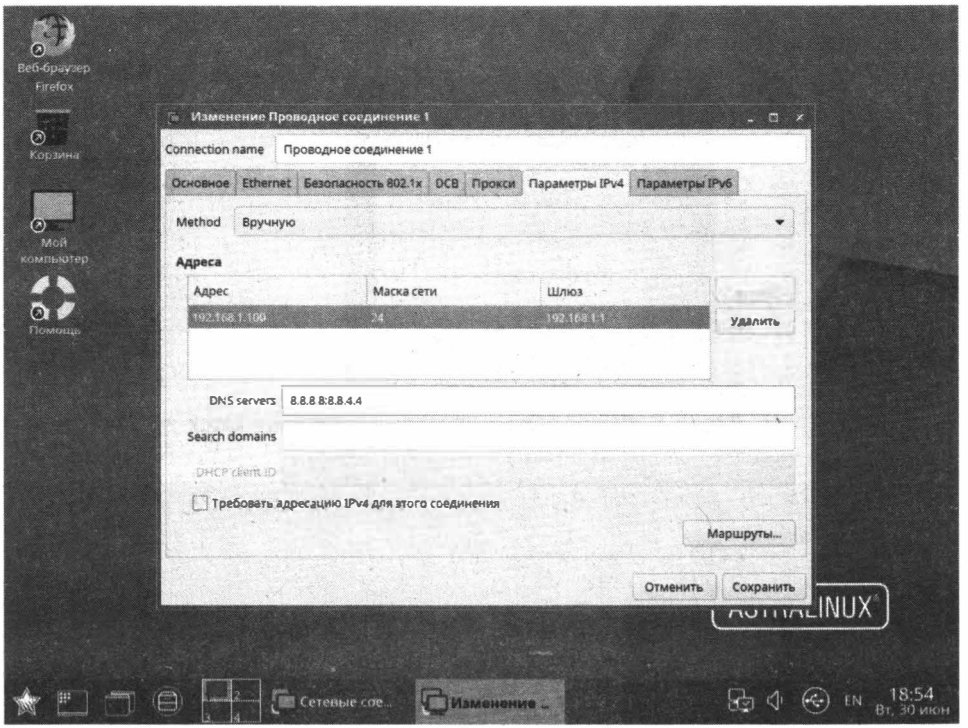


Рис. 5.8. Параметры IPv4

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. 5.9):

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

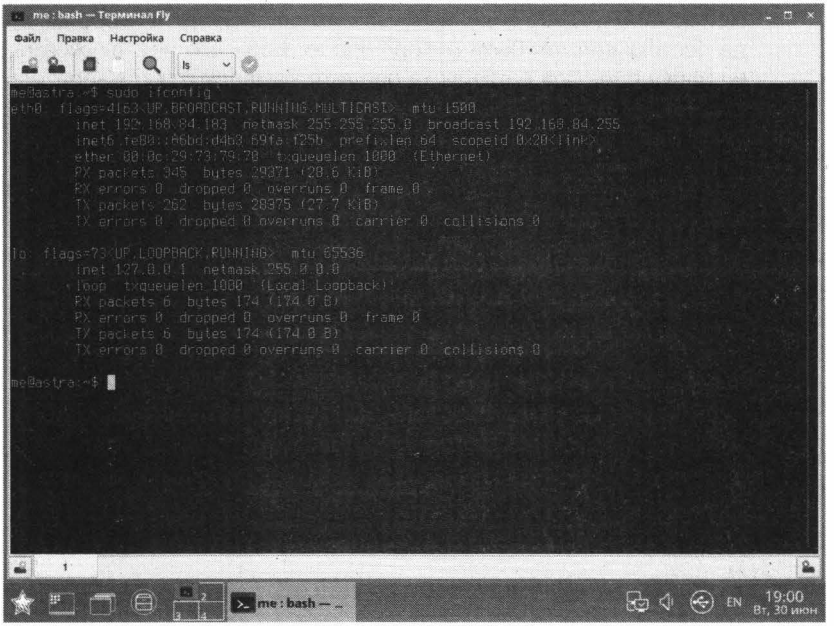


Рис. 5.9. Вкладка Ethernet в Astra Linux

5.3. 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:

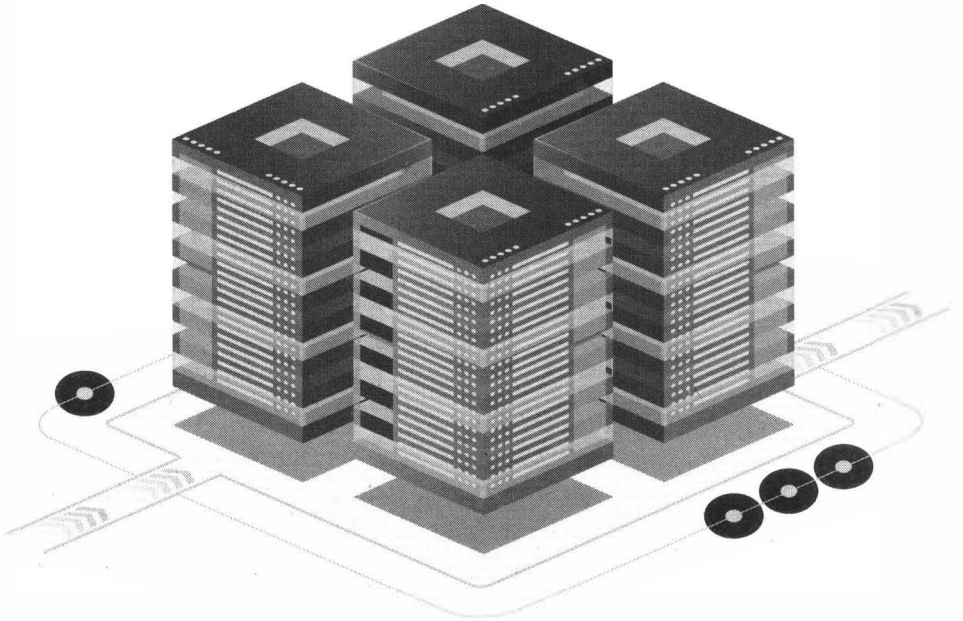
```
sudo ifconfig ens33 down
sudo ifconfig ens33 up
```

Часть II.

Базовая настройка сервера

Глава 6.

Маршрутизация и настройка брандмауэра



6.1. 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. 6.1. Здесь видно, что шлюз по умолчанию - 192.168.52.2. По сути, это самая простая таблица маршрутизации, какая только может быть.

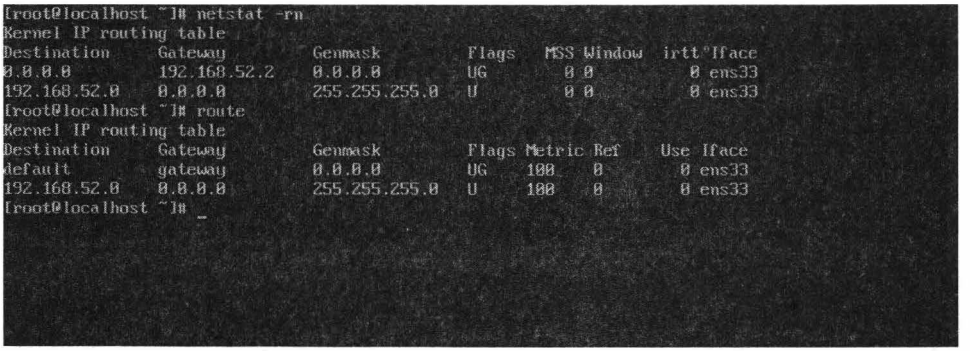


Рис. 6.1. Команды `netstat -rn` и `route`

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.

6.2. 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 6.1 приводится пример этого файла.

Листинг 6.1. 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. 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
```

```
route add -net 192.168.14.0 netmask 255.255.255.0 gw
192.168.17.254 eth0
route add -net 192.168.22.0 netmask 255.255.255.0 gw
192.168.17.254 eth0
```

Тогда в файл /etc/network/interfaces нужно добавить следующие строки:

```
up route add -net 192.168.1.0 netmask 255.255.255.0 gw
192.168.17.254 eth0
up route add -net 192.168.12.0 netmask 255.255.255.0 gw
192.168.17.254 eth0
up route add -net 192.168.21.0 netmask 255.255.255.0 gw
192.168.17.254 eth0
```

В этом примере пакеты ко всем трем сетям направляются на шлюз 192.168.17.254, а он уже сам решает, что с ними делать. Прежде, чем мы рассмотрим настройку брандмауэра, нужно отметить, как превратить компьютер в шлюз. Для этого нужно разрешить пересылку пакетов протокола IPv4 (IPv4 forwarding). Для этого добавьте значение 1 в файл /proc/sys/net/ipv4/ip_forward:

```
# echo 1 > /proc/sys/net/ipv4/ip_forward
```

Конечно, после перезапуска это значение будет потеряно. Чтобы его сохранить, добавьте в файл /etc/sysctl.conf следующую строку:

```
net.ipv4.ip_forward=0
```

Итак, ваш компьютер теперь превращен в шлюз. Но сам по себе перенаправлять пакеты он не будет. Нужно задать ряд правил перенаправления пакетов. Для этого мы будем использовать брандмауэр iptables, который имеется во всех современных дистрибутивах. Именно о нем и пойдет речь в следующем разделе. Забегая наперед, скажу, что iptables можно использовать не только для настройки шлюза, но и просто для защиты сервера или рабочей станции от нежелательных подключений.

В новых версиях Ubuntu (начиная с версии 18.04) настройка маршрутов осуществляется через файл /etc/netplan/01-netcfg.yaml. Откройте этот файл:

```
sudo nano /etc/netplan/01-netcfg.yaml
```

В нем вы найдете конфигурацию по умолчанию, которая может быть примерно 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
```

6.3. 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.

6.3.1. 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.

6.3.2. 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.

6.3.3. 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 6.1 указываются возможные действия, которые вы можете использовать с iptables.

Таблица 6.1. 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. 6.2).

Таблица 6.2. 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 6.2 указаны далеко не все параметры, но для организации собственного шлюза представленных параметров будет более чем достаточно.

6.3.4. 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. 6.2).

Листинг 6.2. 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 6.3.

Листинг 6.3. 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.

6.4. 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.

6.4.1. 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.

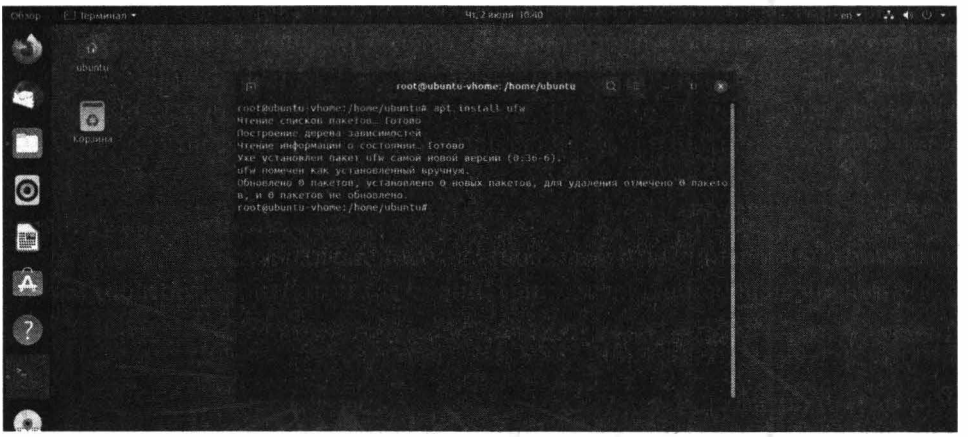


Рис. 6.2. Установлена самая новая версия ufw

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!

6.4.2. 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.).

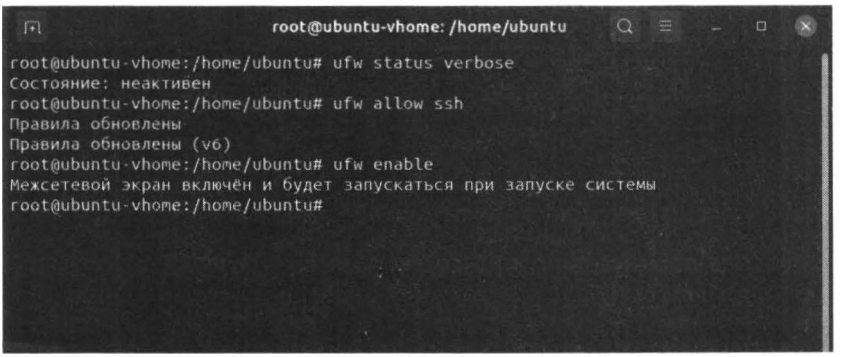


Рис. 6.3. Процесс настройки ufw

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. 6.3).

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.

6.4.3. 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
```

6.4.4. 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
```

6.4.5. 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
```

6.4.6. 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
```

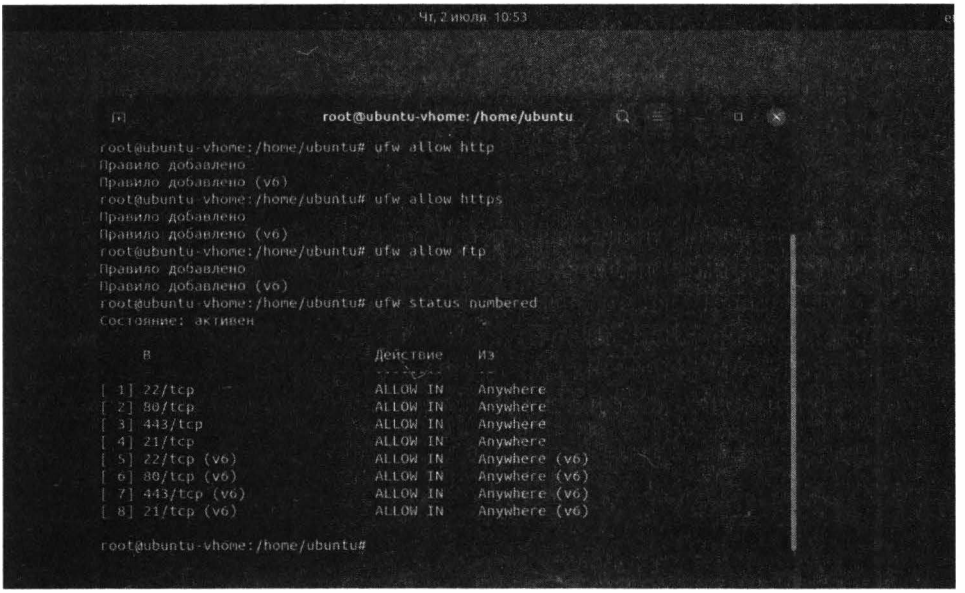


Рис. 6.4. Список правил

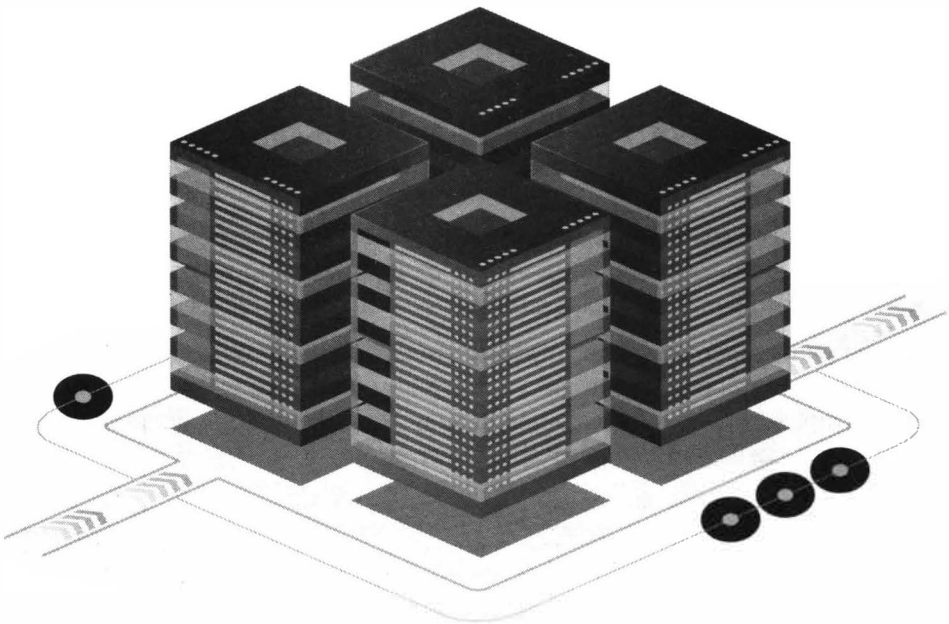
6.4.7. Отключение файрвола

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 7.

Удаленный вход в систему по SSH



7.1. 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.

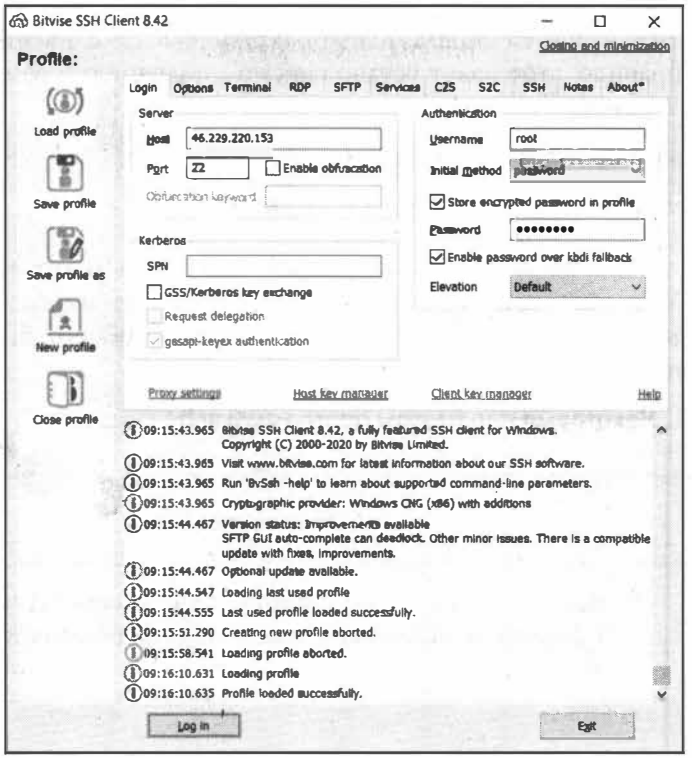


Рис. 7.1. Приложение Bitvise SSH Client

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.

7.2. 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. 7.1). Хочу предупредить сразу: опций у программы много и в таблице 7.1 будут представлены только самые полезные или наоборот, бесполезные (чтобы привлечь к ним ваше внимание) опции.

Таблица 7.1. 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
```

7.3. 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 7.1 (с комментариями на русском языке).

Листинг 7.1. 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.

7.4. 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/
```

7.5. 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
```

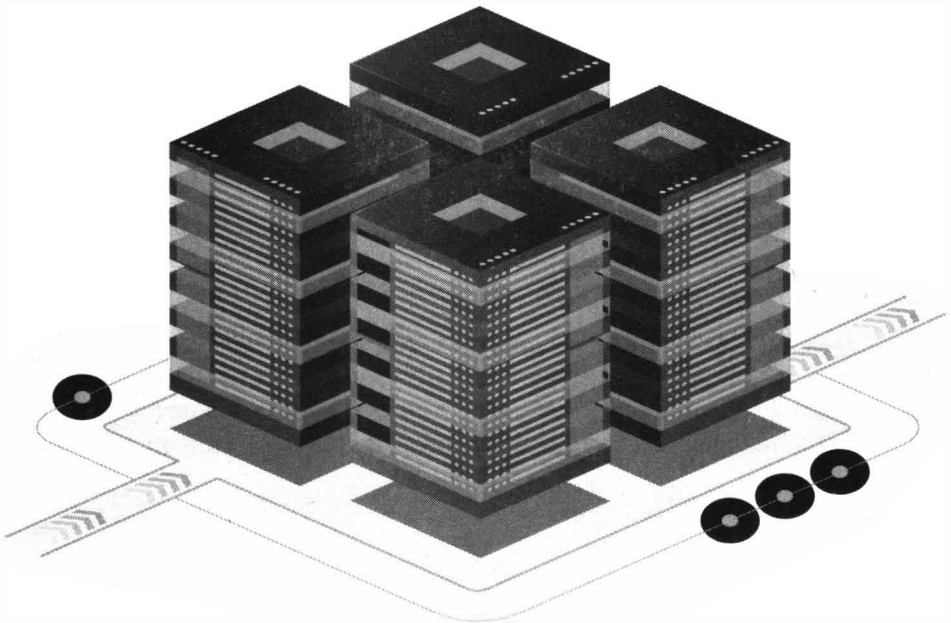
Серьезное «торможение» (порой на несколько секунд) могут добавить PAM-модули. Но отключать PAM-модули полностью не нужно! Достаточно из файлов /etc/pam.d/login и /etc/pam.d/sshd нужно удалить строки:

```
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 8.

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.

8.1. 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.

8.2. 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 8.1. 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.

8.3. 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?

8.4. 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.

8.5. 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).

8.5.1. 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.

8.5.2. 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. 8.1. Копирование файлов на локальный компьютер

8.5.3. 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.

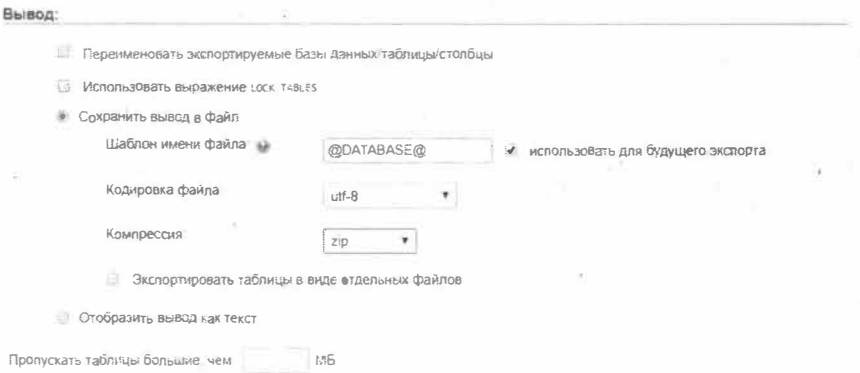


Рис. 8.2. Экспорт базы данных

8.5.4. 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.

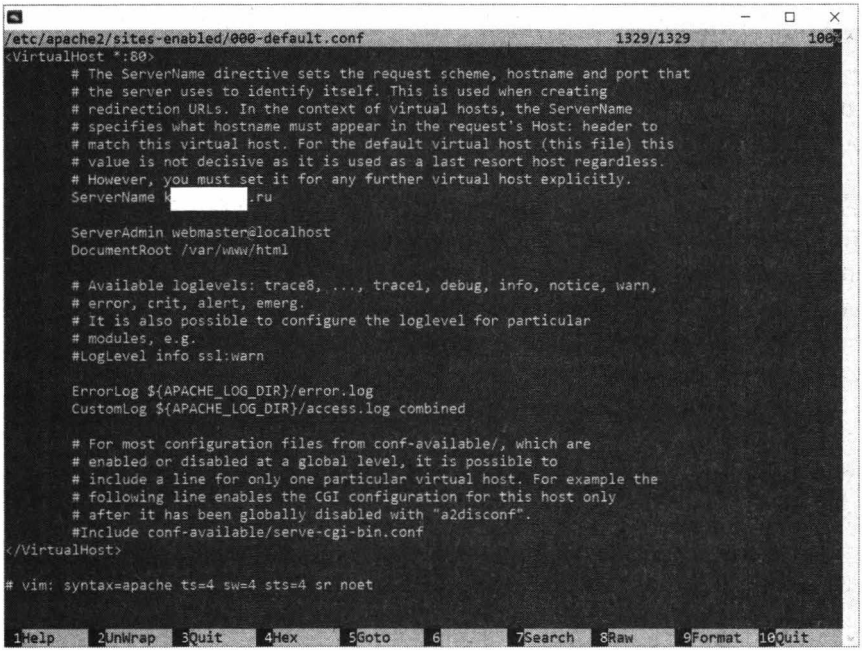


Рис. 8.3. Конфигурация сервера: имя сервера затерто (поскольку производилась настройка реального сервера и его имя скрыто в интересах клиента)

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
```

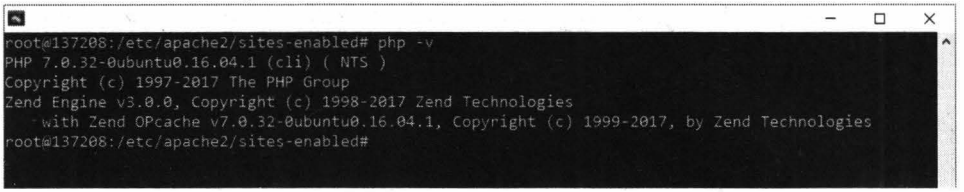


Рис. 8.4. Версия интерпретатора

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.

8.5.5. 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. 8.5. 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).

8.5.6. 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.

8.5.7. 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.

8.5.8. Перенос доменного имени

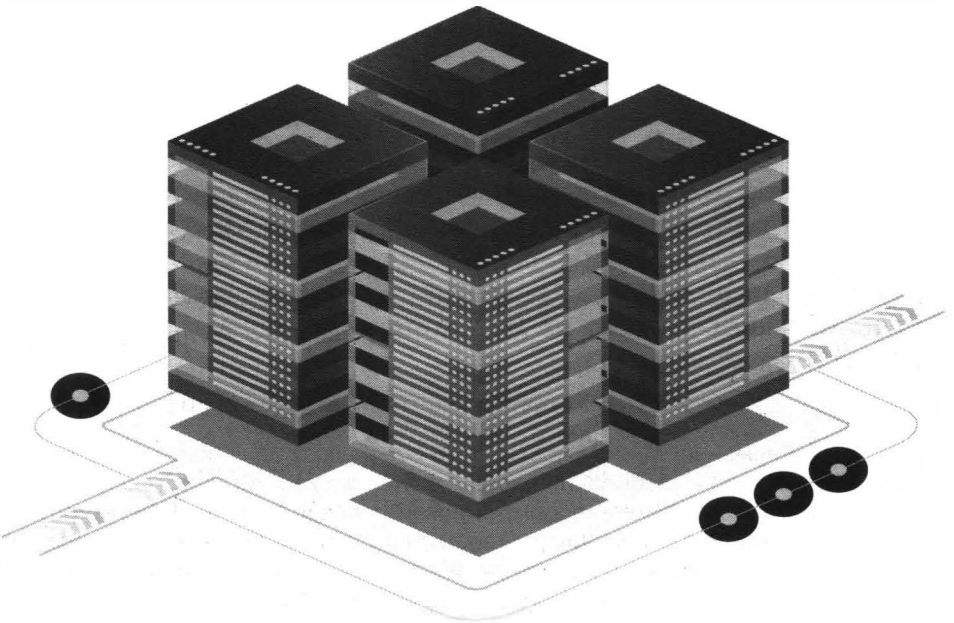
Осталось самое малое – открыть панель управления доменным именем. Если вы покупали домен вместе с хостингом, скорее всего, это будет панель управления хостингом. Если вы покупали домен у регистратора (например, на reg.ru), зайдите в личный кабинет. В настройках домена у вас сейчас прописаны NS-записи – 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 9.

Файловый сервер FTP



9.1. 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.

9.2. Универсальный солдат - ProFTPD

9.2.1. 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.

9.2.2. 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 9.1 и свертесьсверьтесь с вашим файлом конфигурации. В листинге 9.1 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 9.1. 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>
```

9.2.3. 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>
```

9.2.4. 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].

9.3. 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 9.2 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 9.2. 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
```

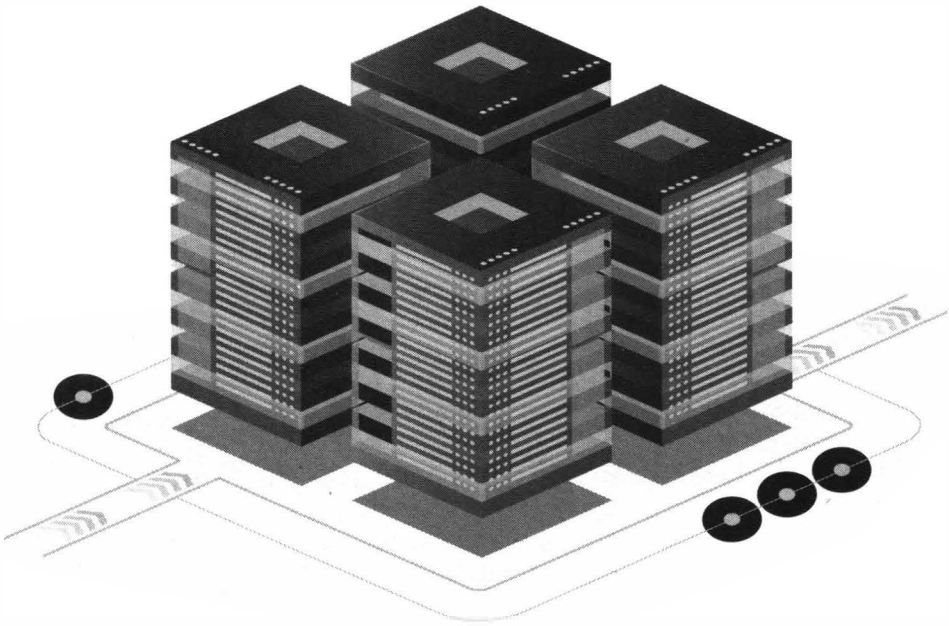
```
# Максимальная скорость передачи данных (в байтах/сек.)
# 0 - без ограничения
local_max_rate=7200

# 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 10.

Доменная система имен



10.1. 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.

10.2. 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 10.1 представлен листинг /etc/unbound/unbound.conf, настраивающий Unbound на работу в кэширующем режиме.

Листинг 10.1. 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.

10.3. 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. 10.2).

Листинг 10.2. 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 10.3.

Листинг 10.3. 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. 10.4).

Листинг 10.4. 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.

10.4. 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. 10.5.

Листинг 10.5. 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. 10.6).

Листинг 10.6. 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 10.6 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
```

10.5. 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; };
};
```

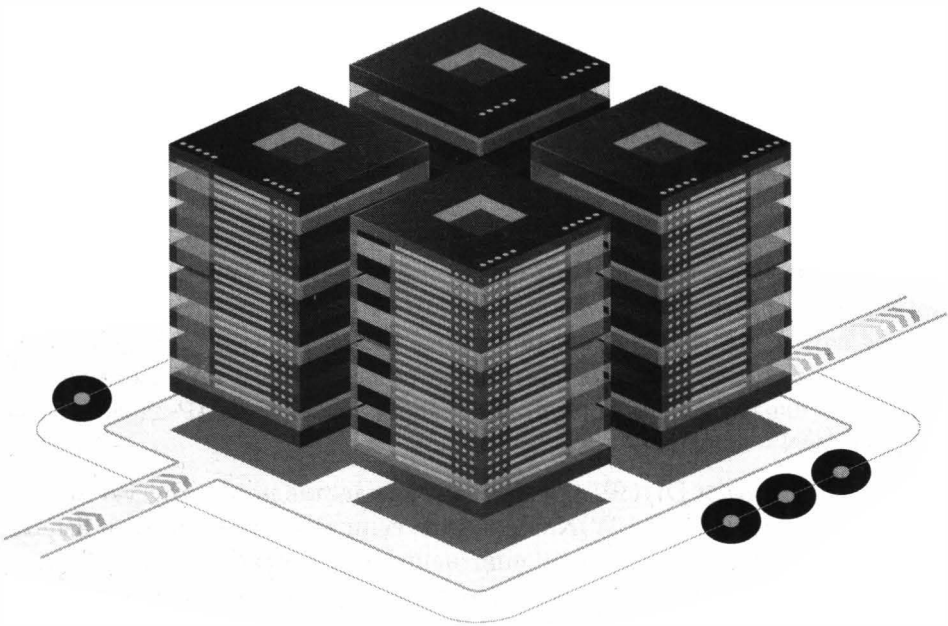
На первичном сервере в блоке options нужно добавить блок allow-transfer, в котором указывают IP-адрес вторичного DNS-сервера:

```
options {
...
allow-transfer { 192.168.1.2; };
}
```

Вот теперь действительно все.

Глава 11.

DHCP-сервер



11.1. 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.

11.2. 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.

11.3. 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 11.1 приводится пример файла dhcpd.conf для небольшой сети с несложной топологией.

Листинг 11.1. 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).

11.4. 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 11.2.

Листинг 11.2. 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.

11.5. 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.

11.6. 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. 11.1. 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.

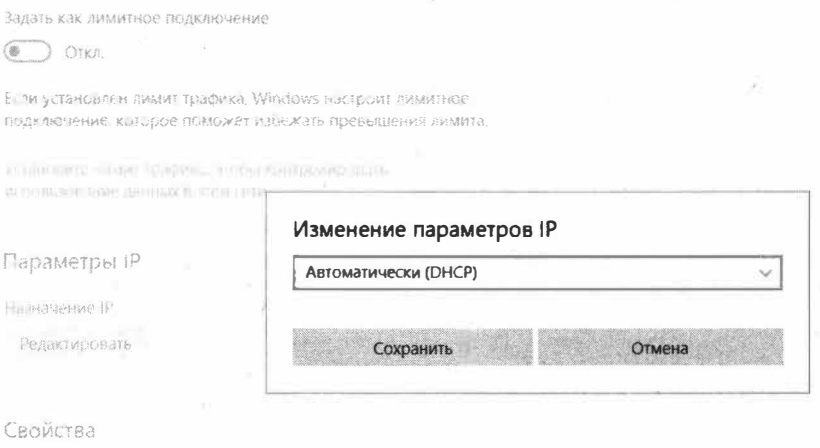
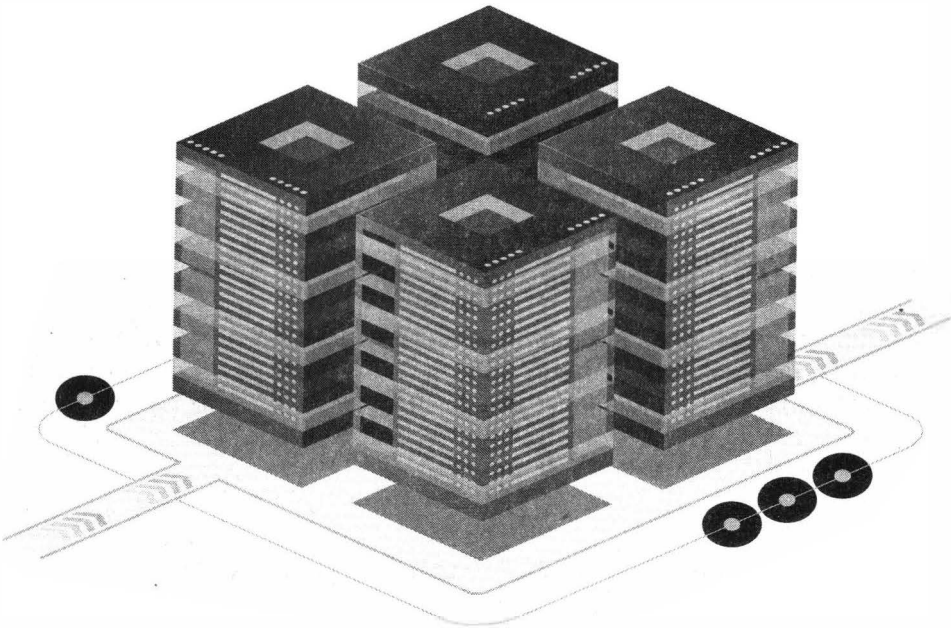


Рис. 11.1. Настройка DHCP-клиента в Windows

Глава 12.

Подключаем Linux к Windows-инфраструктуре



12.1. 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.

12.2. 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

12.3. 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
```

12.4. 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 12.1. Строки, выделенные жирным, необходимо заполнить своими данными.

Листинг 12.1. 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.

12.5. 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 12.2 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 12.2. 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.

