КРИСТОФЕР НЕГУС

WILEY

Linux®

БИБЛИЯ

ИСЧЕРПЫВАЮЩЕЕ РУКОВОДСТВО

ВСЕ ИНСТРУМЕНТЫ | ЛУЧШИЕ ПРАКТИКИ | ВСЕГДА ПОД РУКОЙ

10 ИЗДАНИЕ

Linux®
BIBLE

Tenth Edition

Christopher Negus

WILEY

КРИСТОФЕР НЕГУС

Linux®

БИБЛИЯ

ИСЧЕРПЫВАЮЩЕЕ РУКОВОДСТВО

ПИТЕР®

Санкт-Петербург • Москва • Минск

2022

ББК 32.973.2-018.2
УДК 004.43
Н41

Негус Кристофер

Н41 Библия Linux. 10-е издание. — СПб.: Питер, 2022. — 928 с.: ил. — (Серия «Для профессионалов»).

ISBN 978-5-4461-1797-0

Полностью обновленное 10-е издание «Библии Linux» поможет как начинающим, так и опытным пользователям приобрести знания и навыки, которые выведут на новый уровень владения Linux. Известный эксперт и автор бестселлеров Кристофер Негус делает акцент на инструментах командной строки и новейших версиях Red Hat Enterprise Linux, Fedora и Ubuntu. Шаг за шагом на подробных примерах и упражнениях вы досконально поймете операционную систему Linux и пустите знания в дело. Кроме того, в 10-м издании содержатся материалы для подготовки к экзаменам на различные сертификаты по Linux.

16+ (В соответствии с Федеральным законом от 29 декабря 2010 г. № 436-ФЗ.)

ББК 32.973.2-018.2
УДК 004.43

ISBN 978-1119578888 англ.
ISBN 978-5-4461-1797-0

Краткое содержание

Часть V. Методы обеспечения безопасности в Linux

Часть VI. Работа с облачными вычислениями

Приложения

Оглавление

Часть II. Опытный пользователь Linux

Часть III. Администрирование системы в Linux

Часть IV. Администрирование серверов в Linux

Часть V. Методы обеспечения безопасности в Linux

Часть VI. Работа с облачными вычислениями

Приложения

Как и всегда, посвящаю эту книгу своей жене Шери.

Об авторе

Кристофер Негус — главный специалист по разработке и написанию технической документации в компании Red Hat, Inc. За более чем десять лет работы в Red Hat Крис помог сотням ИТ-специалистов стать сертифицированными инженерами Red Hat (RHCEs) и написал множество технических руководств на различные темы, от Linux до виртуализации, облачных вычислений и контейнеризации.

До прихода в компанию Red Hat Крис написал десятки книг по Linux и UNIX, включая *Red Hat Linux Bible* (все издания), *Docker Containers*, *CentOS Bible*, *Fedora Bible*, *Linux Troubleshooting Bible*, *Linux Toys*, *Linux Toys II* и девять изданий «Библии Linux». Он также был соавтором нескольких книг из серии Linux Toolbox для опытных пользователей: *Fedora Linux Toolbox*, *SUSE Linux Toolbox*, *Ubuntu Linux Toolbox*, *Mac OS X Toolbox* и *BSD UNIX Toolbox*.

Еще до начала писательской деятельности Крис в течение восьми лет работал в организации AT&T, разработавшей операционную систему UNIX, а в начале 1990-х годов переехал в Юту, чтобы внести свой вклад в проект Novell UnixWare.

С Кристофером Негусом можно связаться по электронной почте striker57@gmail.com.

О научных редакторах

Джейсон У. Эккерт — опытный преподаватель, консультант и автор бестселлеров в области информационных технологий (ИТ). В его арсенале 45 аттестаций, более 30 лет опыта работы в ИТ, четыре опубликованных приложения и 24 изданных учебника, охватывающих такие темы, как системы Unix и Linux, обеспечение безопасности, сервер Windows, сервер Exchange, сервер BlackBerry Enterprise, PowerShell и разработка видеоигр. Эккерт делится своим опытом, преподавая в колледже triOS College, он декан технологического факультета. Больше информации о Джейсоне Эккерте можно найти на сайте jasoneckert.net.

Деррик Орнелас — старший инженер по обслуживанию программного обеспечения в компании Red Hat, Inc. В своей нынешней роли ведущего разработчика продукта Red Hat Container technologies, включая OpenShift Container Platform и Red Hat Enterprise Linux CoreOS, Деррик работает над обеспечением как поддержки, так и качества продуктов Red Hat. Ранее он работал старшим руководителем в отделе технической поддержки технологий виртуализации, таких как libvirt, KVM и Red Hat Virtualization.

За 12 лет работы в Red Hat Деррик получил сертификаты Red Hat Certified Engineer и Red Hat Certified Virtualization Administrator и применил свои обширные знания в области Linux для проектирования, развертывания и обслуживания различных аппаратных лабораторий и приложений.

Почти 20 лет назад работа с Linux для него началась с получения степени бакалавра компьютерных наук в Государственном университете города Аппалачи (США). Будучи преданным поклонником Linux, он любит обучать новых пользователей этой системы.

Благодарности

Когда десять лет назад меня приняли в Red Hat, я не знал, что компания вырастет примерно в два раза, будет приобретена компанией IBM за 34 млрд долларов, при этом сохранив дух открытости и энтузиазма, который я отметил, впервые подписав контракт. Каждый день, приходя на работу, я общаюсь со многими величайшими в мире разработчиками Linux и облачных технологий, тестировщиками, консультантами и специалистами поддержки.

Хотя я не могу поблагодарить каждого в отдельности, я хотел бы отметить дух сотрудничества в Red Hat, который помогает мне каждый день улучшать собственные навыки работы в Linux. Я благодарю компанию Red Hat не потому, что работаю там, а потому, что она соответствует моим идеалам компании по разработке программного обеспечения с открытым исходным кодом.

Тем не менее есть несколько сотрудников, которых я хочу особо отметить. В Red Hat я могу взять на себя так много крутых и сложных проектов благодаря свободе, которую дают мне Мишель Беарер, Дон Эйснер и Сэм Кнут. Сэм, в частности, поддерживал меня и поощрял мою работу на протяжении более чем десяти лет.

Хочу также поблагодарить Скотта Маккарти, Бена Брирду, Лори Фридмана, Дейва Дарре, Мика Эбботту, Стива Милнера и Иэна Маклеода (разработчики инструментов контейнеризации, RHCOS и OpenShift), а также Тома Маккея, Джоуи Шорра, Билла Деттелбека, Риче Марваха и Дирка Херрманна (команда Quay). Наконец, особая благодарность Викраму Гоялу, который, к счастью, живет в Австралии, так что всегда готов выручить меня, когда я что-то ломаю посреди ночи.

Что касается работы над книгой, мне помогали два превосходных научных редактора: Джейсон Эккерт и Деррик Орнелас. Я не знал Джейсона до того, как он взял на себя эту роль, но его огромный опыт работы с различными системами Linux помог мне выйти за пределы работы исключительно с Red Hat. Деррика, которого я вижу почти каждый день, попросили поучаствовать в редактировании книги из-за того, что он очень внимателен к деталям и хорошо понимает, как работает система Linux и что нужно знать, чтобы ее использовать. Благодаря проделанной Джейсоном и Дерриком работе любой, кто прочтет книгу, получит бесценный опыт.

Благодарю сотрудников издательства Wiley за то, что позволили мне улучшать эту книгу на протяжении многих лет. Спасибо Гэри Шварцу, который постоянно

и мягко напоминал мне, чтобы я продолжал работать над книгой, в те моменты, когда у меня не было на нее свободного времени. Когда давления Гэри оказывается недостаточно, вмешивается Девон Льюис, чтобы яснее обрисовать картину важности соблюдения сроков. Спасибо также Марго Мэйли Хатчисон из Waterside Productions за то, что она заключила для меня контракт с издательством Wiley и всегда заботилась о моих интересах.

Наконец, спасибо моей жене Шери за то, что она разделила со мной свою жизнь и так замечательно воспитала наших сыновей Сета и Калеба.

Кристофер Негус

Введение

Нельзя изучить систему Linux, не начав в ней работать.

Я пришел к такому выводу, более 20 лет обучая людей работе в Linux. Недостаточно просто прочитать книгу или прослушать лекцию. Вам нужен кто-то, кто будет направлять вас, и необходимо попробовать все сделать самому.

В 1999 году я написал свою первую книгу по Linux — *Red Hat Linux Bible*. Ее громный успех дал мне возможность стать полноценным независимым автором книг о Linux. В течение примерно десяти лет я написал десятки книг и исследовал лучшие способы изучения этой системы в тишине маленького домашнего офиса.

В 2008 году я устроился в компанию Red Hat, Inc. в качестве штатного консультанта, обучающего профессиональных системных администраторов Linux с сертификатом Red Hat Certified Engineer (RHCE). За три года работы там я оттачивал свои навыки преподавания перед живой аудиторией, чей опыт использования системы варьировался от нулевого до уровня опытного профессионала. Со временем я смог расширить собственные знания о Linux, получив более десяти сертификатов, включая сертификат Red Hat Certified Architect (RHCA).

В первом издании этой книги я изложил весь свой опыт преподавания, чтобы позволить новичкам в Linux с базовыми навыками вырасти до профессионалов. Навыки, которые позволяло получить предыдущее издание, можно получить и по прочтении нынешнего.

- **От новичка до сертифицированного профессионала.** Если вы пользуетесь компьютером, мышью и клавиатурой, можете начать с этой книги. Я расскажу, как загрузить систему Linux, начать применять ее, изучить важные темы и в конечном итоге научиться администрировать и защищать ее.
- **Системное администрирование.** Прочитав эту книгу, вы будете знать, как использовать, модифицировать и поддерживать систему Linux. Почти все темы, необходимые для того, чтобы стать сертифицированным инженером Red Hat, в ней представлены. Многие разработчики программного обеспечения также изучали эту книгу, чтобы понять, как работать с системой Linux в качестве платформы разработки или цели их приложений.
- **Акцент на инструментах командной строки.** Несмотря на то что в последние годы интерфейсы управления Linux значительно улучшились, многие расширенные функции можно применить только путем ввода команд и редактиро-

вания файлов конфигурации вручную. Я научу вас профессионально работать с командной строкой Linux, сравнивая при этом функции оболочки с графическими инструментами для выполнения одних и тех же задач.

- **Нацеленность на меньшее количество дистрибутивов Linux.** В предыдущих изданиях я описал около 18 различных дистрибутивов Linux. За несколькими заметными исключениями, большинство популярных дистрибутивов Linux основаны либо на Red Hat (Red Hat Enterprise Linux, Fedora, CentOS и т. д.), либо на Debian (Ubuntu, Linux Mint, KNOPPIX и т. д.). Эта книга наиболее подробно описывает дистрибутивы Red Hat, но я добавил и информацию о дистрибутиве Ubuntu, потому что именно его используют самые большие поклонники Linux.
- **Большое количество примеров и упражнений.** Вместо того чтобы просто рассказывать вам, как работает Linux, я показываю это на примерах. Затем, чтобы усвоить знания, можете выполнить упражнения самостоятельно. Каждая инструкция и каждое упражнение протестированы в системах Fedora или Red Hat Enterprise Linux. Большинство из них работают и в системе Ubuntu.

Основные усовершенствования десятого издания предусматривают акцент на упрощенном администрировании Linux, автоматизации задач и управлении контейнерными приложениями (индивидуально или в масштабе организации).

- **Веб-интерфейс администрирования Cockpit.** С момента создания Linux пользователи пытались разработать простые графические или браузерные интерфейсы для управления системами Linux. Я считаю, что Cockpit — это лучший веб-интерфейс, когда-либо созданный для управления большинством основных функций Linux. Во всей книге я заменил большинство описаний инструментов `system-config*` описаниями Cockpit. С помощью Cockpit вы можете добавлять пользователей, управлять хранилищем, отслеживать действия и выполнять многие другие административные задачи через единый интерфейс.
- **Переход к облачным технологиям.** Введя понятие облачных технологий в предыдущем издании, здесь я расширил его. Сюда относятся настройка собственного хоста Linux для запуска виртуальных машин и запуск Linux в облачной среде, такой как Amazon Web Services. Linux сегодня находится в центре большинства технологических достижений в области облачных вычислений. Это значит, что вам нужно глубоко понимать систему, чтобы эффективно работать в центрах обработки данных будущего. В начале этой книги я помогу изучить основы Linux. А в последних главах продемонстрирую, как настроить системы Linux в качестве гипервизоров, облачных контроллеров и виртуальных машин, а также управлять виртуальными сетями и сетевым хранилищем.
- **Ansible.** Автоматизация задач управления системами становится все более актуальной в современных центрах обработки данных. Используя Ansible, вы можете создавать каталоги, которые определяют состояние системы Linux. Они включают в себя настройку установленных пакетов, запущенных служб и компонентов. Такой каталог поможет настроить одну систему или 1000 систем, он может быть объединен в набор системных служб и запущен для того, чтобы

вернуть систему в определенное состояние. В этом издании я познакомлю вас с Ansible, помогу создать свой первый каталог Ansible и покажу, как запускать специальные команды Ansible.

- **Контейнеры.** Упаковка и запуск приложений в контейнерах становится предпочтительным методом развертывания небольших масштабируемых программных служб и функций, управления ими и их обновления. В книге я описываю, как добавлять контейнеры в систему, запускать и останавливать их и даже создавать собственные образы контейнеров с помощью команд `podman` и `docker`.
- **Kubernetes и OpenShift.** Хотя контейнеры хороши сами по себе, чтобы иметь возможность развертывать контейнеры, управлять ими и обновлять их на крупном предприятии, нужна платформа оркестрации. Проект Kubernetes предоставляет эту платформу. Для коммерческой поддерживаемой платформы Kubernetes можно применять OpenShift.

Структура книги

Книга позволяет изучить систему Linux с начального уровня и довести свои навыки до уровня опытного пользователя и системного администратора.

Часть I «Начало работы» содержит две главы, объясняющие, что такое Linux и как начать работу с системой.

- Глава 1 «Начало работы в Linux» описывает, что такое система Linux, как она появилась и как начать работу с ней.
- В главе 2 «Идеальный рабочий стол в Linux» рассказывается, как создать идеальный настольный компьютер с помощью популярных функций системы.

Часть II «Опытный пользователь Linux» разбирает в деталях функции командной оболочки Linux, файловой системы, управления текстовыми файлами и другими процессами, а также применение скриптов оболочки.

- Глава 3 «Использование оболочки» описывает, как установить командную оболочку, запускать и отменять команды (с помощью истории действий), настроить автозаполнение команд. Здесь также рассказывается, как работать с переменными, псевдонимами (ярлыками) и справочниками (традиционные в Linux справочные страницы команд).
- Глава 4 «Файловая система» включает в себя описание команд для перечисления, создания, копирования и перемещения файлов и каталогов. Здесь рассказывается о том, как обеспечить безопасность файловой системы, то есть о праве собственности, запретах и списках доступа.
- В главе 5 «Работа с текстовыми файлами» рассматривается все, что касается работы с текстом, — от базовых текстовых редакторов до поиска файлов и текста в самих файлах.

- Глава 6 «Управление активными процессами» рассказывает, как посмотреть, какие процессы запущены в системе и как их сменить (с помощью команд `kill`, `pause` и др.).
- В главе 7 «Простые скрипты оболочки» описаны команды и функции оболочки, которые можно объединить в файл-команду.

Часть III «Администрирование системы в Linux» рассказывает о том, как управлять системами Linux.

- Глава 8 «Системное администрирование» дает полную информацию о базовых графических инструментах, командах и конфигурациях для управления системами Linux. Подробно описан инструмент Cockpit web UI, созданный для упрощенного мониторинга и администрирования.
- В главе 9 «Установка Linux» рассказывается об общих деталях установки системы, о разделе дисков и выборе начального программного обеспечения, а также о более продвинутых инструментах установки, к примеру о файлах Kickstart.
- Глава 10 «Управление программами» объясняет, как работают пакеты программного обеспечения и как ими управлять.
- Глава 11 «Управление учетными записями» рассказывает об инструментах добавления и удаления пользователей и групп, а также об общем управлении учетными записями.
- В главе 12 «Управление дисками и файлами» приведена информация о разделении дисков, создании и редактировании файловых систем, а также о системе управления томами.

Часть IV «Администрирование серверов в Linux» поможет создать мощную систему серверов с помощью определенных инструментов.

- Глава 13 «Администрирование серверов» объясняет процессы удаленного сбора логов и загрузки Linux, а также работу инструментов мониторинга.
- В главе 14 «Администрирование сети» рассматривается настройка сетей.
- Глава 15 «Запуск и остановка служб» рассказывает, как запускать и останавливать службы.
- Глава 16 «Настройка сервера печати» описывает, как добавить принтеры к локальной системе Linux или к сети, объединяющей несколько компьютеров.
- Глава 17 «Настройка веб-сервера» рассматривает настройку веб-сервера Apache.
- Глава 18 «Настройка FTP-сервера» рассказывает о том, как настроить сервер `vsftpd`, который позволяет другим пользователям скачивать файлы из системы Linux по сети.
- Глава 19 «Настройка Samba-сервера» описывает настройку файлового обмена между системами Windows и Linux через сервер Samba.

- Глава 20 «Настройка NFS-сервера» объясняет, как задействовать функции сетевой файловой системы для передачи папок с файлами между разными системами по сети.
- В главе 21 «Диагностика Linux» рассказывается о популярных инструментах для диагностики системы Linux.

В части V «Методы обеспечения безопасности в Linux» вы узнаете, как обеспечить безопасность в системах и службах Linux.

- Глава 22 «Базовые методы обеспечения безопасности» рассказывает об основных принципах и методах защиты системы.
- Глава 23 «Продвинутые методы обеспечения безопасности» дает информацию об использовании модулей Pluggable Authentication Modules (PAM) и криптографии для обеспечения безопасности системы и аутентификации.
- Глава 24 «Повышенная безопасность с технологией SELinux» рассказывает, как подключить технологию Security Enhanced Linux (SELinux) для лучшей защиты системы.
- Глава 25 «Защита Linux в сети» объясняет, как обеспечить защиту системы с помощью функций `firewalld` и `iptables`.

Часть VI «Работа с облачными вычислениями» переходит от разбора системы к контейнеризации, облачным вычислениям и автоматизации.

- Глава 26 «Работа с облаками и контейнерами» описывает, как помещать данные в контейнеры, перемещать их, запускать, останавливать, помечать и создавать образы контейнеров.
- Глава 27 «Облачные вычисления в системе Linux» объясняет принципы облачных вычислений в Linux, описывая, как настроить гипервизоры, создать виртуальные машины и совместно использовать ресурсы по сети.
- Глава 28 «Развертывание приложений Linux в облаке» описывает, как развертывать образы Linux в различных облачных средах, включая OpenStack, Amazon EC2, а также в локальной системе визуализации Linux.
- Глава 29 «Автоматизация приложений и инфраструктуры с помощью системы Ansible» рассказывает о создании каталогов и запуске специальных команд Ansible для автоматизации настроек систем Linux и других устройств.
- Глава 30 «Развертывание приложений в контейнеры с помощью кластера Kubernetes» описывает принцип работы кластера Kubernetes и его использование для организации образов контейнеров, позволяя выполнять массовое масштабирование в центрах обработки данных.

Последняя часть содержит два приложения, которые помогут получить максимум от изучения систем Linux. Приложение А содержит рекомендации по загрузке дистрибутивов Linux. Приложение Б содержит ответы для упражнений из глав 2–30.

Условные обозначения

На протяжении всей книги код и команды выделяются моноширинным шрифтом:

```
Вот так выглядят листинги кода.
```

Если пример включает в себя как ввод, так и вывод, также используется моноширинный шрифт, но ввод обозначен полужирным шрифтом, например:

```
$ ftp ftp.handsonhistory.com
Name (home:jake): jake
Password: ******
```

Что касается оформления текста.

- Новые термины и важные слова выделяются *курсивом*.
- Сочетания клавиш выглядят следующим образом: Ctrl+A. Это обозначение указывает, что нужно нажать и удерживать клавишу Ctrl и в этот момент нажать клавишу A.
- Имена файлов и код в тексте выглядят следующим образом: `persistence.properties`.
- URL-адреса выделены вот таким шрифтом.

Следующие блоки текста привлекают ваше внимание к особенно важным моментам.

ПРИМЕЧАНИЕ

В примечании содержится дополнительная информация, на которую следует обратить особое внимание.

СОВЕТ

В таком блоке описывается интересный способ выполнения конкретной задачи.

ВНИМАНИЕ

Предупреждает вас о необходимости соблюдать особую осторожность при выполнении действий, так как это может привести к повреждению оборудования или программного обеспечения компьютера.

Переход к Linux

Если вы новичок в Linux, вы можете смутно представлять, что это за система и откуда она взялась. Возможно, вы слышали что-то о том, что эта система бесплатна или открыта. Прежде чем начать работать с Linux (что будет довольно скоро), в главе 1 найдите ответы на вопросы о ее происхождении и функциях.

Потратьте время и поработайте над этой книгой, чтобы быстро освоиться в Linux и понять, как можно заставить систему работать в соответствии со своими потребностями. Можете считать это приглашением сделать первый шаг к тому, чтобы стать экспертом по работе в Linux!

От издательства

Ваши замечания, предложения, вопросы отправляйте по адресу comp@piter.com (издательство «Питер», компьютерная редакция).

Мы будем рады узнать ваше мнение!

На сайте издательства www.piter.com вы найдете подробную информацию о наших книгах.

Часть I
Начало работы

В этой части

- Глава 1. Начало работы в Linux.
- Глава 2. Идеальный рабочий стол в Linux.

1 Начало работы в Linux

В этой главе

- Что такое Linux.
- Как появилась система Linux.
- Выбор дистрибутивов Linux.
- Профессиональные возможности в Linux.
- Сертификация специалистов в Linux.

Война операционных систем (ОС) окончена, Linux победила. Проприетарные (закрытые) операционные системы попросту не успевают за Linux, качественное улучшение которой выполняется с большой скоростью и обеспечивается благодаря взаимодействию разработчиков и внедрению инноваций. Даже Стив Балмер, бывший генеральный директор Microsoft, который однажды сравнил Linux с раковой опухолью, теперь считает, что она справляется с облачными вычислениями в Microsoft Azure гораздо лучше, чем Windows.

Система Linux — это одно из важнейших технологических достижений века. Она влияет на развитие Интернета и является технологией, которая обеспечивает поддержку целого ряда устройств, управляемых компьютерами. К тому же разработка Linux стала примером того, что совместные проекты могут дать гораздо более значительный результат, чем индивидуальная работа отдельных людей или компаний.

Компания Google использует тысячи и тысячи серверов Linux для поддержания своей поисковой системы. Android-устройства компании Google базируются на системе Linux. Браузер Google Chrome OS поддерживается операционной системой Linux.

Компания Facebook развертывает свой сайт, задействуя так называемый *стек LAMP* (система Linux, веб-сервер Apache, база данных MySQL и язык веб-сценариев PHP) — это все проекты с открытым исходным кодом. Фактически

Facebook задействует доступную для всех модель разработки с открытым исходным кодом приложений и инструментов. Эта модель помогает Facebook быстро исправлять ошибки, получать содействие пользователей со всего мира и развиваться по экспоненте.

На Linux полагаются финансовые организации, которые тратят триллионы долларов на улучшение производительности и безопасности своих операционных систем. К ним относятся Нью-Йоркская фондовая биржа, Чикагская товарная биржа и Токийская фондовая биржа.

Сегодня *облачная* технология по-прежнему остается одной из самых популярных тем, однако не все знают, что Linux и другие технологии с открытым исходным кодом — это основа величайших облачных инноваций. Любой компонент программного обеспечения, необходимый для разработки приватного или публичного облачного инструмента (например, гипервизоров, облачных контроллеров, сетевых хранилищ, виртуальных сетей и служб аутентификации), можно свободно получить в мире открытого исходного кода.

Широкое распространение Linux во всем мире сформировало огромный спрос на экспертов и специалистов в области Linux. Эта глава поможет понять, что такое Linux, откуда взялась эта система и как можно стать настоящим профессионалом в данной области.

В дальнейшем в книге описываются практические знания, которые нужно освоить на этом пути. И наконец, я покажу, как применить свои знания в облачных технологиях, включая инструменты автоматизации, например Ansible, а также технологии оркестрации контейнеров, такие как Kubernetes и OpenShift.

Что такое система Linux

Linux — это компьютерная операционная система. В операционной системе содержится программное обеспечение, которое позволяет управлять компьютером, устанавливать и запускать приложения. Перечислим функции Linux и аналогичных операционных систем.

- **Обнаружение и подготовка оборудования.** При загрузке системы (во время включения компьютера) Linux анализирует подключенные компоненты компьютера (процессор, жесткий диск, сетевую карту и т. д.) и загружает программное обеспечение (драйверы и модули), необходимое для их работы.
- **Управление процессами.** Операционная система должна отслеживать большое количество одновременно протекающих процессов и выбирать, для каких применять процессор и когда. Система также должна предлагать несколько путей запуска, остановки и смены состояния процессов.
- **Управление памятью.** Оперативная память и раздел подкачки (расширенная память) должны выделяться для приложений по мере необходимости. Операционная система определяет, как обрабатывать запросы на использование памяти.

- **Обеспечение пользовательского интерфейса.** Операционная система должна обеспечивать способы доступа к системе. Первые системы Linux были доступны из интерпретатора командной строки, называемого *оболочкой*. Сегодня широко применяются графические интерфейсы для рабочего стола.
- **Управление файловыми системами.** Структуры файловой системы встроены в операционную систему (или загружаются в виде модулей). Операционная система управляет правом собственности на файлы и каталоги (папки), содержащиеся в файловых системах, и доступом к ним.
- **Обеспечение доступа и аутентификации пользователей.** Основная функция Linux — это создание учетных записей и разграничение пользователей. Отдельные учетные записи позволяют пользователям управлять собственными файлами и процессами.
- **Администрирование процессов.** В Linux доступны сотни, а может, тысячи консольных и графических инструментов для выполнения таких задач, как добавление пользователей, управление дисками, мониторинг сети, установка программного обеспечения, а также защита компьютера и управление им (последнее обязательно). Имея инструменты с веб-интерфейсом, такие как Cockpit, выполнять сложные административные задачи стало проще.
- **Запуск служб.** Работающие в фоновом режиме процессы — так называемые *демоны* (daemon) — ожидают событий, чтобы при необходимости применять принтеры, обрабатывать журналы сообщений и запускать различные системные и сетевые службы. В Linux работает множество служб. Операционная система позволяет выбрать способ запуска и остановки таких служб. Другими словами, Linux использует браузеры для просмотра веб-страниц, и также может самостоятельно обслуживать просмотр веб-страниц для других пользователей. К популярным серверным функциям относятся серверы БД, почтовые, файловые, веб-, DNS-, DHCP-серверы и серверы печати.
- **Инструменты программирования.** В Linux доступно множество программных утилит для создания приложений и библиотек, а также для реализации специализированных интерфейсов.

Пользователю системы Linux необходимо научиться работать с этими функциями. Хотя многими функциями можно управлять с помощью графических интерфейсов, понимать, как пользоваться командным интерпретатором оболочки, очень важно для тех, кто администрирует системы Linux.

Современные системы Linux далеко опередили первые системы UNIX, на которых основана Linux. На крупных предприятиях используются следующие расширенные возможности Linux.

- **Кластеризация.** Работу системы Linux можно настроить так, что группа компьютеров будет выглядеть как одна система в сети — кластер. При этом службы настраиваются так, чтобы они передавались между узлами кластера, а пользователи могли задействовать эти службы беспрерывно.

- **Виртуализация.** Чтобы эффективнее управлять вычислительными ресурсами, Linux может стать хост-системой виртуализации. На этом хосте можно виртуально запускать другие варианты Linux, Microsoft Windows, BSD или иные операционные системы. За пределами хоста каждый из этих виртуальных «гостей» выглядит как отдельный компьютер. Технологии KVM и Xen используются в Linux для создания виртуальных хостов.
- **Облачные вычисления.** Для управления большими средами виртуализации можно применять полномасштабные облачные вычислительные платформы на базе Linux. Такие проекты, как OpenStack и Red Hat Virtualization (и дочерний проект oVirt), могут одновременно управлять многими хостами виртуализации, виртуальными сетями, аутентификацией пользователей и систем, виртуальными гостями и сетевым хранилищем. Проекты, например Kubernetes, могут управлять контейнерными приложениями в крупных центрах обработки данных.
- **Вычисления в реальном времени.** Linux можно настроить для вычислений в реальном времени, где высокоприоритетные процессы быстро обрабатываются с особой тщательностью.
- **Специализированное хранилище.** Вместо того чтобы просто хранить данные на жестком диске компьютера, можно размещать их во множестве специализированных локальных и сетевых хранилищ, имеющихся в Linux. Общие устройства хранения данных в Linux — это iSCSI, Fibre Channel и Infiniband. Существуют также платформы хранения данных с открытым исходным кодом, например Ceph (ceph.io) и GlusterFS (www.gluster.org).

Некоторые из этих сложных функций в данной книге не рассматриваются. Однако функции, относящиеся к работе оболочки, дисков, запуску и остановке служб, а также настройке различных серверов, служат основой для работы с более профессиональными функциями.

Отличие Linux от других операционных систем

Если вы впервые слышите о Linux, то, скорее всего, работаете в операционной системе Microsoft Windows или macOS. Несмотря на то что macOS создана на основе свободной операционной системы, относящейся к системе Berkeley Software Distribution (об этом позднее), сами по себе системы компаний Microsoft и Apple считаются проприетарными. Это значит следующее.

- Нельзя увидеть использованный при создании системы код, поэтому нельзя изменять основные уровни системы, если они не подходят по назначению, а также создать собственную операционную систему на основе кода-источника.
- Нельзя проверить код на наличие ошибок, изучить слабые места системы безопасности и просто посмотреть на используемый код.

- Могут возникнуть сложности при загрузке собственного программного обеспечения в операционную систему, если разработчики не позволяют задействовать встроенный интерфейс программирования.

Прочитав об этих ограничениях, вы можете задать вопрос: «А в чем проблема? Я ведь не разработчик. Мне не нужен код системы, и я не собираюсь менять его».

Возможно, проблемы нет. Однако тот факт, что другие люди могут выбирать бесплатное программное обеспечение с открытым исходным кодом и использовать его по своему усмотрению, привел к взрывному росту в сферах Интернета (например, Google), мобильных телефонов (например, Android), специальных вычислительных устройств (например, TiVo), где действуют сотни технологических компаний. Применение свободного программного обеспечения привело к снижению вычислительных затрат и увеличению количества инноваций.

Возможно, вы не хотите использовать систему Linux — как Google, Facebook и др. — для создания компании с многомиллиардным доходом. Тем не менее многие компании, применяющие систему Linux, все больше нуждаются в специалистах, которые работают с этой системой.

Вероятно, вы задаетесь вопросом, как такая мощная и гибкая компьютерная система остается свободной и открытой? Чтобы ответить на него, необходимо узнать, как появилась система Linux. Поэтому далее мы рассмотрим необычный и извилистый путь к открытому программному обеспечению и Linux.

История Linux

Все истории о Linux начинаются с сообщения Линуса Торвальдса от 25 августа 1991 года, помещенного на форуме comp.os.minix под темой «Чего вам больше всего не хватает в minix?» (groups.google.com/forum/#!msg/comp.os.minix/dlNtH7RRrGA/SwRavCzVE7gJ).

> Линус Бенедикт Торвальдс
>
> Привет всем, кто использует minux!
>
> Я пишу (бесплатную) операционную систему (как хобби, ничего особенного и профессионального вроде gnu) для AT 386 (486).
>
> Я занимаюсь этим с апреля, и, мне кажется, система почти готова. Будет полезно узнать, что вам нравится/не нравится в minux, так как моя ОС на нее похожа (то же практичное физическое размещение файловой системы, помимо прочего). Приветствуются любые предложения, но я не обещаю, что использую их :-)
>
> *Линус (torvalds@kruuna.helsinki.fi)*
>
> P. S. Да — она свободна от кода minix и включает мультизадачную файловую систему. Она НЕ переносится (использует переключение задач 386 и пр.) и, возможно, никогда не будет поддерживать ничего, кроме AT-винчестеров, потому что у меня есть только они :-(

Minix — UNIX-подобная операционная система для ПК начала 1990-х годов. Как и Minux, Linux относится к семейству Unix-подобных операционных систем. Кроме некоторых, к примеру Microsoft Windows, большинство современных компьютерных систем (включая macOS и саму Linux) были созданы из операционных систем UNIX компании AT&T.

Чтобы по достоинству оценить создание свободной операционной системы по образцу проприетарной системы компании AT&T Bell Laboratories, важно понять, в какой среде была создана UNIX и какая цепь событий сделала возможным свободное воспроизведение сущности UNIX.

ПРИМЕЧАНИЕ

Чтобы больше узнать о создании системы Linux, прочитайте биографию Линуса Торвальдса: Just for Fun: The Story of an Accidental Revolutionary. Harper Collins Publishing, 2001.

Свободная культура UNIX в стенах Bell Labs

Изначально операционная система UNIX создавалась и взращивалась в коллективной среде. Необходимость в этой ОС была продиктована не потребностями рынка, а желанием преодолеть препятствия при создании программ. Компания AT&T, владеющая торговой маркой UNIX, в итоге сделала из нее коммерческий продукт. Однако к тому времени многие концепции (и даже бóльшая часть раннего кода), которые делали систему UNIX особенной, стали достоянием общественности.

До распада в 1984 году AT&T была телефонной компанией. Вплоть до начала 1980-х у нее почти не было конкурентов, и, если вам нужна была телефонная связь в Соединенных Штатах, необходимо было обращаться в AT&T. И поэтому компания могла себе позволить финансировать чисто исследовательские проекты. Священной Меккой для таких проектов была площадка Bell Laboratories в Мюррей-Хилл, штат Нью-Джерси.

После того как в 1969 году проект под названием Multics прогорел, сотрудники Bell Labs Кен Томпсон и Деннис Ричи самостоятельно приступили к созданию операционной системы, которая предлагала бы улучшенную среду для разработки программного обеспечения. До этого времени большинство программ писалось на бумажных перфокартах, которые должны были пакетами подаваться в мейнфреймы. В лекции 1980 года на тему *The Evolution of the UNIX Time-sharing System* Деннис Ричи положил начало UNIX, сказав: «Мы хотели сохранить не просто хорошую среду для программирования, но и систему, вокруг которой могло бы образоваться сообщество. Мы знали по опыту, что суть общих вычислений, предоставляемых машинами с удаленным доступом и разделением времени, заключается не только в том, чтобы вводить программы в терминал вместо нажатия клавиш, но и в том, чтобы поощрять близкое общение».

Простота и мощь дизайна UNIX разрушили барьеры, которые раньше мешали разработчикам программного обеспечения. В основе UNIX лежали несколько ключевых концепций.

- **Файловая система UNIX.** Поскольку система включала структуру, позволяющую создавать уровни подразделов (для современных пользователей ПК они выглядят как папки внутри папок), UNIX можно было использовать для организации файлов и каталогов интуитивно понятным способом. Более того, сложные методы доступа к дискам, лентам и другим устройствам были значительно упрощены за счет представления этих устройств в виде отдельных файлов, к которым можно было обращаться как к элементам каталога.
- **Перенаправление ввода/вывода.** На раннем этапе системы UNIX включали в себя перенаправление ввода и конвейеры. Из командной строки вывод команды направлялся в файл с помощью клавиши со стрелкой вправо (`>`). Позже была добавлена концепция конвейеров (`|`), в которой выходные данные одной команды могут быть направлены на вход другой команды. Например, строка команд:

  ```
  $ cat file1 file2 | sort | pr | lpr
  ```

  конкатенирует (`cat`) файлы `file1` и `file2`, сортирует (`sort`) строки в этих файлах в алфавитном порядке, разбивает отсортированный текст на страницы для вывода (`pr`) и направляет выходные данные на принтер компьютера по умолчанию (`lpr`).

  Данный метод перенаправления ввода/вывода позволил разработчикам создавать собственные специализированные утилиты, которые можно было объединить с уже существующими. Благодаря этой модульности стала возможна разработка большого количества кода множеством людей. А сам пользователь мог просто собрать вместе нужные ему команды.
- **Мобильность.** Упрощение работы с системой UNIX привело к тому, что она стала широкодоступна для использования на различном компьютерном оборудовании. Имея драйверы устройств (в виде файлов в файловой системе), UNIX могла представлять интерфейс приложений таким образом, что программам не требовалось знать всю информацию о компьютере. Чтобы перенести UNIX в другую систему, разработчикам нужно было только заменить драйверы. А сами программы — нет!

Однако для реализации необходимого программного обеспечения требовался высокоуровневый язык программирования. Поэтому Брайан Керниган и Деннис Ричи создали язык программирования C. В 1973 году UNIX был переписан на нем. Сегодня C — по-прежнему основной язык, используемый для создания ядер операционных систем UNIX (и Linux).

В своей лекции 1979 года (bell-labs.com/usr/dmr/www/hist.html) Ричи сказал: «Сегодня единственная важная программа UNIX, написанная на ассемблере, — это сам ассемблер; практически все служебные программы написаны на языке C, как и большинство прикладных программ, хотя уже существуют сайты, написанные на

языках Fortran, Pascal и Algol 68. Несомненно, успех UNIX во многом обусловлен тем, что система удобочитаемая, модифицируемая и позволяет переносить программное обеспечение, что, в свою очередь, вытекает из ее выражения на языках высокого уровня».

Если вы поклонник Linux и интересуетесь тем, какие изначальные функции в ней сохранились, прочитайте переписанное Деннисом Ричи первое руководство по программированию в UNIX (от 3 ноября 1971 года). Найти его можно на сайте Денниса Ричи: bell-labs.com/usr/dmr/www/1stEdman.html. В документе приведены страницы с руководством к UNIX, в котором по-прежнему сохранены основные форматы документирования команд операционных систем UNIX и Linux и инструментов программирования.

Окунувшись в созданные на ранних этапах документацию и отчеты о системе UNIX, можно понять, что разработка была процессом свободным и направленным исключительно на создание превосходной системы. Это и привело к совместному использованию кода (как внутри, так и за пределами компании Bell Labs), что позволило быстро разработать высококачественную операционную систему UNIX. К тому же это дало возможность создать ОС, которую компании AT&T трудно будет откатить назад.

Распространение UNIX

До разделения в 1984 году на AT&T и еще семь компаний Baby Bell компании AT&T было запрещено продавать компьютерные системы. Такие компании, как Verizon, Qwest, Nokia и Alcatel-Lucen, ранее были частью AT&T. Из-за монополизации AT&T всей телефонной системы правительство США было обеспокоено, что негосударственная компания может повлиять на развивающуюся компьютерную индустрию.

Поскольку AT&T ранее была ограничена в продаже компьютеров непосредственно клиентам, исходный код UNIX был лицензирован для использования в университетах за символическую плату. Благодаря этому UNIX-системы распространились среди самых лучших университетов. Однако к тому времени еще не существовало ни одной операционной системы UNIX, которую не пришлось бы собирать самостоятельно и которую продавала бы сама компания AT&T.

Появление Berkeley Software Distribution

В 1975 году UNIX V6 стала первой версией системы UNIX, доступной для широкого использования за пределами Bell Laboratories. Из этого раннего исходного кода UNIX в Калифорнийском университете в Беркли был создан первый крупный вариант системы. Ее назвали Berkeley Software Distribution (BSD).

На протяжении почти всего следующего десятилетия версии систем UNIX для BSD и Bell Labs развивались в разных направлениях. BSD продолжила работу в свободном коллективном пространстве, что ранее было отличительной чертой

Bell Labs UNIX, тогда как AT&T стала двигаться в сторону коммерциализации UNIX. Этот процесс начался с создания новой лаборатории UNIX, которая переехала из Мюррей-Хилл в Саммит, штат Нью-Джерси. К 1984 году закончилось разделение AT&T, и компания была готова начать продажу системы UNIX.

Лаборатория UNIX и коммерциализация

Компания UNIX Laboratory была бриллиантом без огранки из-за того, что не имела конкретного местоположения и не умела извлекать пользу из своего продукта. При переходах от Bell Laboratories в другие области AT&T компания несколько раз меняла название. Самое известное запомнившееся имя она получила, как только отделилась от AT&T: UNIX System Laboratories (USL).

Исходный код UNIX, родившийся из USL, наследие которой было частично продано компании Santa Cruz Operation (SCO), какое-то время использовался в качестве основы для все уменьшающегося числа судебных исков SCO против крупных поставщиков Linux, таких как IBM и Red Hat, Inc. Я думаю, именно из-за этого немногие знают об усилиях USL, предпринятых для того, чтобы Linux стала успешной.

Конечно, в 1980-е годы многие компьютерные компании опасались, что недавно разделенная AT&T будет представлять бóльшую угрозу для контроля над компьютерной индустрией, чем новенькая компания из Редмонда, штат Вашингтон. Чтобы успокоить IBM, Intel, Digital Equipment Corporation и другие компьютерные компании, UNIX Lab обеспечила равные условия игры.

- **Только исходный код.** Вместо того чтобы выпускать собственный коробочный набор UNIX, AT&T продолжала продавать только исходный код и делать его равно доступным для всех лицензиатов. Каждая компания должна была устанавливать систему Unix на собственном оборудовании. Только в 1992 году, когда компания была отделена вместе с компанией Novell (под именем Univel), а затем в конечном итоге продана Novell, коммерческий набор установки UNIX (называемый UnixWare) создавался непосредственно из этого исходного кода.
- **Опубликованные интерфейсы.** Для создания атмосферы справедливости и одинаковых условий для своих OEM-производителей (производителей оригинального оборудования) AT&T начала стандартизировать возможности UNIX, чтобы различные системы все еще считались UNIX. С этой целью были созданы спецификации Portable Operating System Interface (POSIX) и AT&T UNIX System V Interface Definition (SVID), которые поставщики UNIX могли использовать для создания совместимых систем. Те же спецификации были взяты за основу при создании операционной системы Linux.

ПРИМЕЧАНИЕ

В одном из первых сообщений в группе новостей Линус Торвальдс сделал запрос на копию, предпочтительно онлайн, стандарта POSIX. Не думаю, что в AT&T ожидали, что кто-то действительно сможет написать собственный клон UNIX из этих интерфейсов, не используя ни одного фрагмента исходного кода.

- **Технический подход.** И снова большинство решений по поводу UNIX принималось на базе технических заключений вплоть до покупки компании USL. Технологически компания продвигалась, и, насколько мне известно, не было никаких разговоров о создании программного обеспечения, которое могло бы превзойти программное обеспечение других компаний или иным образом ограничить успех партнеров USL.

Когда USL в конце концов начала нанимать маркетологов и создавать настольную систему UNIX для конечных пользователей, компания Microsoft Windows уже утвердилась на рынке персональных компьютеров. Кроме того, поскольку компания всегда стремилась лицензировать исходный код, изначально созданный для крупных вычислительных систем, USL испытывала трудности с формированием цен на свои продукты. К примеру, при использовании программного обеспечения с UNIX компании приходилось за каждый компьютер платить лицензионный сбор, размер которого рассчитывался исходя из стоимости мейнфреймов (100 000 долларов), а не из стоимости персональных компьютеров (2000 долларов). Вдобавок к этому с UnixWare не поставлялось никаких прикладных программ, и именно поэтому попытка провалилась.

Однако другие компьютерные компании более удачно продавали системы UNIX. SCO нашла свою нишу на рынке, предлагая в основном настольные версии UNIX, работающие на обычных терминалах в небольших офисах. Компания Sun Microsystems продала множество рабочих станций UNIX (первоначально основанных на BSD, но объединенных с UNIX в SVR4) для программистов и высокотехнологичных приложений (к примеру, для торговли акциями).

К 1980-м годам появились и другие коммерческие системы UNIX. Новые права собственности на UNIX начали сказываться на духе сотрудничества. Были возбуждены судебные иски для защиты исходного кода UNIX и товарных знаков. В 1984 году новый ограниченный UNIX дал начало организации под названием Free Software Foundation, что в итоге привело к возникновению Linux.

Проект GNU и свободная система UNIX

В 1984 году Ричард М. Столлман начал работу над проектом GNU (gnu.org). Название является рекурсивным акронимом от английского GNU's Not UNIX — «GNU — не Unix». Проект GNU был создан организацией Free Software Foundation (FSF) с целью разработать новую свободно распространяемую систему на основе всего кода операционной системы UNIX.

На сайте проекта GNU (gnu.org/gnu/thegnuproject.html) рассказывается о том, как начинался проект, со слов самого Столлмана. Излагаются проблемы, которые проприетарные программные компании навязывали тем разработчикам программного обеспечения, которые хотели делиться, создавать и внедрять инновации.

Переписывание миллиона строк кода — дело, крайне затратное по времени для пары человек, однако если эту задачу распределить между десятками и даже сотнями разработчиков, то создать такой проект вполне реально. Ведь UNIX создавалась

именно для того, чтобы различные части кода можно было соединять. Поскольку интерфейсы команд и утилит были опубликованы и хорошо известны, работу над кодом можно было легко разделить между разработчиками.

Оказалось, что новый код не только дает прежний результат, но и в некоторых случаях является лучшей версией исходного кода UNIX. Поскольку каждый мог видеть код, создаваемый для проекта, ошибки быстро замечали и исправляли.

Если вы уже знакомы с UNIX, попробуйте найти в сотнях программных пакетов GNU с тысячами различных команд свою любимую команду из каталога Free Software Directory (directory.fsf.org/wiki/GNU). Велика вероятность того, что вы найдете ее там наряду с множеством других доступных программных проектов.

Со временем термин *«свободное программное обеспечение»* (free software) был заменен термином *«программное обеспечение с открытым исходным кодом»* (open source software). Термин *free software* чаще используется в Free Software Foundation, тогда как *open source software* встречается в Open Source Initiative (opensource.org).

Существует также объединенный вариант обоих терминов — *«свободное программное обеспечение с открытым исходным кодом»* (free and open source software, FOSS). Основополагающий принцип FOSS заключается в том, что, хотя вы можете использовать программное обеспечение по своему усмотрению, вы обязаны предоставлять общественности все улучшения, которые вносите в код. Таким образом, каждый пользователь сообщества может извлечь пользу из вашей работы, так же как вы извлекли пользу из работы других.

Чтобы четко определить принципы работы с программным обеспечением с открытым исходным кодом, проект GNU создал лицензию GNU Public License (GPL). Многие другие лицензии на программное обеспечение предполагают несколько иные подходы к защите свободного программного обеспечения, но GPL является наиболее известной и охватывает само ядро Linux. Лицензия GNU Public License включает в себя следующие основные принципы.

- **Авторские права.** Автор сохраняет за собой права на свое программное обеспечение.
- **Свободное распространение.** Пользователи могут использовать программное обеспечение GNU как им захочется, изменяя его по своему желанию. При этом они обязаны включить свой исходный код в дистрибутив (или сделать его легкодоступным).
- **Сохранение авторских прав.** Даже если вы создаете и перепродаете программное обеспечение, первоначальные авторские права GNU должны быть сохранены, а это означает, что все будущие покупатели ПО могут изменить исходный код, как это сделали вы.

На программное обеспечение GNU нет никаких гарантий. Если что-то пойдет не так, первоначальный разработчик программного обеспечения не обязан устранять проблему. Однако многие организации, большие и малые, предлагают платную поддержку (часто в форме подписки) для ПО, включенного в их дистрибутив Linux, или другого ПО с открытым исходным кодом. (Более подробное определение программного обеспечения с открытым исходным кодом см. в подразделе «Определение открытого источника OSI» далее в этой главе.)

Несмотря на успешную разработку тысяч утилит UNIX, проект GNU не смог создать самый важный фрагмент кода — ядро. Попытки построить ядро с открытым исходным кодом с помощью проекта GNU Hurd (gnu.org/software/hurd) оказались неудачными.

BSD сбавляет обороты

Единственный проект с открытым исходным кодом, который имел шанс превзойти Linux, назывался BSD. К концу 1980-х годов разработчики BSD из Калифорнийского университета (UC) в Беркли уже переписали большую часть исходного кода UNIX, созданного десятилетием ранее.

В 1989 году Калифорнийский университет распространил собственный UNIX-подобный код как Net/1, а позже, в 1991 году, — Net/2. Как раз в то время, когда Калифорнийский университет готовил полноценную UNIX-подобную операционную систему, в которой не использовался код компании AT&T, последняя подала на них в суд (это произошло в 1992 году). В иске утверждалось, что в программном обеспечении задействован код, относящийся к коммерческой тайне AT&T и их UNIX-системы.

Важно отметить, что разработчики BSD полностью переписали защищенный авторским правом код AT&T. И именно авторское право использовала компания AT&T для защиты своих прав на код UNIX. Некоторые считают, что если бы AT&T запатентовала принципы, описанные в коде, то операционной системы Linux (и любого другого клона UNIX) не существовало бы.

Иск был отозван, когда в 1994 году компания Novell купила UNIX System Laboratories у AT&T. Тем не менее в течение этого критического периода было достаточно волнений относительно законности кода BSD, чтобы компания BSD потеряла первоначальный импульс к развитию в сообществе с открытым исходным кодом. Многие начали искать им альтернативу. И тогда настало время студента из Финляндии, который работал над собственным ядром.

ПРИМЕЧАНИЕ

Сегодня версии кода BSD доступны в трех крупных проектах: FreeBSD, NetBSD и OpenBSD. Проект FreeBSD считается самым простым в применении, NetBSD — доступным на большинстве компьютерных платформ, а OpenBSD — максимально безопасным. Большинство тех, кто заинтересован в безопасности системы, все еще предпочитают BSD системе Linux. Кроме того, благодаря лицензии код BSD могут использовать поставщики проприетарных программ, таких как Microsoft и Apple, которые не позволяют делиться своим кодом системы с другими. Система macOS как раз и построена на основе BSD.

Создание недостающей части

Линус Торвальдс, будучи студентом университета Хельсинки в Финляндии, начал работу над системой Linux в 1991 году. Он хотел создать UNIX-подобное ядро, чтобы на домашнем компьютере работать с той же операционной системой, что и в университете. В это время Линус пользовался системой Minix, однако его не устраивали ограничения этой системы и он хотел пойти дальше.

Как отмечалось ранее, впервые Линус рассказал о создании ядра Linux в новостной группе comp.os.minix 25 августа 1991 года, однако сам он утверждает, что ядро не было готово до середины сентября того же года.

Хотя Торвальдс утверждал, что Linux была написана для процессора 386 и, вероятно, не была переносимой, другие упорно настаивали на увеличении переносимости уже в ранних версиях Linux (и способствовали этому). Пятого октября 1991-го была создана версия Linux 0.02, в которой большая часть исходного кода сборки была переписана на языке программирования C, что позволило сделать систему переносимой.

Ядро Linux было последней и самой важной частью кода, необходимой для создания UNIX-подобной операционной системы под лицензией GPL. Когда пользователи начали устанавливать дистрибутивы, появилась проблема с названием — это Linux и не GNU? Некоторые дистрибутивы, такие как Debian, однако, считались дистрибутивами GNU/Linux. (Если GNU не включался в название или подзаголовок операционной системы Linux, то это вызывало большое недовольство некоторых участников проекта GNU. Больше информации см. на gnu.org.)

Сейчас Linux можно описать как UNIX-подобную операционную систему с открытым исходным кодом, которая соответствует требованиям стандартов SVID, POSIX и BSD. Linux продолжает стремиться к соблюдению стандартов POSIX, а также стандартов, установленных владельцем торговой марки UNIX — компанией Open Group (opengroup.org).

Некоммерческая организация Open Source Development Labs, переименованная в Linux Foundation после слияния с Free Standards Group (linuxfoundation.org), в которой работает Линус Торвальдс, в настоящее время направляет усилия на разработку Linux. Организацию спонсируют большинство популярных коммерческих поставщиков систем и приложений Linux, включая IBM, Red Hat, SUSE, Oracle, HP, Dell, Computer Associates, Intel, Cisco Systems и сотни других. Первичная цель Linux Foundation заключается в ускорении роста Linux с помощью предоставления ее разработчикам правовой защиты и стандартов разработки программного обеспечения.

Хотя бóльшая часть усилий Linux в разработке направлена на корпоративные вычисления, значительные улучшения наблюдаются и в области настольных компьютеров. Графические среды KDE и GNOME предназначаются для обычных пользователей и постоянно улучшаются. Новые легкие настольные среды, такие как Chrome OS, Xfce и LXDE, являются эффективными альтернативами от Linux для владельцев нетбуков.

Линус Торвальдс продолжает трудиться над улучшением ядра Linux.

ПРИМЕЧАНИЕ

Историю Linux в подробностях можно прочитать в книге Open Sources: Voices from the Open Source Revolution (O'Reilly, 1999). Полноценное первое издание доступно онлайн на сайте oreilly.com/openbook/opensources/book/.

Определение открытого источника OSI

Linux предоставляет платформу, которая позволяет разработчикам программного обеспечения изменять операционную систему по своему усмотрению и получать разнообразную помощь при создании необходимых им приложений. На страже движения за открытый исходный код стоит организация *Open Source Initiative (OSI)* (opensource.org).

Хотя основная цель разработки открытого программного обеспечения заключается в том, чтобы источник кода оставался доступным всем, организация OSI определяет и другие цели. Большинство следующих правил лицензий с открытым исходным кодом служат для защиты свободы и целостности открытого исходного кода.

- **Право на свободное распространение.** Лицензия не должна ограничивать продажу или свободное распространение программного обеспечения.
- **Исходный код.** Дистрибутив приложения должен включать его исходный код, причем распространение приложения должно быть разрешено как в виде исходных кодов, так и в скомпилированной форме.
- **Производные продукты.** Лицензия должна разрешать также внесение изменений в производные продукты и их распространение на тех же условиях.
- **Целостность исходного кода автора.** Лицензия может потребовать, чтобы те, кто использует исходный код, удалили имя или версию исходного проекта, если они изменяют этот код.
- **Недопустимость дискриминации в отношении пользователей.** В лицензии недопустимы условия, дискриминирующие отдельного человека или группы людей.
- **Недопустимость дискриминации в отношении области применения.** Лицензия не может ограничивать пользователей в выборе области применения программного обеспечения. Например, она не может включать условия, запрещающие использование ПО для бизнеса или генетических исследований.
- **Распространение лицензии.** Вторичные пользователи не должны быть ограничены дополнительными лицензионными соглашениями.
- **Лицензия не должна быть специфичной для конкретного продукта.** Права на применение приложения не должны зависеть от того, является оно частью определенного программного пакета или нет.
- **Лицензия не должна ограничивать использование других продуктов.** В лицензии недопустимы ограничения на применение других продуктов, распространяемых совместно с лицензируемым программным обеспечением.
- **Нейтральность по отношению к технологиям.** Никакие условия лицензии не должны быть обусловлены использованием тех или иных технологии, стиля или интерфейса.

Лицензии с открытым исходным кодом, которые задействуются в проектах разработки программного обеспечения, должны соответствовать этим критериям, чтобы подходить под стандарты OSI. Около 70 разных лицензий принято по этим стандартам. Такое программное обеспечение обозначается наклейкой OSI Certified Open Source Software. Кроме GPL, существуют и другие популярные лицензии, одобренные OSI.

- **LGPL** (GNU Lesser General Public License). Используется для распространения библиотек, от которых зависят другие прикладные программы.
- **BSD** (Berkeley Software Distribution License). Позволяет распространять исходный код, при этом он должен сохранять информацию об авторских правах BSD и не использовать имена авторов для поддержки или продвижения производного программного обеспечения без письменного разрешения. Главное различие между лицензиями GPL и BSD в том, что BSD не требует, чтобы те, кто изменяет код, делились им с сообществом. Именно поэтому производители проприетарного ПО, такие как Apple и Microsoft, применяют код BSD в своих операционных системах.
- **MIT.** Похожа на лицензию BSD, но не требует соблюдения стандартов одобрения и продвижения.
- **Mozilla.** Касается использования и распространения исходного кода, связанного с браузером Firefox и другим программным обеспечением компании Mozilla (mozilla.org). Эта лицензия длиннее, чем упомянутые ранее, так как содержит больше критериев того, как должны вести себя участники и те, кто повторно применяет исходный код. Критерии включают в себя добавление файла изменений, знание проблем и других ограничений при добавлении изменений в код.

Конечным результатом разработки открытого исходного кода является программное обеспечение, более гибкое и имеющее меньше ограничений при использовании. Многие считают: то, что пользователи могут просматривать исходный код проекта, позволяет создать более качественное программное обеспечение. Как сказал адвокат Эрик С. Реймонд, «при достаточном количестве глаз баги выплывают на поверхность».

Как появились дистрибутивы Linux

Наличие в Интернете исходного кода, который можно было бы скомпилировать и упаковать в систему Linux, не вызывало проблем у знающих пользователей. Однако для обычных пользователей был необходим более простой способ создать свою систему Linux. Для этого лучшие специалисты начали собирать собственные дистрибутивы для Linux.

Дистрибутив Linux состоит из компонентов, необходимых для создания функционирующей системы, и команд, требующихся для запуска и работы этих компо-

нентов. Технически сам Linux считается *ядром*. Но, чтобы ядро заработало, необходимо другое программное обеспечение: базовые команды (утилиты GNU), службы (например, удаленный вход в систему или веб-серверы) и, возможно, интерфейс рабочего стола и графические приложения. Затем нужно все это собрать вместе и установить на компьютер.

Slackware — один из самых старых дистрибутивов, поддерживаемый до сих пор (slackware.com). С ним Linux стал удобным для технически хуже подкованных пользователей, а нужное программное обеспечение было уже скомпилировано и сгруппировано в пакеты. (Эти пакеты программного обеспечения поставляются в формате TAR и называются *тарболлами*.) Пользователи могут использовать базовые команды Linux, например, для форматирования диска, включения подкачки и создания учетных записей пользователей.

Вскоре были разработаны многие другие дистрибутивы Linux. Некоторые из них создавались под конкретные нужды, к примеру KNOPPIX («живой» образ на CD), Gentoo (легко настраиваемый) и Mandrake (ранее Mandriva, один из настольных). Но основой для множества дистрибутивов других стали два главных: Red Hat Linux и Debian.

Дистрибутивы Red Hat

Появившись в 1990-х, дистрибутив Red Hat очень быстро завоевал популярность по нескольким причинам.

- **Система управления пакетами RPM.** Пакеты в формате TAR хороши для загрузки программного обеспечения на компьютер, однако они не подходят для работы с ним: обновления, удаления или даже поиска информации о нем же.

  Для Red Hat была создана система управления пакетами RPM, и теперь пакеты с ПО содержали не только сами файлы, но и информацию о версии, авторе, обо всех документах и конфигурациях, а также о дате создания. Формат RPM позволяет сохранять информацию о каждом пакете с ПО в собственной базе данных. Благодаря этому облегчаются поиск, обновление и удаление пакетов.
- **Простая установка.** С помощью инсталлятора Anaconda стало гораздо легче устанавливать Linux. Пользователь отвечает на несколько вопросов, в большинстве случаев оставляя настройки по умолчанию, — и Red Hat Linux установлен.
- **Графическое администрирование.** В Red Hat добавлены простые графические инструменты для настройки принтеров, добавления пользователей, установки времени и даты, а также выполнения других административных задач. Таким образом, пользователи настольных компьютеров могут работать в системе Linux, даже не задействуя команды.

Годами Red Hat Linux был дистрибутивом, наиболее широко используемым как профессионалами, так и обычными пользователями. Компания Red Hat, Inc. поделилась свои исходным кодом и уже скомпилированными, готовыми к работе версиями Red Hat Linux (исполняемыми файлами). Но по мере того, как

потребности пользователей сообщества Linux и крупных компаний начали меняться, расходясь в разные стороны, компания Red Hat отказалась от Red Hat Linux и вместо этого начала разрабатывать две операционные системы: Red Hat Enterprise Linux и Fedora.

Дистрибутив Red Hat Enterprise Linux

В марте 2012-го компания Red Hat, Inc. стала первой компанией ПО с открытым исходным кодом с доходом более 1 млрд долларов в год. Она достигла этого после создания набора продуктов на базе Red Hat Enterprise Linux (RHEL), которые удовлетворяли потребности наиболее востребованных корпоративных вычислительных сред. После расширения линейки продуктов, включающей большинство компонентов гибридных облачных вычислений, в июле 2019 года Red Hat была куплена компанией IBM за 34 млрд долларов.

В то время как другие дистрибутивы Linux были сосредоточены на настольных системах или вычислениях для малого бизнеса, RHEL работала над функциями, необходимыми для работы с критически важными задачами крупного бизнеса и правительства. Были созданы системы, которые ускоряли транзакции для крупнейших мировых финансовых бирж и объединялись в кластеры и виртуальные хосты.

Вместо того чтобы просто продавать дистрибутив RHEL, компания Red Hat предложила целую систему преимуществ, на которую могут опираться пользователи Linux. Чтобы работать с RHEL, они приобретают подписки, которые можно применять для развертывания любой нужной версии RHEL. При переходе на другую систему можно использовать ту же подписку.

Существуют различные уровни поддержки RHEL, которые пользователи выбирают в зависимости от своих потребностей. Им гарантировано, что наряду с поддержкой они могут приобрести аппаратное и стороннее программное обеспечение, сертифицированное для работы с RHEL. А еще могут привлечь консультантов и инженеров Red Hat, чтобы собрать необходимые им вычислительные среды. Могут также оформить для своих сотрудников обучение с последующим экзаменом и получением сертификата (см. обсуждение сертификации RHCE далее в этой главе).

Red Hat добавила и другие продукты в качестве естественных расширений для Red Hat Enterprise Linux. JBoss — это промежуточный сервер для развертывания Java-приложений в Интернете или корпоративной внутренней сети. Red Hat Virtualization включает в себя хосты виртуализации, менеджеров и гостевые компьютеры, которые позволяют устанавливать, запускать, переносить и выводить из эксплуатации огромные виртуальные вычислительные среды, а также управлять ими.

В последние годы Red Hat расширила свое портфолио в сфере облачных вычислений. Red Hat OpenStack Platform и Red Hat Virtualization предлагают комплексные платформы для запуска виртуальных машин и управления ими. Однако наиболее востребованная сейчас технология — *Red Hat OpenShift*, предоставляющая гибридный облачный пакет программного обеспечения, в основе которого лежит

Kubernetes — самый популярный проект оркестрации контейнеров. С приобретением Red Hat компания IBM поставила перед собой цель контейнеризировать большинство своих приложений для работы на OpenShift.

Есть и те, кто пытался клонировать RHEL, используя находящийся в свободном доступе исходный код RHEL, перестраивая и ребрендируя его. Oracle Linux построен на основе исходного кода для RHEL, но в настоящее время его ядро с этим кодом несовместимо. CentOS — это спонсируемый сообществом дистрибутив Linux, основанный на исходном коде RHEL. Недавно Red Hat взяла на себя поддержку проекта CentOS.

В книге во многих примерах используется Red Hat Enterprise Linux, так как существует огромный спрос на специалистов, которые могут администрировать системы RHEL. Однако, если вы только начинаете работать с Linux, дистрибутив Fedora позволит усвоить первые навыки работы, которые понадобятся для использования и администрирования систем RHEL.

Дистрибутив Fedora

RHEL — это коммерческий стабильный поддерживаемый дистрибутив Linux, Fedora же — бесплатный ультрасовременный дистрибутив Linux, спонсируемый Red Hat, Inc. Fedora — это система Linux, с помощью которой Red Hat привлекает сообщество разработчиков Linux и поощряет тех, кто хочет получить бесплатный Linux для личного применения и быстрого развития.

Fedora включает в себя десятки тысяч программных пакетов, многие из которых соответствуют новейшим технологиям с открытым исходным кодом. Пользователь может опробовать новейшие настольные, серверные и административные интерфейсы Linux в Fedora бесплатно. А разработчик программного обеспечения может создавать и тестировать свои приложения, используя новейшие ядра Linux и инструменты разработки.

Fedora в большей степени фокусируется на новейших технологиях и в меньшей — на стабильности. Поэтому могут понадобиться дополнительные действия, чтобы вся система и ее программное обеспечение заработали.

Я рекомендую применять Fedora или RHEL для отработки примеров из книги по следующим причинам.

- Fedora представляет собой испытательный полигон для Red Hat Enterprise Linux. Red Hat тестирует множество новых приложений в Fedora, прежде чем добавить их в RHEL. Используя Fedora, вы получите навыки, необходимые для работы с функциями, разрабатываемыми для Red Hat Enterprise Linux.
- В качестве практики обучения Fedora более удобен, чем RHEL, к тому же включает в себя много продвинутых, готовых к работе инструментов, которые имеются в RHEL.
- Fedora «свободен» благодаря не только открытому коду, но и бесплатному доступу.

Дистрибутив Fedora чрезвычайно популярен у разработчиков программного обеспечения с открытым исходным кодом. Однако в последние несколько лет внимание многих из тех, кто начинает работу с Linux, привлек другой дистрибутив — Ubuntu.

Дистрибутив Ubuntu — еще одна ветвь Debian

Как и Red Hat Linux, Debian GNU/Linux был ранним дистрибутивом Linux, который преуспел в управлении программным обеспечением. Debian использует формат deb для пакетов и специальные инструменты для управления всеми пакетами программного обеспечения в своих системах. Debian также считается максимально стабильным дистрибутивом.

Многие дистрибутивы Linux начинались именно с дистрибутива Debian. По данным новостного портала DistroWatch (distrowatch.com), от него произошли свыше 130 активных дистрибутивов Linux. Популярные дистрибутивы на базе Debian — это Linux Mint, elementary OS, Zorin OS, LXLE, Kali Linux и многие другие. Однако наибольшим успехом пользуется производный от Debian дистрибутив Ubuntu (ubuntu.com).

Опираясь на стабильную разработку и пакеты программного обеспечения Debian, дистрибутив Ubuntu Linux (спонсируемый Canonical Ltd.) восполнил те функции, которых не хватало Debian. Стремясь привлечь новых пользователей в Linux, в проект Ubuntu добавили простой графический установщик и не менее простые в использовании графические инструменты. Дистрибутив также сосредоточился на полнофункциональных настольных системах, предлагая при этом популярные пакеты серверов.

Ubuntu создал несколько новых способов запуска Linux. С помощью «живых» компакт-дисков или «живых» USB-накопителей с Ubuntu можно запустить систему всего за несколько минут. Часто на «живые» компакт-диски добавлялись приложения с открытым исходным кодом, такие как браузеры и текстовые процессоры, которые могли работать в Windows, что облегчило переход с последнего на Linux.

Если вы уже используете Ubuntu, не переживайте: большинство примеров, приведенных в этой книге, будут работать так же хорошо в Ubuntu, как и в Fedora или RHEL.

Профессиональные возможности Linux

Если нужно разработать компьютерный проект для исследовательского института или технологической компании, с чего вы начнете? Вы начнете с идеи. После этого понадобятся инструменты, которые нужно изучить, чтобы осуществить задуманное. Затем потребуются другие люди, которые помогут в процессе творчества.

Сегодня трудности при создании такой компании, как Google или Facebook, заключаются только в том, что нужны компьютер с Интернетом и огромное коли-

чество чашек кофе, чтобы не спать всю ночь и писать код. Если у вас есть идея, как изменить мир, Linux и тысячи программных пакетов помогут вам реализовать свои мечты. В мире с открытым исходным кодом существуют сообщества разработчиков, администраторов и пользователей, всегда готовых помочь.