12.6. 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
```

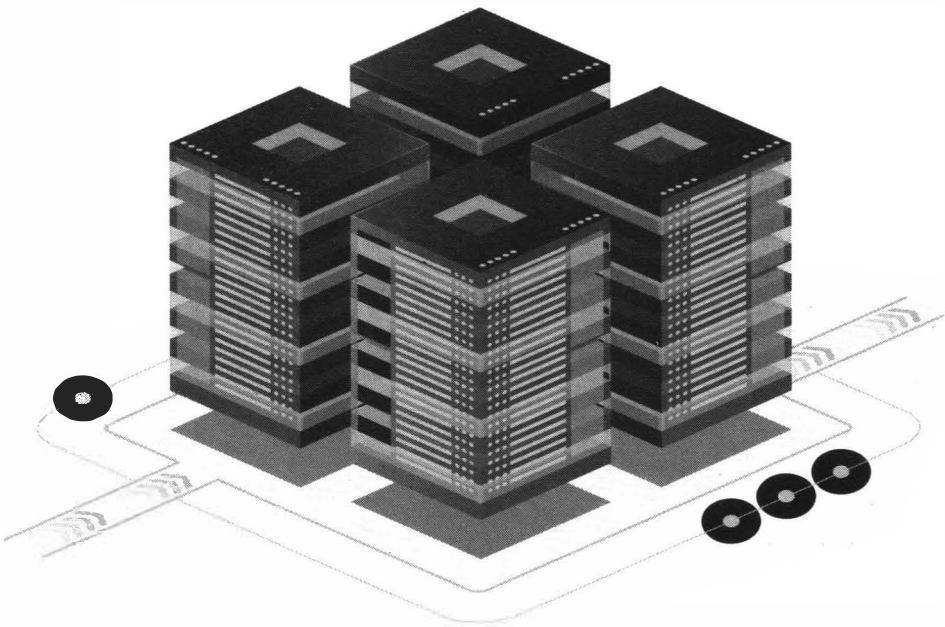
Эти строки нужно изменить так:

```
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 13.

Резервное копирование



13.1. 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.

13.2. 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.

13.3. 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 c, 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 13.1)

Листинг 13.1. 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.

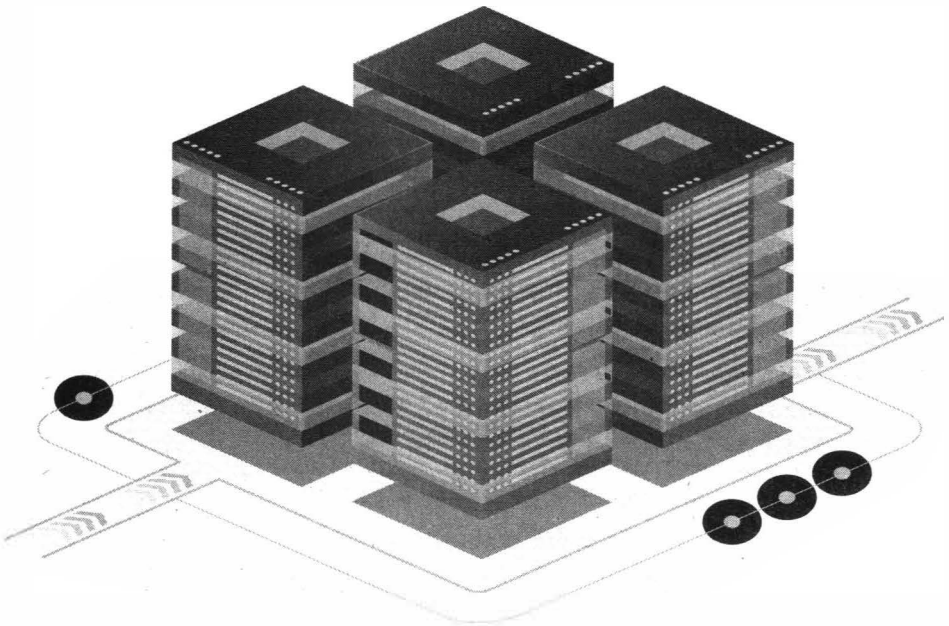
Когда вы убедитесь, что все работает, как нужно, добавьте вызов сценария /bin/backup на вашем веб-сервере в таблицу расписания планировщика (см. гл. 16).

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 14.

Обеспечение безопасности сервера



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 этой главе мы поговорим о том, как настроить брандмауэр, чтобы защи-

тить 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.

14.1. 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. 14.1.

Рис. 14.1. 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. 14.2). Все, теперь вы можете делать с системой все, что вам захочется.

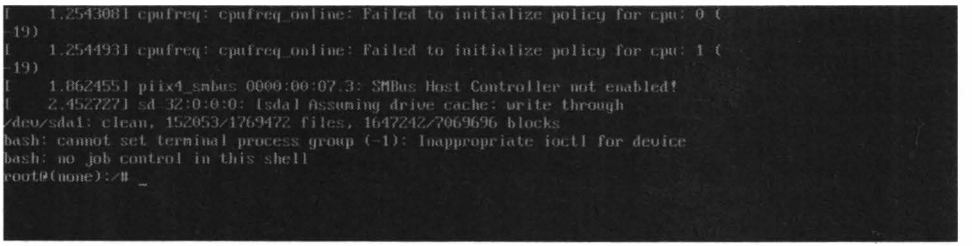


Рис. 14.2. Получены максимальные права

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. Данный пароль попросту не позволит редактировать параметры ядра и подменить систему инициализации.

14.2. 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)

14.2.1. 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.

14.2.2. 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.

14.2.3. 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.

14.2.4. 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.

14.2.5. 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.

14.2.6. 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
```

14.2.7. 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
```

14.2.8. 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
```

14.2.9. 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
```

14.2.10. 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. Защитить-ся 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.

14.3. 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
```

14.4. 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.

14.5. 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.

14.5.1. 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
```

14.5.2. 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 14.1):

```
cd ~/clients
touch make_config
chmod +x make_config
mcedit make_config
```

Листинг 14.1. 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.

14.5.3. 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.

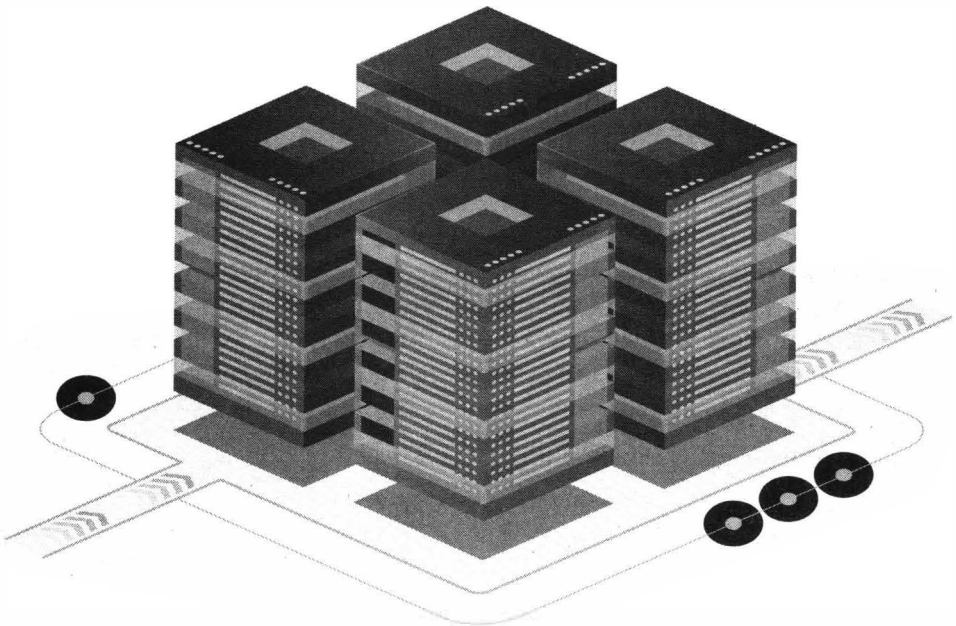
После запуска OpenVPN он должен автоматически увидеть ваш профиль. Щелкните на пиктограмме клиента на панели быстрого запуска правой кнопкой мыши и выберите команду **Подключиться**. Все достаточно просто.

ЧАСТЬ III.

ВЫДЕЛЕННЫЙ ВЕБ-СЕРВЕР НА ОСНОВЕ LINUX

Глава 15.

Настройка веб-сервера



В данной главе будет произведена установка программного обеспечения, необходимого для реализации веб-сервера. По сути, нам необходима связка Apache + PHP + MySQL. В сети очень много руководств по организации такой связки. Повторяться и писать еще одно подобнее руководство не хочется, тем более все равно они не раскрывают всех нюансов подобной установки. А нюансы могут заключаться в том, что устанавливаемая CMS не поддерживает версию PHP, которую вы установили. И узнаете вы об этом только тогда, когда будете устанавливать CMS и она не пройдет проверку системных требований. В этой главе мы подготовим наш веб-сервер под установку CMS Magento. Это довольно требовательная CMS: разные ее версии требуют разных версий PHP, а также кроме базовых пакетов она требует множества расширений PHP.

15.1. Установка и настройка Apache

Администратор может использовать в качестве веб-сервера или Apache или nginx. Выбор зависит от предпочтений администратора. В этой книге мы будем использовать Apache, при необходимости всегда можно перенастроить сервер на работу с nginx (хотя некоторое внимание nginx мы все же уделим – там где вопросы касаются нетривиальной настройки).

Установить веб-сервер можно так:

```
sudo apt install apache2
```

Здесь мы устанавливаем пакет apache2. Команда sudo подразумевает использование root-прав. Если вы вошли на свой сервер уже как пользователь root, тогда ее вводить не нужно и команда будет сокращена до следующей:

```
apt install apache2
```

Команда apt – это менеджер пакетов, использующийся в Ubuntu и Debian. Если же вы используете CentOS или Fedora, то вместо нее нужно использовать команду dnf:

```
dnf install apache2
```

После установки нужно настроить сервер. Как минимум, вам нужно указать, что ваш сервер будет использовать купленное ранее доменное имя.

Первым делом нужно зайти в панель управления доменным именем и добавить A-запись для вашего домена, указывающую на IP-адрес вашего VDS-сервера. Как это сделать? Проще всего обратиться в службу технической поддержки регистратора, у которого вы покупали доменное имя, поскольку регистраторов очень много и все они часто используют панели управления собственной разработки, поэтому невозможно в книге привести руководство абсолютно для всех регистраторов. Тогда бы книга называлась «Как управлять доменом».

Далее желательно установить файловый менеджер Midnight Commander, который позволит производить операции с файлами более удобно:

```
sudo apt install mc
```

Перейдите в каталог /etc/apache2/sites-enabled и откройте файл 00-default.conf. Мы не будем создавать отдельные веб-серверы для каждого сайта. Будем считать, что у нас есть один сайт и его конфигурация как раз будет храниться в файле 00-default.conf. Реальный пример конфигурации приведен на рис. 15.1.

Основные директивы:

- ServerName – здесь нужно указать купленное доменное имя;
- DocumentRoot – указывает, где будут храниться файлы сайта, мы используем каталог по умолчанию /var/www/html
- ServerAdmin – можно указать адрес электронной почты администратора сервера.

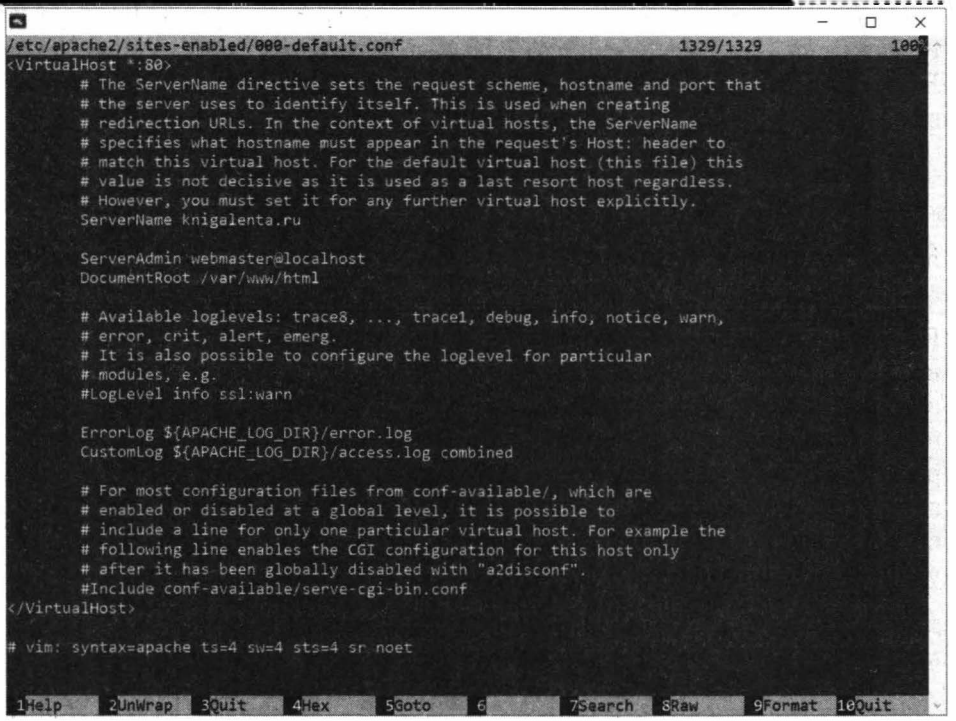


Рис. 15.1. Конфигурация сервера

Мы сейчас не будем подключать SSL-сертификат, а сделаем это в главе 17.

Осталось запустить сервис веб-сервера:

```
sudo systemctl start apache2.service
```

или

```
sudo service apache2 start
```

Первая команда сработает в более современных дистрибутивах, вторая будет работать даже в относительно древних, хотя она с успехом, как показывает практика, работает и в новых версиях Ubuntu – она оставлена по соображениям обратной совместимости, поскольку уж очень нравится администраторам.

Аналогично, для перезапуска и останова сервиса можно использовать следующие команды:

```
sudo systemctl restart apache2.service
sudo systemctl stop apache2.service
```

или

```
sudo service apache2 restart
sudo service apache2 stop
```

15.2. Установка сервера баз данных. Создание базы данных и пользователя

Для установки СУБД MySQL нужно ввести 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:

1. 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.
2. Удаление тестовой базы данных.
3. Запрет подключения к серверу баз данных извне, только с локального узла. Это означает, что к СУБД сможет подключиться ПО, работающее только на этом сервере, а не все желающие. Не беспокойтесь: к вашему магазину 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 magento2;
CREATE USER magento2user@localhost IDENTIFIED BY 'password';
GRANT ALL PRIVILEGES ON magento2.* TO magento2user@localhost
IDENTIFIED BY 'password';
FLUSH PRIVILEGES;
exit
```

Здесь запускается клиент и от имени MySQL-пользователя root выполняются запросы. Первый запрос создает базу данных magento2, второй – создает пользователя magento2user, от имени которого Magento будет обращаться к СУБД. Третий запрос – предоставление полномочий пользователю magento2user ко всем таблицам базы данных magento2. Естественно, вместо 'password' укажите какой-то сложный пароль.

Последний запрос осуществляет применение полномочий, а команда exit – выход из клиента mysql.

15.3. Установка и настройка PHP. Выбор версии

Самое дело остановиться и подумать. Если вы производите установку на виртуальный сервер, сделайте снапшот сервера, чтобы можно было легко откатиться к предыдущему состоянию. Снапшот можно сделать в панели управления VDS, которую вам предоставил ваш облачный провайдер.

Нужно решить, какую версию PHP установить (табл. 15.1). На данный момент существует четыре актуальных версии – 7.0, 7.1, 7.2, 7.3. Дело в том, что текущая версия Magento 2.3 требует версии PHP 7.1.3 – 7.3. Предыдущая версия Magento – 2.2 – изначально не поддерживала 7.2, ей нужна или 7.0 или 7.1. Поддержка версии 7.2 появилась лишь в версии 2.2.10.

Дополнительная информация доступна по ссылкам:

```
https://devdocs.magento.com/guides/v2.3/install-gde/system-
requirements-tech.html
https://devdocs.magento.com/guides/v2.2/install-gde/system-
requirements-tech.html
```

Таблица 15.1. Magento и выбор версии PHP

Версия Magento	7.0.0 – 7.0.12	7.0.13 – 7.0.x	7.1.x	7.2.x	7.3.x
2.2	-	+	+	патч 2.2.10	-
2.3	-	-	7.1.3+	+	+

Казалось бы, что тут такого – нужно установить Magento 2.3 и PHP 7.2 или 7.3 (смотря какая версия есть в дистрибутиве), ведь мы устанавливаем все с нуля. Но это неправильное решение.

Может оказаться так, что нужное вам расширение или тема оформления не поддерживает версию 2.3. Когда я установил Magento 2.3 и подключил MarketPlace (https://marketplace.magento.com), то обнаружил, что в списке расширений не было необходимого мне расширения для моей CRM.

Именно поэтому сначала нужно выбрать шаблон для вашего сайта и составить список расширений, которые вам будут нужны. Возможно, это будут расширения, добавляющие необходимые вам способы оплаты/доставки (интеграция с платежной системой и службой доставки товаров), производящие интеграцию с CRM и т.д.

Составьте табличку совместимости (табл. 15.2). Например, в моем случае она выглядела так

Таблица 15.2. Таблица совместимости

Расширение/шаблон	Magento 2.1	Magento 2.2	Magento 2.3
SM Destino	x	x	x
retailCRM	x	x	-

Поставьте крестик в поле, если шаблон/расширение поддерживает версию Magento. Как правило, вся необходимая информация есть в документации

по расширению/шаблону. Также обратите внимание на младшую версию Magento, например, в документации по шаблону может быть сказано, что он поддерживает версию 2.2.6, а вот с сайта Magento в данный момент доступна для загрузки версия 2.2.7. Лучше не рисковать, а перейти в архивы и скачать версию 2.2.6. Конечно, на момент разработки шаблона могло просто не быть версии 2.2.7 или же разработчик не тестировал свой шаблон в этой версии, но вы хотите потратить несколько часов времени, чтобы это проверить?

Теперь, когда вы выбрали версию PHP, осталось ее установить. Некоторые руководства предлагают устанавливать ее так (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
```

Здесь устанавливается версия PHP по умолчанию для вашего дистрибутива. Например, для Ubuntu 16.04 – это будет 7.0, для Ubuntu 18.04 – PHP 7.2 и т.д.. Но правильнее указывать номера версий необходимых вам пакетов. Конечно, это усложнит задачу, но тем не менее.

```
sudo apt install php7.0 php7.0-cli openssl libcurl3 php7.0-
curl php7.0-gd php7.0-mcrypt php7.0-xml php7.0-intl php7.0-
zip php7.0-mbstring php7.0-soap php7.0-mysql php7.0-json
libapache2-mod-php php7.0-xsl php7.0-bcmath php7.0-ldap
composer
```

Другими словами, после «php» нужно указать нужную версию. Кроме самого PHP данная команда установит все необходимые для работы Magento расширения PHP, composer (также нужен для Magento), OpenSSL и библиотеку libcurl3 (или libcurl4 для PHP 7.2). Если какие-то расширения понадобятся, вы об этом узнаете в процессе установки Magento и сможете их доустановить.

Примечание. Расширение php-mcrypt удалено из PHP 7.2, поэтому устанавливать его не нужно – вы просто не найдете необходимого пакета.

Внимательные читатели заметили два факта. Первый – какая версия 7.0 будет установлена в Ubuntu по умолчанию? Ведь нужна версия 7.0.13 или более новая? Все в порядке, по умолчанию на данный момент устанавливается версия 7.0.32. Узнать версию можно командой:

```
php -v
```

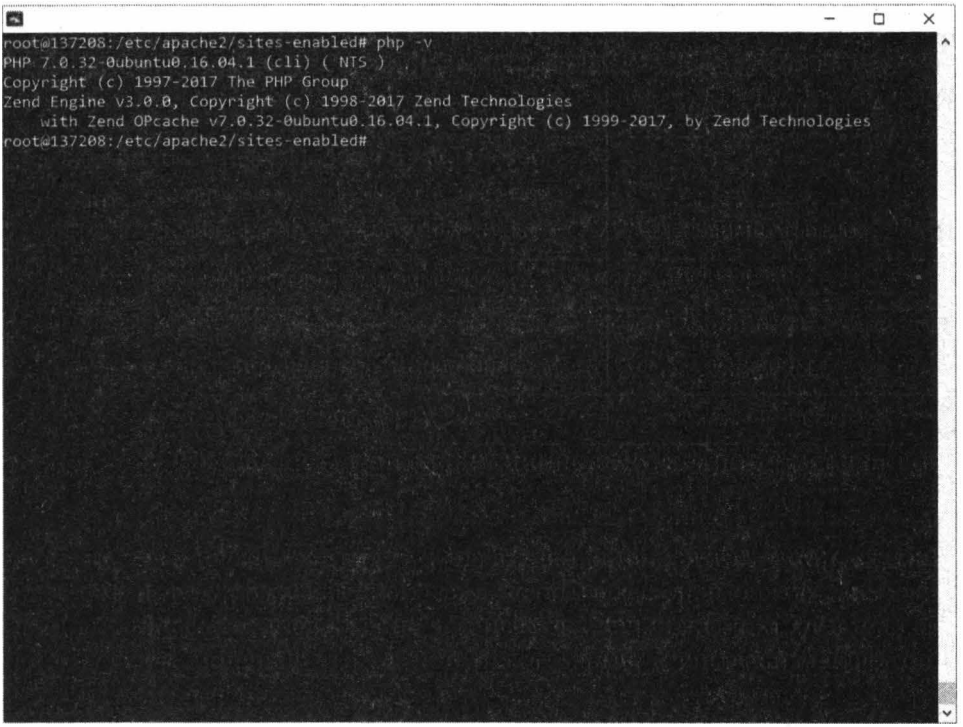


Рис. 15.2. Версия PHP

Что делать, если вам нужна версия 7.2, но ее нет в вашем дистрибутиве? Тогда введите следующие команды:

```
sudo apt-get install software-properties-common
sudo add-apt-repository ppa:ondrej/php
sudo apt-get update
```

Они подключают необходимый репозиторий и обновляют список пакетов. Теперь можно установить PHP:

```
sudo apt install php7.2 php7.2-cli
```

После установки введите команду php –v, чтобы убедиться, что установлена версия 7.2. Если все хорошо, тогда установите необходимые расширения, как было показано ранее.

Теперь нужно настроить PHP и Apache. Первым делом откройте файл конфигурации PHP (X – версия):

```
sudo mcedit /etc/php/7.X/apache2/php.ini
```

В нем нужно установить лимит памяти:

```
memory_limit = 512M
```

Можно установить и другие параметры, но об этом мы поговорим далее в этой книге, когда пойдет речь об оптимизации сервера.

Сохраните файл, выйдите из редактора и добавьте необходимые модули Apache:

```
sudo a2enmod rewrite
```

Также, чтобы нормально работали SEF URL Magento нужно открыть ваш файл конфигурации /etc/apache2/sites-enabled/000-default.conf и добавьте в секцию VirtualHost строки:

```
<Directory /var/www/html/magento_test>
Options Indexes FollowSymLinks MultiViews
AllowOverride All
</Directory>
```

Все, можно перезапускать Apache:

```
sudo service apache2 restart
```

15.4. Директивы файла конфигурации Apache

Основной файл конфигурации Apache называется /etc/apache2/apache2.conf. Конфигурация Apache просто огромна:

1. Основной файл конфигурации apache2.conf «весит» более 7 Кб и содержит 224 строки конфигурации.
2. Посредством директивы Include в конфигурацию Apache включаются другие файлы из каталога /etc/apahce2. Например, директива Include ports.conf в основном файле конфигурации говорит о том, что в месте директивы нужно вставить содержимое ports.conf. Аналогично, вы можете создать файл my.conf и вставить его содержимое с помощью директивы Include my.conf.
3. С помощью директивы IncludeOptional включается содержимое всех файлов конфигурации из каталога /etc/apache2/sites-enabled. Как правило, в этих файлах конфигурации принято хранить описание виртуальных узлов или попросту говоря сайтов. Принцип прост. Сам файл конфигурации для определенного сайта помещается в каталог sites-available (доступные сайты), а затем на него создается ссылка в каталоге sites-enabled (включенные сайты). Если сайт нужно временно отключить, то чтобы не удалять его конфигурацию, просто удаляют ссылку на файл конфигурации из каталога sites-enabled.

Теоретически, вы можете хранить всю конфигурацию в файле apache2.conf. Тогда ваш файл конфигурации будут еще больше. Вряд ли вам будет удобно, но такая возможность есть.

Директивы конфигурационного файла сервера Apache можно условно разделить на такие групы:

1. **Общие.** К общим директивам относятся глобальные директивы, влияющие на работу всего Web-сервера. Это директивы ServerName, ServerType, Port, User и Group, ServerAdmin, ServerRoot, PidFile, DocumentRoot, UserDir.
2. **Директивы протоколирования**: ErrorLog, TransferLog, HostnameLookups.
3. **Директивы ограничения доступа**: AllowOverride, Options, Limit.
4. **Директивы управления производительностью**: StartServers, MaxSpareServers, MinSpareServers, а также директива CacheNegotiatedDocs.
5. **Директивы обеспечения постоянного соединения с клиентом**: Timeout, KeepAlive, KeepAliveTimeout.
6. **Директивы настройки отображения каталога**. Оформить отображение каталогов можно с помощью директив настройки отображения каталогов: DirectoryIndex, FancyIndexing и AddIconByType.
7. **Директивы обработки ошибок**. Директивой обработки ошибок HTTP-сервера является директива ErrorDocument. С ее помощью можно установить реакцию на любую ошибку сервера, например, на ошибку 404 (документ не найден).
8. **Директивы перенаправления**: Redirect, Alias и ScriptAlias.
9. **Директивы для работы с многоязычными документами**: AddLanguage и LanguagePriority.
10. **Директивы обработки MIME-типов**. Настроить свой сервер для обработки различных MIME-типов можно с помощью директив DefaultType, AddEncoding, AddType, AddHandler и Action.
11. **Директивы создания виртуальных узлов**: VirtualHost, Listen, BindAddress.

Директив много, но вам не нужно редактировать все директивы сразу. Вы можете постепенно кастомизировать ваш веб-сервер – по мере поступления задач от руководства. Как минимум, вам нужно изменить две следующие директивы:

- **ServerName** - директива, которая определяет имя сервера Apache. Причем, здесь должно быть задано официальное имя сервера в таком виде, в котором оно должно появляться в строке адреса броузера. Данное имя должно быть зарегистрировано в сервере DNS вашей сети.
- **ServerType** - директива, которая определяет тип сервера. По умолчанию используется значение standalone. Если вы хотите достичь максимальной производительности вашего Web-сервера, не изменяйте эту опцию.

Другие полезные директивы веб-сервера приведены в таблице 15.3.

Таблица 15.3. Некоторые директивы файла конфигурации Apache

Директива	Описание
User и Group	Директивы User и Group определяют идентификаторы пользователя и группы, от имени которых будет работать сервер. Данные идентификаторы присваиваются серверу, если он запущен в автономном режиме. Можно использовать как имена пользователей, так и их числовые эквиваленты -- UID. По умолчанию используется имя пользователя www-data. Из соображений безопасности не рекомендуется изменять это значение и присваивать имя реального пользователя. В этом случае Web-сервер получит доступ только к тем файлам, которые разрешены для чтения для всех пользователей. Нужно заметить, что указанный пользователь и группа должны существовать в вашей системе. Ни в коем случае не запускайте сервер от имени пользователя root!

ServerAdmin	Задает электронный адрес вебмастера вашего Web-узла. Если возникнут какие-то проблемы, связанные с работой сервера, то по этому адресу будет отправлено соответствующее сообщение. Обычно используется значение webmaster@Your_Host.com. Пользователь webmaster, как правило, не существует реально в системе. Для определения имени (псевдонима) webmaster используется файл псевдонимов электронной почты /etc/aliases.
ServerRoot	В этой директиве указывается местонахождение файлов конфигурации сервера Apache. По умолчанию для этих целей используется каталог /etc/apache2.
BindAddress	Данная директива используется для поддержки виртуальных хостов и применяется чтобы сообщить серверу, какой IP-адрес следует прослушивать. Значением данной директивы может быть «*» (любой адрес), IP-адрес или полное имя домена.
ErrorLog и TransferLog	Определяют расположение журналов сервера Apache. Обычно для этих целей используется каталог /var/log/apache2. В журнале errorlog протоколируются диагностические сообщения, а также сообщения об ошибках, которые порождают CGI-сценарии. В журнале trasferlog протоколируются запросы клиентов. Если включена директива HostNameLookups, то вместо IP-адреса клиентов будут регистрироваться имена компьютеров. Данные директивы можно использовать для задания отдельных журналов для каждого виртуального узла. При определении виртуального узла (см. далее) вы можете задать другие журналы, индивидуальные для этого узла.
Timeout	Задает промежуток времени в секундах, в течение которого сервер продолжает попытки возобновления приостановленной передачи данных. Значение директивы Timeout распространяется не только на передачу, но и на прием данных. Если вам нужно получать большие файлы, рекомендую увеличить данное значение.

KeepAlive	Разрешает постоянные соединения, то есть такие соединения, в которых производится более одного запроса за один раз.
KeepAliveTimeOut	Определяет таймаут для постоянного соединения.
MaxClients	Иногда поступающих запросов настолько много, что компьютеру не хватает ресурсов для загрузки новых копий сервера в память и их выполнения. Директива MaxClients определяет максимальное число копий сервера, которые могут выполняться одновременно.
MaxRequestsPerChild	После обработки определенного количества запросов, указанного в директиве MaxRequestsPerChild, копия сервера завершается, а вместо нее запускается новая.
Listen	Позволяет вам связывать Apache с определенным IP-адресом и (или) дополнительными портами
DocumentRoot	Директива, определяющая местонахождение корневого каталога документов вашего сервера. Значение по умолчанию - /var/www/html. Как правило, для каждого виртуального узла задается собственное значение для DocumentRoot.
UserDir	Задает названия подкаталога в домашнем каталога пользователя, из которого берутся документы. В этом случае, вы активизируете возможность использования пользовательских каталогов. Если вы не хотите включать эту возможность, укажите UserDir DISABLED. Более подробно эта директива будет рассмотрена позже.

DirectoryIndex	Позволяет задать название документа, который будет возвращен по запросу, который не содержит имя документа. С помощью данной директивы можно задать несколько имен файлов. Значениями по умолчанию являются index.html index.php index.htm index.shtml index.cgi Default.htm default.htm index.php3. Например, если вы введете в строке адреса броузера http://localhost, то будет возвращен один из указанных в директиве DirectoryIndex документов. Если в каталоге будет несколько документов, описанных в DirectoryIndex, то будет возвращен первый из них (в данном случае – index.html)
FancyIndexing	При получении запроса, не содержащего имя документа, сервер передаст один из файлов, указанных в директиве DirectoryIndex. Если такой файл не существует, клиенту будет возвращено оглавление каталога. При включении директивы FancyIndexing, в оглавлении каталога будут использованы значки и описания файлов. Если директива FancyIndexing выключена, оглавление будет представлено в более простом виде.

AddIconByType	Сопоставляет значок типу файла. Значок будет использоваться при выводе каталога, если включена директива FancyIndexing. Директива AddIconByType имеет следующий формат: AddIconByType (TEXT, URL) mime-type Параметр TEXT определяет текстовое описание типа, которое увидят пользователи, использующие текстовый браузер или пользователи, у которых отключено отображение рисунков. Параметр URL определяет адрес значка, а параметр mime-type -- это тип файла, с которым нужно сопоставить значок. Полный перечень MIME-типов приведен в файле apache-mime.types. В качестве имени файла можно задать не только MIME-тип, но и символы, которыми заканчивается имя файла, но для этого нужно использовать директиву AddIcon вместо AddIconByType. AddIconByType (VID,/icons/movie.gif) video/* AddIcon /icons/binary.gif .bin .exe Первая директива сопоставляет типу video значок / icons/movie.gif. Вторая директива сопоставляет бинарным файлам *.bin и *.exe значок /icons/binary.gif. Значок по умолчанию задается директивой DefaultIcon.
DefaultType	Если запрашиваемый клиентом тип не соответствует ни одному из MIME-типов, используется MIME-тип, указанный в директиве DefaultType.
AddEncoding	Для сокращения времени передачи файла клиентам используется сжатие данных. Браузеры имеют встроенные программы для распаковки, запускаемые при получении архивов определенных MIME-типов. Именно эти MIME-типы и указываются в директиве AddEncoding.

AddLanguage	В большинстве броузеров можно задать предпочитаемый язык. Благодаря этому вы можете предоставлять документы на разных языках. Директива AddLanguage сопоставляет расширение файла аббревиатуре языка. Для русского языка используется аббревиатура ru, для английского -- en. При этом в корневом каталоге вашего сервера могут находиться несколько индексных файлов на разных языках. Например, для русского языка нужно использовать имя файла index.html.ru, а для английского -- index.html.en.
LanguagePriority	Если на вашем сервере размещены документы на разных языках, то с помощью директивы LanguagePriority можно указать приоритеты различных языков. Например, вы установили директиву LanguagePriority так: LanguagePriority en ru Клиент вводит в строке адреса своего браузера адрес http://www.server.com/. Если в свойствах браузера имеется возможность задать предпочитаемый язык, то возвращен будет файл на нужном языке, если такой существует. Если браузер клиента не поддерживает такую возможность, будет возвращен файл на языке, имеющим наиболее высокий приоритет. В рассмотренном случае (см. пример) это английский язык. Если файл на нужном языке, например, на немецком, не существует, то будет возвращен файл на английском языке. Для того, чтобы сервер поддерживал нужный вам язык, предварительно установите правильное значение директивы AddLanguage.
Alias	С помощью директивы Alias можно предоставить доступ не только к файлам, находящимся в каталоге, указанном директивой DocumentRoot, и его подкаталогах, но и в других каталогах. По умолчанию определен только псевдоним для каталога /icons.

ScriptAlias	Аналогична директиве Alias, но позволяет задать месторасположение каталога для CGI-сценариев.
AddType	Спомощью этой директивы можно добавить новый MIME-тип, который не указан в файле apache-mime.types.
AddHandler и Action	Директива AddHandler позволяет сопоставить определенному MIME-типу какой-нибудь обработчик. А с помощью директивы Action можно определить какое-нибудь действие для обработчика. Например, вы можете запустить какую-нибудь программу для обработки файла данного типа. Пример: AddHandler text/dhtml dhtml Action text/dhtml /cgi-bin/dhtml-parse
ErrorDocument	Директива, сопоставляющая коды ошибок сервера URL-адресам на этом же сервере.
Redirect	Используется для перенаправления с одного адреса на другой

В таблице 15.3 не рассмотрены директивы Location и Directory, которые заслуживают отдельного разговора. Директива Directory определяет свойства каталога, например:

```
<Directory />
Options Indexes Includes FollowSymLinks
AllowOverride None
</Directory>
```

Свойства каталога можно указывать в директиве Directory или в файле .htaccess, который находится в том каталоге, для которого необходимо установить нужные параметры.

В блоке Directory могут находиться директивы управления доступом. К ним относятся директивы AllowOverride, Options, Limit. Рассмотрим по порядку все эти директивы. Директива AllowOverride может принимать значения, указанные в табл. 15.4.

Таблица 15.4. Значения директивы AllowOverride

Значение	Описание
None	Сервер Apache будет игнорировать файлы .htaccess. Рекомендую установить данную опцию, так как это повысит производительность сервера
All	Пользователи имеют право переопределять в файлах .htaccess глобальные параметры доступа. Из соображений безопасности лучше не использовать этот режим
Options	Разрешает использовать директиву Options
Limit	Разрешает использовать директиву Limit
AuthConfig	Разрешает использование директив AuthName, AuthType, AuthUserFile и AuthGroupFile
FileInfo	Разрешает использовать в файлах .htaccess директивы AddType и AddEncoding

С помощью директивы Options можно определить функции сервера, которые будут доступны для использования в определяемом каталоге. Данную директиву можно использовать как в файле apache2.conf, так и в файлах .htaccess. Допустимые опции для директивы Options представлены в табл. 15.5.

Таблица 15.5. Допустимые опции директивы Options

Опция	Описание
None	Не разрешается использование каких-либо функций
All	Разрешаются все функции
FollowSymLinks	Разрешается использовать символические ссылки. С точки зрения безопасности не рекомендуется использовать этот режим
SymLinksIfOwnerMatch	Разрешается использование символических ссылок, если ссылка указывает на объект, который принадлежит тому же пользователю, что и ссылка

ExecCGI	Разрешается выполнение CGI-сценариев
Indexes	Если эта опция выключена, сервер не будет передавать содержимое каталога при отсутствии файла index.html/index.php
Includes	Разрешено использование серверных включений. Рекомендую отключить эту опцию, поскольку это сильно нагружает сервер
IncludesNoExec	Разрешает использование серверных включений, но запрещает запуск из них внешних программ

Директива Limit ограничивает доступ к файлам в определенном каталоге. Параметр метод определяет метод передачи, например, GET или POST. Директиву Limit можно использовать внутри блоков Directory, Location или в файле .htacccess.

В блоке Limit можно использовать такие директивы: allow (разрешить), deny (запретить), order (порядок), require (требуется). Директивы allow и deny аналогичны директивам allow и deny файла конфигурации сервера ProFTPD. После директивы allow следует слово from, после которого можно указать IP-адрес, адрес сети, домен или просто имя компьютера. Слово all обозначает все компьютеры. Например, вам требуется запретить доступ всем компьютерам, кроме компьютеров, которые входят в домен ru:

```
<Limit GET>
order deny, allow
deny from all
allow from ru
</Limit>
```

Следующий пример показывает, как разрешить доступ компьютерам только из вашей сети. Пусть, при этом, ваша сеть имеет адрес 192.168.1.0

```
<Limit GET>
order deny, allow
deny from all
allow from 192.168.1.
</Limit>
```

Директива order определяет порядок выполнения директив allow и deny. Кроме значений allow,deny и deny,allow, директива order может содержать значение mutual-failure. В этом случае доступ будет отказан всем компьютерам, которые явно не указаны в списке allow.

Директива Location позволяет задать определенный URL-адрес, который предназначен для обозначения каталогов, файлов или групп файлов. Обозначить группу файлов можно с помощью шаблонов, например, шаблон *.html определяет все файлы, имена которых заканчиваются на .html. В URL-адрес не включается протокол и имя сервера. Пример:

```
<Location URL>
директивы управления доступом
</Location>
```

15.5. Определение виртуальных узлов

Концепция виртуальных хостов позволяет серверу Apache поддерживать несколько Web-узлов – попросту говоря, несколько сайтов. Получается, что один Web-сервер заменяет несколько серверов, и вместо одного узла пользователи видят отдельные Web-узлы. Это очень удобно, если нужно организовать персональные Web-узлы пользователей или собственные Web-узлы подразделений компании, например sales.mycompany.com.

Сервер Apache можно настроить несколькими способами: чтобы запускался один сервер, который будет прослушивать ВСЕ обращения к виртуальным серверам, или запускать отдельный процесс для каждого виртуального сервера. В первом случае один сервер будет одновременно обслуживать все виртуальные. Если вас интересует такой вариант, нужно настраивать виртуальные сервера с помощью директивы VirtualHost. Настройка отдельных процессов для каждого сервера осуществляется с помощью директивы Listen и BindAddress.

В этом разделе рассматривается именно первый случай, поскольку он наиболее часто используется на практике. Внутри блока директивы VirtualHost можно использовать любые директивы, кроме ServerType, BindAddress, Listen, NameVirtualHost, ServerRoot, TypesConfig, PidFile, MinRequestPerChild, MaxSpareServers, MinSpareServers, так как некоторые из них относятся к основному HTTP-серверу (например, ServerType), а некоторые - ко второму варианту настройки виртуальных серверов и здесь неприемлемы.

Обязательно должны присутствовать директивы ServerName, DocumentRoot, ServerAdmin и ErrorLog.

Виртуальные серверы можно идентифицировать по имени или по IP-адресу. Идентификация по имени имеет существенное преимущество перед идентификацией по IP-адресу: вы не ограничены количеством адресов, имеющемся у вас в распоряжении. Вы можете использовать любое количество виртуальных серверов, и при этом вам не потребуются дополнительные адреса. Такое возможно благодаря использованию протокола HTTP/1.1. Данный протокол давно поддерживается всеми современными браузерами.

В листинге 15.1 приведен файл конфигурации реального виртуального узла, изменено только имя сервера.

Листинг 15.1. Конфигурация виртуального узла

```
# Поддерживаем протокол HTTP (порт 80) для перенаправления
на HTTPS-версию
# Если клиент вводит адрес http://example.com, выполняем
перенаправление на
# https://example.com
<VirtualHost *:80>
      ServerName example.com
      Redirect permanent / https://example.com/
</VirtualHost>

# Основная конфигурация
<VirtualHost *:443>
      ServerAdmin it@example.com
      # Имя узла
      ServerName example.com
      # Псевдоним и IP-адрес
            ServerAlias www.example.com 111.111.111.115
      # Каталог документов
      DocumentRoot /srv/www/example.com/htdocs
      # Журнал ошибок
      ErrorLog /srv/www/example.com/logs/error_log
```

```
    # Журнал доступа
   CustomLog /srv/www/example.com/logs/access_log combined
    # Включаем поддержку SSL
    SSLEngine on
    # Подключаем SSL-сертификат
    SSLCertificateFile /etc/ssl/sert.pem
    SSLCertificateKeyFile /etc/ssl/server.key
    SSLCertificateChainFile /etc/ssl/chain.pem
    # Настройка каталога документов
      <Directory /srv/www/example.com/htdocs/>
        DirectoryIndex index.php
          Options FollowSymLinks
          AllowOverride All
          Order allow,deny
          Allow from All
      </Directory>
</VirtualHost>
```

15.6. Пользовательские каталоги

Директива UserDir включает поддержку пользовательских каталогов. Эта директива определяет общее название подкаталога в домашних каталогах всех пользователей. По умолчанию используется каталог public_html. Данная возможность очень удобна при использовании ее в большой корпорации, где каждый сотрудник имеет собственную страничку. Аналогичное решение можно использовать для сервера кампуса, где администратору лень создавать отдельный виртуальный узел для каждого студента.

Раньше эта возможность часто использовалась на серверах, предоставляющих бесплатных хостинг. Может быть, помните адреса вида http://www.chat.ru/~mypage ? Сейчас же все чаще используется технология виртуальных серверов, которую мы рассмотрели ранее, но знать что такое каталоги пользователей и как с ними работать тоже не помешает. Тем более что домашние каталоги настраиваются намного быстрее и проще, чем виртуальный сервер -- нужно всего лишь определить директиву UserDir и указать месторасположения домашних каталогов.

Доступ к файлам, расположенным в этих каталогах, производится с помощью указания через наклонную черту пользователя после имени сервера. Например, пусть имя сервера www.server.com, имя пользователя -- mark, тогда URL-адрес будет выглядеть так: http://www.server.com/~mark/. При этом сервер самостоятельно определит, где именно расположен домашний каталог пользователя. Если домашний каталог пользователя /home/mark, то сервер передаст клиенту файл /home/mark/public_html/index.html.

15.7. Оптимизация веб-сервера

Сервер Apache для каждого соединения запускает отдельную копию, которая будет обрабатывать запросы клиента. Управлять запущенными копиями позволяют директивы StartServers, MinSpareServers, MaxSpareServers.

Директива **StartServers** задает количество копий, которые будут созданы при запуске исходной копии сервера. При этом исходная копия сервера получает запросы и передает их свободным копиям. Это позволяет равномерно распределить нагрузку между отдельными процессами и повысить производительность сервера, однако на практике все не так хорошо, как хотелось бы. Существенного прироста производительности можно добиться только в случае большой загрузки сервера. По умолчанию запускает пять копий сервера.

Если число поступающих запросов превышает количество запущенных копий сервера, запускаются дополнительные процессы-серверы. Эти процессы не завершаются после обработки своего запроса, а продолжают находиться в памяти. Директива **MaxSpareServers** позволяет указать максимальное число таких процессов. Если это количество превышено, то лишние процессы завершаются. Если количество «серверов на подхвате» меньше, чем задано директивой **MinSpareServers**, запускаются дополнительные копии.

Сервер Apache ведет журнал доступа других компьютеров. Если вы включите директиву **HostnameLookup** (значение on), то в журнал будет записано доменное имя компьютера-клиента. Если эта опция выключена (off), в журнал будет записан IP-адрес клиента. Включение данной опции замедляет работу сервера, так как требуется дополнительное время на ожидание ответа от сервера DNS. Поэтому из соображений оптимизации сервера директиву HostnameLookup нужно выключить.

15.8. Защита сервера Apache

По окончании настройки сервера запретим изменение и удаление файла конфигурации apache2.conf:

```
sudo chattr +i /etc/apache2/apache2.conf
```

После этого вы (и никто другой) не сможете изменить этот файл, даже с помощью конфигуратора. Если изменить файл все же нужно, снимите атрибут:

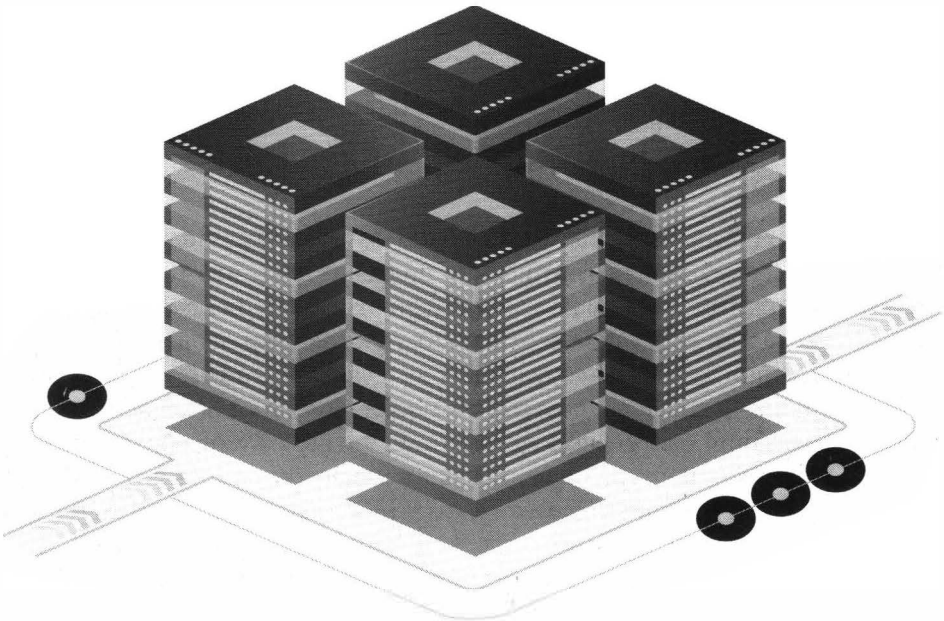
```
sudo chattr -i /etc/apache2/apache2.conf
```

Не нужно, чтобы посторонние глаза смогли посмотреть, а руки – изменить (и выполнить) файлы, находящиеся в каталогах /etc/apache2 и /var/log/apache2:

```
sudo chmod 700 /etc/apache2
sudo chmod 700 /var/log/apache2
```

Глава 16.

Подключение SSL-сертификата



Хотим мы того или нет, но переход на https неизбежен. Активно стимулирует владельцев к переходу на безопасную версию http компания Google: как минимум, в адресной строке Chrome (да уже и других браузеров) будет отметка о том, что соединение не защищено, как максимум, поисковые результаты с сайта не будут появляться в так называемых rich snippets, что довольно плохо для Интернет-магазинов.

16.1. Заказ сертификата

Здесь все достаточно просто: выбор огромен и по сути, если вам нужен просто значок «защищено» в адресной строке браузера, то все равно, у кого купить SSL. Это можно сделать у вашего провайдера, у поставщиков непосредственно сертификатов вроде Comodo. Выбирайте сертификат по цене – где предоставят более выгодные условия, там и покупайте.

Сертификаты отличаются не только вендором и ценой. При покупке сертификата нужно обратить внимание на его тип, определяющий функционал сертификата.

Три основных типа:

- DV (Domain Validation) - используется для подтверждения домена. Данный сертификат подтверждает, что пользователь находится именно на том сайте, на который он осуществил переход, то есть такой сертификат удостоверяет сервер, обслуживающий сайт. DV-сертификат не содержит информации о компании-владельце, поэтому не может считаться безопасным для оказания коммерческих услуг. Если вы будете обрабатывать платежи, поступающие от ваших клиентов, самостоятельно, то такой сертификат нельзя считать безопасным. Если же обработкой поступающих платежей занимается другой сервис, например, PayPal, а ваш сайт только выполняет перенаправление на страницу покупки, то можно обойтись DV-сертификатом. Если вы вообще ничего не продаете, а вам нужен сертификат, чтобы слева от имени вашего сайта в браузере не было строки Не защищено, то данный тип сертификата тоже подойдет.

- OV (Organization Validation) - используется для подтверждения организации и домена. Сертификат этого типа позволяют подтвердить не только доменное имя, но и организацию-владельца веб-сайта. Подлинность организации проверяется по регистрационным данным юридического лица, которые пересылаются провайдеру SSL-сертификата при заказе OV-сертификата. Такие сертификаты наиболее популярны на сегодняшний день.

- EV (Extended Validation) - служит для расширенного подтверждения организации и домена. Данный сертификат обладает самым высоким уровнем доверия со стороны других узлов. Если вам нужна строгая конфиденциальность передаваемых данных (например, обработка платежей именно на вашем сайте), то вам нужен EV-сертификат.

Шифрование между браузером и сайтом обеспечивают все типы сертификатов. Если вам нужно только организовать шифрование трафика по HTTPS, вы можете выбрать любой тип сертификата. Как правило, выбирают DV как самый дешевый.

У всех этих сертификатов есть дополнительные опции WildCard и SAN. Первая подтверждает домен и все его поддомены следующего уровня. То есть можно купить сертификат DV с опцией WildCard для домена example.com и использовать его для подтверждения не только example.com, но и всех его поддоменов следующего уровня, то есть computers.shop.com, phones.shop.com, software.shop.com и т.д.

Опция SAN подтверждает домены по списку, указанному при получении SSL-сертификата.

EV-сертификаты заслуживают отдельного разговора, поскольку являются самыми дорогими. Они подразумевают расширенную периодическую проверку данных владельца сайта для предотвращения подмены этих данных. Так, если OV-сертификат считается действительным на протяжении всего своего срока и проверка организации осуществляется только при покупке сертификата, то в случае с EV-сертификатом такая проверка может осуществляться несколько раз (с определенной периодичностью) на протяжении всего срока действия сертификата.

Изначально все продаваемые SSL-сертификаты были типа OV, то есть подразумевали проверку данных организации. Но спрос рождает предложение и появились более дешевые DV-сертификаты с упрощенной проверкой,

подразумевающей проверку только самого домена. Мошенники могут использовать DV-сертификаты в своих целях - ведь при наличии такого сертификата браузер не будет предупреждать пользователя о возможных проблемах с подлинностью сайта. Отчасти виновником этой проблемы стали браузеры, требующие SSL-сертификат. В погоне за зеленым значком в адресной строке многие стали покупать SSL-сертификаты, платить за OV никому не хочется (да и иногда нет самой организации, которую нужно проверять, например, когда сертификат покупает частное лицо для личного сайта) и провайдеры SSL-сертификатов стали предлагать более дешевые DV-сертификаты.

При желании можно вообще не покупать SSL-сертификат. Вы можете настроить бесплатный сертификат Let's Encrypt и не платить ни копейки. В этой главе будет показано, как это сделать. Для нашей цели – зеленый значок в адресной строке – данного бесплатного сертификата вполне достаточно. Единственный недостаток – его нужно перегенерировать каждые 3 месяца. Но эту задачу можно поручить cron, поэтому можно сказать, что сертификат SSL можно получить бесплатно. В следующем разделе будет показано, как прописать полученный сертификат (без разницы – купили вы его или получили бесплатно) в конфигурационных файлах веб-серверов Apache и nginx, а затем будет рассмотрен процесс получения бесплатного сертификата.

16.2. Настройка Apache

В результате заказа сертификата вам будут предоставлены два файла – SSL-файл сертификата (расширение pem) и ключевой файл (расширение key). Оба эти файла нужно поместить в каталог /etc/ssl.

Перейдите в каталог /etc/apache2/sites-available. В нем хранятся конфигурационные файлы сайтов, работающих на вашем веб-сервере. Откройте файл, содержащий конфигурацию сайта, для которого вы купили сертификат. Далее представим, что наш сайт называется example.com. Конфигурация для него будет следующей:

```
<VirtualHost *:80>
      ServerName example.com
      Redirect permanent / https://example.com/
```

```
</VirtualHost>

<VirtualHost *:443>
        ServerAdmin admin@example.com
        ServerName example.com
        ServerAlias www.example.com xxx.xxx.xxx.xxx
        DocumentRoot /srv/www/example.com/htdocs
        ErrorLog /srv/www/example.com/logs/error_log
CustomLog /srv/www/example.com/logs/access_log combined env=!loopback

        SSLEngine on
        SSLCertificate /etc/ssl/example.pem
        SSLCertificateKeyFile /etc/ssl/server.key
</VirtualHost>
```

Разберемся, что и к чему. Сначала мы создаем виртуальный хост для порта 80, который будет работать как перенаправление – на https-версию сайта. Можно было бы сделать это и через .htaccess, но поскольку у нас есть доступ к конфигу сервера, то можно сделать это прямо здесь.

Далее мы описываем виртуальный хост для порта 443 (используется SSL). Настройки такие же, как и для обычной версии сайта (ServerName, DocumentRoot и т.д.). Отличие заключается только в наличии трех SSL-директив. Первая включает SSL, вторая задает PEM-файл, третья – KEY-файл (имена файлов, понятное дело, у вас будут отличаться).

После этого нужно сохранить файл конфигурации и перезапустить Apache:

```
sudo systemctl restart apache2.service
```

или (в зависимости от вашего дистрибутива)

```
sudo service apache2 restart
```

Обратитесь к вашему сайту. Если вместо надписи **На защищено** появилось изображение зеленого замка, то все хорошо и настройку можно считать завершенной.

Примечание. Иногда замок отображается не зеленым, а серым и не закрытым, а открытым. Это означает, что не все ресурсы сайта загружаются по протоколу https. Откройте исходный код страницы и произведите поиск по строке «http://» (без кавычек). Ваша задача – найти адреса ресурсов (JS, CSS, картинок), которые загружаются по протоколу http. Исправьте URL проблемных ресурсов на https:// и снова обновите страницу сайта. Если вы все сделаете правильно, вы увидите зеленый замок соединения.

16.3. Настройка nginx

Когда сертификат уже есть, осталось дело за малым – настроить веб-сервер. Переходим к нужному файлу конфигурации виртуального узла и приводим его к виду:

```
server {
      listen 443 ssl default_server;
      listen 80;
      server_name <имя>;

      ssl_certificate /путь/fullchain.pem;
      ssl_certificate_key /путь/privkey.pem;

      root /var/www/<каталог>;
      index index.php index.html;

          location /.well-known/acme-challenge {
                  root /var/www/<каталог>;
            }
      ....
      # Другие параметры
}
```

Обратите внимание на следующие моменты:

1. Данный виртуальный узел является сервером по умолчанию (default_server). Если в другом конфигурационном файле уже есть default_server, то произойдет конфликт и придется выбрать сервер по умолчанию.

2. Мы слушаем порты 443 и 80. Порт 80 пока не убирайте – он пригодится, если движок сайта пока еще не готов к SSL.

3. Директивы ssl-* задают путь к сертификату и ключу сертификата. Проверьте правильность пути.

4. Каталоги в директивах root должны быть одинаковы.

5. В конфигурационном файле Apache ключевой файл имел расширение key, сейчас – pem. Никакой ошибки нет. Некоторые сертификаты поставляются в виде двух файлов с расширением pem. Узнать, какой из них ключевой можно по названию самого файла. Let's Encrypt – не исключение. При заказе этого сертификата вам будет сгенерировано два файла – fullchain.pem и privkey.pem. Второй файл – ключевой и при настройке Apache его нужно указать в директиве SSLCertificateKeyFile.

Заставим nginx перечитать конфиг:

```
sudo nginx -t && sudo nginx -s reload
```

16.4. Генерирование сертификата Let's Encrypt

Let's Encrypt – это новый центр сертификации (CA), предоставляющий бесплатные и автоматизированные SSL/TLS-сертификаты. На данный момент Let's Encrypt поддерживается большинством современных браузеров, в том числе IE и даже старыми операционными системами такими как Windows Vista. По сути, все, что вам нужно знать – он бесплатный и поддерживает автоматическое обновление.

16.4.1. Установка клиента Let's Encrypt

Установим клиент с помощью команды:

```
sudo git clone https://github.com/certbot/certbot /opt/
letsencrypt
```

Если git не установлен, то сначала нужно установить его, а потом уже устанавливает клиент для Let's Encrypt (далее certbot). Файлы будут загружены в каталог /opt/letsencrypt.

16.4.2. Каталог webroot-path/.well-known/acme-challenge/

Данный каталог позволяет серверу Let's Encrypt убедиться, что ваш сайт пытается получить бесплатный SSL-сертификат. Каталог нужно создавать в корне веб-сервера. Например, если корень у вас /var/www/shop, то в нем и нужно создать нужный каталог:

```
cd /var/www/shop
mkdir .well-known
mkdir .well-known/acme-challenge
find . -type d -exec chown www-data:www-data {} \;
```

16.4.3. Файл конфигурации

Теперь нужно создать файл конфигурации для вашего домена. Если ваш домен называется example.com, то файл будет называться /etc/letsencrypt/configs/example.com.conf. Содержимое файла:

```
# ваш домен (хотя и можно создать один сертификат для
нескольких доменов
# мы рекомендуем создавать отдельные сертификаты и,
следовательно, отдельные
# файлы конфигурации для разных доменов)
```

```
domains = example.com

# размер ключа
rsa-key-size = 2048 # или 4096

# сервер сертификации
server = https://acme-v01.api.letsencrypt.org/directory

# адрес, на который будут приходить напоминание об обновлении
email = my-email

# отключаем ncurses UI
text = True

# задаем путь к каталогу .well-known (см. выше)
authenticator = webroot
webroot-path = /var/www/shop/
```

16.4.4. Заказ сертификата

Настало время запросить сам сертификат. Во второй команде вам нужно заменить точное имя файла конфигурации:

```
cd /opt/letsencrypt
$ ./certbot-auto --config /etc/letsencrypt/configs/example.
com.conf certonly
```

В результате в каталоге /etc/letsencrypt/live/<название сайта>/ будет сгенерировано два файла – fullchain.pem и privkey.pem. Их и нужно будет прописать в конфигурационном файле сервера, как было показано ранее.

16.4.5. Автоматическое обновление сертификата

Наш сертификат будет действителен в течение 90 дней, после чего должен быть обновлен. Для обновления можно использовать следующий сценарий renew-letsencrypt.sh:

```
#!/bin/sh

cd /opt/letsencrypt/
./certbot-auto  --config  /etc/letsencrypt/configs/my-domain.
conf certonly

if [ $? -ne 0 ]
 then
        ERRORLOG=`tail /var/log/letsencrypt/letsencrypt.log`
       echo -e "The Let's Encrypt cert has not been renewed!\
n\n "
                  $ERRORLOG
 else
         nginx -s reload
fi

exit 0
```

В расписание cron нужно добавить строку:

```
0 0 1 JAN,MAR,MAY,JUL,SEP,NOV * /path/to/renew-letsencrypt.
sh
```

И не забудьте создать каталог /var/log/letsencrypt/ (если он еще не создан) и изменить соответствующим образом права доступа (пользователь, от имени которого выполняется обновление сертификата должен иметь право писать в этот каталог).

16.5. Настройка редиректа

Остались последние штрихи, например, настройка обязательного редиректа с http на https (если вы решите оставить listen 80 в настройках nginx). Для этого в файл .htaccess добавьте строки:

```
RewriteEngine On
RewriteCond %{SERVER_PORT} !^443$
RewriteRule .* https://%{SERVER_NAME}%{REQUEST_URI} [R=301,L]
```

16.6. Готовим движок сайта к SSL

Мало настроить SSL на веб-сервере, нужно движку сайта указать, что он работает по SSL, иначе он будет генерировать ссылки http://, а не https://. Здесь все зависит от движка и более подробные инструкции вы можете получить в документации по нему. Именно поэтому ранее мы рекомендовали не убирать работоспособность обычного протокола http – чтобы вы могли зайти в панель управления сайтом для его настройки на SSL.

16.7. Конвертирование сертификатов разных форматов

Довольно часто для успешной установки SSL-сертификатов на разных платформах и устройствах нужно преобразовать их в другой формат. Так, Windows-серверы используют, как правило, PFX-формат, а для Apache нужны PEM-файлы, имеющие расширение .crt или .cer. Попытаемся разобраться, какие форматы бывают и как конвертировать их из одного формата в другой.

Рассмотрим часто используемые форматы SSL-сертификатов:

- **PEM** - самый распространенный формат сертификата. Файлы в таком формате имеют расширение .pem, .crt, .cer и .key (файл приватного ключа). Сами по себе файлы являются обычными ASCII-файлами, закодированными в формате Base64. При от-

крытии такого сертификата в текстовом редакторе вы видите строку ---BEGIN CERTIFICATE---, после чего следует закодированный сертификат, а после – строка ---END CERTIFICATE---. Веб-сервер Apache использует формат PEM. В одном файле может содержаться несколько SSL-сертификатов и даже приватный ключ. В этом случае каждый сертификат отделяется от остальных тегами BEGIN и END. Однако Apache требует, чтобы сертификаты и приватный ключ должны быть в разных файлах.

- **DER** - бинарный тип сертификата – в отличие от PEM, где сертификат хранится в ASCII-файле. Файлы сертификатов в этом формате часто имеют расширение .cer, но можно встретить и расширение .der. Если перед вами файл с расширением .cer, то для вычисления его формата нужно открыть его в текстовом редакторе. Если вы увидите теги начала и окончания сертификата (BEGIN/END), то это формат PEM. Формат DER используется, как правило, на Java-платформах.

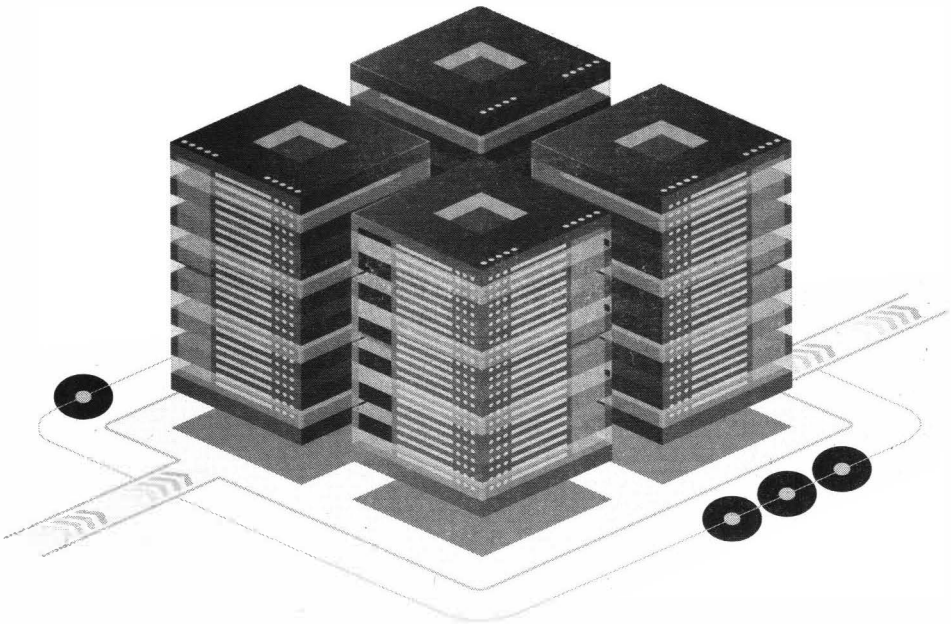
- **PKCS # 7 / P7B** - файл сертификата в этом формате хранятся в формате Base64 ASCII и имеют расширение файла .p7b или .p7c. В файлах находятся теги начала и окончания сертификата – «—— BEGIN PKCS7 ——» и ««—— END PKCS7 ——». Данный формат поддерживается Windows и Java Tomcat.

- PFX - юинарный формат, при использовании этого формата в зашифрованном файле хранятся ваш личный сертификат сервера, промежуточные сертификаты центра сертификации, а также закрытый ключ. PFX файлы, как правило, имеют расширение .pfx или .p12. Обычно используется в Windows для импорта и экспорта файлов SSL сертификатов и приватного ключа.

Таблица 16.1 содержит команды конвертирования SSL-сертификатов из одного формата в другой.

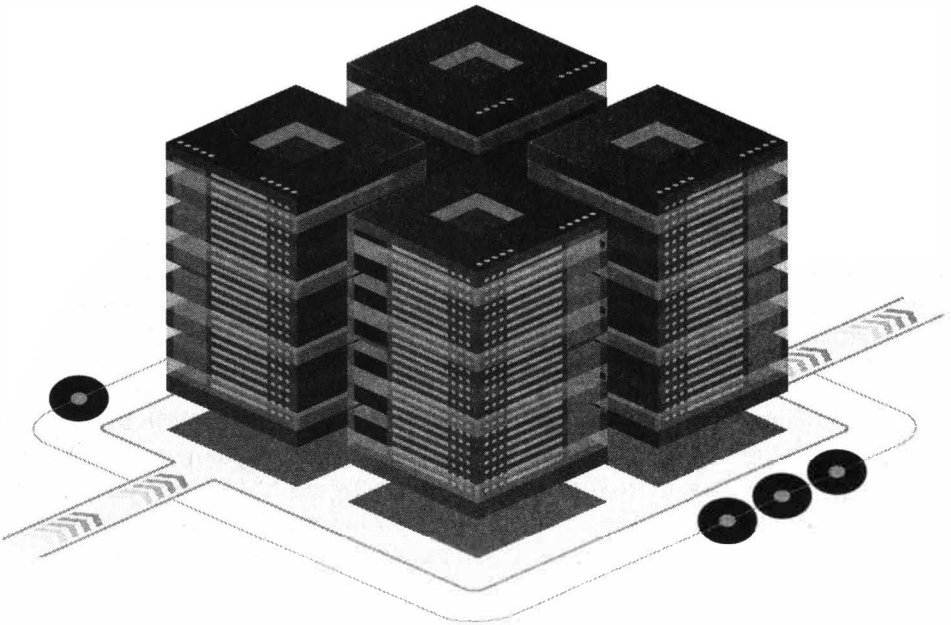
Таблица 16.1. Команды конвертирования

Направление	Команда
PEM -> DER	openssl x509 -outform der -in certificate.pem -out certificate.der
PEM -> P7B	openssl crl2pkcs7 -nocrl -certfile certificate.cer -out certificate.p7b -certfile CACert.cer
PEM -> PFX	openssl pkcs12 -export -out certificate.pfx -inkey privateKey.key -in certificate.crt -certfile CACert.crt
DER -> PEM	openssl x509 -inform der -in certificate.cer -out certificate.pem
P7B -> PEM	openssl pkcs7 -print_certs -in certificate.p7b -out certificate.cer
P7B -> PFX	openssl pkcs7 -print_certs -in certificate.p7b -out certificate.ceropenssl pkcs12 -export -in certificate.cer -inkey privateKey.key -out certificate.pfx -certfile CACert.cer
PFX -> PEM	openssl pkcs12 -in certificate.pfx -out certificate.cer -nodes



Глава 17.

Выбор и установка панели управления сервером



Далеко не всегда удобно администрировать систему посредством консоли. Многие специалисты предпочитают настраивать сервер и управлять ним удаленно посредством панели управления с веб-интерфейсом. В этой главе мы рассмотрим несколько панелей управления сервером, а потом установим одну из них.

17.1. Коммерческие решения

Как правило, пользователям хостинга (веб-сервера, где ресурсы распределены между несколькими клиентами) предоставляется доступ к панели управления (далее ПУ) без возможности выбора самой панели. От установленной панели зависит, насколько удобным будет управление хостингом. При выборе хостинга нужно обращать внимание не только на характеристики тарифа, но еще и на используемую ПУ. Иногда при всех равных наличие удобной или знакомой (часто используемой) «админки» определяет выбор пользователя.

Панель управления может пригодится не только в случае, когда вы настраиваете сервер для хостинг-провайдера. Панель управления иногда полезно установить для самого себя – для удаленного управления сервером. Конечно, всегда можно подключиться к серверу по SSH и произвести все необходимые операции. Но веб-интерфейс куда удобнее SSH и есть множество операций, которые в панели управления можно сделать за пару кликов мышки, а не вспоминать множество команд.

В этом разделе мы рассмотрим популярные коммерческие решения. Они пригодятся, если вы все-таки настраиваете сервер хостинг-провайдера – сервер, который будет предоставлять свои ресурсы клиентам для размещения их сайтов.

17.1.1. cPanel

Часто используемая «админка», популярная не только на зарубежных, но и на отечественных ресурсов. На первом хостинге автора данного обзора была установлена именно cPanel.

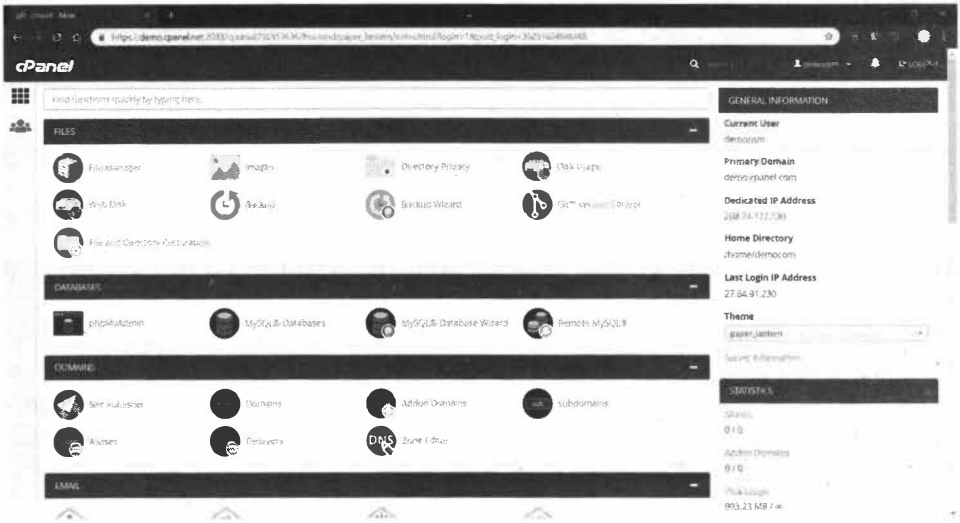


Рис. 17.1. cPanel

Официальный сайт – www.cpanel.net, демо-версия доступна по адресу https://demo.cpanel.net:2083/cpsess8730353636/.

Преимущества:

- Высокая производительность
- Удобный текстовый редактор и диспетчер файловой системы
- Простая и быстрая установка дополнительных модулей
- Наличие русификации
- Удачно реализованы функции бэкапа

Недостатки:

- Перегруженность функциями, что делает ее не очень удобной для новичка

- Ограниченные возможности по управлению DNS
- Имеются проблемы с обновлением панели.

Панель хорошо подходит для хостинг-сервера с огромным количеством клиентов, для отдельных проектов (например, при ее установке на виртуальный сервер для администрирования отдельного сайта) она не подойдет – эффект будет как от стрельбы из пушки по воробьям.

17.1.2. DirectAdmin

Не менее популярная панель управления. Посмотрите на ее скриншот: в ней нет ничего лишнего, зато есть все функции, необходимые для управления сайтами. Можно управлять записями доменной системы имен, учетными записями FTP, почтой, базами данных, расписанием планировщика cron и т.д. Инструменты, как и в случае с cPanel, разбиты на группы для облегчения поиска. Рекомендуем пользователям, которые знакомы с основами Linux, иначе будет сложно разобраться.

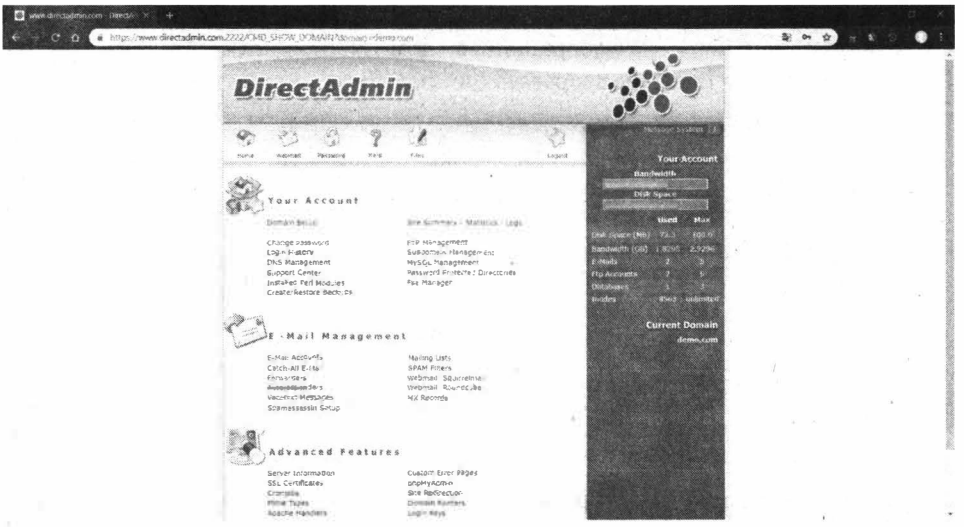


Рис. 17.2. DirectAdmin

Преимущества:

- Разработана на C++ и функционирует как отдельный сервис, поэтому производительность еще выше, чем у cPanel
- Не зависит от наличия веб-сервера, использует собственный сервер, работающий на порту 2222 (по умолчанию).
- Удобные средства резервного копирования.
- Возможность реселлинга ресурсов.
- Неограниченное количество доменов и пользователей.
- Простота использования
- Простота установки, настройки, поддерживается автоустановщик модулей.

Недостатки:

- Сложность локализации (хотя русская локализация имеется).
- Нет встроенной справочной системы (если клиенту что-то будет непонятно, вместо чтения документации, ему придется обращаться в службу поддержки).