Проекты с открытым исходным кодом всегда ищут специалистов для написания кода, тестирования программного обеспечения или создания документации, так что вы тоже можете в этом поучаствовать. В этих проектах задействованы пользователи, которые применяют программное обеспечение, работают над ним и готовы поделиться своим опытом.

Независимо от того, стремитесь ли вы разработать следующий большой проект ПО с открытым исходным кодом или просто хотите получить навыки, необходимые для работы администратором или разработчиком Linux, сообщества помогут узнать, как устанавливать, защищать и поддерживать системы Linux.

К марту 2020 года на сайте для поиска работы Indeed.com было более 60 000 вакансий, требующих навыков работы в Linux. Почти половина из них предлагала зарплату 100 000 долларов в год и более. Сайт Fossjobs.net также размещает вакансии, связанные с Linux и другими возможностями свободного и открытого программного обеспечения.

И все это доказывает, что Linux продолжает расти и поддерживать спрос на знания и умения, касающиеся этой системы. Компании, которые начали использовать Linux, растут и развиваются вместе с системами. Пользователи Linux продолжают расширять его возможности и утверждаются в том, что благодаря экономии средств, безопасности и гибкости Linux — это хорошая инвестиция.

Как компании зарабатывают с помощью Linux

Увлеченные пользователи открытого исходного кода полагают, что именно на основе модели разработки ПО с открытым кодом получается лучшее программное обеспечение, чем на базе моделей разработки проприетарных программ. Теоретически любая компания, создающая программное обеспечение для собственного применения, может сэкономить деньги, добавив свои разработки к разработкам других пользователей, и получить гораздо лучший результат.

Компании, которые хотят зарабатывать на продаже программного обеспечения, должны подходить к этому делу креативнее, чем раньше. Можно продавать созданное вами ПО, которое включает в себя программное обеспечение GPL, при этом необходимо передать его исходный код далее. Конечно, другие пользователи могут затем переделать конечный продукт, используя его по-своему, или даже перепродать. Перечислю несколько способов, с помощью которых компании решают эту проблему.

- **Подписки на программное обеспечение.** Компания Red Hat, Inc. распространяет продукты Red Hat Enterprise Linux по подписке. За определенную сумму в год вы получаете дистрибутив для запуска Linux (не нужно ничего компилировать самостоятельно), гарантированную поддержку, инструменты для отслеживания

аппаратного и программного обеспечения на вашем компьютере, доступ к базе знаний компании и другим ресурсам.

Хотя проект Fedora включает в себя бóльшую часть того же программного обеспечения и также доступен в виде готового к запуску дистрибутива, гарантий, связанных с самим ПО и его будущими обновлениями, не предоставляется. Небольшой офис или обычный пользователь еще может пойти на риск, применяя Fedora (которая сама по себе является отличной операционной системой), но большой компании с критически важными рабочими инструментами предпочтительнее оплатить подписку за RHEL.

- **Обучение и сертификация.** С расширением использования систем Linux на уровне правительств и крупного бизнеса требуются профессионалы для работы в этих системах. Red Hat предлагает учебные курсы и сдачу экзаменов с получением сертификата об их прохождении, чтобы помочь системным администраторам повысить квалификацию в системе Red Hat Enterprise Linux. В частности, стали популярны сертификаты Red Hat Certified Engineer (RHCE) и Red Hat Certified System Administrator (RHCSA) (redhat.com/en/services/training-and-certification/why-get-certified). Подробнее о сертификатах RHCE/RHCSA читайте далее в этой главе.

  Профессиональные ассоциации компьютерной индустрии Linux Professional Institute (lpi.org) и CompTIA (comptia.org) предлагают и другие программы сертификации.

- **Система поощрений.** Применение системы поощрений — увлекательный способ заработать для компании, работающей с открытым исходным кодом. Предположим, вы используете программный пакет XYZ и вам срочно нужна новая функция для него. Заплатив вознаграждение за ПО проекту или другим его разработчикам, можно переместить требующиеся улучшения на первое место в очереди. Программное обеспечение, за которое вы заплатите, останется под лицензией с открытым исходным кодом, но у вас будут необходимые функции и, вероятно, за меньшие деньги, чем стоит разработка проекта с нуля.

- **Пожертвования.** Многие проекты с открытым исходным кодом принимают пожертвования от частных лиц или компаний, которые используют код в своих проектах. Удивительно, но многие такие проекты основаны на деятельности одного или двух разработчиков и держатся на плаву исключительно на пожертвованиях.

- **Коробочные версии, кружки и футболки.** Для некоторых проектов созданы интернет-магазины, где можно купить коробочные версии (некоторые люди все еще любят физические DVD и печатные инструкции) и различные кружки, футболки, коврики для мыши и другие товары. Если вам нравится проект, ради бога, купите футболку!

Это ни в коем случае не исчерпывающий список, потому что каждый день изобретают все более оригинальные способы поддержки тех, кто создает про-

граммное обеспечение с открытым исходным кодом. Помните, что многие люди стали участниками подобных проектов именно потому, что нуждались в данном программном обеспечении или хотели использовать его. Вклад в работу, по сути бесплатный, стоит той отдачи, которую они получают от других участников сообщества и пользователей.

Сертификаты компании Red Hat

Хотя книга не о том, как получить сертификат Linux, она затрагивает те виды деятельности, которые требуется освоить, чтобы сдать необходимые экзамены Linux. В частности, бóльшая часть того, что есть в экзаменах Red Hat Certified Engineer (RHCE) и Red Hat Certified System Administrator (RHCSA) для Red Hat Enterprise Linux 8, описана в этой книге.

Для работы в качестве ИТ-специалиста Linux чаще всего требуется сертификат RHCSA или RHCE. Экзамен RHCSA (EX200) обеспечивает базовую сертификацию, охватывающую такие темы, как настройка дисков и файловых систем, добавление пользователей, настройка веб- и FTP-сервера, а также добавление файла подкачки. Экзамен RHCE (EX300) охватывает более продвинутую конфигурацию сервера, а также расширенное знание функций безопасности, таких как SELinux и брандмауэры.

Те, кто преподавал на курсах RHCE/RHCSA и сдавал экзамены (как я в течение трех лет), не имеют права точно рассказывать о том, что на них спрашивают. Однако компания Red Hat описала, как проходят экзамены, а также опубликовала список тем, которые могут встретиться. Информация об экзаменах приведена на сайтах:

- RHCSA: redhat.com/en/services/training/ex200-red-hat-certified-system-administrator-rhcsa-exam;
- RHCE: redhat.com/en/services/training/ex294-red-hat-certified-engineer-rhce-exam-red-hat-enterprise-linux-8.

Как указано в требованиях, экзамены RHCSA и RHCE являются практическими, а это значит, задачи необходимо выполнить в реальной системе Red Hat Enterprise Linux. Оценивается успешность выполнения поставленных задач.

Если вы планируете сдавать экзамены, время от времени проверяйте страницы с требованиями, так как они периодически меняются. Имейте в виду, что RHCSA — это независимый сертификат, а чтобы получить сертификат RHCE, необходимо сдать экзамены RHCSA и RHCE. Обычно оба экзамена сдают в один день.

Вы можете записаться на обучение и экзамены RHCSA и RHCE на сайте redhat.com/en/services/training-and-certification. Обучение и экзамены проводятся в крупных городах на всей территории Соединенных Штатов и по всему миру. Навыки, необходимые для сдачи этих экзаменов, описаны далее.

Темы для RHCSA

Как говорилось ранее, темы экзамена RHCSA охватывают базовые навыки системного администрирования. Вот список актуальных тем для Red Hat Enterprise Linux 8 на сайте экзамена RHCSA (опять же проверяйте актуальность на сайте) и главы этой книги, в которых они рассматриваются.

- **Понимание основных инструментов.** Практическое знание командной оболочки (bash), включая то, как использовать правильный синтаксис команд и выполнять перенаправление ввода/вывода (< > >>). Необходимо знать, как войти в удаленную и локальную системы, а также уметь создавать, редактировать, перемещать, копировать, связывать, удалять и изменять права доступа и владельцев файлов. Кроме того, нужно знать, как искать информацию на man-страницах с руководством и в `/usr/share/doc`. Эти темы рассматриваются в главах 3 и 4. В главе 5 рассказывается, как редактировать и искать файлы.
- **Управление запущенными системами.** Понимание процесса загрузки Linux, а также того, как выключать, перезагружать и переключаться на другие целевые объекты (ранее называемые *уровнями выполнения*). Необходимо идентифицировать процессы и завершать их в соответствии с запросом. Нужно уметь находить и интерпретировать файлы журналов. В главе 15 описывается, как менять цели и управлять системными службами. В главе 6 представлено больше информации о том, как управлять процессами и менять их. Запись событий (логирование) описывается в главе 13.
- **Конфигурация хранилища.** Конфигурация разделов диска включает в себя создание физических томов и их настройку для использования менеджером Logical Volume Management (LVM) или шифрования (LUKS). Необходимо уметь настраивать эти разделы как файловые системы или разделы подкачки, которые можно монтировать или включать во время загрузки. Раздел дисков и менеджер LVM описываются в главе 12. Шифрование LUKS и другие схожие темы рассматриваются в главе 23.
- **Создание и настройка файловых систем.** Создание и автоматическая установка различных типов файловых систем, включая обычные файловые системы Linux (ext2, ext3 или ext4) и сетевые файловые системы (NFS). Создание совместных каталогов с помощью функции Set-Group-ID. Необходимо также уметь использовать менеджер LVM и расширять размер логического тома. Файловые системы рассматриваются в главе 12. В главе 20 описываются файловые системы NFS.
- **Развертывание, настройка и обслуживание систем.** Этот пункт охватывает целый ряд тем, включая настройку сети и создание задач. Необходимо уметь устанавливать пакеты из сети Red Hat Content Delivery Network (CDN), удаленного репозитория или локальной файловой системы. Об этом рассказывается в главе 13.

- **Управление пользователями и группами.** Необходимо уметь добавлять, удалять и изменять учетные записи пользователей и групп. Важно знать о сроках действия пароля и работе команды `chage`. В главе 11 приводится информация об управлении аккаунтами пользователей и групп.
- **Управление безопасностью.** Необходимо понимать основы обеспечения безопасности с помощью брандмауэров (`irewalld system-config-firewall` или `iptables`) и SELinux. Нужно уметь настраивать SSH для выполнения аутентификации по ключу. В главе 24 рассказывается про SELinux. Брандмауэры описаны в главе 25, а в главе 13 — аутентификация по ключу.

В этой книге затрагиваются большинство необходимых для сдачи экзамена тем. Для изучения тех, которых нет в книге, обратитесь к документации Red Hat (access.redhat.com/documentation). Руководства по системному администрированию содержат множество тем, связанных с RHCSA.

Темы для RHCE

Экзамен RHCE охватывает более продвинутую конфигурацию серверов, а также различные функции безопасности для их защиты в Red Hat Enterprise Linux 8. Опять же обратитесь к сайту по RHCE для ознакомления с актуальными темами к экзамену.

- **Настройка и управление системой.** Сюда входит целый ряд тем для экзамена RHCE, включая следующие.
- **Брандмауэры.** Блокировка и запуск трафика на выбранные порты системы с такими службами, как веб, FTP и NFS, а также блокировка и запуск доступа к службам на основе IP-адреса отправителя. Брандмауэры описаны в главе 25.
- **Сетевой портал аутентификации Kerberos.** Использование портала Kerberos для аутентификации пользователей в системе RHEL.
- **Системные отчеты.** Нужно уметь применять такие функции, как получение отчета об использовании системой памяти, о доступе к диску, сетевом трафике и загрузке процессора. Данная тема описана в главе 13.
- **Скрипты оболочки.** Необходимо уметь создавать простой скрипт оболочки для приема входных данных и получения выходных данных различными способами. Создание скриптов оболочки описывается в главе 7.
- **Система SELinux.** Если система Security Enhanced Linux находится в режиме Enforcing, убедитесь, что все конфигурации серверов, описанные в следующем разделе, должным образом защищены с помощью SELinux. В главе 24 представлена информация о SELinux.
- **Система Ansible.** Необходимо разбираться в основных компонентах системы Ansible (инвентаризациях, модулях, каталогах и т. д.). А также уметь устанавливать и настраивать узел управления Ansible, работать с ролями Ansible

и использовать расширенные функции системы. См. главу 29, где описано применение каталогов Ansible для установки систем Linux и управления ими.

- **Установка и настройка сетевых служб.** Для каждой из сетевых служб, вошедшей в следующий список, необходимо уметь устанавливать пакеты для служб, настраивать SELinux, чтобы разрешить получать доступ к службе, настраивать запуск службы во время загрузки, защищать службу по хосту или пользователю (с помощью утилиты iptables или функций, предоставляемых самой службой) и настраивать ее для базовой работы.
- **Веб-сервер.** Настройка сервера Apache (HTTP/HTTPS). Необходимо уметь создавать виртуальный хост, развертывать скрипт CGI, использовать частные каталоги и разрешать конкретной группе Linux управлять содержимым. В главе 17 описывается, как настроить веб-сервер.
- **DNS-сервер.** Умение настраивать DNS-сервер (привязка пакета) в качестве кэширующего сервера имен, который может пересылать DNS-запросы на другой DNS-сервер. Настраивать главную или подчиненную зоны при этом не нужно. Сервер DNS описывается со стороны клиента в главе 14.
- **NFS-сервер.** Умение настраивать NFS-сервер для обеспечения совместного доступа к определенным каталогам определенным клиентским системам, чтобы их можно было использовать для совместной работы в группах. Тема раскрыта в главе 20.
- **Сервер доступа к файлам Windows.** Необходимо уметь настраивать Linux (Samba) для передачи специальным хостам и пользователям доступа к протоколу SMB, а также объединять ресурсы для совместной работы. В главе 19 объясняется, как настроить объединенные ресурсы.
- **Почтовый сервер.** Настройка агента передачи почты postfix или sendmail для приема входящей почты извне локального хоста. Ретрансляция почты на другой сервер. Настройка почтового сервера в этой книге не рассматривается (не самая легкая задача).
- **SSH-сервер.** Умение настраивать SSH-сервер (sshd), который позволяет войти в удаленную локальную систему и выполнить аутентификацию по ключу. Другими словами, необходима настройка файла `shd.con`. В главе 13 описывается, как настроить службу sshd.
- **NTP-сервер.** Настройка NTP-сервера (ntpd) для синхронизации времени с другими пирами NTP.
- **Сервер базы данных.** Умение настраивать базу данных MariaDB и работать с ней. Настройка базы данных MariaDB описана на сайте mariad.com/kb/en/library/documentation/.

Как отмечалось ранее, на экзамене RHCE ставятся и другие задачи, и большинство из них требует знаний о настройке серверов, а также их защите разными методами. Это могут быть и брандмауэры (iptables), и система SELinux, и любые другие функции, встроенные в файлы конфигурации для конкретной службы.

Резюме

Linux — это операционная система, созданная целым сообществом разработчиков программного обеспечения по всему миру, во главе которого стоит основной ее создатель Линус Торвальдс. Linux произошла из операционной системы UNIX, однако намного превзошла ее по популярности и влиянию.

Историю операционной системы Linux можно проследить от самых ранних систем UNIX, которые бесплатно поставлялись в университеты и усовершенствовались с помощью таких систем распространения, как Berkeley Software Distribution (BSD). Организация Free Software Foundation помогла собрать множество компонентов в одну полноценную UNIX-подобную операционную систему. Ядро Linux само по себе — это самый важный компонент системы.

Большинство проектов по разработке ПО для Linux защищены набором лицензий, за которым следит организация Open Source Initiative. Самая популярная лицензия — GNU Public License (GPL). Такие стандарты, как Linux Standard Base, а также мировые организации и компании Linux (например, Canonical Ltd. и Red Hat, Inc.) позволяют Linux продолжать оставаться стабильной и производительной операционной системой.

Базовые знания о работе и администрировании системы Linux помогут вам при выполнении любых относящихся к ней задач. В следующих главах книги содержатся упражнения, с помощью которых вы можете проверить, как вы ее понимаете. Вот почему при прочтении книги лучше всего сразу практиковаться в системе, чтобы проработать все примеры и упражнения из каждой главы.

В следующей главе мы разберем, как начать работу с Linux, как установить и использовать настольную систему Linux.

2 Идеальный рабочий стол в Linux

В этой главе

- Система X Window System и среда рабочего стола.
- Запуск Linux с «живого» DVD.
- Обзор рабочего стола GNOME 3.
- Расширения для GNOME 3.
- Менеджер файлов в GNOME 3.
- Использование рабочего стола GNOME 2.
- 3D-эффекты на рабочем столе GNOME 2.

Использовать Linux на обычном компьютере становится все проще. Как и во всем, что касается Linux, у вас есть множество вариантов выбора. Существуют как полнофункциональные среды рабочего стола — GNOME или KDE, так и легковесные оболочки для компьютера, такие как LXDE или Xfce. И даже простые оконные системы-менеджеры.

Выбрав подходящий компьютер и запустив Linux, вы поймете, что большинство приложений рабочего стола схожи с приложениями на Windows или macOS. Для тех приложений, аналогов которых нет в Linux, можно запустить Windows-программу в Linux с помощью инструментов совместимости.

Задача данной главы состоит в том, чтобы познакомить читателя с принципами функционирования рабочего стола Linux, а также дать советы по установке системы и работе с ней. В этой главе мы:

- пройдемся по функциям рабочего стола и технологиям Linux;
- обсудим главные функции среды рабочего стола GNOME;
- рассмотрим, как получить максимум от использования среды GNOME.

Чтобы изучить все описанное в главе, я советую для практики установить систему Fedora. Некоторые варианты работы с Fedora.

- **Запуск Fedora с «живого» носителя.** В приложении А представлена информация о загрузке и записи Live-образа Fedora на DVD или USB-накопитель, чтобы запустить систему с оптического диска и реализовывать на практике знания, полученные при прочтении этой главы.
- **Полноценная установка Fedora.** Установка Fedora на жесткий диск и запуск с него (описано в главе 9).

Поскольку сейчас для Fedora используется интерфейс GNOME 3, большинство примеров, описанных в главе, будут работать и в других дистрибутивах с GNOME 3. Если у вас старая версия системы Red Hat Enterprise Linux (в RHEL 6 применяется GNOME 2, а в RHEL 7 и RHEL 8 — GNOME 3), в книге есть описания и GNOME 2.

ПРИМЕЧАНИЕ

В версии 17.10 операционной системы Ubuntu используемый по умолчанию интерфейс Unity был заменен на GNOME 3. Интерфейс Unity по-прежнему доступен в новых версиях системы, но уже неофициально, из репозитория Universe, поддерживаемого сообществом.

Что такое рабочий стол Linux

Современные компьютерные рабочие столы включают в себя графические окна, значки и меню, которыми пользователь управляет с помощью мыши и клавиатуры. Если вам меньше 40 лет, это может показаться обыденным, однако первые системы Linux не имели графического интерфейса. Кроме того, множество современных серверов Linux, выполняющих специальные задачи (например, веб-сервер или файловый сервер), не имеют установленного настольного программного обеспечения.

Почти все крупные дистрибутивы Linux с интерфейсами рабочего стола основаны на системе X Window System, изначально созданной в организации X.Org Foundation (x.org). Система X Window System предоставляет собой структуру, на которой строятся различные типы сред рабочего стола и простые оконные менеджеры. Аналогом X.Org в стадии разработки является система Wayland (wayland.freedesktop.org). В Fedora в качестве X-сервера по умолчанию используется Wayland, однако вы можете выбрать вместо него X.Org.

Система X Window System (или, как ее коротко называют, система *X*) была разработана еще до появления Linux и даже Microsoft Windows. Она создавалась как легковесная оболочка компьютера для удаленной работы.

Система X работает как фоновая клиент-серверная модель. X-сервер действует в локальной системе и имеет интерфейс для экрана, мыши и клавиатуры. X-клиенты (например, текстовые процессоры, музыкальные проигрыватели и браузеры изображений) запускаются из локальной системы или по сети при наличии разрешения от X-сервера.

Система была создана в то время, когда графические терминалы (тонкие клиенты) управляли только клавиатурой, мышью и дисплеем. А приложения, дисковое хранилище и вся вычислительная мощность находились на больших централизованных компьютерах. Таким образом, приложения запускались на больших компьютерах, но отображались и управлялись по сети на тонком клиенте. Позже тонкие клиенты заменили персональными настольными компьютерами. Большинство приложений для клиентов на ПК запускались локально и использовали те же вычислительную мощность, дисковое пространство, память и другие функции, в то время как те приложения, которые не запускались из локальной системы, не были разрешены.

Сама система X имеет простой серый фон и указатель мыши в виде символа X. На экране X нет меню, панелей и значков. Если бы вы запустили X-клиент (например, окно терминала или текстовый процессор), он появился бы на X-дисплее и его нельзя было бы переместить, свернуть или закрыть. Эти функции добавляются вместе с менеджером (администратором) окон.

Менеджер окон позволяет управлять окнами на рабочем столе и меню для запуска приложений и прочих задач, для которых требуется рабочий стол. Полноценная среда рабочего стола включает в себя менеджер окон, а также меню, панели и обычно программный интерфейс приложений, который используется для описания способов взаимодействия программ друг с другом.

Итак, как же понимание принципов действия интерфейса рабочего стола поможет в работе с Linux? Вот несколько вариантов.

- Так как среда рабочего стола в Linux не требуется для запуска самой Linux, систему можно загрузить и без рабочего стола. Это будет выглядеть как командный интерфейс с одной строкой для ввода команды. А рабочий стол можно выбрать позднее. После установки системы можно указать, включать ли рабочий стол после загрузки компьютера или только по необходимости.
- Для систем Linux, предназначенных для работы на маломощных компьютерах, можно выбрать эффективный, хотя и менее функциональный менеджер окон (например, `twm` или `fluxbox`) или легковесную оболочку рабочего стола (например, LXDE или Xfce).
- Для более мощных компьютеров можно выбрать более надежную среду рабочего стола (к примеру, GNOME и KDE), которые могут наблюдать за событиями (допустим, за подключением USB-накопителя) и реагировать на них (например, открытием окна для просмотра содержимого накопителя).
- На компьютере может быть установлено несколько сред рабочего стола, и вы можете выбрать, какую из них запускать при загрузке системы. Таким образом,

разные пользователи на одном компьютере могут задействовать разные рабочие столы.

Для Linux доступно множество различных сред рабочего стола на выбор.

- **GNOME** — одна из сред по умолчанию для Fedora, Red Hat Enterprise Linux и многих других. Это профессиональная среда с акцентом на стабильность, а не на новые забавные эффекты.
- **K Desktop Environment** (KDE) — это вторая по популярности среда для Linux. В ней гораздо больше наворотов, чем в GNOME, и лучше продумана интеграция приложений. KDE также доступна для Fedora, Ubuntu и многих других систем Linux, кроме RHEL 8.
- **Xfce** — одна из первых легковесных оболочек для рабочего стола. Она подходит для более старых или маломощных компьютеров. Доступна для Fedora, Ubuntu и других дистрибутивов.
- **LXDE** (Lightweight X11 Desktop Environment) — быстрая и энергосберегающая рабочая среда. Чаще всего она используется на недорогих устройствах (к примеру, нетбуках) и «живых» носителях (CD и USB). Эта среда по умолчанию установлена на дистрибутиве KNOPPIX. Она недоступна на RHEL, но доступна на Fedora и Ubuntu.

GNOME изначально был разработан как аналог рабочего стола macOS, а KDE предназначался для эмуляции среды рабочего стола Windows. В большинстве процессов и упражнений, описанных в книге, используется рабочий стол GNOME, так как это наиболее популярная среда рабочего стола, которую часто задействуют в бизнес-системах Linux. А еще GNOME можно применять на любых других дистрибутивах Linux.

Запуск Fedora со средой GNOME из образа

«Живой» ISO-образ Linux — это самый быстрый способ установить и запустить систему Linux. Образ можно записать на компакт-диск, DVD или USB-накопитель (это зависит от его размера) и загрузить на компьютер. С помощью «живого» образа можно временно подключить Linux для управления компьютером, что не повлияет на содержимое жесткого диска.

Если у вас установлена операционная система Windows, Linux просто проигнорирует ее и возьмет компьютер под контроль. После работы с образом Linux можно перезагрузить компьютер, вытащить накопитель и запустить штатную операционную систему, установленную на жестком диске.

Чтобы выполнять задания из книги на рабочем столе GNOME, советую обзавестись DVD с дистрибутивом Fedora (информация об этом есть в приложении А).

Поскольку DVD-накопитель выполняет чтение данных с диска и обработку в оперативной памяти, процесс протекает медленнее, чем при полноценно установленной системе Linux. Кроме того, можно изменять файлы, добавлять программное обеспечение и иным образом настраивать свою систему, но если не сохранить эти изменения на жестком диске или внешнем накопителе, то все данные будут удалены после выключения/перезагрузки.

Это хороший вариант, чтобы попрактиковаться в Linux, но совершенно не подходит, если нужна постоянная настольная или серверная система Linux. По этой причине я рекомендую при наличии дополнительного компьютера установить полноценную систему Linux на жесткий диск и применять ее, как описано в главе 9.

Если у вас уже есть CD- или DVD-накопитель, выполните следующие действия.

1. **Используйте компьютер.** Если у вас есть стандартный компьютер (32- или 64-разрядный) с приводом CD/DVD, минимум 1 Гбайт оперативной памяти (RAM) и процессором с частотой 1 ГГц, можно начинать работу. (Убедитесь, что загружаемый образ соответствует архитектуре компьютера — 64-разрядный носитель не работает на 32-разрядном компьютере. Fedora 33 и RHEL 8 не поддерживают 32-разрядные компьютеры, поэтому для работы понадобятся более старые версии этих дистрибутивов.)
2. **Подключите носитель CD/DVD.** Подключите CD/DVD или USB-носитель к компьютеру и перезагрузите его. В зависимости от порядка загрузки, настроенного на компьютере, «живой» образ может запускаться непосредственно из BIOS (кода, управляющего компьютером до запуска операционной системы).
3. **Запустите Fedora.** Если выбранный диск доступен, появится соответствующий загрузочный экран. Выделите строку Start Fedora нажмите клавишу Enter, чтобы запустить образ.
4. **Начинайте работу с рабочим столом.** В случае с Fedora «живой» образ позволяет выбрать между установкой Fedora и ее загрузкой непосредственно с носителя на рабочий стол GNOME 3.

Теперь можно переходить к следующему разделу, где говорится о работе GNOME 3 в Fedora, Red Hat Enterprise Linux и других операционных системах. После этого я расскажу о рабочем столе GNOME 2.

ПРИМЕЧАНИЕ

Если вместо загрузки образа запускается штатная операционная система, необходимо во время загрузки выполнить дополнительное действие. Перезагрузите компьютер, запустите BIOS и найдите команду Boot Order. На экране может быть указано, что при запуске компьютера нужно нажать клавишу F12 или F1. Быстро нажмите ее, находясь на экране BIOS. После этого откроется экран с доступными вариантами. В перечне с порядком загрузки выделите строку для CD/DVD или USB-накопителя и нажмите клавишу Enter, чтобы загрузить «живой» образ. Если этой строки нет, возможно, нужно зайти в настройки BIOS и сменить приоритет загрузки на CD/DVD или USB-накопитель.

Использование рабочего стола GNOME 3

Рабочий стол GNOME 3 радикально отличается от GNOME 2.x, своего предшественника. GNOME 2.x — вариант работоспособный, а GNOME 3 — элегантный. Рабочий стол GNOME 3 больше похож на графические интерфейсы мобильных устройств, здесь меньший акцент делается на кнопках мыши и комбинациях клавиш и больший — на движении указателя и действиях в результате одного щелчка кнопкой мыши.

Он не такой жесткий и структурированный и расширяется по мере необходимости. При запуске нового приложения его значок добавляется на панель приложений. Если одно рабочее место заполнено, открывается новое, на котором можно разместить дополнительные приложения.

После загрузки компьютера

При загрузке образа на рабочем столе появится имя пользователя Live System User. При загрузке установленной системы на экране будут видны учетные записи пользователей, предусматривающие ввод пароля. Войдите в свою учетную запись.

На рис. 2.1 показан пример рабочего стола GNOME 3 в Fedora. Нажмите клавишу Windows или переведите указатель мыши в верхний левый угол стола, чтобы переключить режим экрана с пустого на обзор стола.

Рис. 2.1. Пример рабочего стола GNOME 3 в Fedora

Вначале на рабочем столе GNOME 3 практически ничего нет. В верхнем левом углу находится кнопка Обзор (Activities), сверху посередине — часы и дата и несколько значков справа: звук, интернет-соединение и имя текущего пользователя. В режиме обзора показаны все приложения, активные окна и другие рабочие места.

Навигация с помощью мыши

Воспользуйтесь мышью для навигации по рабочему столу GNOME 3.

1. **Переключение окон.** Переместите указатель мыши в левый верхний угол экрана — туда, где находится кнопка Activities (Обзор). Каждый раз, когда он там оказывается, экран отображает открытые окна или список доступных действий. (То же самое происходит при переключении с нажатой и удерживаемой клавишей Windows.)
2. **Запуск окон с панели приложений.** Щелкните на панели слева, чтобы открыть приложения (Firefox, файлы (File Manager), Rhythmbox или другое). Снова переведите указатель мыши в верхний левый угол и переключитесь в другой режим просмотра (обзора или полноразмерного наложения). На рис. 2.2 показан пример экрана с миниатюрами приложений.

Рис. 2.2. Видны все открытые окна

3. **Запуск приложений из списка.** В режиме наложения нажмите кнопку Show Applications (Показать приложения) внизу слева (квадрат с девятью точками). Представление экрана изменится на набор значков приложений, установленных в системе (рис. 2.3).
4. **Просмотр дополнительных приложений.** Из окна со списком приложений можно сменить отображение всех приложений и поменять их местами.
   - **Перелистывание страниц.** Чтобы увидеть приложения, которых нет на экране, используйте точки внизу, чтобы переключиться на следующую страницу. Можно делать это и с помощью колесика мыши.

Рис. 2.3. Все доступные приложения

- **Популярные.** Нажмите кнопку Frequent (Популярные) внизу экрана, чтобы увидеть часто используемые приложения. Нажмите кнопку All (Все), чтобы снова показать все приложения.
- **Запуск приложения.** Чтобы запустить приложение в действующем рабочем пространстве, щелкните левой кнопкой мыши на его значке. Щелчок правой кнопкой мыши открывает меню (рис. 2.4), в котором можно запустить приложение в новом окне, добавить приложение в Избранное (то есть приложение появится на панели приложений) или удалить оттуда, а также показать информацию о приложении.

5. **Запуск дополнительных приложений.** Запустите дополнительные приложения. Обратите внимание на то, что при открытии нового приложения на панели задач слева появляется его значок. Есть и другие способы запустить приложение.
   - **Через значок приложения.** Щелкните на значке приложения, чтобы запустить его.
   - **Запуск из панели задач.** В оконном режиме левой кнопкой мыши перетащите любой значок приложения с панели задач в любое из рабочих пространств справа.
6. **Использование нескольких рабочих пространств.** Снова переместите мышь в верхний левый угол, чтобы отобразить все окна. Обратите внимание на то, что все приложения справа находятся на одном рабочем месте, в то время как другое место пусто. Перетащите несколько окон на пустое рабочее место. На рис. 2.5

показан пример интерфейса с несколькими рабочими местами. Дополнительное рабочее место будет создаваться каждый раз, когда предыдущее заполнится. Можно перетащить окна приложений на любое рабочее место, а затем выбрать само место с окнами.

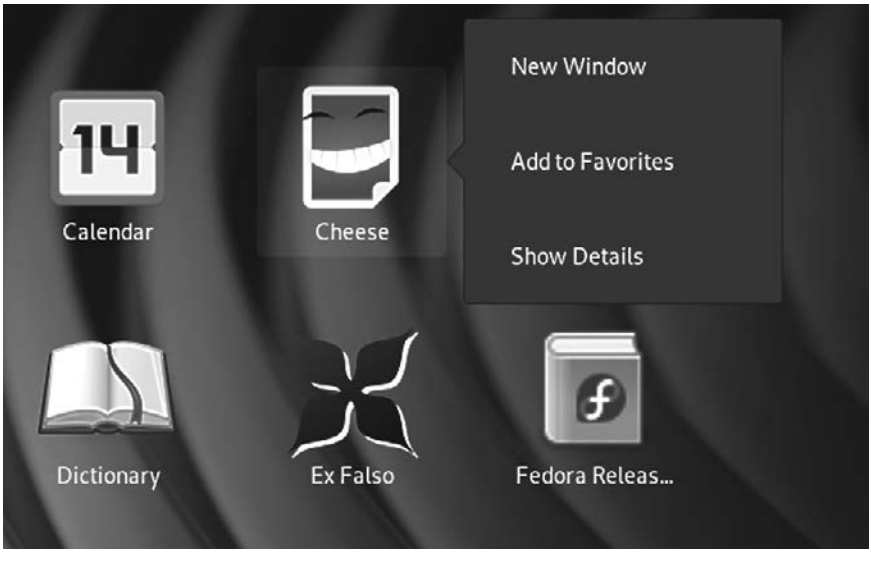


Рис. 2.4. Щелкните правой кнопкой мыши, чтобы открыть меню выбора приложения

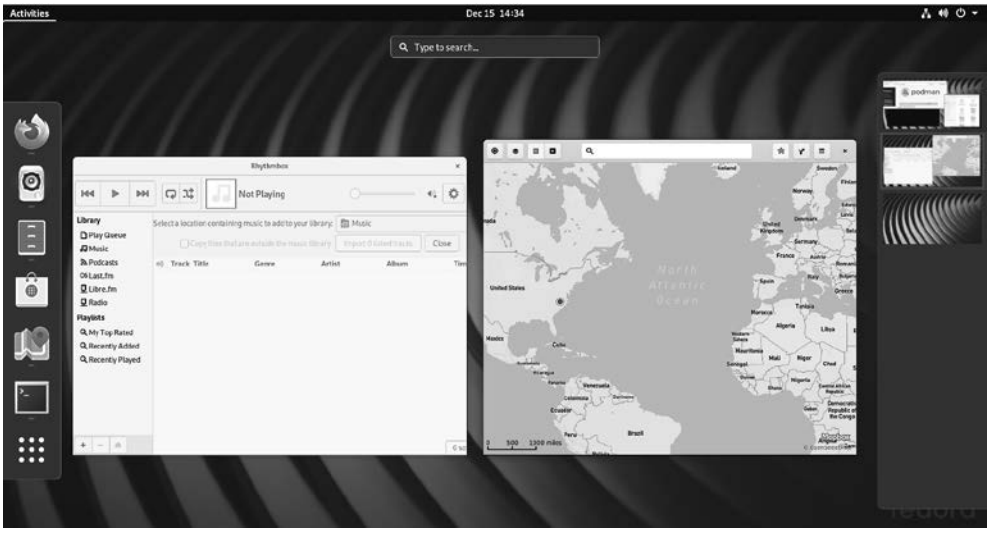


Рис. 2.5. Если используются несколько рабочих мест, переключаться между ними можно на панели слева

7. **Использование меню окна.** Переместите указатель мыши в верхний левый угол экрана, чтобы вернуться в активное рабочее пространство (вид большого окна). Щелкните правой кнопкой мыши на строке заголовка окна, чтобы открыть меню. Действия в меню:
   - Minimize (Свернуть) — временно скрыть окно;
   - Maximize (Развернуть) — открыть окно на полный размер;
   - Move (Переместить) — включить режим перемещения. Переместите окно в нужное место мышью и щелкните кнопкой, чтобы подтвердить изменение;
   - Resize (Изменить размер) — включить режим изменения размера. Мышью измените размер окна, а затем щелкните кнопкой, чтобы подтвердить изменение.

   Несколько вариантов размещения окна: выберите вариант Always on Top (Всегда сверху), чтобы активное окно всегда отображалось поверх других окон рабочего места. Выберите вариант Always on Visible Workspace (Всегда на видимом рабочем месте), чтобы всегда показывать окно в видимом рабочем месте. Выберите вариант Move to Workspace Up (Переместить на рабочее место вверх) или Move to Workspace Down (Переместить на рабочее место вниз), чтобы переместить окно в рабочее место выше или ниже соответственно.

Если вам некомфортно управлять GNOME 3 мышью или у вас ее нет, далее в разделе мы разберем, как действовать на рабочем столе, пользуясь клавиатурой.

Навигация с помощью клавиатуры

Если вы предпочитаете использовать клавиатуру, GNOME 3 позволяет выполнять команды непосредственно с нее несколькими способами, включая следующие.

- **Клавиша Windows.** Нажмите клавишу Windows. На большинстве клавиатур для ПК клавиша с логотипом Microsoft Windows находится рядом с клавишей Alt. Она переключает режимы просмотра с обзора на текущее рабочее место, что очень удобно.
- **Выбор режима.** В режиме просмотра окон или списка приложений нажмите и удерживайте сочетание клавиш Ctrl+Alt+Tab, чтобы просмотреть меню режимов (рис. 2.6). Все еще удерживая клавиши Ctrl+Alt, снова нажмите клавишу Tab, чтобы выделить один из значков в меню, и отпустите, чтобы выбрать режим.
  - Top Bar (Верхняя панель). Открывает верхнюю панель. После выбора можно переключаться между ее элементами (обзор, календарь и меню).
  - Dash (Панель приложений). Выделяет первое приложение на панели задач слева. Используйте клавиши со стрелками для перемещения вверх и вниз по меню, а затем нажмите клавишу Enter, чтобы открыть выбранное приложение.
  - Windows (Окна). Открывает режим просмотра окон.
  - Applications (Приложения). Режим просмотра всех приложений.

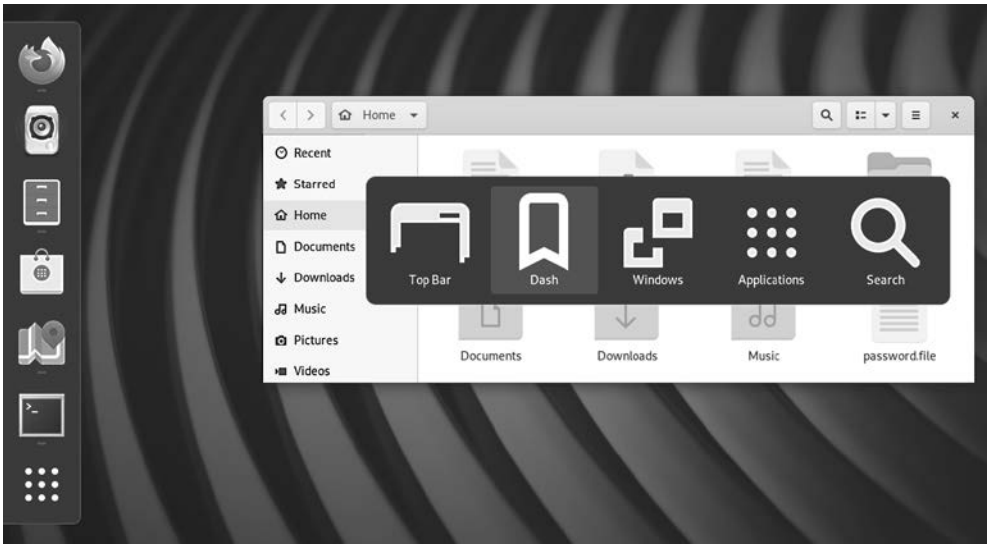


Рис. 2.6. Нажмите сочетание клавиш Ctrl+Alt+Tab, чтобы отобразить дополнительные команды

- Search (Поиск). Выделяет область поиска. Введите несколько букв, чтобы отфильтровать нужные приложения. Затем нажмите клавишу Enter, чтобы запустить нужное приложение.

- **Выбор активного окна.** Вернитесь к любому рабочему месту (нажмите клавишу Windows, если вы не в активном рабочем месте). Нажмите сочетание клавиш Alt+Tab, чтобы просмотреть список всех активных окон (рис. 2.7). Удерживайте клавишу Alt при нажатии клавиши Tab (или клавиш ← или →), чтобы выделить нужное приложение из списка активных окон. Если в приложении открыты несколько окон, нажмите сочетание клавиш Alt+` (расположена над клавишей Tab), чтобы выбрать одно из подокон. Отпустите клавишу Alt, чтобы подтвердить выбор.

Рис. 2.7. Нажмите сочетание клавиш Alt+Tab, чтобы выбрать, к какому из открытых приложений перейти

- **Запустить команду или приложение.** Из любого активного рабочего места можно запустить команду Linux или приложение, например.
  - **Режим приложений.** В режиме обзора нажмите и удерживайте сочетание клавиш Ctrl+Alt+Tab и продолжайте нажимать клавишу Tab до тех пор, пока значок Applications (Приложения) не будет выделен, затем отпустите сочетание клавиш Ctrl+Alt. Будет задействован режим Applications (Приложения) с выделенным значком первого приложения. Используйте клавишу Tab или клавиши со стрелками (←, ↑, ↓ и →), чтобы выделить значок нужного приложения, а затем нажмите клавишу Enter.
  - **Командная строка.** Если вам известно имя (или часть имени) необходимой команды, нажмите сочетание клавиш Alt+F2, чтобы открыть командную строку. Введите имя команды (например, попробуйте `gnome-calculator`, чтобы открыть калькулятор).
  - **Строка поиска.** В режиме обзора нажмите сочетание клавиш Ctrl+Alt+Tab и продолжайте нажимать клавишу Tab до тех пор, пока не будет выделен значок лупы (поиск), затем отпустите сочетание клавиш Ctrl+Alt. В выделенной строке поиска введите несколько букв названия или описания приложения (выпадут различные варианты). Продолжайте печатать до тех пор, пока нужное приложение не будет выделено (например, Firefox), и нажмите клавишу Enter, чтобы запустить его.
  - **Панель приложений.** В режиме обзора нажмите сочетание клавиш Ctrl+Alt+Tab и продолжайте нажимать клавишу Tab до тех пор, пока не будет выделен значок закладки (Dash (Панель приложений)), затем отпустите сочетание клавиш Ctrl+Alt. На панели приложений используйте стрелки вверх и вниз, чтобы выделить нужное приложение, и нажмите клавишу Enter.
- **Выход.** Если вы не хотите выполнять то или иное действие, нажмите клавишу Esc. Например, после нажатия клавиш Alt+F2 (для ввода команды), запуска приложения с верхней панели или перехода в режим обзора клавиша Esc возвратит вас в активное окно на активном рабочем столе.

Я надеюсь, что теперь вы освоились в интерфейсе GNOME 3. Самое время попробовать запустить полезные и забавные приложения в GNOME 3.

Настройка рабочей среды GNOME 3

Многие настройки для GNOME 3 реализуются автоматически. Однако для персонализации стола все же нужно настроить GNOME 3 вручную. Большинство необходимых настроек доступно в окне Settings (Параметры) (рис. 2.8). Найдите в списке приложений одноименный значок.

Несколько советов по настройке рабочего стола GNOME 3.

- **Настройка интернет-соединения.** Проводное сетевое соединение чаще всего настраивается автоматически при загрузке системы Fedora. Для настройки беспроводной связи выберите из списка подходящий вариант и введите пароль.

Значок на верхней панели позволяет выполнить настройку проводной или беспроводной сети. Дополнительную информацию о настройке сети см. в главе 14.

- **Bluetooth-соединение.** Если на вашем компьютере установлен Bluetooth-модуль, то его можно связать с другими устройствами Bluetooth, например с Bluetooth-гарнитурой или принтером.
- **Устройства.** В окне Settings (Параметры) можно подключить и настроить клавиатуру, мышь, сенсорную панель, принтеры, съемные носители и т. д.
- **Звук.** Перейдите на вкладку Sound (Звук), чтобы настроить устройства записи и вывода звука в системе.

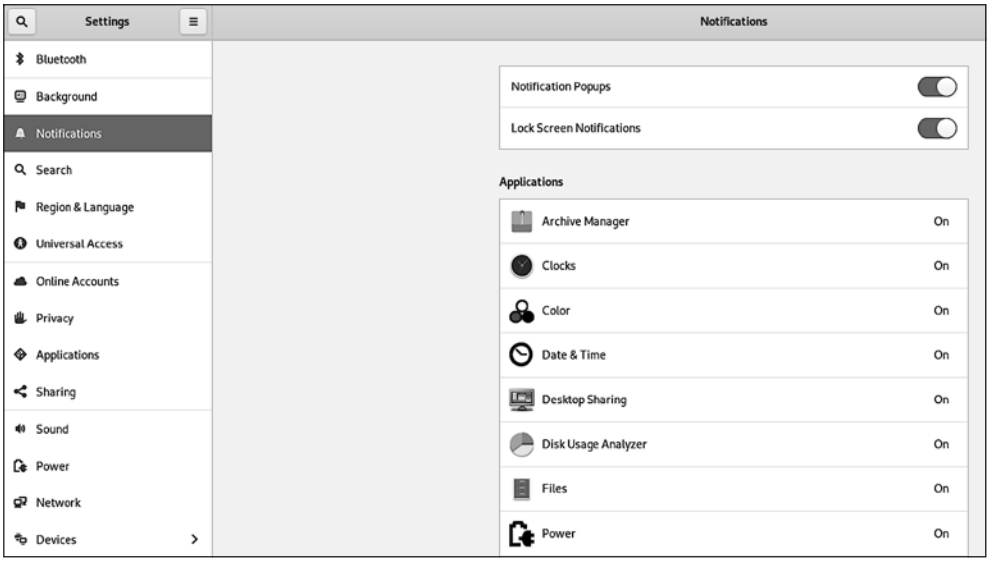


Рис. 2.8. Настройки рабочего стола в окне Settings (Параметры)

Расширения для рабочего стола GNOME 3

Если в оболочке GNOME 3 есть не все, что вам нужно, не отчаивайтесь. Всегда можно добавить к рабочему столу расширения и дополнительные возможности. Кроме того, инструмент под названием GNOME Tweaks позволяет менять расширенные настройки в GNOME 3.

Расширения для GNOME

Расширения для оболочки GNOME позволяют изменять внешний вид и поведение рабочего стола. В браузере Firefox зайдите на сайт расширений extensions.gnome.org. На сайте показано, какие расширения вы уже установили и какие из них доступны. (Необходимо установить дополнение к браузеру и разрешить сайту просматривать расширения на рабочем столе.)

Страница расширений отображает только совместимые с вашей версией системы доступные приложения. Многие расширения позволяют добавить некоторые полезные функции из GNOME 2.

- **Applications Menu.** Добавляет на верхнюю панель меню приложений в духе GNOME 2.
- **Places Status Indicator.** Добавляет индикатор состояния системы, аналогичный меню Places (Места) в GNOME 2, чтобы быстрее перемещаться к полезным папкам системы.
- **Window list.** Добавляет на верхнюю панель список активных окон, аналогичный списку на нижней панели в GNOME 2.

Чтобы активизировать загруженное расширение, установите переключатель, расположенный рядом с его именем, в положение ON. Или можно щелкнуть кнопкой мыши на имени расширения в списке, чтобы просмотреть страницу расширения, и уже на ней установить переключатель в положение OFF или ON. Нажмите кнопку Download (Скачать), чтобы загрузить и установить новое расширение. Затем оно будет добавлено на рабочий стол.

На рис. 2.9 показан интерфейс с меню Applications (Приложения) и Places (Места) (раскрыто, показаны каталоги).

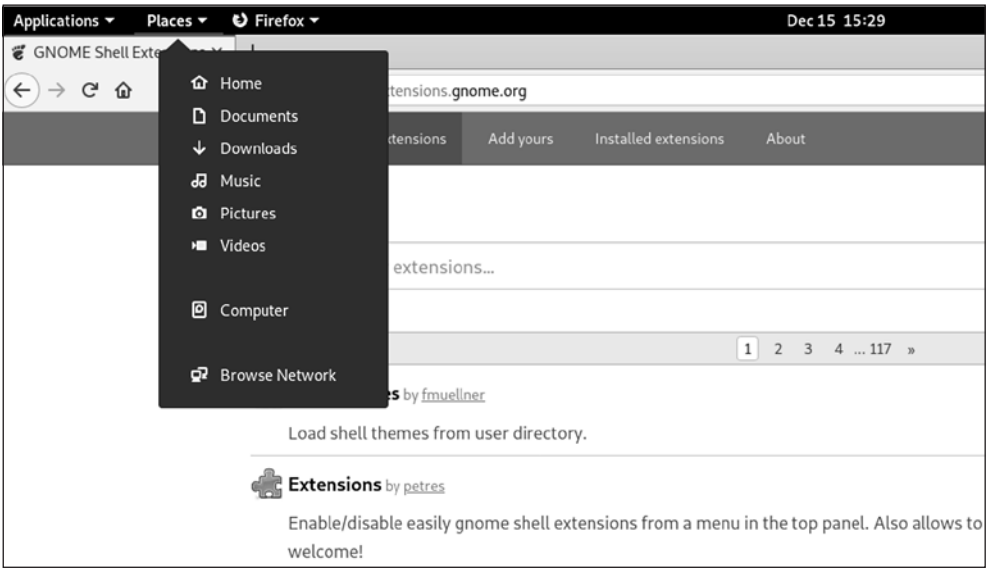


Рис. 2.9. Расширения добавляют новые функции рабочему столу GNOME 3

Для оболочки GNOME доступны свыше 100 расширений, и постепенно появляются все новые. К популярным расширениям относятся Notifications Alert (уведомляет о непрочитанных сообщениях), Presentation Mode (предотвращает включение заставки во время презентации) и Music Integration (интегрирует

музыкальные плееры в GNOME 3, чтобы можно было получать информацию о воспроизводимых песнях).

Поскольку сайт extensions.gnome.org позволяет отслеживать имеющиеся расширения, перейдите на вкладку Installed extensions в верхней части страницы, чтобы увидеть все установленные расширения. Здесь можно отключать и включать расширения и даже удалять их.

Дополнительные настройки Gnome

Если вам не подходят некоторые встроенные функции GNOME 3, их можно изменить с помощью инструмента GNOME Tweak Tool (Дополнительные настройки GNOME). Этот инструмент не устанавливается по умолчанию вместе с Fedora, но его можно добавить с помощью пакета `gnome-tweaks` (см. главу 10, где рассказано, как устанавливать программное обеспечение в Fedora). После установки инструмента на экране приложений появится значок GNOME Tweak Tool (Дополнительные настройки GNOME). Начните с категории Top bar (Верхняя панель), чтобы настроить верхнюю панель. На рис. 2.10 показан раздел Appearance (Внешний вид) приложения GNOME Tweak Tool (Дополнительные настройки GNOME).

Рис. 2.10. Измените настройки рабочего стола с помощью приложения GNOME Tweak Tool (Дополнительные настройки GNOME)

Если шрифт слишком мелкий, выберите категорию Fonts (Шрифты) и нажмите кнопку + рядом с полем Scaling Factor (Коэффициент масштабирования), чтобы увеличить размер шрифта. Можно также установить определенный шрифт для документов, интерфейса и моноширинного текста.

В разделе Top bar (Верхняя панель) можно изменить способ отображения времени и даты. Чтобы изменить внешний вид рабочего стола, выберите категорию Appearance (Внешний вид) и выберите нужную тему значков, приложений и указателей мыши.

Работа с приложениями

Fedora GNOME 3 desktop live DVD включает в себя классные приложения, с которыми сразу можно начать работать. Чтобы использовать GNOME 3 постоянно, следует установить на жесткий диск необходимые приложения (текстовый процессор, графический редактор, приложение для рисования и т. д.). В следующих разделах будут описаны интересные приложения, с которых можно начать.

Управление файлами и папками с помощью файлового менеджера

Для перемещения, копирования, удаления, переименования файлов и папок и других действий с ними в GNOME 3 можно применять файловый менеджер Files (Файлы), также известный как Nautilus. Он добавлен в рабочую среду GNOME и работает так же, как и другие файловые менеджеры, которые есть в Windows и macOS.

Чтобы открыть менеджер, щелкните кнопкой мыши на значке Files (Файлы) на панели приложений или в списке приложений. В вашей учетной записи есть набор папок, в которых хранятся наиболее распространенные типы файлов: Music, Pictures, Videos и др. Все они находятся в каталоге Home. На рис. 2.11 показана папка Home, открытая в программе Files (Файлы).

Файлы, скачанные из Интернета или созданные, к примеру, с помощью текстового редактора, можно сохранить в этих папках. По мере необходимости можно создавать новые папки, перетаскивать, копировать и удалять файлы и папки.

Поскольку программа Files (Файлы) не сильно отличается от большинства файловых менеджеров из других ОС, в этой главе мы не будем рассматривать ее подробно. Но все же я хочу отметить несколько полезных и не всегда очевидных возможностей, имеющихся у программы Files (Файлы).

- **Домашняя папка.** У пользователя есть полный доступ к файлам и папкам, находящимся в домашней папке. Большинство других уголков файловой системы недоступны обычному пользователю.

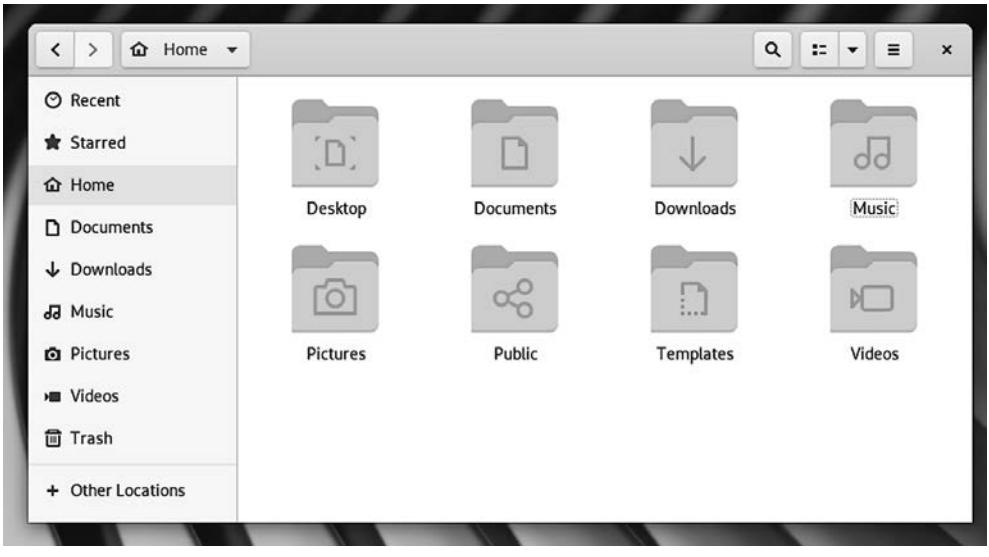


Рис. 2.11. Управление папками и файлами происходит в программе Files (Файлы)

- **Организация файловой системы.** Можно предположить, что в файловой системе каталог Home отображается под именем `/home`, однако он носит имя учетной записи пользователя, например `/home/liveuser` или `/home/chris`. В следующих главах вы узнаете, как организована файловая система (особенно в отношении командной оболочки Linux).
- **Управление файлами и папками.** Щелкните правой кнопкой мыши на файле или папке, чтобы открыть список доступных действий, например скопировать, вырезать, переместить в корзину (удалить) или открыть объект.
- **Создание папок.** Чтобы создать новую папку, щелкните правой кнопкой мыши в окне папки и выберите команду New Folder (Создать папку). Введите новое имя папки и нажмите клавишу Enter.
- **Доступ к удаленным папкам.** Программа Files (Файлы) позволяет отображать содержимое как локальной файловой системы, так и удаленных серверов. В окне программы Files (Файлы) выберите пункт Other Locations (Другие места) на панели слева. В появившемся окне можно подключиться к удаленному серверу через протоколы SSH (secure shell), FTP (с авторизацией и без), сети Windows, WebDav (HTTP) и Secure WebDav (HTTPS). При необходимости введите логин и пароль пользователя — и все содержимое удаленного сервера появится в окне программы Files (Файлы). На рис. 2.12 показан пример окна программы Files (Файлы) с запросом пароля для входа на удаленный сервер по протоколу SSH (ssh://192.168.122.81).

Рис. 2.12. Доступ к удаленным папкам в окне программы Files (Файлы)

Установка дополнительного ПО и управление им

Операционная система Fedora включает в себя браузер (Firefox), файловый менеджер Files (Файлы) и еще несколько популярных приложений. Однако существует множество других полезных приложений, которые просто не поместились бы в дистрибутив из-за своего размера. Если вы установили операционную систему Fedora на жесткий диск (как описано в главе 9), вам почти наверняка понадобится добавить еще какое-нибудь программное обеспечение.

ПРИМЕЧАНИЕ

Можно установить программное обеспечение и в системе, запущенной в «живом» режиме. Однако имейте в виду, что ресурсы виртуальной памяти (ОЗУ) сильно ограничены. Кроме того, после перезагрузки системы все, что вы установили, будет удалено.

Операционная система Fedora автоматически подключается к огромному хранилищу (репозиторию) программного обеспечения Fedora в Интернете. Через Сеть можно загрузить и установить любой из тысяч пакетов Fedora.

Хотя весь механизм управления программным обеспечением в Fedora (`yum` и `rpm`) будет подробно описан лишь в главе 10, уже можно что-нибудь установить. Откройте список приложений и запустите приложение Software (Центр приложений) (рис. 2.13).

Рис. 2.13. Скачивайте и устанавливайте программное обеспечение из огромного репозитория Fedora

Ищите нужные приложения, указав название в поле поиска (нажмите кнопку в виде лупы в левом верхнем углу) или выбрав подходящую категорию. Во всех категориях находятся пакеты приложений, отсортированные по подкатегориям.

Щелкните на значке подзорной трубы в левом верхнем углу, затем введите название нужного программного обеспечения. На странице любого приложения можно прочитать его описание. Чтобы установить приложение, нажмите кнопку Install (Установить).

Теперь, установив нужные приложения, можно еще эффективнее использовать систему. Дополнительную информацию о том, как в Fedora и Red Hat Enterprise Linux добавлять репозитории ПО и применять команды `dn`, `yum` и `rpm`, чтобы ими управлять, см. в главе 10.

Воспроизведение музыки с помощью Rhythmbox

Rhythmbox — это музыкальный плеер, установленный в операционной системе Fedora. Вы можете запустить его в интерфейсе GNOME 3 и воспроизводить музыкальные CD, подкасты и радиошоу из Интернета. Можно также прослушивать аудиофайлы в форматах WAV и Ogg Vorbis и добавлять плагины для поддержки MP3 и других аудиоформатов.

На рис. 2.14 показано окно Rhythmbox, воспроизводящего интернет-радиостанцию.

Как можно использовать Rhythmbox.

- **Радио.** Перейдите на вкладку Radio (Радио) на панели Library (Фонотека) и выберите радиостанцию из списка справа.

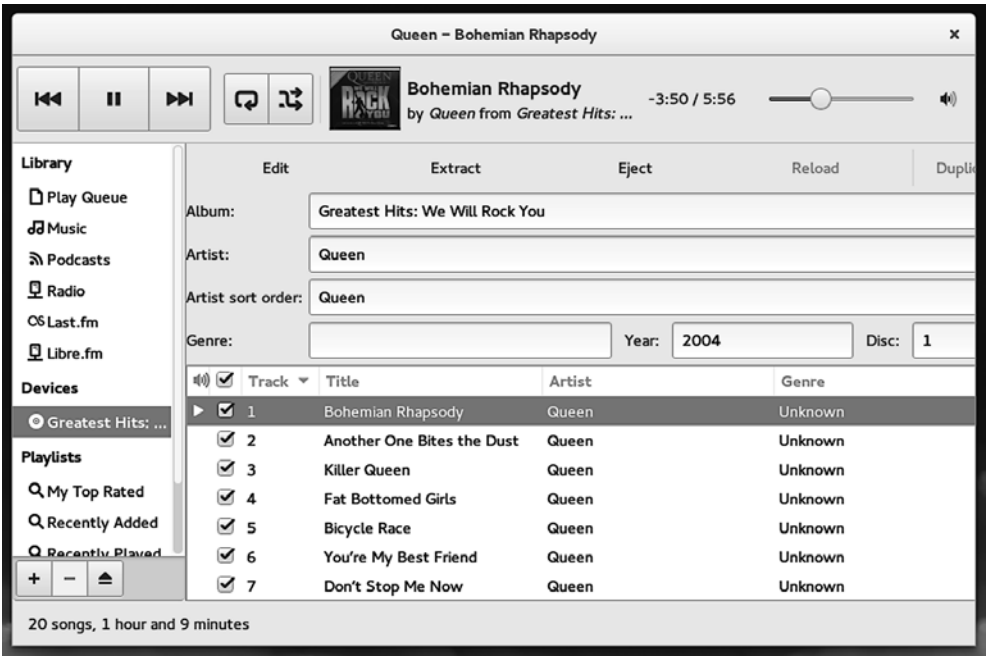


Рис. 2.14. Слушайте музыку, подкасты и интернет-радио с помощью Rhythmbox

- **Подкасты.** Найдите подкаст в Интернете и скопируйте его URL-адрес. Перейдите на вкладку Podcasts (Подкасты), в правой части окна щелкните правой кнопкой мыши и выберите пункт New Podcast Feed (Добавить ленту подкастов). Вставьте или введите URL-адрес подкаста и нажмите кнопку Find (Поиск). Ниже появится список подкастов с выбранного сайта. Дважды щелкните на нужном подкасте, чтобы прослушать его.
- **Музыкальные диски.** Вставьте оптический диск в привод компьютера и нажмите кнопку воспроизведения, когда он отобразится в окне программы Rhythmbox. Rhythmbox может копировать и записывать аудио на CD.
- **Аудиофайлы.** Программа Rhythmbox может проигрывать файлы форматов WAV и Ogg Vorbis. С помощью дополнительных плагинов можно воспроизводить множество других форматов, включая MP3. Поскольку с форматом MP3 связаны проблемы с авторскими правами, возможность воспроизведения MP3-файлов не включена в Fedora по умолчанию. В главе 10 рассказывается, как установить необходимое программное обеспечение, которого нет в репозитории вашего дистрибутива Linux.

Существуют плагины для Rhythmbox, которые подгружают из Интернета обложки дисков, информацию об исполнителях, тексты песен, а также добавляют поддержку музыкальных сервисов (например, Last.fm и Magnatune).

Отключение рабочего стола GNOME 3

После завершения работы в GNOME 3 нажмите кнопку со стрелкой вниз в правом верхнем углу. Там находится кнопка включения/выключения, которая позволяет выйти из системы или, не выходя из нее, переключиться на другую учетную запись пользователя.

Использование рабочего стола GNOME 2

Рабочий стол GNOME 2 — это стандартный интерфейс рабочего стола в Red Hat Enterprise Linux 6. Он наиболее известный, максимально стабильный, но при этом довольно скучный.

Интерфейс GNOME 2 включает в себя стандартные меню, панели, значки и рабочие места. Если у вас установлена система Red Hat Enterprise Linux, RHEL 6 или более устаревший дистрибутив Fedora или Ubuntu, то, вероятно, вы применяете именно рабочий стол GNOME 2. Далее я расскажу о нем, а также о том, как его можно улучшить. GNOME 2 включает в себя 3D-эффекты (см. подраздел «Добавление 3D-эффектов с помощью AIGLX» далее в этой главе).

Перечислю компоненты, из которых состоит рабочий стол GNOME.

- **Metacity (оконный менеджер).** Оконный менеджер для GNOME 2, устанавливаемый по умолчанию. Metacity позволяет управлять темами рабочего стола, рамками окон и элементами управления на рабочем столе.
- **Compiz (оконный менеджер).** Отображает 3D-эффекты для рабочего стола.
- **Nautilus (файловый менеджер/графическая оболочка).** При открытии папки (например, двойным щелчком кнопкой мыши на значке Home на рабочем столе) Nautilus раскрывает ее содержимое. Он может также отображать другие типы содержимого, например содержимое общих папок всех компьютеров сети (с помощью SMB).
- **Панели (запуск приложений/задач).** Панели, расположенные в верхней и нижней частях экрана, предназначены для удобного запуска приложений и управления ими, а также для работы с несколькими виртуальными рабочими столами. По умолчанию верхняя панель содержит кнопки меню (Applications (Приложения), Places (Переход) и System (Система)), значки приложений (электронная почта Evolution и браузер Firefox), переключатель рабочих мест (можно управлять четырьмя виртуальными рабочими столами) и часы. На панели появляются разного рода уведомления, если необходимо обновить программное обеспечение или обнаружена проблема. На нижней панели находятся кнопка Show Desktop (Показать рабочий стол), списки окон, корзина и переключатель рабочих мест.
- **Область рабочего стола.** Здесь расположены окна и значки. Контент можно перетаскивать между приложениями. В области рабочего стола также доступ-

ны меню рабочего стола (щелкните правой кнопкой мыши, чтобы увидеть его) и значки запущенных приложений. Значок компьютера дает доступ к приводу компакт-дисков, дисководам, файловой системе и общим сетевым ресурсам.

GNOME включает в себя также набор окон Preferences (Параметры), которые позволяют настраивать различные аспекты рабочего стола. Можно изменять фон, цвета, шрифты, сочетания клавиш и другие функции, связанные с внешним видом и поведением рабочего стола. На рис. 2.15 показано, как выглядит среда рабочего стола GNOME 2 с запущенными приложениями после авторизации пользователя.

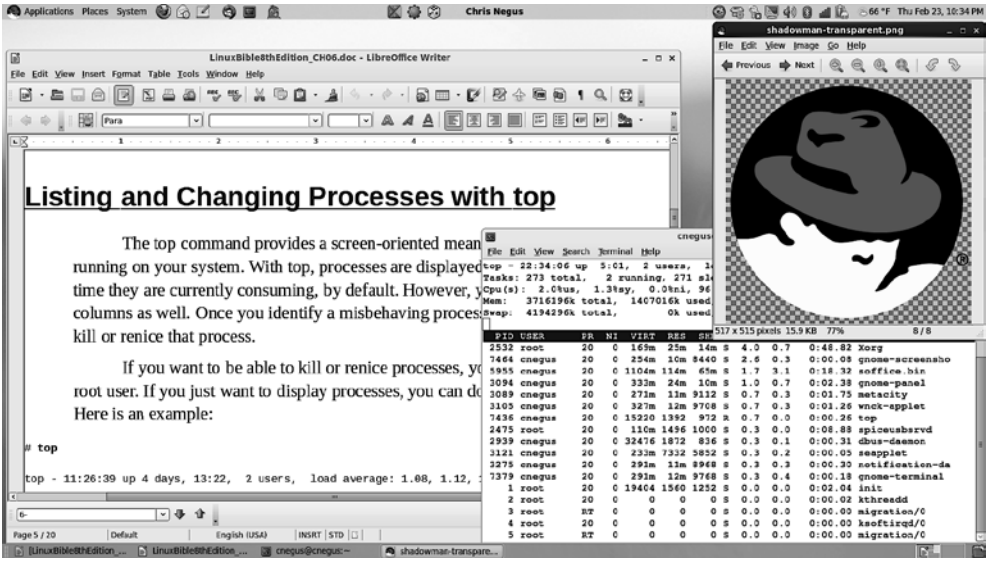


Рис. 2.15. Среда рабочего стола GNOME 2

Здесь показан рабочий стол в Red Hat Enterprise Linux. В следующих разделах мы подробно рассмотрим интерфейс GNOME 2.

Использование Metacity

Менеджер окон Metacity, по-видимому, был выбран для GNOME из-за простоты его применения. Создатель Metacity называет его «скучным менеджером окон для взрослого внутри нас» и сравнивает с другими менеджерами, говоря, что они — красочные сладкие мюсли, а Metacity — серая овсянка.