DirectAdmin рекоменудем, как для администраторам хостинга, так и конечным клиентам. Также ее с успехом можно установить на виртуальный сервер для управления отдельным проектом.

Демо-версия доступна на сайте https://www.directadmin.com/demo.html

17.1.3. ISPManager

Данная панель отличается хорошо продуманным интерфейсом, поэтому отлично подойдет «чайникам». Ее интерфейс чем-то напоминает интерфейс обычных настольных Windows-приложений, имеются качественные переводы на русский язык. Также в «админку» встроены обучающие видеоролики и имеется хорошая справочная система.

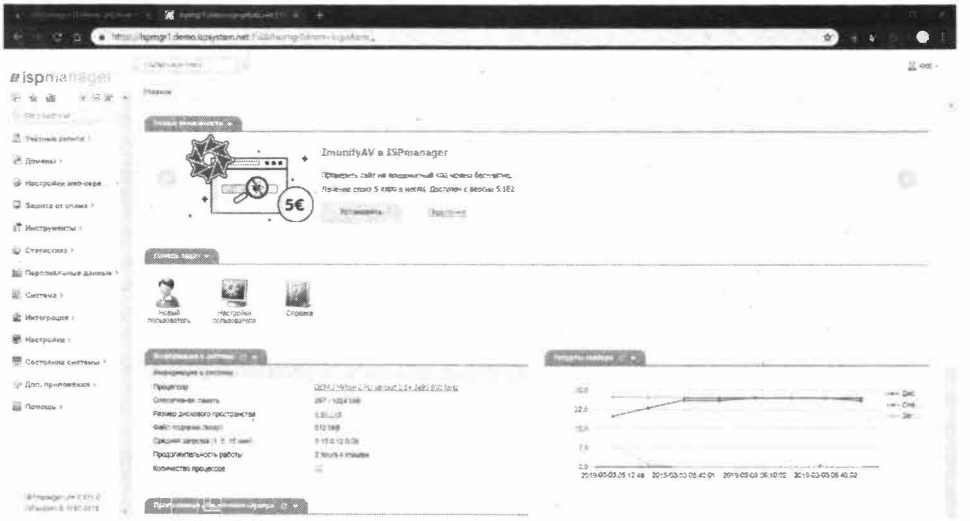


Рис. 17.3. ISPManager

Преимущества:

- Понятный веб-интерфейс.
- Визарды настройки системных демонов.
- Отличный перевод на русский.
- Удобная и продуманная справочная система.
- ∞ количество сайтов и пользователей.
- Возможность кастомизации с указанием собственного лого.

Недостатки:

- Ограниченные возможности миграции пользователей (перенос между серверами).
- Требует установки веб-сервера

ISPMnager идеально смотрится на виртуальном сервере. Эта панель больше подходит для отдельного проекта, чем для установки на хостинг-сервер.

Демо-версия: https://ispmgr1.demo.ispsystem.net:1500/ispmgr

17.1.4. Plesk

Разработчиком является компания SWSoft, но в наши дни она распространяется компанией Parallels.

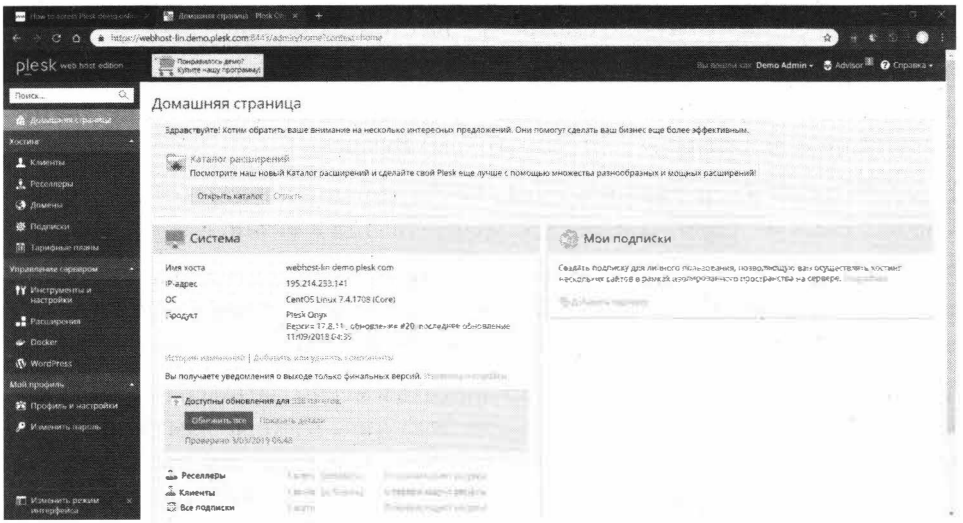


Рис. 17.4. Plesk

Преимущества:

- Богатый набор функций с поддержкой плагинов
- Возможность установки плагинов из ПУ
- Возможность ресселинга

Недостатки:

- Интерфейс нам показался недостаточно удобным
- Перегруженность модулями, которые большинству не нужны

Просмотреть «демку» можно по адресу:
https://support.plesk.com/hc/en-us/articles/115001407434-How-to-access-Plesk-demo-online-servers

Учитывая перегруженность функциями, возможность ресселинга, Plesk больше подходит для хостинг-провайдеров, чем для отдельных пользователей.

17.1.5. Рекомендации

Поскольку все панели управления, представленные в этом разделе – платные, то первое, на что нужно смотреть – на стоимость лицензии.

Недорогая панель ISPManager обойдется от 4.51$ (версия Lite) до 13.54$ (версия Business) в месяц. Для индивидуального проекта вполне хватит версии Lite, а для сервера хостера – только Business.

Та же cPanel обойдется (в зависимости от тарифного плана) – от 20$ до 45$ в месяц. Plesk не намного дешевле – от 10$ до 45$ в месяц.

DirectAdmin тоже не из дешевых – от 16.8$ до 32.5$ в месяц, но у нее есть тарифный план Personal всего за 26.9$ в год, выходит стоимость одного месяца всего 2.25$ - это дешевле даже, чем у ISPManager.

Определите, какую сумму вы готовы платить в месяц, затем сравните функционал панелей управления – будут ли доступны нужные вам функции за ваш бюджет? Если да, то можете смело покупать выбранный вариант. Если же нет, нужно или повышать бюджет или посмотреть на бесплатные решения, представленные в следующем разделе. Таблица 17.1 содержит сравнение некоторых характеристик платных панелей управления. Возможно, она поможет вам определиться, что для вас лучше.

Таблица 17.1. Характеристики панелей управления

	Plesk	cPanel	ISPManager	DirectAdmin
Поддерживаемые ОС	Debian, Ubuntu, CentOS, RHEL, Cloud Linux, Amazon Linux, Virtuozzo Linux, Windows Server	CentOS, CloudLinux, RHEL, Amazon Linux	CentOS, Debian, Ubuntu	Debian, FreeBSD, RHEL, CentOS, CloudLinux
Стоимость лицензии на 1 хост в месяц (на сайте разработчика)	10$ - 45$	15$ – 45$	4.51$ – 13.54$	от 2.24$ до 32.5$

Поддерживаемые веб-серверы	Apache Nginx	Apache поддержка Nginx на стадии тестирования	Apache Nginx	Apache Nginx
Управление доступом по FTP	+	+	+	+
Поддерживаемые СУБД	MySQL MSSQL	MySQL	MySQL PostgreSQL	MySQL MariaDB
Управление почтовым сервисом	+	+	+	+
Настройка доменов и записей DNS	+ (через внешний сервис)	+	+	+
Установка скриптов и CMS	+	+	+	+
Плагины/модули	+	+	+ (незначительное количество)	+
Альтернативные версии PHP	+	+	+	+
Файловый менеджер	+	+	+	+
Резервное копирование	+	+	+	+
Мобильное приложение	Для iOS и Android	-	-	-
Организация хостинга (создание реселлеров и тарифных планов)	Есть в некоторых редакциях	Есть	Есть в версии ISPmanager Business	В версии Standart

17.2. Бесплатные решения

17.2.1. Webmin

Webmin – полностью бесплатная панель управления (ПУ), входящая в состав популярных дистрибутивов Linux. Для ее установки даже не придется подключать сторонние репозитарии.

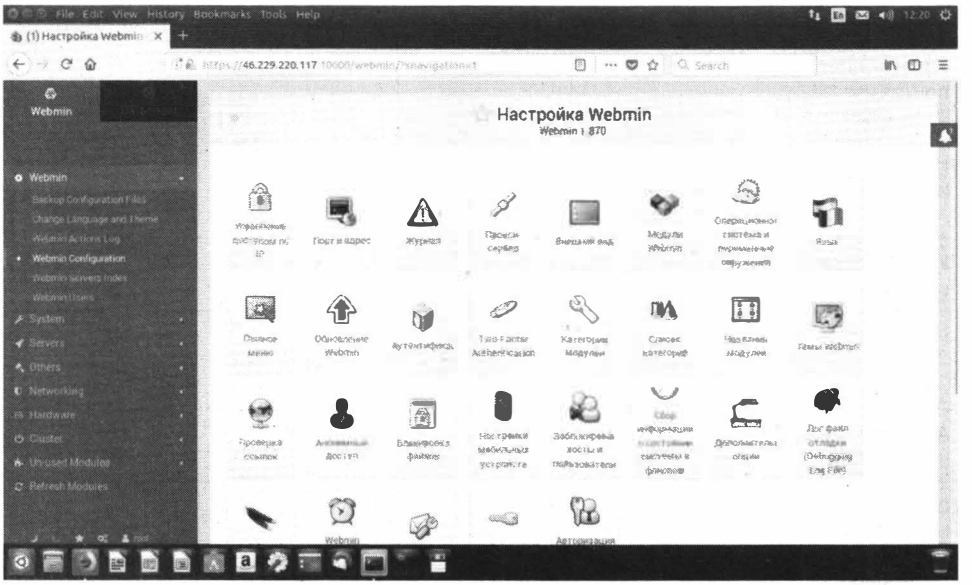


Рис. 17.5. Webmin

Преимущества:

- Простота установки, удобный интерфейс
- Русификация «из коробки»
- Не зависит от веб-сервера и может быть установлена на уже работающий сервер

К недостаткам можно отнести отсутствие поддержки как таковой, в большинстве случаев все пройдет нормально – каких-либо проблем не замечено. Второй недостаток – весьма посредственная русификация, что и показано на иллюстрации выше.

17.2.2. VestaCP

Довольно популярная админка. Она очень удобная, русификация в отличие от Webmin, выполнена полностью.

Рис. 17.6. VestaCP

Преимущества:

- Простой интерфейс пользователя. Как ни крути, Webmin смотрится весьма древне на фоне «весты».
- Готовность к работе «из коробки». Как только вы ее установите, можно сразу приступить к работе.

Недостатков тоже хватает:

- Поддержка у «весты» есть, но она платная.
- Платный файловый менеджер (3 доллара в месяц или 50 долларов на всю жизнь).
- Некоторые сложности с установкой – «веста» требует установки на чистый сервер. Если же вы уже настроили веб-сервер, вам придется его удалить перед установкой «весты».

17.2.3. ISPConfig

Бесплатная админка с открытыми исходниками. Панель управления ISPConfig позволяет управлять несколькими серверами, а также выполнять биллинг клиентов. Подойдет не только для индивидуальных проектов, так и для хостинг-серверов.

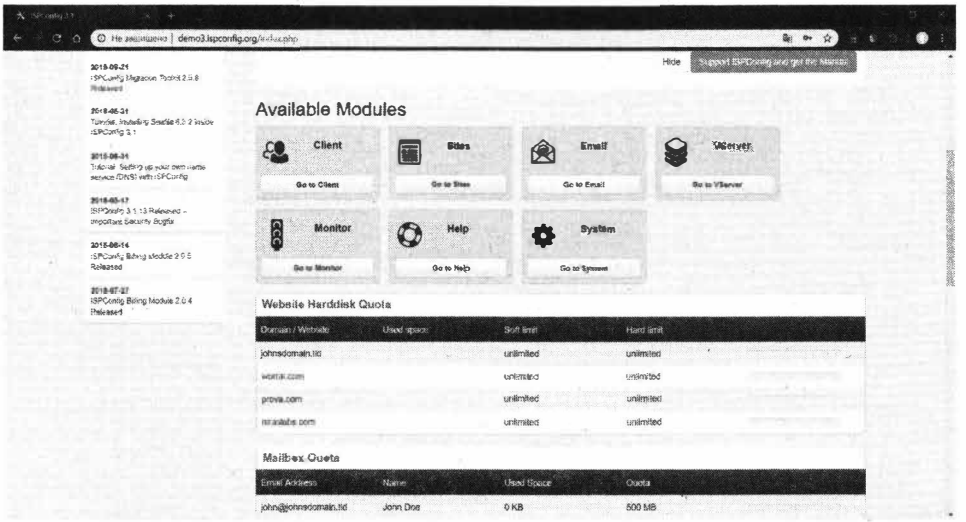


Рис. 17.7. ISPConfig

Преимущества:

- Подходит для администрирования нескольких серверов
- Возможность осуществления биллинга
- Модульная структура

Недостатки:

- Для индивидуального проекта функционала слишком много.
- Сложность установки

17.2.4. Ajenti

Бесплатная ПУ разработки Евгения Панькова. Разработчик отказался от поддержки Apache, отдав предпочтение веб-серверу nginx.

Рис. 17.8. Ajenti

Преимущества:

- Удобный функционал по умолчанию веб-серверами и сайтами
- Модульная структура
- Удобный интерфейс пользователя

Недостатки:

- Нет поддержки Apache
- Сложности с редактированием конфигурационных файлов

17.2.5. Рекомендации

Нужно определиться, какой продукт выбрать. Если инсталляция планируется на только что купленный сервер, где еще не установлено какое-либо ПО (в т.ч. веб-сервер), неплохим вариантом будет VestaCP. Это удобная и быстрая панель управления с удобным интерфейсом. Да, у нее нет бесплатного файлового менеджера, но с этим можно мириться.

ISPConfig не подходит для индивидуальных проектов. Для небольшого хостинг-сервера – отличный выбор, но для одного VDS ее функционал будет избыточный. Да и сложность установки говорит не в пользу ISPConfig.

Ajenti – неплохая и быстрая админка, но она не подойдет пользователям, отдавшим предпочтение веб-серверу Apache. Если по тем или иным причинам вам нужен Apache, эту панель можно сразу вычеркивать из списка претендентов.

Webmin хороша тем, что она входит в состав любого дистрибутива Linux и не привязывается к веб-серверу. Другими словами, установить ее достаточно просто. В отличие от VestaCP вам не придется возвращать сервер в состояние сразу после установки (то есть удалять уже наверняка установленные и настроенные сервисы) для установки этой панели управления. Именно по этой причине Webmin будет рассмотрена в следующем разделе.

17.3. Установка панели управления Webmin на сервер

17.3.1. Знакомство с Webmin

Панель Webmin, как и VestaCP, является бесплатной. Вот только если поддержка у VestaCP платная, но она есть, то у Webmin поддержки как таковой нет - ни платной, ни бесплатной. Если у вас что-то не получилось, вы можете попросить помощи на форумах сообщества Webmin по адресу http://www.webmin.com/community.html. К счастью, какие-либо сложности с настройкой и использованием Webmin возникают редко, поэтому не думаю, что вам когда-либо придется познакомиться с сообществом.

Возможности у Webmin типичные для панели управления Linux-сервером: управление учетными записями пользователей, настройка служб и сервисов, создание почтовых ящиков, мониторинг работы системы. В поставке

по умолчанию Webmin содержит более 500 различных скриптов, которые можно использовать для настройки различных компонентов системы. Благодаря этим скриптам возможностей у Webmin будет больше, чем у VestaCP, где некоторые модули, например, файловый менеджер, придется покупать (50 долларов пожизненно или 3 доллара в месяц).

17.3.2. Установка Webmin

Первым делом установите файловый менеджер mc, чтобы было удобнее работать, в том числе изменять содержимое конфигурационных файлов. Откройте термина или подключитесь к серверу по ssh и введите команду:

```
sudo apt install mc
```

Далее нам нужно подключить репозитарий с webmin. Для этого откройте файл /etc/apt/sources.list и добавьте в него строки:

```
deb http://download.webmin.com/download/repository sarge
contrib
deb http://webmin.mirror.somersettechsolutions.co.uk/
repository sarge contrib.
```

Добавьте GPG-ключ:

```
wget http://www.webmin.com/jcameron-key.asc
sudo apt-key add jcameron-key.asc
```

Осталось обновить списки пакетов и установить пакет webmin:

```
sudo apt-get update
sudo apt install webmin
```

Чтобы веб-интерфейс мог беспрепятственно работать, нужно обновить правила брандмауэра (разрешить порт 10000):

```
sudo ufw allow 10000
```

Собственно, на этом все. Панель управления Webmin установлена.

17.3.3. Вход в панель управления

Запустите браузер и введите адрес:

https://ip_адрес_сервера:10000

Браузер сообщит, что сертификат SSL не является доверенным, добавьте соответствующее исключение в браузер (рис. 17.9). После этого вы увидите форму входа - используйте имя пользователя и пароль с привилегиями root (рис. 17.10).

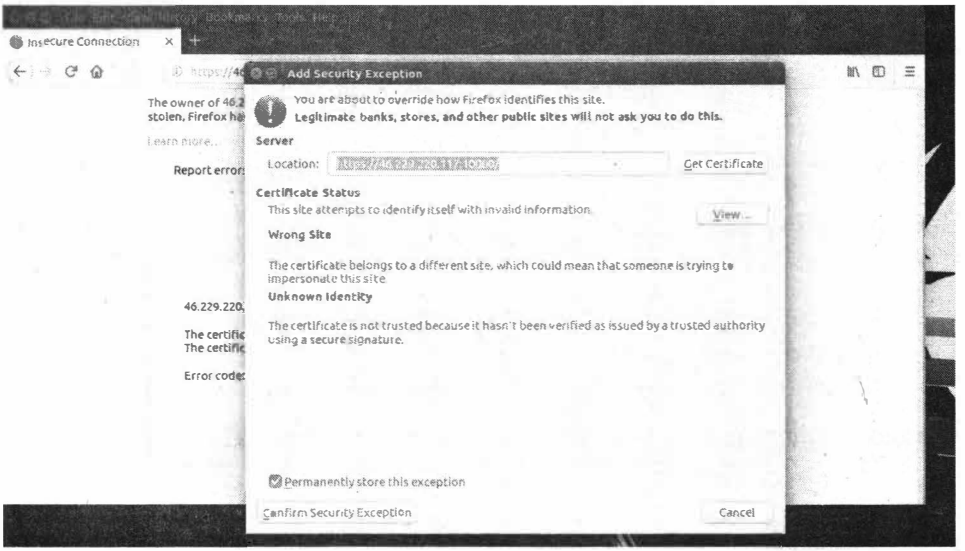


Рис. 17.9. Добавляем исключение безопасности

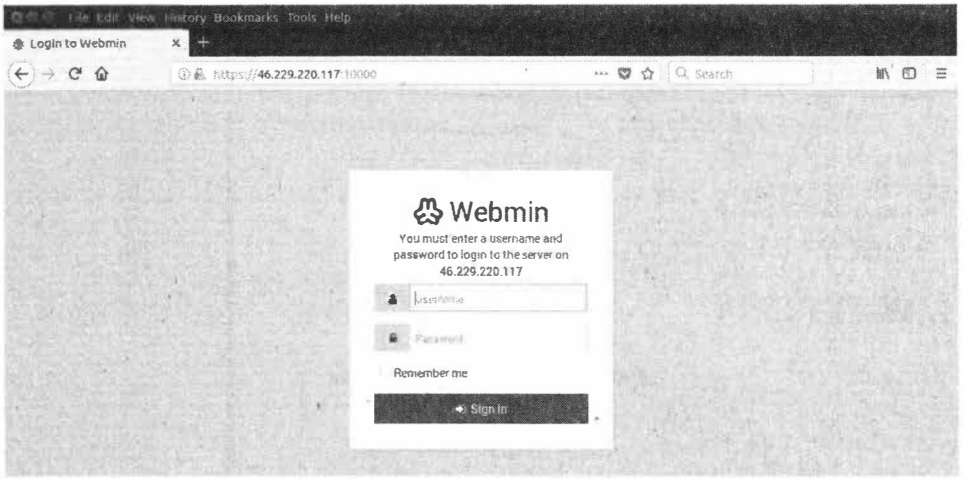


Рис. 17.10. Вход в Webmin

Основной экран Webmin: если вы раньше использовали Webmin, то наверняка заметили изменение дизайна в лучшую сторону. Действительно, у современных версий Webmin интерфейс тоже современный и уже не отдает началом 2000-ых, как было раньше.

По умолчанию язык интерфейса - английский. Русифицировать Webmin очень просто. Зайдите в конфигурацию самой панели (рис. 17.12), выбери-

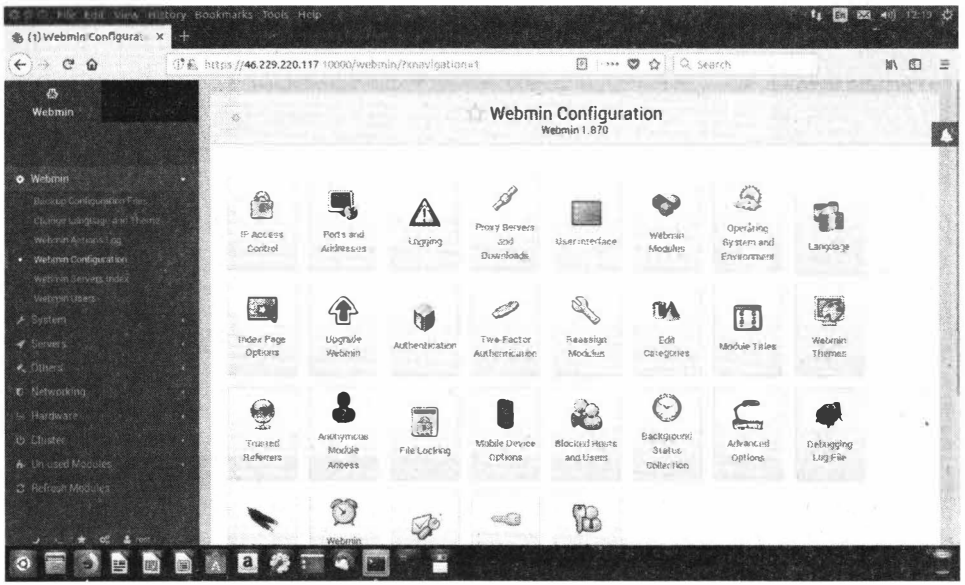


Рис. 17.12. Конфигурация панели управления

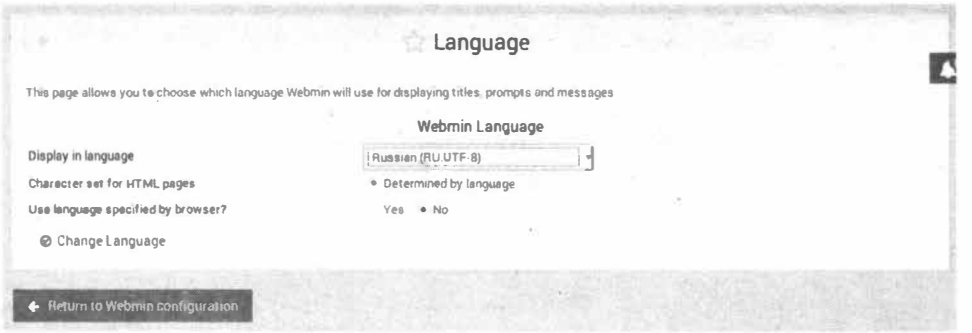


Рис. 17.13. Выбираем язык

те раздел **Language**, далее - выберите русский язык из списка (рис. 17.13) и нажмите кнопку **Change language**. После проделанных манипуляций язык будет изменен на русский (рис. 17.14).

На этом все. Webmin готова к использованию. Далее можно управлять сервером посредством панели управления.

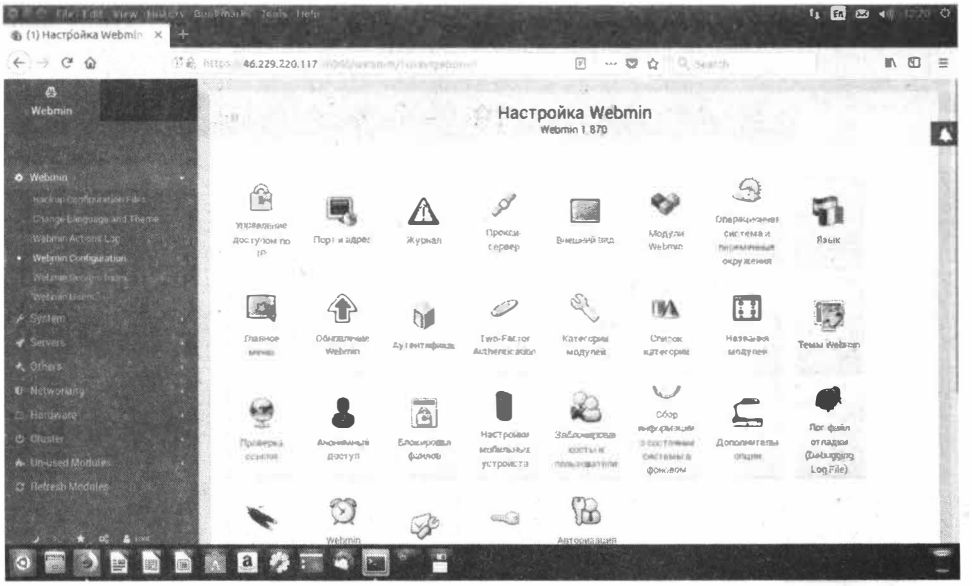
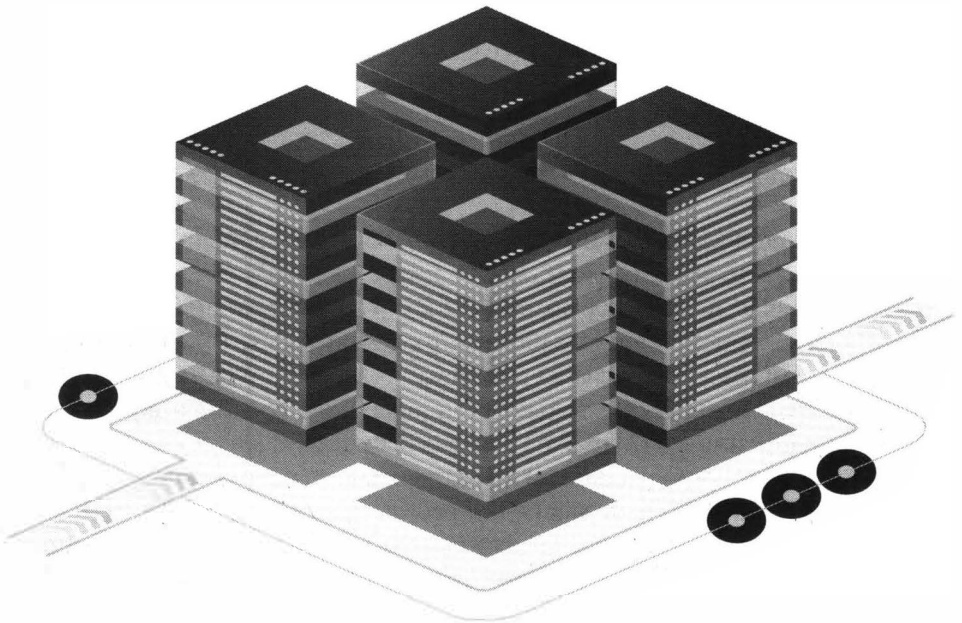


Рис. 17.14. Панель управления русифицирована

Глава 18.

Повышение производительности сервера



Часто сайт работает не так быстро, как бы нам этого хотелось. Порой поиск узкого место может занять довольно много времени.

18.1. Чек-лист производительности сервера

Если сервер (сайт) работает медленно, необходимо проверить следующее:

- **Количество клиентов и конфигурацию сервера** – здесь все просто, если, скажем, месяц назад сайт работал нормально, а сейчас начал «тормозить», посмотрите на количество клиентов, одновременно работающих с сайтом. Это можно сделать, например, с помощью команды netstat | grep http | grep ESTAB (нас интересуют только установленные соединения, а не соединения с TIME_WAIT и другими статусами). Если клиентов много, тогда рассмотрите модернизацию оперативной памяти и выделите больше места под работу каждого процесса PHP (далее будет показано, как это сделать). Если же клиентов немного (рис. 18.1) или сайт тормозит изначально, сразу после его настройки, то дело в чем-то другом.

- **Настройки Apache** – некоторые настройки Apache существенно влияют на производительность. Такие настройки уже были рассмотрены ранее и мы не будем повторяться.

- **Версия PHP** – установка правильной версии PHP позволит добиться существенного ускорения. Так, по данным инструмента PHP Benchmarks (с отчетом вы можете ознакомиться по адресу https://kinsta.com/blog/php-benchmarks/) версии PHP 7.0 и 7.1 в 2.5 раза быстрее, чем версия 5.6. Версии 7.3 и 7.4 – в 3 раза быстрее, чем версия 5.6. Но нужно понимать, что не каждая версия PHP подходит для вашего ПО. Например, Magento 2.3.x требует версии 7.1.3, 7.2.0 и 7.3.0. Magento 2.2.x, наоборот, не поддерживает версию 7.3 (иногда версия CMS подбирается под определенное расширение, которого нет для другой версии, поэтому нет возможности установить самую последнюю версию CMS). Все эти моменты нужно учитывать при смене версии PHP.

- **Директива memory_limit** – данная директива ограничивает объем памяти, выделяемый процессу PHP. Бывает так, что оперативной памяти у сервера достаточно, а под один процесс выделено мало ОЗУ, поэтому тяжелые CMS начинают «тормозить». Здесь нужно просто увеличить количество ОЗУ. Часто память ограничивается на уровне 256Мб, этого на сегодняшний день мало и в зависимости от CMS и нагрузки на сервер, нужно увеличить это число до 512 – 1024 Мб.

- **Директива max_execution_time** – устанавливает максимальное выполнение сценария PHP. Некоторые сценарии выполняются достаточно долго. Например, переиндексирование индекса URL Rewrite на одной из машин с Magento 1 занимало около 14 часов. А теперь представим ситуацию. Планировщик запускает периодически перестроение индексов. Процесс индексатора видит, что индекс требует перестроения и начинает работу. Но выполниться не успевает, поскольку PHP преждевременно завершает его. И так будет продолжаться до бесконечности – а потом мы ломаем голову, почему индексы не перестраиваются и почему сервер тормозит. Нормальная практика – установка этого значения в 0, чтобы время выполнения не ограничивалось. Либо же установка большого значения, например, 5-7 часов, а может и больше – в зависимости от ситуации.

- **Настройка сжатия и кэширования** – включение сжатия страниц позволяет уменьшить объем передаваемых клиенту данных – данные будут переданы раньше, нагрузка на сервер будет ниже. Довольны окажутся все – и клиент, и администратор. Потери производительности, необходимые на сжатие контента и его распаковку (на клиенте) вполне оправданы. Настройка кэширования позволяет задать время кэширования контента на стороне клиента. Сервер «отдает» заголовки, устанавливающие время кэширования определенных ресурсов – картинок, CSS и т.д. Браузер, прокэшировав один раз ресурс, больше не будет его загружать, пока не выйдет время, указанное сервером. В результате имеем: сайт загружается быстрее, а нагрузка на сервер ниже.

- **Включение кэширования на стороне сервера** – для этого используется специальные инструменты вроде Pagespeed и Memcached.

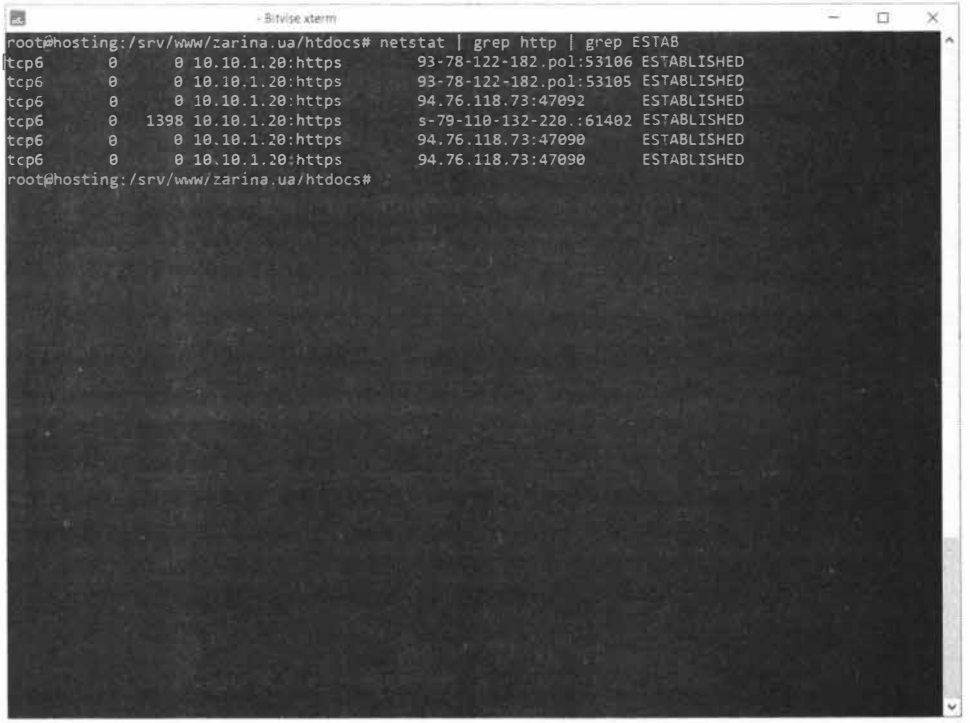


Рис. 18.1. Вывод команды netstat | grep http | grep ESTAB

- **Настройка CMS** – иногда параметры самой CMS не предполагают оптимизации. Например, в настройках CMS можно включить минимизацию CSS/JS и их объединение в один файл. В результате вместо 20 запросов к серверу (для загрузки скриптов и таблиц стилей) будет всего два запроса – один на загрузку стилей, другой – скриптов. Учитывая, что код минимизирован (из него удалены лишние пробелы и комментарии), то время загрузки страницы снижается. Такой трюк возможен не всегда, особенно со скриптами – начинаются всевозможные конфликты, но попробовать стоит.

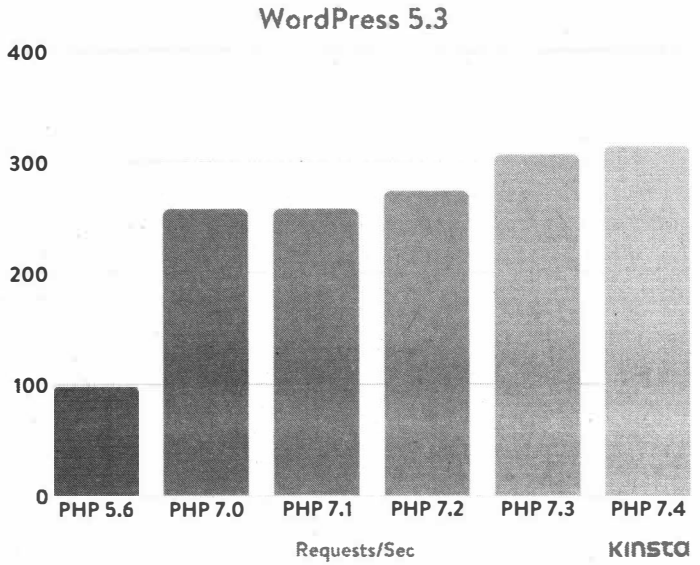


Рис. 18.2. График производительности (источник: kinsta.com)

18.2. Смена версии PHP и его настройка

Здесь все зависит от того, какой у вас сервер. Если у вас обычный физический сервер или же виртуальный сервер с полноценным root-доступом, то следуйте ранее описанным инструкциям для установки нужной версии PHP. Да, все придется делать вручную, но другого выбора у вас нет.

Если же у вас виртуальный сервер, управляемый через панель управления, то сменить версию PHP можно в панели управления. Правда, четкие

инструкции зависят от используемой панели управление. Так, на ресурсе cloudways для смены версии PHP нужно выполнить следующие действия:

1. Войдите в панель управления серверами.
2. Выберите ваш сервер
3. Перейдите в раздел **Settings & Packages**, на вкладку **Packages**
4. Нажмите значок карандаша напротив версии PHP
5. Выберите другую версию PHP (рис. 18.3) и нажмите кнопку **Save**
6. Дождитесь, пока система поменяет версию PHP. Помните, что ваше ПО может перестать работать после смены версии PHP! Поэтому смена версии PHP – это очень ответственный шаг.