ПРИМЕЧАНИЕ

Если вас интересуют 3D-эффекты, лучше всего использовать менеджер Compiz, о котором будет рассказано далее в этой главе. В Metacity же эффектов нет, он лишь позволяет эффективно работать. Темы, цвета и оформление окон для Metacity можно изменить в настройках GNOME (описано ниже).

Основные и интересные функции Metacity — это сочетания клавиш (табл. 2.1) и переключатель рабочих мест.

Таблица 2.1. Сочетания клавиш

Сочетание клавиш	Действие
Alt+Shift+Esc	Вернуться назад, без всплывающих подсказок
Alt+Ctrl+Shift+Tab	Вернуться к предыдущей панели
Esc	Закрыть меню

С менеджером окон можно использовать и другие сочетания клавиш. Выберите команду меню System ▸ Preferences ▸ Keyboard Shortcuts (Система ▸ Параметры ▸ Комбинации клавиш клавиатуры), чтобы увидеть весь список.

- **Показывать диалог запуска программы панели.** Чтобы ввести команду запуска приложения по его имени, нажмите сочетание клавиш Alt+F2. В появившемся диалоговом окне введите название программы и нажмите клавишу Enter. Например, введите `gedit`, чтобы запустить простой графический текстовый редактор.
- **Заблокировать экран.** Чтобы заблокировать экран, нажмите сочетание клавиш Ctrl+Alt+L. Чтобы разблокировать его, понадобится ввести пароль.
- **Вызов главного меню.** Чтобы открыть меню Applications (Приложения), Places (Переход) или System (Система), нажмите сочетание клавиш Alt+F1. Затем нажимайте клавиши ↑ и ↓ для выбора из текущего меню или клавиши ← и → для выбора подменю.
- **Получить снимок экрана.** Нажмите клавишу Print Screen, чтобы сделать снимок экрана. Нажмите сочетание клавиш Alt+Print Screen, чтобы сделать снимок окна, а не всего экрана.

Еще одна интересная функция Metacity — переключатель рабочих мест. На панели GNOME 2 отображаются четыре виртуальных рабочих места в виде переключателя. С его помощью можно выполнять следующие действия.

- **Выбрать текущее рабочее место.** Переключатель содержит четыре виртуальных рабочих места. Выберите одно из них в качестве текущего.
- **Переместить окна в другое рабочее место.** Выберите любое окно на одном рабочем месте и перетащите в другое. Аналогично можно перетащить приложение из списка окон в другое рабочее место.
- **Добавить рабочие места.** Щелкните правой кнопкой мыши на переключателе рабочих мест и выберите пункт Preferences (Параметры). Здесь можно добавить рабочие места — до 32 штук.
- **Присвоить имя рабочему месту.** Щелкните правой кнопкой мыши на переключателе рабочих мест и выберите пункт Preferences (Параметры). Щелкните в диалоговом окне на имени рабочего места, чтобы изменить его.

Здесь можно просматривать и изменять информацию об элементах управления и настройках Metacity с помощью окна `gconf-editor` (введите `gconf-editor` в окне

терминала). Изменять настройки данным способом не рекомендуется — лучше использовать окно настройки GNOME 2. Однако команда `gconf-editor` — хороший способ ознакомиться с описаниями каждой функции Metacity.

В окне `gconf-editor` выберите apps ▸ metacity, а затем general, global_keybindings, keybindings_commands, window_keybindings и workspace_names. Щелкните на каждом ключе, чтобы увидеть его значение и описание.

Изменение внешнего вида GNOME

Чтобы изменить общий внешний вид рабочего стола GNOME, откройте меню System ▸ Preferences (Система ▸ Параметры) и выберите подходящий пункт.

- Theme (Тема). Для GNOME 2 доступны темы, которые меняют цвета, значки, шрифты и другие аспекты рабочего стола. В GNOME встроено несколько тем, также их можно скачать из Интернета.
- Background (Фон рабочего стола). Чтобы изменить фон, выберите из списка подходящий, а также выберите стиль. Чтобы установить собственный фон, поместите его в систему, например загрузите в папку Pictures из Интернета. Затем добавьте изображение из папки, нажав соответствующую кнопку.
- Font (Шрифт). Можно выбрать разные шрифты для приложений, документов, рабочего стола и заголовков окна.

Панели в GNOME

Панели GNOME расположены в верхней и нижней частях рабочего стола. Из них можно запускать приложения (с помощью клавиатуры или мыши), просматривать, какие программы активны, отслеживать, как работает система. Можно также изменить верхнюю и нижнюю панели, например добавив другие приложения или мониторы либо изменив расположение или поведение панели.

Щелкните правой кнопкой мыши на любом пустом месте панели, чтобы открыть контекстное меню. На рис. 2.16 показано меню верхней панели.

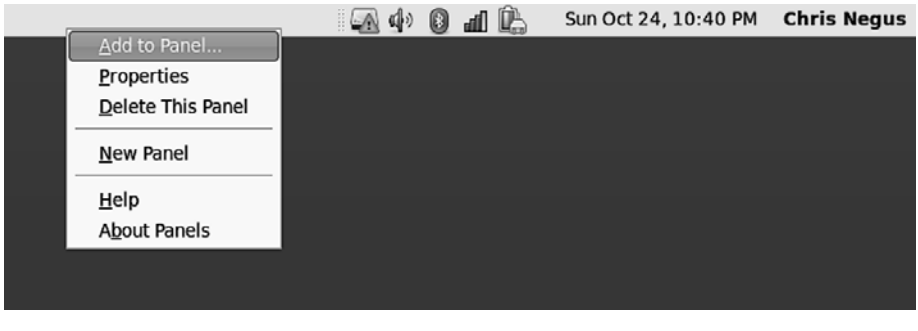


Рис. 2.16. Контекстное меню панели GNOME

На панели GNOME можно выбрать различные меню и выполнять различные команды.

- **Открывать меню:**
  - меню Applications (Приложения) дает возможность запускать различные приложения и системные утилиты;
  - меню Places (Переход) позволяет перейти в нужный каталог или требуемое устройство, например в папку рабочего стола, домашнюю папку, на съемный накопитель или в сетевое расположение;
  - меню System (Система) дает возможность изменять пользовательские и системные настройки, а также получить информацию о GNOME.
- **Выполнять команды из контекстного меню:**
  - Add to Panel (Добавить на панель) позволяет добавить на панель приложение, меню, кнопку запуска и др.;
  - Properties (Свойства) дает возможность изменить положение, размер и свойства фона панели;
  - Delete This Panel (Удалить эту панель) позволяет удалить текущую панель;
  - New Panel (Создать панель) позволяет добавить панели на рабочий стол.

  Можно также изменять элементы на панели.
- **Перенос элементов.** Чтобы переместить элемент на панели, щелкните на нем правой кнопкой мыши, в контекстном меню выберите пункт Move (Переместить) и перетащите элемент на новое место.
- **Изменение размера элементов.** Можно изменять размер некоторых элементов, например списка окон: нажав и удерживая кнопку мыши на краю, растяните список до нужного размера.
- **Использование списка всех открытых окон.** Задачи, выполняемые на рабочем столе, отображаются в списке открытых окон. Щелкните на задаче, чтобы свернуть или развернуть ее.

В следующих разделах описано, что еще можно сделать с помощью панели GNOME.

Меню Applications (Приложения) и System (Система)

Откройте меню Applications (Приложения) — и вы увидите категории приложений и системных утилит, которые можно запустить. Щелкните кнопкой мыши на приложении, которое нужно запустить. Чтобы запускать какое-либо приложение с панели, перетащите его значок из меню на панель.

В меню GNOME 2 можно добавлять различные элементы. Для этого щелкните правой кнопкой мыши на любом меню на панели и в контекстном меню выберите пункт Edit Menus (Изменить меню). Появившееся окно позволяет добавлять или удалять элементы, связанные с приложениями и системными меню. Можно также

добавить кнопки для запуска нового приложения, нажав кнопку New Item (Новый элемент) и указав имя, команду и комментарий.

Добавление апплетов

Можно запустить несколько небольших приложений, называемых апплетами, непосредственно на панели GNOME. Они могут отображать информацию, необходимую постоянно или просто для развлечения. Чтобы отобразить доступные апплеты и добавить их на панель, выполните следующие действия.

1. Щелкните правой кнопкой мыши на пустом пространстве на панели и откройте контекстное меню.
2. Выберите команду Add to Panel (Добавить на панель). Появится одноименное диалоговое окно.
3. Выберите из предложенных необходимые апплеты, к примеру часы, поиск в словаре, трекер инвестиций или прогноз погоды. Они появятся на панели.

На рис. 2.17 показаны (слева направо) апплеты в виде глазок, системного монитора, трекера инвестиций, терминала и игры «рыбка».

Рис. 2.17. Панель позволяет размещать апплеты

После установки апплета щелкните на нем правой кнопкой мыши, чтобы просмотреть доступные параметры. Например, выбрав пункт Preferences (Параметры) для апплета трекера инвестиций, можно настроить отслеживание цен на интересные акции. Чтобы переместить апплет на панели, щелкните на нем правой кнопкой мыши, выберите пункт Move (Переместить), перетащите его в нужное место и щелкните кнопкой мыши, чтобы подтвердить выбор.

Чтобы убрать апплет с панели, щелкните на нем правой кнопкой мыши и выберите команду Remove From Panel (Удалить с панели). Значок апплета исчезнет с нее. Если на панели не хватает места, можно добавить новую панель в другую часть экрана, как будет описано в следующем разделе.

Добавление панелей

Если на верхней или нижней панели не хватает места, можно добавить на рабочий стол еще несколько панелей. В GNOME 2 можно установить несколько панелей сразу: сверху, снизу, слева и справа. Чтобы добавить панели, сделайте следующее.

1. Щелкните правой кнопкой мыши на пустом пространстве панели, чтобы открыть контекстное меню.

2. Выберите пункт New Panel (Создать панель). Сбоку на экране появится новая панель.
3. Щелкните правой кнопкой мыши на пустом пространстве новой панели и выберите пункт Properties (Свойства).
4. В окне настроек выберите ориентацию панели (сверху, снизу, слева и справа).

На новую панель можно добавить апплеты или значки приложений, как и на основную. Чтобы удалить панель, щелкните на ней правой кнопкой мыши и в меню выберите пункт Delete This Panel (Удалить эту панель).

Добавление значков приложений

На панель добавлены значки браузера и приложений Office. Можно добавить и другие значки для запуска приложений с панели.

1. Щелкните правой кнопкой мыши на пустом пространстве панели, чтобы открыть контекстное меню.
2. Выберите команду Add to Panel (Добавить на панель). Появится одноименное диалоговое окно.
3. Выберите пункт Application Launcher (Запуск приложений) и нажмите кнопку Next (Далее). Появятся все категории приложений из меню Applications (Приложения) и System (Система).

Нажмите на стрелку рядом с нужной категорией, чтобы раскрыть ее, выберите приложение и нажмите кнопку Add (Добавить). Значок приложения появится на панели. Чтобы запустить приложение, щелкните кнопкой мыши на его значке на панели.

Если нужного приложения в меню Applications (Приложения) или System (Система) нет, можно самостоятельно создать кнопку запуска следующим образом.

1. Щелкните правой кнопкой мыши на пустом пространстве панели, чтобы открыть контекстное меню.
2. Выберите команду Add to Panel (Добавить на панель). Появится одноименное диалоговое окно.
3. Выберите пункт Custom Application Launcher (Пользовательская кнопка запуска) и нажмите кнопку Add (Добавить). Появится окно Create Launcher (Создать кнопку запуска).
4. Для создания кнопки запуска введите следующие данные.
   - Type (Тип). Выберите один из вариантов: Application (Приложение) (обычное GUI-приложение) или Application in Terminal (Приложение в терминале). Второй вариант подходит для символьных и `ncurses`-приложений. (Приложения, написанные с помощью библиотеки `ncurses`, запускаются в окне терминала, в них доступно также управление с помощью мыши и клавиатуры.)
   - Name (Имя). Присвойте кнопке запуска имя. (При наведении указателя мыши на кнопку будет отображаться всплывающая подсказка с ее именем.)

- Command (Команда). Укажите команду, которая будет выполняться при запуске приложения. Введите полный путь и все необходимые параметры.
- Comment (Комментарий). Введите описание приложения. (Также будет отображаться во всплывающей подсказке при наведении указателя мыши на значок.)

5. Нажмите кнопку No Icon (Нет значка), выберите одно из предложенных изображений и нажмите кнопку OK. Можно также загрузить файл значка с диска.
6. Нажмите OK.

Приложение появится на панели. Щелкните на значке, чтобы запустить приложение.

ПРИМЕЧАНИЕ

Все виды значков для приложений содержатся в каталоге /usr/share/pixmaps в форматах PNG и XPM. Если в каталоге нет нужного значка, создайте собственный (в одном из этих двух форматов) и назначьте приложению.

Добавление ящиков

Ящик (drawer) — это утилита, похожая на панели, которая позволяет создавать стеки из меню, апплетов и кнопок запуска. По сути, любой элемент, который можно добавить на панель, можно добавить и в ящик. Поместив ящик на панель GNOME, можно создать стек из нескольких апплетов и кнопок запуска, чтобы сэкономить пространство на панели. Откройте ящик, чтобы показать все находящиеся в нем апплеты и кнопки запуска.

Для добавления ящика на панель щелкните правой кнопкой мыши на пустом пространстве панели, чтобы открыть контекстное меню, и выберите пункт Add to Panel (Добавить на панель). Появится одноименное диалоговое окно. Выберите пункт Drawer (Ящик) и нажмите кнопку Add (Добавить). Новый ящик появится на панели. Щелкните на нем правой кнопкой мыши, чтобы открыть контекстное меню, и выберите пункт Add to Drawer (Добавить в ящик), чтобы добавить в ящик апплеты и кнопки запуска. Щелкните на значке ящика еще раз, чтобы закрыть его.

На рис. 2.18 показана часть панели с открытым окном ящика, в котором находятся значки игры в «рыбку», кнопки выключения компьютера и глазки.

Рис. 2.18. Кнопки запуска и апплеты, добавленные в ящик на панели в GNOME 2

Изменение параметров панели

У панелей рабочего стола можно изменить ориентацию, размер, режим сворачивания и фон. Чтобы открыть окно Panel Properties (Свойства панели), щелкните правой кнопкой мыши на пустом пространстве панели и выберите в контекстном меню пункт Properties (Свойства). В окне Panel Properties (Свойства панели) находятся элементы управления настройками панели, включая следующие.

- Orientation (Ориентация). Переместите панель в другое место на экране, выбрав один из вариантов.
- Size (Размер). Выберите размер панели, указав высоту в пикселах (по умолчанию установлено 24 пиксела).
- Expand (Расширять до предела). Установите этот флажок, чтобы панель раскрылась на всю длину, или сбросьте его, чтобы масштабировалась в зависимости от количества элементов на ней.
- AutoHide (Автоматически скрывать). Выберите, скрывать ли панель автоматически (появляется только при наведении указателя мыши на край экрана).
- Show Hide buttons (Показывать кнопки сокрытия). Выберите, скрывать или показывать кнопки сокрытия по краям панели.
- Arrows on Hide buttons (Показывать стрелки на кнопках сокрытия). Если установлен флажок Show Hide buttons (Показывать кнопки сокрытия), то можно добавить стрелки на эти кнопки.
- Background (Фон). На этой вкладке можно выбрать цвет панели, наложить на нее растровое изображение или оставить вариант по умолчанию (текущая системная тема). Чтобы использовать изображение в качестве фона панели, установите переключатель в положение Background Image (Фоновое изображение), затем выберите изображение, например, из каталога `/usr/share/backgrounds/tiles` или любого другого.

СОВЕТ

Обычно я включаю функцию автоматического сокрытия и отключаю кнопки сокрытия. Автоматическое сокрытие позволяет экономить место на рабочем столе. При наведении указателя мыши на край экрана панель автоматически всплывает, поэтому кнопки не нужны.

Добавление 3D-эффектов с помощью AIGLX

В последние годы было предпринято множество попыток внедрения 3D-эффектов на рабочий стол Linux. Дистрибутивы Ubuntu, openSUSE и Fedora использовали проект AIGLX (fedoraproject.org/wiki/RenderingProject/aiglx).

Цель проекта Accelerated Indirect GLX (AIGLX) в том, чтобы внедрить 3D-эффекты на рабочие столы Linux на постоянной основе. Проект применяет спецификацию OpenGL (opengl.org) с реализацией технологии Mesa (mesa3d.org) с открытым исходным кодом.

В настоящее время AIGLX поддерживает ограниченный набор видеокарт и реализует всего несколько 3D-эффектов, но и они уже радуют глаз.

Если видеокарта правильно настроена, то можно просто включить функцию Desktop Effects, чтобы увидеть доступные эффекты. Чтобы сделать это, выберите команду меню System ▸ Preferences ▸ Desktop Effects (Система ▸ Параметры ▸ Эффекты рабочего стола). В окне Desktop Effects (Эффекты рабочего стола) выберите плагин Compiz. (Если параметр недоступен, установите соответствующий программный пакет.)

Для включения эффектов Compiz выполните следующие действия.

- Запустите Compiz. Выключает текущий менеджер окон и включает Compiz.
- Активизируйте эффект Windows Wobble When Moved (Окна дрожат при перемещении). В этом случае строка окна заголовка дрожит, когда вы перетаскиваете его. Так же дрожат меню и другие открытые окна.
- Активизируйте эффект Workspaces on a Cube (Рабочие места на кубе). Перетащите окно с рабочего стола вправо или влево, и он будет вращаться, как куб, причем каждое из рабочих мест будет отображаться как сторона куба. Перетащите нужное окно на рабочее место, чтобы открыть его. Внизу на панели находится переключатель рабочих мест, который позволяет вращать куб.

Интересные эффекты возникают при переключении между окнами с использованием сочетания клавиш Alt+Tab. При его нажатии миниатюра каждого окна прокручивается по экрану.

На рис. 2.19 показан пример рабочего стола с включенными эффектами Compiz. Здесь представлено окно браузера, перемещаемое из одного рабочего пространства в другое в виде куба.

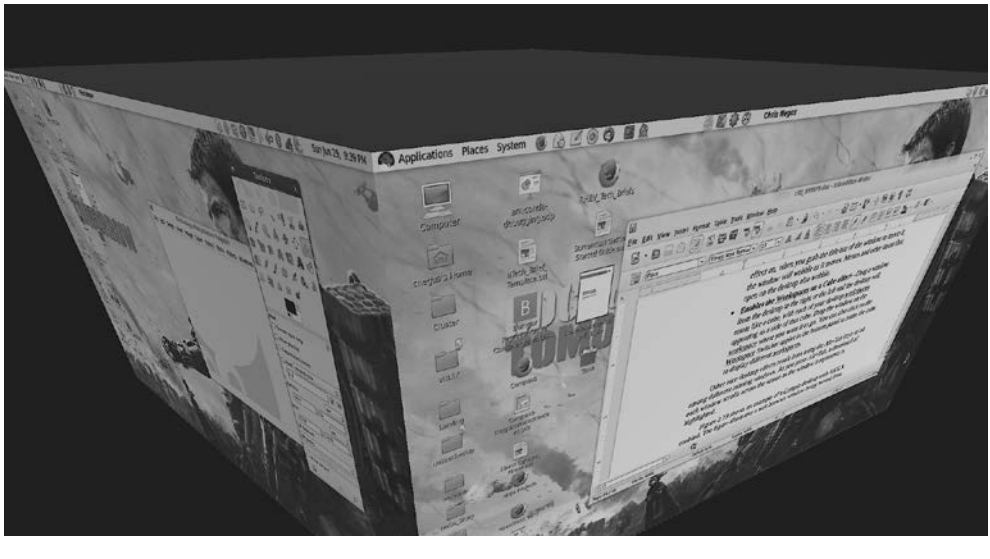


Рис. 2.19. Вращайте куб с эффектами AIGLX

Далее приведены интересные эффекты AIGLX.

- **Вращение куба.** Нажав и удерживая сочетание клавиш Ctrl+Alt, нажимайте клавиши ← и →. Куб рабочего стола будет вращаться в направлении каждого последующего рабочего места (вперед или назад).
- **Медленное вращение куба.** Нажав и удерживая сочетание клавиш Ctrl+Alt, а также левую кнопку мыши, перемещайте указатель мыши на экране. Куб начнет медленно вращаться между рабочими местами.
- **Масштабирование и разделение окон.** Если рабочий стол заполнен, нажав и удерживая сочетание клавиш Ctrl+Alt, нажмите клавишу ↑. Окна переместятся на отдельный рабочий стол. Все еще удерживая нажатым сочетание клавиш Ctrl+Alt, используйте клавиши со стрелками, чтобы выделить нужное окно, и отпустите клавиши, чтобы его открыть.
- **Переключение между окнами.** Нажав и удерживая клавишу Alt, нажмите клавишу Tab. На полосе в середине экрана появятся уменьшенные версии всех окон, текущее окно — в центре. Все еще удерживая нажатой клавишу Alt, нажмите клавишу Tab или сочетание клавиш Shift+Tab для перемещения вперед или назад между окнами. Отпустите клавиши на нужном окне.
- **Масштабирование и разделение рабочих мест.** Нажав и удерживая сочетание клавиш Ctrl+Alt, нажмите клавишу ↓, чтобы увидеть миниатюры рабочих мест. Удерживая нажатым сочетание клавиш Ctrl+Alt, нажимайте клавиши со стрелками ← и → для перемещения между рабочими местами. Отпустите клавиши на нужном рабочем месте.
- **Перемещение активного окна в следующее рабочее место.** Нажав и удерживая сочетание клавиш Ctrl+Alt+Shift, нажмите клавишу ← или →. Слева или справа соответственно на активном рабочем столе появится рабочее место.
- **Перемещение окна.** Нажав и удерживая левую кнопку мыши на заголовке окна, нажмите клавишу ←, ↑, ↓ или → для перемещения активного окна по экрану.

Если эффекты надоели, можно отключить их и вернуться к использованию Metacity в качестве оконного менеджера. Снова выберите команду меню System ▸ Preferences ▸ Desktop Effects (Система ▸ Параметры ▸ Эффекты рабочего стола) и отключите эффекты рабочего стола.

Если видеокарта поддерживает эффекты, но они не включаются, убедитесь, что X-сервер настроен правильно. В частности, убедитесь, что файл `/etc/X11/xorg.conf` содержит корректные настройки, а файлы `dri` и `glx` загружены в раздел модулей. Кроме того, добавьте раздел расширений в любое место файла (обычно в конце) следующим образом:

```
Section "extensions"
 Option "Composite"
EndSection
```

Другой вариант — добавить в файл `/etc/X11/xorg.conf`, в раздел `Device`, следующую строку:

```
Option "XAANoOffscreenPixmaps"
```

Функция `XAANoOffscreenPixmaps` повышает производительность. Проверьте файл `/var/log/Xorg.log`, чтобы убедиться, что функции DRI и AIGLX запущены правильно. Сообщения в этом файле журнала помогут исправить и другие проблемы.

Резюме

Среда рабочего стола GNOME стала основной для многих систем Linux, включая Fedora и RHEL. Рабочий стол GNOME 3 (задействуется в Fedora, RHEL 7 и RHEL 8) — это современный элегантный рабочий стол, разработанный в соответствии с типами интерфейсов для многих современных мобильных устройств. Рабочий стол GNOME 2 (применяется в RHEL 6) — более традиционный вариант рабочего стола.

Помимо рабочих столов GNOME, вы можете попробовать и другие популярные и полезные среды рабочего стола. В среде K Desktop Environment (KDE) гораздо больше наворотов, чем в GNOME, и она используется по умолчанию в нескольких дистрибутивах Linux. На нетбуках и в «живых» дистрибутивах иногда применяют облегченные оболочки LXDE или Xfce.

Теперь, когда мы рассмотрели возможности рабочего стола Linux, пришло время начать изучать более профессиональные административные интерфейсы. В главе 3 мы обсудим интерфейс командного интерпретатора в Linux.

Упражнения

Проверьте свои навыки использования рабочего стола GNOME, выполнив упражнения, приведенные далее. Для этого можете задействовать как GNOME 2.x (Red Hat Enterprise Linux вплоть до RHEL 6), так и GNOME 3.x (Fedora 16 или Ubuntu 11.10 и ранние версии). Если затрудняетесь с решением задания для GNOME 2 и GNOME 3, воспользуйтесь ответами к упражнениям, которые даны в приложении Б.

1. Установите систему Linux с рабочим столом GNOME 2 или GNOME 3. Запустите систему и войдите в учетную запись.
2. Запустите браузер Firefox и перейдите на домашнюю страницу GNOME (gnome.org).
3. Выберите понравившийся фон для рабочего стола с сайта gnome-look.org, скачайте его в папку Изображения и установите в качестве фона.

4. Запустите файловый менеджер и переместите его на второе рабочее место.
5. Найдите скачанное изображение и откройте его в любой программе для просмотра изображений.
6. Перемещайтесь между рабочим местом с Firefox и тем, где открыт файловый менеджер.
7. Откройте список установленных приложений и выберите программу для просмотра изображений. Для этого как можно меньше нажимайте клавиши со стрелками или щелкайте кнопкой мыши.
8. Измените вид окон в текущем рабочем месте на миниатюры. Выберите любое окно из списка.
9. Используя только клавиатуру, запустите музыкальный плеер.
10. Сделайте снимок экрана, используя только клавиатуру.

Часть II

Опытный пользователь Linux

В этой части

- Глава 3. Использование оболочки.
- Глава 4. Файловая система.
- Глава 5. Работа с текстовыми файлами.
- Глава 6. Управление активными процессами.
- Глава 7. Простейшие скрипты оболочки.

3 Использование оболочки

В этой главе

- Что такое оболочка Linux.
- Использование оболочки в консолях и терминалах.
- Применение команд.
- Использование истории команд и автозаполнения.
- Подключение и расширение команд.
- Переменные и псевдонимы.
- Постоянные настройки оболочки.
- Man-страницы и прочая документация.

До того как значки и окна заполонили экраны, для взаимодействия с большинством компьютеров использовались команды. В системах UNIX, на основе которых создана операционная система Linux, программа для взаимодействия и управления командами, называлась *командной оболочкой*.

Независимо от того, какой дистрибутив Linux применяется, командный интерпретатор (оболочка) доступен всегда. Он позволяет создавать файлы скриптов, запускать программы, работать с файловыми системами, компилировать компьютерный код и управлять системой. Хотя командная оболочка не так интуитивно понятна, как привычные графические интерфейсы (GUI), большинство экспертов Linux считают ее гораздо мощнее, чем GUI. Командные интерпретаторы существуют давно, и многие дополнительные функции, которые недоступны через графический интерфейс, можно активизировать с помощью команды.

Оболочка Linux, о которой пойдет речь в этой главе, называется *оболочкой bash*, что расшифровывается как Bourne Again Shell. Такое название она получила благодаря совместимости с одной из самых ранних оболочек UNIX — обо-

лочкой Bourne (названа в честь ее создателя Стивена Борна и представлена командой `sh`).

Оболочка `bash` входит в большинство дистрибутивов и считается стандартной, существуют и другие оболочки, включая C (`csh`), которую любят пользователи BSD UNIX, и Korn (`ksh`), популярную у пользователей UNIX System V. В Ubuntu во время загрузки по умолчанию используется оболочка `dash`, так как она работает быстрее, чем `bash`. Существуют также оболочка `tcsh` (улучшенная C) и оболочка `ash` (тоже похожа на Bourne shell).

Велика вероятность того, что ваш дистрибутив Linux содержит более одной оболочки. Однако в этой главе мы рассмотрим именно оболочку `bash`. Это связано с тем, что дистрибутивы Linux, представленные в этой книге, Fedora, Ubuntu и Red Hat Enterprise Linux, по умолчанию задействуют оболочку `bash` в окне терминала.

Приведу несколько основных преимуществ, которые дает умение работать с оболочкой.

- *Это позволяет научиться ориентироваться в любой Linux- или UNIX-подобной системе*. Например, я могу войти на личный веб-сервер Red Hat Enterprise Linux, домашний мультимедийный сервер, домашний маршрутизатор или компьютер своей жены и использовать любую из этих компьютерных систем с помощью оболочки. Я даже могу войти в Android-систему телефона и запускать в ней команды. Все эти устройства работают под управлением Linux или аналогичных систем.
- *Специальные функции оболочки позволяют собирать входящие данные и распределять их между командами и файловыми системами Linux*. Чтобы сэкономить время на вводе текста, можно находить, изменять и повторять команды из истории оболочки. Многие опытные пользователи практически не применяют графические интерфейсы, выполняя большую часть работы из оболочки.
- *Это позволяет собирать команды в файл с помощью функционального программирования, например условных конструкций, циклов и других операторов, чтобы быстро выполнять сложные операции и не набирать команды снова и снова*. Программы, состоящие из команд, которые хранятся в файле и запускаются из него, называются *скриптами оболочки*. Многие системные администраторы Linux используют скрипты оболочки для автоматизации таких задач, как резервное копирование данных, мониторинг файлов логов или проверка работоспособности системы.

Оболочка — это интерпретатор командного языка. Поработав в операционных системах Microsoft, можно заметить, что в Linux оболочки действуют аналогично, но в целом намного мощнее, чем интерпретатор PowerShell. Можно с удовольствием использовать графический интерфейс Linux, однако по мере того, как ваши знания о Linux будут расширяться, вам наверняка в какой-то момент придется применить оболочку, чтобы отследить проблему или администрировать некоторые функции.

Сначала использование оболочки может показаться совершенно непонятным, но, имея правильные инструкции, вы быстро изучите многие из наиболее важных ее функций. Эта глава — руководство по работе с системными командами, процессами и файловой системой Linux из командной оболочки. В ней описывается среда оболочки и то, каким образом можно адаптировать ее к потребностям пользователя.

Оболочка и терминал

Есть несколько способов доступа к командному интерпретатору в Linux. Три наиболее распространенных — это командная строка, окно терминала и виртуальная консоль, о которых вы узнаете подробнее в следующих разделах.

Запустите систему Linux. На экране появится либо графический интерфейс входа в систему, либо текст, приведенный далее:

```
Red Hat Enterprise Linux Server release 8.0 (Ootpa)
Kernel 4.18.0-42.el8.x86_64 on an X86
mylinuxhost login:
```

В любом случае необходимо войти в систему под учетной записью пользователя. Если при входе в систему появился только текст, переходите к разделу «Использование командной строки» данной главы. Если при входе появился графический экран, прочитайте подраздел «Использование терминала», чтобы узнать, как открыть интерпретатор с рабочего стола. Доступ к большему количеству оболочек описан в разделе «Использование виртуальных консолей» в этой главе.

Использование командной строки

Если у вашей системы Linux нет графического интерфейса или в данный момент он не работает, то после входа в систему на экране, скорее всего, появится текстовое приглашение в интерпретатор. Ввод команд из командной строки, вероятно, и будет основным средством работы в системе Linux.

По умолчанию для обычного пользователя приглашение — это знак доллара:

```
$
```

Приглашение для суперпользователя — знак фунта (также называется *знаком решетки* или *октоторпом*):

```
#
```

В большинстве систем Linux перед знаками `$` и `#` указаны имена пользователя, системы и текущего каталога. Например, запрос на вход для пользователя с именем `jake`, именем системы `pine` и рабочим каталогом `/usr/share/` будет выглядеть следующим образом:

```
[jake@pine share]$
```

Можно изменить текст приглашения на любые другие символы и даже отразить фрагменты информации о вашей системе. Например, в качестве приглашения использовать текущий рабочий каталог, дату, имя локального компьютера или любую иную строку символов. Чтобы настроить приглашение, перейдите в подраздел «Настройка приглашения» далее в этой главе.

В оболочке доступно огромное количество функций, можно начать с ввода простых команд. Выполните некоторые из тех команд, что рассмотрены в этой главе, чтобы ознакомиться с текущей средой оболочки.

В приведенных далее примерах символы доллара (`$`) и фунта (`#`) указывают на приглашение командной строки. Символ `$` говорит о том, что команда может быть запущена любым пользователем, а символ `#` обычно означает, что необходимо запустить команду от имени суперпользователя (то есть многие административные инструменты требуют разрешения суперпользователя для запуска). После приглашения вы должны ввести команду, а затем нажать клавишу Enter.

ПРИМЕЧАНИЕ

Хотя символ # указывает, что команда должна выполняться от имени суперпользователя, не нужно входить в систему, чтобы запустить такую команду. Наиболее распространенный способ запуска команды от имени суперпользователя — добавление ключевого слова sudo. Дополнительную информацию о команде sudo см. в главе 8.

Использование терминала

Запустив графический интерфейс операционной системы, вы можете открыть программу эмулятора терминала, иногда называемую окном терминала, чтобы запустить командный интерпретатор. Большинство дистрибутивов Linux позволяют легко запускать интерпретатор из графического интерфейса пользователя. Приведу два распространенных способа запуска окна терминала с рабочего стола Linux.

- **Щелкните правой кнопкой мыши на рабочем столе.** Появится контекстное меню с командой наподобие Open in Terminal (Открыть терминал), Shells (Оболочки), New Terminal (Новый терминал), Terminal Window (Окно терминала), Xterm и т. п. (В некоторых дистрибутивах этой функции нет.)
- **Выберите соответствующий пункт в меню на панели.** Во многих системах Linux в верхней или нижней части экрана есть панель, на которой можно запускать приложения. Например, в некоторых системах с рабочим столом GNOME 2 выберите команду меню Applications ▸ System Tools ▸ Terminal (Приложения ▸ Системные утилиты ▸ Терминал), чтобы открыть окно терминала. В GNOME 3 щелкните кнопкой мыши на меню Activities (Обзор), введите в строку поиска `Terminal` и нажмите клавишу Enter.

Во всех случаях команды вводятся так же, как и из интерпретатора без графического интерфейса. В Linux доступны различные эмуляторы терминалов. В Fedora, Red Hat Enterprise Linux (RHEL) и других дистрибутивах Linux с рабочим столом

GNOME окно эмулятора терминала по умолчанию — это терминал GNOME (запускается командой `gnome-terminal`).

Терминал поддерживает множество функций, выходящих за рамки возможностей обычной базовой оболочки. Например, можно вставлять скопированный текст в окно терминала GNOME или копировать текст из него, менять шрифт, устанавливать заголовки, выбирать цвета или изображения для фона, а также задавать, сколько текста следует показывать при прокрутке экрана.

Чтобы попробовать применить некоторые функции терминала, запустите систему Fedora или RHEL и войдите на рабочий стол. Затем следуйте инструкции.

1. Перейдите в меню Application ▸ Utilities ▸ Terminal (Приложения ▸ Утилиты ▸ Терминал) или введите в строку поиска в меню Activities (Обзор) текст `Terminal`. Откроется окно терминала.
2. В меню окна терминала выберите команду меню Preferences (Параметры) или Profile Preferences (Параметры профиля).
3. На вкладке General (Основное) или на вкладке текущего профиля (в зависимости от версии GNOME) установите флажок Custom font (Пользовательский шрифт).
4. Выберите подходящий шрифт и его размер, а затем нажмите кнопку Select (Выбрать). В окне терминала появится новый шрифт.
5. Снимите флажок Custom font (Пользовательский шрифт), чтобы вернуться к шрифту по умолчанию.
6. На вкладке Colors (Цвета) снимите флажок Use colors from system theme (Использовать цвета из системной темы), чтобы выбрать свой цвет для текста и фона.
7. Установите флажок Use colors from system theme (Использовать цвета из системной темы), чтобы вернуться к цветам по умолчанию.
8. Перейдите в окно профиля. Существуют и другие функции, с которыми можно поэкспериментировать, например установить, сколько строк прокручиваемых данных будет показано.
9. После выполнения настроек закройте окно профиля. Теперь можно использовать окно терминала.

Если применяется графический рабочий стол, то доступ к интерпретатору легче всего получать именно из окна терминала.

Использование виртуальных консолей

Большинство систем Linux с интерфейсом рабочего стола задействуют несколько виртуальных консолей, работающих на компьютере. Виртуальные консоли позволяют одновременно открывать несколько сеансов интерпретатора вместе с одновременным задействованием графического интерфейса.

Между виртуальными консолями можно переключаться, удерживая клавиши Ctrl и Alt и нажимая одну из функциональных клавиш F1–F6. Например, в дистрибутиве Fedora нажмите сочетание клавиш Ctrl+Alt+F1 (или F2, F3, F4, F4 и F6 в большинстве систем Linux), чтобы отобразить одну из семи виртуальных консолей. Графический интерфейс обычно имеется в одной из первых двух виртуальных консолей, а остальные шесть — текстовые.

Чтобы вернуться к графическому интерфейсу (если он запущен), нажмите сочетание клавиш Ctrl+Alt+F1. В некоторых системах графический интерфейс может быть расположен в другой виртуальной консоли, например во второй (Ctrl+Alt+F2). Более новые системы, такие как Fedora 33, постоянно запускают `gdm` (экран входа в систему) на `tty1`, чтобы разрешить несколько одновременных сеансов интерфейса: `gdm` находится на `tty1`, первый рабочий стол запускается на `tty2`, второй рабочий стол — на `tty3` и т. д.

Попробуйте. Нажмите и удерживайте сочетание клавиш Ctrl+Alt и нажмите клавишу F3. Появится текст приглашения входа в систему. Введите имя пользователя и пароль. Примените несколько команд. По окончании введите `exit`, чтобы покинуть командный интерпретатор, а затем нажмите сочетание клавиш Ctrl+Alt+F1 или Ctrl+Alt+F2, чтобы вернуться в графический интерфейс. Можно перемещаться между этими консолями сколько угодно.

Выбор командного интерпретатора

В большинстве систем Linux оболочкой по умолчанию является утилита `bash`. Чтобы узнать, какая у вас оболочка по умолчанию, введите следующие команды:

```
$ who am i
chris    pts/0       2019-10-21 22:45 (:0.0)
$ grep chris /etc/passwd
chris:x:13597:13597:Chris Negus:/home/chris:/bin/bash
```

Обратите внимание: примеры командной строки, показанные здесь и во всей книге, — это команды с выходными данными. Когда команда завершится, вы снова увидите приглашение командной строки.

Команда `who am i` отображает имя пользователя, а команда `grep` (замените `chris` своим именем) — определение учетной записи в файле `/etc/passwd`. Последняя часть в этой записи показывает, что оболочка `bash` (`/bin/bash`) является оболочкой по умолчанию (той, которая запускается при входе в систему или при открытии окна терминала).

Вполне возможно, хотя и маловероятно, что у вас будет другой набор оболочек по умолчанию. Чтобы испытать другой интерпретатор, просто введите имя оболочки, например `ksh`, `tcsh`, `csh`, `sh`, `dash` или другие, если они установлены. Попробуйте ввести несколько команд в другой оболочке и наберите `exit` по окончании, чтобы вернуться в интерпретатор `bash`.

Для чего использовать различные командные интерпретаторы?

- Вы уже привыкли применять системы UNIX System V (часто с `ksh` по умолчанию) или Sun Microsystems и другие UNIX-подобные дистрибутивы Berkeley (часто с `csh` по умолчанию), и вам удобнее работать в этих оболочках.
- Вы хотите запускать скрипты оболочки, созданные для конкретной среды интерпретатора, и ее нужно протестировать или использовать из текущей оболочки.
- Вы просто предпочитаете функции конкретного командного интерпретатора. Например, некоторые из сообществ Linux, в которые я вхожу, выбирают `ksh`, а не `bash`, потому что им не нравится, как в `bash` используются псевдонимы.

Большинство пользователей Linux отдают предпочтение определенной оболочке, но если вы знаете, как применять одну из них, то сможете быстро изучить любую другую, иногда обращаясь к руководству на соответствующей man-странице (например, введите `man bash`). Справочные man-страницы (описанные далее, в разделе «Информация о командах») содержат документацию по командам, форматам файлов и другим компонентам Linux. Большинство людей используют оболочку `bash` только потому, что у них нет причин работать с другим интерпретатором. Оставшаяся часть этой главы посвящена оболочке `bash`.

`bash` включает в себя функции, разработанные для интерпретаторов `ksh` и `sh` в ранних системах UNIX, а также некоторые функции `csh`. Интерпретатор `bash` является оболочкой входа по умолчанию в большинстве систем Linux, за исключением некоторых специализированных (например, работающих на встроенных устройствах), которым необходим меньший по использованию памяти и выполняемым функциям интерпретатор. Большинство примеров, приведенных в книге, выполнено в интерпретаторе `bash`.

СОВЕТ

Знание интерпретатора bash полезно не только потому, что он применяется по умолчанию в большинстве систем, но и потому, что именно он включен в бóльшую часть экзаменов по Linux.

Запуск команд

Самый простой способ выполнить команду — ввести ее имя в интерпретаторе. С рабочего стола откройте окно терминала. Введите:

```
$ date
Thu Jun 29 08:14:53 EDT 2019
```

Команда `date` без каких-либо параметров или аргументов вызывает отображение текущих дня, месяца, времени, часового пояса и года, как показано ранее.

Попробуйте другие команды:

```
$ pwd
/home/chris
$ hostname
mydesktop
$ ls
Desktop    Downloads  Pictures  Templates
Documents  Music      Public    Videos
```

Команда `pwd` отображает текущий каталог. Команда `hostname` показывает имя компьютера. Команда `ls` отображает список файлов и каталогов, находящихся в текущем каталоге. Хотя многие команды можно выполнить, набрав только их имена, стоит вводить после команды дополнительные символы, чтобы изменить ее поведение. Символы и слова, которые можно добавить к команде, называются *параметрами* и *аргументами*.

Синтаксис команд

У большинства команд есть один или несколько *параметров*, с помощью которых можно изменить их поведение. Параметры обычно состоят из одной буквы, перед которой стоит дефис. Чтобы применять более одного параметра одновременно, можно сгруппировать однобуквенные параметры или поставить дефис перед каждым. Например, следующие два варианта использования параметров для команды `ls` одинаковы:

```
$ ls -l -a -t
$ ls -lat
```

В обоих случаях команда `ls` выполняется с параметрами `-l` (длинный список), `-a` (показать скрытые файлы с точкой) и `-t` (сортировать по времени).

В некоторые команды включаются параметры в виде целого слова. Чтобы команда использовала в качестве параметра целое слово, перед ним ставится двойной дефис (`--`). Например, чтобы вызывать помощь для многих команд, введите в командной строке `--help`. Без двойного дефиса буквы `h`, `e`, `l` и `p` будут интерпретироваться как отдельные параметры. Существуют некоторые команды, у которых срабатывает один дефис перед словом-параметром, а не два, но в большинстве команд все же применяют двойной дефис.

ПРИМЕЧАНИЕ

Используйте параметр -help, чтобы просмотреть параметры и аргументы большинства команд. Например, введите hostname -help.

Многие команды также поддерживают аргументы, располагающиеся после ввода определенных параметров или в конце командной строки. *Аргумент* — это

дополнительная информация, например имя файла, каталог, имя пользователя, устройство или другой элемент, который указывает команде, что делать. Например, команда `cat /etc/passwd` отображает содержимое файла `/etc/passwd` на экране. Часть `/etc/passwd` и есть аргумент. Обычно в командной строке можно использовать любое количество аргументов, ограниченное только общим количеством символов, разрешенных в командной строке. Иногда аргумент связан с параметром. В этом случае он должен стоять сразу за параметром. При однобуквенных параметрах аргумент обычно отделяется пробелом. Если параметр — это целое слово, то аргумент стоит за знаком равенства (`=`), например:

```
$ ls --hide=Desktop
Documents  Music     Public    Videos
Downloads  Pictures  Templates
```

Здесь параметр `--hide` указывает команде `ls` не отображать файл или каталог с именем `Desktop` при перечислении содержимого каталога. Обратите внимание на то, что знак равенства стоит сразу за параметром (без пробела), а затем указывается аргумент (опять же без пробела).

Вот пример однобуквенного параметра, за которым следует аргумент:

```
$ tar -cvf backup.tar /home/chris
```

Здесь параметры создают (`c`) файл (`f`) с именем `backup.tar`, который включает в себя все содержимое каталога `home/chris` и его подкаталогов и показывает подробные сообщения (`v`) при создании резервной копии. Поскольку текст `backup.tar` является аргументом для параметра `f`, аргумент `backup.tar` должен располагаться сразу за параметром.

Примените следующие команды. Посмотрите, как они ведут себя с различными параметрами:

```
$ ls
Desktop  Documents  Downloads  Music  Pictures  Public  Templates
Videos
$ ls -a
.              Desktop     .gnome2_private  .lesshst  Public
..             Documents   .gnote           .local    Templates
.bash_history  Downloads   .gnupg           .mozilla  Videos
.bash_logout   .emacs      .gstreamer-0.10  Music     .xsession-errors
.bash_profile  .esd_auth   .gtk-bookmarks   Pictures  .zshrc
.bashrc        .fsync.log  .gvfs            Pictures
$ uname
Linux
$ uname -a
Linux mydesktop 5.3.7-301.fc31.x86_64 #1 SMP Mon Oct 21 19:18:58 UTC
     2019 x86_64 x86_64 x86_64 GNU/Linux
$ date
Wed 04 Mar 2020 09:06:25 PM EST
$ date +'%d/%m/%y'
04/03/20
$ date +'%A, %B %d, %Y'
Wednesday, March 04, 2020
```

Сама по себе команда `ls` выводит все обычные файлы и каталоги в текущем каталоге. Добавив параметр `-a`, можно отобразить скрытые файлы, находящиеся в каталоге (те, которые начинаются с точки). Команда `uname` выводит тип используемой системы (Linux). При добавлении параметра `-a` отобразятся также имя хоста, выпуск ядра и версия ядра.

Для команды `date` существует несколько особых параметров. Сама по себе она просто отображает текущие день, дату и время, как показано ранее. Однако будучи примененной с форматирующим параметром `+`, команда `date` поддерживает специальную функцию, которая позволяет отображать дату в разных форматах. Введите `date --help`, чтобы узнать больше о доступных форматах.

Попробуйте применить команды `id` и `who`, чтобы получить представление о текущей среде Linux, как будет описано далее.

Когда вы входите в Linux, система определяет вашу личность, то есть имя пользователя, имя группы, идентификатор пользователя (user ID) и идентификатор группы (group ID). Linux также отслеживает весь сеанс в системе: когда вы вошли в систему, как долго она бездействовала и откуда вы вошли.

Чтобы получить информацию о своей личности Linux, используйте команду `id`:

```
$ id
uid=1000(chris) gid=1000(chris) groups=1005(sales), 7(lp)
```

В данном примере имя пользователя — `chris`, оно представлено числовым идентификатором пользователя (`uid`) `1000`. Первичная группа также называется `chris` и имеет идентификатор группы (`gid`) `1000`. В Fedora и Red Hat Enterprise Linux чаще всего имя пользователя и имя группы одинаковые. Пользователь `chris` принадлежит также к другим группам, которые называются `sales` (`gid 1000`) и `lp` (`gid 7`). Данные имена и числа — это сведения, говорящие о том, что пользователю `chris` разрешен доступ к ресурсам компьютера.

ПРИМЕЧАНИЕ

Дистрибутивы Linux с включенной функцией Security Enhanced Linux (SELinux), такие как Fedora и RHEL, показывают дополнительную информацию в конце вывода после команды id. Это может выглядеть примерно так:

```
context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

SELinux обеспечивает жесткую блокировку для обеспечения безопасности системы Linux. Информация о SELinux представлена в главе 24.

Сведения о текущем сеансе входа в систему отображается с помощью команды `who`. В примере далее параметр `-u` добавляет информацию о времени простоя и ID процесса, а параметр `-H` запрашивает вывод заголовка:

```
$ who -uH
NAME      LINE     TIME             IDLE     PID     COMMENT
chris     tty1     Jan 13 20:57     .        2019
```

Вывод команды `who` показывает, что пользователь `chris` вошел в систему на `tty1` (является первой виртуальной консолью на мониторе, подключенном

к компьютеру), а сеанс входа начался 13 января в 20:57. Время IDLE показывает, как долго интерпретатор был открыт без ввода какой-либо команды (точка указывает на активность в данный момент). PID показывает идентификатор процесса входа пользователя. COMMENT будет показывать имя удаленного компьютера, с которого пользователь вошел в систему, если он сделал это с другого компьютера в сети, или имя локального дисплея X, если пользователь использует окно терминала (например, :0.0).

Местоположение команд

Теперь, когда вы попробовали ввести некоторые команды, вас может заинтересовать, где хранятся все команды, доступные для конкретного командного интерпретатора. Чтобы найти введенные команды, интерпретатор ищет так называемый *путь*. Для команд, которые не находятся в вашем пути, можно ввести полный идентификатор их местоположения.

Если известен каталог с нужной командой, один из способов ее запуска — ввести полный (или точный) путь к ней. Например, чтобы запустить команду date из каталога /bin, введите:

```
$ /bin/date
```

Конечно, это может быть не совсем удобно, особенно если команда находится в каталоге с длинным путем. Лучший способ — это хранить команды в хорошо известных каталогах, а затем добавлять эти каталоги в PATH (переменная окружения интерпретатора). Путь состоит из списка каталогов, которые последовательно проверяются для вводимых команд. Чтобы просмотреть текущий путь, введите:

```
$ echo $PATH
/usr/local/bin:/usr/bin:/bin:/usr/local/sbin:/usr/sbin:/sbin:/home/chris/bin
```

В результате отображается общий путь по умолчанию для обычного пользователя Linux. Каталоги в списке путей разделяются двоеточиями. Большинство команд для обычного пользователя Linux находятся в каталогах /bin, /usr/bin или /usr/local/bin. Каталоги /sbin и /usr/sbin содержат административные команды (некоторые системы Linux не добавляют эти каталоги в пути обычных пользователей). Последний показанный каталог — это bin в каталоге home пользователя (/home/chris/bin).

СОВЕТ

Если вы хотите добавить собственные команды или скрипты оболочки, поместите их в каталог bin в своем каталоге home (например, /home/chris/bin для пользователя с именем chris). В одних системах Linux этот каталог автоматически добавляется в ваш путь, а в других может потребоваться создать этот каталог или добавить его в переменную PATH. Таким образом, если вы добавляете команду в каталог bin с разрешением исполнения, ее можно будет использовать, просто введя имя в командной строке интерпретатора. Чтобы сделать команды доступными для всех пользователей, добавьте их в каталог /usr/local/bin.

В отличие от некоторых других операционных систем, Linux по умолчанию не проверяет текущий каталог на наличие исполняемых файлов перед поиском пути. Она сразу же начинает поиск пути, и исполняемые файлы в текущем каталоге запускаются только в том случае, если находятся в переменной PATH или вы указываете точное (например, /home/chris/scriptx.sh) или относительное (например, ./scriptx.sh) местоположение.

Порядок каталогов в пути очень важен. Каталоги сверяются слева направо. Например, если команда foo расположена как в каталоге /usr/bin, так и в каталоге /bin, выполняется команда именно в каталоге /usr/bin. Чтобы запустить другую команду foo, нужно либо ввести полный путь к команде, либо изменить переменную PATH. (Изменение пути (PATH) и добавление к нему каталогов описано далее в этой главе.)

Не все выполняемые команды находятся в каталогах переменной PATH. Некоторые из них встроены в командный интерпретатор. Другие можно заменить, создав псевдонимы, определяющие любые команды и параметры, которые нужно запустить. Существуют также способы определения функции, состоящей из сохраненной серии команд. Вот так выглядит порядок, в котором интерпретатор проверяет наличие вводимых команд.

1. **Псевдонимы (ярлыки).** Это имена, заданные командой alias, которые представляют определенную команду и набор параметров. Введите в строку alias, чтобы увидеть, какие псевдонимы установлены. Псевдонимы позволяют присвоить короткое имя длинной сложной команде. (Далее в этой главе описано, как создавать псевдонимы.)
2. **Зарезервированные (ключевые) слова.** Слова, которые используются интерпретатором для определенных целей. Многие из них можно применять в программировании, например, do, while, case и else. (Подробнее о некоторых зарезервированных словах см. в главе 7.)
3. **Функция.** Это набор команд, которые выполняются вместе в текущем интерпретаторе.
4. **Встроенная команда.** Это команда, встроенная в оболочку. В файловой системе отсутствует представление таких команд. Некоторые наиболее распространенные команды — это, например, cd (изменение каталогов), echo (вывод текста на экран), exit (выход из интерпретатора), fg (вывод на передний план команды, выполняемой в фоновом режиме), history (просмотр списка ранее запущенных команд), pwd (отображение текущего рабочего каталога), set (установка параметров оболочки) и type (отображение местоположения команды).
5. **Команды файловой системы.** Такие команды хранятся в файловой системе и выполняются из нее. (Это команды, которые обозначаются значением переменной PATH.)

Чтобы определить расположение конкретной команды, можно использовать команду type. (Если у вас другая оболочка, не bash, то возьмите команду which.)

Например, чтобы узнать, где находится команда оболочки `bash`, введите следующее:

```
$ type bash
bash is /bin/bash
```

Примените слова `which`, `case`, `return` с командой `type`, чтобы увидеть другое расположение команд. Если команда находится в нескольких местах, добавьте параметр, чтобы вывести все местоположения команды на экран. Например, команда `type -a ls` отобразит псевдоним и расположение в файловой системе команды `ls`.

СОВЕТ

Иногда при запуске команды возникает сообщение об ошибке — о том, что команда не найдена или разрешения на ее запуск нет. Если команда не была найдена, убедитесь, что она правильно написана и находится в переменной PATH. Если запуск команды был отклонен, она может находиться в переменной PATH, но ее нельзя запустить без разрешения. Помните, что регистр важен, поэтому при вводе слов CAT или Cat команда cat найдена не будет.

Если команды нет в переменной `PATH`, можно использовать команду `locate`, чтобы найти ее. С помощью `locate` можно найти какую-либо часть системы при условии, что она доступна пользователю (некоторые файлы — только суперпользователю). Например, если нужно найти расположение команды `chage`, введите:

```
$ locate chage
/usr/bin/chage
/usr/sbin/lchage
/usr/share/man/fr/man1/chage.1.gz
/usr/share/man/it/man1/chage.1.gz
/usr/share/man/ja/man1/chage.1.gz
/usr/share/man/man1/chage.1.gz
/usr/share/man/man1/lchage.1.gz
/usr/share/man/pl/man1/chage.1.gz
/usr/share/man/ru/man1/chage.1.gz
/usr/share/man/sv/man1/chage.1.gz
/usr/share/man/tr/man1/chage.1.gz
```

Обратите внимание на то, что инструкция `locate` нашла не только команду `chage`, но и команду `lchage`, и множество справочных страниц, связанных с `chage`, для разных языков. Команда `locate` просматривает всю файловую систему, а не только каталоги с командами. (Если `locate` не находит файлы, недавно добавленные в систему, запустите команду `updatedb` от имени суперпользователя, чтобы обновить локальную базу данных.)

В следующих главах мы научимся применять дополнительные команды. Пока необходимо поближе познакомиться с тем, как работает командный интерпретатор. Поэтому далее обсудим функции для вызова команд и их выполнения, использования переменных и создания псевдонимов.

Вызов команд из истории

Удобно, что есть возможность повторить команду, выполненную ранее в интерпретаторе. Воспроизведение длинной и сложной командной строки может стать тем еще испытанием. К счастью, функции оболочки позволяют вызывать предыдущие командные строки, изменять их и завершать частично введенную командную строку.

История интерпретатора — это список команд, которые были введены ранее. Используя команду `history` в интерпретаторе `bash`, можно просмотреть введенные прежде команды. Затем с помощью различных функций оболочки можно вызвать отдельные командные строки из этого списка и изменить их.

Далее в этом разделе мы рассмотрим, как изменять команды, завершать части командных строк, вызывать список истории и работать с ним.

Изменение командной строки

Если в командную строку вкралась ошибка, оболочка `bash` позволяет не удалять ее всю и не начинать все сначала. И точно так же можно вызвать предыдущую командную строку и изменить в ней элементы, чтобы создать новую команду.

По умолчанию интерпретатор `bash` допускает изменение командной строки на основе текстового редактора emacs. (Введите `man emacs`, чтобы открыть справочные страницы и узнать о нем больше.) Если вы уже знакомы с emacs, то, вероятно, знаете большинство комбинаций клавиш, описанных далее.

СОВЕТ

По желанию можно выбрать команду vi для изменения командных строк. Добавьте следующую строку в файл .bashrc в своем домашнем каталоге:

```
set -o vi
```

В следующий раз, когда будет запущен интерпретатор, примените команду vi для изменения командных строк.

Для редактирования можно задействовать комбинацию управляющих клавиш, клавиши Meta и клавиш со стрелками. Например, сочетание клавиш Ctrl+F означает, что нужно, удерживая нажатой клавишу Ctrl, нажать клавишу f. Alt+F означает, что нужно удерживать нажатой клавишу Alt и ввести f. (Вместо Alt на клавиатуре можно использовать клавиши Meta или Esc. А на клавиатуре Windows — клавишу Windows.)

Чтобы изменить командную строку, введите:

```
$ ls /usr/bin | sort -f | less
```

Эта команда выводит список содержимого каталога `/usr/bin`, сортирует содержимое в алфавитном порядке (независимо от регистра) и передает выходные данные команде `less`. Эта команда воспроизводит первую страницу вывода, после чего можно отобразить остальную часть вывода (нажмите клавишу Enter) или страницу (нажмите Пробел). Когда закончите, нажмите клавишу `q`. Теперь предположим, что вы хотите изменить каталог `/usr/bin` на `/bin`. Для этого выполните следующие действия.

1. **Нажмите клавишу ↑.** Отобразится самая последняя команда из истории интерпретатора.
2. **Нажмите сочетание клавиш Ctrl+A.** Курсор переместится в начало командной строки.
3. **Нажмите сочетание клавиш Ctrl+F или клавишу →.** Повторите эту команду несколько раз, чтобы передвинуть курсор к первой косой черте (`/`).
4. **Нажмите сочетание клавиш Ctrl+D.** Введите эту команду четыре раза, чтобы удалить `/usr` из строки.
5. **Нажмите клавишу Enter.** Командная строка сохранится и выполнит действие.

При изменении командной строки в нее можно добавлять обычные символы. Они появятся там, где установлен текстовый курсор. Используйте стрелки ← и → для перемещения курсора из одного конца командной строки в другой. Если нажать клавиши ↑ и ↓, можно перейти к предыдущим командам в списке истории и выбрать другую командную строку для редактирования. (См. подраздел «Вызов командной строки», чтобы узнать, как вызвать команды из списка истории.) Можно применять разные сочетания клавиш, чтобы изменять командные строки. В табл. 3.1 перечислены комбинации клавиш для перемещения по командной строке.

Таблица 3.1. Сочетания клавиш для перемещения по командной строке

Сочетание клавиш	Название	Описание
Ctrl+F	Символ вправо (вперед)	Перейти на один символ вправо (вперед)
Ctrl+B	Символ влево (назад)	Перейти на один символ влево (назад)
Alt+F	Слово вправо (вперед)	Перейти на одно слово вправо (вперед)
Alt+B	Слово влево (назад)	Перейти на одно слово влево (назад)
Ctrl+A	Начало строки	Перейти к началу текущей строки
Ctrl+E	Конец строки	Перейти к концу текущей строки
Ctrl+L	Очистка экрана	Очистить экран и оставить строку в верхней части экрана

Комбинации клавиш, приведенные в табл. 3.2, используются для изменения командных строк.

Таблица 3.2. Сочетания клавиш для изменения командной строки

Сочетание клавиш	Название	Описание
Ctrl+D	Удалить текущий символ	Удалить символ до курсора
Backspace	Удалить предыдущий символ	Удалить символ перед курсором
Ctrl+T	Перенести символ	Поменять местами последние два символа перед курсором
Alt+T	Перенести слова	Поменять текущее слово на предыдущее
Alt+U	Слово в верхнем регистре	Изменить слово — перевести в верхний регистр
Alt+L	Слово в нижнем регистре	Изменить слово — перевести в нижний регистр
Alt+C	Слово с прописной буквы	Изменить слово под курсором — начать с прописной буквы
Ctrl+V	Специальный символ	Вставляет специальный символ, например символ табуляции — Ctrl+V+Tab

Сочетания клавиш, данные в табл. 3.3, позволяют вставлять и вырезать текст в командной строке.

Таблица 3.3. Сочетания клавиш для работы с текстом в командной строке

Сочетание клавиш	Название	Описание
Ctrl+K	Вырезать конец строки	Вырезать часть строки после курсора
Ctrl+U	Вырезать начало строки	Вырезать часть строки перед курсором
Ctrl+W	Вырезать предыдущее слово	Вырезать слово перед курсором
Alt+D	Вырезать следующее слово	Вырезать слово после курсора
Ctrl+Y	Вставить последний текст	Вставить последний вырезанный текст
Alt+Y	Вставить предыдущий текст	Вставить предыдущий вырезанный текст
Ctrl+C	Удалить всю строку	Прервать (убить) текущий процесс, удалить всю строку

Автозавершение командной строки

Чтобы вам не приходилось вводить множество сочетании клавиш, интерпретатор bash реализует различные варианты заполнения частично введенных значений. Чтобы заполнить значение, введите первые несколько символов и нажмите клавишу Tab. Вот некоторые значения, которые можно использовать из bash.

- **Команда, псевдоним или функция.** Если вводимый текст начинается с обычных символов, интерпретатор попытается дополнить его командой, псевдонимом или названием функции.
- **Переменная.** Если вводимый текст начинается со знака доллара ($), интерпретатор завершает текст переменной из текущей оболочки.
- **Имя пользователя.** Если текст, который вы вводите, начинается с тильды (~), оболочка завершает текст именем пользователя. Тогда ~username указывает на домашний каталог пользователя.
- **Имя хоста.** Если вводимый текст начинается с символа «эт» (@), оболочка завершает текст именем хоста, взятым из файла /etc/hosts.

СОВЕТ

Чтобы добавить имена хостов из дополнительного файла, установите переменную HOSTFILE в качестве имени этого файла. Он должен быть в том же формате, что и файл /etc/hosts.

Вот несколько примеров завершения командной строки. (Символ <*Tab*> означает, что нужно нажать клавишу Tab на клавиатуре):

```
$ echo $OS<Tab>
$ cd ~ro<Tab>
$ userm<Tab>
```

В первом примере $OS расширяется до переменной $OSTYPE. Далее ~ro расширяется до домашнего каталога суперпользователя (~root/). Затем userm расширяется до команды usermod.

Двойное нажатие клавиши Tab творит чудеса. Бывает, что доступны несколько вариантов автозавершения введенной строки. В таком случае рассмотрите возможные способы расширения текста, дважды нажав клавишу Tab в том месте, где нужно выполнить автозавершение.

Далее показан пример того, что получилось бы при проверке вариантов автозавершения для $P:

```
$ echo $P<Tab><Tab>
$PATH $PPID $PS1 $PS2 $PS4 $PWD
$ echo $P
```

В этом случае существует шесть переменных, которые начинаются с $P. После отображения всех вариантов командная строка возвращается в исходное положение, из которого и нужно выбрать один из вариантов. Например, если ввести

другую P и снова нажать Tab, командная строка будет завершена так: $PPID (единственный возможный вариант).

Вызов командной строки

После ввода команды вся командная строка сохраняется в истории. Список истории сохраняется вплоть до конца сеанса. Далее текущий список записывается в файл истории, из которого можно вызвать любую команду в любом другом сеансе. После вызова команды можно изменять командную строку, как описано ранее.

Чтобы просмотреть список истории, используйте команду `history`. Введите команду без параметров или с числом после нее, чтобы установить предел отображения последних команд, например:

```
$ history 8
 382 date
 383 ls /usr/bin | sort -a | more
 384 man sort
 385 cd /usr/local/bin
 386 man more
 387 useradd -m /home/chris -u 101 chris
 388 passwd chris
 389 history 8
```

Перед каждой командной строкой в списке стоит число. Чтобы вызвать одну из этих команд, используйте восклицательный знак (!). Имейте в виду, что в этом случае команда выполняется вслепую, без подтверждения оригинала команды. Существует несколько способов немедленного запуска команды из списка, например такие.

- `!n` — *запустить команду под номером n*. Замените n номером командной строки, и она будет запущена. Например, так выполняется повтор команды `date` под номером `382` в предыдущем списке истории:

  ```
  $ !382
  date
  Fri Jun 29 15:47:57 EDT 2019
  ```

- `!!-!!` — *запустить предыдущую команду*. Запускает предыдущую командную строку. Вот так можно было бы сразу запустить ту же команду `date`:

  ```
  $ !!
  date
  Fri Jun 29 15:53:27 EDT 2019
  ```

- `!?строка–?` — *запустить команду со строкой*. Запускает самую последнюю команду, содержащую определенную строку символов. Например, можно снова запустить команду `date`, просто выполнив поиск части этой командной строки следующим образом:

  ```
  $ !?dat?
  date
  Fri Jun 29 16:04:18 EDT 2019
  ```

Вместо того чтобы сразу запускать команду `history`, можно вызвать конкретную строку и отредактировать ее. Для этого используйте сочетания клавиш из табл. 3.4.

Таблица 3.4. Сочетания клавиш для работы с историей

Сочетание клавиш	Название	Описание
Клавиши стрелок	Шаг	Нажимайте клавиши со стрелками ↑ и ↓, чтобы перемещаться по командным строкам в списке истории и добираться до нужной (Ctrl+P и Ctrl+N соответственно выполняют те же функции)
Ctrl+R	Обратный последовательный поиск	Нажмите и удерживайте это сочетание клавиш и начните печатать, чтобы найти команду из истории bash
Ctrl+S	Последовательный поиск вперед	Делает то же самое, но ищет по списку вперед (не всегда может сработать)
Alt+P	Обратный поиск	Нажмите и введите строку для выполнения обратного поиска. После нажмите клавишу Enter, чтобы увидеть самую последнюю командную строку, содержащую эту строку
Alt+N	Поиск вперед	Делает то же самое, но ищет по списку вперед (не всегда может сработать)

Еще один способ работы со списком истории — использование команды `fc`. Введите `fc`, за которой следует номер строки в истории, и эта командная строка откроется в текстовом редакторе (в `vi` по умолчанию введите `:wq` для сохранения и выхода или `:q!` для выхода, если происходят ошибки в `vi`). Внесите нужные изменения. При выходе из редактора команда будет выполнена.

Можно также указать диапазон номеров строк (например, `fc 100 105`). Все команды открываются в текстовом редакторе, а затем выполняются одна за другой при выходе из редактора.

После закрытия интерпретатора список истории сохраняется в файле `.bash_history` в домашнем каталоге. По умолчанию в истории сохраняется до 1000 команд.

ПРИМЕЧАНИЕ

Некоторые суперпользователи отключают функцию истории, вызывая переменную HISTFILE в /dev/null или просто оставляя пустой переменную HISTSIZE. Таким образом предотвращается потенциальное использование информации о действиях суперпользователя. Если вы администратор с правами суперпользователя, вам может быть полезно отключить эту функцию по тем же причинам. Кроме того, так как история сохраняется при правильном выходе из интерпретатора, предотвратить сохранение можно, завершив процесс интерпретатора. Например, чтобы завершить оболочку с идентификатором процесса 1234, введите kill -9 1234 из любого интерпретатора.

Соединение и расширение команд

Мощной особенностью интерпретатора является возможность перенаправления ввода и вывода команд в другие команды и файлы и обратно. Чтобы связать команду, интерпретатор применяет метасимволы. *Метасимвол* — это печатный символ с особым значением для командного интерпретатора: он соединяет и расширяет команды.

К метасимволам относятся символ конвейера (`|`), амперсанд (`&`), точка с запятой (`;`), правая скобка (`)`), левая скобка (`(`), знаки «меньше» (`<`) и «больше» (`>`). В следующих разделах мы рассмотрим, как использовать метасимволы в командной строке для смены поведения команд.

Расширение команд с помощью конвейера

Метасимвол конвейера (`|`) соединяет выходные данные одной команды с входными данными другой. Таким образом одна из команд может предоставлять данные, а другая — результаты. Вот пример командной строки, которая включает в себя конвейеры:

```
$ cat /etc/passwd | sort | less
```

Эта команда выводит список содержимого файла `/etc/passwd` и передает выходные данные в команду `sort`. Команда `sort` собирает имена пользователей, которые начинаются с каждой строки файла `/etc/passwd`, сортирует их в алфавитном порядке и передает выходные данные команде `less` (чтобы пролистать выходные данные).

Конвейеры являются прекрасной иллюстрацией того, что система UNIX, предшественница Linux, состояла из множества строительных блоков. Обычно в UNIX утилиты подключались различными способами, чтобы выполнять разные задания. Например, до появления графических текстовых процессоров пользователи создавали простые текстовые файлы, которые содержали макросы с форматированием. Чтобы увидеть, как выглядит документ в итоге, они использовали бы следующую команду:

```
$ gunzip < /usr/share/man/man1/grep.1.gz | nroff -c -man | less
```

В данном примере содержимое справочной страницы `grep` (`grep.1.gz`) направляется в команду `gunzip` для распаковки. Выходные данные `gunzip` передаются в команду `nroff` для форматирования справочной страницы с помощью ручного макроса (`-man`). Чтобы отобразить выходные данные, они передаются по конвейеру в команду `less`. Поскольку отображаемый файл является обычным текстом, в него можно добавить любое количество параметров редактирования перед выводом. Можно сортировать содержимое, изменять или удалять некоторые части, а также

вводить текст из других документов. Суть в том, что все эти функции не находятся в одном месте, а их результаты с конвейерной передачи перенаправляются между несколькими командами.

Последовательные команды

Может потребоваться выполнить последовательность команд, причем предшествующая команда должна завершаться до начала следующей. Это возможно, если ввести несколько команд в виде одной командной строки, разделив их точкой с запятой (`;`):

```
$ date ; troff -me verylargedocument | lpr ; date
```

В этом примере я форматировал огромный документ и хотел знать, сколько времени это займет. Первая команда (`date`) показывала дату и время до начала форматирования. Команда `troff` отформатировала документ, а затем передала вывод на печать. Когда процесс форматирования закончился, дата и время отобразились снова, так я узнал, сколько времени потребовалось команде `troff` для его завершения.

`mail` — еще одна полезная команда, которую можно добавить в конец длинной командной строки, например:

```
; mail -s "Finished the long command" c hris@example.com
```

Затем, когда команда завершит работу, сообщение об этом будет отправлено выбранному пользователю.

Выполнение команд в фоновом режиме

Выполнение отдельных команд может занять некоторое время. Однако бывает, что интерпретатор необходим и для других дел и нет возможности дожидаться окончания работы такой команды. В таких случаях можно запустить команды в фоновом режиме с помощью амперсанда (`&`).

Команды форматирования текста (например, `mail` и `troff`, описанные ранее) могут выполняться в фоновом режиме, если документ большой. Можно создать собственные скрипты оболочки, которые работают в фоновом режиме, чтобы постоянно проверять, произошли ли определенные события, например заполнение жесткого диска или вход пользователей в систему.

Далее приведен пример команды, выполняемой в фоновом режиме:

```
$ troff -me verylargedocument | lpr &
```

Не закрывайте интерпретатор, пока процесс не закончится или не будет завершен принудительно. Другие способы управления фоновыми процессами описаны в главе 6.

Расширение команд

С помощью подстановки команд можно получить выходные данные команды, которые интерпретирует оболочка, а не сама команда. В таком случае стандартный вывод одной команды может стать аргументом для другой. Две формы подстановки команд — это `$(command)` и `` `command` `` (обратные кавычки, а не одинарные).

Команда в этом случае может включать параметры, метасимволы и аргументы. Далее приведен пример использования подстановки команд:

```
$ vi $(find /home | grep xyzzy)
```

Здесь подстановка команд происходит до выполнения команды `vi`. Во-первых, команда `find` запускается в каталоге `/home` и выводит все файлы и каталоги, размещенные ниже в файловой системе. Выходные данные передаются команде `grep`, фильтрующей все файлы, за исключением тех, в имени которых есть строка `xyzzy`. Наконец, команда `vi` открывает все файлы для редактирования (по одному за раз), если их имена содержат `xyzzy`. (Если вы запускаете эту программу и незнакомы с `vi`, введите `q!`, чтобы выйти из файла.)

Этот пример полезен, если нужно отредактировать файл, у которого известно имя, но не местоположение. Локально вы можете найти и открыть каждый файл, расположенный в выбранном месте в файловой системе. (Другими словами, не используйте команду `grep` из корневого каталога файловой системы, иначе интерпретатор попытается отредактировать несколько тысяч файлов.)

Выполнение арифметических операций

Бывает, необходимо передать команде арифметические результаты. Существуют две формы для расширения арифметического выражения и передачи его в интерпретатор: `$[`*выражение*`]` и `$(`*выражение*`)`, например:

```
$ echo "I am $[2020 - 1957] years old."
I am 63 years old.
```

Оболочка сначала интерпретирует арифметическое выражение (`2020 - 1957`), а затем передает эту информацию команде `echo`. Эта команда отображает текст с результатом вычислений (`63`).

Другой пример:

```
$ echo "There are $(ls | wc -w) files in this directory."
There are 14 files in this directory.
```

Здесь перечисляется содержимое текущего каталога (`ls`) и запускается команда подсчета слов для определения количества найденных файлов (`wc -w`). Полученное число (в данном случае 14) выводится вместе с остальной частью приведенного предложения.

Расширение переменных

Переменные, которые хранят информацию в интерпретаторе, можно расширить с помощью знака доллара (`$`). При расширении переменной в командной строке вместо имени переменной выводится ее значение, как показано далее:

```
$ ls -l $BASH
-rwxr-xr-x. 1 root root 1219248 Oct 12 17:59 /usr/bin/bash
```

Применение `$BASH` в качестве аргумента для `ls -l` приводит к выводу длинного списка команды `bash`.

Использование переменных интерпретатора

Командный интерпретатор хранит информацию, которая может быть полезна для сеанса, в так называемых *переменных*. Примеры переменных: `$SHELL` (определяет задействуемую оболочку), `$PS1` (приглашение командной строки) и `$MAIL` (определяет местоположение почтового ящика пользователя).

С помощью команды `set` можно просмотреть все переменные, установленные для текущего интерпретатора. Подмножество локальных переменных называется *окружением переменных*. Переменные окружения — это переменные, которые можно экспортировать в другие интерпретаторы, открытые из текущей оболочки. Введите `env`, чтобы увидеть переменные окружения.

Можно ввести `echo $`*`ЗНАЧЕНИЕ`*, где *`ЗНАЧЕНИЕ`* заменяется именем конкретной переменной из среды, чтобы ее указать. Поскольку в Linux что угодно можно сделать несколькими способами, попробуйте ввести `declare`, чтобы вывести список текущих переменных окружения и их значений вместе со списком функций интерпретатора.

Помимо тех, что задает пользователь, есть переменные, устанавливаемые системными файлами. Они хранят, например, расположение файлов конфигурации, почтовых ящиков и каталогов путей. Они также могут хранить значения для подсказок интерпретатора, размер списка истории и тип операционной системы. Вы можете ссылаться на значение любой из этих переменных, ставя перед ней знак доллара (`$`) в любом месте командной строки:

```
$ echo $USER
chris
```

Эта команда выводит значение переменной `USER`, которая содержит имя пользователя (`chris`). Введите любое значение для `USER`, чтобы получить новое значение вместо текущего.

При запуске интерпретатора (вход в систему через виртуальную консоль или окно терминала) большинство переменных окружения уже готовы к работе. В табл. 3.5 показаны некоторые переменные, задаваемые либо при использовании интерпретатора `bash`, либо для работы с различными функциями.

Таблица 3.5. Общие переменные окружения

Переменная	Описание
BASH	Отображает путь к оболочке, которая занимается интерпретацией команд. Обычно это путь /bin/bash
BASH_VERSION	Отображает конкретную версию bash
EUID	Идентификатор текущего пользователя. Он назначается при запуске оболочки на основе записи пользователя в файле /etc/passwd
FCEDIT	Редактор по умолчанию, используемый командой fc для работы с командой history. Если эта переменная не задана, берется команда vi
HISTFIL	Имя файла для сохранения списка истории оболочки (по умолчанию хранится в $HOME/.bash_history)
HISTFILESIZE	Максимальное количество строк, сохраняемых в файле истории. После достижения лимита самые старые команды удаляются. Лимит по умолчанию — 1000
HISTCMD	Номер истории текущей команды
HOME	Текущая домашняя папка пользователя. Это текущий рабочий каталог пользователя при входе в систему или применении команды cd с любыми параметрами
HOSTTYPE	Строка, описывающая компьютерную архитектуру устройства, на котором запущена операционная система. Для персональных компьютеров, совместимых с Intel, это i386, i486, i586, i686, например i386-linux. Для 64-разрядных машин AMD это значение x86_64
MAIL	Отображает электронную почту текущего пользователя. Это файл пользователя в каталоге /var/spool/mail
OLDPWD	Отображает предыдущий рабочий каталог, используемый в оболочке
OSTYPE	Строка, определяющая операционную систему, в которой работает оболочка. Для Fedora Linux значение OSTYPE равно linux или linux-gnu в зависимости от типа оболочки. (Bash может работать и в других операционных системах)
PATH	Список каталогов, которые будет проверять система при обращении к той или иной команде. Когда пользователь вводит команду, система проверяет указанные папки последовательно. Пример папки: /bin:/usr/bin:/usr/local/bin:/usr/bin/X11:/usr/X11R6/bin:~/bin. Если команды нет в переменной PATH, необходимо ввести полный или относительный путь к ней. Для суперпользователя значение также включает /sbin, /usr/sbin и /usr/local/sbin
PPID	Идентификатор родительского процесса (PID) оболочки bash (например, окно терминала, содержащее оболочку)

Продолжение ⇗

Таблица 3.5 *(продолжение)*

Переменная	Описание
PROMPT_COMMAND	Если установлен в имени команды, команда выполняется каждый раз перед отображением первичного приглашения. После установки команда PROMPT_COMMAND=date выводит текущие дату/время до появления приглашения
PS1	Строка приглашения на ввод. Эта переменная используется для определения того, как будет выглядеть данная строка. Есть еще вторая строка PS2, которая применяется при многострочной команде
PWD	Текущий рабочий каталог. Значение изменяется каждый раз, когда каталоги меняются с помощью команды cd
RANDOM	Возвращает случайное число от 0 до 99 999. Значение переменной дает генератор случайных чисел
SECONDS	Количество секунд с момента запуска оболочки
SHLVL	Указывает уровень оболочки, увеличивающийся на единицу при каждом запуске новой оболочки bash. При входе в оболочку переменная SHLVL равна 1
TMOUT	Продолжительность (в секундах) ожидания ввода оболочкой. Значение по умолчанию, равное нулю, показывает, что нужно ждать бесконечно. После того как время истекло, оболочка завершает работу. Используется как функция безопасности, снижающая вероятность применения оболочки несанкционированными лицами. (В настройках входа в систему необходимо разрешить оболочке выходить из системы)

Создание и использование псевдонимов

С помощью команды `alias` вы можете эффективно создавать псевдонимы для любых команды и параметров, чтобы запускать их позднее. А еще добавлять и перечислять псевдонимы. Рассмотрим следующие примеры применения псевдонимов из оболочки `bash`:

```
$ alias p='pwd ; ls –CF'
$ alias rm='rm -i'
```

В первом примере буква `p` назначается для выполнения команды `pwd`, а затем для запуска `ls-CF`, чтобы вывести на экран текущий рабочий каталог и перечислить его содержимое в виде столбца.

Во втором примере команда выполняется с параметром `-i` при каждом вводе `rm`. (Это псевдоним, который устанавливается автоматически для суперпользователя. Вместо того чтобы просто удалять файлы, оболочка будет предлагать удалить каждый файл по отдельности. Это предотвращает автоматическое удаление всех файлов в каталоге при ошибке ввода команды типа `rm *`.)

Находясь в командном интерпретаторе, можно проверить, какие псевдонимы установлены, введя команду `alias`. Чтобы удалить все псевдонимы, введите `unalias`. (Имейте в виду, что, если псевдоним задан в файле конфигурации, он будет установлен снова при открытии другой оболочки.)

Выход из командного интерпретатора

Чтобы выйти из оболочки, введите `exit` или нажмите сочетание клавиш Ctrl+D. Если вы используете оболочку из окна терминала, выход из нее закроет и терминал. Если работаете в виртуальной консоли, оболочка завершит работу и откроет приглашение войти в систему.

Если в одном сеансе открыты несколько оболочек, выход из одной вернет пользователя к предыдущей. Например, команда `su` открывает оболочку нового пользователя. Выход из этой оболочки просто возвратит вас в исходную оболочку.

Создание среды оболочки

Оболочку можно настроить под себя, чтобы работать более эффективно. Это значит, что можно установить псевдонимы для создания ярлыков любимых командных строк и переменных окружения для хранения информации. Добавляя эти параметры в файлы конфигурации оболочки, можно сделать их доступными при каждом открытии оболочки.

Настройка оболочки

За работу оболочки отвечают несколько файлов конфигурации. Некоторые файлы используются для всех пользователей и всех интерпретаторов, а другие подходят только для пользователя, создавшего конкретный файл конфигурации. В табл. 3.6 показаны файлы, необходимые для тех, кто применяет оболочку `bash` в Linux. (Обратите внимание на символ ~ в именах файлов. Он говорит о том, что файл находится в домашнем каталоге соответствующего пользователя.)

Таблица 3.6. Файлы конфигурации bash

Файл	Описание
/etc/profile	Загружается и выполняется при запуске любой командной оболочки в системе. Этот файл устанавливает значения для пути и настраивает переменные окружения для таких параметров, как расположение почтового ящика и размер файлов истории. Файл /etc/profile собирает также параметры оболочки из файлов конфигурации в каталоге /etc/profile.d

Продолжение ⇗

Таблица 3.6 *(продолжение)*

Файл	Описание
/etc/bashrc	Файл выполняется для каждого пользователя, который запускает оболочку bash. Он устанавливает приглашение по умолчанию и может добавлять один или несколько псевдонимов. Значения в этом файле могут быть переопределены информацией, содержащейся в файле ~/.bashrc каждого пользователя
~/.bash_profile	Используется каждым пользователем для ввода информации, специфичной для его взаимодействия с интерпретатором. Выполняется только один раз — при входе пользователя в систему. По умолчанию устанавливает несколько переменных окружения и запускает пользовательский файл .bashrc. Позволяет добавлять переменные окружения и передавать их будущим оболочкам
~/.bashrc	Содержит информацию, специфичную для оболочек пользователя. Считывается при входе в систему, а также каждый раз, когда открывается новая оболочка bash. Лучше добавлять псевдонимы в этот файл, чтобы они сработали в оболочке
~/.bash_logout	Выполняется при выходе из системы (из последней оболочки bash)

Чтобы изменить файлы `/etc/profile` и `/etc/bashrc`, необходимо быть суперпользователем. Однако будет эффективнее создать файл `/etc/profile.d/custom.sh` для добавления общесистемных настроек, а не редактировать эти файлы напрямую. Пользователи могут изменять информацию в файлах `$HOME/.bash_profile`, `$HOME/.bashrc` и `$HOME/.bash_logout` в собственных домашних каталогах.

Более продвинутый редактор `vi` описан в главе 5, а пока можно работать со встроенным редактором для простых текстовых файлов. Например, чтобы отредактировать и добавить информацию в файл `$HOME/.bashrc`, введите следующую команду:

```
$ nano $HOME/.bashrc
```

В открытом файле переместите курсор в его нижнюю часть с помощью клавиши ↓. Введите нужную строку, например, `alias d='date+%D'`. Чтобы сохранить файл, нажмите сочетание клавиш Ctrl+O (буква «O»), чтобы выйти — Ctrl+X. В следующий раз, войдя в систему или открыв новую оболочку, вы сможете использовать созданный псевдоним (в данном случае `d`). Чтобы новая информация была доступна из текущей оболочки, введите:

```
$ source $HOME/.bashrc
$ d
06/29/19
```

В следующих разделах мы рассмотрим элементы, которые необходимо добавлять в файлы конфигурации оболочки. В большинстве случаев эти значения добавляются в файл `.bashrc` в домашней папке. Но если вы администрируете систему, то сможете установить некоторые из них используемыми по умолчанию для всех пользователей системы Linux.

Настройка приглашения

Приглашение состоит из набора символов, которые появляются каждый раз, когда оболочка готова выполнять команду. Переменная окружения `PS1` задает, что приглашение содержит и с чем будет происходить взаимодействие бóльшую часть времени. Если оболочке требуется дополнительный ввод, она задействует значения `PS2`, `PS3` и `PS4`.

В установленных системах Linux приглашение содержит не только знак доллара или фунта. Например, в дистрибутивах Fedora или Red Hat Enterprise Linux приглашение содержит имена пользователя и хоста и базовое имя текущего рабочего каталога. Эта информация заключается в квадратные скобки и сопровождается знаком доллара (для обычных пользователей) или знаком фунта (для суперпользователей). Вот как может выглядеть приглашение:

```
[chris@myhost bin]$
```

Если папки будут изменены, то вместо имени файла `bin` будет стоять имя нового каталога. И при входе на другой хост или под другим пользователем информация в строке изменится.

Можно использовать несколько специальных символов (с обратной косой чертой), чтобы добавить в приглашение информацию. Специальные символы применяются для вывода номера терминала, даты и времени, а также другой информации. В табл. 3.7 приведены некоторые примеры (более подробная информация имеется на справочной странице `bash`).

Таблица 3.7. Символы для добавления информации в приглашение оболочки

Символ	Описание
\!	Порядковый номер данной команды в истории команд. Включает в себя все предыдущие команды, сохраненные для конкретного пользователя
\#	Текущий номер команды. Включает в себя только команды активной оболочки
\$	Символ #, если оболочка запущена суперпользователем, и $, если оболочка запущена обычным пользователем
\W	Текущий рабочий каталог (без указания пути). Например, если текущий рабочий каталог — /var/spool/mail, то это значение отображается как mail
\[	Начало последовательности непечатаемых символов (этот символ может использоваться для того, чтобы включить в текст подсказки последовательность управляющих символов терминала)
\]	Конец последовательности непечатаемых символов
\\	Обратный слеш
\d	Дата в формате «день, месяц, число», например Sat Jan 23
\h	Имя хоста, на котором запущена оболочка
\n	Новая строка (перевод строки)

Продолжение ⇗

Таблица 3.7 *(продолжение)*

Символ	Описание
\nnn	Символ, имеющий восьмеричный код nnn
\s	Имя текущей оболочки. Для оболочки bash — значение bash
\t	Текущее время в 24-часовом формате «часы, минуты, секунды», например 10:14:39
\u	Имя пользователя, запустившего оболочку
\w	Полное имя текущего рабочего каталога, начиная с корневого

СОВЕТ

Если приглашение временное, то, набирая его в командной строке, поместите значение PS1 в кавычки. Например, введите export PS1="[\t \w]\$", чтобы увидеть следующее приглашение:

```
[20:26:32 /var/spool]$.
```

Чтобы сделать изменение в приглашении постоянным, добавьте значение `PS1` в файл `.bashrc` в домашнем каталоге (при условии, что используете оболочку `bash`). Возможно, в этом файле уже есть значение `PS1`, которое можно изменить. Обратитесь к руководству Bash (tldp.org/HOWTO/Bash-Prompt-HOWTO), чтобы узнать больше об изменении цветов, команд и других функций командной строки интерпретатора `bash`.

Добавление переменных окружения

Бывает, что необходимо добавить несколько новых переменных окружения в файл `.bashrc`. Вот какие переменные позволяют сделать работу с оболочкой более эффективной и результативной.

- `TMOUT`. Устанавливает, как долго оболочка может быть неактивной, прежде чем `bash` автоматически завершит работу. Значение выражается в количестве секунд, в течение которых оболочка ожидает получения входных данных. Эта функция полезна в качестве системы безопасности в случае, если пользователь покинет рабочее место, но не выйдет из Linux. Чтобы предотвратить выход из системы во время работы, установите значение `TMOUT=180` (разрешение на 30 минут ожидания). Можно использовать любой сеанс терминала, чтобы закрыть текущую оболочку через заданное количество секунд, например `TMOUT=30`.
- `PATH`. Как описывалось ранее, переменная `PATH` задает каталоги, в которых выполняется поиск применяемых команд. При частом использовании каталогов, которых нет в пути, команды можно добавлять. Для этого нужно добавить в переменную `PATH` в файл `.bashrc`. Например, чтобы добавить каталог `/getstuff/bin`, введите:

  ```
  PATH=$PATH:/getstuff/bin ; export PATH
  ```

Здесь сначала все текущие каталоги пути считываются в новый путь (`$PATH`), затем добавляется каталог `/getstuff/bin` и экспортируется новая переменная `PATH`.

ВНИМАНИЕ!

Некоторые пользователи присоединяют текущий каталог к своему пути, добавляя каталог, идентифицируемый просто как точка (.), как в примере:

```
PATH=.:$PATH ; export PATH
```

Таким образом можно запускать команды в текущем каталоге перед вычислением любой другой команды в пути (легко привыкнуть, если есть опыт работы с DOS). Однако безопасность системы снижается из-за того, что можно оказаться в каталоге, содержащем команду, которую вы не собираетесь запускать из этого каталога. Например, злоумышленник может поместить команду ls в каталог, который вместо того, чтобы перечислить содержимое вашего каталога, будет выполнять другие задачи. Поэтому настоятельно не рекомендуется добавлять точки в путь.

- `WHATEVER`. Можно создавать собственные переменные окружения, чтобы добавить ярлыки в работу. Выберите любое незадействованное имя и присвойте ему нужное значение. Например, если вы часто используете файлы в каталоге `/work/time/files/info/`, попробуйте добавить такую переменную:

  ```
  M=/work/time/files/info/memos ; export M
  ```

  Для этого в текущем каталоге введите команду `cd $M`. Можно запустить из этого каталога программу `hotdog`, набрав `$M/hotdog`. Здесь попробуйте отредактировать файл под названием `bun`, напечатав `vi $M/bun`.

Информация о командах

На первый взгляд работа с оболочкой может показаться устрашающей. Все, что отображается, — это приглашение командной строки. Как же тогда узнать, какие команды доступны, какие параметры они принимают и как использовать дополнительные функции? К счастью, найти ответы легко. Перечислю, где можно посмотреть дополнительную информацию к этой главе.

- **Проверьте путь PATH.** Введите `echo $PATH`. Появится список каталогов с доступными командами. В эти каталоги входит большинство стандартных команд Linux, например:

  ```
  $ ls /bin
  arch     dd            fusermount  loadkeys   mv
  awk      df            gawk        login      nano
  basename dmesg         gettext     ls         netstat
  bash     dnsdomainname grep        lsblk      nice
  cat      domainname    gtar        lscgroup   nisdomainname
  chgrp    echo          gunzip      lssubsys   ping
  ```

```
chmod    ed          gzip        mail       ping6
chown    egrep       hostname    mailx      ps
cp       env         ipcalc      mkdir      pwd
cpio     ex          kbd_mode    mknod      readlink
csh      false       keyctl      mktemp     red
cut      fgrep       kill        more       redhat_lsb_init
dash     find        link        mount      rm
date     findmnt     ln          mountpoint rmdir
```

- **Используйте команду help.** Некоторые команды уже встроены в оболочку, поэтому не отображаются в каталоге. Команда `help` выводит список этих команд и параметры, доступные для каждой из них. (Введите `help | less`, чтобы пролистать список.) Чтобы узнать о конкретной встроенной команде, введите `help` *команда*, заменив слово *команда* нужным именем. Команда `help` работает только в оболочке `bash`.
- **Применяйте --help с другой командой.** Многие команды принимают параметр `--help`, который информирует о том, как используется команда. Например, если ввести `date --help | less`, то в выводе будут показаны не только параметры, но и форматы времени, которые можно применять с командой `date`. Другие команды просто задействуют параметр `-h`, например `fdisk -h`.
- **Используйте команду info.** Команда `info` также отображает информацию о командах из интерпретатора. Она может перемещаться по иерархическим связям и искать сведения о командах и других элементах. Не у всех команд есть описание, помещенное в информационную базу данных, но иногда там можно найти больше данных, чем на справочной странице.
- **Воспользуйтесь командой man.** Чтобы узнать больше о конкретной команде, введите `man` *команда*, заменив слово *команда* нужным именем. На экране появится описание команды и ее параметров.

Справочные man-страницы — это наиболее распространенное средство получения информации о командах и других основных компонентах системы Linux. Справочные страницы разделены на категории, перечисленные в табл. 3.8. Обычному пользователю будут интересны справочные страницы из раздела 1. Системному администратору подойдут разделы 5 и 8, а иногда 4. Программистов чаще всего интересуют разделы 2 и 3.

Таблица 3.8. Разделы справочных страниц

Номер раздела	Название раздела	Описание
1	Команды пользователя	Команды, которые могут быть запущены из оболочки обычным пользователем (как правило, никаких административных прав не требуется)
2	Системные вызовы	Функции программирования, которые используются в приложении для вызова ядра

Номер раздела	Название раздела	Описание
3	Библиотечные функции	Функции программирования, обеспечивающие интерфейсы к определенным библиотекам программирования (например, для определенных графических интерфейсов или других библиотек, работающих в пользовательском пространстве)
4	Специальные файлы и внешние устройства	Узлы файловой системы, представляющие аппаратные устройства (например, терминалы или CD-приводы) или программные устройства (к примеру, генераторы случайных чисел)
5	Форматы файлов	Типы файлов (например, графический или текстовый файл обработки) или конкретные файлы конфигурации (к примеру, passwd или group)
6	Игры и демонстрации	Доступные в системе игры
7	Разное	Протоколы, файловые системы, наборы символов и т. д.
8	Команды для системного администрирования	Команды, для использования которых требуются права суперпользователя или другие привилегии администратора

Параметры этой команды позволяют выполнять поиск в базе данных справочных страниц или отображать их на экране. Далее представлены несколько примеров параметров:

```
$ man -k passwd
...
passwd               (1) - update user's authentication tokens
passwd               (5) - password file
$ man passwd
$ man 5 passwd
```

Параметр `-k` выполняет поиск по разделам «имя» и «описание» всех справочных страниц, установленных в системе. Существует около дюжины справочных страниц, на которых слово `passwd` есть в названии или описании команды.

ПРИМЕЧАНИЕ

Если man -k не выводит никаких данных, возможно, база данных справочных страниц не была добавлена. Введите mandbas в корневом каталоге, чтобы обработать базу данных страниц.

Предположим, что две справочные страницы, которые меня интересуют, — это команда `passwd` (из раздела 1) и файл `passwd` (из раздела 5). Если просто набрать `man passwd`, появится страница из раздела 1, а мне нужна страница из раздела 5, поэтому ввожу `man 5 passwd`.

При просмотре справочной страницы можно изучать части файла с помощью клавиш Page Down и Page Up (листание постранично). Используйте клавишу Enter

или ↑ и ↓ для перемещения на одну строку за раз. Нажмите клавишу / и введите термин для его поиска в документе. Нажмите n, чтобы повторить поиск вперед, или N, чтобы повторить поиск назад. Для выхода со справочной страницы нажмите q.

Резюме

Чтобы стать опытным пользователем Linux, необходимо уметь работать с оболочкой для ввода команд. Эта глава посвящена оболочке bash, наиболее часто применяемой в системах Linux. Вы узнали, как структурированы команды и как задействуются специальные функции, такие как переменные, завершение команд и псевдонимы.

В следующей главе рассмотрим, как перемещаться по файловой системе Linux из командной строки оболочки.

Упражнения

Выполните упражнения, чтобы проверить свои знания об использовании интерпретатора. Для этого применяйте систему Fedora или Red Hat Enterprise Linux (хотя некоторые примеры сработают и в других системах Linux). Если затрудняетесь с решением заданий, воспользуйтесь ответами к упражнениям, приведенными в приложении Б (хотя Linux позволяет решать задачи разными способами).

1. С рабочего стола перейдите в третью виртуальную консоль и войдите в свою учетную запись. Запустите несколько команд. Затем выйдите из оболочки и вернитесь на рабочий стол.
2. Откройте окно терминала и измените цвет шрифта на красный, а фона — на желтый.
3. Найдите расположение команды mount и справочной страницы tracepath.
4. Введите следующие три команды:

```
$ cat /etc/passwd
$ ls $HOME
$ date
```

   а затем вызовите и измените их, как описано далее.

   - Используйте командную строку, чтобы вызвать команду cat и сменить файл /etc/passwd на файл /etc/group.
   - Вызовите команду ls, определите, как отсортировать файлы по времени (с помощью справочной страницы), и добавьте этот параметр в командную строку ls $HOME.
   - Добавьте формат в команду date, чтобы отобразить выходные данные даты в виде «месяц/день/год».

5. Выполните следующую команду, набрав как можно меньше символов (используя автозавершение с помощью клавиши Tab):

   ```
   basename /usr/share/doc/
   ```

6. Применяйте команду `cat` для перечисления содержимого файла `/etc/services` и передачи этого содержимого в команду `less`, чтобы пролистать его (нажмите `q`, чтобы выйти, когда закончите).
7. Выполните команду `date` таким образом, чтобы выходные данные этой команды выдавали текущие день, месяц, дату и год. Пусть текст появится в другой командной строке и будет выглядеть следующим образом (ваша дата, конечно, будет другой): Today is Thursday, December 19, 2019.
8. С помощью переменных выясните имя вашего хоста, имя пользователя, оболочки и домашних каталогов.
9. Создайте псевдоним `mypass`, который всегда будет отображать содержимое файла `etc/passwd` при входе в систему или открытии новой оболочки из своей учетной записи.
10. Отобразите справочную страницу для системного вызова `mount`.

4 Файловая система

В этой главе

- Файловая система Linux.
- Атрибуты файлов и каталогов.
- Создание файлов и каталогов.
- Изменение владельца и прав доступа.
- Копирование и перемещение файлов.

Файловая система Linux — это структура, в которой хранится вся информация на компьютере. Фактически одним из определяющих свойств систем UNIX, на которых основана Linux, является то, что почти все, что находится в вашем компьютере (данные, команды, символические ссылки, устройства и каталоги), представлено элементами файловых систем. Для работы с Linux очень важно знать, где что находится и как задействовать файловую систему из оболочки.

В Linux организована иерархия каталогов. В каждом каталоге находятся файлы и другие каталоги. Можно создать ссылку на любой файл или каталог, используя либо полный путь (например, `/home/joe/myfile.txt`), либо относительный путь (например, если `/home/joe` — это текущий каталог, то можно просто ссылаться на файл через `myfile.txt`).

Если составить карту файлов и каталогов в Linux, то она будет выглядеть как перевернутое дерево. Вверху находится *корневой* каталог (также называемый каталогом root), который представлен одной косой чертой (`/`). Ниже располагается список общих каталогов системы Linux, таких как `bin`, `dev`, `home`, `lib` и `tmp`. Каждый из них, а также каталоги, добавленные в корневой, могут содержать подкаталоги.

На рис. 4.1 показано, как организована файловая система Linux. В качестве примера связи каталогов здесь приведен каталог `/home`, содержащий подкаталог для пользователя `joe`. В каталоге `joe` находятся подкаталоги `Desktop`, `Documents` и др. Чтобы добраться до файла `memo1.doc`, находящегося в каталоге `memos`, нужно ввести полный путь к нему: `/home/joe/Documents/memos/memo1.doc`. Если текущий каталог — `/home/joe/`, то можно ввести `Documents/memos/memo1.doc`.

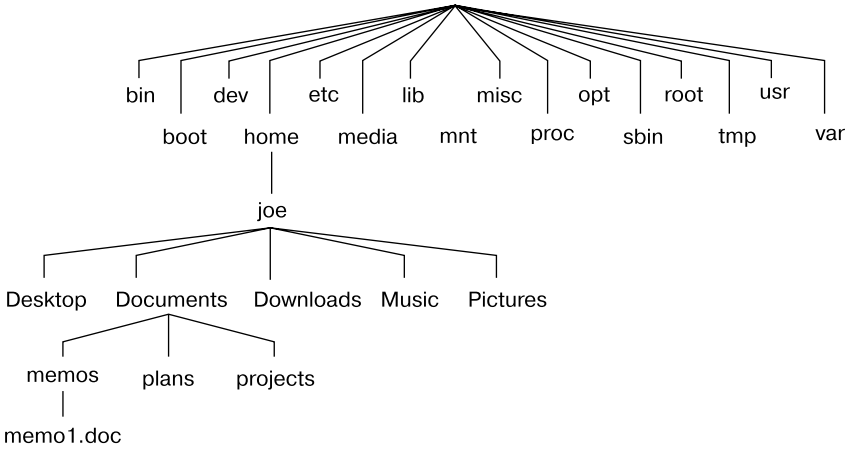


Рис. 4.1. Иерархия файловой системы Linux

Некоторые из этих каталогов Linux могут вас заинтересовать.

- `/bin` — содержит общие пользовательские команды Linux, такие как `ls`, `sort`, `date` и `chmod`.
- `/boot` — включает в себя загрузочное ядро Linux, диск начальной инициализации и файлы конфигурации загрузчика (GRUB).
- `/dev` — содержит файлы, представляющие точки доступа к устройствам в системах пользователя. К ним относятся устройства терминала (`tty*`), жесткие диски (`hd*` и `sd*`), оперативная память (`ram*`) и CD-ROM (`cd*`). Пользователи могут получить доступ к этим устройствам непосредственно через файлы устройств, однако приложения часто скрывают фактические имена устройств от конечных пользователей.
- `/etc` — содержит файлы конфигурации администратора. Большинство этих файлов являются обычными текстовыми файлами, которые, если у пользователя есть соответствующие права доступа, можно отредактировать с помощью любого текстового редактора.
- `/home` — содержит каталоги, назначенные каждому обычному пользователю с учетной записью входа. (Суперпользователь — исключение, он использует `/root` в качестве домашнего каталога.)
- `/lib` — содержит общие библиотеки, необходимые приложениям в `/bin` и `/sbin` для загрузки системы.
- `/media` — стандартное расположение для автоматически монтируемых устройств, в частности съемных носителей. Если у тома носителя есть имя, то оно обычно используется в качестве точки монтирования. Например, USB-накопитель с именем `myusb` будет смонтирован как `/media/myusb`.
- `/mnt` — общая точка монтирования для многих устройств до того, как она была вытеснена стандартным каталогом `/media`. Некоторые загрузочные системы

Linux еще задействуют этот каталог для монтирования разделов жесткого диска и удаленных файловых систем. Многие все еще применяют его для временного монтирования локальных или удаленных файловых систем, которые не монтируются постоянно.

- `/misc` — каталог, который иногда используется для автоматического монтирования файловых систем по запросу.
- `/opt` — структура каталогов, доступная для хранения дополнительного прикладного программного обеспечения.
- `/proc` — содержит информацию о системных ресурсах.
- `/root` — домашний каталог суперпользователя. Этот домашний каталог не находится ниже `/home` из соображений безопасности.
- `/sbin` — содержит административные команды и демонические процессы.
- `/sys` — содержит параметры для настройки хранения блоков и управления контрольными группами.
- `/tmp` — содержит временные файлы приложений.
- `/usr` — содержит пользовательскую документацию, игры, графические файлы (`X11`), библиотеки (`lib`) и множество других команд и файлов, не требующихся в процессе загрузки. Каталог `/usr` предназначен для файлов, которые не изменяются после установки (теоретически `/usr` может быть смонтирован только для чтения).
- `/var` — содержит каталоги данных приложений. В частности, именно здесь размещаются файлы, которые передаются через FTP-сервер (`/var/ftp`) или веб-сервер (`/var/www`). Он также содержит все файлы системного журнала (`/var/log`) и файлы, находящиеся в очереди на обработку в `/var/spool` (такие как `mail`, `cups`, `news`). Каталог `/var` содержит каталоги и файлы, которые часто изменяются. На серверных компьютерах он обычно создается как отдельная файловая система, которую можно легко расширить.

Файловые системы в операционных системах DOS или Microsoft Windows отличаются от файловой структуры Linux (см. далее врезку «Файловые системы Linux и Windows»).

Файловые системы Linux и Windows

Хотя файловая система Linux во многом схожа с системами MS-DOS и Windows, у нее есть поразительные отличия, например следующие.

- В файловых системах MS-DOS и Windows буквы дисков представляют различные устройства хранения. В Linux все устройства хранения данных подключены

к иерархии файловой системы. Таким образом, то, что весь файл `/usr` может находиться на отдельном жестком диске или что файл `/mnt/remote1` является файловой системой с другого компьютера, пользователю не видно.

- В Linux для разделения имен каталогов используется прямая, а не обратная косая черта (слеш). Поэтому путь в системе Microsoft выглядит таким образом: `C:\home\joe`, а в системе Linux — `/home/joe`.
- В DOS имена файлов почти всегда имеют суффиксы (например, `.txt` для текстовых файлов или `.docx` для файлов Word). Трехсимвольные суффиксы иногда можно применять и в Linux, однако они не имеют особого значения. Они полезны скорее для идентификации типа файла. Многие приложения и окружения Linux используют суффиксы файлов для определения содержимого последних. Однако в Linux расширения команд DOS, такие как `.com`, `.exe` и `.bat`, не всегда означают исполняемый файл. (Исполняемость файлов в Linux определяют флаги прав доступа.)
- Каждый файл и каталог в системе Linux имеют владельцев и права доступа, связанные с ним. Их безопасность варьируется в зависимости от системы Microsoft. Поскольку DOS и Microsoft Windows создавались как однопользовательские системы, принадлежность файлов в них не была встроена изначально. Чтобы решить эту проблему, в более поздние версии были добавлены такие функции, как атрибуты файлов и папок.

Базовые команды файловой системы

Я хочу познакомить вас с несколькими простыми командами, чтобы вы могли начать работать с файловой системой. Можете пробовать их сразу, параллельно с чтением. При запуске системы Linux и открытии оболочки пользователь входит в свой домашний каталог. Большинство файлов, которые вы сохраняете и с которыми работаете, вероятно, будут находиться в этом каталоге или в подкаталогах, созданных вручную. В табл. 4.1 показаны команды для создания и применения файлов и каталогов.

Таблица 4.1. Создание и использование файлов

Команда	Действие
cd	Меняет каталог
pwd	Выводит имя текущего (или настоящего) рабочего каталога
mkdir	Создает каталог
chmod	Изменяет права доступа к файлу или каталогу
ls	Перечисляет содержимое каталога

Одна из самых простых команд — это `cd`. Она может использоваться без каких-либо параметров (чтобы перенести пользователя в домашний каталог) или с полными или относительными путями. Рассмотрим следующие команды:

```
$ cd /usr/share/
$ pwd
/usr/share
$ cd doc
$ pwd
/usr/share/doc
$ cd
$ pwd
/home/chris
```

Параметр `/usr/share` представляет *абсолютный путь* к каталогу в системе. Поскольку он начинается с косой черты (`/`), этот путь указывает оболочке начать с корневого каталога файловой системы и привести к каталогу `share`, находящемуся в каталоге `usr`. Параметр `doc` для команды `cd` ищет каталог с именем `doc`, который относится к текущему каталогу. Таким образом, эта команда открывает `/usr/share/doc` и делает его текущим каталогом.

После этого, введя только `cd`, вы вернетесь в домашний каталог. Если необходимо узнать, где вы находитесь в файловой системе, команда `pwd` поможет. Вот еще несколько интересных вариантов использования команды `cd`:

```
$ cd ~
$ pwd
/home/chris
$ cd ~/Music
$ pwd
/home/chris/Music
$ cd ../../../usr
$ pwd
/usr
```

Тильда (`~`) ссылается на ваш домашний каталог. С помощью команды `cd ~` к нему можно перейти. Можно также применять тильду для ссылки на каталоги относительно вашего домашнего каталога, например, `/home/chris/Music` с помощью `~/Music`. Ввод имени в качестве параметра приведет вас в каталог ниже текущего каталога, а две точки (`..`) можно использовать, чтобы перейти в каталог уровнем выше текущего. В приведенном примере вы поднимитесь на три уровня (до `/`), а затем перейдете в каталог `/usr`.

Далее рассмотрим, как создавать каталоги в домашнем каталоге, перемещаться между ними и устанавливать соответствующие права доступа.

1. Перейдите в домашний каталог. Для этого введите `cd` в интерпретаторе и нажмите клавишу Enter. (Другие способы перехода в домашний каталог см. во врезке «Обозначения каталогов» далее.)

2. Чтобы убедиться, что вы находитесь в домашнем каталоге, введите `pwd`. Далее появится такая строка:

```
$ pwd
/home/joe
```

3. Создайте в домашнем каталоге новый каталог с именем `test`:

```
$ mkdir test
```

4. Проверьте права доступа каталога:

```
$ ls -ld test
drwxr-xr-x 2 joe sales 1024 Jan 24 12:17 test
```

Список показывает, что `test` — это каталог (`d`). После буквы `d` указаны права доступа (`rwxr-xr-x`), которые мы рассмотрим далее, в разделе «Владельцы и права доступа к файлам». Прочие сведения указывают на владельца (`joe`), группу (`sales`) и дату последнего изменения файлов в каталоге (24 января в 12:17).

ПРИМЕЧАНИЕ

Новый пользователь системы Fedora или Red Hat Enterprise Linux по умолчанию назначается в группу с тем же именем. В предыдущем примере пользователь joe назначен в группу joe. Данный подход к управлению группами называется личной группой пользователя (user private group, UPG).

Введите:

```
$ chmod 700 test
```

Таким образом дается право на предоставление полного доступа вам и никакого — всем остальным. (Новые права доступа читаются как `rwx------`.)

5. Сделайте каталог `test` своим текущим каталогом следующим образом:

```
$ cd test
$ pwd
/home/joe/test
```

Теперь подкаталог `test` домашнего каталога является вашим текущим рабочим каталогом. В каталог `test` можно добавлять файлы и каталоги, следуя инструкциям, приведенным далее в этой главе.

Метасимволы и операторы

Независимо от того, перечисляете вы, перемещаете, копируете, удаляете или иным образом воздействуете на файлы в системе Linux, определенные специальные символы, называемые метасимволами и операторами, помогают работать с файлами более эффективно. Метасимволы помогают обработать один или несколько файлов,

не вводя полностью имя каждого. Операторы позволяют направлять информацию из одной команды в другую или из одного файла в другой.

Метасимволы для пакетной обработки файлов

Чтобы экономить время и быстрее ссылаться на группу файлов, оболочка `bash` позволяет применять метасимволы. Всякий раз, когда нужно обратиться к файлу или каталогу, например, чтобы перечислить, открыть или удалить его, используйте соответствующие файлам метасимволы. Примеры полезных метасимволов:

- `*` — соответствует любому количеству символов;
- `?` — соответствует любому одному символу;
- `[...]` — соответствует любому из символов между скобками, которые могут включать диапазон букв или цифр, разделенных дефисом.

Чтобы обработать файлы с помощью метасимволов, сначала перейдите в пустой каталог (например, в каталог `test`, описанный в предыдущем разделе) и создайте несколько пустых файлов:

```
$ touch apple banana grape grapefruit watermelon
```

Команда `touch` создает пустые файлы. Далее показано, как использовать метасимволы с командой `ls` для пакетной обработки файлов. Выполните следующие команды и проанализируйте результаты:

```
$ ls a*
apple
$ ls g*
grape grapefruit
$ ls g*t
grapefruit
$ ls *e*
apple grape grapefruit watermelon
$ ls *n*
banana watermelon
```

Первый пример соответствует любому файлу, имя которого начинается с `a` (`apple`). Следующий соответствует всем файлам, имена которых начинается с `g` (`grape`, `grapefruit`). Далее обрабатываются файлы, имена которых начинаются с буквы `g` и заканчиваются на букву `t` (`grapefruit`). Затем обрабатываются файлы, содержащие в именах букву `e` (`apple`, `grape`, `grapefruit`, `watermelon`). Наконец, обрабатываются все файлы, имена которых в любом месте содержат букву `n` (`banana`, `watermelon`).

Примеры выборки файлов с вопросительным знаком (`?`):

```
$ ls ????e
apple grape
$ ls g???e*
grape grapefruit
```

Первый пример соответствует любому файлу из пяти символов, имена которых заканчиваются на букву e (apple, grape). Второй соответствует любому файлу, имя которого начинается с буквы g и в пятой позиции есть буква e (grape, grapefruit).

Примеры выборки с квадратными скобками:

```
$ ls [abw]*
apple banana watermelon
$ ls [agw]*[ne]
apple grape watermelon
```

В первом примере ищется любой файл, имя которого начинается с буквы a, b или w. Во втором берется любой файл, имя которого начинается с буквы a, g или w и заканчивается либо на n, либо на e. В скобки можно включать и диапазоны букв, например:

```
$ ls [a-g]*
apple banana grape grapefruit
```

В таком случае сопоставляются любые файлы, имена которых начинаются с букв от a до g.

Метасимволы перенаправления файлов

Команды получают данные после стандартного ввода и передают их на стандартный вывод. С помощью конвейеров (описаны ранее) можно направлять стандартный вывод одной команды на стандартный ввод другой. Для направления данных в файлы и из файлов используются знаки «меньше» (<) и «больше» (>). Символы перенаправления файлов:

- < — направляет содержимое файла в команду. В большинстве случаев по умолчанию команда ожидает этого действия, и здесь символ необязателен, то есть less bigfile — то же самое, что less < bigfile;
- > — направляет стандартный вывод команды в файл. Если файл существует, то его содержимое перезаписывается;
- 2> — направляет стандартную ошибку (сообщение об ошибках) в файл;
- &> — направляет в файл как стандартный вывод, так и стандартную ошибку;
- >> — направляет в файл вывод команды, добавляя его в конец существующего файла.

Далее приведены примеры, в которых информация направляется в файлы и из файлов:

```
$ mail root < ~/.bashrc
$ man chmod | col -b > /tmp/chmod
$ echo "I finished the project on $(date)" >> ~/projects
```

В первом примере содержимое файла .bashrc, находящегося в домашнем каталоге, отправляется в виде сообщения на почту суперпользователю. Вторая

командная строка форматирует справочную страницу chmod (с помощью команды man), удаляет лишние пробелы (col -b) и отправляет выходные данные в файл /tmp/chmod (стирая предыдущий файл /tmp/chmod, если он существует). После выполнения последней команды в файл проекта пользователя добавляется следующий текст:

```
I finished the project on Sat Jun 15 13:46:49 EDT 2019
```

Другой тип перенаправления, называемый *встроенным документом* (*here*-документом), позволяет вводить текст, который используется в качестве стандартного ввода для команды. Встроенные документы включают в себя ввод двух символов «меньше» (<<) после команды, за которой следует слово. Весь текст, набранный после этого слова, принимается в качестве пользовательского ввода до тех пор, пока слово не повторится в строке само по себе, например:

```
$ mail root cnegus rjones bdecker << thetext
> I want to tell everyone that there will be a 10 a.m.
> meeting in conference room B. Everyone should attend.
>
> -- James
> thetext
$
```

В этом примере сообщения отправляются на имена пользователей root, cnegus, rjones и bdecker. Текст, введенный между <<thetex и thetex, становится содержанием сообщения. Обычно встроенный документ используется в текстовом редакторе для создания или добавления информации в файл из скрипта:

```
/bin/ed /etc/resolv.conf <<resendit
a
nameserver 100.100.100.100
.
w
q
resendit
```

С помощью этих строк в скрипте, выполняемом суперпользователем, текстовый редактор ed добавляет IP-адрес DNS-сервера в файл /etc/resolv.conf.

Применение фигурных скобок

С помощью фигурных скобок ({}) можно использовать группу символов для имен файлов, каталогов и других аргументов, которые даются командам. Например, группа файлов с именами от memo1 до memo5 создается так:

```
$ touch memo{1,2,3,4,5}
$ ls
memo1 memo2 memo3 memo4 memo5
```

Значения в скобках не обязательно должны быть числами или даже однозначными цифрами. Например, можно применять диапазоны чисел или цифр, а также любую строку символов, разделив их запятыми, например:

```
$ touch {John,Bill,Sally}-{Breakfast,Lunch,Dinner}
$ ls
Bill-Breakfast Bill-Lunch John-Dinner Sally-Breakfast Sally-Lunch
Bill-Dinner John-Breakfast John-Lunch Sally-Dinner
$ rm -f {John,Bill,Sally}-{Breakfast,Lunch,Dinner}
$ touch {a..f}{1..5}
$ ls
a1 a3 a5 b2 b4 c1 c3 c5 d2 d4 e1 e3 e5 f2 f4
a2 a4 b1 b3 b5 c2 c4 d1 d3 d5 e2 e4 f1 f3 f5
```

В первом примере два набора фигурных скобок означают, что у пользователей `John`, `Bill` и `Sally` есть свои файлы, связанные с файлами `Breakfast`, `Lunch` и `Dinner`. Если бы я ошибся, то легко вспомнил бы команду и изменил `touch` на `rm -f` — и удалил все файлы. В следующем примере две точки между буквами и цифрами указывают на диапазоны, которые будут применяться. Обратите внимание на файлы, которые были созданы из этих нескольких символов.

Перечисление файлов и каталогов

Команда `ls` наиболее широко распространена и используется для перечисления информации о файлах и каталогах. Множество параметров, применяемых с командой `ls`, позволяет выводить различные списки файлов и каталогов, а также просматривать информацию о них.

По умолчанию при вводе команды `ls` выводятся все нескрытые файлы и каталоги, содержащиеся в текущем каталоге. Однако при вводе `ls` многие системы Linux (включая Fedora и RHEL) назначают псевдоним `ls` для добавления параметров. Чтобы проверить, является ли `ls` псевдонимом, введите следующее:

```
$ alias ls
alias ls='ls --color=auto'
```

Параметр `--color=auto` позволяет отображать различные типы файлов и каталогов разными цветами. Итак, вернитесь в каталог `$HOME/test`, созданный ранее, добавьте несколько различных типов файлов, а затем посмотрите, как они выводятся с помощью команды `ls`:

```
$ cd $HOME/test
$ touch scriptx.sh apple
$ chmod 755 scriptx.sh
$ mkdir Stuff
$ ln -s apple pointer_to_apple
$ ls
apple pointer_to_apple scriptx.sh Stuff
```

На примере в книге не видно, что каталог `Stuff` отображается синим цветом, `pointer_to_apple` (символическая ссылка) — голубым и `scriptx.sh` (исполняемый файл) — зеленым. Все остальные обычные файлы отображаются черным цветом. Введите команду `ls -l`, чтобы отобразить длинный список этих файлов и разделить их на типы еще более понятным образом:

```
$ ls -l
total 4
-rw-rw-r--. 1 joe joe 0 Dec 18 13:38 apple
lrwxrwxrwx. 1 joe joe 5 Dec 18 13:46 pointer_to_apple -> apple
-rwxr-xr-x. 1 joe joe 0 Dec 18 13:37 scriptx.sh
drwxrwxr-x. 2 joe joe 4096 Dec 18 13:38 Stuff
```

При просмотре длинного списка обратите внимание на то, что первый символ каждой строки указывает на тип файла. Дефис (`-`) обозначает обычный файл, `d` — каталог, а `l` (строчная L) — символическую ссылку. Исполняемый файл (скрипт или двоичный файл, который выполняется как команда) имеет включенные флаги исполнения (`x`). Подробнее о флагах исполнения читайте в разделе «Владельцы и права доступа к файлам».

Далее необходимо ознакомиться с содержимым своего домашнего каталога. Используйте параметры `-l` и `-a` с командой `ls`:

```
$ ls -la /home/joe
total 158
drwxrwxrwx 2     joe   sales   4096 May 12 13:55 .
drwxr-xr-x 3     root  root    4096 May 10 01:49 ..
-rw------- 1     joe   sales   2204 May 18 21:30 .bash_history
-rw-r--r-- 1     joe   sales     24 May 10 01:50 .bash_logout
-rw-r--r-- 1     joe   sales    230 May 10 01:50 .bash_profile
-rw-r--r-- 1     joe   sales    124 May 10 01:50 .bashrc
drw-r--r-- 1     joe   sales   4096 May 10 01:50 .kde
-rw-rw-r-- 1     joe   sales 149872 May 11 22:49 letter

^          ^     ^     ^     ^     ^             ^
col 1  col 2 col 3 col 4 col 5 col 6         col 7
```

Отображение длинного списка (параметр `-l`) с содержимым домашнего каталога дает больше информации о размерах файлов и каталогов. В строке `total` показан общий объем диска, занимаемый файлами, входящими в список (в данном примере 158 Кбайт). При добавлении параметра (`-a`) отображаются файлы, имена которых начинаются с точки (`.`). Каталоги, например текущий (`.`) и родительский (`..`) — каталог выше текущего, отмечаются буквой `d` в начале каждой записи. Все каталоги начинаются с буквы `d`, а каждый файл — с дефиса (`-`).

Имена файлов и каталогов приведены в столбце 7. В данном примере точка (`.`) указывает на каталог `/home/joe`, а две точки (`..`) — на `/home` — родительский каталог `/joe`. Большинство файлов в этом примере — это скрытые файлы (`.`), которые используются для хранения свойств графического интерфейса (каталог `.kde`) или

свойств оболочки (файлы `.bash`). Единственный файл без точек в этом списке — это файл со словом `letter`. Столбец 3 отображает каталог или владельца файла. Каталог `/home` принадлежит суперпользователю, а все остальное — пользователю `joe`, который относится к группе `sales` (группы перечислены в столбце 4).

В дополнение к `d` или `-` столбец 1 содержит права доступа, установленные для этого файла или каталога. Другая информация в списке — это несколько жестких ссылок на элемент (столбец 2), размер каждого файла в байтах (столбец 5), а также дата и время последнего изменения каждого файла (столбец 6).

Вот еще несколько фактов о списках файлов и каталогов.

- Количество символов, показанных в строке каталога (в примере 4096 байт), отражает размер файла, содержащего информацию о каталоге. Хотя это число может быть больше 4096 байт для каталога с бóльшим количеством файлов, это число не отражает размер файлов, содержащихся в каталоге.
- Формат столбца даты и времени может меняться. Вместо `May 12` дата может отображаться как `2019-05-12` в зависимости от дистрибутива и языковой настройки (переменная `LANG`).
- Иногда вместо флага исполнения (`x`), установленного в исполняемом файле, может быть указана буква `s`. Если атрибут `s` добавлен к правам владельца (`-rwsr-xr-x`), группы (`-rwsr-xr-x`) или их обоих (`-rwsr-xr-x`), приложение может быть запущено любым пользователем, но право собственности на идущий процесс остается у пользователя/группы, а не у того, кто запустил команду. Это называется флагами *set UID* или *set GID* (установка ID пользователя во время выполнения). Например, команда `mount` имеет установленные права доступа `-rwsr-xr-x`. Это позволяет любому пользователю запускать `mount` для отображения списка смонтированных файловых систем (хотя в большинстве случаев необходимо быть суперпользователем, чтобы применять `mount` для фактического монтирования файловых систем из командной строки).
- Если в конце каталога указан символ `t`, это говорит о том, что для этого каталога установлен атрибут *sticky bit* (например, `drwxrwxr-t`). Установив атрибут sticky bit на каталоге, владелец этого каталога может разрешить другим пользователям и группам добавлять файлы в каталог, но запретить удалять в нем файлы друг друга. С помощью флага set GID, назначенного каталогу, все созданные в нем файлы назначаются группе каталога. (Если в каталоге вместо флага исполнения установлена прописная буква `S` или `T`, это означает, что были установлены флаг set GID или атрибут sticky bit, но по какой-то причине флаг исполнения не был подключен.)
- Если в конце строки прав доступа указан знак плюс (например, `-rw-rw-r--+`), это значит, что к файлу применены расширенные атрибуты (`+`), такие как списки управления доступом (ACL). Точка в конце (`.`) указывает на примененный к файлу атрибут SELinux.

Обозначения каталогов

Если нужно обозначить домашний каталог в командной строке интерпретатора, введите:

- `$HOME` — эта переменная окружения отображает имя домашнего каталога;
- `~` — тильда (`~`) указывает на домашний каталог в командной строке. Ее можно использовать также для того, чтобы идентифицировать чужой домашний каталог. Например, `~joe` указывает на домашний каталог пользователя `joe` (вероятно, `/home/joe`). Если бы я хотел перейти в каталог `/home/joe/test`, то ввел бы `cd ~joe/test`.

Другие специальные способы обозначения каталогов:

- `.` — точка (`.`) отображает текущий каталог;
- `..` — две точки (`..`) относятся к каталогу, расположенному непосредственно над текущим каталогом;
- `$PWD` — эта переменная окружения ссылается на текущий рабочий каталог;
- `$OLDPWD` — эта переменная окружения ссылается на предыдущий рабочий каталог до смены его на текущий. (Ввод команды `cd` возвращает пользователя в каталог, представленный в виде `$OLDPWD`.)

Как уже упоминалось, существует много полезных параметров для команды `ls`. Вернемся к каталогу `$HOME/test`. Далее показано несколько примеров параметров для `ls`. Не волнуйтесь, если в вашем случае результаты не совсем соответствуют тому, что находится в вашем каталоге.

Любой файл или каталог, начинающийся с точки (`.`), считается скрытым и не отображается с помощью команды `ls`. Эти dot-файлы обычно являются файлами конфигурации или каталогами, которые находятся в домашнем каталоге, но должны быть скрыты. Параметр `-a` позволяет увидеть их.

Параметр `-t` отображает файлы в том порядке, в котором они были изменены в последний раз. При использовании параметра `-F` в конце имен каталогов указывается обратная косая черта (`/`), звездочка (`*`) добавляется к исполняемым файлам, а знак `@` отображается рядом с символическими ссылками.

Вот пример того, как показать скрытые и нескрытые файлы:

```
$ ls -a
. apple docs grapefruit pointer_to_apple .stuff watermelon
.. banana grape .hiddendir script.sh .tmpfile
```

Пример того, как перечислить все файлы по дате последнего изменения:

```
$ ls -at
.tmpfile .hiddendir .. docs watermelon banana script.sh
. .stuff pointer_to_apple grapefruit apple grape
```

Пример того, как перечислить файлы и добавить индикаторы типа файлов:

```
$ ls -F
apple banana docs/ grape grapefruit pointer_to_apple@ script.sh*
watermelon
```

Чтобы не показывать определенные файлы и каталоги при запуске команды `ls`, используйте параметр `-hide=`. В приведенных далее примерах любой файл, имя которого начинается с буквы `g`, не отображается в выводе. Параметр `-d` показывает информацию о каталоге вместо того, чтобы выводить содержащиеся в нем файлы и каталоги. Параметр `-R` перечисляет все файлы в текущем каталоге, а также любые файлы или каталоги, связанные с исходным каталогом. Параметр `-S` перечисляет файлы по размеру.

Пример того, как исключить из списка файлы, имена которых начинаются с буквы `g`:

```
$ ls --hide=g*
apple banana docs pointer_to_apple script.sh watermelon
```

Пример того, как вывести информацию о каталоге вместо содержащихся в нем файлов:

```
$ ls -ld $HOME/test/
drwxrwxr-x. 4 joe joe 4096 Dec 18 22:00 /home/joe/test/
```

Пример создания нескольких уровней вложенности каталогов (нужен параметр `-p`):

```
$ mkdir -p $HOME/test/documents/memos/
```

Пример того, как рекурсивно перечислить все файлы и каталоги из текущего каталога сверху вниз:

```
$ ls -R
...
```

Пример того, как перечислить файлы по размеру.

```
$ ls -S
...
```


Владельцы и права доступа к файлам

Во время работы с Linux вы, скорее всего, успели получить сообщение об ошибке Permission denied (Отказано в доступе). Права доступа, связанные с файлами и каталогами, были разработаны для защиты важных системных файлов, а также для того, чтобы пользователи не могли получить доступ к личным файлам других пользователей.

Девять флагов, назначенных каждому файлу, определяют доступ к нему. Биты прав доступа выглядят как `-rwxrwxrwx`. Они используются для определения того, кто может читать, записывать или исполнять файл.

ПРИМЕЧАНИЕ

У обычного файла перед девятибитным индикатором прав доступа указывается дефис. Вместо него можно встретить также символ d (для каталога), l (для символьной ссылки), b (для блочного устройства), c (для символьного устройства), s (для сокета) или p (для именованного конвейера).

Существует девять бит (флагов) прав доступа: первые три определяют права для владельца, следующие три — права для основной группы пользователя, а последние три — для всех остальных пользователей в системе. Для файлов `r` обозначает право на чтение файла, `w` разрешает запись в файл, `x` позволяет исполнить файл. Если вместо буквы указан дефис, это означает, что для соответствующего бита чтения, записи или исполнения право доступа отключено.

Поскольку файлы и каталоги представляют собой различные типы элементов, чтение, запись и выполнение отдельных операций над файлами и каталогами дают разные результаты. В табл. 4.2 объясняется, что означают типы прав доступа.

Таблица 4.2. Установка прав на чтение, запись и исполнение

Право	Файл	Каталог
Чтение	Просмотр того, что находится в файле	Просмотр того, какие файлы и подкаталоги содержит каталог
Запись	Изменение содержимого файла, его переименование или удаление	Добавление файлов и подкаталогов в каталог. Удаление файлов и подкаталогов из каталога
Исполнить	Запуск файла как программы	Переход в текущий каталог, поиск по каталогу или исполнение программы из этого каталога. Доступ к метаданным (размер файла, дата создания и т. д.) файлов в этом каталоге

Как отмечалось ранее, чтобы увидеть права доступа к любому файлу или каталогу, введите команду `ls -ld`. Именованные файлы и каталоги отображаются так же, как показано в этом примере:

```
$ ls -ld ch3 test
-rw-rw-r-- 1 joe sales 4983 Jan 18 22:13 ch3
drwxr-xr-x 2 joe sales 1024 Jan 24 13:47 test
```

Первая строка показывает, что файлу `ch3` присвоены права на чтение и запись для владельца и группы. Все остальные пользователи имеют право на чтение, то есть им разрешается просматривать файл, но они не могут изменить его содержимое или удалить его. Во второй строке показан каталог `test` (обозначается буквой `d`

перед битами прав доступа). Владелец имеет право на чтение, запись и выполнение, в то время как группа и другие пользователи — только на чтение и выполнение. То есть владелец может добавлять, изменять или удалять файлы в этом каталоге, а все остальные — только читать содержимое, переходить в этот каталог и перечислять его содержимое. (Если бы не было параметров `-d` для `ls`, были бы указаны файлы в тестовом каталоге вместо прав доступа к этому каталогу.)

Изменение прав доступа командой chmod с помощью чисел

Чтобы изменить права доступа к файлу, можно воспользоваться командой `chmod`. В таком случае каждому праву доступа (чтение, запись и выполнение) присваивается номер — `r=4`, `w=2` и `x=1` соответственно, то есть для задания прав доступа используются наборы чисел. Например, чтобы установить полные права доступа для себя как владельца, необходимо определить первое число — `7` (4 + 2 + 1), а затем дать группе и другим пользователям право только на чтение, указав второе и третье числа — `4` (4 + 0 + 0), чтобы в итоге получилось число `744`. Любая комбинация прав доступа включает числа от 0 (нет прав доступа) до 7 (полные права доступа).

Вот примеры того, как изменить права доступа к файлу (с именем `file`) и как это будет выглядеть.

Установим с помощью команды `chmod` права доступа `rwxrwxrwx`:

```
# chmod 777 file
```

Установим с помощью команды `chmod` права доступа `rwxr-xr-x`:

```
# chmod 755 file
```

Установим с помощью команды `chmod` права доступа `rw-r--r--`:

```
# chmod 644 file
```

Установим с помощью команды `chmod` права доступа `---------`:

```
# chmod 000 file
```

Команда `chmod` может использоваться и рекурсивно. Например, предположим, что вы хотите предоставить права доступа `755` (`rwxr-xr-x`) на всю структуру каталогов, начиная с каталога `$HOME/myapp`. Для этого задействуйте параметр `-R` следующим образом:

```
$ chmod -R 755 $HOME/myapps
```

Все файлы и каталоги, расположенные ниже, включая каталог `myapp` в домашнем каталоге, будут иметь установленные права доступа `755`. Поскольку числовой подход к настройке прав доступа изменяет все биты прав доступа одновременно, чаще всего для повторного изменения битов в большом наборе файлов используются буквы.

Изменение прав доступа командой chmod с помощью букв

Можно определять права доступа к файлам, применяя знаки плюс (+) и минус (-), а также буквы, указывающие на изменения. С помощью букв для каждого файла можно изменить права доступа для пользователя (u), группы (g), других (o) и всех пользователей (a). Биты прав доступа включают в себя право на чтение (r), запись (w) и выполнение (x). Начните с файла, у которого все права доступа открыты (rwxrwxrwx). Выполните следующие команды chmod, используя параметры со знаком минус. Полученные права доступа отображаются справа от каждой команды.

Установим с помощью команды chmod права доступа r-xr-xr-x:

```
$ chmod a-w file
```

Установим с помощью команды chmod права доступа rwxrwxrwx-:

```
$ chmod o-x file
```

Установим с помощью команды chmod права доступа rwx------:

```
$ chmod go-rwx file
```

Таким же образом выполнены примеры с недоступными правами доступа (---------). Знак плюс используется с командой chmod, чтобы установить права доступа.

Установим с помощью команды chmod права доступа rw-------:

```
$ chmod u+rw files
```

Установим с помощью команды chmod права доступа --x----x:

```
$ chmod a+x files
```

Установим с помощью команды chmod права доступа r-xr-x---:

```
$ chmod ug+rx files
```

Буквы для рекурсивного изменения прав доступа с помощью команды chmod обычно удобнее, чем числа, так как буквенные биты прав доступа можно менять выборочно. Например, вам необходимо отменить право на запись для «прочих пользователей» без изменения каких-либо других битов прав доступа для набора файлов и каталогов. Чтобы это сделать, можно ввести:

```
$ chmod -R o-w $HOME/myapps
```

В этом примере рекурсивно удаляются права на запись для «прочих пользователей» в любые файлы и каталоги ниже каталога myapps. С числом 644 право на выполнение было бы отключено для каталогов, а с числом 755 — включено для обычных файлов. При использовании o-w отключается только один бит, остальные биты остаются неизменными.

Настройка прав доступа к файлу по умолчанию с помощью umask

Когда вы создаете файл как обычный пользователь, ему по умолчанию назначаются права доступа `rw-rw-r--`. Каталогу назначаются права доступа `rwxrwxr-x`.

Для суперпользователя права доступа к файлам и каталогам выглядят так: `rw-r--r--` и `rwxr-xr-x` соответственно. Такие значения по умолчанию определяются значением команды `umask`. Введите команду `umask`, чтобы увидеть ее значение, например:

```
$ umask
0002
```

Если игнорируется начальный ноль, значение `umask` маскирует то, что считается полностью открытыми правами доступа для файла, `666`, или каталога, `777`. Значение `umask 002` приводит к установке прав доступа для каталога `775` (`rwxrwxr-x`). Та же самая команда `umask` приводит к установке прав доступа для файла `644` (`rw-rw-r--`). (Права на выполнение по умолчанию отключены для обычных файлов.)

Чтобы временно изменить значение `umask`, выполните команду `umask`. Затем попробуйте создать несколько файлов и каталогов, чтобы увидеть, как значение `umask` влияет на установку прав доступа, например:

```
$ umask 777 ; touch file01 ; mkdir dir01 ; ls -ld file01 dir01
d---------. 2 joe joe 6 Dec 19 11:03 dir01
----------. 1 joe joe 0 Dec 19 11:02 file01
$ umask 000 ; touch file02 ; mkdir dir02 ; ls -ld file02 dir02
drwxrwxrwx. 2 joe joe 6 Dec 19 11:00 dir02/
-rw-rw-rw-. 1 joe joe 0 Dec 19 10:59 file02
$ umask 022 ; touch file03 ; mkdir dir03 ; ls -ld file03 dir03
drwxr-xr-x. 2 joe joe 6 Dec 19 11:07 dir03
-rw-r--r--. 1 joe joe 0 Dec 19 11:07 file03
```

Если нужно навсегда изменить значение `umask`, добавьте команду `umask` в файл `.bashrc` в домашнем каталоге (ближе к концу этого файла). В следующий раз, когда вы запустите оболочку, `umask` будет установлена на любое выбранное вами значение.

Смена владельца файла

Как обычный пользователь, вы не можете изменять владельца файлов или каталогов. Однако, будучи суперпользователем, *можете* это сделать. Предположим, что вы создали файл `memo.tx` в домашнем каталоге пользователя `joe` как суперпользователь. Вот как можно сделать `joe` владельцем этого файла:

```
# chown joe /home/joe/memo.txt
# ls -l /home/joe/memo.txt
-rw-r--r--. 1 joe root 0 Dec 19 11:23 /home/joe/memo.txt
```

Обратите внимание на то, что команда chown изменила пользователя на joe, но оставила группу суперпользователя. Чтобы изменить и пользователя, и группу на joe, нужно ввести следующее:

```
# chown joe:joe /home/joe/memo.txt
# ls -l /home/joe/memo.txt
-rw-r--r--. 1 joe joe 0 Dec 19 11:23 /home/joe/memo.txt
```

Команду chown также можно использовать рекурсивно. Рекурсивный параметр (-R) применяется, если нужно изменить всю структуру каталогов, сделав их собственностью конкретного пользователя. Например, чтобы передать полное владение содержимым USB-накопителя, который смонтирован в каталоге /media/myusb, пользователю joe, введите следующее:

```
# chown -R joe:joe /media/myusb
```

Перемещение, копирование и удаление файлов

Команды для перемещения, копирования и удаления файлов довольно просты. Чтобы переместить файл, используйте команду mv. Чтобы скопировать файл, примените команду cp. Чтобы удалить файл, выполните команду rm. Эти команды можно использовать с отдельными файлами и каталогами или рекурсивно со множеством файлов сразу, например:

```
$ mv abc def
$ mv abc ~
$ mv /home/joe/mymemos/ /home/joe/Documents/
```

Первая команда mv перемещает файл abc в файл def в том же каталоге (по существу, переименовывая его), тогда как вторая команда перемещает файл abc в домашний каталог (~). Следующая команда mv перемещает каталог mymemo (и все его содержимое) в каталог /home/joe/Documents.

По умолчанию команда mv перезаписывает все имеющиеся файлы, если файл назначения существует. Однако многие системы Linux используют псевдоним команды mv с параметром –i, который заставляет mv запрашивать согласие пользователя перед перезаписью существующих файлов. Вот так можно проверить, делает ли это ваша система:

```
$ alias mv
alias mv='mv -i'
```

Примеры применения команды cp, которая копирует файлы из одного места в другое:

```
$ cp abc def
$ cp abc ~
```

```
$ cp -r /usr/share/doc/bash-completion* /tmp/a/
$ cp -ra /usr/share/doc/bash-completion* /tmp/b/
```

Первая команда (`cp`) копирует файл `abc` в новую папку с именем `def` в том же каталоге, а вторая копирует файл `abc` в домашний каталог (`~`), сохраняя изначальное имя. Две рекурсивные копии (`-r`) копируют каталог `bash-completion` и все файлы, которые он содержит, в новые каталоги `/tmp/a/` и `/tmp/b/`. Если запустить `ls -l` в этих двух каталогах, то для команды `cp`, выполняемой с параметром `-a`, метки даты/времени и права доступа наследуются. Без параметра `-a` используются текущие метки даты и времени, а права доступа определяются командой `umask`.

Команда `cp`, как правило, также является псевдонимом с параметром `v` и предотвращает случайную перезапись файлов.

Как и у команд `cp` и `mv`, у `rm` обычно есть псевдоним для параметра `-i`. Это помогает предотвратить проблему непреднамеренного рекурсивного удаления (`-r`). Примеры использования команды `rm`:

```
$ rm abc
$ rm *
```

Первая команда удаляет файл `abc`, вторая удаляет все файлы в текущем каталоге (но не трогает каталоги и/или любые файлы, имена которых начинаются с точки). Если нужно удалить каталог, задействуйте рекурсивный параметр `-r` для `rm`, а для пустого каталога можно взять команду `rmdir`, например:

```
$ rmdir /home/joe/nothing/
$ rm -r /home/joe/bigdir/
$ rm -rf /home/joe/hugedir/
```

Команда `rmdir` в этом примере удаляет каталог (`nothing`), только если он пуст. Команда `rm -r` удаляет каталог `bigdir` и все его содержимое (файлы и несколько уровней подкаталогов), каждый раз запрашивая разрешение пользователя. При использовании параметра `-f` каталог `hugedir` и все его содержимое немедленно удаляются без запроса.

ВНИМАНИЕ!

Когда параметр -i применяется с командами mv, cp и rm, возникает риск удаления некоторых (или даже многих) файлов по ошибке. С подстановочными знаками (например, *) и без -i ошибки еще более вероятны. Тем не менее это не всегда актуально при удалении файлов. Вот что можно сделать.

- Как уже отмечалось, с параметром -f можно принудить команду rm удалять файлы без запроса. Можно также запустить команду rm, cp или mv с обратной косой чертой перед ней (\rm bigdir). Обратная косая черта приводит к тому, что любая команда выполняется без псевдонимов.
- Другой вариант — использовать параметр -b с командой mv. В таком случае, если файл с таким же именем существует в месте назначения, создается резервная копия старого файла перед перемещением нового.

Резюме

Команды перемещения по файловой системе, копирования, перемещения и удаления файлов являются одними из основных команд, применяемых в оболочке. В этой главе мы рассмотрели множество команд перемещения и манипулирования файлами, а также команд, которые позволяют изменять владельца и устанавливать права доступа.

В следующей главе описываются команды редактирования и поиска файлов, а именно текстовые редакторы `vim/vi`, команды `find` и `grep`.

Упражнения

Выполните упражнения, чтобы проверить знание эффективных способов работы в файловой системе Linux и действий с файлами и каталогами. Старайтесь как можно меньше использовать сочетания клавиш для ввода, чтобы добиться желаемых результатов. Задачи подходят для систем Fedora и Red Hat Enterprise Linux (некоторые сработают и в других системах Linux).

Если затрудняетесь с решением заданий, воспользуйтесь ответами к упражнениям, приведенным в приложении Б (хотя Linux позволяет решать задачи разными способами).

1. В своем домашнем каталоге создайте папку `projects`. В ней создайте девять пустых файлов с именами `house1`, `house2`, `house3` и т. д вплоть до `house9`. Предположим, в этом каталоге есть много других файлов. Придумайте один аргумент для `ls`, который будет перечислять только эти девять файлов.
2. Создайте путь к каталогу `$HOME/projects/houses/doors/`. Создайте пустые файлы на пути к каталогу (используйте абсолютный и относительный пути из домашнего каталога):

   ```
   $HOME/projects/houses/bungalow.txt
   $HOME/projects/houses/doors/bifold.txt
   $HOME/projects/outdoors/vegetation/landscape.txt
   ```

3. Скопируйте файлы `house1` и `house5` в каталог `$HOME/projects/houses/`.
4. Рекурсивно скопируйте каталог `/usr/share/doc/initscripts*` в домашний каталог `$HOME/projects/`. Сохраните текущие метки даты/времени и права доступа.
5. Рекурсивно перечислите содержимое каталога `$HOME/projects/`. Передайте выходные данные в команду `less`, чтобы просмотреть их на странице.
6. Удалите файлы `house6`, `house7` и `house8` без запроса от команды.
7. Переместите `house3` и `house4` в каталог `$HOME/projects/houses/doors`.

8. Удалите каталог `$HOME/projects/houses/doors` и его содержимое.
9. Измените права доступа к файлу `$HOME/projects/house2` таким образом, чтобы владелец файла мог его прочитать и записать, а группа и прочие пользователи — только прочитать.
10. Рекурсивно измените права доступа к каталогу `$HOME/projects/` так, чтобы никто не имел прав на запись в любые файлы или каталоги, расположенные ниже этого уровня в файловой системе.

5 Работа с текстовыми файлами

В этой главе

- Использование программ `vim` и `vi` для редактирования текстовых файлов.
- Поиск файлов.
- Поиск в файлах.

Когда была создана система UNIX, бóльшая часть обмена данными в системе осуществлялась в виде простых текстовых файлов. Было крайне важно, чтобы пользователи знали, как использовать инструменты для поиска обычных текстовых файлов, а также могли изменять и настраивать их.

Сегодня настройки систем Linux все еще можно выполнять путем редактирования простых текстовых файлов. Независимо от того, изменяете ли вы файлы в каталоге `/etc` для настройки локальной службы или редактируете файлы инвентаризации Ansible, чтобы настроить хост-компьютеры, обычные текстовые файлы по-прежнему широко используются для подобных задач.

Прежде чем стать полноценным системным администратором, необходимо научиться пользоваться простым текстовым редактором. То, что большинство профессиональных серверов Linux не имеют доступного графического интерфейса, требует умения редактировать текстовые файлы конфигураций с помощью консольного текстового редактора.

После того как вы научитесь редактировать текстовые файлы, вам все равно может быть трудно понять, где находятся файлы, которые нужно отредактировать. С помощью таких команд, как `find`, можно искать файлы на основе различных атрибутов (имя файла, размер, дата изменения и владелец). С помощью команды `grep` можно проводить поиск контента внутри текстовых файлов.

Редактирование файлов в программах vim и vi

При использовании Linux практически невозможно обойтись без текстового редактора, потому что, как отмечалось ранее, большинство файлов конфигурации в Linux — это обычные текстовые файлы, которые почти наверняка хоть раз придется изменить вручную.

В GNOME существует интуитивно понятный текстовый редактор `gedit` (введите `gedit` в поле поиска и нажмите клавишу Enter или выберите команду меню Applications ▸ gedit (Приложения ▸ Текстовый редактор)). Можно также запустить простой текстовый редактор из командного интерпретатора. Однако большинство пользователей Linux для редактирования текстовых файлов применяют именно команды `vi` или `emacs`.

Преимущество инструментов `vi` и `emacs` перед графическим редактором заключается в том, что их можно использовать из любых оболочки, терминала или символьного соединения по сети (например, с помощью `telnet` или `ssh`), где графический интерфейс не требуется. Каждый из них также содержит массу функций, которые полезно изучить.

В следующих разделах будет описана работа с текстовым редактором `vi`, который используется для неавтоматизированного редактирования текстового файла из любой оболочки. Будет рассказано про улучшенные версии `vi`, называемые `vim`. (Если вам не подходит `vi`, изучите врезку «Другие текстовые редакторы».)

Другие текстовые редакторы

Есть десятки текстовых редакторов для операционной системы Linux. Некоторые из них могут быть установлены в вашем дистрибутиве Linux. Попробуйте их, если редактор `vi` покажется слишком сложным.

- **nano** — популярный оптимизированный текстовый редактор, который используется со многими системами Linux и другими ее средами, в которых пространство ограничено. Например, он позволяет редактировать текстовые файлы в процессе установки Gentoo Linux.
- **gedit** — текстовый редактор с поддержкой интерфейса GNOME.
- **jed** — текстовый редактор с графическим интерфейсом, созданный специально для программистов. Он подсвечивает код различными цветами и выделяет в нем синтаксические ошибки. Клавиша Alt применяется для управления текстом.
- **joe** — редактор `joe` похож на многие текстовые редакторы для ПК. Используйте клавишу Ctrl и стрелки для перемещения. Нажмите сочетание клавиш Ctrl+C для выхода без сохранения или Ctrl+X для сохранения при выходе.
- **kate** — приятный на вид редактор включен в пакет `kdebase`. У него есть множество наворотов, таких как подсветка синтаксиса для различных типов языков программирования и элементы управления для переноса слов.

- **kedit** — графический текстовый редактор, используется рабочим столом KDE.
- **mcedit** — в этом редакторе функциональные клавиши помогают перемещаться по тексту, сохранять, копировать, перемещать и удалять его. Как `jed` и `joe`, это экранный редактор. Он задействуется в пакете `mc` для RHEL и Fedora.
- **nedit** — отличный редактор для программистов. Для его установки необходимо установить дополнительный пакет `nedit`.

Если вы подключаетесь к другим компьютерам по сети с помощью `ssh`, то можете использовать любой доступный текстовый редактор. Если же для подключения к удаленной системе указываете `ssh -X`, на локальном экране появится графический редактор. Если графический интерфейс недоступен, то понадобится текстовый редактор в оболочке, например `vi`, `jed` или `joe`.

Редактор `vi` трудно освоить, но, хорошо ознакомившись с ним, вы сможете быстро и эффективно редактировать файлы и перемещаться внутри них с помощью клавиатуры.

Редактор vi

Чаще всего `vi` запускается, если нужно открыть определенный файл. Например, чтобы открыть файл с именем `/tmp/test`, введите следующую команду:

```
$ vi /tmp/test
```

Если это новый файл, появится что-то похожее на следующее:

```
□
~
~
~
~
~
"/tmp/test" [New File]
```

Мигающий прямоугольник сверху показывает, где находится курсор. Нижняя строка информирует о том, что происходит с редактированием (в примере был открыт новый файл). Тильды (~), стоящие между ними, — это заполнители, потому что в файле еще нет текста. А теперь самое пугающее. Здесь нет никаких подсказок, меню или значков, которые объясняли бы, что нужно делать. Что еще хуже, вы не можете просто начать печатать. Если вы это сделаете, компьютер, скорее всего, звуком оповестит об ошибке (и потом некоторые пользователи жалуются, что Linux недружелюбен).

Здесь нужно знать, что существует два основных режима работы: командный и ввода. Редактор `vi` всегда запускается в командном режиме. Прежде чем добавить

текст в файл или изменить его, необходимо ввести команду — одну или две буквы, иногда (не обязательно) с числом после них, чтобы сообщить `vi`, что нужно сделать. Регистр важен, поэтому используйте прописные и строчные буквы точно так, как показано в примерах!

ПРИМЕЧАНИЕ

В Red Hat Enterprise Linux, Fedora и других дистрибутивах Linux для обычных пользователей команда vi имеет псевдоним для запуска vim. Если вы наберете alias vi, то увидите alias vi='vim'. Первое очевидное различие между vi и vim заключается в том, что любой известный тип текстового файла, например HTML, C или обычный файл конфигурации, отображается в цвете. Цвета указывают на структуру файла. Другие функции vim, которых нет в vi, включают в себя визуальную подсветку синтаксиса и режим разделения экрана. По умолчанию суперпользователь не имеет псевдонима vim. Если в вашей системе отсутствует vim, установите соответствующий пакет.

Добавление текста

Чтобы войти в режим ввода, укажите букву команды ввода. Для начала наберите любую из следующих букв. После ввода текста нажмите клавишу Esc (иногда дважды), чтобы вернуться в командный режим. Не забывайте про клавишу Esc!

- `a` — команда *«Добавить»*. Текст вводится *справа* от курсора.
- `A` — команда *«Добавить в конце»*. Текст вводится в конце текущей строки.
- `i` — команда *«Вставить»*. Текст вводится *слева* от курсора.
- `I` — команда *«Вставить в начале»*. Текст вводится в начале текущей строки.
- `o` — команда *«Открыть ниже»*. Добавляет строку под текущей строкой и переводит в режим ввода.
- `O` — команда *«Открыть выше»*. Добавляет строку над текущей строкой и переводит в режим ввода.

СОВЕТ

В режиме ввода внизу экрана появляется надпись -- INSERT --.

Введите несколько слов и нажмите клавишу Enter. Повторите, чтобы добавить несколько строк с текстом. Когда закончите, нажмите клавишу Esc, чтобы вернуться в командный режим. Теперь перемещайтесь по тексту с помощью клавиш или букв, описанных далее.

СОВЕТ

Не забывайте про клавишу Esc! Ее нажатие всегда возвращает редактор в командный режим. Помните, что иногда необходимо нажать клавишу Esc дважды. Например, при наборе двоеточия (:) при переходе в режим ввода нужно дважды нажать клавишу Esc, чтобы вернуться в командный режим.

Перемещение по тексту

Для перемещения по тексту можно использовать клавиши ←, ↑, ↓ и →. Однако есть и более удобные клавиши, применяемые в режиме набора текста.

- Клавиши ←, ↑, ↓ и →. Перемещайте курсор вверх, вниз, влево или вправо в файле по одному символу за раз. Для перемещения влево и вправо используйте клавиши Backspace и Пробел соответственно. Если вы предпочитаете, чтобы пальцы лежали на клавиатуре, перемещайте курсор с помощью клавиш `h` (влево), `l` (вправо), `j` (вниз) или `k` (вверх).
- `w` — перемещает курсор в начало следующего слова, отделенного пробелами, отступами или знаками препинания.
- `W` — перемещает курсор в начало следующего слова, отделенного пробелами или отступами.
- `b` — перемещает курсор в начало предыдущего слова, отделенного пробелами, отступами или знаками препинания.
- `B` — перемещает курсор в начало предыдущего слова, отделенного пробелами или отступами.
- `0` (ноль) — перемещает курсор в начало текущей строки.
- `$` — перемещает курсор в конец текущей строки.
- `H` — перемещает курсор в верхний левый угол экрана (на первую строку на экране).
- `M` — перемещает курсор на первый символ средней строки на экране.
- `L` — перемещает курсор в нижний левый угол экрана (на последнюю строку на экране).

Удаление, копирование и изменение текста

Необходимо также знать, как удалять, копировать или изменять текст. Команды `x`, `d`, `y` и `c` используются для удаления и изменения текста. Они могут действовать совместно с клавишами перемещения (←, ↑, ↓ и →, PgUp, PgDn, буквы и специальные клавиши) и цифрами, чтобы точно выделить, что нужно удалить, копировать или изменить, например:

- `x` — удаляет символ под курсором;
- `X` — удаляет символ непосредственно перед курсором;
- `d<?>` — удаляет часть текста;
- `c<?>` — изменяет часть текста;
- `y<?>` — копирует часть текста.

Символы `<?>` после каждой буквы указывают место, где нужно использовать команду, чтобы выбрать то, что нужно удалить, изменить или копировать, например:

- `dw` — удаляет (`d`) слово (`w`) после курсора;
- `db` — удаляет (`d`) слово (`b`) перед курсором;

- `dd` — удаляет (`d`) всю текущую строку (`d`);
- `c$` — изменяет (`c`) символы (фактически стирает их) от текущего символа до конца строки (`$`) и переходит в режим ввода;
- `c0` — изменяет (`c`) (опять же стирает) символы с предыдущего символа до начала текущей строки (`0`) и переходит в режим ввода;
- `cl` — стирает (`c`) текущую букву (`l`) и переходит в режим ввода;
- `cc` — стирает (`c`) строку (`c`) и переходит в режим ввода;
- `yy` — копирует (`y`) текущую строку (`y`) в буфер;
- `y)` — копирует (`y`) текущее предложение (`)`) справа от курсора в буфер;
- `y}` — копирует (`y`) текущий абзац (`}`) справа от курсора в буфер.

Любая команда из приведенных ранее примеров может быть изменена с помощью чисел, например:

- `3dd` — удаляет (`d`) три (`3`) строки (`d`), начинающиеся с текущей строки;
- `3dw` — удаляет (`d`) следующие три (`3`) слова (`w`);
- `5cl` — изменяет (`c`) следующие пять (`5`) букв (`l`), то есть удаляет буквы и переходит в режим ввода;
- `12j` — перемещается вниз (`j`) на 12 строк (`12`);
- `5cw` — стирает (`c`) следующие пять (`5`) слов (`w`) и переходит в режим ввода;
- `4y)` — копирует (`y`) следующие четыре (`4`) предложения (`)`).

Вставка текста

После того как текст скопирован в буфер (путем удаления, изменения или копирования), можно поместить его обратно в файл, используя буквы `p` или `P`. Обе команды позволяют по-разному вставить в файл текст, сохраненный в буфере.

- `P` — помещает скопированный текст слева от курсора, если текст состоит из букв или слов, и над текущей строкой, если содержит строки.
- `p` — помещает текст справа от курсора, если он состоит из букв или слов, и ниже текущей строки, если содержит строки.

Повтор команд

Действие удаления, изменения или вставки текста можно повторить, введя точку (`.`). Например, если курсор находится в начале имени `Joe`, введите `cw`, а затем `Jim`, чтобы изменить `Joe` на `Jim`. Далее найдите следующее упоминание `Joe` в файле, установите курсор в начале имени и нажмите точку. Слово меняется на `Jim`, и можно продолжить поиск по тексту. Таким образом можно откорректировать весь файл, переходя к нужному слову и нажимая точку, чтобы изменить его.

Выход из редактора vi

Чтобы завершить работу, используйте следующие команды для сохранения или выхода из файла:

- `ZZ` — сохраняет текущие изменения в файле и выходит из `vi`;
- `:w` — сохраняет текущий файл, можно продолжать редактирование;
- `:wq` — делает то же самое, что и `ZZ`;
- `:q` — выходит из текущего файла. Работает только в случае, если все изменения сохранены;
- `:q!` — завершает работу с текущим файлом, не сохраняя только что внесенные в него изменения.

СОВЕТ

Если вы удалили файл по ошибке, команда :q! — лучший способ выйти из редактора и отменить все изменения. Файл вернется к последней измененной версии. Поэтому, если вы просто сохранили его с помощью :w, все сделанные до этого момента изменения сохранились. Но, несмотря на то что файл сохранен, можно ввести u и отказаться от изменений (вплоть до начала всего редактирования), а затем сохранить снова.

Мы рассмотрели несколько команд редактирования в редакторе `vi`. В следующих разделах полезных команд будет еще больше. Однако для начала дам несколько советов, чтобы упростить для вас начало взаимодействия с `vi`.

- Esc. Помните, что клавиша Esc возвращает программу в командный режим. (Я видел, как люди тыкали в каждую клавишу, пытаясь выйти из файла.) Нажмите Esc, затем введите `ZZ`, и редактор вернется в командный режим, сохранит файл и завершит работу.
- u. Нажмите u, чтобы отменить предыдущее изменение. Продолжайте нажимать u, чтобы отменить и более ранние.
- Ctrl+R. Если вы отменили лишнее, используйте сочетание клавиш Ctrl+R, чтобы вернуть отмененное. По сути, эта команда отменяет отмену.
- Caps Lock. Постарайтесь случайно не нажать клавишу Caps Lock. Все, что вводится в `vi`, имеет другое значение при наборе прописных букв. Редактор не предупреждает о смене регистра, а в тексте начинают происходить не те действия.
- `:!`*команда*. Находясь в `vi`, вы можете запустить команду оболочки, набрав `:!` после имени оболочки. Например, введите `:!date`, чтобы увидеть текущие дату и время, `:!pwd`, чтобы опознать текущий каталог, `:!jobs`, чтобы узнать о процессах, протекающих в фоновом режиме. Когда команда завершится, нажмите клавишу Enter — и вернетесь к редактированию файла. Этот метод можно использовать даже для запуска оболочки `:!bash` из `vi`. Выполните несколько команд в оболочке, а затем введите `exit`, чтобы вернуться в `vi`. (Я рекомендую

перед выходом в оболочку на всякий случай сохранять процесс — вдруг забудете вернуться к `vi`.)

- `Ctrl+g`. Если вы забыли, что именно редактируете, то нажмите эти клавиши — и в нижней части экрана отобразятся имя редактируемого файла и строка, на которой вы находитесь. Там показаны также общее количество строк в файле, процент прочтения файла и номер столбца, на котором находится курсор. Это поможет сориентироваться после похода за очередной чашкой кофе в три часа ночи.

Прокрутка файлов

Помимо команд перемещения, описанных ранее, существуют и другие способы передвигаться по файлу `vi`. Чтобы опробовать их, откройте большой файл. (Можете скопировать файл `/var/log/messages` в файл `/tmp` и открыть его в `vi`.) Примеры команд перемещения:

- `Ctrl+f` — прокрутка вперед по одной странице за раз;
- `Ctrl+b` — прокрутка назад по одной странице за раз;
- `Ctrl+d` — прокрутка вперед на половину страницы за раз;
- `Ctrl+u` — прокрутка назад на половину страницы за раз;
- `G` — переход к последней строке файла;
- `1G` — переход к первой строке файла;
- `35G` — переход к указанной строке (в данном случае 35-й).

Поиск текста

Для поиска по тексту файла вперед или назад применяйте либо косую черту (`/`), либо знак вопроса (`?`). В процессе поиска можно использовать метасимволы, например:

- `/hello` — поиск впереди слова `hello`;
- `?goodbye` — поиск позади слова `goodbye`;
- `/The.*foot` — ищет впереди строку, в которой есть слово `The`, а после него — слово `foot`;
- `?[pP]rint` — ищет позади слова `print` и `Print`. Помните, что в Linux регистр важен, поэтому используйте скобки для поиска слов, которые могут быть написаны по-разному.

Сделав поисковый запрос, введите `n`, чтобы снова выполнить поиск в том же направлении, или `N` — для поиска в противоположном.

Режим редактирования

Редактор `vi` изначально не был способен работать в полноэкранном режиме. Однако он позволяет запускать команды, которые могут находить и изменять текст в одной или нескольких строках одновременно. Если ввести двоеточие, курсор перемещается в нижнюю часть экрана и редактор переходит в режим редактирования. Далее приведены некоторые из команд `ex` для поиска и изменения текста. (Я для поиска выбрал слова `Local` и `Remote`, вы можете использовать любые другие.)

- `:g/Local` — ищет в файле слово `Local` и выводит каждое его появление. (Если данных больше, чем умещается на экран, они передаются по конвейеру в команду `more`).
- `:s/Local/Remote` — заменяет `Remote` на первое появление слова `Local` в текущей строке.
- `:g/Local/s//Remote` — заменяет первое появление слова `Local` в каждой строке файла словом `Remote`.
- `:g/Local/s//Remote/g` — во всем файле заменяет каждое появление слова `Local` словом `Remote`.
- `:g/Local/s//Remote/gp` — во всем файле заменяет каждое появление слова `Local` словом `Remote`, а затем выводит каждую строку с изменениями (через `es`, если выходные данные заполняют более одной страницы).

Подробнее о vi и vim

Чтобы узнать больше о редакторе `vi`, введите `vimtutor`. Эта команда открывает в редакторе `vim` учебник, где описываются общие команды и функции, которые можно задействовать в `vim`. Чтобы применить команду `vimtutor`, установите пакет `vim`.

Поиск файлов

Даже в базовой инсталляции Linux могут быть установлены тысячи файлов. Чтобы найти файлы в системе, используйте команды `locate` (найти команды по имени), `find` (найти файлы на основе множества различных атрибутов) и `grep` (в текстовых файлах найти строки, содержащие нужный текст).

Команда locate для поиска файлов по имени

В большинстве систем Linux, включая Fedora и RHEL, команда `updatedb` выполняется один раз в день и по всей системе собирает имена файлов в базу данных. Запустив команду `locate`, можно выполнить поиск в этой базе данных и определить расположение файлов, хранящихся в ней.

Вот что нужно знать о поиске файлов с помощью команды `locate`.

- Команда `locate` имеет и преимущества, и недостатки по сравнению с командой `find`. Она находит файлы быстрее, потому что ищет по готовой базе данных, а не по всей файловой системе. Недостатком является то, что `locate` не учитывает файлы, добавленные в систему после последнего обновления базы данных.
- Не каждый файл файловой системы хранится в базе данных. Содержимое файла `/etc/updatedb.conf` ограничивает, какие имена файлов отсекаются в зависимости от выбранных типов монтирования, типов файловой системы, типов файлов и точек монтирования. Например, не собираются имена файлов из удаленных файловых систем (`cifs`, `inf` и т. д.) или локально смонтированных CD или DVD (`iso9660`). Пути, содержащие временные файлы (`/tmp`) и файлы в очереди обработки (`/var/spool/cups`), также исключаются. При необходимости отсекаемые элементы можно добавить в базы данных `locate` или удалить из них. В дистрибутиве RHEL 8 файл `updatedb.conf` содержит:

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

Обычный пользователь не может просматривать файлы из базы данных `locate`, которые не отображаются в файловой системе. Например, если не получается ввести `ls` для просмотра файлов в каталоге `/root`, то и файлы, хранящиеся в этом каталоге, найти не получится.

- Строка, которую ищут, может находиться в любом месте пути к файлу. Например, при поиске `passwd` можно найти `/etc/passwd`, `/usr/bin/passwd`, `/home/chris/passwd/pwdfiles.txt` и многие другие файлы со словом `passwd` в пути.
- Если добавить файлы в систему после запуска `updatedb`, их нельзя будет найти до тех пор, пока `updatedb` не запустится снова (вероятно, в ту же ночь). Чтобы база данных содержала все файлы, появившиеся до текущего момента, можно снова запустить `updatedb` из оболочки от имени суперпользователя.

Пример использования команды `locate` для поиска файлов:

```
$ locate .bashrc
/etc/skel/.bashrc
/home/cnegus/.bashrc
# locate .bashrc
/etc/skel/.bashrc
```

```
/home/bill/.bashrc
/home/joe/.bashrc
/root/.bashrc
```

При запуске от имени обычного пользователя `locate` находит `bashrc` только в файле `/etc/skel` и домашнем каталоге текущего пользователя. При запуске от имени суперпользователя та же команда находит файлы `bashrc` в домашнем каталоге каждого пользователя системы.

```
$ locate dir_color
/usr/share/man/man5/dir_colors.5.gz
...
$ locate -i dir_color
/etc/DIR_COLORS
/etc/DIR_COLORS.256color
/etc/DIR_COLORS.lightbgcolor
/usr/share/man/man5/dir_colors.5.gz
```

С помощью команды `locate -i` можно найти имена файлов независимо от регистра. В предыдущем примере `DIR_COLORS` был найден именно с помощью параметра `-i`.

```
$ locate services
/etc/services
/usr/share/services/bmp.kmgio
/usr/share/services/data.kmgio
```

В отличие от команды `find`, которая использует параметр `-name` для поиска имен файлов, команда `locate` находит введенную строку, если она находится в любой части пути к файлу. В этом примере поиск `services` с помощью команды `locate` находит файлы и каталоги, содержащие текстовую строку `services`.

Поиск файлов с помощью команды find

Команда `find` лучше всего находит файлы в файловой системе на основе различных атрибутов. После того как файлы найдены, с ними можно взаимодействовать (используя параметры `-exec` или `-okay`) с помощью нужных команд.

Команда `find` выполняет поиск в реальном времени, из-за чего он идет медленнее, чем с командой `locate`, но по всей файловой системе. Можно задать поиск так, чтобы он начинался в определенной точке файловой системы: ограничив область поиска, мы ускоряем его. Практически любой атрибут файла может стать параметром для поиска. Можно искать имена файлов, владельца, права доступа, размер, время изменения и другие атрибуты, а также их комбинации. Примеры использования команды `find`:

```
$ find
$ find /etc
# find /etc
$ find $HOME -ls
```

Команда `find` запускается в отдельной строке и находит все файлы и каталоги уровнем ниже текущего. Если нужно выполнить поиск из определенной точки системы, добавьте имя каталога, который требуется найти (например, `/etc`). Обычному пользователю команда `find` не дает специальных прав на поиск файлов, которые имеет только суперпользователь. И потому он получит кучу сообщений об ошибках. Запустите поиск от имени суперпользователя, и команда `find/etc` найдет все файлы в каталоге `/etc`.

Специальный параметр для команды `find` — это `-ls`. Длинный список характеристик (владелец, права доступа, размер и т. д.) выводится с каждым файлом при использовании параметра `-ls` для команды `find` (он аналогичен выводу команды `ls -l`). Этот параметр поможет в последующих примерах, когда потребуется проверить, что нужные файлы (определенного владельца, размера, времени изменения или имеющие другие атрибуты) найдены.

ПРИМЕЧАНИЕ

Если обычный пользователь выполнит поиск в той части файловой системы, к которой у него нет полного доступа (например, в каталоге /etc), появится множество сообщений об ошибках при поиске с помощью find. Чтобы убрать сообщения, направьте стандартные ошибки в файл /dev/null. Для этого добавьте в конец командной строки 2> /dev/null. Выражение 2> перенаправляет стандартную ошибку к следующему параметру — в данном случае файлу /dev/null, в котором выходные данные отбрасываются.

Поиск файлов по имени

Чтобы найти файлы по имени, используйте параметры `-name` и `-iname`. Поиск осуществляется по базовому имени файла, имена каталогов по умолчанию не учитываются. Чтобы сделать поиск более гибким, можно применить подстановочные символы, например звездочки (`*`) и вопросительные знаки (`?`), как в следующих примерах:

```
# find /etc -name passwd
/etc/pam.d/passwd
/etc/passwd
# find /etc -iname '*passwd*'
/etc/pam.d/passwd
/etc/passwd-
/etc/passwd.OLD
/etc/passwd
/etc/MYPASSWD
/etc/security/opasswd
```

Если параметр без звездочек, как в первом примере, то будут перечислены все имеющиеся в каталоге `/etc` файлы, которые носят точное имя `passwd`. С параметром `-iname` можно сопоставить любую комбинацию в верхнем и нижнем регистрах. А используя звездочки, можно сопоставить с ним любое имя файла, содержащее слово `passwd`.

Поиск файлов по размеру

Когда диск заполняется и нужно узнать местонахождение самых больших файлов, можно выполнить поиск по размеру файла. Параметр `-size` позволяет искать файлы указанного размера, а также те, что меньше или больше него, например:

```
$ find /usr/share/ -size +10M
$ find /mostlybig -size -1M
$ find /bigdata -size +500M -size -5G -exec du -sh {} \;
4.1G /bigdata/images/rhel6.img
606M /bigdata/Fedora-16-i686-Live-Desktop.iso
560M /bigdata/dance2.avi
```

Первая строка ищет файлы более 10 Мбайт. Вторая — файлы менее 1 Мбайт. Третья — файлы от 500 Мбайт до 5 Гбайт. В примере есть параметр `-exec` (будет описан позже), который нужен для запуска команды `du`, чтобы увидеть размер каждого файла.

Поиск файлов по имени пользователя

Можно искать файлы по имени конкретного владельца (`-user`) или названию группы (`-group`). С помощью параметров `-not` и `-or` можно конкретизировать поиск файлов, связанных с определенными пользователями и группами, например:

```
$ find /home -user chris -ls
131077  4 -rw-r--r-- 1 chris   chris 379 Jun 29 2014 ./.bashrc
# find /home \( -user chris -or -user joe \) -ls
131077  4 -rw-r--r-- 1 chris   chris 379 Jun 29 2014 ./.bashrc
181022  4 -rw-r--r-- 1 joe     joe    379 Jun 15 2014 ./.bashrc
# find /etc -group ntp -ls
131438  4 drwxrwsr-x 3 root    ntp   4096 Mar 9 22:16 /etc/ntp
# find /var/spool -not -user root -ls
262100  0 -rw-rw---- 1 rpc     mail    0 Jan 27 2014 /var/spool/mail/rpc
278504  0 -rw-rw---- 1 joe     mail    0 Apr 3 2014 /var/spool/mail/joe
261230  0 -rw-rw---- 1 bill    mail    0 Dec 18 14:17 /var/spool/mail/bill
277373 2848 -rw-rw---- 1 chris mail 8284 Mar 15 2014 /var/spool/mail/chris
```

В первом примере выводится длинный список всех файлов из каталога `/home`, принадлежащих пользователю `chris`. Далее перечисляются файлы пользователей `chris` или `joe`. Команда `find` из каталога `/etc` открывает все файлы, которые имеют атрибут `ntp` в качестве первичной группы. В последнем примере показаны все файлы каталога `/var/spool`, которые не принадлежат суперпользователю. В выходных данных в примере можно увидеть файлы, принадлежащие другим пользователям.

Поиск файлов по правам доступа

Поиск файлов по правам доступа — отличный способ выявить проблемы с безопасностью или доступом в системе. Помимо файлов, где права доступа можно изменять с помощью чисел или букв (командой `chmod`), существуют и файлы, основанные на числовых или буквенных правах доступа с параметром `-perm`. (См. главу 4,

чтобы узнать, как задействовать числа и буквы с командой chmod и выводить права доступа к файлам.)

При использовании числовых прав доступа, как показано в примерах далее, помните, что эти три числа представляют собой права доступа для пользователя, группы и других лиц. Каждое из трех чисел варьируется от отсутствия прав доступа (0) до полных прав доступа на чтение/запись/выполнение (7) и образуется в результате сложения битов прав на чтение (4), запись (2) и выполнение (1). Все три указанных бита должны совпадать и иметь перед собой дефис (-), если применена косая черта (/), могут совпадать любые из чисел. Определенные целые числа должны совпадать, если не используется ни дефис, ни косая черта.

Приведу пример:

```
$ find /usr/bin -perm 755 -ls
788884  28 -rwxr-xr-x  1 root   root        28176 Mar 10 2014 /bin/echo

$ find /home/chris/ -perm -222 -type d -ls
144503  4 drwxrwxrwx   8 chris chris 4096 Jun 23 2014 /home/chris/OPENDIR
```

При поиске с параметром -perm 755 выводятся любые файлы или каталоги с точными правами доступа rwxr-xr-x. При поиске с параметром -perm -222 выводятся только файлы, имеющие права на запись для пользователя, группы и других. Обратите внимание на то, что в этом случае -type d добавляется только для соответствия каталогам.

```
$ find /myreadonly -perm /222 -type f
685035   0 -rw-rw-r-- 1 chris  chris     0 Dec 30 16:34 /myreadonly/abc

$ find . -perm -002 -type f -ls
266230   0 -rw-rw-rw- 1 chris  chris     0 Dec 30 16:28 ./LINUX_BIBLE/abc
```

Используя параметр -perm /222, можно найти любой файл (-type f), у которого есть право на запись для пользователя, группы или других. Так можно убедиться, что все файлы доступны только для чтения в определенной части файловой системы (в данном случае под каталогом /myreadonly). Последний пример, -perm /002, полезен при поиске файлов, право на запись в которые есть у других, независимо от того, как установлены другие права доступа.

Поиск файлов по дате и времени

Метки даты и времени сохраняются для каждого файла при его создании и открытии, изменении его содержимого или метаданных. Метаданные включают в себя имя владельца, название группы, отметку времени, размер файла, права доступа и другую информацию, хранящуюся в дескрипторе файла. Поиск изменений данных файлов или метаданных может понадобиться в следующих случаях.

- Вы только что изменили содержимое файла конфигурации и не можете вспомнить, какого именно. Поэтому в каталоге /etc ищете, какие файлы изменились за последние десять минут:

  ```
  $ find /etc/ -mmin -10
  ```

- Вы подозреваете, что кто-то взломал вашу систему три дня назад. Поэтому просматриваете систему, чтобы узнать, изменились ли за последние три дня права доступа или владельцы у каких-либо файлов:

```
$ find /bin /usr/bin /sbin /usr/sbin -ctime -3
```

- Вы хотите найти на своем FTP-сервере (`/var/ftp`) и веб-сервере (`/var/www`) файлы, которые были недоступны более 300 дней, чтобы узнать, нужно ли их удалять:

```
$ find /var/ftp /var/www -atime +300
```

Как видно из приведенных примеров, можно искать изменения содержимого или метаданных в течение определенного времени. Параметры времени (`-atime`, `-ctime` и `-mtime`) позволяют выполнять поиск в зависимости от количества дней, прошедших с момента обращения к каждому файлу, его изменения или изменения метаданных. Параметры `min` (`-amin`, `-cmin` и `-mmin`) делают то же самое, только в минутах.

Числам, которые применяются в качестве аргументов для параметров `min` и `time`, предшествуют дефис (чтобы указать время от настоящего времени до этого количества минут или дней назад) или знак плюс (чтобы указать прошедшее время в минутах или днях). Без дефиса или плюса точное число будет одним и тем же.

Использование ключевых слов not и or при поиске файлов

С помощью параметров `-not` и `-or` можно дополнительно уточнить поиск. Бывает, нужно найти файлы, принадлежащие определенному пользователю, но не назначенные определенной группе. Эти файлы могут быть и большего размера, чем в реальности, и меньшего, чем другие. Или нужны файлы, принадлежащие одному из нескольких пользователей. Параметры `-not` и `-or` могут помочь со всем этим.

- Существует общий каталог `/var/allusers`. Эта командная строка позволяет найти файлы, принадлежащие пользователям `joe` и `chris`:

```
$ find /var/allusers \( -user joe -o -user chris \) -ls
679967    0 -rw-r--r-- 1 chris chris 0 Dec 31 12:57
  /var/allusers/myjoe
679977 1812 -rw-r--r-- 1 joe joe  4379 Dec 31 13:09
  /var/allusers/dict.dat
679972    0 -rw-r--r-- 1 joe sales   0 Dec 31 13:02
  /var/allusers/one
```

- Эта командная строка выполняет поиск файлов, принадлежащих пользователю `joe`, но только тех, которые не назначены группе `joe`:

```
$ find /var/allusers/ -user joe -not -group joe -ls
679972 0 -rw-r--r-- 1 joe sales 0 Dec 31 13:02 /var/allusers/one
```

- При поиске можно назначить несколько требований. Например, файл должен принадлежать пользователю joe и быть более 1 Мбайт:

```
$ find /var/allusers/ -user joe -and -size +1M -ls
679977 1812 -rw-r--r-- 1 joe root 1854379 Dec 31 13:09
  /var/allusers/dict.dat
```

Поиск файлов и выполнение команд

Одна из самых мощных функций команды find — возможность выполнять команды для любых найденных файлов. С параметром -exec используемая команда выполняется для каждого выявленного файла без запроса. С параметром -ok команда останавливается на каждом совпадении и спрашивает, нужно ли выполнить команду для этого файла.

Преимущество параметра -ok заключается в возможности убедиться в том, что пользователь подтверждает любое действие для каждого файла по отдельности. Синтаксис для параметров -exec и -ok один и тот же:

```
$ find [ параметры ] -exec команда {} \;
$ find [ параметры ] -ok команда {} \;
```

Можно запустить команду find с любым из параметров, -exec или -ok, чтобы найти нужные файлы. В этом случае вводится параметр -exec или -ok, за которым следует команда, которую нужно запустить для каждого файла. Фигурные скобки указывают, где в командной строке читать каждый найденный файл. Любой файл может быть включен в командную строку не один раз. Чтобы закончить строку, нужно добавить обратную косую черту и точку с запятой (\;). Вот несколько примеров.

- Эта команда ищет любой файл с именем passwd в каталоге /etc и включает это имя в выходные данные команды echo:

```
$ find /etc -iname passwd -exec echo "I found {}" \;
I found /etc/pam.d/passwd
I found /etc/passwd
```

- Такая команда находит каждый файл в каталоге /usr/share размером более 5 Мбайт. Затем перечисляет размеры всех файлов с помощью команды du. Дальше выходные данные find сортируются по размеру, от самого большого до самого маленького. При вводе параметра -exec все найденные записи обрабатываются без запроса:

```
$ find /usr/share -size +5M -exec du {} \; | sort -nr
116932 /usr/share/icons/HighContrast/icon-theme.cache
69048  /usr/share/icons/gnome/icon-theme.cache
20564  /usr/share/fonts/cjkuni-uming/uming.ttc
```

- Параметр -ok позволяет выбрать, будет ли каждый найденный файл обрабатываться с помощью введенной команды. Например, нужно найти все файлы

пользователя joe в каталоге /var/allusers и его подкаталогах и переместить их в каталог /tmp/joe:

```
# find /var/allusers/ -user joe -ok mv {} /tmp/joe/ \;
< mv ... /var/allusers/dict.dat > ? y
< mv ... /var/allusers/five > ? y
```

Обратите внимание на то, что в предыдущем примере предлагается подтвердить перемещение в каталог /tmp/joe каждого найденного файла. Введите y и нажмите клавишу Enter в каждой строке, чтобы переместить файл, или просто нажмите клавишу Enter, чтобы пропустить его.

Чтобы получить дополнительную информацию о команде find, введите man find.

Поиск по файлам с помощью grep

Если нужно выполнить поиск файлов, содержащих определенный поисковый запрос, используйте команду grep. С ее помощью можно рекурсивно искать один файл или всю структуру каталогов файлов.

В поиске нужно напечатать каждую строку, содержащую термин (стандартный вывод), или просто перечислить имена файлов, содержащих поисковый запрос. По умолчанию grep выполняет поиск текста с учетом регистра, хотя его можно и не учитывать.

Можно использовать grep не только для простого поиска файлов, но и для поиска стандартных выходных данных. Если команда выдает много текста и нужно найти только строки, содержащие определенный текст, применяйте grep для фильтрации данных.

Несколько примеров командных строк grep:

```
$ grep desktop /etc/services
desktop-dna    2763/tcp          # Desktop DNA
desktop-dna    2763/udp          # Desktop DNA

$ grep -i desktop /etc/services
sco-dtmgr      617/tcp           # SCO Desktop Administration Server
sco-dtmgr      617/udp           # SCO Desktop Administration Server
airsync        2175/tcp          # Microsoft Desktop AirSync Protocol
```

В первом примере результат применения grep для слова desktop в файле /etc/services содержал две строки. Новый поиск с помощью параметра -i отключил чувствительность к регистру (как во втором примере), и были найдены 29 строк текста.

Для поиска строк, не содержащих выделенную текстовую строку, используйте параметр -v. В следующем примере отображаются все строки из файла, кроме тех, которые содержат текст tcp (без учета регистра):

```
$ grep -vi tcp /etc/services
```

Для рекурсивного поиска возьмите параметр `-r` и каталог в качестве аргумента. В следующем примере берется параметр `-l`, который просто перечисляет файлы, содержащие текст поиска, не отображая реальные строки. Этот поиск нашел файлы, в которых есть слово `peerdns` (без учета регистра):

```
$ grep -rli peerdns /usr/share/doc/
/usr/share/doc/dnsmasq-2.66/setup.html
/usr/share/doc/initscripts-9.49.17/sysconfig.txt
...
```

В следующем примере выполняется рекурсивный поиск термина `root` в каталоге `/etc/sysconfig`. В результате перечисляется каждая строка во всех файлах под каталогом, содержащим этот текст. Чтобы выделить термин `root`, добавлен параметр `-color`. По умолчанию цвет соответствия поиску — красный:

```
$ grep -ri --color root /etc/sysconfig/
```

Чтобы выполнить поиск выходных данных команды по термину, можно передать выходные данные в команду `grep`. Я знаю, что в приведенном далее примере IP-адреса перечислены в выходных строках команды `ip`, которые включают строку `net`, поэтому использую `grep` для отображения только этих строк:

```
$ ip addr show | grep inet
inet 127.0.0.1/8 scope host lo
inet 192.168.1.231/24 brd 192.168.1.255 scope global wlan0
```

Резюме

Умение обращаться с обычными текстовыми файлами — один из необходимых для работы в Linux навыков. Поскольку множество файлов конфигурации и документов сохранены в текстовом формате, необходимо владеть текстовым редактором и эффективно его использовать. Поиск имен файлов и их содержимого также является критически важным навыком. В этой главе мы изучили, как применять команды `locate` и `find` для поиска файлов и `grep` — для поиска в файлах.

В следующей главе рассматриваются различные способы взаимодействия с процессами. Мы узнаем, как просматривать, какие процессы выполняются, как запускать процессы в фоновом режиме и изменять процессы (отправлять сигналы).

Упражнения

Выполните упражнения, чтобы проверить свои навыки работы с текстовым редактором `vi` (или `vim`), командами поиска файлов (`locate` и `find`) и поиска в файлах (`grep`). Задачи подходят для систем Fedora или Red Hat Enterprise Linux (некоторые сработают и в других системах Linux). Если затрудняетесь с решением заданий,

воспользуйтесь ответами к упражнениям, приведенными в приложении Б (хотя Linux позволяет решать задачи разными способами).

1. Скопируйте файл `/etc/services` в каталог `/tmp`. Откройте файл `/tmp/services` в `vim` и найдите файл со словом `WorldWideWeb`. Измените его на `World Wide Web`.
2. Найдите в файле `/tmp/services` следующий абзац (если его там нет, выберите другой абзац) и переместите его в конец этого файла.

```
# Note that it is presently the policy of IANA to assign a single well-known
# port number for both TCP and UDP; hence, most entries here have two entries
# even if the protocol doesn't support UDP operations.
# Updated from RFC 1700, "Assigned Numbers" (October 1994). Not all ports
# are included, only the more common ones.
```

3. Используя режим редактирования, найдите каждое упоминание термина `tcp` (регистр важен) в своем файле `/tmp/services` и измените его на любой другой.
4. Как обычный пользователь найдите в каталоге `/etc` все файлы с именем `passwd`. Перенаправьте сообщения об ошибках из результатов поиска в `/dev/null`.
5. Создайте в своем домашнем каталоге каталог с именем `TEST`. Создайте в этом каталоге файлы с именами `one`, `two` и `three`, у которых есть полные права на чтение/запись/исполнение для всех (пользователей, групп и других). Введите команду `find`, чтобы найти эти файлы и любые другие с правами на запись, открытые для других пользователей, из вашего домашнего каталога и всех прочих, расположенных ниже.
6. Найдите в каталоге `/usr/share/doc` файлы, которые не менялись в течение 300 дней.
7. Создайте каталог `/tmp/FILES`. Найдите в каталоге `/usr/share` все файлы размером более 5 Мбайт и менее 10 Мбайт и скопируйте их в каталог `/tmp/FILES`.
8. Найдите все файлы в каталоге `/tmp/FILES` и сделайте в нем резервную копию каждого из файлов. Используйте существующие имена файлов и просто добавьте файл `.mybackup` для создания резервных копий всех файлов.
9. Установите пакет `kernel-doc` в Fedora или Red Hat Enterprise Linux. С помощью команды `grep` найдите в файлах, содержащихся в каталоге `/usr/share/doc/kernel-doc*`, термин `e1000` (без учета регистра) и перечислите имена файлов, содержащих его.
10. Вновь найдите там же термин `e1000`, но на этот раз перечислите каждую строку, содержащую его. Термин должен быть выделен цветом.

6 Управление активными процессами

В этой главе

- Отображение процессов.
- Запуск процессов в фоновом и обычном режимах.
- Прерывание и изменение приоритетов процесса.

Linux не только многопользовательская, но и многозадачная операционная система. *Многозадачность* означает, что много программ могут выполняться одновременно. Экземпляр запущенной программы называется *процессом*. Linux позволяет перечислять запущенные процессы, мониторить использование системы и прерывать (убивать) процессы по необходимости.

Из оболочки можно запускать процессы, а затем приостанавливать, останавливать или завершать их. Их можно также выполнять в обычном или фоновом режимах. В этой главе описываются команды `ps`, `top`, `kill`, `jobs` и др., которые позволяют перечислять процессы и управлять ими.

Что такое процесс

Процесс — это запущенный экземпляр команды. Например, в системе может быть одна команда `vi`. Однако если с `vi` работают одновременно 15 пользователей, то она представлена 15 различными запущенными процессами.

Процесс идентифицируется в системе с помощью так называемого *идентификатора процесса (PID)*. Этот PID уникален для текущей системы. Другими словами, ни один прочий процесс не может использовать этот номер в качестве идентификатора процесса, пока первый процесс работает. Однако после завершения первого процесса другой процесс может повторно задействовать это число.

У процесса есть не только идентификатор, но и другие атрибуты. Каждый выполняемый процесс связан с определенной учетной записью пользователя и учетной записью группы. Информация об учетной записи помогает определить, к каким системным ресурсам процесс может получить доступ. Например, у процесса, запущенного от имени суперпользователя, доступ к системным файлам и ресурсам больший, чем у процесса, запущенного от имени обычного пользователя.

Системному администратору Linux необходимо уметь управлять процессами, протекающими в системе. Иногда запущенные процессы могут привести к снижению производительности системы. Управление процессами, которые используют память и процессор, также будет описано в данной главе.

ПРИМЕЧАНИЕ

Команды, отображающие информацию о запущенных процессах, получают бóльшую часть этой информации в виде необработанных данных, хранящихся в файловой системе /proc. Каждый процесс хранит свою информацию в подкаталоге /proc с именем идентификатора этого процесса. Часть необработанных данных можно просмотреть, отобразив содержимое файлов в одном из этих каталогов (используя команды cat или less).

Перечисление процессов

Команда `ps` — одна из первых и наиболее распространенных команд для перечисления процессов, запущенных в данный момент в системе. Linux-версия команды `ps` содержит множество параметров из устаревших систем UNIX и BSD, часть из которых конфликтуют и реализуются нестандартными способами. На справочной странице `ps` представлена информация об этих параметрах.

Команда `top`, которая также перечисляет процессы, больше ориентирована на экран и может служить для изменения статуса процессов. В интерфейсе GNOME можно использовать утилиту System Monitor (Системный монитор), которая отслеживает процессы в системе и представляет информацию с помощью графических средств. Эти команды будут описаны в следующих разделах.

Перечисление процессов с помощью команды ps

Наиболее распространенная команда для проверки запущенных процессов — это `ps`. Она позволяет посмотреть, какие программы запущены, какие ресурсы они используют и кем были запущены. Пример работы команды `ps`:

```
$ ps u
USER  PID %CPU %MEM  VSZ   RSS  TTY   STAT  START  TIME  COMMAND
jake  2147 0.0  0.7 1836  1020  tty1  S+    14:50  0:00  -bash
jake  2310 0.0  0.7 2592   912  tty1  R+    18:22  0:00   ps u
```

В этом примере параметр `u` (как и `-u`) запрашивает отображение имен пользователей и другой информации о времени запуска и использовании памяти и процессора для процессов, связанных с активным пользователем. Показанные процессы

связаны с текущим терминалом (`tty1`). Понятие терминала восходит к былым временам, когда люди работали исключительно с символьными терминалами, которые обычно представляли одного человека на одном экране. Теперь же можно иметь несколько терминалов на одном экране, открыв несколько виртуальных терминалов или окон на рабочем столе.

В примере в этом сеансе оболочки событий немного. Первый процесс показывает, что пользователь по имени `jake` открыл оболочку `bash` после входа в систему. Следующий процесс показывает, что `jake` выполнил команду `ps u`. Терминал `tty1` используется для входа в систему. Столбец `STAT` отображает статус процесса, указывая, запущен ли процесс (`R`) или находится в спящем режиме (`S`).

ПРИМЕЧАНИЕ

В столбце STAT могут отображаться несколько значений. Например, знак плюс (+) указывает, что процесс связан с операциями, протекающими в обычном режиме.

В столбце `USER` отображается имя пользователя, запустившего процесс. У каждого процесса есть номер, называемый идентификатором процесса, или PID. Задействуйте PID, если нужно убить процесс или отправить другой сигнал процессу. Столбцы `CPU` и `MEM` показывают процентное соотношение работы процессора и оперативной памяти, которые использует текущий процесс.

VSZ (виртуальный размер процесса) показывает размер процесса изображения (в килобайтах), а *RSS (размер страниц памяти)* — размер программы в памяти. Значения VSZ и RSS могут различаться, потому что VSZ — это объем памяти, выделенный для процесса, тогда как RSS — это объем, который используется фактически. RSS-память представляет собой физическую незаменяемую память.

Столбец `TART` показывает время начала выполнения процесса, а `TIM` — совокупное используемое системное время. (Большинство команд потребляет мало процессорного времени, для тех, которые не использовали и секунды, время отражается как `0:00`.)

Многие запущенные на компьютере процессы не связаны с терминалом. Обычно в системе Linux множество процессов работает в фоновом режиме. Фоновые системные процессы выполняют различные задачи, к примеру измеряют активность системы или изучают данные из сети. Они часто запускаются при загрузке Linux и работают непрерывно, пока система не выключится. Аналогично вход на рабочий стол Linux запускает многие фоновые процессы, к примеру процессы управления аудио, панели рабочего стола, аутентификацию и другие функции рабочего стола.

Чтобы просмотреть все процессы, запущенные в системе Linux для текущего пользователя, добавьте в `ps ux` конвейер (`|`) и команду `less`:

```
$ ps ux | less
```

Чтобы просмотреть все процессы, запущенные для всех пользователей системы, используйте команду `ps aux` следующим образом:

```
$ ps aux | less
```

Конвейер | (на клавиатуре располагается над символом обратной косой черты) позволяет направлять вывод одной команды на вход следующей. В этом примере выходные данные команды `ps` (список процессов) направляются в команду `less`, которая позволяет просматривать эту информацию на странице. Нажимайте клавишу Пробел, чтобы пролистать страницу, и введите `q`, чтобы закончить список. Можно также применять клавиши со стрелками для перемещения по строкам вывода данных.

Команда `ps` может отображать выбранные столбцы информации и сортировать информацию по одному из этих столбцов. Используя параметр `-o`, можно добавить ключевые слова для указания столбцов, которые нужно перечислить с помощью команды `ps`. Далее в примере отображаются все запущенные процессы (`-e`), а за ними следует параметр `-o` для каждого столбца, содержащего нужную информацию, включая идентификатор процесса (`pid`), имя пользователя (`user`), идентификатор пользователя (`uid`), название группы (`group`), идентификатор группы (`gid`), количество выделенной виртуальной памяти (`vsz`), количество резидентной памяти (`rss`) и запущенную командную строку (`come`).

По умолчанию выходные данные сортируются по номеру процесса (PID):

```
$ ps -eo pid,user,uid,group,gid,vsz,rss,comm | less
  PID  USER       UID GROUP      GID    VSZ   RSS COMMAND
  1    root         0 root         0 187660 13296 systemd
  2    root         0 root         0      0     0 kthreadd
```

Если нужно отсортировать данные иначе, примените параметр `sort=`. Например, чтобы увидеть, какие процессы используют больше всего памяти, я сортирую список по столбцу `vsz` от меньшего к большему. Поскольку мне сначала нужно увидеть самые затратные процессы, я поставил дефис перед параметром сортировки (`sort=-vsz`):

```
$ ps -eo pid,user,group,gid,vsz,rss,comm --sort=-vsz | head
    PID USER    GROUP       GID    VSZ   RSS COMMAND
 2366 chris   chris      1000 3720060 317060 gnome-shell
 1580 gdm     gdm          42 3524304 205796 gnome-shell
 3030 chris   chris      1000 2456968 248340 firefox
 3233 chris   chris      1000 2314388 316252 Web Content
```

Обратитесь к справочной странице `ps`, чтобы узнать больше о других столбцах, по которым можно выводить и сортировать информацию.

Перечисление и изменение процессов с помощью команды top

Команда `top` показывает процессы, запущенные в системе. По умолчанию порядок их отображения зависит от того, сколько процессорного времени они потребляют в данный момент. Однако можно сортировать информацию и по другим столбцам.

После того как будет найден неправильно работающий процесс, с помощью команды `top` можно его завершить (`kill`) или изменить его приоритет (`renice`).

Чтобы завершать или переименовывать процессы, нужно запускать команду от имени суперпользователя. Чтобы просто отображать процессы и, возможно, завершать или изменять свои собственные, можно запускать команды и от имени обычного пользователя. На рис. 6.1 показан пример диалогового окна команды `top`.