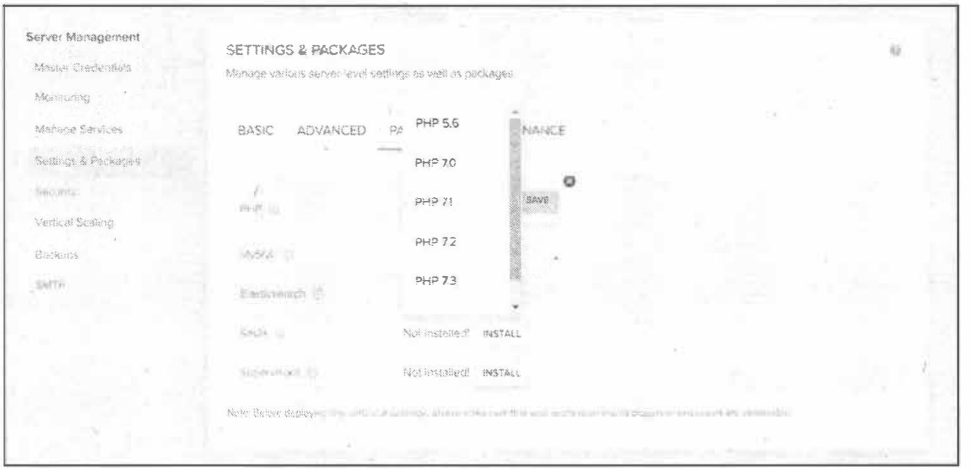


Рис. 18.3. Смена версии PHP

Теперь нам нужно установить максимальные лимиты для доступной процессу памяти и доступного времени выполнения. Опять-таки, все зависит от того, какой у вас сервер. При наличии полноценного root-доступа (или физического сервера), откройте файл конфигурации PHP. Как правило, это /etc/phpN/apache2/php.ini (N – номер версии PHP) и установите новые значения для директив max_execution_time и memory_limit (рис. 18.4).

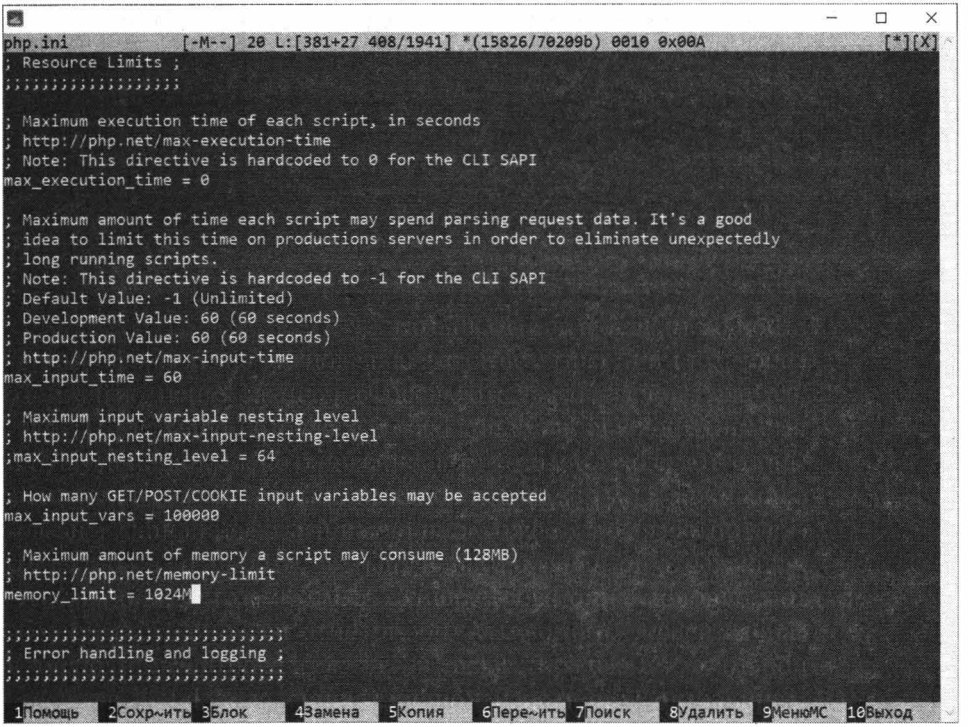


Рис. 18.4. Редактирование php.ini

После этого заставьте сервер перечитать файл конфигурации или попросту перезагрузите его.

Примечание. Для директивы max_execution_time значение указывается в секундах. Например, 3600 – это 1 час. Значение 0 отключает проверку времени выполнения и сценарии могут выполняться вечно.

Создайте сценарий info.php в корне вашего веб-сервера:

```
<?php
phpinfo();
?>
```

Выполните этот сценарий (http://имя_сервера/info.php) и убедитесь, что изменения вступили в силу и сценариям доступны выделенные ресурсы.

Если же вы используете Cloudways или подобные сервисы, изменить необходимые параметры можно в панели управления. В случае с Cloudways они находятся на вкладке **Basic** раздела **Settings & Packages** (рис. 18.5). Просто установите нужные значения и нажмите кнопку **Save changes**.

Рис. 18.5. Изменение параметров PHP

18.3. Настройка сжатия и кэширования

Сегодня все браузеры поддерживают сжатия (картинок, файлов), поскольку это является частью спецификации протокола HTTP 1.1. Сжатие текстовых форматов (например, CSS, Javascipt или HTML) сможет уменьшить их объем до 70%. Работает это следующим образом:

1. Прежде чем отправить ответ, сервер выполняет сжатие данных
2. При получении сжатого ответа от сервера браузер выполняет обратную процедуру и разжимает сжатый контент и выводит результат.

Кэширование также помогает ускорить загрузку страницы. Когда ресурс находится в кэше браузера, то браузер будет использовать его локальную версию, и не будет загружать с сервера. Как работает кэширование, хорошо видно в средствах разработчика. Откройте вкладку **Network** (Сеть) средств разработчика браузера (рис. 18.6) и вы увидите, что часть ресурсов сайта загружаются с локального кэша. На загрузку этих ресурсов браузер не тратит время, что снижает время загрузки страницы и нагрузку на сервер.

Далее будет показано, как включить сжатие и кэширование в двух самых популярных веб-серверах - nginx и Apache.

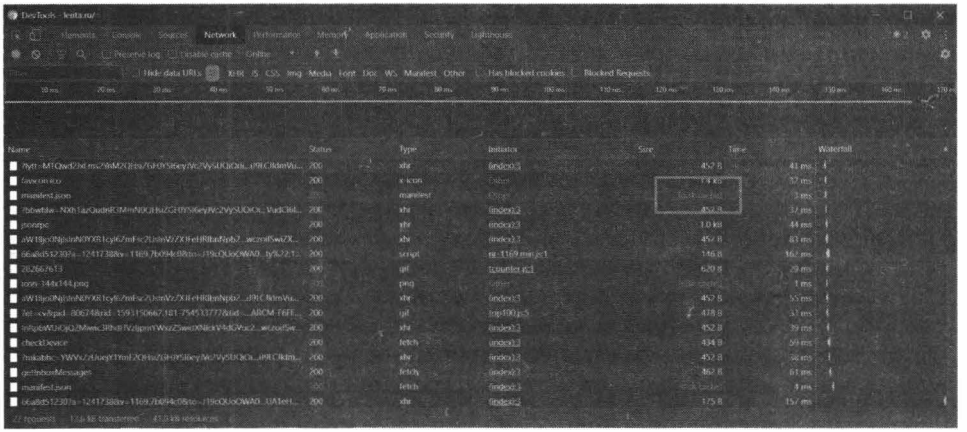


Рис. 18.6. Средства разработчика Chrome

18.3.1. Настройка nginx

Файл конфигурации nginx называется /etc/nginx/nginx.conf. Нам и придется его редактировать. В файл конфигурации сервера (или отдельного хоста, если нужно включить сжатие не для всех сайтов, а только для конкретного) нужно добавить следующие строки:

```
#GZIP
 gzip on;
 gzip_min_length 1000;
 gzip_proxied expired no-cache no-store private auth;
 gzip_types text/plain application/xml;
```

```
# Выделяем буфер для gzip
 gzip_buffers 32 4k;
# Устанавливаем уровень сжатия, от 1-9
 gzip_comp_level 9;
# Убираем поддержку IE6
 gzip_disable «msie6»;
# Устанавливаем версию http для использования gzip (1.0 или
1.1)
 gzip_http_version 1.1;
# Разрешаем использовать статику
 gzip_static on;
 gzip_vary on;
# Определяем, какие типы файлов нужно сжимать
 gzip_types text/css text/javascript text/xml text/plain text/
x-component application/javascript application/x-javascript
application/json application/xml application/rss+xml font/
truetype application/x-font-ttf font/opentype application/
vnd.ms-fontobject image/svg+xml;
```

Теперь настроим кэширование. При настройке времени кэширования помните и об обратном эффекте кэширования. Если вы внесете изменения в ваши стили или скрипты, то пользователи увидят их только, когда загрузят новую версию. Чтобы увидеть изменения раньше, нужно будет очистить кэш браузера, а как это сделать, как показывает практика, знают не все пользователи. В любом случае приведенный конфиг предполагает хранения файлов в кэше 1 сутки (86400 секунд). Если вы вносите изменения в сайт редко, можно увеличить это время до 10-30 суток. Для статического контента, который никогда не меняется (выложили файл и забыли о нем) кэширование включено сроком на 1 год.

```
# Хранить кэш 24ч\1 сутки
 expires 86400s;
# Добавляем заголовки (хеадеры)
 add_header Pragma public;
  add_header Cache-Control "max-age=86400, public, must-
revalidate, proxy-revalidate";
```

```
 #add_header "X-UA-Compatible" "IE=Edge,chrome=1";

# Правила rewrite для версированного CSS + JS через дериктиву
filemtime
 location ~* ^.+\.(css|js)$ {
 rewrite ^(.+)\.(\d+)\.(css|js)$ $1.$3 last;
# Задаем, сколько будет храниться кэш
 expires 31536000s;
# Выключаем логирование
 access_log off;
 log_not_found off;
# Добавляем заголовки (хеадеры)
 add_header Pragma public;
 add_header Cache-Control "max-age=31536000, public";
 }

# Агрессивное кэширование для статических файлов
 location ~* \.(asf|asx|wax|wmv|wmx|avi|bmp|class|divx|doc|
docx|eot|exe|gif|gz|gzip|ico|jpg|jpeg|jpe|mdb|mid|midi|mov|
qt|mp3|m4a|mp4|m4v|mpeg|mpg|mpe|mpp|odb|odc|odf|odg|odp|ods
|odt|ogg|ogv|otf|pdf|png|pot|pps|ppt|pptx|ra|ram|svg|svgz|s
wf|tar|t?gz|tif|tiff|ttf|wav|webm|wma|woff|wri|xla|xls|xlsx|x
lt|xlw|zip)$ {
# Задаем сколько будет храниться кэш
 expires 31536000s;
# Выключаем логирование
 access_log off;
 log_not_found off;
# Добавляем заголовки (хеадеры)
 add_header Pragma public;
 add_header Cache-Control "max-age=31536000, public";
 }
```

18.3.2. Настройка Apache

Файл конфигурации Apache называется /etc/apache2/apache2.conf. Это основной файл конфигурации. Если нужно изменить конфигурацию для определенного сайта, поищите его конфиг в папке /etc/apache2/sites-available.

По умолчанию модуль mod_deflate должен быть включен в apache. Но лучше убедиться и выполнить проверку и поискать следующую строку в конфиге веб-сервера apache:

```
LoadModule deflate_module modules/mod_deflate.so
```

Мы можем определить, какие типы файлов нужно сжать:

```
AddOutputFilterByType DEFLATE text/html text/plain text/css
application/javascript
```

Установите следующую конфигурацию в виртуальный хост Apache и это включит сжатие mod_deflate для вашего сайта.

```
<Directory /var/www/html/>
<IfModule mod_mime.c>
 AddType application/x-javascript .js
 AddType text/css .css
</IfModule>
<IfModule mod_deflate.c>
 # Сжимаем HTML, CSS, JavaScript, Text, XML и шрифты
 AddOutputFilterByType DEFLATE application/javascript
 AddOutputFilterByType DEFLATE application/rss+xml
 AddOutputFilterByType DEFLATE application/vnd.ms-fontobject
 AddOutputFilterByType DEFLATE application/x-font
 AddOutputFilterByType DEFLATE application/x-font-opentype
 AddOutputFilterByType DEFLATE application/x-font-otf
 AddOutputFilterByType DEFLATE application/x-font-truetype
 AddOutputFilterByType DEFLATE application/x-font-ttf
 AddOutputFilterByType DEFLATE application/x-javascript
```

```
 AddOutputFilterByType DEFLATE application/xhtml+xml
 AddOutputFilterByType DEFLATE application/xml
 AddOutputFilterByType DEFLATE font/opentype
 AddOutputFilterByType DEFLATE font/otf
 AddOutputFilterByType DEFLATE font/ttf
 AddOutputFilterByType DEFLATE image/svg+xml
 AddOutputFilterByType DEFLATE image/x-icon
 AddOutputFilterByType DEFLATE text/css
 AddOutputFilterByType DEFLATE text/html
 AddOutputFilterByType DEFLATE text/javascript
 AddOutputFilterByType DEFLATE text/plain
 AddOutputFilterByType DEFLATE text/xml
 # Следующая строка для .js и .css
 AddOutputFilter DEFLATE js css
AddOutputFilterByType DEFLATE text/plain text/xml application/
xhtml+xml text/css application/javascript application/xml
application/rss+xml application/atom_xml application/x-
javascript application/x-httpd-php application/x-httpd-
fastphp text/html
</IfModule>
<IfModule mod_setenvif.c>
 # Удалить ошибки браузера (требуется только для очень старых
браузеров)
 BrowserMatch ^Mozilla/4 gzip-only-text/html
 BrowserMatch ^Mozilla/4\.0[678] no-gzip
 BrowserMatch \bMSIE !no-gzip !gzip-only-text/html
</IfModule>
<IfModule mod_headers.c>
 Header append Vary User-Agent env=!dont-vary
</IfModule>
<ifModule mod_gzip.c>
 mod_gzip_on Yes
 mod_gzip_dechunk Yes
 mod_gzip_item_include file .(html?|txt|css|js|php|pl)$
 mod_gzip_item_include handler ^cgi-script$
```

```
 mod_gzip_item_include mime ^text/.*
 mod_gzip_item_include mime ^application/x-javascript.*
 mod_gzip_item_exclude mime ^image/.*
 mod_gzip_item_exclude rspheader ^Content-Encoding:.*gzip.*
</ifModule>
</Directory>
```

Для включения кэширования можно вынести конфигурацию за пределы основного файла и создать файл .htaccess в корневом каталоге документов. Содержимое этого файла будет следующим:

```
# кеширование в браузере на стороне пользователя
<IfModule mod_expires.c>
 #Включаем поддержку директивы Expires
  ExpiresActive On
 # Задаем время для хранения файлов (картинок) в кэше для
каждого типа
  ExpiresDefault "access 7 days"
  ExpiresByType application/javascript "access plus 1 year"
  ExpiresByType text/javascript "access plus 1 year"
  ExpiresByType text/css "access plus 1 year"
  ExpiresByType text/html "access plus 7 day"
  ExpiresByType text/x-javascript "access 1 year"
  ExpiresByType image/gif "access plus 1 year"
  ExpiresByType image/jpeg "access plus 1 year"
  ExpiresByType image/png "access plus 1 year"
  ExpiresByType image/jpg "access plus 1 year"
  ExpiresByType image/x-icon "access 1 year"
  ExpiresByType application/x-shockwave-flash "access 1 year"
</IfModule>
# Cache-Control
<ifModule mod_headers.c>
# Задаем 30 дней для данного типа файла
<filesMatch «\.(ico|pdf|flv|jpg|jpeg|png|gif|swf)$»>
  Header set Cache-Control "max-age=2592000, public"
```

```
</filesMatch>

# Задаем 30 дней для данного типа файла
<filesMatch "\.(css|js)$">
 Header set Cache-Control "max-age=2592000, public"
</filesMatch>

# Задаем 2 дня для данного типа файла
<filesMatch "\.(xml|txt)$">
Header set Cache-Control "max-age=172800, public, must-
revalidate"
</filesMatch>

# Задаем 1 день для данного типа файла
<filesMatch "\.(html|htm|php)$">
 Header set Cache-Control "max-age=172800, private, must-
revalidate"
</filesMatch>
</ifModule>

# использование кеша браузеров
FileETag MTime Size
<ifmodule mod_expires.c>
<filesmatch ".(jpg|jpeg|gif|png|ico|css|js)$">
 ExpiresActive on
 ExpiresDefault "access plus 1 year"
</filesmatch>
</ifmodule>

#Запрет отдачи HTTP-заголовков Vary браузерам семейства MSIE
<IfModule mod_setenvif.c>
 BrowserMatch "MSIE" force-no-vary
 BrowserMatch «Mozilla/4.[0-9]{2}» force-no-vary
</IfModule>
```

Чтобы все это добро работало, вам нужно включить модули headers и expires. Это делается командами:

```
# a2enmod headers
# a2enmod expires
# service apache2 restart
```

Последняя команда перезагружает сервер – в принципе вам все равно это нужно было сделать, чтобы он перечитал конфигурационный файл. Конечно, можно было бы обойтись reload, но тогда бы модули headers и expires не были бы загружены, если до этого вы их не использовали.

18.4. Включение кэширования на стороне сервера

Теперь поговорим о кэшировании на стороне сервера. Хорошо, если движок сайта обладает возможностью кэширования страниц – это может существенно повысить его производительность. Для многих CMS разработаны расширения кэширования. Вы можете использовать эти расширения или же настроить инструменты Pagespeed и Memcached на сервере.

Выбрать средство кэширования поможет следующий алгоритм:

- Существуют ли расширения кэширования на вашей CMS:
  - Да – можете использовать их. Если они платные, то или их придется покупать или использовать инструменты сервера.
  - Нет – использовать инструменты сервера
- Планируете ли вы смену CMS?
  - Да – используйте инструменты сервера, тогда ничего не придется решать с кэширование после установки новой CMS, все уже будет работать.
  - Нет – можно использовать расширения кэширования для CMS.

Далее будет показано, как настроить Pagespeed и Memcached для сервера Apache.

18.4.1. Настройка Pagespeed

Первым делом нужно определить разрядность операционной системы. Конечно, сегодня найти 32-разрядный VDS – еще та задача, но все же. Введите команду:

```
uname -a
```

Если увидите в выводе x86_64 – ваша система 64-разрядная и нужно ввести следующие команды:

```
cd /tmp
wget https://dl-ssl.google.com/dl/linux/direct/mod-
pagespeed-stable_current_amd64.deb
sudo dpkg -i mod-pagespeed-stable_current_amd64.deb
```

Для 32-разрядной системы нужно ввести команды:

```
cd /tmp
wget https://dl-ssl.google.com/dl/linux/direct/mod-
pagespeed-stable_current_i386.deb
sudo dpkg -i mod-pagespeed-stable_current_i386.deb
```

Хотя, как правило, ваша система окажется 64-разрядной и последний абзац можно было бы и не писать. Но мало ли. Вдруг кому-то пришла в голову светлая мысль организовать веб-сервер на 32-битной машине.

После этого нужно перезапустить Apache:

```
sudo service apache2 restart
```

Собственно, на этом все. Больше ничего делать не нужно.

18.4.2. Настройка Memcached

Демон Memcached позволяет добиться существенного ускорения загрузки страниц. Он доступен из репозитариев Ubuntu:

```
sudo apt install memcached
```

После установки демона нужно узнать порт, на котором он работает. Введите команду:

```
netstat -tap | grep memcached
```

Порт будет выведен после слова localhost, например, localhost:11211. Это стандартный порт для Memcached, который нужно указать в конфигурации mod_pagespeed.

Для ускорения PHP-приложений нужно установить пакет php-memcached:

```
apt install php-memcached
```

Осталось настроить mod_pagespeed на работу с помощью Memcached. Для этого откройте файл /etc/apache2/mods-available/pagespeed.conf:

```
mcedit /etc/apache2/mods-available/pagespeed.conf
```

Произведите поиск по строке ModPagespeedMemcachedServers. Раскомментируйте строку:

```
# ModPagespeedMemcachedServers localhost:11211
```

Она должна быть такой:

```
ModPagespeedMemcachedServers localhost:11211
```

Перезапустите Apache:

```
sudo service apache2 restart
```

После этого можете наблюдать некоторое ускорение работы вашего сайта. Для более тонкой настройки обратитесь к документации по memcached.

18.5. Ускорение CMS

В этом разделе будет рассмотрено ускорение Magento 2 – одной из самых требовательных к ресурсам CMS. Приведенные советы будут актуальны и для других CMS, но, интерфейс панели управления, ясное дело, будет отличаться.

Часто свежеустановленная Magento работает очень медленно. Особенно ситуация ухудшается после установки нестандартной темы оформления. Без скина магазин работает еще более или менее, но как только «одели» тему, начинаются заметные тормоза. Рано или поздно, конечно, вам все равно придется добавить ресурсов (количество vCPU и оперативной памяти), но до этого нужно выжать максимум из того, что есть. Добавить ресурсов всегда можно.

18.5.1. Включение кэша

Включение кэширования существенно ускоряет загрузку страниц магазина. При установке расширений обычно нужно выключить кэширование. Вполне возможна ситуация, когда вы кэш выключили, но забыли включить

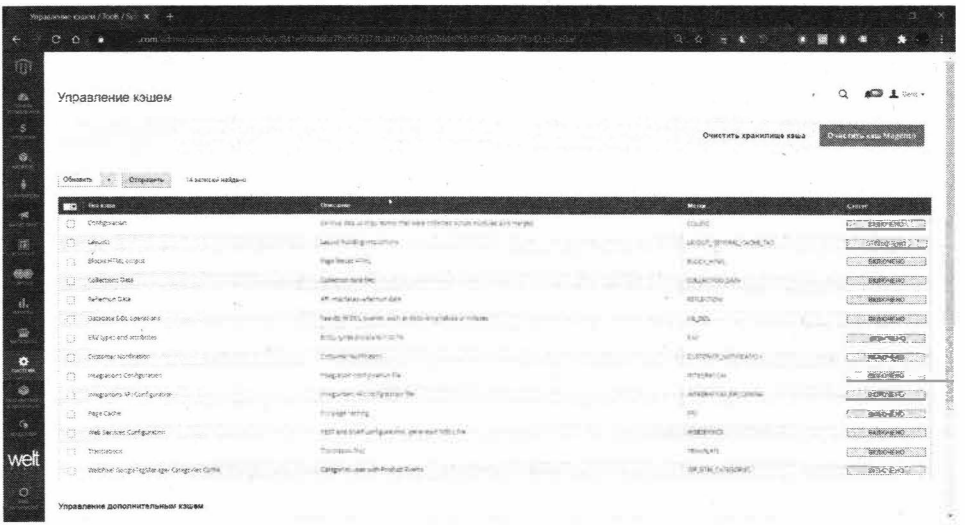


Рис. 18.7. Включаем кэш

Перейдите в **Система, Управление кэшем**. Убедитесь, что все типы кэша включены. Для включения кэша:

1. Выберите все выключенные типы кэша.
2. Из списка выберите действие **Включить**
3. Нажмите кнопку **Отправить**

На рис. 18.7 показано, что один из типов кэша отключен. Это кэш разметки страницы. В данном случае система вынуждена проверять разметку каждый раз при генерировании страницы, что, разумеется, негативно сказывается на производительности.

18.5.2. Включаем сжатие и объединение JavaScript и CSS

Тема оформления и расширения добавляют множество JavaScript-сценариев и CSS. Ускорить загрузку этих ресурсов можно путем включения сжатия (минимизации – когда из кода удаляются все лишние пробельные символы) и объединения сценариев в один большой блок кода. Логика проста – минимизация уменьшает размер передаваемых данных, а объединение

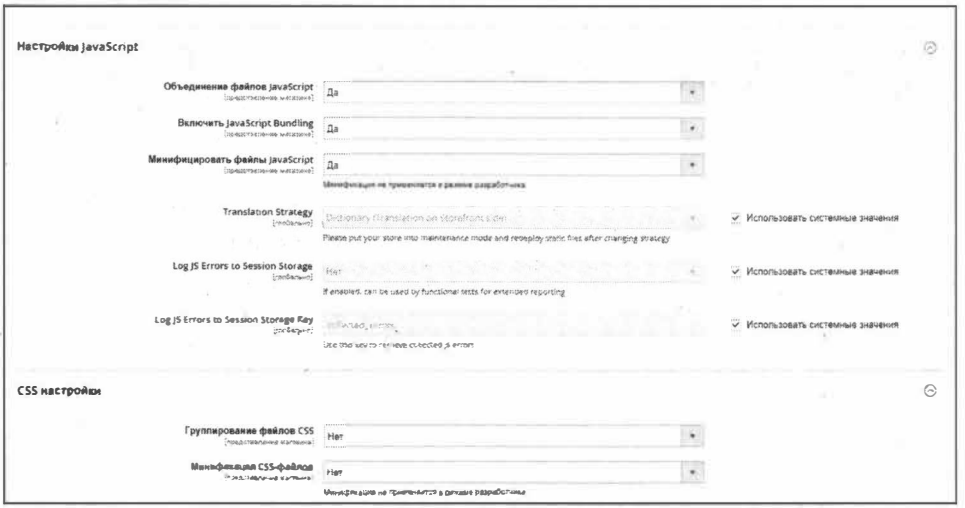


Рис. 18.8. Параметры JS/CSS

ресурсов в один файл сокращает количество запросов к серверу. Например, у вас было 20 сценариев JavaScript, следовательно, браузер должен был сделать 20 запросов к серверу. Вместо этого будет один запрос.

Перейдите в **Магазины, Конфигурация, Расширенные, Разработчик, Настройки Javascript /Настройки CSS**. Включите параметры **Объединение файлов JavaScript / Группирование файлов CSS** (или **Объединение файлов CSS** – название параметра зависит от локализации Magento). Также включите параметры **Минифицировать файлы JavaScript** и **Минификация CSS-файлов**. На рис. 18.8 с настройками JavaScript все хорошо – они объединяются и минимизируются. А вот файлы CSS отравляются как есть – без группировки и минимизации. Это тоже может быть причиной плохой производительности.

18.5.3. Оптимизируйте изображения

Используйте изображения подходящего размера и качества. Например, если слайдер на домашней странице подразумевает использование картинок размером 1920x1080, то не нужно использовать картинки в 2-3 раза больше – это только увеличит время загрузки страницы.

Существенно уменьшить размер JPEG-картинок можно, если уменьшить качество до 80%: визуально это не будет заметно, а вот размер файла станет намного меньше.

Также для ускорения загрузки изображений можно использовать CDN. Правда, они платные в большинстве случаев, но, тем не менее, использование CDN позволяет ускорить загрузку изображений, поскольку изображения будут загружаться с ближайшего к пользователю сервера.

18.5.4. Использование Flat-категорий и продуктов

Перейдите в раздел **Магазины, Конфигурация, Каталог, Каталог, Витрина** и установите параметры, как показано на рис. 18.9. Вы должны включить параметры **Использовать плоский каталог категории** и **Использовать плоский Каталог продукции**.

Рис. 18.9. Настройки плоских категорий и продуктов

Подобные настройки уменьшают количество запросов к БД и сайт будет работать быстрее.

18.5.6. Переводим Magento в production-режим

Magento сама по себе нерасторопная система. А в режиме разработчика она еще более медленная. После того, как все будет готово, вы можете перевести ее в режим production, который подразумевает компиляцию кода. Для этого подключитесь к своему серверу по SSH, перейдите в каталог с Magento и введите команду:

```
php bin/magento deploy:mode:set production
```

Переводить в режим production нужно только тогда, когда сайт полностью готов – установлены все необходимые расширения, тема оформления, внесены изменения в CSS/JS. Разница в том, что в режиме разработчика все изменения вносятся сразу же. В режиме production вам нужно выполнить развертывание системы после внесения изменений, иначе вы их попросту не увидите. Команда, выполняющая развертывание статических файлов:

```
php bin/magento setup:static-content-deploy -f
```

Для дополнительной информации о режиме разработчика обратитесь к документации Magento.

18.5.7. Настройка MySQL

При настройке MySQL обратите внимание на параметры раздела Fine Tuning. Опять тут все зависит от размера вашей базы данных и от размера оперативной памяти. На рис. 18.10 приведена конфигурация рабочего сервера.

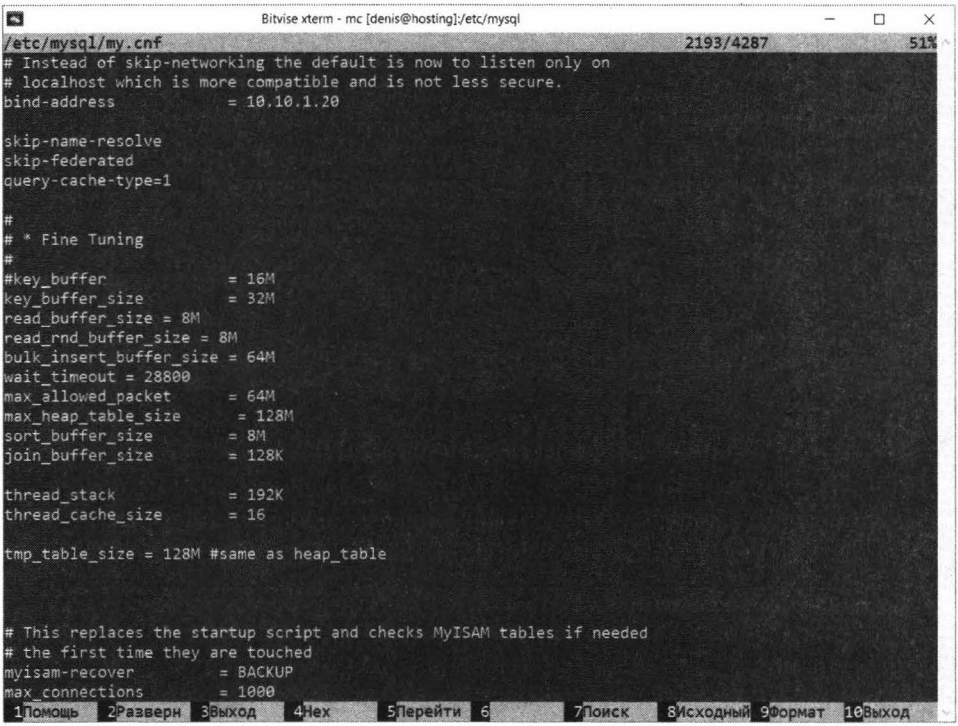
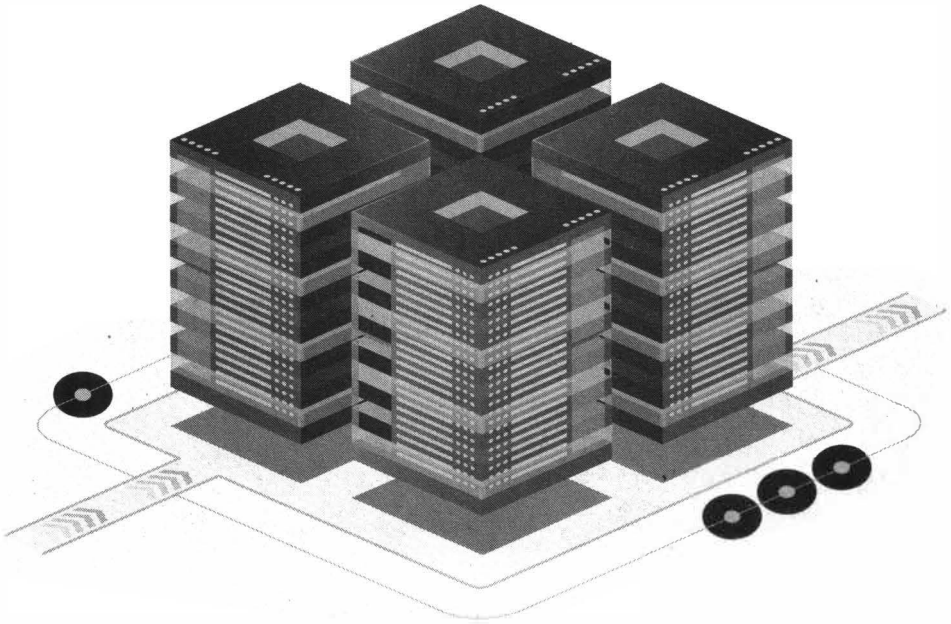


Рис. 18.10. Конфигурация MySQL для Magento

Глава 19.

Настройка почты на веб-сервере



Залог успешной работы веб-сервера – правильная настройка почты. Ведь, если почта настроена неправильно, он не сможет отправить пользователям важные уведомления:

- Письмо со ссылкой подтверждения регистрации – пользователь не получит письмо со ссылкой для подтверждения учетной записи и не сможет зарегистрироваться.
- Письмо с уведомлением о заказе – это не так страшно, но все равно не приятно.
- Письмо администратору с уведомлением о возникших проблемах – администратор ничего не узнает о проблемах на сервере.

Для успешной работы отправки писем PHP должен быть настроен правильно. В этой главе будет показано, как это сделать.

19.1. Проверяем, работает ли почта

Напишем небольшой сценарий mail.php:

```
<?php
      mail('ваш email', 'Test', 'Hello');
?>
```

Выполним сценарий:

```
php mail.php
```

Проверьте ваш почтовый ящик, в том числе и папку **Спам**. Если письмо есть, можете дальше не читать эту главу. Но, скорее всего, письма не будет – ведь никто кроме вас почту не настраивал.

19.2. Принцип настройки

Для отправки почты нам нужен SMTP-сервер. Но далеко не все облачные/обычные провайдеры предоставляют своим клиентам почтовые серверы. Это первый момент. Второй заключается в том, что стандартная функция mail() не поддерживает аутентификацию, а все современные SMTP-серверы просто так письма не отправляют.

Выход есть. Мы создадим почтовый ящик, на том же gmail, установим программу-посредник ssmtp, настроим ее соответствующим образом, а затем укажем PHP, что нужно отправлять почту через эту программу. Все довольно просто и на все про все (с регистрацией почты) у вас уйдет не более 15 минут.

19.3. Установка и настройка ssmtp

Итак, регистрируйте почтовый ящик, от имени которого ваш магазин будет отправлять письма. Возможно, уже такой ящик создан. Тогда можно приступить к установке ssmtp:

```
sudo apt install ssmtp
```

После этого открываем файл конфигурации /etc/ssmtp/ssmtp.conf и приводим его к следующему виду:

```
# Почта пользователя root
root=root@localhost
# Yes - будет писаться отладочная информация (в случае
проблем)
# No - ничего писаться не будет
debug=No

# Параметры SMTP-сервера - имя и порт (настройки для Google)
mailhub=smtp.gmail.com:587
```

```
# Перезаписать домен:
# rewritreDomain=your.shop.com

# Имя пользователя
AuthUser=email@gmail.com
AuthPass=PrettySecretPassword
UseSTARTTLS=Yes
```

Данная настройка идеально подходит для Google, все, что вам остается – ввести ваш адрес электронной почты и пароль. Также не забудьте настроить сам Google-аккаунт и разрешить использование небезопасных приложений. Инструкция находится по адресу:

https://support.google.com/accounts/answer/6010255?hl=ru

19.4. Настройка PHP

Откройте нужный файл php.ini. Он хранится, как правило, в каталоге /etc/php7/apache2/php.ini или /etc/php5/apache2/php.ini (если используется PHP 5). Найдите параметр sendmail_path и установите его так:

```
sendmail_path = /usr/sbin/ssmtp -t
```

Теперь попробуйте отправить почту функцией mail(). Все должно быть нормально. Если же нет, функция mail() сразу же вам сообщит о проблеме и это будет не просто сообщение, а сообщение с подробным описанием ошибки.

19.5. Возможные проблемы

Нужно понимать, что не только PHP будет использовать ssmtp. Эта программа при установке заменяет системную программу sendmail, следовательно, почта, отправляемая другими процессами, также будет поступать на указанный в конфигурационном файле почтовый ящик. Самый опасный процесс – cron. В его расписании могут быть команды, выполняемые раз в минуту. Следовательно, раз в минуту cron будет отправлять администратору отчет о выполнении команд. На практике это приведет не только к за-

хламлению почтового ящика, но и к тому, что Google перестанет вас обслуживать (посчитает спаммером, так как вы очень часто отправляете почту).

Что делать? Нужно отредактировать расписание cron как минимум для трех пользователей – текущего (от имени которого вы обычно работаете), www-data и root. Соответствующие команды:

```
crontab -e
crontab -e -u www-data
crontab -e -u root
```

В расписании нужно убедиться, что в качестве параметра MAILTO указана пустая строка:

```
MAILTO=»»
```

Вот теперь с почтой должно быть все хорошо. Если что-то пойдет не так, вы всегда сможете просмотреть журнал /var/log/mail.log (рис. 19.1).

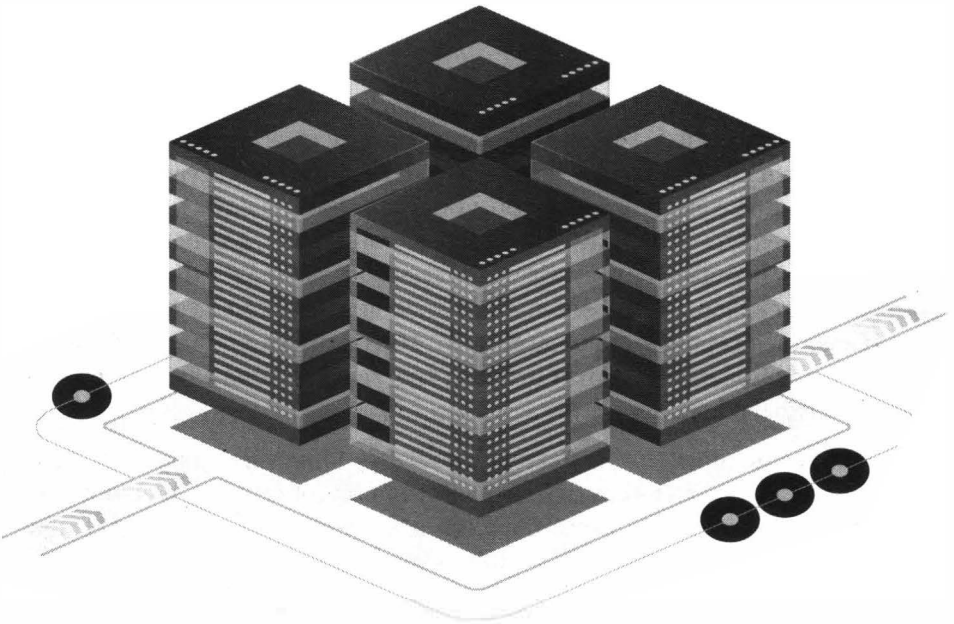
```
root@hosting:/var/log# tail mail.log
Feb 12 12:40:02 hosting sSMTP[30966]: X-Cron-Env: <HOME=/var/lib/sendmail>
Feb 12 12:40:02 hosting sSMTP[30966]: X-Cron-Env: <PATH=/usr/bin:/bin>
Feb 12 12:40:02 hosting sSMTP[30966]: X-Cron-Env: <LOGNAME=smmsp>
Feb 12 12:40:02 hosting sSMTP[30966]:
Feb 12 12:40:02 hosting sSMTP[30966]: /usr/share/sendmail/sendmail: 899: /usr/share/sendmail/sendmai
l: /usr/sbin/sendmail-msp: not found
Feb 12 12:40:03 hosting sSMTP[30966]: .
Feb 12 12:40:04 hosting sSMTP[30966]: 250 2.0.0 OK  1549968006 w23sm2127268wmc.38 - gsmtp
Feb 12 12:40:04 hosting sSMTP[30966]: QUIT
Feb 12 12:40:04 hosting sSMTP[30966]: 221 2.0.0 closing connection w23sm2127268wmc.38 - gsmtp
Feb 12 12:40:04 hosting sSMTP[30966]: Sent mail for smmsp@hosting (221 2.0.0 closing connection w23s
m2127268wmc.38 - gsmtp) uid=108 username=smmsp outbytes=736
root@hosting:/var/log#
```

Рис. 19.1. Журнал mail.log

Как видите, на все про все ушло немного времени и почта теперь настроена. Аналогичным образом можно использовать и другие аккаунты, а не только Google. При использовании Google помните, что максимум вы можете отправить не более 500 сообщений в день. Именно поэтому нам и пришлось отключить отправку писем планировщиком cron – он может генерировать слишком много писем и на отправку полезных писем пользователем уже не хватит квоты. Если 500 писем для вас – мало, рассмотрите использование Яндекса – там лимит 3000 сообщений в сутки, что значительно выше, чем у Google.

Глава 20.

Решение типичных проблем



Во время эксплуатации веб-сервера периодически возникают проблемы. Это не очень хорошо, но вполне нормально и в некоторых случаях – предсказуемо. В этой главе мы рассмотрим ряд типичных проблем и их решение.

20.1. Сайт недоступен. Поиск причины

Иногда сайт «падает», попросту говоря – становится недоступным. Если ваш сайт – не ваша персональная страничка, то простой может стоить как денег, так и репутации. С репутацией понятно. С деньгами тоже – вы не сможете продавать свой продукт, если сайт «лежит». Чем быстрее вы поймете, в чем причина, тем лучше. А чтобы оперативно узнавать, что сайт недоступен, рекомендуем или установить агент мониторинга вроде Zabbix (это сложнее, зато сообщаемая статистика более качественная – можно узнать и о нагрузке на сервер, и о переполнении дискового пространства и т.д.) или воспользоваться сервисами вроде Яндекс.Метрика – такие сервисы периодически опрашивают веб-сайт и отправляют уведомление администратору, если сайт недоступен.

Далее мы рассмотрим три наиболее вероятных причины «падения» сайта. Будем считать, что сайт «крутится» на виртуальном сервере, а если на физическом, то с «железом» все в порядке и нет никаких аппаратных неисправностей.

20.1.1. Закончилось дисковое пространство/свободные иноды

Самая банальная причина падения сайта. Во время работы сетевые сервисы вроде Apache, MySQL генерируют приличные объемы данных – журналы, временные файлы, не говоря уже о загружаемых пользователями данных.

Сервису просто некуда дальше писать данные и он «крашится». Также причиной падения того же Apache может быть чрезвычайно большой размер журнала. Проверить свободное дисковое пространство можно командой df –h.

Мы рекомендуем следующее:

- Чтобы регулярно не проверять дисковое пространство, установить сервис мониторинга для автоматизации данного процесса. Можно настроить тот же Zabbix или написать сценарий на bash для мониторинга дискового пространства.

- Настроить logrotate для ротации и сжатия файлов журналов основных сервисов. Как правило, после установки logrotate все необходимые настройки уже будут в конфигурационных файлах, администратору останется только убедиться в том, что logrotate запускается автоматически. Далее будет показано, как это сделать.

Куда подевалось дисковое пространство?

- Временные файлы – загляните в /tmp. Как правило, оттуда можно удалить сразу все.

- Файлы сессий - каталог зависит от настроек движка.

- База данных – для сокращения ее размера базу данных можно оптимизировать. Также можно выполнить специфические для движка запросы. Например, для WordPress следующий запрос удаляет комментарии со спамом, что может существенно сократить размер базы данных:

```
DELETE FROM wp_commentmeta WHERE comment_id NOT IN
(SELECT comment_id FROM wp_comments)
```

- Мэйл-боксы – стандартные почтовые ящики мало кто проверяет, а они могут быть заспамлены. Обычно они хранятся в /var/mail. Несколько гигабайтов спама – вполне реально.

- Журналы – просмотрите в /var/log. Лучше всего настроить logrotate для ротации и сжатия журналов, но если некоторые журналы слишком большие и из-за них сервис не запускается, их можно удалить.

- Резервные копии – возможно, вы не воспользовались услугой резервного копирования, а настроили создание бэкапов с помощью системного ПО. В этом случае проверьте каталог с бэкапами и удалите старые резервные копии.

А теперь внимание! Иногда заканчивается не свободное место, а свободное количество инодов (i-node). Результат такой же – система не может создать файл и сайт «падает» из-за подвисания сетевого сервиса. Куда подевались иноды? Некоторые движки, например, Magento, генерируют слишком много файлов сессий (обычно хранятся в /var/session, но все зависит от настроек движка). Такие файлы очень маленькие, места практически не занимают, зато крадут драгоценные иноды. Одна из тактик – регулярно чистить каталог сессий. В cron добавляется вот такой сценарий:

```
cd /var/session
for i in sess_*; do rm -fv $i; done
```

Недостаток этого способа в следующем:

- При удалении сессий стирается вся информация о пользователе, например, его корзина и данные о регистрации. Заходит пользователь в ваш магазин, выбирает товары и помещает в корзину. Тут запускается сценарий очистки и мало того, что корзина стерта, так ему еще и предлагают заново войти. Продаж магазину это не добавит.

- На одном из сайтов популярность была такова, что сессии очень быстро съедали все доступные иноды. Приходилось запускать их чистку чуть ли не раз в час, иначе сайт нормально не работал. Сами понимаете, раз в час всех пользователей, в том числе и менеджеров будет «выбрасывать» с сайта – дело не очень хорошее.

Альтернативное и более правильное решение заключается в создании еще одного диска (или раздела) с файловой системой reiserfs. Она не блещет производительностью, зато позволяет в одном блоке хранить несколько файлов – пока размер блока физически не заполнится. Reiserfs полностью решила проблему с инодами. Теперь сервер спокойно держит миллионы файлов инодов и чистить /var/session приходится довольно редко – на данном сервере автоматическая чистка вообще выключена и производится вручную администратором раз в месяц при регулярном обслуживании сервера.

20.1.2. Нехватка прочих ресурсов

Кроме дискового пространства, не нужно забывать о памяти и ресурсах процессора. Здесь поможет команда htop, которая показывает не только запущенные процессы, но и потребляемые ресурсы.

Одна из причин падения сайта – нехватка ресурсов, особенно, если речь идет о монстрообразных движках вроде Symfony, Magento и т.д.. Они чувствительны не только к оперативке, но и к количеству ядер.

Если у вас виртуальный сервер, вам повезло – добавить новые ресурсы можно с помощью пары кликов мышкой. А вот с физическим сложнее – придется докупать оперативную память и модернизировать сервер вручную. Все это – время и деньги.

20.1.3. Неправильная конфигурация сервера

Мало выделить ресурсы, важно еще их правильно использовать. Если программное обеспечение использует ресурсы неэффективно, сайт может «падать». Типичный пример – «крушение» Apache из-за нехватки максимального количества процессов, которые может запустить Apache. В логах сообщение выглядит примерно так:

```
[Sat Jun 27 11:59:59.443036 2020] [mpm_prefork:error] [pid 37252]
AH00161: server reached MaxRequestWorkers setting, consider raising
the MaxRequestWorkers setting
```

Здесь все казалось бы просто. Нужно открыть конфиг /etc/apache/apache2.conf и повысить значение MaxRequestWorkers. Но не спешите этого делать. Сначала выполните htop и посмотрите размер процесса Apache. Например, пусть процесс Apache занимает 100 Мб. Если вы установите 150 одновременных процессов, то понадобится 150 * 100 почти 15 Гб памяти и это нужно учитывать. Не забывайте, что для системы и других процессов (того же MySQL) также нужна память. Поэтому не переусердствуйте с этим параметром и не забывайте добавлять необходимые ресурсы в панели управления сервером.

Если же установить MaxRequestWorkers больше, чем может поместиться в памяти, в логах появится неоднозначная запись:

```
[Sat Jun 27 15:29:04.747616 2020] [core:notice] [pid 37252] AH00051:
child pid 32103 exit signal Segmentation fault (11), possible
coredump in /var/coredumps
```

Экспериментальным путем было выяснено, что причина в нехватке оперативной памяти.

Сервис MySQL особенно с его движком InnoDB тоже требователен к правильной настройке. База данных может легко подвиснуть при сохранении изменений вот с такой ошибкой:

```
SQLSTATE[HY000]: General error: 1205 Lock wait timeout
exceeded; try restarting transaction
```

Если MySQL еще жив, можно попробовать от админа БД ввести запрос:

```
SET GLOBAL innodb_lock_wait_timeout = 120;
```

Этот же оператор поможет, если перезапуск MySQL временно невозможен. Для постоянной установки значения конфигурации нужно открыть файл /etc/my.cnf и установить новое значение:

```
[mysqld]
innodb_lock_wait_timeout=120
```

После этого нужно перезапустить MySQL и попробовать поработать при новом значении.

Ошибки в работе могут быть вызваны и неправильной установкой параметра innodb_buffer_pool_size. В этом случае в логе может быть такое сообщение:

```
Memory size allocated for the temporary table is more than
20% of innodb_buffer_pool_size.
```

Такое сообщение будет сгенерировано, если размер памяти, выделяемый для временной таблицы на 20% больше, чем значение innodb_buffer_pool_size. Нужно просто увеличить значение этого параметра.

В большинстве случаев журналы помогут в расследовании падения сервера – в них даже приводятся четкие указания относительно того, какие параметры нужно изменить для нормальной работы сайта. Если подытожить, то причины падения сайта сводятся к нехватке ресурсов и неправильной конфигурации программного обеспечения.

20.2. Сценарии автоматического перезапуска служб в случае сбоя

Очень плохо, когда сайт падает. Гораздо хуже, когда он падает по непонятной причине. Иногда на установление причины сбоя может понадобиться продолжительное время, если все это время сайт будет «лежать» администратора могут просто уволить. К счастью, есть решение, позволяющее немного скрасить «горечь падения» и оно будет описано в этом разделе.

Суть решения в следующем: нужно настроить/написать самому средство мониторинга работоспособности сайта. Если сайт «упал», это средство будет перезапускать сервисы Apache и MySQL (или только Apache – все зависит от специфики сайта и причины сбоя).

Решение, конечно же, временное. Нужно найти и устранить причину падения, но зато такой «костыль» позволяет быстро восстановить работоспособность сайта: руководство даже не заметит сбой. Что же касается обычных пользователей, то даже если кто-то и получит ошибку Server Error, а через минуту сайт уже будет работать, вряд ли кто-то будет особо жаловаться.

Проверять работоспособность сайта можно двумя способами: или пытаться получить реальный файл (можно для этого создать пустой файл в корне, важно получить 200-ый ответ от сервера) или же мониторить наличие сервисов apache/mysql в памяти. Второй способ не защищает от ситуации, когда сервисы «подвисли» – в памяти они есть, но на самом деле сайт «лежит». Первый способ не позволяет убедиться в работоспособности MySQL: файл веб-сервер может и «отдать», а база данных может не работать.

20.2.1. Проверка работоспособности веб-сервера

Проще всего проверить работоспособность веб-сервера путем обращения к его главной странице: если получен ответ 200, значит с сервером все хорошо. Bash-сценарий будет выглядеть так (адрес можно заменить на https://, если используется защищенный протокол):

```
#!/bin/bash

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

Вызов сценария нужно поместить в расписание cron. Периодичность зависит от частоты падения сервера – если сервер сбоит иногда, можно раз в 5-10 минут, если сайт «лежит» часто – раз в минуту. Да, это создаст дополнительную нагрузку на сервер, но зато обеспечит его перезапуск в течение одной минуты.

20.2.2. Проверка работоспособности MySQL

Напишем небольшой PHP-сценарий, подключающийся к БД и возвращающий 1, если соединение удалось и 0, если соединение не работает. Собственно, код этого сценария очень прост:

```
<?
include «config.php»;     // здесь параметры доступа к БД

$mysqli = new mysqli($DBHOST, $DBUSER, $DBPASSWD, $DBNAME);

if ($mysqli->connect_errno) {
      echo "0";
}
else «1»;          // connect ok
?>
```

Затем создается bash-сценарий подобный этому:

```
#!/bin/bash
RESULT=$(/usr/bin/php test-mysql.php)
if [ $RESULT -eq 0 ]; then
echo "Restarting MySQL"
/etc/init.d/mysqld restart
fi
```

Как и в случае с первым нашим bash-сценарием, вызов этого сценария нужно поместить в расписание планировщика. Можно объединить эти два bash-сценария в один и вызывать все сразу.

20.2.3. Если падают процессы

Если процессы не виснут, а «падают», то есть после сбоя вообще нет процессов Apache/MySQL в таблице процессов, тогда поможет следующий сценарий:

```
#!/bin/bash
RESTART="/etc/init.d/apache2 restart"
PGREP="/usr/bin/pgrep"
```

```
HTTPD="apache2"
$PGREP ${HTTPD}
if [ $? -ne 0 ]; then
$RESTART
date >> /var/log/srvmon.log
echo "Apache restarted" >> /var/log/srvmon.log
fi

RESTARTM="/etc/init.d/mysql restart"
MYSQLD="mysqld"
$PGREP ${MYSQLD}
if [ $? -ne 0 ]; then
$RESTART
$RESTARTM
date >> /var/log/srvmon.log
echo "Services restarted" > /var/log/srvmon.log
fi
```

Он не только перезапускает «упавшие» сервисы, но и ведет небольшой лог – записывает время, когда сервисы были перезапущены.

20.2.4. Monit: если нет таланта программиста

Все приведенное ранее можно сделать и с помощью сервиса monit. Его задача – мониторинг работоспособности сервера. Если что-то не так, то monit может перезапустить ту или иную службу.

В Интернете есть множество статей, посвященных monit, к тому же у него очень хорошая документация. Поэтому переписывать ее не станем. Вместо этого приведем реальный конфигурационный файл мониторинга Apache:

```
check process apache with pidfile /var/run/apache2.pid
   group www
   group apache
   start program = "/etc/init.d/apache2 start"
```

```
   stop program  = "/etc/init.d/apache2 stop"
    # если загрузка cpu > 90% 5 циклов то перезапустить
процесс.
   if cpu > 90% for 5 cycles then restart
  # если не удается получить файл server-status, перезапустить
   if failed host localhost port 80 with protocol http and
request "/server-status" with timeout 25 seconds for 4 times
within 5 cycles then restart
   depend apache_bin
   depend apache_rc

 check file apache_bin with path /usr/sbin/apache2
   group apache
   include /etc/monit/templates/rootbin

 check file apache_rc with path /etc/init.d/apache2
   group apache
   include /etc/monit/templates/rootbin
```

Конфигурация monit, понятна даже новичку. На конкретном сервере была проблема с большой загрузкой процессора, после пика загрузки сервер падал. Поэтому было добавлено две проверки: если загрузка процессора выше 90%, сервис перезапускался и если сервер уже упал (не удается получить файл), то сервис тоже перезапускался.

20.3. Борьба с сессиями Magento: защита от переполнения диска

Magento хранит в сессиях информацию о деятельности пользователя на сайте, в том числе информацию о корзине посетителя. Примечательно, но эта информация так и не удаляется, в конечном счете, в каталоге var/session собирается огромнее количество файлов (конечно, если в настройках Magento указано, что сессии хранятся в файловой системе, а не в базе данных). Хранение сессий в базе данных решает эту проблему, но создает новую – база данных будет изрядно подтормаживать со временем.

Когда в каталоге var/session накопится очень много файлов, это приведет к тому, что обычная команда rm * работать не будет – вы получите сообщение о том, что список файлов слишком длинный. Другими словами, просто так их уже не удалишь.

Вторая часть проблемы в том, что на диске, на котором хранится каталог с сессиями, могут закончиться свободные иноды. Свободное пространство еще будет, но не будет свободных блоков для хранения файлов. Эта проблема характерна для файловых систем, которые используют минимум один блок для хранения одного файла. Представьте, что размер блока равен 4096 байтов, а файл занимает 200 байтов. Тогда эти 200 байтов займут весь блок размером 4096 байтов.

Для диагностики проблемы помогут следующие команды:

```
df -h
df -i
cd var/session
ls -l | wc -l
```

Первая возвращает информацию о свободном пространстве, вторая – об использовании инодов, последние 2 команды – подсчитывают количество файлов сессий. Несколько миллионов сессий за один месяц – вполне нормальное число.

Все это приводит к тому, что сама Magento начинает изрядно подтормаживать, а если закончатся свободные иноды, то сервер перестанет функционировать, поскольку система не сможет записать данные на диск.

20.3.1. Решение 1: сценарий

Следующий сценарий удаляет все файлы из каталога сессий:

```
#!/bin/bash
# измените путь к каталогу с сессиями
cd /путь/var/session
for i in sess_*; do rm -fv $i; done
```

Данный сценарий нужно запускать регулярно (добавить в расписание cron). Слишком часто запускать его не нужно – ведь при удалении файла сессии у пользователя будет очищена корзина, а все менеджеры будут «выброшены» из панели управления. Именно поэтому запускать его нужно не чаще раза в сутки и не в бизнес-время. Лучше всего часа в 3-4 ночи, когда посетителей на сайте минимум.

20.3.2. Решение 2: смена файловой системы

Если файлы сессий «плодятся» чаще, чем того хочется, тогда придется создать еще один виртуальный диск и отформатировать его как reiserfs. Преимущество этой файловой системы в том, что в одном блоке она может хранить несколько мелких файлов и проблема с недостатком инодов будет решена сама собой. Да и работает reiserfs с множеством мелких файлов увереннее – Magento перестанет тормозить.

Отформатировать раздел диска как reiserfs можно так:

```
sudo mkreiserfs /dev/sdb1
```

Здесь раздел /dev/sdb1 будет использовать файловую систему reiserfs. Учитывая, что сессии занимают немного места, не нужно покупать огромный виртуальный диск. Несколько гигабайтов (до 10) будет вполне достаточно.

Затем этот раздел нужно указать в /etc/fstab для его автоматического монтирования (будем считать, что монтироваться будет к /mnt/reiserhdd – этот каталог должен существовать):

```
/dev/sdb1    /media/reiserhdd  reiserfs          defaults
1     0
```

Подмонтируем каталог:

```
sudo mount /dev/sdb1
```

Осталось указать новый каталог в файле app/etc/config.xml:

```
<default>
      <system>
            <filesystem>
                  ...
                  <session>/media/reiserhdd</session>


</default>
```

20.4. Ротация журналов сервера

Журналы в Linux иногда разрастаются до неприличных размеров, что приводит к снижению производительности и в некоторых особо запущенных случаях может даже вызвать переполнение дискового пространства. К счастью, администратор может использовать утилиту logrotate для управления журналами.

Logrotate – это утилита, выполняющая ротацию и сжатие файлов протоколов (журналов). При правильной настройке файл журнала никогда не разрастется до огромных размеров. Если же Logrotate не установлена или же неправильно настроена, файлы журналов некоторых сервисов (например, Apache) могут занять все доступное дисковое пространство.

По умолчанию Logrotate установлена в Ubuntu, начиная с версии 16.04 и настроена для обработки журналов всех установленных пакетов и приложений, в том числе rsyslog. В других дистрибутивах она может быть даже не установлена, а на некоторых виртуальных серверах – по непонятным причинам – не настроена.

Не смотря на то, что Logrotate установлена по умолчанию, будет нелишним проверить ее конфигурацию. На одном из моих серверов почему-то не выполнялась ротация журналов Apache, в результате произошло переполнение дискового пространства.

Конфигурация Logrotate хранится в следующих местах:

- Файл /etc/logrotate.conf – основной файл конфигурации. Как правило, он содержит некоторые параметры по умолчанию, настройки для нескольких журналов, не принадлежащим системным пакетам и инструкцию include для подключения конфигурации, хранящейся в файлах каталога /etc/logrotate.d

- Каталог /etc/logrotate.d содержит файлы с конфигурацией Logrotate. Здесь вы найдете конфигурацию для rsyslog, apport, dpkg и других системных пакетов. Каждый файл – это конфигурация ротации того или иного журнала. Вы можете добавить свои файлы конфигурации, которые также будут обработаны утилитой Logrotate.

Рассмотрим конфигурацию ротации журналов для apport – файл /etc/logrotate.d/apport:

```
/var/log/apport.log {
      daily
      rotate 7
      delaycompress
      compress
      notifempty
      missingok
}
```

Вот что означают все эти параметры:

- daily – ротацию выполнять ежедневно (для редко используемых сервисов можно использовать команду monthly – ежемесячно или weekly - еженедельно).
- rotate 7 – хранить последние 7 файлов журналов.
- compress – сжать обновленные файлы (по умолчанию используется сжатие gzip)
- delaycompress – отложить сжатие предыдущего файла журнала

до следующего циклического сдвига. Эта директива имеет силу только в комбинации с compress. Это может быть использовано в том случае, если некой программе нельзя указать закрыть ее файл журнала, и таким образом, можно некоторое время продолжать запись в предыдущий файл журнала.

- notifempty – не ротировать пустой файл.
- missingok – не записывать сообщение об ошибке, если журнал пуст.

Конечно, это далеко не все допустимые параметры конфигурации. О дополнительных вы можете узнать в справке (команда man logrotate) или по ссылке **https://www.opennet.ru/man.shtml?topic=logrotate&category=8&russian=0**.

Теперь представим, что у нас есть некий сервис daemond, хранящий свои файлы журналов в каталоге /var/log/daemond. Нужно настроить ротацию журналов этого сервиса.

Все достаточно просто: нужно в /etc/logrotate.d создать файл daemond (название может быть другим, важно, чтобы вы понимали, что находится в этом файле сразу по его названию) и заполнить конфигурацию ротации.

Создадим файл конфигурации ротации:

```
touch /etc/logrotate.d/daemond
```

Конфигурация может быть такой:

```
/var/log/daemond/*.log {
      daily
      missingok
      rotate 7
      compress
      notifempty
      create 0640 daemon-data daemon-data
      sharedscripts
```

```
    postrotate
    systemctl reload daemond
    endscript
}
```

Первая строчка задает файлы журналов. В нашем случае – это все log-файлы из каталога /var/log/daemond. С первыми пятью командами вы уже знакомы. Команда create определяет, что после ротации будет создан новый пустой файл журнала с правами 0640, владельцем daemon-data и группой daemon-data. Если ваш сервис выполняется от имени какого-то пользователя (например, daemon-data), а не от root, то очень важно указать правильно имя пользователя и группы.

Параметр sharedscripts означает, что скрипт postrotate будет выполняться только один раз, а не после обработки каждого файла.

Параметры postrotate и endscript позволяют указать скрипт, который будет запущен после того, как файл журнала обновится. В примере приложение перезагружается.

Осталось настроить Logrotate на автоматический запуск. Чтобы эта утилита запускалась автоматически, от имени root введите команду crontab –e и в появившемся редакторе добавьте строку:

```
10 * * * * /usr/sbin/logrotate /etc/logrotate.conf
```

Сохраните файл и выполните выход из редактора. Мы создали расписание планировщика, которое будет выполнять logrotate на 10-ой минуте каждого часа. Конфигурация будет загружаться из файла /etc/logrotate.conf.

20.5. Десятка утилит для мониторинга сервера

Мониторинг сервера – одна из обязанностей администратора, а сейчас мы рассмотрим некоторые полезные инструменты, позволяющие сделать выполнение этой обязанности комфортнее. В этом разделе будут рассмотрены только простые утилиты, не требующие сложной настройки и даже установки – многие из них устанавливаются по умолчанию. Конечно, для пол-

ноценного мониторинга необходимо установить систему вроде Zabbix, но не всегда администратор захочет разбираться с установкой довольно сложной системы, если есть с десяток отличных и простых утилит.

20.5.1. htop – информативная версия top

Команда htop является усовершенствованной версией всем нам знакомой команды top. Кроме информации об использовании системных ресурсов процессами, которая мало чем отличается от команды top, но представлена в более удобном виде, команда htop строит псевдографические графики загрузки каждого ядра процессора, памяти и свопа. В принципе практически то же самое, что и top, но красивее и нагляднее.

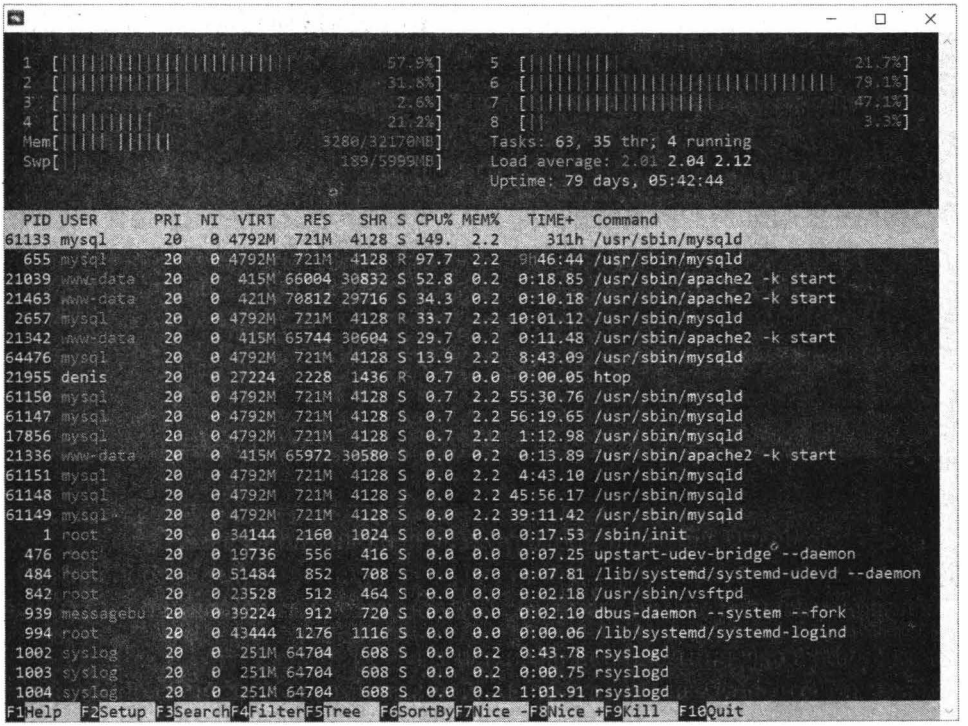


Рис. 20.1. htop – более информативная версия top

20.5.2. atop – продвинутый монитор процессов

Если команда htop – более наглядная версия top, то atop – совсем другая зверушка. Это продвинутый интерактивный монитор производительности. В отличие от top, выводит только новые изменения об активных системных процессах. Позволяет контролировать нагрузку процессора, памяти, накопителя, сети, а также просматривать распределение нагрузок по работающим процессам. В Ubuntu и Debian запускается автоматически и постоянно записывает информацию о процессах в /var/log/atop.log.

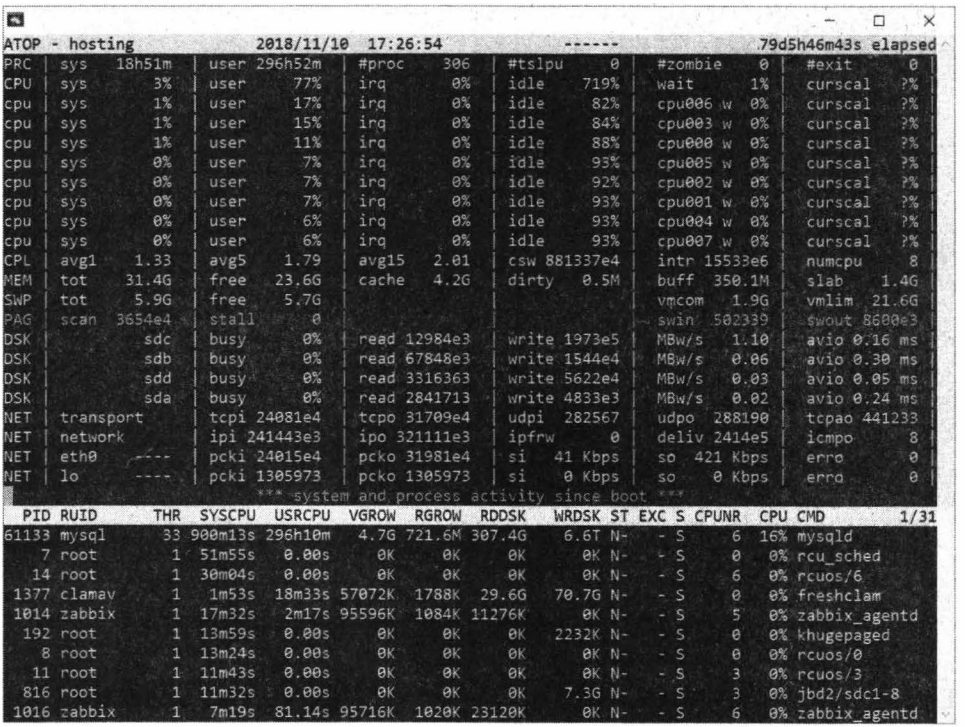


Рис. 20.2. atop – монитор производительности

20.5.3. apachetop – мониторинг веб-сервера в реальном времени

Утилита apachetop не устанавливается по умолчанию, для ее установки нужно инсталлировать одноименный пакет. Опция –f позволяет задать

файл access_log, если его расположение отличается от стандартного или если нужно мониторинг журнал доступа определенного виртуального хоста. Предоставляет информацию о текущем положении дел веб-сервера и даже о URL, к которым обращаются в данный момент.

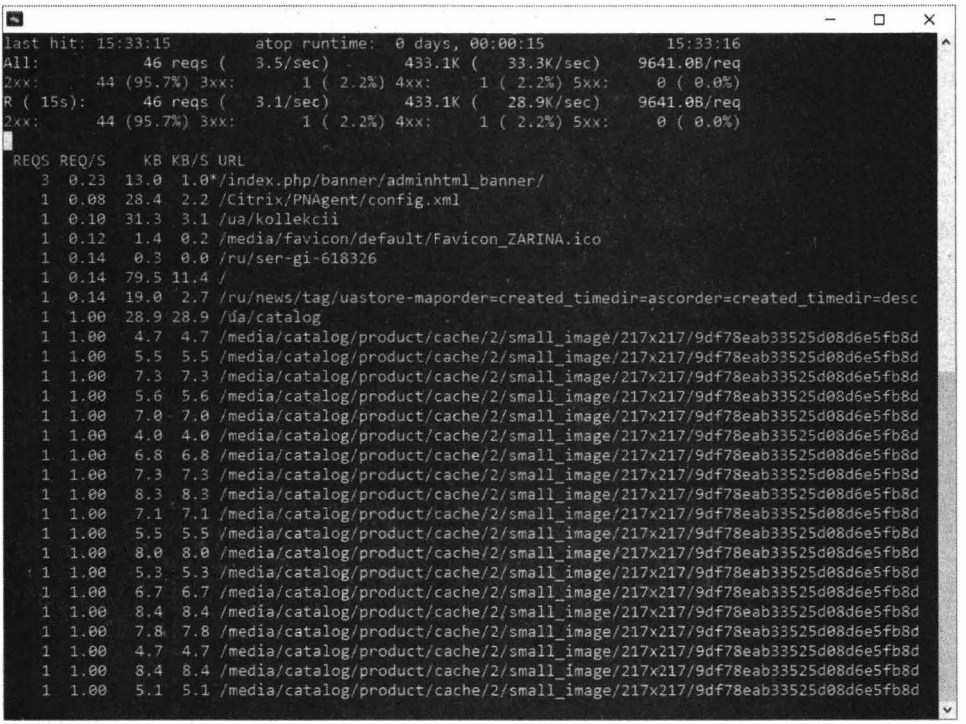


Рис. 20.3. Утилита apachetop

20.5.4. mytop – мониторинг MySQL

Если есть утилита мониторинга Apache, то, вероятно, должна быть и утилита мониторинга MySQL. Для ее запуска нужно указать три параметра:

- -u – задает имя пользователя;
- -p – задает пароль;
- -d – база данных.

Предоставляет информацию о нагрузке на сервер базы данных и даже показывает текущие SQL-запросы, поступаемые от MySQL-пользователей.

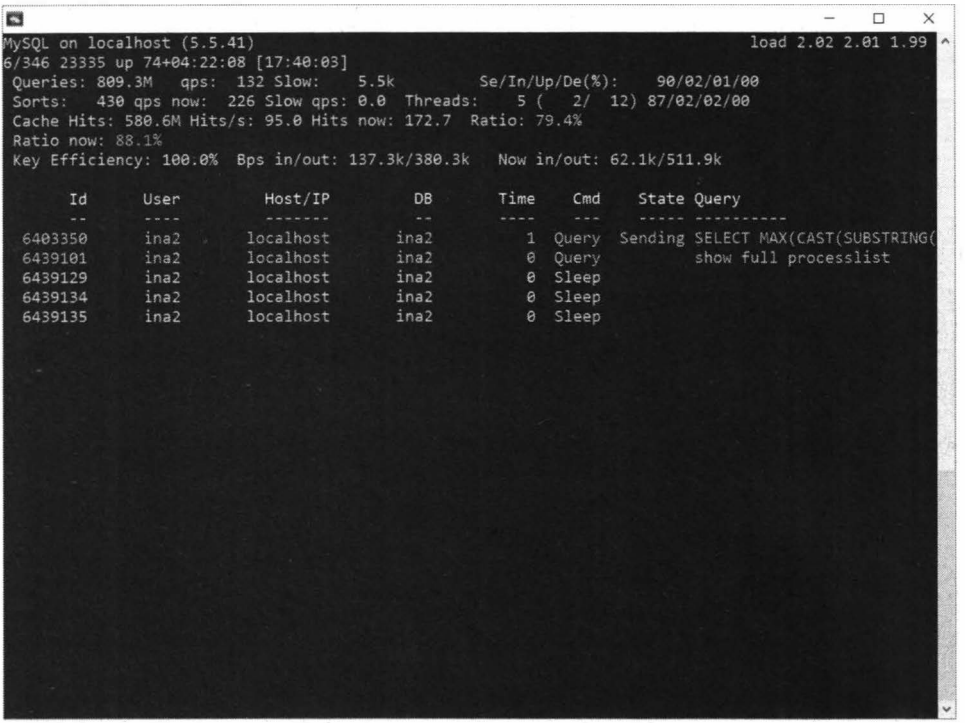


Рис. 20.4. Утилита mytop: мониторинг MySQL

20.5.5. iotop – мониторинг ввода/вывода

Сервер тормозит? Есть подозрение на дисковую подсистему? Утилита iotop позволяет определить процесс, узурпировавший всю подсистему ввода/вывода.

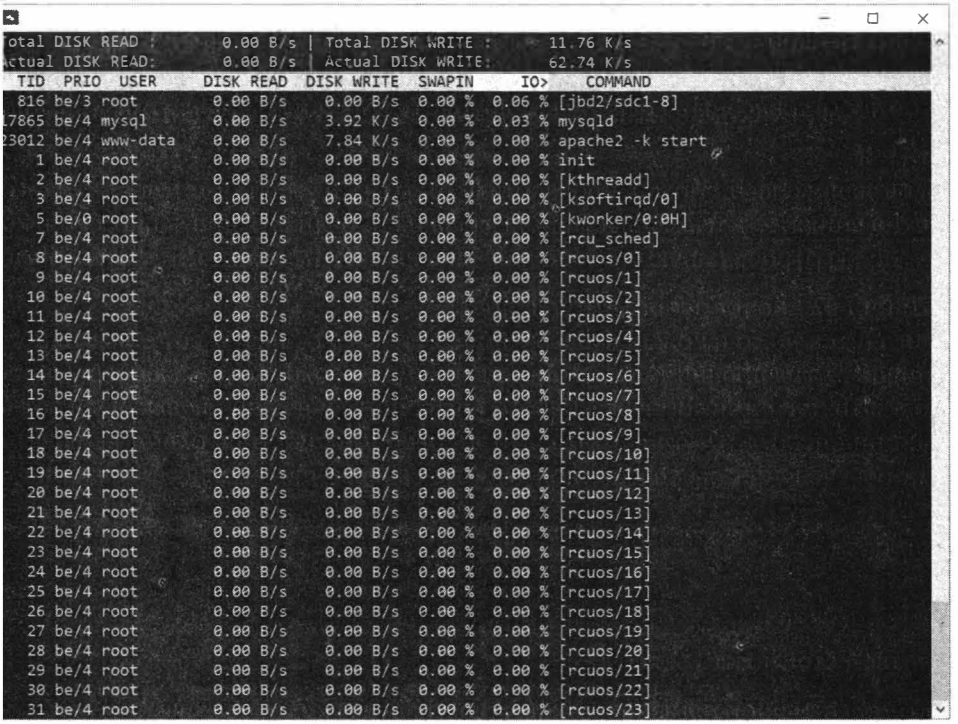


Рис. 20.5. Утилита iotop

20.5.6. iftop – мониторинг сетевого интерфейса

Утилита iftop похожа на утилиту top, но вместо проверки использования процессора она прослушивает сетевой трафик на выбранных интерфейсах и отображает таблицу текущего использования. Она позволяет ответить на вопрос: «Почему мое соединение такое медленное?».

Если у вас – виртуальный сервер, то вы будете удивлены, когда узнаете, какими жадными могут быть облачные провайдеры. Некоторые из них выделяют канал для виртуального сервера – не более 10 Мбит/с. В 2020 году... А дополнительная пропускная способность будет доступна, правильно, за дополнительную плату.

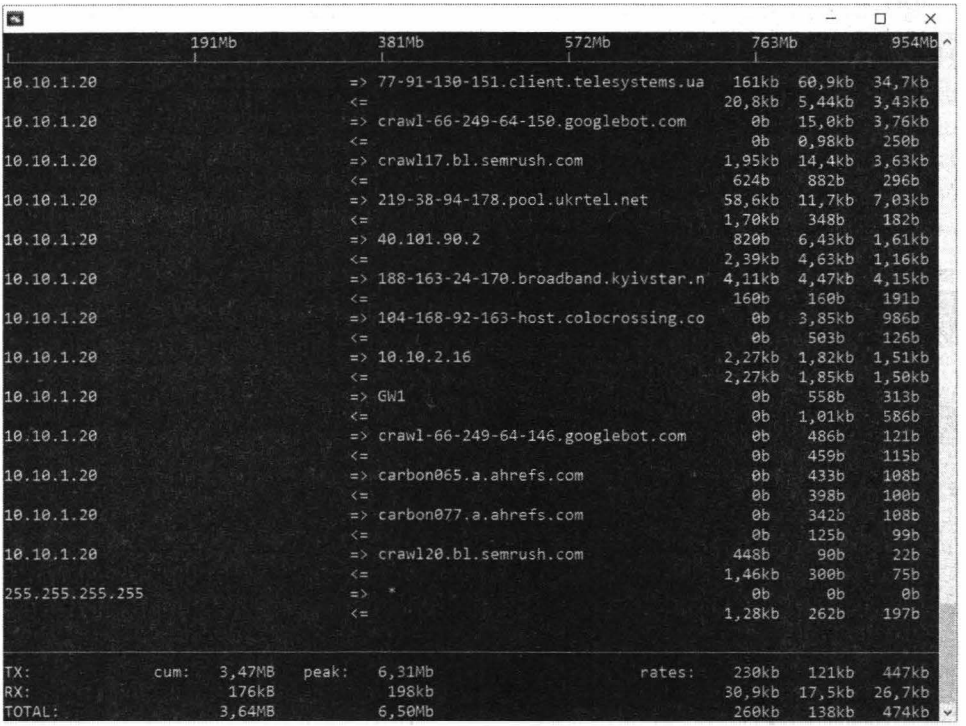


Рис. 20.6. Утилита iftop

20.5.7. jnettop – еще один монитор сетевого интерфейса

На первый взгляд похожа на iftop, но отличия есть и существенные. Если iftop показывает только обмен между узлами, то jnettop показывает и URL, и передаваемый в данный момент файл. Можно увидеть, какие файлы (картинки, сценарии, HTML-страницы и т.д.) передаются в данный момент клиентам. Не устанавливается по умолчанию, для ее установки нужно инсталлировать одноименный пакет.

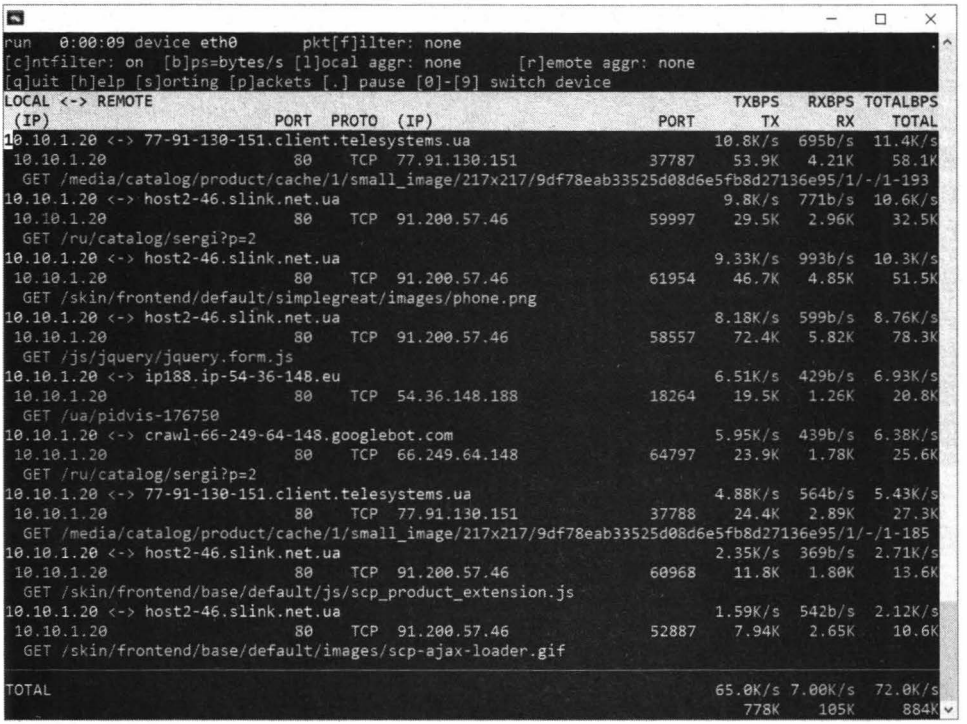


Рис. 20.7. Утилита jnettop

20.5.8. iptraf – мониторинг трафика

Утилита предоставляет различные полезные метрики вроде счетчика TCP-пакетов, счетчика байтов, статистику по интерфейсу, индикаторы активности и т.д.

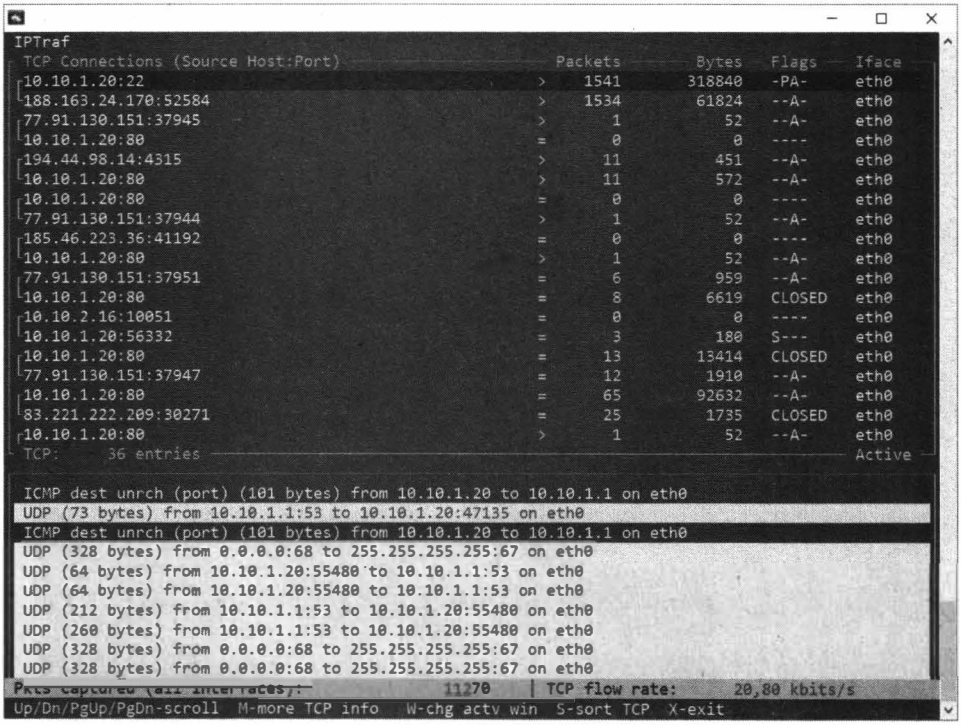


Рис. 20.8. Мониторинг трафика с помощью iptraf

20.5.9. ngrep – утилита для профессионала

Утилита ngrep – это то же самое, что и обычный grep, но для сетевого уровня. Она позволяет просмотреть все передаваемые пакеты. Есть возможность задания фильтров, в том числе с помощью регулярных выражений. Примеры использования ngrep можно найти по этой ссылке: http://ngrep.sourceforge.net/usage.html.

Рис. 20.9. Утилита ngrep

20.10. nmon – швейцарский нож

Утилита nmon – мониторинг на все случаи жизни. Запустите ее и выберите объект, по которому вы хотите получить статистику – процессор, память, диски, ядро, сеть, виртуальная память и т.д.

В заключение этой статьи отметим, что для выхода из большинства утилит нужно использовать q, Ctrl + X или Ctrl + C. Один из этих способов да сработает – это на случай, если программа не выводит подсказку.

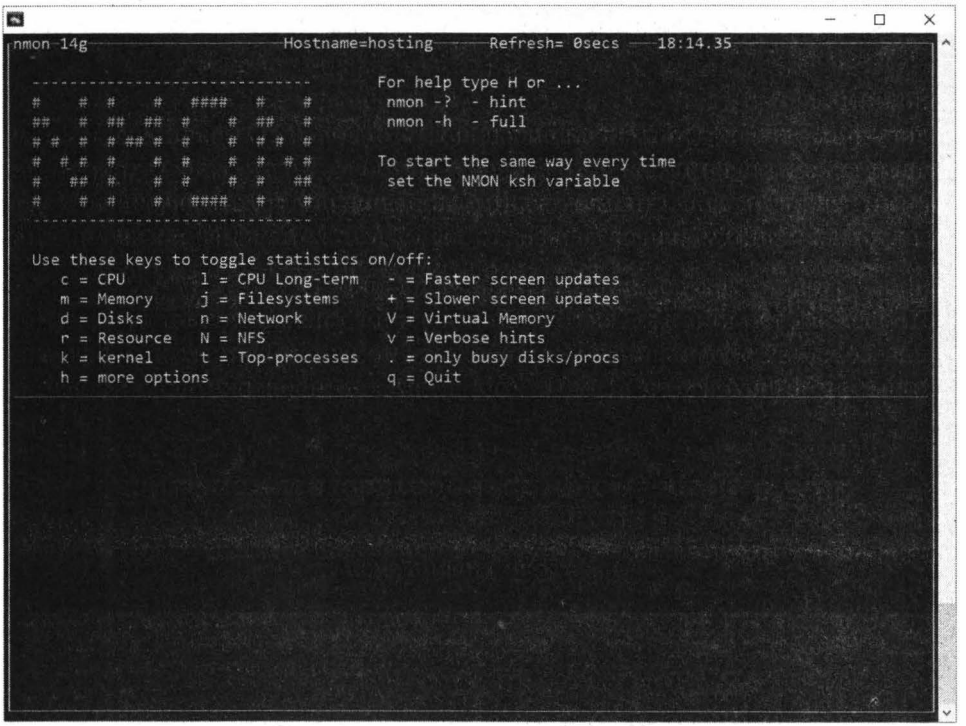
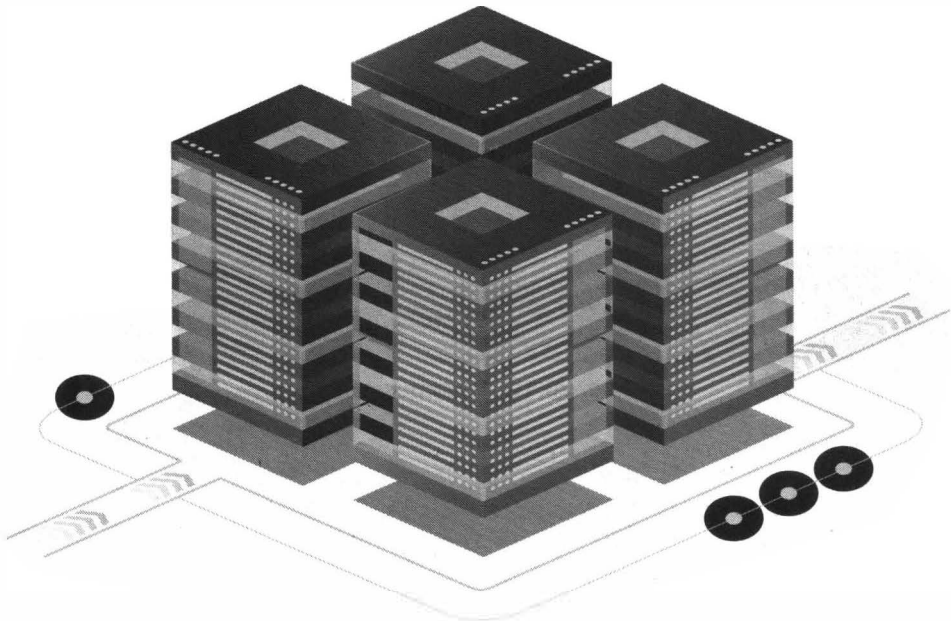


Рис. 20.10. Утилита nmon

Глава 21.

Переезд в облако



Ранее, было сказано, что содержание виртуального сервера обходится гораздо дешевле содержания физического сервера. Попробуем убедиться, так ли это.

21.1. Сколько стоит физический сервер

Возьмем средний конфигурации физический сервер. Ничего примечательного, пусть это будет HP Proliant DL180. Конфигурация весьма посредственная – 8 ядер на Xeon Sliver 4110 2,1 ГГц (не самый мощный Xeon), 16 Гб ОЗУ и нет ни одного накопителя – их придется купить отдельно. Цена такого варианта – 136 260 рублей.

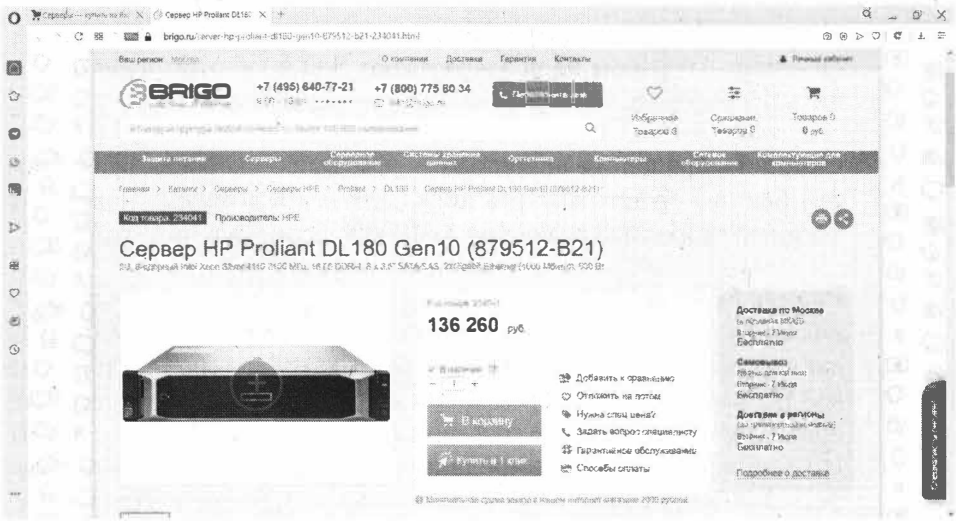


Рис. 21.1. Стоимость физического сервера

Добавьте к этой стоимости 1 или 3 накопителей. В двух накопителях особо нет смысла, поскольку RAID-массив с чередованием вы не построите из двух дисков, разве что в режиме зеркалирования, когда второй диск будет точной копией основного диска, но это очень нерационально с точки зрения расходования ресурсов. Но тем не менее такая защита данных лучше, чем вообще ничего.

Посмотрим, сколько стоят серверные накопители. Здесь все зависит от совести администратора и жадности руководства. Если устанавливать то, что действительно является серверным диском, а не только называется ним, то цена будет около 13 500 рублей (по состоянию на 3 июля 2020 года) за один диск. За эти деньги вы получите SAS-диск со скоростью вращения шпинделя 15000 об/минуту, с буфером 16 Мб, но емкостью всего 300 Гб – это вполне нормально для сервера. Конечно, можно купить обычный SATA-диск со скоростью вращения 7200 об/минуту и ценой 5000 рублей. При этом емкость этого диска будет составлять не 300 Гб, а 1 Тб. Если скорость и надежность для вас – не пустой звук, то вы выберите первый вариант, если же на первом месте экономия – то второй вариант будет предпочтительнее.

При выборе диска нужно обращать внимание на следующие характеристики:

- **Форм-фактор** – наш сервер поддерживает 3.5» диски, поэтому подбираемые под него диски должны быть в этом форм-факторе. Другие (2.5») не подойдут.

- **Интерфейс** – наш сервер поддерживает интерфейсы SAS и SATA. Некоторые серверы не поддерживают SATA, об этом нужно помнить. Некоторые, наоборот, не поддерживают SAS.

- **Скорость вращения шпинделя** – чем выше скорость вращения, тем быстрее будет работать диск. Для сервера нужно подбирать диски со скоростью вращения от 10000 оборотов в минуту.

- **Размер буфера** – чем выше размер буфера, тем быстрее будут происходить операции записи данных. В то же время – тем больше данных вы потеряете, если произойдет какой-то сбой до того, как все эти данные будут перемещены из буфера на сам диск. Мы выбрали диск с небольшим буфером на сегодняшний день – 16 Мб. Но есть в продаже более продвинутые накопители с размером буфера 128 Мб, правда, и емкость там около 2Тб и цена совсем другая.

- **Состояние** – как ни странно, но обязательно обращайте внимание на состояние (новый или бу) покупаемого диска. На том же Яндекс.Маркете можно найти б/у SAS-диски по цене от 3000 рублей. Гарантии на такие диски, как правило, или нет вообще или символическая гарантия 1-3 месяца. Как они будут работать в будущем – никому не известно. Скорее всего, они свой ресурс уже отработали, их заменили на новые, а старые – продают. Особенно на других досках объявлений веселят объявления «серверы их Европы» или «компьютерная техника из Европы». Жесткие диски изнашиваются из-за вращения, а не из-за своего месторасположения.

Итак, пусть разум победил над желанием сэкономить и вы остановились на варианте за 13500 рублей, но все-таки желание сэкономить присутствовало, поэтому вы купили два, а не три диска, решив, что 300 Гб суммарного пространства будет достаточно. В принципе да, если не заниматься обработкой видео и не хранить фотографии тысяч пользователей, то для организации того же Интернет-магазина, такого пространства хватит с запасом.

Считаем стоимость сервера:

```
136260 + 13500 x 2 = 163260
```

Для обеспечения бесперебойной работы сервера нам нужно помещение с кондиционером. Будем считать, что таковое имеется и не будем включать его в затраты. Стоимость резервного Интернет-канала не так высока и тоже спишем это на прочие затраты организации, тем более, что резервный канал пригодится не только для сервера, но и для всего офиса. А вот о резервном питании нужно поговорить отдельно.

Выбору ИБП посвящаются довольно большие и глубокие статьи, в которые нужно вникать. У нас сейчас другая задача – просчитать примерно, сколько часов наш сервер сможет проработать от ИБП и узнать стоимость такого ИБП.

Стоимость серверных ИБП может доходить до 1 млн. рублей. Нам, конечно же, такие не нужны, но стоимость «бесперебойника» все равно будет достаточно высокой.

Мощность источника бесперебойного питания для IT-оборудования определяется по тому же принципу, что и для любой другой техники. Необходимо просуммировать мощности всех подключаемых к устройству по-

требителей и сверх этого заложить запас, компенсирующий возможные эксплуатационные перегрузки (актуальное значение - 30%). Подбор подходящего ИБП производится по полученному в результате вышеуказанных действий значению (с округлением в большую сторону).

В технической документации и на заводских этикетках IT-оборудования часто указывается максимальная мощность блока питания, а не действительное энергопотребление устройства. Рекомендуется уточнить реальную мощность, потребляемую нагрузкой, данные можно запросить у производителя, либо произвести самостоятельные замеры с помощью электроизмерительной аппаратуры (мультиметры, ваттметры).

Номенклатура (мощностные линейки) большинства производителей ИБП строится на основе полной мощности измеряемой в вольт-амперах – ВА. Если мощность электрической нагрузки представлена только в ваттах – Вт (активная мощность), то перевод в вольт-амперы осуществляется делением на коэффициент мощности – Р (может обозначаться как cosφ или PF), равный для простейшего IT-оборудования – 0,6 - 0,8.

Современное серверное и сетевое оборудование может быть оснащено блоком питания с коррекцией коэффициента мощности (PFC), приближающей его значение к единице – 0,99. Если уверенности в наличии данной функции нет, то применяется типовое значение из указанного интервала.

Обратите внимание – в характеристиках источника бесперебойного питания указываются входной и выходной коэффициенты мощности, зависящие от электронной схемы самого устройства:

- входной – отражает влияние ИБП на внешнюю сеть и не имеет прямого отношения к подключаемой нагрузке.
- выходной – необходим при определении максимальной нагрузки в ваттах, которую устройство способно запитать, для этого умножаем полную мощность ИБП на выходной коэффициент мощности.

Расчет полной мощности защищаемой техники следует проводить, используя соответствующий ей коэффициент мощности, а не значения входного и выходного коэффициентов ИБП (на практике cosφ прописывается в руководстве по эксплуатации большинства потребителей электрической энергии)!

Полная (ВА) и активная мощность (Вт) правильно выбранного ИБП должна быть не меньше соответствующих мощностей подключенных электро-

приемников, а для гарантированно надежной работы - превышать их. Вычислить примерное время работы от ИБП можно по ссылке http://proline.biz.ua/calculator-ups-time-count. Для расчета времени вам нужно указать:

Среднюю потребляемую мощность – нужно учитывать не только мощность блока питания (точнее потребляемую мощность) сервера, но и другого оборудования, которое будет подключено к ИБП. Помните о 30%. Если вы вычислили, что средняя потребляемая мощность будет 700 Вт, добавим еще 30%, получим **910** Вт.

- Суммарная емкость аккумуляторов – укажите емкость ИБП, планируемого для покупки. Пусть это будет 1000 А/час.
- Остальные параметры оставьте по умолчанию, если вам не понятно их назначение, нажмите вопросительный знак возле названия параметра.

Получится, что при средней потребляемой мощности в 910 Вт и суммарной емкости аккумуляторов 1000 А/ч, наше оборудование сможет проработать 7 часов 11 минут. Вполне солидно – за это время смогут отремонтировать возникшие проблемы с электропитанием.

Рис. 21.2. Стоимость серверного ИБП

Теперь попробуем подобрать ИБП. Поскольку мы не планируем менять батареи раз в году, то лучше покупать ИБП с небольшим запасом, чтобы планировать старение батарей. Неплохим вариантом будет ИБП APC SMC1500I-2U Smart-UPS C 1500VA (1500 ВА/**900** Вт). Цена 57860 рублей за именно серверный ИБП с возможностью горячей замены батарей (без выключения оборудования) и возможностью установки в стойку, действительно немного.

Получилось 221 120 рублей – без стоимости обеспечения охраны помещения, организации видеонаблюдения и т.д.

21.2. Стоимость содержания физического сервера

Стоимость содержания физического сервера состоит из следующих факторов:

- Стоимость основного и резервного Интернет-канала. В стоимость виртуального сервера уже входит один выделенный IP-адрес и канал со скоростью 10 Мбит/с. И будьте уверены: канал резервируемый. Никто не захочет, чтобы серверы клиентов остались без Интернета. Вам же для своего физического сервера придется обеспечить резервный канал самостоятельно и платить за оба канала.
- Стоимость электричества. Мы уже посчитали, что будем потреблять 900 Вт (при средней нагрузке). На практике эта цифра может быть выше. Стоимость электроэнергии несложно подсчитать самостоятельно. Сюда же посчитайте затраты на кондиционирование помещения, в котором стоит сервер - ему не должно быть слишком жарко, что особенно актуально для летнего периода.
- Стоимость охраны серверной комнаты. Охрана может осуществляться как собственными силами (в этом случае будут расходы на зарплату охранникам), так и путем подключения к пульту охранного предприятия. В любом случае за физическую безопасность оборудования нужно платить.
- Амортизация оборудования. Ваше железо – это ваше железо. Гарантийный срок составляет 12-24 месяца в зависимости от

производителя оборудования. Даже именитые вендоры вроде HP редко предоставляют более длительную гарантию. Если по истечении гарантийного срока любые комплектующие могут выйти из строя. Как правило, первыми выходят со строя блоки питания, модули оперативной памяти и жесткие диски. Это группа риска. Так что можете включить в будущие расходы стоимость перечисленных компонентов. Также добавьте время простоя – пока вы купите и установите «железо» может пройти несколько часов, а то и дней. Для кого-то – это допустимо, а кто-то может потерять сотни тысяч рублей прибыли (например, если идет речь о популярном магазине).

Другими словами, есть постоянные затраты, которые вам придется нести каждый месяц.

21.3. Варианты снижения стоимости владения

Стоимость владения можно снизить:

- Размещение сервера в дата-центре (colocation) – если сервер уже куплен, его можно разместить в дата-центре. Да, ваш сервер будет физически размещен в дата-центре. При этом вам не нужно платить ни за охрану, ни за электричество, ни за кондиционирование. Вам не нужно заботиться о резервном канале, об ИБП и т.д. Обо всем этом позаботиться провайдер – и о резервировании канала, и о бесперебойной работе сервера. И все это стоит от 3 до 5 тысяч рублей в месяц. Средняя стоимость размещения 1 юнита с выделенным каналом 100 Мбит/с составляет 3000 рублей в месяц.

- Аренда виртуального сервера – если сервера пока нет, гораздо дешевле арендовать виртуальный сервер. Средняя стоимость сервера средней конфигурации (4 ядра по 2.8 ГГц, 8 Гб ОЗУ, 100 Гб накопителя) составляет 1790 рублей. Когда ваш проект выйдет за эти рамки, вы можете арендовать виртуальный сервер, приближенный по конфигурации к нашему физическому

(8 ядер по 2.8 ГГц, 16 Гб ОЗУ, 260 Гб накопителя) – он стоит в среднем 3600 рублей.

- Виртуализация физического сервера – если сервер уже настроен, вы оценили стоимость затрат на его содержание, потом посмотрели на стоимость colocation и поняли, что оно всего лишь на 600 рублей дешевле необходимой вам конфигурации сервера. Решение правильное. Далее будет показано, как виртуализировать физический сервер.

Примечание. Если разместить свой физический сервер в ДЦ, то у вас становится гораздо меньше головной боли и из всех расходов остается стоимость colocation, стоимость ПО (если нужно) и зарплата администратору (именно тому человеку, который будет отвечать за здоровье железа сервера). Конечно, спустя два года добавится и стоимость технического обслуживания. Гарантия на узлы сервера составляет как раз 2 года, а в составе сервера есть много механических частей, которые могут выйти из строя - вентиляторы, жесткие диски и т.д.

21.4. Что лучше? Муки выбора

У виртуального сервера есть свои преимущества. Одно из них - простота обслуживания. Администратор сервера не нужен как таковой, поскольку управление сервером, его конфигурацией осуществляется через удобную панель управления и вы можете в любой момент сотворить с ним все, что угодно. Например, модернизировать сервер, добавив еще оперативной памяти или дополнительный жесткий диск всего за пару кликов мышки. Все это не требует каких-либо знаний и навыков и справиться с подобными задачами может любой пользователь, владеющий компьютером. Также автоматически создаются резервные копии сервера (или за дополнительную плату или бэкапы уже входят в тариф) и все, что нужно вам – это настроить их периодичность.

Также у вас не будет никаких волнений относительно выхода из строя компонентов сервера – если даже что-то и выйдет из строя, то, скорее всего, об этом вы не узнаете. Во-первых, это проблемы провайдера, во-вторых, современные системы обеспечения отказоустойчивости, применяемые в дата-

центрах, позволяют минимизировать возможные простои в работе серверов в случае выхода «железа» из строя.

Отдельного внимания заслуживает возможность создания снапшотов – моментальных снимков сервера. Например, вы настраиваете сервер и перед настройкой вы делаете снапшот. Если что-то пойдет не так, за считанные минуты вы сможете вернуть все, как было. Да, хранение снапшотов – это платная функция, но она того стоит.

Клонирование сервера – это еще одна функция, которая достанется вам в виде приятного бонуса, если вы арендуете виртуальный сервер. Обычным пользователям она редко нужна, но пригодится интеграторам, которые продают типичные серверы своим клиентам. Все, что вам нужно – сначала создать эталонный сервер, а затем клонировать его для каждого клиента. Максимум, что понадобится – изменить некоторые настройки, ориентированные на клиента. Но это гораздо быстрее, чем создавать подобный сервер с нуля.

Арендовать виртуальный сервер в большинстве случаев выгоднее, проще и удобнее, чем связываться с физическим сервером. Сегодня, по сути, физическое оборудование имеет смысл приобретать, если вы сами планируете предоставлять виртуальные серверы в аренду. Во всех остальных случаях можно обойтись виртуальным сервером.

21.5. Виртуализация физического сервера

Любой физический Linux-сервер можно относительно легко подвергнуть виртуализации. Первым делом нужно снять образ диска для VMware с физического Linux-сервера. Никакое дополнительное программное обеспечение для начала виртуализации нам не нужно, поскольку образ можно создать стандартной командой dd:

```
dd if=/dev/sda of=/mnt/temp/sda.img bs=8M conv=sync,noerror
```

Здесь мы параметром if задаем входящий файл — весь диск /dev/sda. В качестве выходного файла (of) мы устанавливаем /mnt/share/sda.img — это и будет наш образ диска. Параметр bs задает размер блока для ускорения про-

цедуры копирования (по умолчанию 512 байт, следовательно, данные будут сбрасываться небольшими блоками по 512 байт, что существенно замедлит процедуру создания образа). Последний параметр указывает на необходимость копирования с игнорированием ошибок и создания точной (бит-в-бит) копии физического диска.

Примечание. В точке монтирования /mnt/temp должно быть достаточно места. Как минимум, должно быть свободно столько, сколько займет копируемый физический диск после виртуализации.

После создания образа диска его нужно преобразовать в образ, подходящий для гипервизора вашего облачного провайдера. Один из наиболее часто используемых гипервизоров – VMWare. Далее будет показано, как преобразовать созданный с помощью команды dd образ в формат VMWare. Для этого мы будем использовать qemu, точнее утилиту qemu-img, входящую в состав этого пакета эмуляции. Для проведения виртуализации физического сервера придется установить весь пакет:

```
sudo apt install qemu
```

Команда конвертирования выглядит так:

```
qemu-img convert -o compat6 /mnt/temp/sda.img -O vmdk /mnt/
share/vmware.vmdk
```

Обратите внимание, что в /mnt/temp должно быть достаточно места для хранения файла vmware.vmdk, который также будет большого размера. Созданный vmdk-файл уже можно использовать. Например, вы можете указать его при создании виртуальной машины в VMWare Workstation в качестве образа жесткого диска и сможете загрузиться с него.

21.6. Формат OVF

Не все облачные провайдеры используют гипервизор VMware. Второй часто используемый формат для виртуальных машин – OVF. Для конверти-

Рис. 21.3. Создание новой виртуальной машины в VMWare Workstation

рования в формат ovf нам понадобится программа VMWare Workstation, которую можно бесплатно скачать после регистрации на сайте компании. Подойдет Windows-версия – это избавит вас от установки графического интерфейса на сервере. Установите приложение и создайте новую виртуальную машину (рис. 21.3).

Выберите, что вы установите операционную систему позже (рис. 21.4).

Выберите тип операционной системы – Linux и выберите ваш дистрибутив (либо его ближайшего «родственника»), см. рис. 21.5.

Введите название виртуальной машины и путь к ней (рис. 21.6).

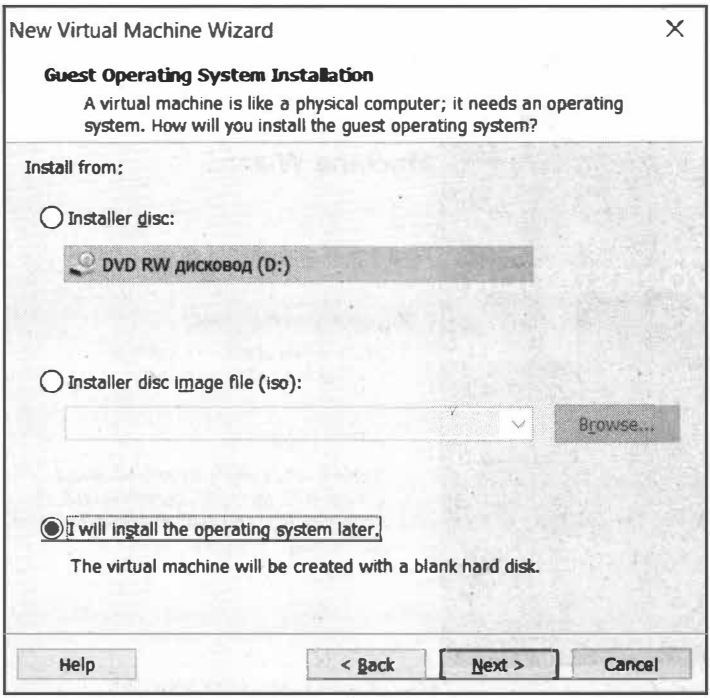


Рис. 21.4. Установим операционную систему позже

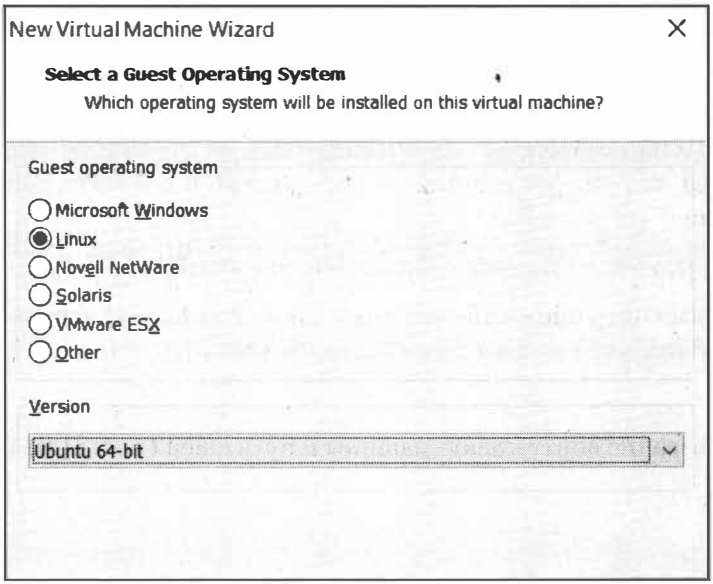


Рис. 21.5. Выбор типа операционной системы

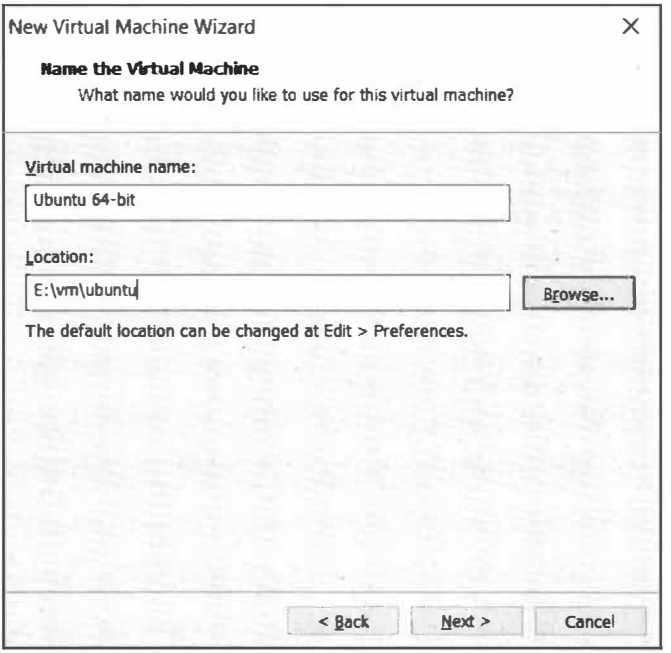


Рис. 21.6. Имя и путь к виртуальной машине

Следующий этап – это установка размера виртуального диска. Поскольку созданный диск нами будет удален (ведь мы будем подключать к виртуальной машине ранее конвертированные диски), то можно создать диск минимального размера.

После создания виртуальной машины выберите команду **Edit virtual machine settings** (рис. 21.8).

В окне **Virtual Machine Settings** (рис. 21.9) нажмите кнопку **Add**.

Нужно добавить жесткий диск. Выберите **Hard Disk** (рис. 21.10), тип интерфейса SCSI (рис. 21.11), а вот когда мастер дойдет до выбора диска, то нужно выбрать **Use an existing virtual disk** (использовать существующий жесткий диск), рис. 21.12.

Рис. 21.7. Установка размера виртуального диска

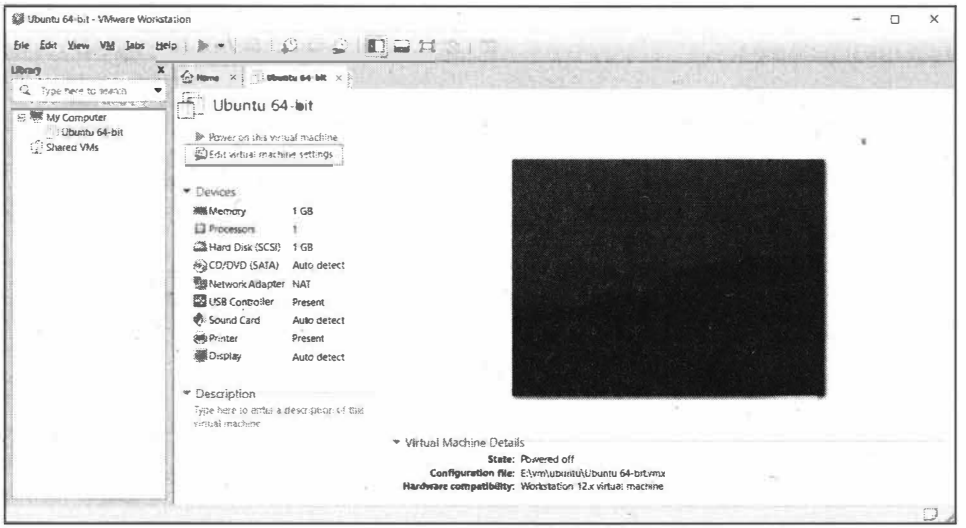


Рис. 21.8. Виртуальная машина создана

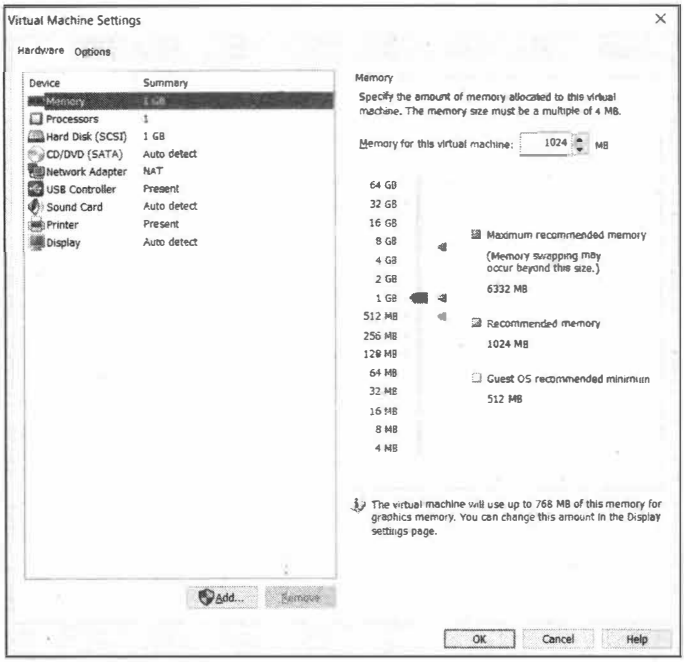


Рис. 21.9. Параметры виртуальной машины

Рис. 21.10. Выбор добавляемого «железа»

Рис. 21.11. Выбор типа интерфейса диска

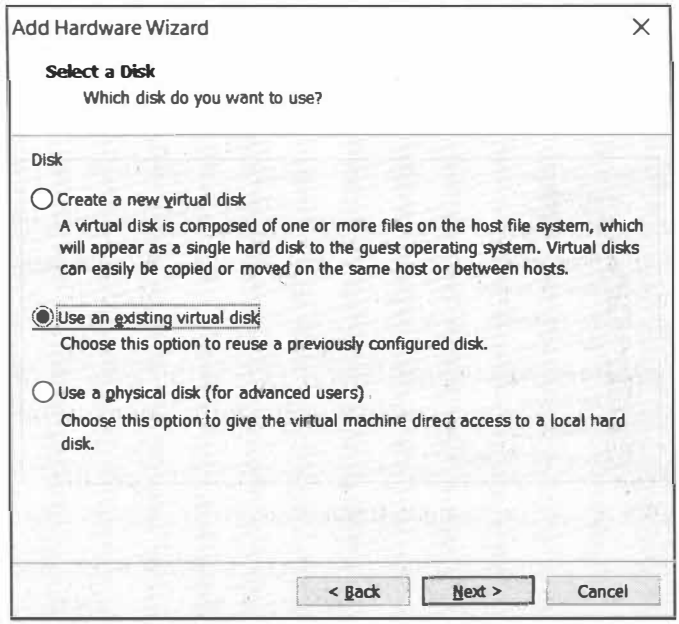


Рис. 21.12. Выберите Use an existing virtual disk

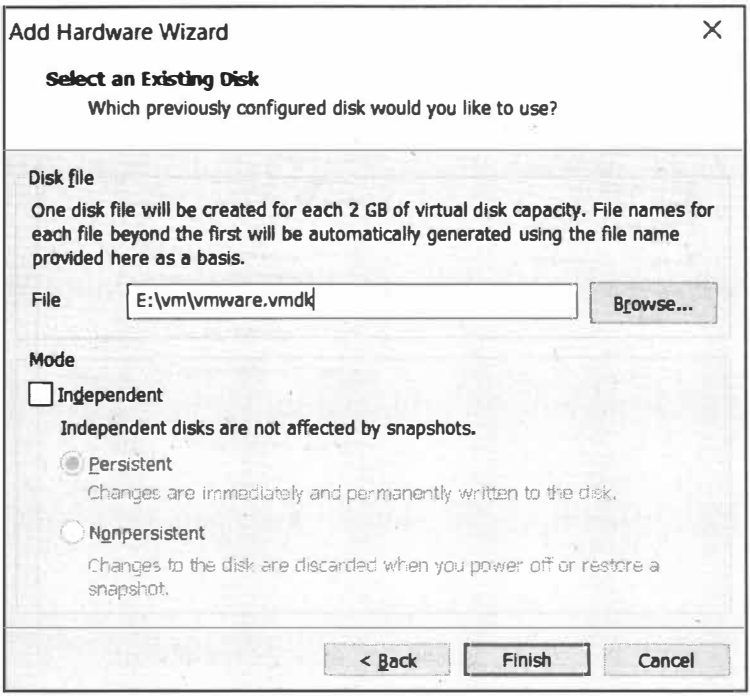


Рис. 21.13. Указываем путь к vmdk-файлу

Укажите путь к файлу, который был получен в результате конвертации (vmware.vmdk), см. рис. 21.13.

Теперь нам осталось удалить первый диск, который был создан при создании виртуальной машины. Выделите его в списке оборудования и нажмите кнопку **Remove** (рис. 12). Закройте окно **Virtual Machine Settings**, нажав кнопку **OK**.

Включите виртуальную машину, чтобы проверить, что все корректно. Если все хорошо, тогда выберите команду меню **File, Export to OVF**. В появившемся окне (рис. 21.15) выберите, куда нужно сохранить OVF-файл. Будет запущен процесс конвертации. Дождитесь его завершения, после чего вы получите OVF-файл виртуальной машины.

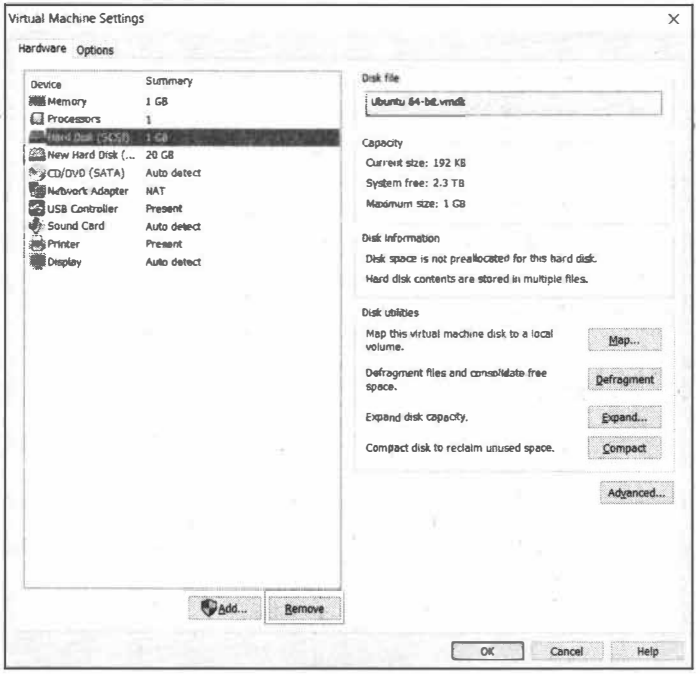
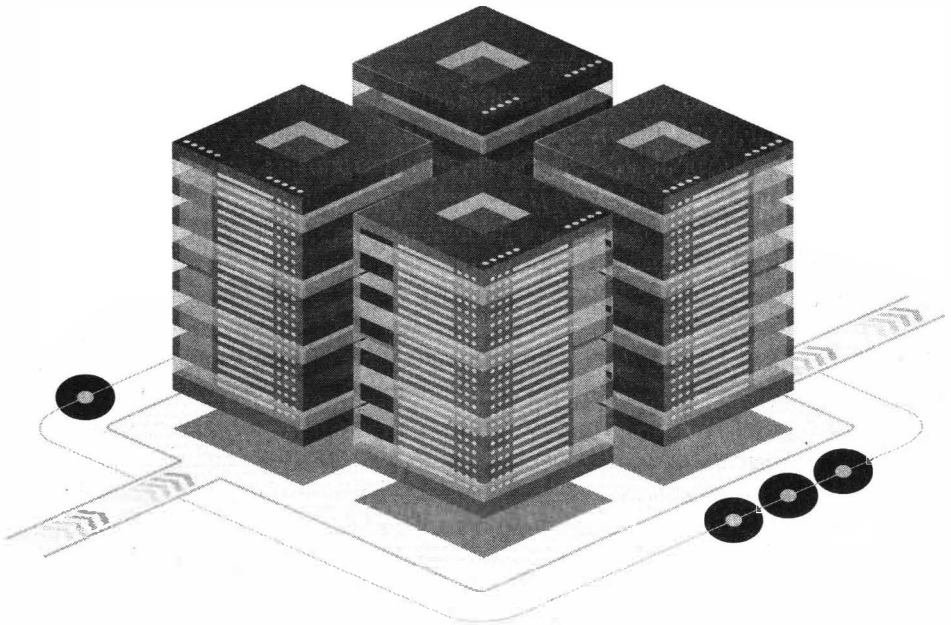


Рис. 21.14. Удаление первого диска

Рис. 21.15. Экспорт в OVF

Глава 22.

BigBlueButton — свой Zoom своими руками



В последнее время возникла необходимость в проведении веб-конференций для обучения или же просто для проведения бизнес-встреч. Это удобно (каждый участник конференции может находиться в любой точке мира), безопасно (не нужно куда-то ехать/лететь и подвергать себя опасности) и дешево (авиа-билеты и гостиница могут стоить достаточно дорого). Существуют различные платформы для проведения конференций. Как правило, все эти платформы – платные. Существует возможность бесплатного использования, но с теми или иными ограничениями, например, ограничение на количество участников, ограничение на время конференции и т.д. В этой главе мы рассмотрим систему BigBlueButton, которую вы можете использовать для организации собственных конференций (для нужд своей компании) и при этом все совершенно бесплатно. Все, что нужно – это относительно мощный компьютер, работающий под управлением Ubuntu и навыки работы с Linux, которые вы уже успели получить, читая эту книгу.

22.1. Назначение платформы

BigBlueButton (далее BBB) — открытое программное обеспечение для проведения веб-конференции. Система разработана в первую очередь для дистанционного обучения. При использовании BBB вы получаете основные функции, которые вы ожидаете от коммерческой системы веб-конференций (но по лицензии с открытым исходным кодом). Эти функции включают в себя обмен аудио, видео, презентациями и экранами в режиме реального времени, а также инструменты для совместной работы, такие как доска, общие заметки, опрос и комнаты обсуждения. BigBlueButton может записывать ваши сеансы для последующего воспроизведения.

Название BigBlueButton происходит от первоначальной концепции, что, начало веб-конференции должно быть максимально простым, как нажатие метафорической большой синей кнопки.

Основные возможности платформы:

- Поддерживает наличие нескольких аудиодорожек и обмен видео, возможность показа презентаций, документов Microsoft Office и OpenOffice, изображений, PDF документов.

- Поддерживаются расширенные возможности доски — такие, как указатель, масштабирование и рисование, доступ к рабочему столу. Для обратной связи со слушателями веб-конференции существуют публичные и приватные чаты.

- Интегрирована VoIP на базе FreeSWITCH.

- Пользователь может войти в конференцию либо как зритель либо как модератор. В режиме зрителя пользователь может присоединиться к голосовой конференции, использовать web-камеру, поднять руку (попросить слово) и общаться с другими людьми. В режиме модератора пользователь имеет возможность отключить / включить микрофон любого зрителя, удалить любого зрителя из веб-конференции, а также передать слово любому зрителю для выступления (сделать любого пользователя ведущим). Ведущий может загружать презентации, документы, использовать доску.

Примечание. FreeSWITCH — открытая телефонная платформа, распространяемая в исходных кодах, созданная для удовлетворения потребности в управляемых голосом или текстом системах, масштабируемых от софтфона до софтсвича.

22.2. История создания

Проект – не новый и за время своего существования он уже успел заслуженно получить признание многочисленной аудитории.

Изначально проект зародился в 2007 году в Карлтонском университете, внедряющем программы инновационных технологий. Первая версия была написана Ричардом Аламом и проект тогда назывался Blindside. В 2009 году Ричард Аллам и вместе с другими разработчиками загрузили код

BigBlueButton (уже новое название, полученное в 2009 году и неизменное до сих пор) на хостинг проектов Google Code. С тех порт компания проводит традиционную бизнес-модель открытых источников, предоставляя платную поддержку и услуг для сообщества BigBlueButton.

В 2019 году основные разработчики добавили доску для аннотирования загруженных презентации. Джереми Томерсон добавил интерфейс прикладного программирования (API), который сообщество BigBlueButton впоследствии использует для интеграции с различными CMS: Wordpress, Moodle, Drupal, LAMS, TikiWiki CMS Groupware и др.

Google принял BigBlueButton в рамках программы 2010 Google Summer of Code. С целью поощрения взносов от других пользователей, основные разработчики исходного кода переехали из Google Code на GitHub. Проект заявил о своем намерении создать независимый Фонд BigBlueButton «не для прибыли» для наблюдения за будущим развитием проекта.

В 2001 году была добавлена возможность записи и воспроизведения веб-конференции, а в 2012 году был опубликован сервер веб-конференций BigBlueButton 0.80. Начались работы над версией 0.81, которой должна появиться возможность записи и воспроизведения всех видов конференций в области презентации вместе со всеми веб-камерами.

Версия 1.0 официально была представлена в мае 2017 года, а на данный момент доступна версия 2.2, которая появилась 11 марта 2020 года. Проект постоянно развивается и нет ни одной причины, которая бы останавливала вас от экспериментов с этим проектом. Вы получите полноценную систему конференций бесплатно.

22.3. Системные требования

Для работы BBB необходимо выполнение следующих условий:

1. Ubuntu 16.04 или более новая с ядром 4.x (более старые ядра не подойдут)
2. 8 Гб оперативной памяти, рекомендуется 16 Гб
3. 4-ядерный процессор, рекомендуется 8-ядерный
4. Белый IP-адрес и доменное имя на этот адрес
5. Минимум 500 Гб на жестком диске для видеозаписей

6. Локаль en_US.UTF-8

Подобная конфигурация – не самая дорогая. По сути, любой современный компьютер подойдет под эти требования. Можно даже арендовать VPS, но виртуальный сервер нужной конфигурации обойдется довольно дорого от 3600 р. в месяц, учитывая большой объем накопителя. В этом случае особо нет смысла, поскольку бизнес-тариф для Zoom обойдется дешевле.

Если локаль отличается от en_US.UTF-8 (скорее всего, так оно и есть), перенастроить ее можно с помощью следующих команд:

```
sudo sed -i -e 's/# en_US.UTF-8 UTF-8/en_US.UTF-8 UTF-8/' /
etc/locale.gen
sudo sed -i -e 's/# ru_RU.UTF-8 UTF-8/ru_RU.UTF-8 UTF-8/' /
etc/locale.gen
sudo dpkg-reconfigure --frontend=noninteractive locales
sudo update-locale LANG=en_US.UTF-8
```

Убедитесь, что локаль указана в переменных окружения:

```
sudo systemctl show-environment
```

Вывод должен быть таким:

```
LANG=en_US.UTF-8
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/
sbin:/bin
```

22.4. Установка и настройка системы веб-конференций

Будем считать, что сеть уже настроена и у нас есть доступ к Интернету. Проверим, так ли это, пропинговав сервер Google DNS:

```
$ ping -c 3 8.8.8.8
```

Обновим список пакетов программ, чтобы в процессе установки не было ошибок:

```
$ sudo apt update
```

Установим пакет iptables-persistent для сохранения правил iptables:

```
$ sudo apt-get install iptables-persistent
```

Подготовим правила iptables:

```
$ sudo iptables -F
$ sudo iptables -A INPUT -i lo -j ACCEPT
$ sudo iptables -A INPUT -m state --state ESTABLISHED,RELATED
-j ACCEPT
$ sudo iptables -A INPUT -p icmp -j ACCEPT
$ sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT
$ sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT
$ sudo iptables -A INPUT -p tcp --dport 443 -j ACCEPT
$ sudo iptables -A INPUT -p tcp --dport 7443 -j ACCEPT
$ sudo iptables -A INPUT -p udp --dport 16384:32768 -j ACCEPT
$ sudo iptables -P INPUT DROP
```

Данные правила необходимы для нормальной работы веб-конференции BBB. Сохраним правила:

```
$ sudo /etc/init.d/netfilter-persistent save
```

Установим вспомогательное программное обеспечение:

```
$ sudo apt-get install apt-transport-https apt-transport-tor
ca-certificates curl gnupg-agent software-properties-common
```

Для BigBlueButton необходимы два приложения: ffmpeg (создание записей) и yq (обновление файлов YAML). Версия по умолчанию ffmpeg в Ubuntu 16.04 устарела и yq не существует в репозиториях по умолчанию.

Поэтому перед установкой BigBlueButton необходимо добавить следующие личные архивы пакетов (PPA) на сервер, чтобы убедиться, что установлены правильные версии.

```
$ sudo add-apt-repository ppa:bigbluebutton/support -y
$ sudo add-apt-repository ppa:rmescandon/yq -y
```

Затем обновляем сервер до последних пакетов.

```
$ sudo apt update
$ sudo apt full-upgrade
```

HTML5-клиент BigBlueButton использует MongoDB, устанавливаем его:

```
$ wget -qO - https://www.mongodb.org/static/pgp/server-
3.4.asc | sudo apt-key add -
$ echo "deb [ arch=amd64,arm64 ] http://repo.mongodb.org/
apt/ubuntu xenial/mongodb-org/3.4 multiverse" | sudo tee /
etc/apt/sources.list.d/mongodb-org-3.4.list
$ sudo apt-get update
$ sudo apt-get install -y mongodb-org curl
```

HTML5-клиент BigBlueButton требует наличия сервера nodejs, установим его:

```
$ curl -sL https://deb.nodesource.com/setup_8.x | sudo -E
bash -
$ sudo apt-get install -y nodejs
```

Скачиваем и добавляем ключ репозитория BigBlueButton. Поскольку Роскомнадзор закрыл доступ до репозитория, используем tor:

```
$ curl --socks5-hostname localhost:9050 -O https://ubuntu.
bigbluebutton.org/repo/bigbluebutton.asc
$ sudo apt-key add ./bigbluebutton.asc
```

Если у вас не получится скачать ASC-файл, он будет приложен к материалам для этой книги и вы сможете скачать его с сайта издательства. В книге листинг приводить не станем, поскольку очень сложно будет его перепечатать вручную и не допустить ошибок.

Добавляем репозиторий. указывая tor+https для того чтобы гарантировано добраться до пакетов:

```
$ echo "deb tor+https://ubuntu.bigbluebutton.org/xenial-220/
bigbluebutton-xenial main" | sudo tee /etc/apt/sources.
list.d/bigbluebutton.list
```

Устанавливаем bigbluebutton и bbb-html5:

```
$ sudo apt-get update
$ sudo apt-get install bigbluebutton
$ sudo apt-get install bbb-html5
```

Поскольку мы используем Tor, загружаться пакеты будут медленно, но ничего, мы это переживем.

> **Примечание.** Если вы используете Ubuntu 16.04, игнорируйте сообщение «Невозможность загрузить дополнительные файлы данных». Это известная ошибка 16.04 и она ни на что не влияет.

Теперь, когда все установлено, проверим, соответствует ли наша система требованиям BBB:

```
$ sudo bbb-conf -check
```

Любой последующий вывод после строчки ** Potential problems described below ** указывает на ошибки конфигурации или ошибки установки.

Указываем наш домен (замените домен своим доменом):

```
bbb-conf --setip bbb.example.com
```

Современные браузеры запрещают передачу видео и звука по открытым каналам, для нормальной работы нам нужен сертификат для работы https. проще всего его взять у Let's Encrypt. Хотя мы уже рассматривали, как это сделать, для полноты руководства рассмотрим еще раз в контексте настройки BBB.

Установим инструмент для работы с Let's Encrypt:

```
$ sudo add-apt-repository ppa:certbot/certbot
$ sudo apt-get install certbot
```

Генерируем набор данных:

```
$ sudo mkdir -p /etc/nginx/ssl
$ sudo openssl dhparam -out /etc/nginx/ssl/dhp-4096.pem 4096
```

На слабом компьютере выполнение данных команд может занять достаточно много времени.

Запрашиваем сертификат (укажите собственное доменное имя):

```
$ sudo certbot --webroot -w /var/www/bigbluebutton-default/
-d bigbluebutton.example.com certonly
```

В результате будут сгенерированы слеующие файлы:

```
$ ls /etc/letsencrypt/live/bigbluebutton.example.com/
cert.pem  chain.pem  fullchain.pem  privkey.pem
```

Теперь осталось отредактировать конфигурацию веб-сервера. Для упрощения этой главы будем считать, что мы используем nginx. Откройте файл nginx /etc/nginx/sites-available/bigbluebutton (это файл сайта для BBB) и добавьте в него следующие строки:

```
server {
  server_name bigbluebutton.example.com;
  listen 80;
  listen [::]:80;
  listen 443 ssl;
  listen [::]:443 ssl;
      ssl_certificate   /etc/letsencrypt/live/bigbluebutton.
example.com/fullchain.pem;
    ssl_certificate_key  /etc/letsencrypt/live/bigbluebutton.
example.com/privkey.pem;
  ssl_session_cache shared:SSL:10m;
  ssl_session_timeout 10m;
  ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
  ssl_ciphers "ECDH+AESGCM:DH+AESGCM:ECDH+AES256:DH+AES256:
ECDH+AES128:DH+AES:ECDH+3DES:DH+3DES:RSA+AESGCM:RSA+AES:RSA
+3DES:!aNULL:!MD5:!DSS:!AES256";
  ssl_prefer_server_ciphers on;
  ssl_dhparam /etc/nginx/ssl/dhp-4096.pem;
}
```

Перезапустите ngnix. Напоминаем, что сертификаты Let's Encrypte действительны в течение 90 дней и могут быть автоматически продлены. Чтобы автоматически запрашивать обновление раз в неделю, отредактируйте файл crontab для root:

```
$ sudo crontab -e
```

Добавьте в него две строки:

```
30 2 * * 1 /usr/bin/certbot renew >> /var/log/le-renew.log
35 2 * * 1 /bin/systemctl reload nginx
```

Откройте файл /etc/bigbluebutton/nginx/sip.nginx и измените протокол и порт в строке proxy_pass:

```
location /ws {
  proxy_pass https://203.0.113.1:7443;
  proxy_http_version 1.1;
```

```
  proxy_set_header Upgrade $http_upgrade;
  proxy_set_header Connection "Upgrade";
  proxy_read_timeout 6h;
  proxy_send_timeout 6h;
  client_body_timeout 6h;
  send_timeout 6h;
}
```

Отредактируйте /usr/share/bbb-web/WEB-INF/classes/bigbluebutton.properties и обновите свойство bigbluebutton.web.serverURLдля использования HTTPS:

```
#-----------------------------------------------------
# This URL is where the BBB client is accessible. When a user
successfully
# enters a name and password, she is redirected here to load
the client.
bigbluebutton.web.serverURL=https://bigbluebutton.example.
com
```

Отредактируйте файл /usr/share/red5/webapps/screenshare/WEB-INF/screenshare.properties и обновите свойство jnlpUrl и jnlpFile - укажите HTTPS (везде по мере редактирования конфигурации указывайте точное имя вашего сервера):

```
streamBaseUrl=rtmp://bigbluebutton.example.com/screenshare
jnlpUrl=https://bigbluebutton.example.com/screenshare
jnlpFile=https://bigbluebutton.example.com/screenshare/
screenshare.jnlp
```

Также нужно обновить файл /var/www/bigbluebutton/client/conf/config.xml чтобы сообщить клиенту BigBlueButton о загрузке компонентов через HTTPS. Вы можете сделать такое обновление с помощью одной команды:

```
$ sudo sed -e 's|http://|https://|g' -i /var/www/
bigbluebutton/client/conf/config.xml
```

Откройте файл /usr/share/meteor/bundle/programs/server/assets/app/config/settings.yml и найдите в нем следующий код:

```
kurento:
    wsUrl: ws://bbb.example.com/bbb-webrtc-sfu
```

Замените на:

```
kurento:
    wsUrl: wss://bigbluebutton.example.com/bbb-webrtc-sfu
```

Также найдите код:

```
note:
    enabled: true
    url: http://bbb.example.com/pad
```

Замените его на:

```
note:
    enabled: true
    url: https:// bigbluebutton.example.com/pad
```

Редактируем /usr/local/bigbluebutton/core/scripts/bigbluebutton.yml для работы видеозаписей через https:

```
playback_protocol: https
```

Поскольку флеш уже устарел, включим использование html5 по умолчанию. Отредактируйте /usr/share/bbb-web/WEB-INF/classes/bigbluebutton.properties и замените false на true в строчках

```
# Force all attendees to join the meeting using the HTML5
client
attendeesJoinViaHTML5Client=true
# Force all moderators to join the meeting using the HTML5
client
moderatorsJoinViaHTML5Client=true
```

Перезапускаем все компоненты bigbluebutton

```
$ sudo bbb-conf -restart
```

Дальше нам понадобится установить панель управления greenlight. Установим docker и docker-compose

```
$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg |
sudo apt-key add -
$ sudo add-apt-repository "deb [arch=amd64] https://
download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
$ sudo apt update
$ sudo apt install docker-ce docker-ce-cli containerd.io
$ sudo curl -L "https://github.com/docker/compose/releases/
download/1.25.4/docker-compose-$(uname -s)-$(uname -m)" -o
/usr/local/bin/docker-compose
$ sudo chmod +x /usr/local/bin/docker-compose
$ sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-
compose
```

Примечание. Greenlight - приложение на Ruby on Rails, предоставляющее простой интерфейс для пользователей, чтобы создавать комнаты, начинать конференции, управлять записями конференций.

Создайте каталог Greenlight для его конфигурации

```
mkdir ~/greenlight && cd ~/greenlight
```

Сгенерируйте образ Greenlight

```
docker run --rm bigbluebutton/greenlight:v2 cat ./sample.env
> .env
```

docker нужен секретный ключ для запуска, генерируем его:

```
docker run --rm bigbluebutton/greenlight:v2 bundle exec rake
secret
```

Редактируем .env файл. Установите SECRET_KEY_BASE опцию на этот ключ.

Просмотрим секретный ключ bigbluebutton:

```
$ sudo bbb-conf -secret
```

Отредактируем .env файл, установите BIGBLUEBUTTON_ENDPOINT URL-адрес и установите BIGBLUEBUTTON_SECRET секретный ключ.

Проверяем настройку

```
docker run --rm --env-file .env bigbluebutton/greenlight:v2
bundle exec rake conf:check
```

Добавьте виртуальный каталог в nginx, Конфигурация Nginx для этого подкаталога хранится в образе Greenlight.

Чтобы добавить этот файл конфигурации на ваш сервер BigBlueButton, запустите:

```
docker run --rm bigbluebutton/greenlight:v2 cat ./greenlight.
nginx | sudo tee /etc/bigbluebutton/nginx/greenlight.nginx
```

Поскольку на корневой странице ничего не будет, настроим перенаправление на Greenlight. Для этого добавим следующую запись в нижней части / etc/nginx/sites-available/bigbluebutton перед последним } символом.

```
location = / {
  return 307 /b;
}
```

Выполните команду:

```
docker run --rm bigbluebutton/greenlight:v2 cat ./docker-
compose.yml > docker-compose.yml
```

Сгенерируем случайный пароль для базы данных:

```
export pass=$(openssl rand -hex 8); sed -i 's/POSTGRES_
PASSWORD=password/POSTGRES_PASSWORD='$pass'/g' docker-compose.
yml;sed -i 's/DB_PASSWORD=password/DB_PASSWORD='$pass'/g' .env
```

Запустим Greenlight:

```
docker-compose up -d
```

Осталось совсем немного. Нужно создать учетную запись администратора и перезагрузить систему. Создаем учетную запись администратора:

```
docker exec greenlight-v2 bundle exec rake user:create["me","me@
mail","Password","admin"]
```

Перезагружаем систему:

```
sudo shutdown -r now
```

На рис. 22.1 показана панель управления BBB (для получения доступа к ней перейдите по адресу, сконфигурированному для BBB, в нашем случае это bigbluebutton.example.com). По сути, BBB готова к использованию.

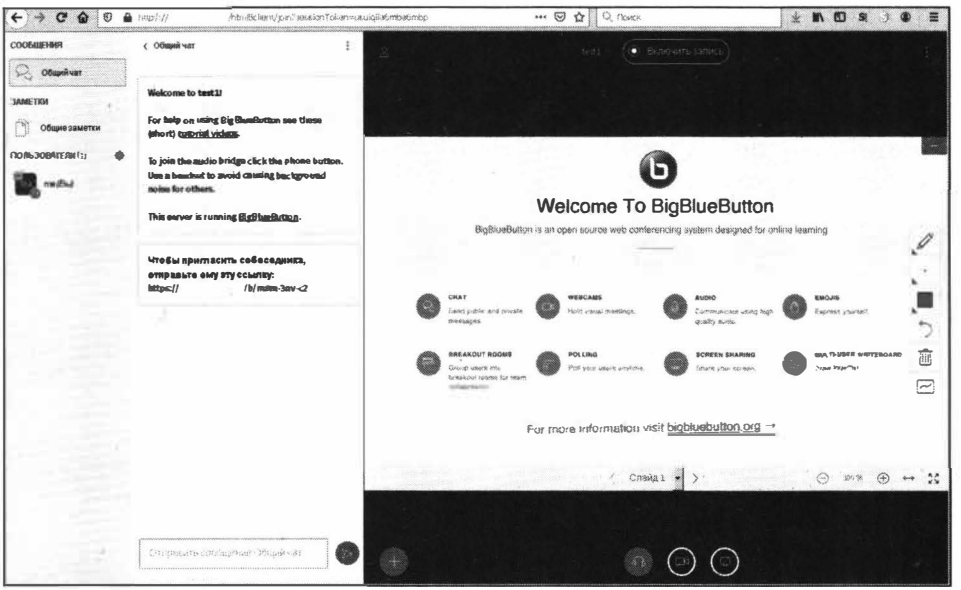


Рис. 22.1. bigbluebutton - бесплатные видеоконференции

На docs.bigbluebutton.com есть установочный скрипт, который делает, по сути, все вышеизложенное, но он работать не будет, поскольку он не устанавливает Greenlight, HTTPS. не настраивает HTML5, а также может не работать из-за блокировок РКН. Именно поэтому процедура установки получилось такой сложной, намного сложнее, чем зарубежных руководствах. Но зато вы получите уже рабочую систему и не будете разбираться с различными подводными камнями, на которые вы бы обязательно наткнулись при использовании стандартного скрипта.

Левицкий Никита Денисович

Удаленный сервер

СВОИМИ РУКАМИ

От азов создания до практической работы

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 30.11.2020. Формат 70x100 1/16.

Бумага газетная. Печать офсетная. Объем 25 п. л.

Тираж 1300. Заказ №10783.

Отпечатано с готовых файлов заказчика
в АО «Первая Образцовая типография»,
филиал «УЛЬЯНОВСКИЙ ДОМ ПЕЧАТИ»
432980, Россия, г. Ульяновск, ул. Гончарова, 14

УДАЛЕННЫЙ СЕРВЕР СВОМИ РУКАМИ

От азов создания до практической работы

Эта книга поможет вам самостоятельно освоить полный цикл создания своего собственного выделенного сервера: от установки операционной системы (для удобства это будет Linux) на сервер до настройки и администрирования. Будет подробно рассказано: как выполнить первоначальную настройку сервера; как использовать командную строку; вы узнаете как настраивать сетевые интерфейсы сервера; поговорим о маршрутизации и настройке брандмауэра; как пользоваться удаленным входом в систему по ssh; как настаивать файловый сервер FTP; что такое DHCP-сервер и как подключаться к Windows-инфраструктуре; как повысить производительность сервера и многое другое.

Отдельно внимание будет уделено вопросам обеспечения безопасности вашего сервера: локальная безопасность, защита от сетевых атак, шифрование данных, защита сетевых служб и т.д.

Также будет рассмотрена настройка веб-сервера и его администрирование: подключение ssl-сертификата, выбор и установка панели управления сервером; настройка почты на веб-сервере и т.п. Отдельно будет рассказано как создавать свой zoom — сервер видеоконференций!

Книга будет полезна как опытным, так и начинающим администраторам, поскольку изложение будет вестись от самых азов до более сложных тем, и каждый сможет найти то, что ему нужно!

ISBN 978-5-94387-568-7

9 78- 5- 94387- 568- 7

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