```
top - 14:59:56 up  1:02,  1 user,  load average: 0.44, 0.41, 0.31
Tasks: 254 total,   1 running, 253 sleeping,   0 stopped,   0 zombie
%Cpu(s):  3.7 us,  1.2 sy,  0.0 ni, 94.9 id,  0.0 wa,  0.2 hi,  0.2 si,  0.0 st
MiB Mem :   2336.0 total,    163.9 free,   1723.2 used,    448.9 buff/cache
MiB Swap:      0.0 total,      0.0 free,      0.0 used.    412.1 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
 2366 chris     20   0 3754664 360232  82412 S   4.3  15.1   5:04.14 gnome-shell
 3233 chris     20   0 2315412 323812 112896 S   2.3  13.5   1:55.87 Web Content
15222 cockpit+  20   0  607588  13200  10212 S   0.7   0.6   0:06.82 cockpit-ws
16924 chris     20   0  680312  49244  35320 S   0.7   2.1   0:22.68 gnome-system-mo
 1797 root      20   0   49132   2456   2084 S   0.3   0.1   0:00.83 spice-vdagentd
 3030 chris     20   0 2456968 252124 101972 S   0.3  10.5   0:48.93 firefox
15246 root      20   0  887040  12060   7584 S   0.3   0.5   0:04.45 cockpit-bridge
    1 root      20   0  187660  13236   7884 S   0.0   0.6   0:04.81 systemd
    2 root      20   0       0      0      0 S   0.0   0.0   0:00.00 kthreadd
    3 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_gp
    4 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_par_gp
```

Рис. 6.1. Отображение запущенных процессов с помощью команды top

Общая информация о системе отображается в верхней части, а затем указываются сведения о каждом запущенном процессе (сколько поместится на экране). Сверху показано, как долго система работает, сколько пользователей в настоящее время вошли в нее и сколько было запросов за последние одну, пять и десять минут.

Общая информация включает в себя данные о том, сколько процессов (задач) в настоящее время запущено, насколько активно задействован ЦП, сколько используется оперативной памяти и подкачки (swap). Далее приведены списки для каждого процесса, отсортированные по степени задействования ресурсов процессора. Вся эта информация по умолчанию обновляется каждые 5 секунд.

Далее приведен список действий, которые можно выполнить с помощью команды `top`, чтобы отобразить информацию различными способами и изменить запущенные процессы.

- Нажмите клавишу `h`, чтобы просмотреть параметры справки, а затем — любую клавишу, чтобы вернуться к основному дисплею.
- Нажмите клавишу `M`, чтобы отсортировать информацию об использовании памяти, а затем — клавишу `P`, чтобы вернуться к сортировке по процессору.
- Нажмите цифру `1`, чтобы переключать отображение степени использования процессоров, если в системе их больше одного.
- Нажмите клавишу `R` для обратной сортировки выходных данных.
- Нажмите `u` и введите имя пользователя, чтобы увидеть только его процессы.

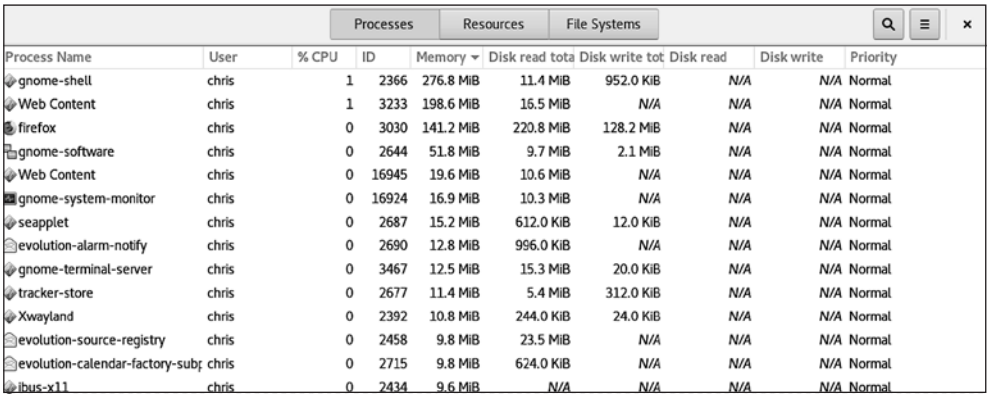
Обычно команда `top` применяется для поиска процессов, которые потребляют слишком много памяти или вычислительной мощности, и взаимодействия с ними. У процесса, потребляющего слишком много ресурсов ЦП, можно сменить приоритет. Или полностью завершить его (убить). Пример использования `top`.

- **Смена приоритета.** Запомните идентификатор процесса, для которого нужно сменить приоритет, и, когда появится сообщение PID to renice, введите номер. При появлении запроса Renice PID to value на изменение значения PID введите число от –20 до 19. (См. подраздел «Настройка приоритета с помощью команд nice и renice» далее в этой главе, чтобы узнать о значениях для `renice`.)
- **Убийство процесса.** Запомните идентификатор процесса и нажмите клавишу k. Введите `15`, чтобы процесс завершил работу, или `9`, чтобы остановить его сразу. (Дополнительную информацию о различных сигналах, которые можно посылать процессам, см. в подразделе «Завершение процессов с помощью команд kill и killall» далее в этой главе.)

Перечисление процессов с помощью программы «Системный монитор»

Если у вас рабочий стол GNOME, то вы можете воспользоваться программой «Системный монитор» (System Monitor), которая отображает все процессы в системе в графическом представлении. Процессы сортируются щелчками кнопкой мыши на столбцах. Можно также щелкнуть правой кнопкой мыши на процессе, чтобы заморозить или завершить его либо изменить его приоритет.

Чтобы запустить программу System Monitor в GNOME, нажмите клавишу Windows, а затем введите `System Monitor` и нажмите клавишу Enter. Затем выберите вкладку Processes (Процессы). На рис. 6.2 показано окно System Monitor (Системный монитор) с процессами текущего пользователя, отсортированными по степени использования памяти.



Process Name	User	% CPU	ID	Memory	Disk read tota	Disk write tot	Disk read	Disk write	Priority
gnome-shell	chris	1	2366	276.8 MiB	11.4 MiB	952.0 KiB	N/A	N/A	Normal
Web Content	chris	1	3233	198.6 MiB	16.5 MiB	N/A	N/A	N/A	Normal
firefox	chris	0	3030	141.2 MiB	220.8 MiB	128.2 MiB	N/A	N/A	Normal
gnome-software	chris	0	2644	51.8 MiB	9.7 MiB	2.1 MiB	N/A	N/A	Normal
Web Content	chris	0	16945	19.6 MiB	10.6 MiB	N/A	N/A	N/A	Normal
gnome-system-monitor	chris	0	16924	16.9 MiB	10.3 MiB	N/A	N/A	N/A	Normal
seapplet	chris	0	2687	15.2 MiB	612.0 KiB	12.0 KiB	N/A	N/A	Normal
evolution-alarm-notify	chris	0	2690	12.8 MiB	996.0 KiB	N/A	N/A	N/A	Normal
gnome-terminal-server	chris	0	3467	12.5 MiB	15.3 MiB	20.0 KiB	N/A	N/A	Normal
tracker-store	chris	0	2677	11.4 MiB	5.4 MiB	312.0 KiB	N/A	N/A	Normal
Xwayland	chris	0	2392	10.8 MiB	244.0 KiB	24.0 KiB	N/A	N/A	Normal
evolution-source-registry	chris	0	2458	9.8 MiB	23.5 MiB	N/A	N/A	N/A	Normal
evolution-calendar-factory-subp	chris	0	2715	9.8 MiB	624.0 KiB	N/A	N/A	N/A	Normal
ibus-x11	chris	0	2434	9.6 MiB	N/A	N/A	N/A	N/A	Normal

Рис. 6.2. Окно программы System Monitor

По умолчанию отображаются только запущенные процессы, связанные с учетной записью пользователя, в алфавитном порядке. Чтобы отсортировать процессы, щелкните кнопкой мыши на любом из заголовков таблицы. Например, щелкните на заголовке %CPU, чтобы увидеть, какие процессы потребляют больше всего ресурсов процессора. Или на заголовке Memory (Память), чтобы увидеть, каким процессам требуется больше всего оперативной памяти.

Процессы можно изменять различными способами, щелкнув правой кнопкой мыши на имени процесса и выбрав действие в появившемся меню (рис. 6.3).

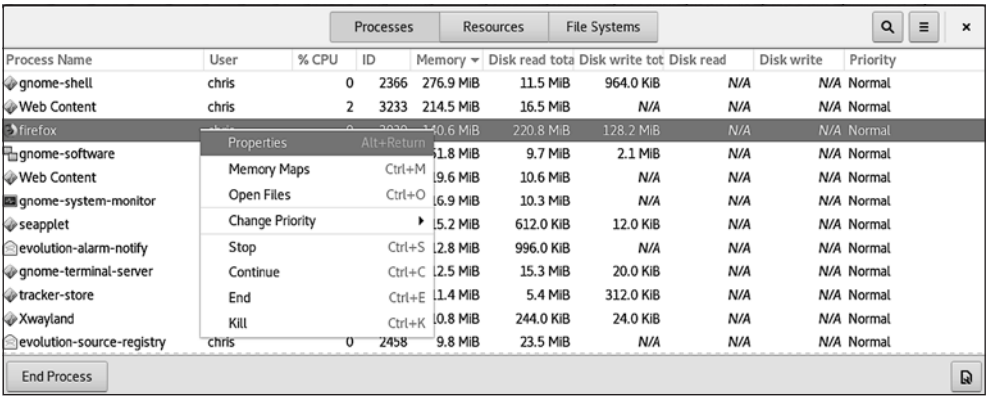


Рис. 6.3. Настройка приоритетов, завершение и остановка процессов в окне программы System Monitor

Рассмотрим действия с процессами.

- Stop (Остановить) — приостанавливает процесс, пока не будет выбрано действие Continue Process (Продолжить), — это то же самое, что нажать сочетание клавиш Ctrl+Z на процессе из оболочки.
- Continue (Продолжить) — продолжает остановленный процесс.
- End (Завершить) — посылает процессу сигнал завершения (15). В большинстве случаев это действие полностью завершает процесс.
- Kill (Убить) — посылает процессу сигнал уничтожения (9). Немедленно завершает процесс в любом случае.
- Change Priority (Изменить приоритет) — список приоритетов от очень низкого до очень высокого. Выберите пункт Custom (Другой), чтобы отобразить ползунковый регулятор приоритета. Обычный приоритет равен 0. Высокий приоритет обозначается отрицательными цифрами от –1 до –20. Низкий приоритет обозначается цифрами от 0 до 19. Только суперпользователь может назначать отрицательные приоритеты, поэтому, чтобы установить высокий приоритет, при появлении запроса необходимо указать пароль.
- Memory Maps (Карты памяти) — позволяет просмотреть карту системной памяти, чтобы увидеть, какие компоненты хранятся в памяти для данного процесса.

- Open Files (Открытые файлы) — позволяет просмотреть, какие файлы в данный момент открыты из-за процесса.
- Properties (Свойства) — позволяет просматривать другие параметры, связанные с процессом (например, контекст безопасности, использование памяти и процент использования процессора).

Можно отображать запущенные процессы других пользователей. Для этого выделите любой процесс (просто щелкните на нем кнопкой мыши). Затем в меню (кнопка с тремя полосами) выберите пункт All Processes (Все процессы). Изменять процессы других пользователей можно только от имени суперпользователя или с помощью его пароля.

Бывает, что графического интерфейса нет. Чтобы изменять процессы без графического интерфейса, можно использовать набор специальных команд и сочетаний клавиш. Некоторые из них мы рассмотрим позже.

Управление обычными и фоновыми процессами

Если вы работаете с Linux по сети или через *неинтеллектуальный терминал* (монитор, который позволяет только вводить текст, без графического интерфейса), командный интерпретатор — единственное, что у вас есть. Легко привыкнуть к графической среде, где есть множество активных программ, между которыми можно быстро перемещаться по мере необходимости. А вот оболочка может показаться в этом смысле ограниченной.

Хотя оболочка `bash` не использует графический интерфейс для одновременного запуска многих программ, она позволяет перемещать активные программы в фоновый режим и наоборот. Таким образом можно одновременно работать с несколькими процессами или только с одним.

Существует несколько способов переведения активной программы в фоновый режим. Один из них — добавить амперсанд (`&`) в конец командной строки при первом запуске команды. Эту команду можно использовать, чтобы выполнять команды, не подключаясь к оболочке.

Чтобы остановить запущенную команду и перевести ее в фоновый режим, нажмите сочетание клавиш Ctrl+Z. После того как команда остановлена, можно либо вернуть ее в обычный режим (команда `fg`), либо запустить в фоновом режиме (команда `bg`). Имейте в виду, что любая команда, работающая в фоновом режиме, может выдать выходные данные во время выполнения команд в обычном режиме. Например, если появляется вывод из команды, работающей в фоновом режиме во время сеанса `vi`, нажмите сочетание клавиш Ctrl+L, чтобы обновить экран и убрать вывод.

СОВЕТ

Чтобы избежать появления выходных данных, заставьте любой процесс, работающий в фоновом режиме, отправлять свои выходные данные сразу в файл или в null (введите 2> /dev/null в конец командной строки).

Запуск фоновых процессов

Если нужно продолжить работу в оболочке, запустите нужный процесс в фоновом режиме. Чтобы перевести программу в фоновый режим во время запуска, введите амперсанд (&) в конце командной строки, например:

```
$ find /usr > /tmp/allusrfiles &
[3] 15971
```

В примере команда находит все файлы в системе Linux (начиная с /usr), выводит имена файлов и помещает их в файл /tmp/allusrfiles. Амперсанд (&) запускает эту командную строку в фоновом режиме. Обратите внимание на то, что при запуске команды отображаются номер задания [3] и идентификатор процесса 15971. Чтобы проверить, какие команды выполняются в фоновом режиме, используйте команду job следующим образом:

```
$ jobs
[1]  Stopped (tty output)  vi /tmp/myfile
[2]  Running        find /usr -print > /tmp/allusrfiles &
[3]  Running        nroff -man /usr/man2/* >/tmp/man2 &
[4]- Running        nroff -man /usr/man3/* >/tmp/man3 &
[5]+ Stopped        nroff -man /usr/man4/* >/tmp/man4
```

Первое задание показывает команду редактирования текста (vi), которая приостановлена (с помощью сочетания клавиш Ctrl+Z) и находится в фоновом режиме. Задание 2 показывает только что запущенную команду find. Задания 3 и 4 показывают команды nroff, работающие в данный момент в фоновом режиме. Задание 5 выполнялось в оболочке (в обычном режиме), пока я не решил, что запущено слишком много процессов, и не нажал Ctrl+Z, чтобы остановить его, пока не будут завершены другие процессы.

Знак плюс (+) рядом с цифрой 5 показывает, что процесс совсем недавно переведен в фоновый режим. Знак минус (-) рядом с цифрой 4 показывает, что процесс был переведен в фоновый режим предпоследним. Поскольку задание 1 требует ввода терминала, оно не может выполняться в фоновом режиме. В результате оно остановлено (Stopped) до тех пор, пока его не переведут в обычный режим.

СОВЕТ

Чтобы увидеть идентификатор процесса для фонового задания, добавьте параметр -l (строчная буква L) в команду задания. Введите ps, чтобы с помощью идентификатора процесса выяснить, какая команда предназначена для конкретного фонового задания.

Команды для обычного и фонового режимов

В том же примере можно вывести на передний план любую из команд, входящую в список заданий. Например, для редактирования файла myfile нужно снова ввести:

```
$ fg %1
```

В результате команда `vi` будет запущена вновь. Весь текст остается таким же, каким был до остановки задания `vi`.

ВНИМАНИЕ!

Прежде чем перевести текстовый редактор или аналогичную программу в фоновый режим, убедитесь, что файл сохранен. Легко забыть о программе, работающей в фоновом режиме, выйти из системы и потерять данные.

Чтобы вернуться к фоновому заданию (отменить его или перевести в обычный режим), используйте знак процента (`%`), после которого стоит номер задания. Для работы с фоновым процессом применяйте следующие команды.

- `%`. Относится к последней команде, помещенной в фоновый режим (обозначается знаком плюс при вводе команды `jobs`). Выводит команду на передний план.
- `%string`. Относится к заданию, в котором команда начинается с определенной строки символов. Строка должна быть однозначной. (Другими словами, ввод `%vi` при наличии двух команд `vi` в фоновом режиме приводит к появлению сообщения об ошибке.)
- `%?string`. Относится к заданию, в котором командная строка содержит строку в любой точке. Строка должна быть однозначной, в ином случае совпадения не будет.
- `%--`. Относится к последнему остановленному процессу.

Если команда остановлена, ее можно снова запустить в фоновом режиме с помощью команды `bg`. Чтобы рассмотреть, как это происходит, вернемся к заданию 5 из списка заданий в предыдущем примере:

```
[5]+ Stopped nroff -man /usr/man4/* >/tmp/man4
```

Введите:

```
$ bg %5
```

После этого задание станет выполняться в фоновом режиме. Запись о заданиях будет выглядеть следующим образом:

```
[5] Running nroff -man /usr/man4/* >/tmp/man4 &
```

Завершение процессов и изменение их приоритетов

Точно так же, как можно изменять поведение процесса с помощью графических инструментов, таких как System Monitor (описан ранее в этой главе), можно использовать средства командной строки, чтобы завершить процесс или изменить его приоритет. Команда `kill` посылает сигнал завершения любому процессу при

условии, что у пользователя есть разрешение на это. Команда может также посылать различные сигналы процессу, чтобы изменить его поведение. Команды `nice` и `renice` применяются для настройки или изменения приоритета.

Завершение процессов с помощью команд kill и killall

Обычно команды `kill` и `killall` применяются для завершения запущенного процесса, однако они могут отправлять запущенному процессу любую допустимую инструкцию. Помимо отправки сигнала завершения, команда может сообщить процессу о необходимости пересмотреть файлы конфигурации, приостановиться (остановиться) или продолжить выполнение после приостановки, и это еще не все варианты.

Сигналы представляются как числами, так и именами. Чаще всего для команд используют сигналы `SIGKILL (9)`, `SIGTERM (15)` и `SIGHUP (1)`. Сигналом по умолчанию является `SIGTERM`, который аккуратно завершает процесс. Чтобы немедленно завершить процесс, задействуйте `SIGKILL`. Сигнал `SIGHUP` в зависимости от программы может сообщить процессу, чтобы он пересмотрел свои файлы конфигурации. `SIGSTOP` приостанавливает процесс, а `SIGCONT` продолжает остановленный процесс.

Разные процессы по-разному реагируют на различные сигналы. Однако процессы не могут блокировать сигналы `SIGKILL` и `SIGSTOP`. В табл. 6.1 приведены примеры некоторых сигналов (введите `man 7 signal`, чтобы узнать о других доступных сигналах).

Таблица 6.1. Сигналы в Linux

Сигнал	Номер	Описание
SIGHUP	1	Обнаружен обрыв связи с управляющим терминалом либо завершение управляющего процесса
SIGINT	2	Прерывание с клавиатуры
SIGQUIT	3	Выход с клавиатуры
SIGABRT	6	Сигнал прерывания, посланный функцией abort(3)
SIGKILL	9	Сигнал немедленного завершения
SIG-TERM	15	Сигнал завершения
SIGCONT	19, 18, 25	Продолжить выполнение, если остановлен
SIGSTOP	17, 19, 23	Приостановить выполнение процесса

Обратите внимание на то, что для `SIGCONT` и `SIGSTOP` существует несколько возможных чисел, поскольку разные номера используются в разных компьютерных сборках. Для большинства сборок x86 и Power берите среднее значение. Первое значение обычно работает в Alpha и SPARC, а последнее — в MIPS.

Отправка сигналов с помощью kill по PID

Используя команды `ps` и `top`, можно найти процессы и отправить им сигналы. Затем можно использовать идентификатор этого процесса (PID) в качестве параметра для команды `kill`.

Например, вы запускаете команду `top` и видите, что процесс `bigcommand` потребляет бóльшую часть вычислительной мощности компьютера:

```
  PID USER    PR  NI  VIRT  RES   SHR S %CPU %MEM   TIME+   COMMAND
10432 chris   20  0   471m  121m  18m S 99.9 3.2 77:01.76  bigcommand
```

В примере процесс `bigcommand` потребляет 99,9 % ресурсов ЦП. Пользователь решает завершить его, чтобы освободить ресурсы компьютера для других. Вот как можно завершить процесс несколькими способами, используя идентификатор запущенного процесса `bigcommand`:

```
$ kill 10432
$ kill -15 10432
$ kill -SIGKILL 10432
```

По умолчанию сигнал, посылаемый командой `kill`, равен `15` (`SIGTERM`), поэтому у первых двух примеров результаты одинаковые. Иногда `SIGTERM` не может завершить процесс, поэтому понадобится сигнал `SIGKILL`, чтобы убить его. Вместо `SIGKILL` можно использовать `-9` и получить тот же результат.

Полезен и сигнал `SIGHUP`. Если, например, на рабочем столе GNOME произошел сбой, отправьте в `gnome-shell` сигнал `SIGHUP`, чтобы перечитать его файлы конфигурации и перезапустить. Если бы идентификатор процесса для `gnome-shell` был `1833`, то отправить ему сигнал `SIGHUP` можно было бы двумя способами:

```
# kill -1 1833
# killall -HUP gnome-shell
```

Отправка сигналов процессам с помощью killall по имени

С помощью команды `killall` можно отправлять сигналы процессам, указывая их имена, а не идентификаторы. Преимущество этого приема в том, что в таком случае для завершения процесса идентификатор не нужен. Потенциальный недостаток заключается в том, что по неосторожности можно завершить больше процессов, чем нужно. (Например, набрав `killall bash`, можно завершить кучу оболочек, чего не стоило бы делать.)

Как и `kill`, команда `killall` по умолчанию использует сигнал `SIGTERM` (`15`), если не указан другой. Кроме того, как и в случае с `kill`, можно отправить любой сигнал в процесс, вызванный с помощью `killall`. Например, чтобы завершить процесс под названием `testme`, запущенный в системе, введите:

```
$ killall -9 testme
```

Команда `killall` особенно полезна, если нужно завершить множество команд с одинаковым именем.

Настройка приоритета с помощью команд nice и renice

Когда ядро Linux решает, какие запущенные процессы получают доступ к процессорам, оно принимает во внимание и указанный для процесса приоритет. Каждый запущенный в системе процесс имеет приоритет в диапазоне от –20 до 19. По умолчанию устанавливается приоритет 0. Приведу сведения о приоритетах.

- Чем ниже значение приоритета, тем процесс приоритетнее для процессора. Чем выше значение приоритета, тем меньше процессор «обращает внимания» на процесс. Таким образом, процесс со значением –20 приоритетнее для процессора, чем процесс со значением 19.
- Обычный пользователь может установить приоритет только от 0 до 19. Отрицательные значения ему недоступны. Таким образом, обычный пользователь не может устанавливать приоритет процессов выше, чем задано по умолчанию.
- Обычный пользователь может только увеличивать значения, но не уменьшать. Так, например, если пользователь установит значение приоритета 10, а затем захочет вернуться к значению 5, действие выполнено не будет. Любая попытка установить отрицательное значение также окажется неудачной.
- Обычный пользователь может устанавливать приоритет только для собственных процессов.
- Суперпользователю разрешено устанавливать приоритеты для любого процесса, и это может быть любое допустимое значение.

Используйте команду `nice`, чтобы установить приоритет процесса. Для изменения приоритета уже запущенного процесса задействуйте команду `renice` вместе с идентификатором процесса, как показано далее:

```
# nice -n +5 updatedb &
```

Команда `updatedb` применяется для принудительного создания базы данных путем сбора имен файлов по всей файловой системе. В примере мне было необходимо, чтобы процесс `updatedb` работал в фоновом режиме (`&`) и не прерывал работу, выполняемую другими процессами в системе. Я запустил команду `top`, чтобы убедиться, что приоритет установлен правильно:

```
PID USER      PR  NI  VIRT  RES  SHR  S  %CPU %MEM  TIME+  COMMAND
20284 root    25   5 98.7m  932  644  D   2.7 0.0  0:00.96 updatedb
```

Обратите внимание на то, что в столбце `Ni` указан приоритет `5`. Поскольку команда запущена от имени суперпользователя, позже значение приоритета можно уменьшить с помощью команды `renice`. (Помните, что обычный пользователь не может уменьшить значение приоритета и установить отрицательное число.) Так, можно изменить приоритет команды `updatedb`, задав значение `-5`:

```
# renice -n -5 20284
```

Снова запустив команду `top`, можно заметить, что команда `updatedb` теперь находится в верхней части списка процессов, потребляющих ЦП, благодаря установленному приоритету.

Ограничения для процессов с помощью cgroups

Используйте функцию установки приоритета, чтобы дать одному процессу больший или меньший доступ к процессорному времени. Однако значение приоритета процесса не применяется к его дочерним процессам или любым другим связанным процессам. Другими словами, приоритет не ограничивает общее количество ресурсов, которые конкретный пользователь или приложение может потреблять из системы Linux.

По мере развития облачных вычислений многие системы Linux будут работать скорее как гипервизоры, чем как компьютеры общего пользования. Их память, вычислительная мощность и доступ к хранилищу станут широко задействоваться множеством обычных пользователей. В таком случае необходимо иметь больше возможностей для управления объемом системных ресурсов, к которым имеют доступ конкретный пользователь, приложение, конвейер или виртуальная машина в системе Linux.

Вот тут-то и появляются контрольные группы механизма *cgroups*.

Данный механизм используется для идентификации процесса как задачи, принадлежащей определенной контрольной группе. Задачи могут быть выстроены в соответствии с иерархией, вверху которой может находиться, например, задача-демон, устанавливающая ограничения по умолчанию для всех процессов сервера демонов, а ниже — подзадачи, которые задают определенные ограничения для демона веб-сервера (`httpd`) и демона службы FTP (`vsftpd`).

Поскольку задача запускает процесс, то процессы, начатые начальным процессом (их называют дочерними), имеют те же ограничения, что и родительский процесс. Ограничиваться может доступ к определенным процессорам или определенным наборам оперативной памяти. Можно также ограничить доступ к 30 % общей вычислительной мощности.

Типы ресурсов, которые могут быть ограничены механизмом `cgroups`, таковы.

- **Устройства** (`blkio`) — устанавливает лимиты на доступ к запоминающим устройствам (жестким дискам, USB-накопителям и т. д.).
- **Процессор** (`cpu`) — обеспечивает доступ процессов в рамках контрольной группы к CPU.
- **Учет процессов** (`cpuacct`) — генерирует отчеты об использовании ресурсов процессора. С помощью данной информации можно рассчитать, сколько клиенты должны будут заплатить за объем задействуемой ими вычислительной мощности.
- **Распределение процессоров** (`cpuset`) — при наличии нескольких процессорных ядер распределяет между ними задачи в рамках контрольной группы.

- **Доступ к устройствам** (`devices`) — разрешает или блокирует доступ (`mknod`) к выбранным устройствам.
- **Приостановка процесса** (`freezer`) — приостанавливает и возобновляет выполнение задач в рамках контрольной группы.
- **Использование памяти** (`memory`) — управляет выделением памяти для групп процессов и создает отчеты об используемых ресурсах.
- **Пропускная способность сети** (`net_cls`) — ограничивает доступ к сети для выбранных задач в рамках контрольной группы. Помечает сетевые пакеты специальной меткой, позволяя идентифицировать порождаемые определенной задачей в рамках контрольной группы.
- **Сетевой трафик** (`net_prio`) — используется для динамической установки приоритетов по трафику, позволяет администратору изменять приоритеты.
- **Пространство имен** (`ns`) — разделяет контрольные группы на пространства имен, причем для одной контрольной группы видны только связанные пространства имен. Пространства имен могут включать отдельные таблицы процессов, таблицы монтирования и сетевые интерфейсы.

На базовом уровне создание контрольных групп и управление ими не является основной задачей системных администраторов Linux. Однако важно уметь редактировать файлы конфигурации и создавать контрольные группы (`/etc/cgconfig.conf`) или устанавливать ограничения для определенных пользователей либо групп (`/etc/cgrules.conf`). Для создания групп `cgroups` можно использовать также команду `cgcreate`, в результате эти группы будут добавлены в иерархию `/sys/fs/cgroup`. Настройка контрольных групп — задача непростая, и, если ее выполнить неправильно, система может перестать загружаться.

Я рассказываю о группах потому, что необходимо понимать основные функции Linux, которые применяются для ограничения используемых ресурсов и мониторинга. В будущем вы, вероятно, столкнетесь с этими функциями у контроллеров облачной инфраструктуры. Вы сможете устанавливать правила, например «разрешать виртуальным машинам отдела маркетинга потреблять до 40 % доступной памяти» или «привязать базу данных к определенному процессору и набору памяти».

Знание того, как Linux может ограничивать использование ресурсов набором процессов, назначенных задаче, в конечном счете поможет лучше управлять вычислительными ресурсами. Чтобы узнать больше о контрольных группах, изучите следующие документы.

- Руководство Red Hat Enterprise Linux Resource Management and Linux Containers, access.redhat.com/documentation/en-us/red_hat_Enterprise_linux/7/html-single/resource_management_ guide/index.
- Информация о контрольных группах на сайте Kernel. Перейдите к файлу `/usr/share/doc/kernel-doc-*/Documentation/cgroups` после установки пакета `kernel- doc`.

Резюме

Даже в не особо активной системе Linux обычно функционируют десятки и даже сотни процессов в фоновом режиме. С помощью инструментов, описанных в этой главе, можно просматривать процессы, запущенные в системе, и управлять ими. Управление процессами включает в себя их просмотр различными способами, запуск в обычном или фоновом режиме, а также изменение или завершение. Ограничения на использование ресурсов устанавливаются с помощью функции контрольных групп. В следующей главе вы узнаете, как объединить команды и функции программирования в файлы, которые можно запускать как скрипты оболочки.

Упражнения

Выполните упражнения, чтобы применить знания о просмотре запущенных процессов и работе с ними. Задачи подходят для систем Fedora или Red Hat Enterprise Linux (некоторые сработают и в других системах Linux). Если затрудняетесь с решением заданий, воспользуйтесь ответами к упражнениям, приведенными в приложении Б (хотя Linux позволяет решать задачи разными способами).

1. Перечислите все процессы, запущенные в системе с полным набором столбцов информации. Передайте эти выходные данные в команду `less`, чтобы пролистать список процессов.
2. Перечислите все процессы, запущенные в системе, и отсортируйте их по имени пользователя.
3. Перечислите все процессы, запущенные в системе, и отобразите следующие столбцы информации: идентификатор процесса, имя пользователя, имя группы, размер виртуальной памяти, размер резидентной памяти и команда.
4. Выполните команду `top`, чтобы просмотреть процессы, запущенные в системе. Отсортируйте информацию по использованию процессора и потреблению памяти от большего к меньшему и наоборот.
5. Запустите процесс `gedit` с рабочего стола. Убедитесь, что запускаете его от имени пользователя, под которым вошли в систему. Воспользуйтесь окном System Monitor (Системный монитор), чтобы завершить этот процесс.
6. Снова запустите процесс `gedit`. На этот раз с помощью команды `kill` отправьте сигнал процессу `gedit`, чтобы приостановить его. Попробуйте ввести в окно `gedit` какой-нибудь текст и убедитесь, что он не добавляется.
7. Используйте команду `killall`, чтобы сообщить команде `gedit` о приостановке (предыдущее упражнение) и продолжить работу. Убедитесь, что набираемый в окне `gedit` текст теперь появляется на экране.

8. Установите команду `xeyes` (в дистрибутиве Fedora она находится в пакете `xorg-x11-apps`). Выполните команду около 20 раз в фоновом режиме, чтобы на экране появилось 20 окон `xeyes`. Перемещайте мышь и смотрите, как глаза следят за указателем. Когда закончите веселиться, завершите все процессы `xeyes` командой `killall`.
9. От имени обычного пользователя запустите команду `gedit` так, чтобы она имела приоритет 5.
10. С помощью команды `renice` измените приоритет команды `gedit` на 7. Используйте любую команду, чтобы убедиться, что текущее значение приоритета для команды `gedit` теперь равно 7.

7 Простые скрипты оболочки

В этой главе

- Работа со скриптами оболочки.
- Числа в скриптах оболочки.
- Циклы и команда `case` в скриптах оболочки.
- Создание простых скриптов.

В системе Linux не все процессы, команды которых просто вводятся в интерпретатор, будут выполняться. И конечно, можно работать эффективнее, сгруппировав наборы часто используемых команд. Скрипты оболочки как раз для этого и созданы.

Скрипт оболочки — это группа команд, функций, переменных или почти все, что можно задействовать из оболочки. Эти элементы помещаются в обычный текстовый файл, который затем можно запустить как команду. В Linux скрипты оболочки традиционно используют для загрузки системы во время запуска, чтобы выполнять команды и запускать необходимые службы. Пользователь может создавать собственные скрипты оболочки для автоматизации регулярных задач.

Десятилетиями создание скриптов оболочки было основным навыком, необходимым для объединения наборов задач в системах UNIX и Linux. По мере того как требования к настройке систем Linux переставали быть требованиями к одной системе и становились сложными конфигурациям автоматизированных кластеров, появились более структурированные методы. Эти методы включают в себя Ansible playbooks и файлы Kubernetes YAML, описанные далее в главах, связанных с облаком. Тем не менее написание скриптов оболочки — по-прежнему наилучший способ автоматизации повторяющихся задач в системах Linux.

В этой главе дается краткий обзор внутренней работы скриптов оболочки и способов их применения. Приводится информация о том, как использовать простые

скрипты (например, `cron` или `at`) для планирования с целью упрощения административных задач или просто запускать по требованию по мере необходимости.

Скрипты оболочки

Если у вас есть регулярно повторяющаяся задача с большим количеством текста для командной строки, думали ли вы когда-нибудь: «Ох, хотел бы я просто набрать одну команду, чтобы сделать все сразу»? Скорее всего, скрипт оболочки — именно то, что вам нужно.

Скрипты оболочки эквивалентны пакетным файлам в Windows и могут содержать длинные списки команд, сложное управление потоками, арифметические вычисления, пользовательские переменные и функции, а также сложную проверку условий. Скрипты оболочки способны обрабатывать все: от простых однострочных команд до чего-то столь же сложного, как запуск системы Linux. Хотя в Linux доступны десятки различных оболочек, используемая по умолчанию оболочка для большинства систем Linux называется `bash` (Bourne Again SHell).

Выполнение и отладка скриптов

Одно из основных преимуществ скриптов оболочки — можно открыть их в любом текстовом редакторе и посмотреть содержимое. Серьезный недостаток — большие или сложные скрипты оболочки часто выполняются медленнее, чем скомпилированные программы. Скрипт оболочки выполняется двумя основными способами.

- Имя файла используется в оболочке в качестве аргумента (как в `bash myscript`). В этом случае файл не обязательно должен быть исполняемым — он просто содержит список команд оболочки. Оболочка, указанная в командной строке, применяется для интерпретации команд в файле скрипта. Этим способом часто реализуют быстрые простые задачи.
- В первой строке скрипта оболочки может стоять имя интерпретатора, перед которым указаны символы `#!` (как в `#!/bin/bash`) и флаг исполнения файла с набором скриптов (реализуется с помощью `chmod +x filename`). Затем можно запустить свой скрипт так же, как и любую другую программу в пути, просто введя его имя в командной строке.

При выполнении скриптов любым способом параметры программы можно указать в командной строке. Все, что следует за именем скрипта, называется *аргументом командной строки*.

Как и при написании любого ПО, нет никакой замены четкому, продуманному дизайну и большому количеству комментариев. Знак фунта (`#`) указывается перед комментарием и может занимать целую строку или стоять в строке после кода

скрипта. Лучше всего реализовывать более сложные скрипты оболочки постепенно, убедившись, что на каждом этапе все работает логично. Вот несколько кратких советов, чтобы убедиться, что все работает так, как нужно.

- В некоторых случаях можно поместить оператор `echo` в начале строки в теле цикла и взять его в кавычки. Таким образом вместо того, чтобы выполнять код, можно увидеть, что будет выполнено без внесения каких-либо изменений.
- Для этой же цели можно разместить фиктивные операторы `echo` по всему коду. Если строки с ними появляются на экране, значит, логика соблюдается.
- Используйте команду `set -x` в начале скрипта, чтобы отобразить каждую выполняемую команду или запустить скрипты с помощью:

```
$ bash -x myscript
```

- Полезные скрипты имеют тенденцию увеличиваться со временем, поэтому чрезвычайно важно сохранить код читаемым. Делайте все возможное, чтобы логика вашего кода оставалась стройной и понятной.

Переменные оболочки

Часто в скрипте оболочки требуется повторно использовать определенные элементы информации. В процессе обработки скрипта имя или число, несущие эту информацию, могут изменяться. Чтобы информация, используемая скриптом оболочки, сохранялась такой, которую легко применить повторно, можно задать *переменные*. Имена переменных в скриптах оболочки чувствительны к регистру и могут быть определены следующим образом:

```
ИМЯ=значение
```

Первая часть переменной — это ее имя, а вторая — набор значений для него. Убедитесь, что `ИМЯ` и *`значение`* соединены знаком равенства без пробелов. Значениями переменных могут быть константы, такие как текст, цифры и символы подчеркивания. Это помогает инициализировать значения и экономить текст для записи длинных констант. В следующих примерах показаны переменные, заданные в виде строки символов (`CITY`) и числового значения (`PI`):

```
CITY="Springfield"
PI=3.14159265
```

Переменные могут содержать выходные данные команды или последовательности команд. Для этого нужно поставить перед командой знак доллара и открыть скобку, а после нее набрать закрывающую скобку. Например, `MYDATE=$(date)` присваивает выходные данные команды `date` переменной `MYDATE`. То же самое получится, если добавить в команду обратные тики (`` ` ``). В этом случае команда `date` выполняется при введении переменной, а не при каждом ее чтении.

Использование в оболочке специальных символов

Имейте в виду, что такие символы, как знак доллара (`$`), обратный тик (`` ` ``), звездочка (`*`), восклицательный знак (`!`) и др., имеют особые значения в оболочке. В каких-то случаях они нужны, а в каких-то — нет. Например, если набрать `echo $HOME`, оболочка отобразит на экране имя домашнего каталога, хранящегося в переменной `$HOME` (например, `/home/chris`), так как символ `$` указывает, что за ним следует имя переменной.

Если в действительности нужно показать текст `$HOME`, то знак `$` следует экранировать. Для этого введите `echo'$HOME'` или `echo\$HOME`. Поэтому, если нужно, чтобы оболочка интерпретировала один символ буквально, введите перед ним обратную косую черту (`\`). Чтобы интерпретировать все значение буквально, окружите эти символы одинарными кавычками (`'`).

С двойными кавычками все немного сложнее. Окружите набор текста двойными кавычками, если нужно, чтобы все символы, кроме некоторых, обрабатывались буквально. Например, если текст окружен двойными кавычками, знаки доллара (`$`), обратные тики (`` ` ``) и восклицательные знаки (`!`) интерпретируются по-особенному, а другие символы (например, `*`) — как обычно. Введите следующие три строки, чтобы получить различные выходные данные (показаны справа):

```
echo ' $HOME * ` date` '  $HOME * `date`
echo "$HOME * `date`"     /home/chris * Tue Jan 21 16:56:52 EDT 2020
echo $HOME * `date`       /home/chris file1 file2 Tue Jan 21 16:56:52 EDT 2020
```

Переменные — отличный способ получить информацию, которая может меняться от компьютера к компьютеру или изо дня в день. В следующем примере выходные данные команды `uname -n` присваиваются переменной `MACHINE`. Затем с помощью скобок я установил параметр `NUM_FILES` на нужное количество файлов в текущем каталоге с помощью конвейеров (`|`) и вывел данные команды `ls` в команду подсчета слов (`wc -l`):

```
MACHINE=`uname -n`
NUM_FILES=$(/bin/ls | wc -l)
```

Переменные могут содержать также значения других переменных, что полезно, когда нужно сохранить изменяющееся значение и позже задействовать его в скрипте. В примере далее `BALANCE` устанавливается в значение переменной `CurBalance`:

```
BALANCE="$CurBalance"
```

ПРИМЕЧАНИЕ

При назначении переменных используйте только имя переменной (например, BALANCE). Ссылаясь на значение переменной, введите перед ней знак доллара (например, $CurBalance). В таком случае будет задействовано именно значение переменной, а не ее имя.

Позиционные параметры оболочки

Существуют специальные переменные, которые назначает оболочка. Набор часто используемых символов называется *позиционными параметрами аргументов командной строки*. Они указываются в виде значений `$0`, `$1`, `$2`, `$3`... `$n`. `$0` — особенный, ему присваивается имя, используемое для вызова скрипта, остальным присваиваются значения параметров, передаваемых в командной строке, в том порядке, в котором они появились.

Предположим, что скрипт оболочки с именем `myscript` содержит:

```
#!/bin/bash
# Script to echo out command-line arguments
echo "The first argument is $1, the second is $2."
echo "The command itself is called $0."
echo "There are $# parameters on your command line"
echo "Here are all the arguments: $@"
```

Допустим, что скрипт является исполняемым и находится в каталоге `$PATH`, тогда вот что произойдет, если запустить эту команду с аргументами `foo` и `bar`:

```
$ chmod 755 /home/chris/bin/myscript
$ myscript foo bar
The first argument is foo, the second is bar.
The command itself is called /home/chris/bin/myscript.
There are 2 parameters on your command line
Here are all the arguments: foo bar
```

Как видно в примере, позиционный параметр `$0` — это полный или относительный путь к `myscript`, `$1` — к `foo`, а `$2` — к `bar`.

Другая переменная, `$#`, сообщает, сколько параметров было задано в скрипте. В этом примере переменная `$#` будет равна 2. Переменная `$@` содержит все аргументы, введенные в командной строке. Другой специальной переменной оболочки является `$?`, которая получает статус выхода последней выполненной команды. Как правило, нулевое значение означает, что команда завершилась успешно, а все, что не равно нулю, указывает на ошибку. Полный список позиционных параметров оболочки см. в руководстве `bash`.

Чтение параметров

С помощью команды `read` можно запросить у пользователя информацию и сохранить ее для последующего использования в скрипте. Пример скрипта с командой `read`:

```
#!/bin/bash
read -p "Type in an adjective, noun and verb (past tense): " adj1 noun1 verb1
echo "He sighed and $verb1 to the elixir. Then he ate the $adj1 $noun1."
```

В примере после того, как скрипт запросил прилагательное (adjective), существительное (noun) и глагол (verb), пользователь должен ввести слова, которые

затем назначаются переменным adj1, noun1 и verb1. Эти три переменные в дальнейшем добавляются в шутливое предложение скрипта. Если бы скрипт назывался sillyscript, вот как он мог бы работать:

```
$ chmod 755 /home/chris/bin/sillyscript
$ sillyscript
Type in an adjective, noun and verb (past tense): hairy football danced
He sighed and danced to the elixir. Then he ate the hairy football.
```

Расширение параметров в bash

Как уже упоминалось, если нужно получить значение переменной, перед ней следует указать знак $ (например, $CITY). На самом деле это просто сокращенное обозначение ${CITY}, фигурные скобки используются, когда значение параметра не должно отделяться от другого текста пробелом.

У оболочки bash есть специальные правила, которые позволяют расширить значение переменной различными способами. Конечно, сразу все правила изучить нельзя, однако далее представлено несколько общих конструкций, которые встречаются в скриптах bash.

- ${*var*:-*значение*} — если переменная не задана или пуста, расширяет до значения *значение*.
- ${*var*#*шаблон*} — удаляет кратчайшее совпадение шаблона *шаблон* от начала параметра *var*.
- ${*var*##*шаблон*} — удаляет наибольшее совпадение шаблона *шаблон* с началом параметра *var*.
- ${*var*%*шаблон*} — удаляет кратчайшее совпадение шаблона *шаблон* до конца параметра *var*.
- ${*var*%%*шаблон*} — удаляет наибольшее совпадение шаблона *шаблон* с концом параметра *var*.

Введите следующие команды из командной оболочки, чтобы проверить, как работает расширение параметров:

```
$ THIS="Example"
$ THIS=${THIS:-"Not Set"}
$ THAT=${THAT:-"Not Set"}
$ echo $THIS
Example
$ echo $THAT
Not Set
```

В приведенных примерах переменная THIS изначально задается словом Example. Далее в двух строках переменные THIS и THAT устанавливаются в свои текущие значения или в Not Set, если они в данный момент не установлены. Обратите внимание: поскольку я просто установил THIS в строку Example, при повторе

значение THIS появляется в качестве Example. А так как THAT не установлен, то параметр отображается в виде Not Set.

ПРИМЕЧАНИЕ

Далее в этом разделе показано, как переменные и команды могут отображаться в скрипты оболочки. Чтобы опробовать любую из них, просто введите их в оболочку, как показано в предыдущем примере.

В следующем примере MYFILENAME имеет значение /home/digby/myfile.txt. Далее переменная FILE имеет значение myfile.txt, а DIR — значение /home/digby. В переменной NAME имя файла сокращается просто до myfile, а в переменной EXTENSION расширение файла устанавливается как txt. (Чтобы попрактиковаться, введите их в командной строке, как в предыдущем примере, и повторите значение каждой переменной, чтобы увидеть, в каком значении она установлена.) Введите код, приведенный слева, справа описано его действие:

- MYFILENAME=/home/digby/myfile.txt — устанавливает значение MYFILENAME;
- FILE=${MYFILENAME##*/} — FILE присваивается myfile.txt;
- DIR=${MYFILENAME%/*} — DIR присваивается /home/digby;
- NAME=${FILE%.*} — NAME присваивается myfile;
- EXTENSION=${FILE##*.} — EXTENSION присваивается txt.

Арифметические операции в скриптах оболочки

Bash использует *нетипизированные переменные*, то есть обрабатывает переменные как строки текста, что при желании можно быстро изменить.

bash задействует *нетипизированные переменные*, что означает: не обязательно указывать, является переменная текстом или числом. Обычно оболочка обрабатывает переменные как строки текста, поэтому без помощи команды declare переменные — просто набор букв для bash. Но если выполнять арифметические действия, оболочка стремится преобразовать строки в целые числа. Благодаря этому в bash возможна довольно сложная арифметика.

Арифметика с целыми числами выполняется с помощью встроенной команды let, или внешних команд expr, или bc. После установки значения переменной BIGNUM равным 1024 все три следующие команды будут хранить значение 64 в переменной RESULT. Команда bc — это приложение калькулятора, имеющееся в большинстве дистрибутивов Linux. Последняя команда получает случайное число в диапазоне от 0 до 10 и возвращает результаты:

```
BIGNUM=1024
let RESULT=$BIGNUM/16
RESULT=`expr $BIGNUM / 16`
RESULT=`echo "$BIGNUM / 16" | bc`
let foo=$RANDOM; echo $foo
```

Другой способ увеличения значения переменной — использовать `$(())` с добавлением `++` для увеличения значения `I`. Введите:

```
$ I=0e
$ echo "The value of I after increment is $((++I))"
The value of I after increment is 1

$ echo "The value of I before and after increment is $((I++)) and $I"
The value of I before and after increment is 1 and 2
```

Повторите любую из этих команд, чтобы продолжить увеличивать значение переменной `$I`.

ПРИМЕЧАНИЕ

Большинство элементов в скриптах оболочки имеют относительно свободную форму (где пробелы или отступы неважны), тогда как для команд let и expr интервалы особенно важны. Команда let требует, чтобы не было пробелов между каждым аргументом операции (операндом) и математическим оператором, в то время как синтаксис команды expr требует указания пробелов между каждым операндом и его оператором. В отличие от них команда bc не зависит от пробелов, но сама по себе сложнее, потому что выполняет вычисления с плавающей запятой.

Чтобы просмотреть полный список арифметических операций, которые можно выполнить с помощью команды `let`, введите `help let` в командной строке `bash`.

Программные конструкции в скриптах оболочки

Одна из особенностей, которая делает скрипты оболочки максимально полезными, заключается в том, что реализуются циклические и условные конструкции, аналогичные тем, которые встречаются в более сложных скриптах и языках программирования. Вы можете использовать несколько различных типов циклических конструкций в зависимости от своих потребностей.

Конструкция if...then

Наиболее часто применяемой конструкцией программирования является условное выражение с оператором `if`. Оно используется, если нужно, чтобы действия выполнялись только при определенных условиях. Существует несколько разновидностей выражений с `if` для разных нужд.

Первый вариант, `if...then`, проверяет, присвоено ли параметру `VARIABLE` значение `1`. Если все верно, то команда `echo` отобразит результат. Затем оператор `if` указывает, что конструкция `if` завершена и обработка может быть продолжена:

```
VARIABLE=1
if [ $VARIABLE -eq 1 ] ; then
echo "The variable is 1"
fi
```

Вместо `-eq` можно поставить знак равенства (`=`), как показано в следующем примере. Знак `=` лучше всего подходит для сравнения значений строк, в то время как с помощью `eq` удобнее сравнивать числа. С помощью оператора `else` можно повторять слова, если условие конструкции `if` не выполняется (`$STRING = "Friday"`). Имейте в виду, что строки нужно помещать в двойные кавычки:

```
STRING="Friday"
if [ $STRING = "Friday" ] ; then
echo "WhooHoo.  Friday."
else
echo "Will Friday ever get here?"
fi
```

Вы также можете изменить вывод, используя восклицательный знак (`!`). В следующем примере, если `STRING` — это не `Monday`, то строка `At least it's not Monday` повторяется:

```
STRING="FRIDAY"
if [ "$STRING" != "Monday" ] ; then
   echo "At least it's not Monday"
fi
```

В следующем примере конструкция `elif` (что означает else if) используется для проверки существования дополнительного условия (например, является `filename` файлом или каталогом):

```
filename="$HOME"
if [ -f "$filename" ] ; then
   echo "$filename is a regular file"
elif [ -d "$filename" ] ; then
   echo "$filename is a directory"
else
   echo "I have no idea what $filename is"
fi
```

Как видно из предыдущих примеров, тестируемое условие помещается в квадратные скобки `[]`. Вычисление условия выдает или 0 (выражение истинно), или 1 (выражение ложно). Обратите внимание на то, что оператор `echo` отбивается отступами. Отступ необязателен и делается только для того, чтобы скрипт был более удобочитаемым.

В табл. 7.1 перечислены условия, которые можно проверить и удобно использовать. (Можно набрать в командной строке `help test`, чтобы вывести ту же информацию.)

Таблица 7.1. Операторы проверки условий

Оператор	Что проверяет
-a file	Файл существует? (-e проверяет то же)
-b file	Является ли файл частью съемного устройства?

Оператор	Что проверяет
-c file	Является ли символ специальным (например, символьным устройством)? Используется для идентификации последовательных соединений и терминальных устройств
-d file	Это каталог?
-e file	Файл существует? (-a проверяет то же)
-f file	Существует ли этот файл и является ли он обычным файлом (например, не каталогом, сокетом, каналом, ссылкой или файлом устройства)?
-g file	Установлен ли в файле бит set group id (SGID)?
-h file	Является ли этот файл символической ссылкой? (-L проверяет то же)
-k file	Есть ли в файле набор sticky bit?
-L file	Является ли этот файл символической ссылкой?
-n string	Длина строки превышает 0 байт?
-O file	Это ваш файл?
-p file	Является ли файл именованным конвейером?
-r file	Можете ли вы просматривать этот файл?
-s file	Существует ли этот файл и больше ли он 0 байт?
-S file	Существует ли этот файл и является ли он сокетом?
-t fd	Подключен ли файловый дескриптор к терминалу?
-u file	Установлен ли в файле бит set user id (SUID)?
-w file	Можете ли вы переписывать этот файл?
-x file	Можете ли вы выполнять этот файл?
-z строка	Равна ли длина строки 0 (ноль)?
expr1 -a expr2	Верны ли и первое, и второе выражения?
expr1 -o expr2	Верно ли одно из этих двух выражений?
file1 -nt file2	Первый файл новее, чем второй (используется метка времени модификации)?
file1 -ot file2	Первый файл старше, чем второй (используется метка времени модификации)?
file1 -ef file2	Связаны ли эти два файла ссылкой (жесткой или символической)?
var1 = var2	Равна ли первая переменная второй?
var1 -eq var2	Равна ли первая переменная второй?
var1 -ge var2	Первая переменная больше второй переменной или равна ей?
var1 -gt var2	Больше ли первая переменная, чем вторая?
var -le var2	Первая переменная меньше второй переменной или равна ей?
var1 -lt var2	Первая переменная меньше второй?
var1 != var2	Не равна ли первая переменная второй переменной?
var1 -ne var2	Не равна ли первая переменная второй переменной?

Существует также сокращенный метод проверки, полезный для простых *однокомандных действий*. В примере далее два конвейера (||) указывают, что если проверяемого каталога не существует (-d dirname), то нужно создать его (mkdir$$dirname):

```
# [ test ] || action
# Выполнить однокомандное действие, если тест неверен.
dirname="/tmp/testdir"
[ -d "$dirname" ] || mkdir "$dirname"
```

Вместо конвейеров можно использовать два амперсанда и проверить выражение. В следующем примере проверяется, содержит ли команда по крайней мере три аргумента командной строки:

```
# [ test ] && {action}
# Выполнить однокомандное действие, если тест верен.
[ $# -ge 3 ] && echo "There are at least 3 command line arguments."
```

Можно комбинировать операторы && и ||, чтобы ускорить проверку с помощью if...then...else. В следующем примере проверяется, существует ли уже каталог $dirname. Если да, появится сообщение о том, что каталог существует. Если нет, конструкция создаст его:

```
# dirname=mydirectory
[ -e $dirname ] && echo $dirname already exists || mkdir $dirname
```

Команда case

Часто используется и конструкция с командой case. Подобно оператору switch в языках программирования, он может заменить несколько операторов if. Далее приводится общая форма выражения case:

```
case "VAR" in
   Result1)
      { body };;
   Result2)
      { body };;
   *)
      { body } ;;
esac
```

Помимо прочего, можно использовать команду case, чтобы разобраться с резервными копиями. Следующее выражение проверяет первые три буквы текущего дня (case 'date+%a' in). Затем в зависимости от дня недели устанавливаются определенный каталог резервных копий (BACKUP) и носитель (TAPE):

```
# Значение VAR не обязательно должно быть переменной,
# оно также может быть выводом команды.
# Выполняйте действия в зависимости от дня недели
case `date +%a` in
   "Mon")
```

```
        BACKUP=/home/myproject/data0
        TAPE=/dev/rft0
# Обратите внимание: используется два знака точки с запятой
# в конце каждого варианта.
        ;;
# Обратите внимание: используется знак | для обозначения варианта ИЛИ.
   "Tue" | "Thu")
        BACKUP=/home/myproject/data1
        TAPE=/dev/rft1
        ;;
   "Wed" | "Fri")
        BACKUP=/home/myproject/data2
        TAPE=/dev/rft2
        ;;
# Не делайте резервные копии в выходные дни.
   *)

BACKUP="none"
        TAPE=/dev/null
        ;;
esac
```

Звездочка (`*`) используется в качестве ключевого слова, аналогичного слову `default` в языке программирования C. В этом примере, если ни одна из прочих записей не сопоставляется на пути вниз по циклу, это делает звездочка, и значение `BACKUP` становится `none`. Обратите внимание на `esac` (`case` наоборот) в завершении проверки выражения `case`.

Цикл for...do

Циклы применяются для реализации повторяющихся действий до тех пор, пока не будет выполнено условие выражения или не обработаны все данные. Чаще всего используется цикл `for...do`. Он перебирает список значений, выполняя тело цикла для каждого элемента в списке. Синтаксис цикла выглядит так:

```
for VAR in LIST
do
   { body }
done
```

Цикл `for` присваивает значения, указанные в *LIST* для *VAR*, по одному за раз. Затем для каждого значения выполняется *body*, стоящее в фигурных скобках между `do` и `done`. `VAR` может быть именем переменной, а `LIST` — состоять практически из любого списка значений или всего, что генерирует список:

```
for NUMBER in 0 1 2 3 4 5 6 7 8 9
do
   echo The number is $NUMBER
done
```

```
for FILE in `/bin/ls`
do
   echo $FILE
done
```

Вариант с более чистым кодом:

```
for NAME in John Paul Ringo George ; do
   echo $NAME is my favorite Beatle
done
```

Каждый элемент в `LIST` отделен от следующего пробелом. Будьте с этим осторожны, потому что некоторые команды, такие как `ls -l`, выводят в строке несколько полей, каждое из которых отделено пробелом. Строка `done` завершает выражение `for`.

Заядлый программист на языке C может использовать его синтаксис для работы с циклами:

```
LIMIT=10
# Используются двойные скобки без знака $ в значении LIMIT,
# несмотря на то что это переменная!
for ((a=1; a <= LIMIT ; a++)) ; do
   echo    "$a"
done
```

Циклы while...do и until...do

Еще два популярных цикла — это `while...do` и `until...do`. Вот их структуры:

```
while condition    until condition
do                 do
   { body }           { body }
done               done
```

Оператор `while` выполняется, пока условие выражения истинно. Оператор `until` выполняется до тех пор, пока условие не станет истинным, другими словами, пока оно ложно.

Пример цикла `while` для вывода числа `0123456789`:

```
N=0
while [ $N -lt 10 ] ; do
   echo -n $N
   let N=$N+1
done
```

Вывод числа `0123456789` с циклом `until`:

```
N=0
until [ $N -eq 10 ] ; do
   echo -n $N
   let N=$N+1
done
```

Полезные программы для работы с текстом

В оболочке bash есть множество полезных встроенных команд, однако для работы с ними обычно требуется помощь. Одни из самых популярных программ — это grep, cut, tr, awk и sed. Как и все лучшие инструменты UNIX, большинство этих программ предназначены для работы со стандартными вводом и выводом и позволяют использовать конвейеры и скрипты.

Синтаксический анализатор общих регулярных выражений

Название *«синтаксический анализатор общих регулярных выражений»* (general regular expression print (grep)) звучит устрашающе, но по сути это просто способ найти шаблоны в файлах или тексте. Инструмент полезный. Полностью овладеть умением работать с регулярными выражениями довольно сложно, но после того, как это начнет получаться, можно выполнить множество задач с помощью самых простых форм.

Например, можно отобразить список всех учетных записей обычных пользователей, используя grep для поиска всех строк, содержащих текст /home в файле /etc/passwd, следующим образом:

```
$ grep /home /etc/passwd
```

Или найти все переменные окружения, начинающиеся с HO, с помощью следующей команды:

```
$ env | grep ^HO
```

ПРИМЕЧАНИЕ

Символ ^ на верхней линии шрифта — это именно символ каретки ^, а не то, что обычно отображается для обратного пробела, ^H.

Введите ^, H и O (прописная буква), чтобы увидеть, какие элементы начинаются с прописных символов HO.

Чтобы увидеть список параметров для команды grep, введите man grep.

Удаление части текста (команда cut)

Команда cut извлекает поля из строки текста или файлов. Это полезно при разделении файлов конфигурации на легко читаемые и понятные фрагменты. Можно добавить разделитель полей или разбить строку на байты.

В следующем примере перечислены все домашние каталоги пользователей в системе. Командная строка grep передает список обычных пользователей из файла /etc/passwd и отображает шестое поле (-f6), разделенное двоеточием (-d':').

Дефис в конце дает команде cut инструкцию читать данные из стандартного ввода (из конвейера):

```
$ grep /home /etc/passwd | cut  -d':' -f6 -
/home/chris
/home/joe
```

Перевод и удаление символов (команда tr)

Команда tr — это символьный переводчик, который используется для замены одного символа или набора символов другим или для удаления символа из строки текста.

В следующем примере все прописные буквы переводятся в строчные и в результате выводятся слова в верхнем и нижнем регистрах:

```
$ FOO="Mixed UPpEr aNd LoWeR cAsE"
$ echo $FOO | tr [A-Z] [a-z]
смешанные нижний и верхний регистры
```

В примере далее команда tr применяется для переименования файлов в списке так, чтобы любые отступы или пробелы в имени файла переводились в подчеркивания, как указано параметром [:blank:]. Попробуйте запустить следующий код в тестовом каталоге:

```
for file in * ; do
   f=`echo $file | tr [:blank:] [_]`
   [ "$file" = "$f" ] || mv -i -- "$file" "$f"
done
```

Потоковый текстовый редактор (sed)

Команда sed — это простой редактор скриптов. Она может выполнять только простые изменения, например удалять строки по шаблону, заменять один шаблон символов другим и т. д. Чтобы лучше понять скрипты sed, рассмотрим несколько примеров общего применения.

Можно использовать команду sed для того, чтобы сделать то же самое, что и в примере с grep, — выполнить поиск слова home в файле /etc/passwd. В этом случае команда sed анализирует весь файл /etc/passwd, ищет слово home и печатает любую строку с ним:

```
$ sed -n '/home/p' /etc/passwd
chris:x:1000:1000:Chris Negus:/home/chris:/bin/bash
joe:x:1001:1001:Joe Smith:/home/joe:/bin/bash
```

В следующем примере команда sed выполняет поиск файла somefile.txt и заменяет Mac на Linux в каждой строке. Обратите внимание: буква g в конце команды замены необходима для того, чтобы каждое вхождение Mac заменялось на Linux,

иначе изменится только первое вхождение `Mac` в каждой строке. Затем выходные данные отправляются в файл `fixed_file.txt`. Выходные данные из `sed` поступают в `stdout`, поэтому команда перенаправляет их в файл для сохранности:

```
$ sed 's/Mac/Linux/g' somefile.txt > fixed_file.txt
```

Можно получить тот же результат, используя конвейер:

```
$ cat somefile.txt | sed 's/Mac/Linux/g' > fixed_file.txt
```

При поиске шаблона и замене его пустым шаблоном исходный удаляется. В примере выполняется поиск содержимого файла `somefile.txt` и лишние пробелы в конце каждой строки (`s/ *$`) заменяются на «ничего» (`//`). Выходные данные отправляются в файл `fixed_file.txt`:

```
$ cat somefile.txt | sed 's/ *$//' > fixed_file.txt
```


Простые скрипты оболочки

Иногда самые простые скрипты оказываются самыми полезными. Если приходится вводить одну и ту же последовательность команд повторно, имеет смысл сохранить эти команды (один раз!) в файле. В следующих разделах мы рассмотрим несколько простых, но полезных скриптов оболочки.

Телефонный список

Эта идея передавалась из поколения в поколение еще со времен хакеров UNIX. Скрипт действительно довольно простой, но в нем используются некоторые из концепций, описанных ранее:

```
#!/bin/bash
# (@)/ph
# A very simple telephone list
# Type "ph new name number" to add to the list, or
# just type "ph name" to get a phone number

PHONELIST=~/.phonelist.txt

# If no command line parameters ($#), there
# is a problem, so ask what they're talking about.
if [ $# -lt 1 ] ; then
  echo "Whose phone number did you want? "
    exit 1
fi

# Did you want to add a new phone number?
if [ $1 = "new" ] ; then
  shift
```

```
  echo $* >> $PHONELIST
  echo $* added to database
  exit 0
fi

# Nope. But does the file have anything in it yet?
# This might be our first time using it, after all.
if [ ! -s $PHONELIST ] ; then
  echo "No names in the phone list yet! "
  exit 1
else
  grep -i -q "$*" $PHONELIST     # Quietly search the file
  if [ $? -ne 0 ] ; then         # Did we find anything?
    echo "Sorry, that name was not found in the phone list"
    exit 1
  else
    grep -i "$*" $PHONELIST
  fi
fi
exit 0
```

Итак, если вы создали файл `ph` со списком телефонов в своем текущем каталоге, введите приведенные ниже данные, чтобы проверить его работу:

```
$ chmod 755 ph
$ ./ph new "Mary Jones" 608-555-1212
Mary Jones 608-555-1212 added to daTabase
$ ./ph Mary
Mary Jones 608-555-1212
```

Благодаря команде `chmod` скрипт `ph` становится исполняемым. Команда `./ph` запускает команду `ph` из текущего каталога с параметром `new`. Таким образом имя `Mary Jones` и номер `608-555-1212` добавляются в базу данных (`$HOME/.phonelist.txt`). Далее команда `ph` ищет в базе данных имя `Mary` и отображает записанный для него телефон. Если скрипт работает, добавьте его в каталог в своем пути (например, `$HOME/bin`).

Скрипт резервного копирования

Поскольку ничто не вечно и ошибки случаются у всех, резервные копии — это необходимость в ходе работы с компьютерными данными. Следующий простой скрипт создает резервные копии всех данных в домашних каталогах всех пользователей систем Fedora или RHEL:

```
#!/bin/bash
# (@)/my_backup
# A very simple backup script
#

# Change the TAPE device to match your system.
# Check /var/log/messages to determine your tape device.
```

```
TAPE=/dev/rft0

# Rewind the tape device $TAPE
mt $TAPE rew
# Get a list of home directories
HOMES=`grep /home /etc/passwd | cut –f6 –d':'`
# Back up the data in those directories
tar cvf $TAPE $HOMES
# Rewind and eject the tape.
Mt $TAPE rewoffl
```


Резюме

Скрипты оболочки дают возможность автоматизировать многие наиболее распространенные задачи системного администрирования. В этой главе мы рассмотрели общие команды и функции, которые можно применять в скриптах с оболочкой `bash`. Мы также рассмотрели конкретные примеры скриптов для создания резервных копий и других процессов.

В следующей главе мы перейдем от изучения пользовательских функций к изучению тем системного администрирования. Глава 8 описывает, как стать суперпользователем, а также как использовать административные команды, отслеживать файлы журналов и работать с файлами конфигурации.

Упражнения

Выполните данные упражнения, чтобы попробовать написать простые скрипты оболочки. Задачи подходят для систем Fedora или Red Hat Enterprise Linux (некоторые сработают и в других системах Linux). Если затрудняетесь с решением заданий, воспользуйтесь ответами к упражнениям, приведенными в приложении Б (хотя Linux позволяет решать задачи разными способами).

1. Создайте скрипт `myownscript` в каталоге `$HOME/bin`. Скрипт должен будет вывести информацию, которая выглядит следующим образом:

   ```
   Сегодня Sat Jan 4 15:45:04 EST 2020.
   Вы находитесь в каталоге /home/joe и ваш хост-адрес – это abc.example.com.
   ```

   Конечно, у вас будут свои значения даты/времени, текущего рабочего каталога и имени хоста. Включите сюда комментарии о работе скрипта и укажите, что скрипт должен работать с оболочкой `/bin/bash`.

2. Создайте скрипт, который считывает из командной строки три позиционных параметра, присваивает их переменным с именами `ONE`, `TWO` и `THREE` и выводит эту информацию в следующем формате:

   ```
   Существует X параметров, которые включают в себя параметр Y.
   Первый – это A, второй – это B, третий – это C.
   ```

Замените `X` количеством параметров, а `Y` — всеми введенными параметрами. Затем замените `A` содержимым переменной `ONE`, `B` — содержимым переменной `TWO` и `C` — содержимым переменной `THREE`.

3. Создайте скрипт, который запрашивает у пользователей названия улицы и города, в котором они выросли. Назначьте город и улицу в качестве переменных `mytown` и `mystreet` соответственно и выведите их в предложении, как показано далее (конечно, вместо `$mystreet` и `$mytown` появятся данные, которые вводит пользователь):

```
Улица, где я вырос, называется $mystreet, а город называется $ mytown
```

4. Создайте скрипт с названием `myos`, который спросит пользователя: «Какая у вас любимая операционная система?» Выведите что-нибудь обидное для тех, кто вписывает вариант Windows или macOS. Для тех, кто укажет Linux, выведите ответ: «Отличный выбор!» Для других вариантов: «А разве <*введенный вариант*> — это операционная система?»
5. Создайте скрипт, который проходит по словам *moose*, *cow*, *goose* и *sow* с помощью цикла `for`. Подставьте каждое из этих слов в конец строки «У меня есть...».

Часть III

Администрирование системы в Linux

В этой части

- Глава 8. Системное администрирование.
- Глава 9. Установка Linux.
- Глава 10. Управление программами.
- Глава 11. Управление учетными записями.
- Глава 12. Управление дисками и файлами.

8 Системное администрирование

В этой главе

- Инструменты администрирования с графическим интерфейсом.
- Вход в систему в качестве суперпользователя.
- Административные команды, файлы конфигураций и логов.
- Работа с устройствами и файловыми системами.

Операционная система Linux, как и другие UNIX-системы, предназначалась для одновременной работы нескольких пользователей. *Многопользовательская функция* позволяет иметь учетные записи разных пользователей в одной системе Linux, защищая данные каждого. *Многозадачность* позволяет одновременно запускать на компьютере множество программ, причем каждый пользователь может запускать более одной программы. Сложные сетевые протоколы и приложения позволяют системе Linux распространять свои возможности на пользователей сети и компьютеры по всему миру. Лицо, которому поручено управлять всеми ресурсами системы Linux, считается *системным администратором*.

Даже если вы единственный, кто использует систему Linux, системное администрирование все еще является важной частью настройки компьютера. Для выполнения большинства административных задач вам необходимо войти в систему как *пользователь root* (также называемый *суперпользователем*) или временно получить права root (обычно с помощью команды `sudo`). Пользователи без прав суперпользователя не могут изменять систему Linux, а в некоторых случаях даже видеть некоторые сведения о ее конфигурации. В частности, от общего просмотра защищены функции безопасности, например сохраненные пароли.

Поскольку системное администрирование Linux — это огромная тема, в данной главе основное внимание уделяется его общим принципам. Мы рассмотрим основные инструменты, необходимые для администрирования системы Linux для

персонального рабочего стола или небольшого сервера. Вы не только усвоите основы, но и научитесь работать с файловыми системами и контролировать состояние и производительность системы Linux.

Системное администрирование

Роль системного администратора отделена от роли других пользователей, и это придает работе определенные особенности. В системе со множеством пользователей ограничения в сфере управления позволяют сохранять ее безопасность. Наличие администратора также не позволяет другим пользователям нанести вред системе, что может произойти, даже если они просто пишут текст в документе или просматривают страницы в Интернете.

Системный администратор системы Linux обычно входит в систему под простой учетной записью пользователя, а затем при необходимости запрашивает права администратора. Приведу варианты того, как это можно сделать.

- **Команда su.** Команда `su` используется в основном, чтобы открыть оболочки от имени суперпользователя. После открытия оболочки администратор может запустить множество команд, выйти, а затем вернуться в оболочку под именем обычного пользователя.
- **Команда sudo.** С помощью команды `sudo` обычный пользователь получает привилегии суперпользователя, но только при использовании команды `sudo` для запуска другой команды. После выполнения этой команды с помощью `sudo` пользователь немедленно возвращается в оболочку как обычный пользователь. Дистрибутивы Ubuntu и Fedora по умолчанию назначают привилегии `sudo` первой установленной учетной записи пользователя. Этого не происходит в дистрибутиве RHEL, но во время его установки можно присвоить привилегию `sudo` первому пользователю.
- **Администрирование серверов с помощью инструмента Cockpit.** RHEL, Fedora и другие дистрибутивы Linux взяли на себя обязательство применять Cockpit в качестве основного браузерного средства администрирования системы. Инструмент позволяет отслеживать и изменять настройки в основных действиях системы, в хранилище, сети, учетных записях, службах и других функциях.
- **Инструменты с графическим интерфейсом.** До того как интерфейс Cockpit стал популярным, RHEL, Fedora и другие дистрибутивы Linux задействовали индивидуальные инструменты администрирования с графическим интерфейсом, которые запускались командами, начинающимися с `system-config-*`. Большинство этих инструментов администрирования в последних версиях RHEL и Fedora не используются. Однако я их упомянул, так как они доступны в более старых версиях Linux.

Задачи, которые может выполнить только суперпользователь, как правило, затрагивают систему в целом или влияют на ее безопасность или работоспособность.

Далее приведен список общих функций, которыми должен управлять системный администратор.

- **Файловые системы.** При первой установке Linux структура каталогов настраивается таким образом, чтобы система стала пригодной для использования. Однако, если позже пользователи захотят добавить дополнительное хранилище или изменить расположение файловой системы за пределами своего домашнего каталога, для этого им понадобятся административные права. Кроме того, суперпользователь имеет разрешение на доступ к файлам, которые принадлежат пользователям. Суперпользователь может копировать, перемещать или изменять файлы любого другого пользователя — привилегия, необходимая для создания резервных копий файловой системы.
- **Установка программного обеспечения.** Поскольку вредоносное программное обеспечение может нанести ущерб системе или сделать ее небезопасной, для установки программ необходимы права суперпользователя. К тому же важно сделать так, чтобы программное обеспечение было доступно всем пользователям системы. Обычные пользователи могут устанавливать лишь некоторые программы в собственных каталогах и выводить информацию об установленных программах.
- **Учетные записи пользователя.** Только суперпользователь может добавлять и удалять учетные записи пользователей или групп.
- **Сетевой интерфейс.** Раньше суперпользователь должен был настраивать сетевые интерфейсы, запускать и останавливать их. Сейчас многие настольные компьютеры Linux позволяют обычным пользователям запускать и останавливать сетевые интерфейсы со своего рабочего стола с помощью программы Network Manager. Особенно подходит она для беспроводных сетевых интерфейсов, которые меняются в зависимости от местоположения ноутбука или устройства с Linux.
- **Серверы.** Настройка, запуск и остановка веб-серверов, файловых серверов, серверов доменных имен, почтовых серверов и десятков других серверов требует привилегий суперпользователя. Содержимое, например веб-страницы, могут добавить на серверы обычные пользователи, если настройки системы позволяют. Службы часто запускаются под специальные административные учетные записи пользователей, такие как `apache` (для службы `httpd`) и `rpc` (для службы `rpcbind`). Таким образом, если кто-то взломает службу, он не сможет получить привилегии администратора для других служб или системных ресурсов.
- **Безопасность.** Настройка функций безопасности, таких как брандмауэры и списки доступа для пользователей, обычно выполняется при наличии прав суперпользователя. Кроме того, суперпользователь должен следить за использованием служб и за тем, чтобы ресурсы сервера не исчерпывались и ими не злоупотребляли.

Самый простой способ начать администрировать систему — применить соответствующие инструменты с графическим интерфейсом.

Инструменты администрирования с графическим интерфейсом

В первых системах Linux бóльшая часть администрирования осуществлялась из командной строки. По мере роста популярности Linux большинство распространенных административных задач обрели графические интерфейсы и интерфейсы командной строки.

В следующих разделах описаны некоторые интерфейсы типа «укажи и щелкни» (point-and-click).

Администрирование с помощью Cockpit

Cockpit — это лучший браузерный инструмент системного администрирования Linux, который я когда-либо видел. Он сочетает в себе целый ряд административных действий и подключается к разнообразному набору функций API Linux с помощью пакета `cockpit-bridge`. Системному администратору Linux необходимо знать, как применять Cockpit для своих нужд.

Начать работу с Cockpit просто: нужно включить сокет Cockpit и подключить службу Cockpit к браузеру. Благодаря дизайну плагинов Cockpit постоянно создаются новые инструменты, которые можно добавить в интерфейс Cockpit своей системы.

Если вы работаете с последними версиями систем RHEL или Fedora, выполните описанные далее действия, чтобы включить и начать использовать Cockpit.

ПРИМЕЧАНИЕ

Для запуска этой процедуры не требуется никаких настроек. Однако можно подключить сертификат OpenSSL вместо применяемого по умолчанию. Таким образом не нужно будет подтверждать непроверенный самоподписанный сертификат при открытии интерфейса из браузера.

1. Если программа Cockpit еще не установлена, введите:

```
# dnf install cockpit
```

2. Залогиньтесь как суперпользователь и включите сокет Cockpit:

```
# systemctl enable --now cockpit.socket
Created symlink /etc/systemd/system/sockets.target.wants/cockpit.socket
    → /usr/lib/systemd/system/cockpit.socket.
```

3. Откройте браузер и перейдите на порт 9090 сервера, на котором установлен Cockpit. Используйте имя узла или IP-адрес. Порт 9090 по умолчанию настроен для https, при желании его можно перенастроить на http. Вот примеры адресов, которые нужно ввести в адресную строку браузера:

```
https://host1.example.com:9090/
https://192.168.122.114:9090/
```

4. Предположим, что вы не заменили самоподписанный сертификат для Cockpit, в таком случае браузер предупреждает о том, что соединение небезопасно. В зависимости от браузера можно выбрать вариант Advanced (Дополнительно), принять риск и продолжить, чтобы позволить браузеру задействовать службу Cockpit.
5. Введите логин и пароль своей обычной учетной записи. Используйте права суперпользователя или пользователя с правами sudo, если хотите изменить конфигурацию своей системы. Обычный пользователь может видеть большинство настроек, но не изменять их. На рис. 8.1 показано окно браузера.

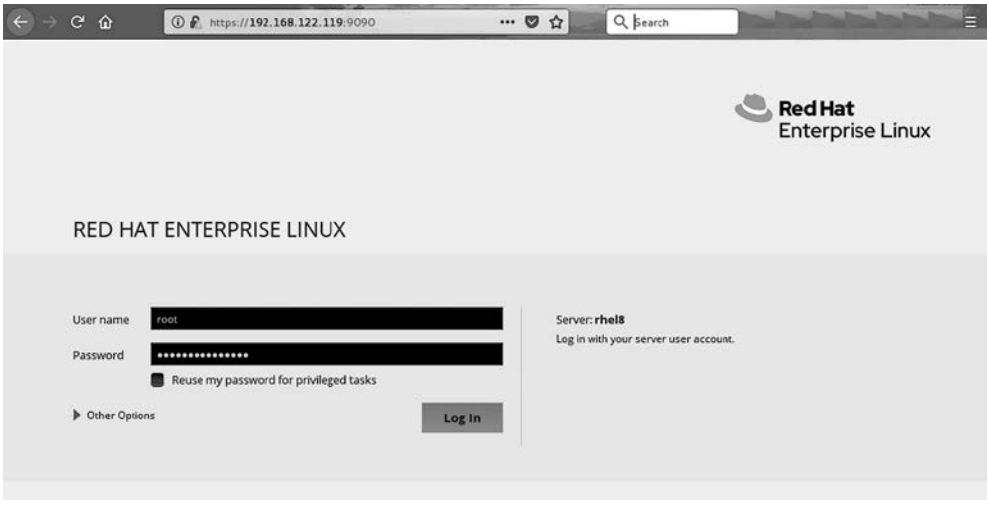


Рис. 8.1. Вход в Cockpit

6. Приступайте к работе с Cockpit. На панели управления Cockpit содержится большой набор функций по умолчанию (другие можно добавить позже) для систем RHEL и Fedora. На рис. 8.2 показан пример раздела System (Система) на панели Cockpit.

Сразу же после входа в Cockpit будет показана системная активность, связанная с использованием процессора, памяти и подкачки, дискового ввода-вывода и сетевого трафика. Элементы управления на левой навигационной панели позволяют

начать работу с журналами, хранилищем, сетевыми учетными записями пользователей и групп, службами и многими другими функциями системы.

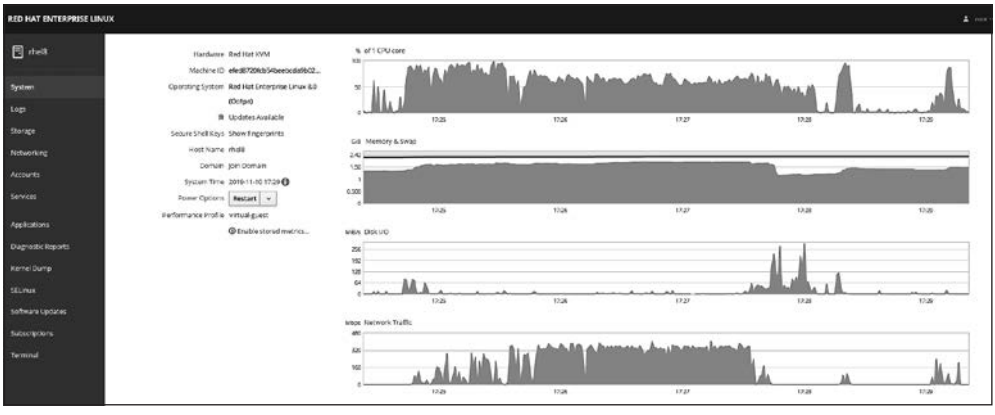


Рис. 8.2. Просмотр активности системы и других тем с панели управления Cockpit

По ходу книги в соответствующих разделах будут встречаться описания применения различных функций Cockpit. Чтобы глубже погрузиться в изучение программы, я рекомендую зайти на сайт Cockpit: cockpit-project.org.

Утилиты system-config-*

До появления Cockpit в системах Fedora и RHEL инструменты с графическим интерфейсом были доступны из подменю Administration (Администрирование) в меню System (Система) в GNOME 2, с экрана Activities (Обзор) в GNOME 3 или из командной строки. В старых версиях систем Fedora и RHEL можно управлять этими инструментами из командной строки, запустив набор команд, начинающихся со строки `system-config*` (например, `system-config-network`).

Инструменты `system-config*` требуют прав суперпользователя. Если вы вошли в систему как обычный пользователь, необходимо ввести пароль root до того, как откроется окно графического интерфейса пользователя (GUI), или в некоторых случаях до выполнения специального действия.

Далее описаны многие инструменты с графическим интерфейсом, применявшиеся в более ранних системах Fedora или RHEL (некоторые могут быть доступны только в Fedora и не установлены по умолчанию). Команда для включения функции показана в скобках (часто она совпадает с именем пакета). В Fedora использовались следующие инструменты с графическим интерфейсом.

- **Служба Domain Name System (system-config-bind).** Позволяет создавать и настраивать зоны, если компьютер работает в качестве DNS-сервера.

- **Служба HTTP (system-config-httpd).** Настраивает компьютер под веб-сервер Apache.
- **Служба NFS (system-config-nfs).** Настраивает каталоги системы для совместного использования с другими компьютерами в сети.
- **Служба Root Password (system-config-rootpassword).** Позволяет сменить пароль суперпользователя.
- **Служба Samba NFS (system-config-samba).** Настраивает общий доступ к файлам Windows (SMB). Для настройки других функций Samba можно задействовать окно SWAT.

До появления RHEL 8 в системах Fedora и RHEL были доступны следующие инструменты с графическим интерфейсом.

- **Службы (Services) (system-config-services).** Отображение и изменение запущенных в системе Fedora служб на разных уровнях выполнения в окне Service Configuration (Настройка служб).
- **Аутентификация (Authentication) (system-config-authentication).** Изменение способа аутентификации пользователей в системе. Как правило, выбираются теневые пароли и пароли MD5. Однако, если ваша сеть поддерживает аутентификацию LDAP, Kerberos, SMB, NIS или Hesiod, можете выбрать любой из этих типов аутентификации.
- **Дата и время (Date & Time) (system-config-date).** Установка даты и времени вручную или синхронизация системного времени с сервером NTP.
- **Брандмауэр (Firewall) (system-config-firewall).** Настройка брандмауэра, которая разрешает или запрещает службам доступ к компьютерам из сети.
- **Язык (Language) (system-config-language).** Установка для системы языка по умолчанию.
- **Печать (Printing) (system-config-printer).** Настройка локальных и удаленных принтеров.
- **Управление SELinux (SELinux Management) (system-config-selinux).** Установка режимов принудительного использования SELinux и политики по умолчанию.
- **Пользователи и группы (Users & Groups) (system-config-users).** Добавление, отображение и изменение учетных записей пользователей и групп системы Fedora.

Другие административные утилиты собраны в меню Applications (Приложения) на верхней панели. Выберите пункт System Tools (Системные утилиты) в GNOME 2 или перейдите в меню Activities (Обзор) в GNOME 3, чтобы выбрать один из приведенных далее инструментов (если они установлены).

- **Пакет Configuration Editor (gconf-editor).** Позволяет редактировать базу данных настроек GNOME.

- **Программа Disk Usage Analyzer (gnome-utils).** Отображает подробную информацию о жестких дисках и съемных устройствах хранения.
- **Программа Disk Utility (gnome-disks).** Управляет разделами диска и добавляет файловые системы (пакет `gnome-disk-utility`).
- **Инструмент Kickstart (system-config-kickstart).** Создает файл конфигурации `kickstart`, который можно использовать для установки нескольких систем Linux, не взаимодействуя с пользователем.

В предыдущих изданиях книги приводились описания данных инструментов. Теперь они заменены функциями программы Cockpit.

Другие браузерные инструменты администрирования

Чтобы упростить управление многими корпоративными проектами с открытым исходным кодом, стали применять браузерные инструменты с графическим интерфейсом. В большинстве случаев управлять такими проектами можно также с помощью инструментов командной строки.

Например, в дистрибутиве Red Hat Enterprise Linux существуют браузерные интерфейсы для управления следующими проектами.

- **Red Hat OpenShift.** Проект *OpenShift*, основанный на проекте Kubernetes, задействует браузерный интерфейс для развертывания и настройки кластеров уровней управления и рабочих узлов, а также развертывания контейнеров в так называемых общих контейнерах *pod*. Больше информации — на сайтах Red Hat OpenShift (openshift.com) и OKD (okd.io).
- **Red Hat Enterprise Linux OpenStack Platform (RHELOSP).** Проект OpenStack является платформой как услугой (Platform as a Service, PaaS) и позволяет управлять собственным частным облаком через браузер. Включает в себя панель мониторинга OpenStack из проекта OpenStack Horizon (horizondocs.openstack.org/horizon/latest). Интерфейс позволяет запускать виртуальные машины и управлять ими, а также всеми ресурсами вокруг них: хранилищем, сетью, аутентификацией, распределением ресурсов обработки и т. д. В главе 27 описана работа с панелью мониторинга OpenStack.
- **Red Hat Virtualization (RHV).** С помощью RHEV менеджер RHV применяет браузерный интерфейс для управления виртуальными машинами, включая работу с хранилищем и доступ пользователей к ресурсам. Многие браузерные инструменты администрирования с графическим интерфейсом доступны в проектах с открытым исходным кодом. Новичку в Linux проще начать работу с этих интерфейсов. Однако имейте в виду: чтобы устранить неполадки, нужно использовать инструменты командной строки, потому что функционал инструментов с графическим интерфейсом в этой области ограничен.

Учетная запись суперпользователя

В каждой системе Linux есть как минимум одна учетная запись администратора (суперпользователя) и, вероятно, одна и более учетных записей обычных пользователей (с именем, выбранным вручную или назначенным дистрибутивом Linux). В большинстве случаев вы входите в систему как обычный пользователь и становитесь суперпользователем для выполнения конкретной административной задачи.

Суперпользователь полностью контролирует систему Linux. Он может открывать любые файлы и запускать любые программы. Он также устанавливает пакеты с программным обеспечением и добавляет учетные записи для тех, кто использует систему.

СОВЕТ

Суперпользователь в Linux по функционалу похож на администратора в Windows.

При первой установке большинства систем Linux (хотя и не всех) для суперпользователя устанавливается пароль. Запомните его и не сообщайте никому: он нужен для входа в систему от имени суперпользователя или для получения прав суперпользователя, если вы находитесь в системе с другой учетной записью.

Чтобы ознакомиться с учетной записью суперпользователя, просто войдите в систему под его именем. Советую делать это с помощью виртуальной консоли. Нажмите сочетание клавиш Ctrl+Alt+F3. Когда появится приглашение для входа, введите `root`, нажмите клавишу Enter и наберите пароль. Система запустится под именем суперпользователя. По окончании введите `exit`, затем нажмите сочетание клавиш Ctrl+Alt+F1, чтобы вернуться в режим обычного пользователя.

Домашний каталог для пользователя с правами root — это каталог `/root`. Домашний каталог и информация, связанная с учетной записью суперпользователя, находятся в файле `/etc/passwd`. Вот как выглядит корневая запись в файле `/etc/passwd`:

```
root:x:0:0:root:/root:/bin/bash
```

Это значит, что для пользователя с именем `root` идентификатор пользователя (суперпользователя) равен `0`, идентификатор группы (корневая группа) равен `0`, а домашний каталог `/root` и оболочка для данного пользователя — это `/bin/bash`. (Linux использует файл `/etc/shadow` для хранения зашифрованных данных пароля, поэтому в поле пароля здесь стоит `x`.) Можно изменить домашний каталог или оболочку, изменив значения в этом файле. Однако лучший способ сделать это — взять команду `usermod` (дополнительную информацию см. в подразделе «Изменение пользователей с помощью команды usermod» в главе 11).

На этом этапе любая команда, выполняемая из оболочки, реализуется с правами root, поэтому будьте осторожнее, так как у суперпользователя гораздо больше возможностей изменить (и повредить) систему, чем у обычного. По окончании введите команду `exit`. Если вы задействуете виртуальную консоль с интерфейсом рабочего

стола из другой консоли, нажмите сочетание клавиш Ctrl+Alt+F1, чтобы вернуться к графическому экрану входа в систему.

ПРИМЕЧАНИЕ

В Ubuntu по умолчанию пароль для учетной записи суперпользователя не установлен. Это означает, что, даже если учетная запись существует, вы не можете войти с ее помощью или использовать команду su, чтобы стать суперпользователем. Таким образом Ubuntu обеспечивает дополнительную безопасность. Чтобы выполнить команду от имени суперпользователя, необходимо применять команду sudo перед каждой командой.

Суперпользователь из оболочки (команда su)

Можно стать суперпользователем, войдя в систему под соответствующей учетной записью, однако иногда это не очень удобно.

Например, когда из учетной записи обычного пользователя вы хотите быстро внести изменения в систему и при этом не выполнять манипуляции с входом-выходом из нее. Или, к примеру, вам требуется войти в систему по сети, чтобы изменить в ней что-то, но система не позволяет суперпользователям входить по сети (обычная практика для защищенных систем Linux). Для подобных ситуаций есть решение — команда `su`. В любом окне терминала или оболочки введите следующее:

```
$ su
Password: ******
#
```

Когда отобразится приглашение командной строки, введите пароль суперпользователя. Приглашение обычного пользователя (`$`) станет приглашением суперпользователя (`#`). Теперь у вас есть полный доступ к запуску любых команд и файлов системы. Тем не менее при таком способе команда `su` не позволяет выполнить чтение данных в среде пользователя root. То есть можно ввести существующую и доступную команду, но получить сообщение о том, что она не найдена (`"Command Not Found"`). Чтобы исправить это, введите вместе с командой параметр с дефисом (`-`):

```
$ su -
Password: ******
#
```

Понадобится ввести пароль, но после этого все, что обычно происходит при входе в систему для суперпользователя, произойдет и здесь. Текущий каталог будет домашним каталогом root (вероятно, `/root`), и станет применяться переменная `PATH` суперпользователя. Если вы становитесь суперпользователем с помощью команды `su`, а не `su -`, каталоги и окружение текущего сеанса не меняются.

Можно использовать эту команду и для того, чтобы залогиниться в качестве другого пользователя. Это полезно для устранения неполадок, с которыми сталкивается конкретный пользователь (например, не может печатать или

отправлять письма). Так, чтобы получить права пользователя с именем `jsmith`, необходимо ввести:

```
$ su - jsmith
```

Даже если вы были суперпользователем до ввода этой команды, после этого у вас будут только права на открытие файлов и запуск программ, доступные для `jsmith`. Однако вам, как суперпользователю, после ввода команды `su`, чтобы стать другим пользователем, не нужно вводить его пароль для продолжения. Если же вы вводите эту команду как обычный пользователь, то должны набрать пароль нового пользователя.

Закончив задействовать права суперпользователя, вернитесь к предыдущей оболочке, выйдя из текущей. Для этого нажмите сочетание клавиш Ctrl+D или введите `exit`. Если вы администратор доступного нескольким пользователям компьютера, не оставляйте свою оболочку открытой на чужом экране, если не хотите, чтобы человек случайно или намеренно повредил систему.

Административный доступ через графический интерфейс

Как упоминалось ранее, когда инструменты с графическим интерфейсом запускаются от имени обычного пользователя (в Fedora, Red Hat Enterprise Linux или других системах Linux), необходимо ввести пароль суперпользователя, чтобы получить к ним доступ. Пароль суперпользователя дает привилегии при выполнении задач.

В системах Linux с рабочим столом GNOME 2 после ввода пароля на верхней панели появляется желтый значок, указывающий, что авторизация суперпользователя по-прежнему доступна для запуска других инструментов с графическим интерфейсом в текущем сеансе. Для систем с рабочими столами GNOME 3 необходимо вводить пароль суперпользователя каждый раз при запуске любого из инструментов настройки системы `system-config`.

Административный доступ с помощью команды sudo

Определенные пользователи могут получить административные права для определенных задач, набрав команду `sudo`, а затем команду, которую нужно запустить, и все это без ввода пароля суперпользователя. Файл `sudoers` — самый популярный способ предоставления прав любым пользователю или группе. С его помощью можно:

- назначить права доступа администратора любой группе командой `sudo`;
- назначить права доступа администратора для набора команд;
- дать пользователям права доступа, но без передачи пароля — им нужно ввести собственный пароль;

- разрешить пользователям запускать sudo вообще без ввода пароля (по желанию);
- отследить, какие пользователи выполняли административные команды в вашей системе. (Команда su сообщает, что кто-то зашел с паролем суперпользователя в систему, а как команда sudo регистрирует, какой пользователь запускает административную команду.)

Функция sudoers предоставляет полные или ограниченные права доступа любому пользователю, добавляя его в файл /etc/sudoers. Администратор определяет, какие права может иметь данный пользователь. Затем последний может выполнить любую команду, на которую он имеет права, введя перед ней команду sudo.

Рассмотрим пример применения пользователем по имени joe команды sudo, которая передает ему все права администратора.

СОВЕТ

Файл sudoers в Ubuntu по умолчанию дает права начальному пользователю в системе и привилегии членам группы sudo. Чтобы предоставить другому пользователю такие же привилегии, просто добавьте дополнительного пользователя в группу администратора, запустив команду visudo.

1. Как суперпользователь измените файл /etc/sudoers, выполнив команду visudo:

   ```
   # /usr/sbin/visudo
   ```

   По умолчанию файл открывается в редакторе vi, если только переменная EDITOR не установлена для другого редактора, подходящего для команды visudo (например, export EDITOR=gedit). Команда visudo блокирует файл /etc/sudoers и выполняет базовую проверку файла, чтобы убедиться, что он был отредактирован правильно.

ПРИМЕЧАНИЕ

Если возникли затруднения, запустите команду vimtutor, чтобы ознакомиться с руководством для редакторов vi и vim.

2. Добавьте следующую строку, чтобы дать пользователю joe полные права суперпользователя:

   ```
   joe     ALL=(ALL)     ALL
   ```

   Пользователь joe должен ввести собственный пароль, чтобы применить административные команды. Чтобы дать joe все права без использования пароля, введите вместо приведенной ранее строки следующее:

   ```
   joe     ALL=(ALL)     NOPASSWD: ALL
   ```

3. Сохраните изменения в файле /etc/sudoers (в vi введите Esc, а затем :wq). Далее приведен пример сеанса пользователя joe после того, как ему были назначены привилегии с помощью команды sudo:

   ```
   [joe]$ sudo touch /mnt/testfile.txt
     We trust you have received the usual lecture
   ```

```
  from the local System Administrator. It usually
  boils down to these two things:
    #1) Respect the privacy of others.
    #2) Think before you type.
Password: *********
[joe]$ ls -l /mnt/testfile.txt
-rw-r--r--. 1 root root 0 Jan  7 08:42 /mnt/testfile.txt
[joe]$ rm /mnt/testfile.txt
rm: cannot remove '/mnt/testfile.txt': Permission denied
[joe]$ sudo rm /mnt/textfile.txt
[joe]$
```

В этом сеансе пользователь joe запускает команду `sudo`, чтобы создать файл (`/mnt/textfile.txt`) в каталоге, для которого у него нет прав на запись. Появляется диалоговое окно предупреждения, после которого нужно ввести пароль (пароль пользователя joe, а не пароль администратора).

Даже после того, как пользователь joe ввел свой пароль, он все равно должен применять команду `sudo` для запуска последующих административных команд от имени суперпользователя (то есть команда `rm` даст сбой, а вот `sudo rm` будет выполнена). Обратите внимание на то, что для второй команды `sudo` пароль не запрашивается. Дело в том, что в системах RHEL и Fedora после ввода пароля можно использовать команду `sudo` в течение пяти минут, не вводя его снова. В Ubuntu такое не сработает, если не изменить период времени до нужной продолжительности, установив другое значение переменной `passwd_timeout` в файле `/etc/sudoers`.

В предыдущем примере пользователю joe права предоставляются по типу «все или ничего». Однако файл `/etc/sudoers` очень гибкий и позволяет пользователям и группам работать с отдельными приложениями или группами приложений. Обратитесь к справочным страницам `sudoers` и `sudo`, чтобы узнать, как их настроить.

Административные команды, файлы конфигурации и файлы журналов

Множество команд, файлов конфигурации и файлов журналов находятся в одних и тех же местах файловой системы независимо от используемого дистрибутива. В следующих разделах рассмотрим, как найти эти важные элементы системы.

ПРИМЕЧАНИЕ

Если инструменты администрирования с графическим интерфейсом так хороши, зачем нужно знать об административных файлах? Во-первых, инструменты с графическим интерфейсом различаются в зависимости от версии Linux, а вот многие базовые файлы конфигурации одинаковы в любом дистрибутиве. Так что если научиться обращаться с ними, то можно работать практически с любой системой Linux. Кроме того, если функция сбоит или нужно сделать что-то не поддерживаемое графическим интерфейсом, специалисты Linux практически всегда советуют запускать команды или непосредственно изменять файл конфигурации.

Административные команды

Только суперпользователь может задействовать административные команды.

Когда вы входите в систему как суперпользователь (или с помощью команды `su` — из оболочки), переменная `$PATH` открывает каталоги с командами для администратора. Раньше они включали в себя следующие каталоги:

- `/sbin` — изначально содержал команды, необходимые для загрузки системы, включая команды для проверки файловых систем (`fsck`) и включения устройств подкачки (`swapon`);
- `/user/sbin` — изначально содержал команды для управления учетными записями пользователей (например, `useradd`) и проверки процессов, которые удерживали файлы открытыми (например, `lsof`). Здесь же содержатся команды, выполняемые как демоны. *Демон-процессы (daemon)* — это процессы, которые выполняются в фоновом режиме, ожидая запроса служб, например доступ к принтеру или веб-странице. (Поищите команды, которые заканчиваются на `d`, например `sshd`, `pppd`, и `cupsd`.)

В последних версиях Ubuntu, RHEL и Fedora все административные команды из этих двух каталогов хранятся в каталоге `/usr/sbin`, символически связанном с `/sbin`. Кроме того, в переменную `PATH` суперпользователя добавляется только `/usr/sbin`, как и в переменную `PATH` всех обычных пользователей.

Некоторые административные команды содержатся в каталогах обычных пользователей, таких как `/bin` и `/usr/bin`. Обычно это команды, у которых есть доступные для всех параметры. Например, команду `/bin/mount` любой пользователь может задействовать для перечисления смонтированных файловых систем, но только суперпользователь — для самого монтирования файловых систем. (Однако некоторые компьютеры предназначены для того, чтобы обычные пользователи могли применять команду `mount` для монтирования CD, DVD или других съемных носителей.)

ПРИМЕЧАНИЕ

Руководство по монтированию файловой системы см. в главе 12.

Чтобы найти команды, предназначенные в первую очередь для системного администратора, ознакомьтесь с разделом 8 справочных страниц (в каталоге `/usr/share/man/man8`). В нем содержатся описания и параметры большинства административных команд Linux. Чтобы добавить команды в свою систему, попробуйте сохранить их в каталоги, например, `/usr/local/bin` или `/usr/local/sbin`. Некоторые дистрибутивы Linux автоматически добавляют эти каталоги в переменную `PATH`, обычно перед стандартными каталогами `bin` и `sbin`. Таким образом, команды, установленные в этих каталогах, не только доступны, но и могут переопределять команды с тем же именем в других каталогах. Некоторые сторонние приложения, не входящие в состав дистрибутивов Linux, иногда помещаются в каталоги `/usr/local/bin`, `/opt/bin` или `/usr/local/sbin`.

Файлы конфигурации

Файлы конфигурации — это еще одна основа администрирования Linux. Почти все настройки конкретного компьютера — учетные записи пользователей, сетевые адреса или предпочтения графического интерфейса — сохраняются в текстовых файлах. Это и преимущество, и недостаток.

Преимущество обычных текстовых файлов заключается в том, что их легче читать и редактировать. Подойдет любой текстовый редактор. Недостатком же является то, что при редактировании файлов конфигурации проверка ошибок обычно не выполняется. Иногда нужно запустить программу, которая прочитает эти файлы (например, сетевой демон или рабочий стол X), чтобы узнать, правильно ли они настроены.

Некоторые файлы конфигурации применяют стандартные структуры, например XML, для хранения информации, но большинство этого не делает. Поэтому нужно изучить конкретные структурные правила для каждого конфигурационного файла. Запятая или кавычка в неправильном месте иногда может привести к сбою всего интерфейса. Проверить правильность структуры многих файлов конфигурации можно множеством способов.

В некоторых пакетах программного обеспечения есть своя команда для проверки работоспособности файла конфигурации перед запуском службы. Например, команда `testparm` используется на сервере Samba для проверки работоспособности файла `smb.conf`. В других случаях проверить файл конфигурации может демон-процесс. Например, запустите команду `httpd-t`, чтобы проверить конфигурацию веб-сервера Apache перед его запуском.

ПРИМЕЧАНИЕ

Отдельные текстовые редакторы, например vim (но не vi), понимают структуры некоторых типов файлов конфигурации. Если открыть такой файл в vim, то различные элементы файла будут отображаться разными цветами. В частности, строки с комментариями будут отличаться по цвету от строк с данными.

В этой книге есть описания конфигурационных файлов, необходимых для настройки различных функций систем Linux. Два основных места расположения файлов конфигурации — это домашний каталог, где хранятся личные файлы конфигурации, и каталог `/etc`, в котором хранятся общесистемные файлы конфигурации.

Далее приведены описания каталогов и подкаталогов, содержащих полезные файлы конфигурации. За описаниями следуют особенно интересные отдельные файлы, находящиеся в файле `/etc`. Изучение содержимого таких файлов позволяет хорошо разобраться в том, как работает администрирование систем Linux.

- `$HOME` — все пользователи хранят в своих домашних каталогах информацию, которая определяет, как ведут себя их учетные записи. Многие файлы конфигурации хранятся непосредственно в домашнем каталоге каждого пользователя (например, в `/home/joe`) и начинаются с точки (`.`), поэтому не появляются

в каталоге при использовании стандартной команды `ls` (для их просмотра необходимо ввести `ls -a`). Аналогично точечные файлы и каталоги по умолчанию не отображаются в большинстве окон менеджера файлов. Существуют точечные файлы, которые определяют поведение оболочки каждого пользователя, внешний вид рабочего стола и параметры текстового редактора. Есть даже файлы, которые находятся в каталоге `$HOME/.ssh` каждого пользователя и настраивают права для входа в удаленные системы. (Чтобы увидеть имя своего домашнего каталога, введите команду `echo $HOME` из оболочки.)

- `/etc` — этот каталог содержит большинство основных файлов конфигурации системы Linux.
- `/etc/cron*` — содержат файлы, определяющие, как команда `crond` запускает приложения ежедневно (`cron.daily`), ежечасно (`cron.hourly`), ежемесячно (`cron.monthly`) или еженедельно (`cron.weekly`).
- `/etc/cups` — содержит файлы для настройки службы печати CUPS.
- `/etc/default` — содержит файлы, устанавливающие значения по умолчанию для различных утилит. Например, файл для команды `useradd` определяет номер группы по умолчанию, домашний каталог, срок действия пароля, оболочку и каталог шаблонов (`/etc/skel`), используемый при создании новой учетной записи пользователя.
- `/etc/httpd` — содержит множество файлов для настройки поведения веб-сервера Apache (в частности, демона `httpd`). (В Ubuntu и других системах Linux вместо него применяется файл `/etc/apache` или `/etc/apache2`.)
- `/etc/mail` — содержит файлы для настройки почтового агента `sendmail`.
- `/etc/postfix` — содержит файлы конфигурации для агента `postfixmail`.
- `/etc/ppp` — содержит несколько конфигурационных файлов для настройки протокола «точка — точка» (Point-to-Point Protocol, PPP) при подключении компьютера к Интернету. (PPP чаще всего использовался, когда были широко распространены модемы.)
- `/etc/rc?.d` — существует отдельный каталог `rc?.d` для каждого допустимого состояния системы: `rc0.d` (выключенное), `rc1.d` (однопользовательское), `rc2.d` (многопользовательское), `rc3.d` (многопользовательское плюс сетевое), `rc4.d` (пользовательское), `rc5.d` (многопользовательское, сетевое плюс графический интерфейс входа) и `rc6.d` (состояние перезагрузки). Эти каталоги поддерживаются для совместимости со старыми службами инициализации UNIX SystemV.
- `/etc/security` — содержит файлы, которые устанавливают различные условия безопасности по умолчанию для компьютера, в основном определяя, как выполняется аутентификация. Эти файлы — часть пакета подключаемых модулей аутентификации `pam` (pluggable authentication modules).
- `/etc/skel` — все файлы, содержащиеся в этом каталоге, автоматически копируются в домашний каталог пользователя при добавлении этого пользователя в систему. По умолчанию большинство этих файлов точечные (`.`),

например kde (каталог для настройки параметров рабочего стола KDE по умолчанию) и bashrc (каталог для установки значений по умолчанию, применяемых с оболочкой bash).

- /etc/sysconfig — содержит важные файлы конфигурации системы, которые создаются и поддерживаются различными службами, включая firewalld, samba и большинство сетевых служб. Эти файлы важны для дистрибутивов Linux, таких как Fedora и RHEL, которые задействуют инструменты администрирования с графическим интерфейсом, совершенно не используемые в других системах Linux.
- /etc/systemd — содержит файлы, связанные с командой systemd, которая управляет процессом загрузки и системными службами. В частности, при запуске команды systemctl файлы включения и отключения служб хранятся в подкаталогах каталога /etc/systemd system.
- /etc/xinetd.d — содержит набор файлов, каждый из которых определяет сетевую службу по требованию, которую демон xinetd прослушивает через определенный порт. Когда демон xinetd получает запрос на включение службы, он использует информацию, содержащуюся в этих файлах, чтобы определить, какие процессы следует запустить для обработки запроса.

Далее приведены интересные файлы конфигурации, находящиеся в каталоге /etc.

- aliases — может содержать списки рассылки для почтовых служб Linux. (В Ubuntu этот файл находится в файле /etc/mail после установки пакета sendmail.)
- bashrc — устанавливает общесистемные значения по умолчанию для пользователей оболочки bash. (В некоторых дистрибутивах Linux может называться bash.bashrc.)
- crontab — задает время выполнения автоматических задач и переменных, связанных с функцией cron (например, SHELL или PATH).
- csh.cshrc (или cshrc) — устанавливает общесистемные значения по умолчанию для пользователей оболочки csh (оболочка C).
- exports — содержит список локальных каталогов, доступных для совместного применения удаленными компьютерами с помощью сетевой файловой системы (NFS).
- fstab — определяет устройства для носителей информации (жесткий диск, DVD, CD-ROM и т. д.) и места их установки в системе Linux. Используется командой mount для выбора того, какие файловые системы монтировать при первой загрузке системы.
- group — назначает имена групп и их идентификаторы (gid), определенные в системе. Групповые права в Linux определяются вторым из трех наборов битов чтения, записи и выполнения rwx (read, write, execute), связанных с каждым файлом и каталогом.

- `gshadow` — содержит теневые пароли для групп.
- `host.conf` — используется более старыми приложениями для установки местоположений, в которых доменные имена (например, `redhat.com`) ищутся в сетях TCP/IP (например, в Интернете). По умолчанию выполняется поиск локального файла, а затем всех записей сервера имен в файле `resolv.conf`.
- `hostname` — содержит имя узла локальной системы (доступен в RHEL 7 и в последних версиях систем Fedora и Ubuntu).
- `hosts` — содержит IP-адреса и имена узлов, которые вы можете получить со своего компьютера. (Обычно этот файл используется только для хранения имен компьютеров в локальной или небольшой частной сети.)
- `inittab` — в более ранних версиях систем Linux в данном файле содержалась информация, которая определяла, какие программы запускаются и останавливаются при загрузке Linux, выключении или переходе в различные промежуточные состояния. Этот файл конфигурации считывался первым после запуска процесса `init`. Он больше не применяется в системах Linux, поддерживающих функцию `systemd`.
- `mtab` — содержит список файловых систем, которые в данный момент монтируются.
- `mtools.conf` — содержит настройки, применяемые инструментами DOS в Linux.
- `named.conf` — содержит настройки DNS, если используется собственный DNS-сервер (пакет `bind` или `bind9`).
- `nsswitch.conf` — с помощью диспетчера службы имен (Name Service Switch, NSS) определяет, откуда (через локальный узел или через сетевые службы) поступает важная системная информация (учетные записи пользователей, сопоставления имени узла с адресом и т. д.).
- `ntp.conf` — использует информацию, необходимую для запуска протокола сетевого времени (Network Time Protocol, NTP).
- `passwd` — хранит информацию об учетных записях всех пользователей в локальной системе. Хранит также другую информацию наподобие имен домашнего каталога и оболочки по умолчанию. (Редко хранит сами пароли пользователей, они обычно находятся в файле `/etc/shadow`.)
- `printcap` — содержит информацию о принтерах, подключенных к компьютеру. (Если файл `printcap` не существует, информацию о принтере можно найти в каталоге `/etc/cups`.)
- `profile` — устанавливает общесистемную среду и программы запуска для всех пользователей. Этот файл считывается при входе пользователя в систему.
- `protocols` — устанавливает номера протоколов и имена для различных интернет-служб.
- `rpc` — определяет имена и номера вызовов удаленных процессов.
- `services` — определяет имена служб TCP/IP и UDP и их назначения портам.

- shadow — содержит зашифрованные пароли для пользователей из файла passwd. (Рассматривается как более безопасный способ хранения паролей, чем исходный зашифрованный пароль в файле passwd. Файл passwd должен быть общедоступным, а вот файл shadow может быть доступным только для суперпользователя.)
- shells — перечисляет интерпретаторы командной строки оболочки (bash, sh, csh и т. д.), доступные в системе, а также их расположение.
- sudoers — задает команды, на выполнение которых у пользователя нет прав, но которые могут им выполняться. В частности, этот файл применяется для предоставления конкретным пользователям прав суперпользователя.
- rsyslog.conf — определяет, какие сообщения журнала собирает демон rsyslogd и в каких файлах они хранятся. (Обычно сообщения журнала хранятся в файлах, находящихся в каталоге /var/log.)
- xinetd.conf — содержит простую информацию о конфигурации, используемую демоном xinetd. Этот файл в основном указывает на каталог /etc/xinetd.d, в котором находится информация об отдельных службах.

Другой каталог, /etc/X11, включает подкаталоги, каждый из которых содержит общесистемные файлы конфигурации, задействуемые процессами X и различными X-оконными менеджерами, доступными для Linux. Файл xorg.con настраивает компьютер, монитор и каталоги конфигурации с файлами, используемыми командами xdm или xinit, чтобы запустить процесс X.

Каталоги, относящиеся к менеджерам окон, содержат файлы со значениями по умолчанию, которые устанавливаются при запуске одного из этих менеджеров в системе. Менеджер окон twm может иметь в этих каталогах общесистемные файлы конфигурации.

Файлы журналов и systemd

Одно из действий, которые система Linux выполняет хорошо, — это отслеживание состояний системы, что очень полезно, учитывая массив происходящего в системе.

Иногда при попытке запуска новой задачи происходит ошибка, и не всегда понятно почему. Или же необходимо проконтролировать систему, чтобы увидеть, не пытается ли кто-то незаконно проникнуть в ваш компьютер. В любом случае необходимо получать информационные сообщения, поступающие от ядра и служб, работающих в системе.

В системах Linux, которые не используют инструмент systemd, основной утилитой для регистрации ошибок и сообщений отладки является демон rsyslogd. (В некоторых старых версиях системы Linux — демоны syslogd и syslogd.) Можно применять rsyslogd с системами systemd, однако у systemd есть собственный метод сбора и отображения сообщений, называемый журналом systemd (команда journalctl).

Команда journalctl для просмотра журнала systemd

Основной командой для просмотра сообщений из журнала `systemd` является `journalctl`. Процесс загрузки, ядро и все управляемые `systemd` службы направляют сообщения о состоянии и ошибках в журнал `systemd`.

С помощью команды `journalctl` можно отображать сообщения журнала различными способами, например:

```
# journalctl
# journalctl --list-boots | head
-2 93bdb6164... Sat 2020-01-04 21:07:28 EST—Sat 2020-01-04 21:19:37 EST
-1 7336cb823... Sun 2020-01-05 10:38:27 EST—Mon 2020-01-06 09:29:09 EST
 0 eaebac25f... Sat 2020-01-18 14:11:41 EST—Sat 2020-01-18 16:03:37 EST
# journalctl -b 488e152a3e2b4f6bb86be366c55264e7
# journalctl -k
```

В приведенных примерах команда `journalctl` без параметров позволяет просматривать все сообщения в журнале. Чтобы перечислить идентификаторы загрузки для каждой загрузки системы, используйте параметр `-list-boots`. Чтобы просмотреть сообщения, связанные с конкретным сеансом загрузки, применяйте для него параметр `-b`. Чтобы увидеть только сообщения ядра, берите параметр `-k`. Примеры:

```
# journalctl _SYSTEMD_UNIT=sshd.service
# journalctl PRIORITY=0
# journalctl -a -f
```

Используйте параметр `_SYSTEMD_UNIT=` для вывода сообщений о работе определенных служб (в примере — служба `shd`) или при работе любого другого файла модуля `systemd` (например, для других служб или монтирования). Чтобы просмотреть сообщения, связанные с определенным уровнем системного журнала, установите значение переменной `PRIORITY` от 0 до 7. В примере отображаются только экстренные сообщения (`0`). Чтобы отследить сообщения по мере их поступления, применяйте параметр `-f`, чтобы отобразить все поля информации — параметр `-a`.

Управление сообщениями с помощью команды rsyslogd

Команда `rsyslogd` и ее предшественница `syslogd` собирают сообщения и направляют их в файлы или удаленные узлы журнала. Журнал ведется в соответствии с информацией, содержащейся в файле `/etc/rsyslog.conf`. Сообщения обычно направляются в файлы журналов, которые находятся в каталоге `/var/log`, но могут быть направлены также на узлы журналов для обеспечения дополнительной безопасности.

Вот несколько распространенных файлов журнала:

- `boot.log` — содержит загрузочные сообщения о службах по мере их запуска;
- `messages` — содержит множество общих информационных сообщений о системе;

- `secure` — содержит сообщения, связанные с безопасностью, такие как вход в систему или любые другие действия, которые аутентифицируют пользователей.

В главе 13 описана настройка команды `rsyslogd`.

Использование других административных учетных записей

Вряд ли вы слышали о возможности входа в систему под другими административными учетными записями, за исключением суперпользователя. В системах UNIX было довольно распространенной практикой иметь несколько различных учетных записей, которые позволяли распределять административные задачи между несколькими пользователями. Например, у пользователей, которые сидят рядом с принтером, могли быть дополнительные права `lp` для управления выводом на печать.

В любом случае административные учетные записи в Linux доступны, однако подобный вход непосредственно в систему по умолчанию отключен. Учетные записи поддерживаются в основном для обеспечения права собственности на файлы и процессы, связанные с конкретными службами. Если демоны запускаются под отдельными административными учетными записями, то, когда один из них оказывается взломан, получить доступ к другим процессам и файлам нельзя. Рассмотрим следующие примеры.

- `lp` — пользователь владеет файлом журнала печати `/var/log/cups`, различными файлами кэша печати и `spool`-файлами. Домашний каталог для `lp` — это `/var/spool/lpd`.
- `apache` — пользователь может настроить файлы содержимого и каталоги на веб-сервере Apache. Он применяется в основном для запуска процессов веб-сервера (`httpd`) в системах RHEL и Fedora, в то время как пользователь `www-data` запускает службу Apache (`apache2`) в системах Ubuntu.
- `avahi` — пользователь использует демон `avahi` и набор технологий `zeroconf` для работы служб в сети.
- `chrony` — пользователь запускает демон `chronyd`, который обеспечивает точную работу часов компьютера.
- `postfix` — пользователь владеет различными каталогами и файлами `spool` почтового сервера. Он запускает демонические процессы для работы службы postfix (`master`).
- `bin` — пользователь владеет множеством команд в каталоге `/bin` в традиционных системах UNIX, кроме Ubuntu, Fedora и Gentoo, потому что в них суперпользователь владеет большинством исполняемых файлов. Домашний каталог для `bin` — это `/bin`.
- `news` — пользователь может управлять новостными сервисами в Интернете в зависимости от того, какие права установлены для каталога `/var/spool/news`

и других ресурсов, связанных с новостями. Домашний каталог для `news` — это `/etc/news`.

- `rpc` — пользователь запускает демон удаленного вызова процедур (`rpcbind`), который применяется для получения вызовов служб в хост-системе. Служба NFS задействует службу RPC.

По умолчанию административные учетные записи в предыдущем списке отключены. Вам нужно будет изменить оболочку по умолчанию с ее текущими настройками (обычно `/sbin/nologin` или `/bin/false`) на реальную оболочку (чаще всего `/bin/bash`), чтобы иметь возможность войти в систему как перечисленные ранее пользователи. Однако, как уже упоминалось, они не предназначены для интерактивного входа в систему.

Проверка и настройка оборудования

В идеальном мире после установки и загрузки Linux все ваше оборудование сразу обнаружено и доступно для использования. Сейчас в системах Linux процесс обнаружения оборудования улучшился, но бывает так, что необходимо предпринять специальные действия, чтобы оно начало работать. Кроме того, все более частое применение съемных USB-устройств (CD, DVD, USB-накопителей, цифровых камер и съемных жестких дисков) увеличило важность следующих характеристик Linux:

- способности эффективно управлять съемным оборудованием;
- умения видеть одну часть оборудования по-разному (например, принтер должен быть виден как факс, сканер, накопитель и, конечно, принтер).

Добавленные за последние несколько лет функции ядра Linux позволили кардинально изменить процессы обнаружения аппаратных устройств и управления ими. Подсистема Udev управляет устройствами и обслуживает их файлы.

Не пугайтесь, если это прозвучало странно. Подсистема создана для облегчения работы с Linux. Встроенные в ядро функции позволяют автоматизировать обнаружение устройств и сделать процесс гибким.

- **Автоматизация.** Большинство распространенных устройств автоматически обнаруживаются и идентифицируются. Добавлены интерфейсы доступа к оборудованию. Информация о том, что оборудование подключено (или удалено), передается на уровень пользователя, где приложения, связанные с изменением оборудования, готовы смонтировать его и/или запустить приложение (например, менеджер изображений или музыкальный плеер).
- **Гибкость.** При желании можно изменить процесс подключения устройства. Например, функции, встроенные в рабочие столы GNOME и KDE позволяют выбирать, что происходит при подключении музыкального CD или DVD с данными либо цифровой камеры. Или, например, можно изменить программу обработки этого оборудования.

В следующих разделах мы узнаем, как правильно работать с оборудованием в Linux. Во-первых, рассмотрим, как проверить информацию об оборудовании вашей системы. Во-вторых, как настроить Linux для работы со съемными носителями. И в третьих, как использовать инструменты ручной загрузки и работы с драйверами оборудования, которое не обнаруживается и не загружается должным образом.

Проверка оборудования

При загрузке системы ядро определяет подключенное оборудование и запускает драйверы, которые позволяют Linux работать с ним. Поскольку сведения об обнаружении оборудования быстро прокручиваются на экране при загрузке, необходимо повторно отобразить их после запуска системы, чтобы просмотреть сообщения о потенциальных проблемах.

Существует несколько способов просмотра сообщений о загрузке ядра после запуска Linux. Любой пользователь может запустить команду `dmesg`, чтобы увидеть, какое оборудование обнаружено и какие драйверы загружены ядром во время запуска. Поскольку ядро генерирует новые сообщения, они также доступны для команды `dmesg`.

Второй способ просмотра загрузочных сообщений — это команда `journalctl`, которая отображает сообщения, связанные с конкретным сеансом загрузки, как описано ранее в этой главе.

ПРИМЕЧАНИЕ

После запуска системы многие сообщения ядра отправляются в файл /var/log/messages. Так, например, если нужно увидеть, что происходит при подключении USB-накопителя, можно ввести команду tail -f /var/log/messages и понаблюдать, как создаются устройства и точки монтирования. А команду journalctl -f использовать для отслеживания сообщений, поступающих в журнал systemd.

Далее приведен пример интересных фрагментов выходных данных команды `dmesg`:

```
$ dmesg | less
[    0.000000] Linux version 5.0.9-301.fc30.x86_64
    (mockbuild@bkernel04.phx2.fedoraproject.org) (gcc version 9.0.1
20190312
    (Red Hat 9.0.1-0.10) (GCC)) #1 SMP Tue Apr 23 23:57:35 UTC 2019
[     0.000000] Command line:
    BOOT_IMAGE=(hd0,msdos1)/vmlinuz-5.0.9-301.fc30.x86_64
    root=/dev/mapper/fedora_localhost--live-root ro
    resume=/dev/mapper/fedora_localhost--live-swap
    rd.lvm.lv=fedora_localhost-live/root
    rd.lvm.lv=fedora_localhost-live/swap rhgb quiet
...
             S31B1102 USB DISK         1100 PQ: 0 ANSI: 0 CCS
[79.177466] sd 9:0:0:0: Attached scsi generic sg2 type 0
```

```
[79.177854] sd 9:0:0:0: [sdb]
             8343552 512-byte logical blocks: (4.27 GB/3.97 GiB)
[79.178593] sd 9:0:0:0: [sdb] Write Protect is off
```

Из этого вывода сначала отображается версия ядра Linux, а затем параметры командной строки ядра. В последних строках — описание подключенного USB-накопителя на 4 Гбайт.

Если при обнаружении или загрузке драйверов что-то пойдет не так, используйте эту информацию, чтобы увидеть название и номер модели оборудования. Затем поищите информацию на форумах Linux или в документации, чтобы решить проблему. Существует также несколько дополнительных команд, которые позволяют просмотреть подробную информацию об оборудовании компьютера после запуска системы. Команда `lspci` выводит список шин ввода-вывода для подключения устройств (PCI) на компьютере. Фрагмент вывода:

```
$ lspci
00:00.0 Host bridge: Intel Corporation
     5000X Chipset Memory ControllerHub
00:02.0 PCI bridge: Intel Corporation 5000 Series Chipset
     PCI Express x4 Port 2
00:1b.0 Audio device: Intel Corporation 631xESB/632xESB
     High Definition Audio Controller (rev 09)
00:1d.0 USB controller: Intel Corporation 631xESB/632xESB/3100
     Chipset UHCI USBController#1 (rev 09)
07:00.0 VGA compatible controller: nVidia Corporation NV44
0c:02.0 Ethernet controller: Intel Corporation 82541PI
     Gigabit Ethernet Controller (rev 05)
```

Мост хоста соединяет локальную шину с другими компонентами моста PCI. Я сократил список, чтобы показать информацию о подключенных устройствах с разными функциями: звук (аудиоустройство — audio device), накопители и другие USB-устройства (USB-контроллер — USB controller), видеодисплеи (VGA-совместимый контроллер — VGA compatible controller) и проводные сетевые карты (Ethernet-контроллер — Ethernet controller). Если возникают проблемы с работой любого из этих устройств, обратите внимание на их модели и номера и поищите информацию о них в Интернете.

Чтобы получить более подробный вывод из `lspci`, добавьте один или несколько параметров `-v`. Например, используя `lspci -vvv`, я получил сведения о своем контроллере Ethernet, включая отклик, возможности контроллера и драйвер Linux (`e1000`), применяемый для устройства.

Команда `lsusb` особенно хорошо подходит для вывода информации о USB-устройствах. По умолчанию `lsusb` перечисляет информацию о USB-концентраторах, установленных на компьютере, и всех подключенных USB-устройствах:

```
$ lsusb
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 002 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 003 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 004 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
```

```
Bus 005 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 002 Device 002: ID 413c:2105 Dell Computer Corp.
    Model L100 Keyboard
Bus 002 Device 004: ID 413c:3012 Dell Computer Corp.
    Optical Wheel Mouse
Bus 001 Device 005: ID 090c:1000 Silicon Motion, Inc. -
    Taiwan 64MB QDI U2 DISK
```

В предыдущем примере отображаются модели клавиатуры, мыши и USB-накопителя, подключенных к компьютеру. Как и в случае с `lspci`, добавьте один или несколько параметров `-v`, чтобы увидеть более подробную информацию.

Для просмотра подробной информации о процессоре выполните команду `lscpu`. Она дает основную информацию о процессорах компьютера:

```
$ lscpu
Architecture:           x86_64
CPU op-mode(s):         32-bit, 64-bit
CPU(s):                 4
On-line CPU(s) list:    0-3
Thread(s) per core:     1
Core(s) per socket:     4
...
```

Из части данных `lscpu` видно, что это 64-битная система (x86-64), она может работать в 32- или 64-битном режимах и в ней четыре процессора.

Управление съемным оборудованием

Системы Linux (например, Red Hat Enterprise Linux, Fedora и др.), которые поддерживают полную среду рабочего стола GNOME, применяют простые инструменты с графическим интерфейсом для настройки подключения часто используемых съемных устройств к компьютеру. То есть при запущенном рабочем столе GNOME можно просто подключить USB-устройство или вставить CD или DVD — и появится окно для работы с устройством.

Несмотря на то что различные среды рабочего стола задействуют одни и те же базовые механизмы (в частности, Udev) для обнаружения съемного оборудования и присвоения ему имен, инструменты, а также способы монтирования и применения все же различаются. Подсистема Udev с помощью демона `udevd` создает и удаляет устройства (каталог `/dev`) по мере их ввода/вывода. Однако параметры, интересующие пользователей настольной системы Linux, настраиваются с помощью простых инструментов.

Файловый менеджер Nautilus с рабочего стола GNOME позволяет настроить, что происходит при подключении и отключении съемных устройств, в окне File Management Preferences (Параметры управления файлами). Описание настроек в этом разделе основано на GNOME 3.32 в Fedora 30.

С рабочего стола GNOME 3.32 перейдите в окно Activities (Обзор) и введите `Removable Media` (Съемный носитель). Затем выберите пункт Removable Media Setting (Настроить съемный носитель).

Откроется окно с доступными настройками съемного оборудования. Эти параметры устанавливают, как обрабатываются съемные носители при их вставке или подключении. В большинстве случаев подходящий вариант действий предлагается системой.

- **Звуковой компакт-диск (CD audio).** При вставке CD можно выбрать следующие возможности: спрашивать, что делать (по умолчанию); ничего не делать; открыть папку или выбрать один из предложенных проигрывателей для воспроизведения. Программы Rhythmbox (музыкальный проигрыватель), Audio CD Extractor (запись дисков) и Brasero (запись дисков) — одни из основных вариантов работы с CD.
- **Видео-DVD (DVD video).** После вставки DVD с видео программа предложит варианты действий с ним. Можно изменить вариант по умолчанию, чтобы запустить Totem (видеоплеер), Brasero (DVD-запись) или другой установленный медиаплеер (например, MPlayer).
- **Музыкальный плеер (Music player).** Если носитель содержит аудиофайлы, программа спросит о дальнейших действиях. Можно указать, чтобы Rhythmbox или какой-либо другой музыкальный плеер начал воспроизведение файлов.
- **Фото (Photos).** Когда вставленный носитель (например, карта памяти цифровой камеры) содержит цифровые изображения, программа спросит о дальнейших действиях. Можно выбрать: ничего не делать, или открывать изображения в Shotwell image viewer (приложение по умолчанию для просмотра изображений на рабочем столе GNOME) или в другом установленном менеджере фотографий.
- **Приложение (Software).** Если на носителе содержится устанавливаемое программное обеспечение, то его окно установки открывается по умолчанию. Чтобы изменить это, выберите один из предложенных вариантов (спрашивать, что делать, ничего не делать или открыть содержимое в папке).
- **Другой носитель (Other Media).** В поле Type (Тип) выберите тип часто используемого носителя. Можно указать, какие действия совершать, к примеру, для звукового DVD или чистых дисков Blu-ray, CD или DVD. Можно также выбрать, какие приложения запускать для видеодисков, электронных книг и дисков Picture CD.

Обратите внимание на то, что описанные настройки действуют только для пользователя, который в данный момент вошел в систему. Если у системы несколько пользователей, каждый выбирает себе подходящий способ обработки съемных носителей.

ПРИМЕЧАНИЕ

Видеоплеер Totem не воспроизводит DVD с фильмами, если не установлено дополнительное программное обеспечение для расшифровки диска. Если хотите смотреть коммерческие DVD-фильмы в Linux, вам следует изучить юридические и другие вопросы, касающиеся воспроизведения фильмов.

Параметры подключения обычных USB-накопителей или жестких дисков в этом окне не перечисляются. Однако при подключении одного из таких накопителей к компьютеру они автоматически загружаются в систему (с именами `/dev/sda`, `/dev/sdb` и т. д.). Любые файловые системы, найденные на этих устройствах, автоматически монтируются в каталоге `/run/media/username`, и система предложит открыть окно Nautilus (Файлы) для просмотра файлов на этих устройствах. Это делается автоматически, и специальные настройки для этого не нужны.

Когда закончите работу с USB-накопителем, щелкните правой кнопкой мыши на имени устройства в окне Nautilus (Файлы) и выберите пункт Safely Remove Drive (Безопасное извлечение). Это действие размонтирует диск и удалит точку монтирования в каталоге `/run/media/username`. После этого можете безопасно отсоединить USB-накопитель от компьютера.

Загружаемые модули

Если вы добавили на свой компьютер оборудование, но оно не было обнаружено должным образом, возможно, потребуется загрузить модуль для него вручную. В Linux существует набор команд для загрузки, выгрузки и получения информации о модулях оборудования.

Модули ядра устанавливаются в подкаталоги `/lib/modules/`. Имя каждого подкаталога основано на номере ядра. Драйверы для ядра 5.3.8-200.fc30.x86_64 содержатся в каталоге `/lib/modules/5.3.8-200.fc30.x86_64`. Модули в этих каталогах могут загружаться и выгружаться по мере необходимости.

В Linux доступны команды для перечисления, загрузки, выгрузки и получения информации о модулях. В следующих разделах будет описано, как использовать эти команды.

Перечисление загружаемых модулей

Чтобы увидеть, какие модули загружены в активное ядро компьютера в данный момент, возьмите команду `lsmod`, например:

```
# lsmod
Module                  Size  Used by
vfat                   17411  1
fat                    65059  1 vfat
uas                    23208  0
usb_storage            65065  2 uas
fuse                   91446  3
ipt_MASQUERADE         12880  3
xt_CHECKSUM            12549  1
nfsv3                  39043  1
rpcsec_gss_krb5        31477  0
nfsv4                 466956  0
dns_resolver           13096  1 nfsv4
nfs                   233966  3 nfsv3,nfsv4
```

```
.
.
.
i2c_algo_bit           13257  1 nouveau
drm_kms_helper         58041  1 nouveau
ttm                    80772  1 nouveau
drm                   291361  7 ttm,drm_kms_helper,nouveau
ata_generic            12923  0
pata_acpi              13053  0
e1000                 137260  0
i2c_core               55486  5 drm,i2c_i801,drm_kms_helper
```

Данный вывод отображает множество модулей, загруженных в систему Linux, включая модуль карты сетевого интерфейса (`e1000`).

Чтобы найти информацию о любом из загруженных модулей, используйте команду `modinfo`, например:

```
# /sbin/modinfo -d e1000
Intel(R) PRO/1000 Network Driver
```

Не для всех модулей доступны описания, и если данных нет, то они не выводятся на экран. В данном же случае `e1000` описывается как модуль сетевого драйвера `Intel(R) PRO/1000`. С помощью параметра `-a` можно увидеть имя создателя модуля, а с помощью параметра `-n` — объектный файл, представляющий модуль. В информации об авторе часто содержится адрес электронной почты создателя драйвера, поэтому с ним можно связаться, если возникнут проблемы или вопросы.

Загрузка модулей

Можно загрузить любой модуль (от имени суперпользователя), который был скомпилирован и установлен в работающее ядро (в подкаталог `/lib/modules`), с помощью команды `modprobe`. Распространенная причина загрузки модуля — временное использование этой функции, например загрузка модуля для поддержки специальной файловой системы на некоторых съемных носителях, к которым нужно получить доступ. Другой причиной загрузки модуля является идентификация его как модуля, который будет использоваться определенным оборудованием, не обнаруживаемым автоматически.

Пример с командой `modprobe`, задействуемой для загрузки модуля `parport`, который отвечает за основные функции для совместного применения параллельных портов с несколькими устройствами:

```
# modprobe parport
```

После загрузки модуля `parport` можно загрузить модуль `parport_pc`, чтобы определить порты в стиле ПК, доступные через интерфейс. Модуль `parport_pc` позволяет определить также адреса и номера IRQ, связанные с каждым устройством, совместно использующим параллельный порт, как в следующем примере:

```
# modprobe parport_pc io=0x3bc irq=auto
```

Здесь устройство идентифицируется с адресом `0x3bc` и IRQ для него определяется автоматически.

Команда `modprobe` загружает модули временно — они исчезают при следующей загрузке системы. Чтобы навсегда загрузить модуль в систему, добавьте командную строку `modprobe` в один из сценариев запуска, выполняемых во время загрузки.

Удаление модулей

Применяйте команду `rmmod` для удаления модуля из активного ядра. Например, чтобы удалить модуль `parport_pc` из текущего ядра, введите следующее:

```
# rmmod parport_pc
```

Если модуль `parport_pc` в данный момент не используется, то он будет удален из ядра. Если же он занят, попробуйте завершить любой процесс, который устройство может задействовать. Затем снова запустите команду `rmmod`. Иногда модуль, который нужно удалить, зависит от других загруженных модулей. Например, модуль `usbcore` нельзя удалить, так как он встроенный:

```
# rmmod usbcore
rmmod: ERROR: Module usbcore is builtin.
```

Вместо команды `rmmod` используйте для удаления команду `modprobe -r`. С ее помощью вместо простого удаления запрашиваемого модуля можно удалить зависимые модули, не задействованные другими модулями.

Резюме

Многие функции Linux, особенно те, которые потенциально могут навредить системе или повлиять на других пользователей, требуют владения правами суперпользователя. В этой главе описаны различные способы получения прав суперпользователя: прямой вход в систему, команды `su` и `sudo`. Глава также охватывает некоторые ключевые обязанности системного администратора и компоненты системы (файлы конфигурации, браузерные инструменты и т. д.), имеющие решающее значение для работы системного администратора.

В следующей главе описывается, как реализовать полноценную установку системы Linux как с «живого», так и с установочного носителя.

Упражнения

Выполните данные упражнения, чтобы проверить свои знания в области системного администрирования и изучить информацию о системном оборудовании. Задания предназначены для дистрибутивов Fedora или Red Hat Enterprise Linux (некоторые можно выполнить и в других системах Linux). Если затрудняетесь с решением за-

даний, воспользуйтесь ответами к упражнениям, приведенными в приложении Б (хотя Linux позволяет решать задачи разными способами).

1. Из оболочки от имени суперпользователя (или с помощью `sudo`) включите Cockpit (`cockpit.socket`) с помощью команды `systemctl`.
2. Откройте браузер для интерфейса Cockpit (9090) в своей системе.
3. Найдите в каталоге `/var/spool` все файлы, принадлежащие пользователям (кроме суперпользователя), и отобразите их длинным списком.
4. Станьте суперпользователем с помощью команды `su`. Чтобы подтвердить, что у вас есть права, создайте пустой или обычный текстовый файл `/mnt/test.txt`. Когда закончите, выйдите из оболочки. Если вы используете дистрибутив Ubuntu, сначала установите пароль суперпользователя (`sudo passwd root`).
5. Войдите в систему как обычный пользователь и станьте суперпользователем с помощью команды `su`. Отредактируйте файл `/etc/sudo`, чтобы ваша обычная учетная запись пользователя получила полные права суперпользователя с помощью команды `sudo`.
6. Как пользователь, которому даны права `sudoer`, используйте команду `sudo` для создания файла `/mnt/test2.txt`. Убедитесь, что файл существует и принадлежит суперпользователю.
7. Выполните команду `journalctl -f` и подключите USB-накопитель к USB-порту компьютера. Если он не монтируется автоматически, смонтируйте его в каталоге `/mnt/test`. Во втором терминале размонтируйте устройство и извлеките его, продолжая наблюдать за выводом данных из `journalctl -f`.
8. Выполните ту команду, которая позволит узнать, какие USB-устройства подключены к компьютеру.
9. Представьте, что вы добавили на компьютер ТВ-тюнер, но модуль, необходимый для его использования (`bttv`), не был обнаружен и загружен правильно. Самостоятельно загрузите модуль `bttv`, а затем проверьте, был ли он загружен. Загрузились ли вместе с ним другие модули?
10. Удалите модуль `bttv` и любые другие модули, которые загрузились вместе с ним. Перечислите все свои модули, чтобы убедиться, что `bttv` удален.

9 Установка Linux

В этой главе

- Выбор метода установки.
- Установка одной или нескольких систем на компьютер.
- Установка системы Fedora с «живого» носителя.
- Установка Red Hat Enterprise Linux.
- Облачные установки.
- Разбиение диска на разделы для установки.
- Загрузчик операционной системы GRUB.

Полноценная установка Linux стала довольно простой, особенно если обзавестись компьютером с соответствующими характеристиками жесткого диска, оперативной памяти, процессора и т. д. Она подразумевает, что необходимо отформатировать жесткий диск.

С облачными вычислениями и виртуализацией установка может быть еще проще. Они позволяют обойти традиционную установку и загрузить систему Linux в течение нескольких минут, добавив метаданные к предварительно созданным образам.

Сначала мы рассмотрим простую установку на физический компьютер с «живого» носителя, затем перейдем к более сложным вариантам.

Чтобы облегчить восприятие темы установки Linux, я расскажу о трех различных способах и пошагово опишу каждый.

- **Установка с «живого» носителя.** Linux Live ISO-образ — это единый доступный только для чтения образ, содержащий все необходимое для запуска операционной системы Linux. Этот образ можно записать на DVD или USB-накопитель и загрузить с данного носителя. Работая с «живым» диском, можно

не принимать в расчет жесткий диск компьютера — фактически систему можно запустить в уже установленной системе, не используя жесткий диск. После загрузки с «живого» носителя Linux некоторые операционные системы позволяют запустить инсталлятор, который на постоянной основе устанавливает содержимое «живого» носителя на ваш жесткий диск. Описывая первый способ установки в этой главе, расскажем, как установить полноценную систему Linux из ISO-файла операционной системы Fedora.

- **Установка с установочного DVD.** Установочный DVD (для Fedora, RHEL, Ubuntu и других дистрибутивов) обеспечивает более гибкие способы установки Linux. В частности, вместо того, чтобы просто копировать все содержимое «живого» носителя на компьютер, с помощью установочного DVD можно выбрать именно тот пакет программного обеспечения, который вам нужен. Второй способ установки, рассмотренный в этой главе, позволяет установить дистрибутив Red Hat Enterprise Linux 8 с установочного DVD.
- **Установка в корпоративной среде.** При установке одной системы нет никаких проблем — сидишь и отвечаешь на вопросы инсталлятора. Но что, если нужно установить сразу 10 или 100 систем Linux? Что делать, если требуется установить несколько систем определенным образом? Далее в этой главе я опишу эффективные способы установки нескольких систем Linux с помощью функций сетевой установки и файлов `kickstart`.

Четвертый способ установки не описывается в этой главе, но позволяет установить Linux в облако (например, Amazon Web Services) или виртуальную машину на узле виртуализации, например в Virtual Box или в системе VMware. В главах 27 и 28 описываются способы установки или развертывания виртуальной машины на узле Linux KVM или в облаке.

Чтобы вместе со мной опробовать все варианты установки, описанные в этой главе, вы должны иметь отдельный компьютер, данные с которого можно полностью стереть. В качестве альтернативы можно использовать компьютер, на котором установлена другая операционная система (например, Windows), при условии, что на жестком диске достаточно незадействуемого дискового пространства. Этот способ и риск потери данных тоже будут описаны на случай, если вы решите выполнить двойную загрузку (Linux и Windows).

Выбор компьютера

Дистрибутив Linux можно установить даже на портативное устройство или старый ПК с всего лишь 24 Мбайт оперативной памяти и процессором 486. Но, чтобы получить реальный опыт работы с настольным Linux, нужно выбрать подходящий по характеристикам компьютер.

Следует учитывать основные технические характеристики, которые нужны компьютеру, чтобы запустить на нем дистрибутивы Fedora и Red Hat Enterprise Linux.

Поскольку Fedora является частью Red Hat Enterprise Linux, требования к оборудованию у них аналогичны для базовых настольного и серверного оборудования.

- **Процессор.** Процессор Pentium с частотой 1 ГГц — это необходимый минимум для установки графического интерфейса. Для большинства приложений подойдет 32-разрядный процессор (x86). Однако, если нужно установить виртуализацию, понадобится 64-разрядный процессор (x86_64).

ПРИМЕЧАНИЕ

Если мощность вашего компьютер меньше необходимого минимума, присмотритесь к легковесным дистрибутивам Linux. Легкие дистрибутивы для Ubuntu — это Peppermint OS (peppermintos.com) и Lubuntu (lubuntu.net). Легкий дистрибутив для Fedora — это LXDE (spins.fedoraproject.org/lxde). Дистрибутив Linux, требующий наименьшего количества ресурсов, — это Tiny Core Linux (tinycorelinux.net).

- **Оперативная память (ОЗУ).** Для установки Fedora рекомендуется не менее 1 Гбайт оперативной памяти, но лучше было бы 2 или 3 Гбайт. На своем рабочем столе RHEL я использую только браузер, текстовый процессор и менеджер почты — и это занимает более 2 Гбайт оперативной памяти.
- **DVD или USB-накопитель.** Необходимо иметь возможность запустить процесс установки с DVD или USB-накопителя. Последние версии образов Fedora live media ISO слишком большие, чтобы поместиться на CD, поэтому нужно записывать их на DVD или USB-накопитель. Если нет возможности загрузиться с DVD или USB-накопителя, есть другие способы запустить установку с жесткого диска или с помощью среды PXE. Иногда после того, как установка запущена, из разных мест можно извлечь дополнительное программное обеспечение (например, по сети или с жесткого диска).

ПРИМЕЧАНИЕ

Среда PXE (произносится «пикси») расшифровывается как Preboot eXecution Environment. Вы можете загрузить клиентскую версию системы с карты сетевого интерфейса (NIC), в которую входит среда PXE. Если загрузочный сервер PXE доступен в сети, он предоставляет все необходимое для загрузки клиента. То, что он загружает, может являться инсталлятором. Таким образом, с помощью среды PXE можно выполнить полную установку Linux без CD, DVD или любого другого физического носителя.

- **Сетевая карта.** Чтобы иметь возможность добавлять или обновлять программное обеспечение, необходимо проводное или беспроводное сетевое оборудование. Дистрибутив Fedora предоставляет бесплатные репозитории программного обеспечения при подключении к Интернету. Для дистрибутива RHEL обновления доступны как часть платной подписки.
- **Пространство на диске.** Для Fedora рекомендуется иметь не менее 20 Гбайт дискового пространства, хотя в зависимости от приобретенных пакетов требования к установке могут варьироваться от 600 Мбайт (для минимального сервера без установки графического интерфейса) до 7 Гбайт (для установки

всех пакетов с установочного DVD). Принимайте в расчет размер сохраненных данных. Документы обычно занимают мало места, а видеофайлам требуется много памяти. (Для сравнения: дистрибутив Tiny Core Linux можно установить на диск с 16 Мбайт дискового пространства, и это при наличии графического интерфейса.)

- **Специальное оборудование.** Некоторые функции Linux требуют специального оборудования. Например, чтобы применять Fedora или RHEL в качестве узла виртуализации с помощью KVM, компьютер должен иметь процессор с поддержкой технологий виртуализации. К ним относятся чипы AMD-V или Intel-VT.

Если вам неизвестны характеристики компьютера, есть несколько способов узнать о них. Если вы используете систему Windows, то в окне System Properties (Свойства системы) найдете информацию о процессоре и объеме установленной оперативной памяти. В качестве альтернативы, загрузив образ Fedora Live CD, откройте оболочку и введите `dmesg | less`, чтобы увидеть весь список оборудования системы.

С помощью подходящего оборудования выберите способ установки Linux с «живого» или установочного носителя, как описано в следующих разделах.

Установка Fedora с «живого» носителя

В главе 2 описано, как получить и загрузить «живую» систему Linux. В этой главе мы рассмотрим полноценную установку операционной системы Fedora на жесткий диск.

Главное преимущество установки с «живого» носителя — это простота процесса. По сути, вы просто копируете ядро, приложения и настройки из образа ISO на жесткий диск. В ходе установки пользователь принимает лишь несколько решений, причем не выбирая конкретные пакеты программного обеспечения. Уже после установки можно добавлять и удалять пакеты по своему усмотрению.

Однако потребуется решить, где вы хотите установить систему и хотите ли сохранить существующие операционные системы после завершения установки.

- **Односистемный компьютер.** Самый простой способ установить Linux — это установить ее в качестве единственной операционной системы. По окончании будет загружена система Fedora.
- **Многосистемный компьютер.** Если у вас уже есть система Windows и вы не хотите ее стирать, можно установить Fedora на одном компьютере с Windows. Правда, во время загрузки компьютера каждый раз придется выбирать, какую операционную систему запустить. Чтобы установить Fedora в системе с другой операционной системой, необходимо иметь либо дисковое пространство за пределами Windows, либо возможность уменьшить его затраты на систему Windows, чтобы освободить место для Fedora. Поскольку мультизагрузочные

компьютеры утомительно настраивать и всегда есть риск повредить уже установленную систему, я все же советую устанавливать Linux на отдельный компьютер, даже старый, или на виртуальную машину.

- **Коммутатор без ОС или виртуальная система.** Система Fedora может быть установлена для загрузки непосредственно с компьютерного оборудования или из существующей на компьютере операционной системы. Если у вас есть виртуальный хост, можете установить Fedora в этой системе в качестве виртуального гостя. Программное обеспечение хоста виртуализации включает KVM, Xen и VirtualBox (для систем Linux, UNIX, Windows и macOS), Hyper-V (для систем Microsoft) и VMware (для Linux, Windows и macOS). Вы можете использовать ISO-образ Fedora или записать его на физический DVD, чтобы начать установку в выбранном хосте виртуализации. (В главе 27 описывается настройка хоста виртуализации KVM.)

Далее приведен план установки операционной системы Fedora, о которой мы говорили в главе 2, на ваш компьютер. Поскольку установка Fedora 30 очень похожа на установку Red Hat Enterprise Linux 8, описанную далее в этой главе, придерживайтесь этого же порядка действий, если хотите расширить показанный здесь выбор настроек (особенно по части настройки хранилища).

ВНИМАНИЕ!

Перед началом установки обязательно сделайте резервные копии всех данных на компьютере, которые нужны сохранить. При установке можно не стирать выбранные разделы диска, если в других разделах достаточно свободного места, однако всегда существует риск потери данных при работе с разделами диска. Кроме того, отключите все подключенные к компьютеру USB-накопители, потому что они могут быть перезаписаны.

1. Скачайте дистрибутив Fedora. Выберите образ Fedora Live, который хотите использовать, загрузите его в локальную систему и запишите на соответствующий носитель. В приложении А подробно описано, как получить образ Fedora и записать его на DVD или USB-накопитель.
2. Загрузите образ. Вставьте DVD или USB-накопитель. Когда появится экран BIOS, найдите сообщение с указанием определенной функциональной клавиши (например, F12) и нажмите ее, чтобы прервать процесс загрузки и выбрать загрузочный носитель. Выберите DVD или USB-накопитель, после чего система Fedora должна отобразиться на загрузочном экране. Когда увидите загрузочный экран, выберите Start Fedora — Workstation — Live.
3. Начните установку. Когда появится экран Welcome to Fedora (Добро пожаловать в Fedora), выберите вариант Install to Hard Drive (Установить на жесткий диск). На рис. 9.1 показан экран выбора установки.
4. Выберите язык. При появлении соответствующего окна укажите язык системы, например Russian (Russia) (Русский (Россия)), и нажмите кнопку Next (Продолжить). Появится экран обзора установки (рис. 9.2).

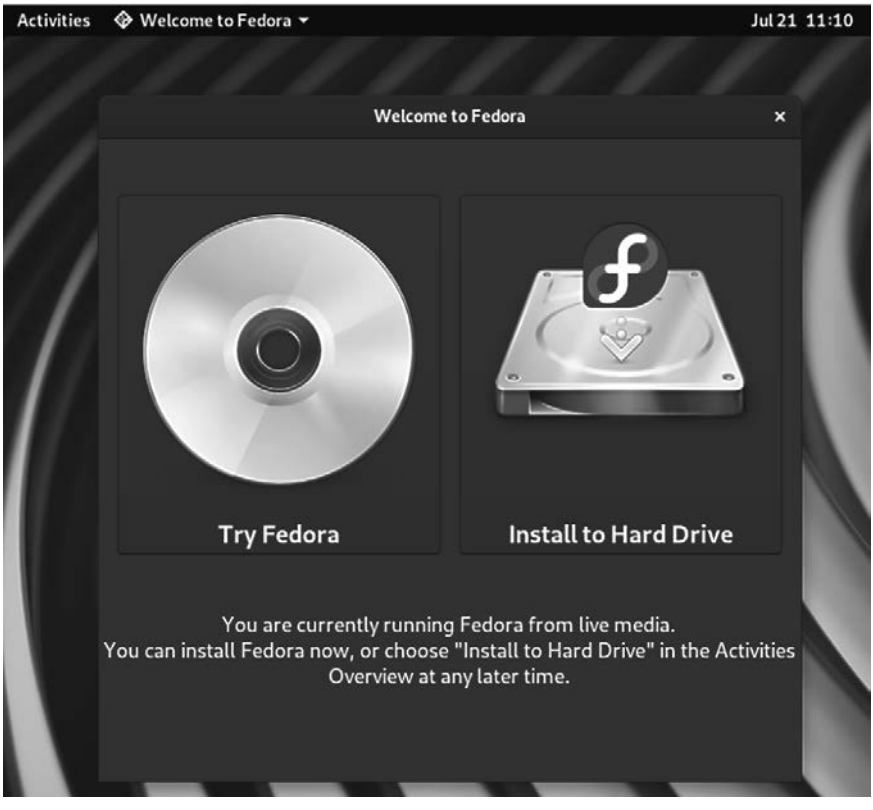


Рис. 9.1. Запустите процесс установки с «живого» носителя

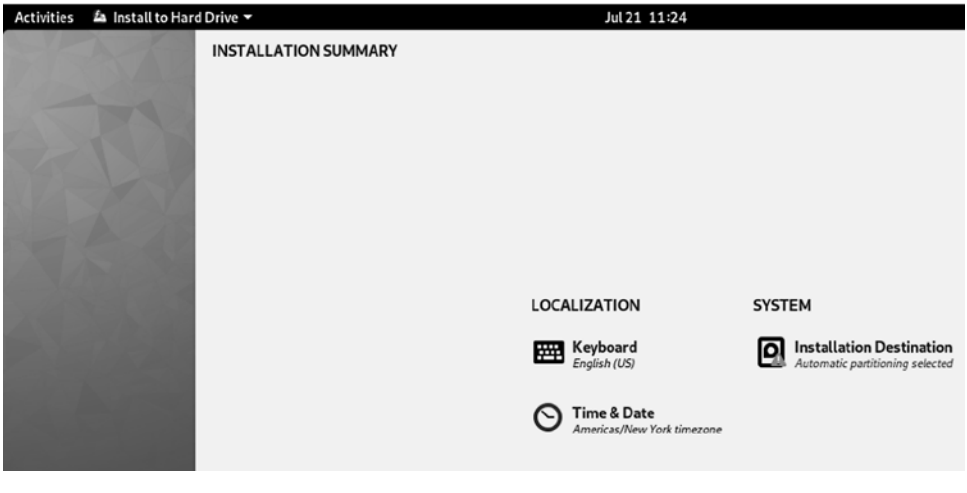


Рис. 9.2. Выберите параметры конфигурации на экране обзора установки

5. Установите время и дату. На экране Time & Date (Дата и время) вы можете задать свой часовой пояс, щелкнув на карте или выбрав регион и город из раскрывающихся списков. Чтобы установить дату и время при подключении к Интернету, сделайте переключатель Network Time (Сетевое время) активным. Или задайте дату и время вручную в нижней части экрана. Нажмите кнопку Done (Готово) в верхней части экрана.
6. Выберите место установки. Отобразятся доступные устройства хранения данных (например, жесткий диск), а в качестве места установки будет выбран жесткий диск. Если нужно, чтобы программа установила Fedora автоматически, освободив пространство на диске, убедитесь, что выбран именно диск, а не USB-накопитель или другое устройство, подключенное к компьютеру, и выполните следующие действия:
   - Automatic (Готово). Если на выбранном диске достаточно свободного места, можете продолжить установку, выбрав кнопку Continue (Продолжить). В противном случае необходимо освободить место на диске, установив флажок I would like to make additional space available (Выделить дополнительное пространство). Если хотите полностью стереть жесткий диск, выберите этот вариант и нажмите кнопку Continue (Продолжить). Вы сможете очистить все или некоторые разделы с данными;
   - Reclaim Disk Space (Восстановить пространство на диске). Здесь выберите вариант Delete All (Удалить все). Затем нажмите кнопку Reclaim Space (Восстановить пространство). Разбиение дисков будет настроено автоматически, и программа вернется на экран Installation Summary (Обзор установки).
7. Выберите клавиатуру. Можно оставить английскую (English (U.S.)) по умолчанию или перейти в раздел Keyboard (Клавиатура), чтобы выбрать другую раскладку.
8. Начните установку. Нажмите кнопку Begin Installation (Начать установку), чтобы установить систему на жесткий диск.
9. Завершите установку. Когда первая часть процесса будет завершена, нажмите кнопку Quit (Выход).
10. Перезагрузите систему. Нажмите значок включения/выключения в меню в правом верхнем углу экрана. В появившемся окне выберите пункт Restart (Перезапустить). При появлении экрана загрузки системы извлеките или удалите «живой» носитель. Компьютер загрузит только что установленную систему Fedora. (Может потребоваться выключить компьютер, чтобы он снова загрузился.)
11. Начните использовать Fedora. Появится первый загрузочный экран, позволяющий, помимо прочего, создать учетную запись пользователя и установить к ней пароль. После завершения установки вы автоматически войдете в систему с этой учетной записью. Она предполагает права `sudo`, поэтому можете сразу приступать к решению административных задач.

12. Обновите программное обеспечение. Чтобы поддерживать безопасность и актуальность системы, первое, что нужно сделать после установки Fedora, — обновить только что установленное программное обеспечение. Если компьютер подключен к Интернету (проводная сеть Ethernet или беспроводная), откройте Terminal (Терминал) как новый пользователь и введите команду `sudo dnf update`, чтобы загрузить программное обеспечение из Интернета и обновить все пакеты. Если будет установлено новое ядро, перезагрузите систему, чтобы изменения вступили в силу.

На данном этапе можно начать использовать рабочий стол, как описано в главе 2. Эту систему можно применять для выполнения упражнений из любой главы книги.

Установка Red Hat Enterprise Linux с установочного носителя

Помимо установки с «живого» DVD-носителя, большинство дистрибутивов Linux предлагают один образ или набор образов, с помощью которых можно установить дистрибутив. В этом случае вместо копирования всего содержимого носителя на диск программное обеспечение разбивается на пакеты, которые можно выбрать вручную в зависимости от предпочтений. Например, полный установочный DVD позволяет установить что угодно, от минимальной системы до полнофункционального рабочего стола и полноценного сервера с несколькими службами.

В этой главе мы рассмотрим установку Red Hat Enterprise Linux 8 с установочного DVD. Перед началом установки RHEL ознакомьтесь с информацией о необходимом оборудовании и описанием двойной загрузки из предыдущего раздела.

Далее представлен план действий по загрузке Red Hat Enterprise Linux 8 с установочного DVD.

1. Создайте установочный носитель. Процесс загрузки ISO-образов RHEL описан на странице Red Hat Enterprise Linux. Если вы не являетесь клиентом Red Hat, подайте заявку на получение ознакомительной демоверсии здесь: redhat.com/en/technologies/linux-platforms/enterprise-linux. Для этого необходимо создать учетную запись Red Hat. Если это невозможно, можете загрузить установочный DVD с зеркального сайта проекта CentOS: wiki.centos.org/Download.

   Для примера я использовал образ `rhel-8.3-x86_64-dvd`. ISO-образ нужно записать на физический USB-накопитель или двухслойный DVD, как описано в приложении А.

2. Загрузите установочный носитель. Вставьте USB-накопитель или DVD в компьютер и перезагрузите его. (При необходимости прервите процесс загрузки, чтобы выбрать загрузку с выбранного USB или DVD.) Появится экран приветствия.

3. Выберите вариант Install или Test Media. Выберите пункт Test this media & install Red Hat Enterprise Linux, чтобы выполнить установку RHEL. Тестирование носителя проверяет, не был ли поврежден DVD после копирования или записи. Если нужно изменить процесс установки, добавьте параметры загрузки, нажав клавишу Tab. (См. раздел «Параметры установки» далее в этой главе.)
4. Выберите язык системы. Укажите подходящий язык и нажмите кнопку Continue (Продолжить). Появится экран Installation Summary (Обзор установки). На нем настраиваются параметры в разделах Localization (Региональные настройки), Software (Программное обеспечение) и System (Система) (рис. 9.3).

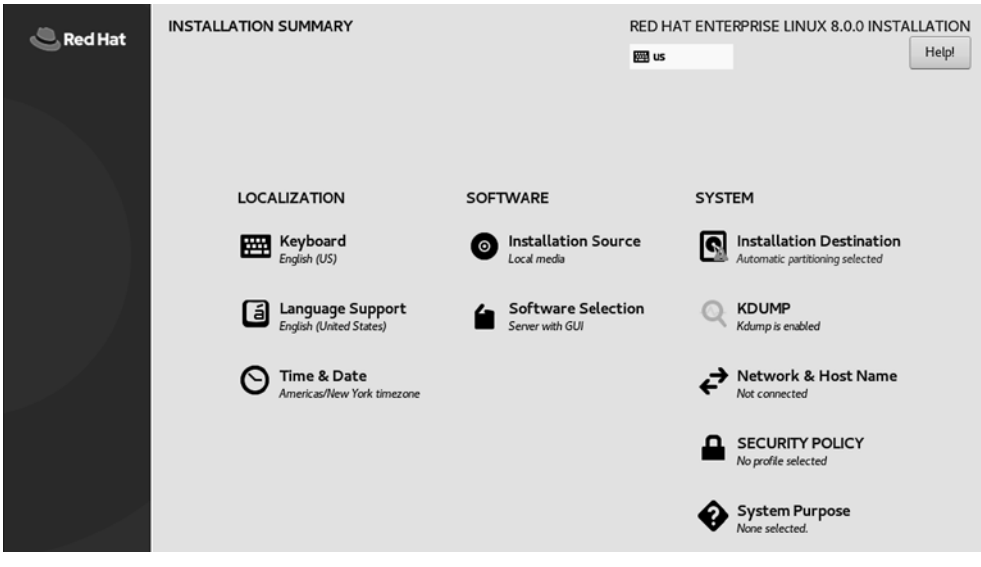


Рис. 9.3. Выберите нужные параметры в разделах Localization (Региональные настройки), Software (Программное обеспечение) и System (Система) на экране Installation Summary (Обзор установки)

5. Раздел Keyboard (Клавиатура). Выберите раскладку клавиатуры, подходящую для указанного ранее языка системы. Введите текст, чтобы увидеть, как расположены клавиши.
6. Раздел Language Support (Языковая поддержка). Система позволяет добавить поддержку дополнительных языков, помимо установленных по умолчанию ранее. По окончании нажмите кнопку Done (Готово).
7. Раздел Time & Date (Дата и время). Выберите часовой пояс на карте или в списке, как описано в разделе «Установка Fedora с “живого” носителя». Установите время вручную с помощью стрелок ↑ и ↓ либо выберите вариант Network Time (Сетевое время), чтобы система автоматически синхронизировала время по сети. По окончании нажмите кнопку Done (Готово).
8. Раздел Installation Source (Инсталляционный источник). Установочный DVD с пакетами RPM используется по умолчанию. Здесь можно выбрать вариант

On the network (В сети) с протоколами Web URL (`http`, `https` или `ftp`), который определяет, где хранится программное обеспечение Red Hat Enterprise Linux. После выбора источника — DVD или в сети — можно добавить дополнительные репозитории `yum`, чтобы они тоже применялись во время установки. По окончании нажмите кнопку Done (Готово).

9. Раздел Software Selection (Выбор программ). По умолчанию выбирается Server with GUI (Сервер с GUI), что предоставляет GNOME 3, помимо базовой установки сервера. Есть также варианты Server (Сервер) без GUI, Minimal Install (Минимальная установка), которая начинается с базового пакета установок, и Workstation (Рабочая станция), подходящий для конечных пользователей. Можно добавить другие службы или базовые среды. Нажмите кнопку Done (Готово), чтобы продолжить.
10. Раздел Installation Destination (Место установки). Новая система RHEL устанавливается по умолчанию на локальный жесткий диск с помощью автоматического разбиения на разделы. Можно также подключить сетевое хранилище или специальное, например, Firmware RAID. (См. раздел «Разбиение жестких дисков» далее в этой главе, чтобы получить подробную информацию о настройке хранилища.) По окончании нажмите Done (Готово). На этом этапе программа может спросить, можно ли очистить существующее хранилище.
11. Раздел Kdump. Подключение функции `kdump` позволяет выделить оперативную память, которая будет использоваться для создания аварийных дампов при сбоях ядра. Без `kdump` невозможно осуществить диагностику ломаного ядра. По умолчанию `kdump` выделяет 160 Мбайт плюс 2 бита на каждые 4 Кбайт оперативной памяти для сохранения информации о сбоях ядра.
12. Раздел Network & Host Name (Сеть и имя узла). На этом этапе настраиваются любые обнаруженные карты сетевого интерфейса. Если в сети служба DHCP доступна, информация о сетевом адресе назначается интерфейсу после его включения. Нажмите кнопку Configure (Настроить), чтобы вручную изменить настройки сетевого интерфейса. Заполните поле Hostname (Имя узла), чтобы задать имя узла системы. Настройка сети и имени узла во время установки облегчает начало использования системы. Нажмите Done (Готово), чтобы продолжить.
13. Раздел Security Policy (Политика безопасности). Выберите профиль безопасности (по умолчанию не выбран), чтобы ваша система соответствовала стандарту безопасности. Все настройки являются необязательными и могут быть изменены позже.
14. System Purpose (Предназначение системы). Это необязательный пункт, он позволяет выбрать цель системы, соглашение об обслуживаниии и использовании.
15. Раздел Root Password (Пароль root). Установите пароль для администратора (суперпользователь, пользователь root) и подтвердите его (введите еще раз). Нажмите кнопку Done (Готово). Если пароль слишком короткий или слишком простой, понадобится придумать новый. Если же вы все-таки решили сохранить слабый пароль, нажмите кнопку Done (Готово) еще раз.

16. Раздел User Creation (Создание пользователя). Рекомендуется войти в систему Linux с готовой учетной записью пользователя, не являющегося суперпользователем, и запрашивать права по мере необходимости. Вы можете настроить учетную запись пользователя, включая имя, полное имя и пароль. Выберите пункт Make this user administrator (Сделать этого пользователя администратором), чтобы дать права sudo (учетная запись сможет использовать права по необходимости). По окончании нажмите кнопку Done (Готово). Если пароль слишком короткий или слишком простой, придумайте новый. Если же вы хотите сохранить слабый пароль, нажмите кнопку Done (Готово) еще раз.
17. Начало установки. Нажмите кнопку Begin Installation (Начать установку), чтобы процесс стартовал. Появится индикатор хода установки. Во время этого можно назначить пароль root и создать новую учетную запись пользователя.
18. Окончание установки. Когда установка будет завершена, нажмите кнопку Reboot (Перезагрузка). Когда система перезагрузится, извлеките DVD. Red Hat Enterprise Linux запустится с жесткого диска.
19. Откройте окно начальной настройки. Если установлен интерфейс рабочего стола, то при первой загрузке системы появится экран начальной настройки. Он включает в себя разделы:
    - License Information (Лицензионная информация). Прочтите информацию о лицензии и примите лицензионное соглашение, затем нажмите кнопку Done (Готово);
    - Subscription Manager (Менеджер подписок). При появлении соответствующего запроса оставьте менеджер подписок настроенным по умолчанию (subscription.rhn.redhat.com) или введите адрес сервера Red Hat Satellite, чтобы зарегистрировать систему. Нажмите кнопку Next (Далее). Введите свою учетную запись Red Hat и пароль, а затем нажмите кнопку Register (Зарегистрироваться), чтобы зарегистрироваться и получить право на обновление системы. Если подписка найдена и доступна, нажмите кнопку Attach (Прикрепить), чтобы подключить подписку.
20. В конце выберите пункт Finish Configuration (Завершить установку).

Теперь вы можете войти в систему Red Hat Enterprise Linux. И снова самое важное для начала — обновить программное обеспечение. Войдите в систему и запустите команду `sudo dnf upgrade` из окна Terminal (Терминал).

Облачные установки

При установке системы Linux на физический компьютер программа установки может видеть его жесткий диск, сетевые интерфейсы, процессоры и другие компоненты. При установке Linux в облачную среду эти физические компоненты абстрагируются в пул ресурсов. Итак, чтобы установить дистрибутив Linux

в Amazon EC2, Google Compute Engine или облачную платформу OpenStack, нужно действовать иначе.

Распространенный способ установить Linux в облаке — это начать с ISO-образа установленной системы. Как правило, он включает в себя все файлы, необходимые для базовой системы Linux. Метаданные добавляются к этому образу из файла конфигурации или путем заполнения формы из облачного контроллера, который создает и запускает операционную систему как виртуальную машину.

Информация, добавляемая к образу, может включать конкретное имя узла, пароль суперпользователя и новую учетную запись пользователя. Можно также выбрать определенные объем пространства на диске, сетевую конфигурацию и количество процессоров и оперативной памяти.

Методы установки Linux в локальную облачную среду KVM описаны в главе 28. Там же рассказывается, как запустить систему Linux в виде образа виртуальной машины в средах KVM, Amazon EC2 и OpenStack.

Установка Linux в корпоративной среде

Если бы вы управляли десятками, сотнями и даже тысячами систем Linux на большом предприятии, было бы неэффективно идти к каждому компьютеру и отдельно устанавливать на нем систему. К счастью, работая с Red Hat Enterprise Linux и другими дистрибутивами, можно автоматизировать установку таким образом, что потребуется лишь включить компьютер и загрузить систему из сети.

Хотя мы основательно разобрали ход установки Linux с DVD или USB-носителя, существует множество других способов установки Linux и ее завершения. Далее пошагово рассмотрим процесс установки и методы его изменения.

- **Запуск установочного носителя.** Вы можете запустить установку с любого носителя, который можно загрузить с компьютера, будь то CD, DVD, USB-накопитель, жесткий диск или сетевой адаптер с поддержкой PXE. Компьютер выполнит загрузку по порядку и просмотрит запись о ней на физическом носителе или найдет PXE-сервер в сети.
- **Запуск инсталлятора Anaconda.** Задача загрузчика состоит в том, чтобы указать на специальное ядро (и, возможно, начальный RAM-диск), которое запускает инсталлятор Linux, называемый Anaconda. Таким образом, любой из описанных ранее типов носителей просто должен указать на местоположение ядра и начального RAM-диска, чтобы начать установку. Если пакеты программного обеспечения не находятся на одном носителе, в процессе установки будет задан вопрос, откуда их взять.
- **Файл kickstart и другие параметры загрузки.** Параметры загрузки (описаны далее в этой главе) передаются в ядро Anaconda для настройки запуска. Один из параметров, поддерживаемых для Fedora и RHEL, позволяет передать инсталлятору расположение файла `kickstart`. Этот файл содержит всю информацию,

необходимую для завершения установки: пароль администратора, разбиение дисков, часовой пояс и т. д. После запуска инсталлятор либо запрашивает необходимую информацию, либо использует информацию из файла `kickstart`.

- **Поиск пакетов программного обеспечения.** Пакеты программного обеспечения не обязательно должны находиться на установочном носителе. Это позволяет запускать установку с носителя, содержащего только ядро и начальный диск оперативной памяти (RAM-диск). Из файла `kickstart` или выбранного вручную в инсталляторе нужно определить местоположение репозитория с пакетами программного обеспечения RPM. Это может быть локальный CD (`cdrom`), сайт (`http`), FTP-сервер (`ftp`), NFS (`nfs`), NFS ISO (`nfsiso`) или диск (`hd`).
- **Изменение установки со скриптами kickstart.** Скрипты в файле `kickstart` могут запускать команды, выбранные до или после установки, для дальнейшей настройки системы Linux. Эти команды могут добавлять пользователей, изменять права, создавать файлы и каталоги, захватывать файлы по сети или иным образом настраивать установленную систему.

Хотя тема установки Linux в корпоративной среде выходит за рамки этой книги, мне бы хотелось объяснить технологии для автоматизации процесса установки Linux. Вот список нескольких технологий со ссылками, доступных для использования в Red Hat Enterprise Linux.

- **Сервер установки.** Если настроить сервер установки, то не нужно будет переносить пакеты программного обеспечения RHEL на каждый компьютер. По сути, вы копируете все пакеты программного обеспечения с установочного носителя RHEL на веб-сервер (`http`), FTP-сервер (`ftp`) или NFS-сервер (`nfs`), а затем указываете местоположение этого сервера при загрузке инсталлятора. В руководстве по установке RHEL 8 описано, как настроить локальный или сетевой источник установки: access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/html-single/performing_a_standard_rhel_installation/index#prepareinstallation-source_preparing-for-your-installation.
- **Сервер PXE.** Если у вас есть карта сетевого интерфейса (network interface card, NIC), которая поддерживает загрузку PXE (как и большинство других), вы можете настроить в BIOS загрузку с этой карты. Если настроить PXE-сервер в этой сети, то он представит компьютеру меню с параметрами запуска процесса установки. В руководстве по установке RHEL содержится информация о том, как настроить PXE-серверы: access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/html-single/performing_a_standard_rhel_installation/index#booting-theinstallation-using-pxe_booting-the-installer.
- **Файлы kickstart.** Чтобы полностью автоматизировать установку, создаются так называемые *файлы kickstart*. Инсталлятор Linux собирает всю представленную в файле информацию и автоматизирует установку.

  При установке RHEL файл `kickstart`, содержащий реакцию на все вопросы установки, находится в файле `/root/anaconda-ks.cfg`. Используйте этот файл для следующей установки, чтобы точно повторить конфигурацию, или в ка-

честве модели. См. расширенное руководство по установке RHEL (Advanced RHEL Installation Guide), чтобы получить информацию о настройке файла `kickstart`: access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/html-single/performing_an_advanced_rhel_installation/index/#performing_an_automated_installation_using_kickstart.

А также о том, как создать собственный `kickstart`-файл: access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/html-single/performing_an_advanced_rhel_installation/index/#creating-kickstart-files_installing-rhel-as-an-experienced-user.

Общие параметры установки

Некоторые темы установки, затронутые ранее в этой главе, раскрыты не полностью и требуют дальнейшего объяснения. В следующих разделах общие темы, касающиеся процесса установки, описываются более подробно.

Обновление или установка с нуля

Если на вашем компьютере уже установлена более ранняя версия Linux, то Fedora, Ubuntu и другие дистрибутивы позволяют обновить систему. Red Hat Enterprise Linux позволяет обновиться только с RHEL 7 до RHEL 8.

Обновление позволяет перенести систему Linux из одной основной версии в другую. Между второстепенными версиями можно просто обновлять пакеты по мере необходимости (например, набрав команду `yum update`). Вот несколько действий, которые нужно выполнить перед обновлением.

- **Удалите дополнительные пакеты.** Если у вас есть ненужные пакеты программного обеспечения, удалите их перед обновлением. Обычно обновляются только системные пакеты. Процесс обновления обычно подразумевает большее количество проверок и сопоставлений, чем первоначальная установка, поэтому чем меньше ненужных пакетов, тем быстрее пройдет процесс.
- **Проверьте файлы конфигурации.** В ходе обновления Linux часто остаются копии старых файлов конфигурации. Проверьте, что появились новые и что они работают.

СОВЕТ

Установка Linux с нуля происходит быстрее, чем обновление. К тому же новая система чище обновленной. Поэтому, если вам не нужны данные, содержащиеся в системе (или если у вас есть их резервная копия), я рекомендую заново установить систему. Затем можете восстановить свои данные в только что установленной системе.

Некоторые дистрибутивы Linux, в первую очередь Gentoo, непрерывно обновляют систему. Вместо того чтобы выпускать обновленную версию каждые несколько

месяцев, они постоянно обновляют пакеты, которые нужно устанавливать в системе по мере их появления.

Мультизагрузка

На одном компьютере может быть установлено несколько операционных систем. Для этого нужно иметь несколько разделов на жестком диске и/или несколько жестких дисков, а затем установить операционные системы в разные разделы. Загрузчик компьютера содержит информацию о загрузке для каждой из установленных операционных систем, и можно выбрать, какую запустить в этот раз.

ВНИМАНИЕ!

За последние годы инструменты манипуляции с размерами разделов системы Windows и настройки мультизагрузочных систем улучшились, однако все еще существует риск потери данных в системах с двойной загрузкой Windows/Linux. Различные операционные системы часто по-разному представляют таблицы разделов и основных загрузочных записей, из-за этого компьютер может перестать загружаться (по крайней мере временно) или вовсе утратит данные. Всегда создавайте резервные копии данных, прежде чем пытаться изменить размер файловой системы Windows, чтобы освободить место для Linux.

Если на вашем компьютере уже установлена система Windows, вполне возможно, что она занимает весь жесткий диск. Можно запустить загрузочный Linux, например KNOPPIX или Tiny Core Linux, не трогая жесткий диск. Однако, чтобы полноценно установить Linux, необходимо освободить место на диске за пределами системы Windows. Есть несколько вариантов действий.

- **Добавьте жесткий диск.** Вместо того чтобы возиться с разбиением диска, просто добавьте еще один, только для Linux.
- **Измените размер диска для Windows.** Если в разделе диска, в котором находится система Windows, есть свободное место, можно его сжать и освободить место на диске для Linux. Коммерческие инструменты, такие как Acronis Disk Director (acronis.com/en-us/personal/disk-manager), позволяют изменять размер разделов диска и настраивать менеджер загрузки. Некоторые дистрибутивы Linux (особенно загрузочные, которые используются в качестве спасательных носителей) включают инструмент GParted, содержащий программное обеспечение из проекта Linux-NTFS для изменения размера разделов Windows NTFS.

ПРИМЕЧАНИЕ

Введите команду dnf install gparted (в Fedora) или apt-get install gparted (в Ubuntu), чтобы установить инструмент GParted. Запускайте команду gparted от имени суперпользователя.

Перед изменением размера раздела Windows может потребоваться его дефрагментация. Чтобы дефрагментировать диск в системах Windows, приводя в порядок

используемое пространство, откройте My Computer (Мой компьютер), щелкните правой кнопкой мыши на значке жесткого диска (обычно это диск C:), выберите в контекстном меню пункт Properties (Свойства), перейдите на вкладку Tools (Сервис), а затем нажмите кнопку Defragment Now (Оптимизировать).

Дефрагментация диска может занять много времени. В результате все данные на диске станут смежными, создав много свободного пространства. Иногда для этого нужно выполнить дополнительные действия.

- Если файл подкачки Windows не перемещается во время дефрагментации, его необходимо удалить. Затем после того, как вы снова дефрагментируете диск и измените его размер, нужно восстановить этот файл. Чтобы удалить файл подкачки, откройте Control Panel (Панель управления) и щелкните кнопкой мыши на пункте System (Система). Откройте дополнительные свойства системы и в разделе Performance (Быстродействие) нажмите кнопку Options (Параметры). Перейдите на вкладку Advanced (Дополнительно) и нажмите кнопку в разделе Virtual Memory (Виртуальная память). Чтобы отключить файл подкачки, установите переключатель в положение Disable Virtual Memory (Без файла подкачки).
- Если в разделе DOS есть скрытые файлы, требуется их найти. В некоторых случаях их нельзя удалить. В других же — можно, например, файлы подкачки, созданные программой. Это довольно сложный процесс, потому что некоторые файлы, например системные файлы DOS, удалять нельзя. Для работы со скрытыми файлами можно взять команду `attrib -s -h` из корневого каталога.

После дефрагментации диска используйте коммерческие инструменты, описанные ранее (Acronis Disk Director), для разбиения жесткого диска. Или же задействуйте альтернативный вариант GParted с открытым исходным кодом.

Освободив на диске достаточно места для установки Linux (см. требования к пространству диска, приведенные ранее в этой главе), можете установить Ubuntu, Fedora, RHEL или другой дистрибутив Linux. При настройке загрузчика системы можно выбрать, какая система — Windows, Linux и любой другой загрузочный вариант — будет загружаться при запуске компьютера.

Установка и запуск Linux в виртуальной среде

Используя технологии виртуализации, такие как KVM, VMware, VirtualBox или Xen, можно настроить компьютер для одновременного запуска нескольких операционных систем. В таком случае имеется главная операционная система-хост (например, рабочий стол Linux или Windows), а затем настраиваются гостевые операционные системы для работы в этой среде.

В системе Windows можно применять коммерческие продукты VMware для запуска Linux на рабочем столе Windows. Пробная версия VMware Workstation доступна на сайте vmware.com/try-vmware. Бесплатный проигрыватель VMware Player

позволяет запускать образы ранее установленных виртуальных гостевых систем. Полная версия VMware Workstation позволяет запускать несколько дистрибутивов одновременно.

Виртуализация с открытым исходным кодом, доступная в системах Linux, включает в себя VirtualBox (virtualbox.org), Xen (xenproject.org), а также KVM (linux-kvm.org). Некоторые дистрибутивы до сих пор используют Xen. Однако все системы Red Hat в настоящее время применяют KVM в качестве гипервизора Red Hat в RHEL, Red Hat Virtualization и других облачных проектах. В главе 28 описана установка Linux в качестве виртуальной машины на узле Linux KVM.

Параметры установки

Когда ядро Anaconda запускается во время загрузки RHEL или Fedora, параметры самой загрузки из командной строки ядра изменяют поведение процесса установки. Прерывая работу перед загрузкой ядра установки, можно добавить собственные параметры загрузки и управлять установкой.

Когда вы увидите экран установки, в зависимости от загрузчика нажмите клавишу Tab или какую-либо другую, чтобы отредактировать командную строку ядра Anaconda. Строка с ядром может выглядеть примерно так:

```
vmlinuz initrd=initrd.img ...
```

`Vmlinuz` — это сжатое ядро, а `initrd.img` — начальный RAM-диск, содержащий модули и другие инструменты, необходимые для запуска инсталлятора. Чтобы добавить параметры, просто введите их в конце этой строки и нажмите клавишу Enter.

Например, если у вас есть файл `kickstart` в каталоге `/root/ks.cfg` на CD, приглашение загрузки Anaconda может выглядеть следующим образом:

```
vmlinuz initrd=initrd.img ks=cdrom:/root/ks.cfg
```

Для Red Hat Enterprise Linux 8 и последних выпусков Fedora параметры загрузки ядра, используемые во время установки, именуются по-новому. В соответствии с новым способом префикс `inst` помещается перед любым из параметров загрузки, специфичным для процесса установки (например, `nst.xdriver` или `inst.repo=dvd`). Пока все еще можете применять параметры, показанные в следующих разделах, с префиксом `inst`.

Параметры отключения функций

Иногда установка Linux завершается неудачно из-за того, что компьютер имеет какое-то нефункционирующее или неподдерживаемое оборудование. Чтобы обойти это, нужно передать инсталлятору параметры, которые отключат выбранное оборудование при выборе собственного драйвера. В табл. 9.1 показаны примеры.

Таблица 9.1. Параметры отключения функций

Параметр	Действие
nofirewire	Не загружать поддержку устройств firewire
nodma	Не загружать поддержку DMA для жестких дисков
noide	Не загружать поддержку IDE-устройств
nompath	Не включать поддержку многопутевых устройств
noparport	Не загружать поддержку параллельных портов
nopcmcia	Не загружать поддержку контроллеров PCMCIA
noprobe	Не выполнять пробу оборудования, вместо этого запрашивать у пользователя драйверы
noscsi	Не загружать поддержку устройств SCSI
nousb	Не загружать поддержку USB-устройств
noipv6	Не включать сеть IPV6
nonet	Не выполнять пробу сетевых устройств
numa-off	Отключить неравномерный доступ к памяти (NUMA) для архитектуры AMD64
acpi=off	Отключить управление конфигурацией и питанием (ACPI)

Параметры загрузки при проблемах с видеофайлами

Если у вас возникли проблемы с воспроизведением видео, можете добавить нужные настройки, как показано в табл. 9.2.

Таблица 9.2. Параметры загрузки для проблем с видео

Параметр загрузки	Действие
xdriver=vesa	Использовать стандартный видеодрайвер vesa
resolution=1024x768	Установить точное разрешение
nofb	Не применять драйвер фрейм-буфера VGA 16
skipddc	Не выполнять пробу DDC-монитора (проба прекращает работу инсталлятора)
graphical	Принудительная установка с графическим интерфейсом

Параметры загрузки специальных типов установки

По умолчанию установка выполняется в графическом режиме, когда пользователь отвечает на вопросы. Если у вас есть текстовая консоль или графический интерфейс не работает, можно запустить установку в обычном текстовом режиме, набрав команду `text`.

Если нужно начать установку на одном компьютере, но ответить на вопросы установки с другого, включите систему удаленного доступа VNC (virtual network computing). После ее запуска можно перейти в другую систему и запустить `vnc viewer`, указав ему адрес целевой машины (например, 192.168.0.99:1). В табл. 9.3 приведены необходимые команды и действия.

Таблица 9.3. Параметры загрузки для установки VNC

Параметр загрузки	Действие
vnc	Запустить установку от имени VNC-сервера
vncconnect=имя_хоста[:порт]	Подключить к VNC имя узла и дополнительный порт
vncpassword=пароль	Использовать пароль клиента (не менее восьми символов) для подключения к инсталлятору

Параметры загрузки для файлов kickstart и удаленных репозиториев

С установочного носителя можно загрузить процессы, которые могут содержать не только ядро и начальный RAM-диск. В таком случае необходимо определить репозиторий, в котором находятся пакеты программного обеспечения. Для этого предоставьте готовый файл `kickstart` или определите местоположение репозиториев. Чтобы заставить программу установки запрашивать расположение репозитория (CD/DVD, жесткий диск, NFS или URL-адрес), добавьте команду `askmethod` в параметры загрузки установки.

С помощью параметров `repo=` можно определить местоположение репозитория программного обеспечения. В следующих примерах показан синтаксис, используемый для ввода `repo=`:

```
repo=hd:/dev/sda1:/myrepo
Репозиторий в каталоге / myrepo в первом разделе диска
repo=http://abc.example.com/myrepo
Репозиторий из каталога /myrepo на веб-сервере
repo=ftp://ftp.example.com/myrepo
Репозиторий из каталога /myrepo на FTP-сервере
repo=cdrom
Репозиторий на  CD или DVD
repo=nfs::mynfs.example.com:/myrepo/
Репозиторий из каталога /myrepo на NFS-сервере
repo=nfsiso::nfs.example.com:/mydir/rhel7.iso
Установочный ISO-образ, доступный на сервере NFS
```

Вместо того чтобы напрямую вводить местоположение репозитория, можно указать его в файле `kickstart`. Далее приведены примеры способов определения местоположения файла `kickstart`:

```
ks=cdrom:/stuff/ks.cfg
Загрузить файл kickstart с CD- или DVD-носителя
```

```
ks=hd:sda2:/test/ks.cfg
Загрузить файл kickstart из тестового каталога на жестком диске (sda2)
ks=http://www.example.com/ksfiles/ks.cfg
Загрузить файл kickstart с веб-сервера
ks=ftp://ftp.example.com/allks/ks.cfg
Загрузить файл kickstart с FTP-сервера
ks=nfs:mynfs.example.com:/someks/ks.cfg
Загрузить файл kickstart с NFS-сервера
```

Прочие параметры загрузки

Рассмотрим несколько других параметров, которые можно передать инсталлятору.

- **rescue.** Вместо установки запускает ядро, открывающий режим восстановления Linux.
- **mediacheck.** Проверяет установочный CD/DVD на наличие ошибок контрольной суммы.

Дополнительные сведения о применении инсталлятора Anaconda в режиме восстановления (для восстановления поврежденной системы Linux) см. в главе 21. Информация об актуальных параметрах загрузки, используемых в RHEL 8, есть в руководстве по установке RHEL 8: access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/html-single/performing_a_standard_rhel_installation/index#custom-bootoptions_booting-the-installer.

Специальные устройства хранения

Обычно операционная система и данные для больших корпоративных вычислительных сред хранятся вне локального компьютера. Вместо этого в инсталлятор добавляется специальное запоминающее устройство, находящееся за пределами локального жесткого диска и используемое во время установки.

После идентификации устройства, добавленные во время установки, могут применяться так же, как и локальные диски. Их можно разделить и назначить им структуру (файловую систему, пространство подкачки и т. д.) или оставить в покое и просто смонтировать в удобном месте.

При установке Red Hat Enterprise Linux, Fedora или других дистрибутивов Linux на соответствующем экране можно выбрать следующие варианты устройств.

- **Firmware RAID.** Встроенная RAID-сборка — это тип устройства, которое влияет на BIOS, что позволяет использовать его для загрузки операционной системы.
- **Многопутевые устройства.** Как следует из названия, такие устройства обеспечивают несколько путей ввода-вывода между компьютером и его запоминающими устройствами. Эти пути объединены между собой, поэтому все устройства выглядят как одно для применяющей их системы, а базовая технология обеспечивает улучшенную производительность, избыточность или все вместе.

Соединения реализуются устройствами iSCSI или Fibre Channel over Ethernet (FCoE).

- **Другие устройства SAN.** Любое устройство, представляющее собой сеть хранения данных (SAN).

Хотя описание настройки этих устройств не входит в эту книгу, имейте в виду, что на предприятии, где доступны устройства iSCSI и FCoE, можно настроить свою систему Linux и задействовать их во время загрузок. Для этого необходимо знать:

- **об устройствах iSCSI.** Попросите администратора хранилища предоставить вам целевой IP-адрес устройства iSCSI и тип аутентификации, необходимый для использования устройства. Для устройств iSCSI могут потребоваться учетные данные;
- **устройствах Fibre Channel over Ethernet (FCoE).** Чтобы воспользоваться протоколом FCoE, необходимо иметь сетевой интерфейс, подключенный к вашему коммутатору FCoE. В этом интерфейсе находятся доступные устройства FCoE.

Разбиение жестких дисков

Жесткий диск (или диски) на компьютере обеспечивает постоянное место хранения файлов данных, прикладных программ и самой операционной системы. *Разбиение* — это процесс разделения диска на логические области (тома), с которыми можно работать по отдельности. В системе Windows обычно имеется один раздел, занимающий весь жесткий диск. Однако при использовании Linux появляются причины разделить жесткий диск на секции.

- **Множество операционных систем.** Если вы устанавливаете Linux на компьютер, на котором уже имеется операционная система Windows, то разбиение диска позволит сохранить обе. Каждая операционная система должна находиться отдельно от другой. При загрузке компьютера выбирается нужная.
- **Множество разделов внутри операционной системы.** Чтобы защитить всю операционную систему от нехватки места на диске, пользователи назначают отдельные разделы для различных частей файловой системы Linux. Например, если бы `/home` и `/var` были назначены отдельным разделам, то заполненный каталог `/home` не помешал бы демонам записывать данные в файлы журнала в каталоге `/var/log`.

  Разбиение на разделы облегчает также выполнение определенных видов резервного копирования (например, резервное копирование образа). Резервная копия образа `/home` будет сохраняться намного быстрее (и, вероятно, этот вариант предпочтителен), чем резервная копия образа всей корневой файловой системы (`/`).
- **Различные типы файловых систем.** Разные типы файловых систем имеют различную структуру. Файловые системы разных типов должны находиться

отдельно друг от друга. Кроме того, разные файловые системы могут понадобиться, чтобы иметь разные параметры монтирования для специальных функций (например, только для чтения или для дисковых квот). В большинстве систем Linux требуется по крайней мере один тип файловой системы для корня файловой системы (/) и один — для области подкачки. Файловые системы на CD используют тип файловой системы iso9660.

СОВЕТ

При создании разделов для Linux тип файловой системы назначается как Linux native (с помощью типа ext2, ext3, ext4 или xfs в большинстве систем Linux). Если запущенные приложения требуют длинных имен файлов, значительных размеров файлов или большого количества индексных узлов (каждый файл использует один индексный узел), можно выбрать другой тип файловой системы.

Отличие от Windows

Если раньше вы работали только с операционной системой Windows, то, вероятно, весь жесткий диск был назначен как диск C: без каких-либо разделов. Во многих системах Linux есть возможность просматривать и изменять разделы по умолчанию в зависимости от того, как предполагается использовать систему.

Во время установки такие системы, как Fedora и RHEL, позволяют разбивать жесткий диск на разделы с помощью инструментов с графическим интерфейсом. В следующих пунктах описано, как разбить диск на разделы во время установки Fedora. Интересные идеи по разбиению диска приведены в пункте «Советы по созданию разделов» далее.

Типы разделов дисков

Многие дистрибутивы Linux позволяют выбрать различные типы разделов при разбиении жесткого диска во время установки.

- **Разделы Linux.** Используйте этот тип для создания раздела файловой системы типа ext2, ext3 или ext4, который добавляется непосредственно в раздел на жестком диске или другом носителе данных. Тип файловой системы xfs также может применяться в разделе Linux. (Фактически xfs теперь является типом файловой системы по умолчанию для систем RHEL 8.)
- **Разделы LVM.** Создайте раздел LVM, если планируете добавить его в группу томов LVM. Эти разделы более гибки при расширении, сжатии и перемещении, чем другие.
- **Разделы RAID.** Создайте два раздела RAID или более для объединения их в массив RAID. Они должны располагаться на отдельных дисках, чтобы RAID-массив

стал эффективным. RAID-массивы помогают повысить производительность и надежность, поскольку эти функции связаны с чтением, записью и хранением ваших данных.

- **Разделы подкачки (swap).** Создайте раздел подкачки, чтобы увеличить объем доступной в системе виртуальной памяти.

Далее мы рассмотрим, как добавить обычные разделы Linux, а также разделы LVM, RAID и разделы подкачки с помощью инсталлятора Fedora с графическим интерфейсом. Если вы все еще не уверены, когда и какие типы разделов следует использовать, обратитесь к главе 12.

Советы по созданию разделов

Разбиение диска на разделы для работы нескольких операционных систем может показаться сложным отчасти потому, что каждая система имеет собственные представления том, как следует обрабатывать информацию о разделах и с помощью каких инструментов. Вот несколько советов, которые помогут все сделать правильно.

- Если вы создаете систему со сдвоенной загрузкой, особенно с системой Windows, сначала установите ее после того, как разделите диск. В противном случае установка Windows может заблокировать разделы Linux.
- На справочной странице `fdisk` рекомендуется использовать инструменты разбиения, которые входят в пакет с операционной системой, для создания разделов именно в ней. Например, программа `fdisk` в Windows знает, как правильно создавать разделы, подходящие Windows, а `fdisk` в Linux с радостью разделит диск для Linux. После настройки жесткого диска под две системы не применяйте инструменты разделения, предназначенные для Windows. Задействуйте Linux `fdisk` или программу, созданную для мультизагрузочных систем, например Acronis Disk Director.
- В таблице MBR может содержаться четыре основных раздела, каждый из которых способен содержать 184 логических диска. В таблице разделов GPT имеется не более 128 основных разделов в большинстве операционных систем, включая Linux. Обычно столько их и не нужно. А если требуется больше разделов, с помощью функции LVM создайте нужное количество логических томов.

Если Linux работает в качестве настольной системы, в большом количестве различных разделов необходимости нет. Однако существуют веские причины создать несколько разделов в совместно используемых или общедоступных системах Linux. Наличие нескольких разделов в Fedora или RHEL дает следующие преимущества.

- **Защита от атак.** Хакерские атаки типа DoS (Denial of Service, отказ в обслуживании) воздействуют на жесткий диск, чтобы довести систему до отказа. Если общедоступные области системы, например `/var`, находятся в отдельных разделах, атака заполнит весь раздел, но не затронет весь компьютер. Поскольку каталог `/var` — это основной каталог для веб- и FTP-серверов по умолчанию

и он содержит много данных, чаще всего в этот каталог назначаются отдельные жесткие диски целиком.

- **Защита от поврежденных файловых систем.** Если у вас только одна файловая система (/), ее повреждение может привести к нарушению работы всей системы Linux. Повреждение раздела поменьше легче исправить, и это позволяет компьютеру оставаться в рабочем состоянии, пока выполняется исправление.

В табл. 9.4 перечислены каталоги, для которых можно создать отдельные разделы файловой системы.

Таблица 9.4. Назначение разделов определенным каталогам

Каталог	Описание
/boot	Иногда BIOS на старых компьютерах может получить доступ только к первым 1024 цилиндрам жесткого диска. Чтобы убедиться, что информация в каталоге /boot доступна для BIOS, создайте отдельный раздел диска (по умолчанию RHEL 8 устанавливает этот раздел на 1024 Мбайт) для /boot. Даже при наличии нескольких ядер редко бывает так, чтобы размер файла /boot превышал 1024 Мбайт
/usr	Эта структура каталогов содержит большинство приложений и утилит, доступных пользователям Linux. Первоначально идея состояла в следующем: если /usr находится в отдельном разделе, эту файловую систему можно смонтировать только для чтения, что предотвратит замену или удаление важных системных приложений их собственными обновлениями и вероятность проблем с безопасностью. Отдельный раздел /usr полезен и тогда, когда у вас есть бездисковые узлы (рабочие станции) в локальной сети. Используя протокол NFS, вы можете задействовать каталог /usr по сети совместно с этими рабочими станциями
/var	Каталоги FTP (/var/ftp) и веб-сервера (/var/www) по умолчанию во многих системах Linux хранятся в каталоге /var. Отдельный раздел для /var не дает атаке на эти объекты повредить или заполнить весь жесткий диск
/home	Поскольку каталоги учетных записей расположены в этом каталоге, наличие отдельной учетной записи /home не даст неосторожному пользователю заполнить весь жесткий диск. Ему также удобно отделять пользовательские данные от операционной системы для легкого резервного копирования или новой установки. Часто /home создается как логический том LVM, поэтому он может увеличиваться в размерах по мере роста потребностей пользователя. Ему также могут быть назначены пользовательские квоты для ограничения применения диска
/tmp	Если каталог /tmp отделен от остальной части жесткого диска, то приложения, которым необходимо записать туда временные файлы, будут завершать свою обработку, даже если остальная часть диска заполнится

Обычные пользователи системы Linux редко видят необходимость в большом количестве разделов, но те, кто занимается крупными системами, в том числе их восстановлением, находят множество плюсов в том, что система разделена. Разделы уменьшают последствия преднамеренного повреждения (например, атаки типа DoS), проблемы из-за действий пользователей и случайного повреждения файловой системы.

Загрузчик операционной системы GRUB

Загрузчик позволяет выбрать, когда и как загружать операционные системы, установленные на жестких дисках компьютера. *GRand Unified Bootloader (GRUB)* — это самый популярный загрузчик, используемый для установленных систем Linux. Сегодня доступны две основные версии GRUB:

- GRUB Legacy (версия 1) — работала с более ранними версиями RHEL, Fedora и Ubuntu;
- GRUB 2 (версия 2) — текущие версии Red Hat Enterprise Linux, Ubuntu и Fedora применяют GRUB 2 в качестве загрузчика по умолчанию.

ПРИМЕЧАНИЕ

SYSLINUX — это еще один загрузчик для систем Linux. Обычно он не используется для установленных систем Linux. Однако применяется в качестве загрузчика для установочных CD и DVD Linux. SYSLINUX особенно хорошо подходит для загрузки образов CD ISO9660 (isolinux) и USB-накопителей (syslinux), а также для работы на старом оборудовании или загрузки PXE (pxelinux) системы по сети.

Если вы хотите загрузиться до определенного уровня запуска, добавьте нужный уровень в конец строки ядра. Например, чтобы загрузиться до третьего уровня запуска RHEL (многопользовательский плюс сетевой режим), добавьте `3` в конец строки ядра. Можно также загрузиться в однопользовательский режим (`1`), многопользовательский режим (`2`) или режим X GUI (`3`). Уровень `3` подходит, когда графический интерфейс временно недоступен. Уровень `1` подходит, если вы забыли свой пароль root.

По умолчанию при загрузке Linux появится заставка. Чтобы видеть сообщения с действиями, требующимися при загрузке системы, удалите параметр `rhgb quiet` из строки ядра. Это позволит просматривать сообщения по мере их поступления. Нажатие клавиши `Esc` во время загрузки приводит к тому же результату.

GRUB 2 — это новая версия GRUB Legacy, сильно отличающаяся от предыдущей. GRUB 2 был принят в качестве загрузчика по умолчанию для последних версий Red Hat Enterprise Linux, Fedora и Ubuntu. Основная функция загрузчика GRUB 2 по-прежнему заключается в поиске и запуске нужной операционной системы, но теперь инструменты и файлы конфигурации гораздо более мощные и гибкие.

В GRUB 2 файл конфигурации теперь называется `/boot/grub2/grub.cfg` или `/etc/grub2efi.cfg` (для систем, загружаемых с помощью EFI). Все, начиная с содержимого файла `grub.cfg` и заканчивая способом создания `grub.cfg`, отличается от параметров файла GRUB Legacy `grub.conf`.

Вот что нужно знать о файле `grub.cfg`.

- Не требуется редактировать файл `grub.cfg` вручную или добавлять RPM-пакеты ядра — `grub.cfg` автоматически генерируется из содержимого файла

/etc/default/grub и каталога /etc/grub.d/. Нужно только изменить или добавить эти файлы, чтобы настроить GRUB 2 самостоятельно.

- Файл grub.cfg может содержать синтаксис скриптов, включая функции, циклы и переменные.
- Имена устройств, необходимые для определения местоположения ядер и начальных RAM-дисков, могут быть более надежно идентифицированы с помощью меток или универсально уникальных идентификаторов (UUID). Это предотвращает изменение дискового устройства, такого как /dev/sda, на /dev/sdb при добавлении нового диска (из-за этого ядро не будет найдено).
- Для систем Fedora и RHEL файлы *con в каталоге /boot/loader/entries используются для создания записей, которые появляются в меню GRUB во время загрузки.

Можно создать свою запись для меню загрузки GRUB в соответствии с форматом существующей. Следующий файл в каталоге /boot/loader/entries создает пункт меню для загрузки ядра RHEL 8 и initrd:

```
title Red Hat Enterprise Linux (4.18.0-80.el8.x86_64) 8.0 (Ootpa)
version 4.18.0-80.el8.x86_64
linux /vmlinuz-4.18.0-80.el8.x86_64
initrd /initramfs-4.18.0-80.el8.x86_64.img $tuned_initrd
options $kernelopts $tuned_params
id rhel-20190313123447-4.18.0-80.el8.x86_64
grub_users $grub_users
grub_arg --unrestricted
grub_class kernel
```

Пункт в меню загрузки GRUB 2 здесь отображается как Red Hat Enterprise Linux (4.18.0-80.el8.x8_64) 8.0 (Ootpa).

Строка linux определяет местоположение ядра (/vmlinuz-4.18.0-80.el8.x86_64), а затем местоположение initrd (/initramfs-4.18.0-80.el8.x86 64.img).

В GRUB 2 существует еще множество возможностей, о которых вы можете узнать самостоятельно. Для этого введите команду info grub2. Команда info для GRUB 2 содержит множество информации о загрузке различных операционных систем, написания собственных файлов конфигурации, работы с файлами образов GRUB, настройки переменных среды GRUB и работы с другими функциями GRUB.

Резюме

Для каждого дистрибутива Linux имеется особый метод установки, но все равно необходимо выполнить много общих действий независимо от того, какую систему Linux вы устанавливаете. Важно разбираться в вопросах разбиения диска, параметров загрузки и настройки загрузчиков для каждой системы Linux.

В этой главе я пошагово описал процедуры установки для рабочей станции Fedora (с использованием «живого» носителя) и Red Hat Enterprise Linux (с установочного носителя). Вы узнали, как развертывание Linux в облачных средах может отличаться от традиционных методов установки путем объединения метаданных с предварительно созданными файлами образов для работы на больших пулах вычислительных ресурсов.

Здесь рассматривались также специальные темы из области установки, включая использование параметров загрузки и разбиение диска на разделы. В главе 10 мы поговорим о том, как начать управлять программным обеспечением в системе Linux, когда она установлена.

Упражнения

Выполните данные упражнения, чтобы проверить знания об установке Linux. Я рекомендую делать это на компьютере, на котором нет операционной системы или важных данных (то есть данных, которые нельзя стирать). Можно воспользоваться компьютером, который позволяет устанавливать виртуальные системы, если он есть. Упражнения составлялись для установки Fedora 30 Workstation с «живого» носителя и RHEL 8 с установочного DVD.

1. Начните установку Fedora с «живого» носителя, используя как можно больше параметров по умолчанию.
2. После полной установки Fedora обновите все пакеты в системе.
3. Начните установку RHEL с установочного DVD, но сделайте так, чтобы она выполнялась в текстовом режиме. Завершите установку любым способом.
4. Начните установку RHEL с установочного DVD и создайте следующие разделы для диска: `/boot` — 1024 Мбайт, `/` — 6 Гбайт, `/var` — 2 Гбайт и `/home` — 2 Гбайт. Остальное пространство оставьте пустым.

ВНИМАНИЕ!

Выполнение четвертого задания приведет к удалению всего содержимого жесткого диска. Если вы просто хотите попрактиковаться в разделении диска на разделы без вреда для него, перезагрузите компьютер, не нажимая кнопку применения изменений. Если же это сделать, все незатронутые данные будут удалены.

10 Управление программами

В этой главе

- Установка программного обеспечения с рабочего стола.
- Управление пакетами с помощью RPM.
- Управление пакетами с помощью YUM.
- Управление пакетами с помощью команды `rpm`.
- Установка программного обеспечения на предприятии.

В дистрибутивах Linux, таких как Fedora и Ubuntu, нет необходимости изучать, как программное обеспечение упаковывается и управляется, чтобы его получить. Эти дистрибутивы включают в себя отличные инструменты установки программного обеспечения, которые связаны с огромными репозиториями данных. Пара щелчков кнопкой мыши — и вы используете программное обеспечение.

То, что сейчас управление программным обеспечением Linux так упростилось, — заслуга сообщества Linux, которое усердно работало над созданием форматов упаковки, сложных инструментов установки и высококачественных программных пакетов. Теперь не только легко получить обеспечение, но и легко им управлять, обновлять его и удалять после установки.

Эта глава начинается с описания установки программного обеспечения в Fedora с помощью нового инструмента с графическим интерфейсом. Обычному пользователю вряд ли понадобится нечто большее, чем несколько настольных программ и обновлений системы безопасности.

Чтобы углубиться в управление программным обеспечением Linux, далее поговорим о том, из чего состоят пакеты Linux (сравним `deb` и `rpm`), какие существуют базовые компоненты управления программным обеспечением и команды (`dnf`, `yum` и `rpm`) для управления программным обеспечением в Fedora и Red Hat Enterprise Linux. Затем рассмотрим, как управлять пакетами программного обеспечения для корпоративных вычислений.

Управление программным обеспечением на рабочем столе

Окно Software (Центр приложений) Fedora интуитивно понятно и позволяет выбрать и установить нужные настольные приложения, что не похоже на типичные методы установки обеспечения в Linux. Окно Software (Центр приложений) в Ubuntu имеет тот же интерфейс, что и в Fedora. В любом случае в этом окне самым маленьким программным компонентом, который можно установить, является приложение. В Linux программное обеспечение устанавливается через пакеты (например, `rpms` и `debs`).

На рис. 10.1 показан пример окна Software (Центр приложений).

Рис. 10.1. Устанавливайте пакеты программ и управляйте ими из окна Software (Центр приложений)

Чтобы перейти к этому окну в Fedora или Ubuntu, нажмите кнопку Activities (Обзор), затем введите Software (Центр приложений) и нажмите клавишу Enter. При первом открытии окна можно включить разрешение на использование сторонних репозиториев программного обеспечения. Окно Software (Центр приложений) — это

лучший способ установки приложений, ориентированных на рабочий стол, таких как текстовые процессоры, игры, графические редакторы и образовательные приложения.

В окне можно выбрать нужные приложения из раздела популярных приложений Editor's Picks (Выбор редакции), из категорий приложений Audio & Video (Аудио и видео), Games (Игры), Graphics & Photography (Графика и фотография) и т. д. или выполнить поиск по имени или описанию. Нажмите кнопку Install (Установить), чтобы загрузить и установить все пакеты программного обеспечения, необходимые для работы приложения.

В окне также можно просмотреть все установленные приложения (вкладка Installed (Установлено)) или список приложений, который можно обновить (вкладка Updates (Обновления)). Если вы хотите удалить установленное приложение, просто нажмите кнопку Remove (Удалить) рядом с именем пакета.

Если использовать Linux исключительно как настольную систему, то есть создавать документы, воспроизводить музыку и выполнять другие задачи, в центре приложений можно найти все, что для этого нужно. По умолчанию система подключается к основному репозиторию программного обеспечения Fedora и предоставляет доступ к сотням приложений. Как отмечалось ранее, также есть возможность получить доступ к сторонним приложениям, которые вы можете бесплатно использовать, но не распространять.

Окно Software (Центр приложений) позволяет загружать и устанавливать сотни приложений из репозитория Fedora, но на самом деле в нем содержатся десятки тысяч пакетов программного обеспечения. В таком случае какие пакеты из этого репозитория недоступны, когда они могут понадобиться и как получить их, а также пакеты из других репозиториев программного обеспечения?

За пределами центра приложений

Используя лишь настольную систему Linux, можно вполне довольствоваться имеющейся сотней приложений из центра приложений. Версии с открытым исходным кодом наиболее распространенных типов настольных приложений становятся доступными после подключения Fedora к Интернету.

Далее приведены плюсы выхода за пределы того, что доступно в центре приложений обычному пользователю.

- **Больше репозиториев.** Fedora и Red Hat Enterprise Linux используют только свободно распространяемое программное обеспечение с открытым исходным кодом. Возможно, вам понадобится установить какое-то коммерческое программное обеспечение, например Adobe Flash Player, или несвободное из таких репозиториев, как `rpmfusion.org`.
- **Не только настольные приложения.** Десятки тысяч программных пакетов в репозитории Fedora нельзя получить с помощью окна Software (Центр

приложений). Большинство их вообще не связаны с приложениями с графическим интерфейсом. Например, некоторые пакеты содержат только инструменты командной строки, системные службы, средства программирования или документацию, которая не отображается в окне Software (Центр приложений).

- **Гибкость.** Не каждый пользователь знает, что при установке приложения через окно Software (Центр приложений) на самом деле можно устанавливать множество пакетов RPM. Этот набор пакетов может быть просто набором по умолчанию, включающим документацию, дополнительные шрифты, программные плагины или несколько языковых пакетов, которые могут понадобиться, а могут и не понадобиться. Команды `yum` и `rpm` позволяют определять, какие именно пакеты, связанные с приложением или другой программной функцией, установлены в вашей системе.
- **Усложненные запросы.** Команды `yum` и `rpm` предоставляют подробную информацию о пакетах, группах пакетов и репозиториях.
- **Проверка программного обеспечения.** Команда `rpm` и другие инструменты позволяют проверить, был ли подписан пакет, или его компоненты изменены до либо после его установки.
- **Управление установкой программ.** Окно Software (Центр приложений) отлично справляется с установкой настольных программ в одной системе, но не позволяет расширить управление программным обеспечением на несколько систем. Для этого есть другие инструменты, помимо `rpm`.

Прежде чем переходить к инструментам командной строки для установки программного обеспечения в Linux и управления им, в следующем разделе мы рассмотрим, как в Linux работают базовые системы упаковки пакетов и управления ими. В частности, рассмотрим программу RPM, поскольку она используется в Fedora, Red Hat Enterprise Linux и связанных с ними дистрибутивах, а также в пакетах Deb, которые связаны с Debian, Ubuntu, Linux Mint и др.

Управление пакетами с помощью RPM и DEB

В первых системах Linux чтобы добавить программное обеспечение, необходимо было взять исходный код из проекта, скомпилировать его в исполняемые двоичные файлы и перенести на свой компьютер. При большом везении чья-то скомпилированная версия могла подойти и для вашего компьютера.

Формой пакета может быть *tar*-файл, содержащий исполняемые файлы (команды), документацию, файлы конфигурации и библиотеки (*тарбол* — это один файл, в котором несколько файлов собраны вместе для удобного хранения или распространения). При установке программного обеспечения из тарбола файлы из него распределяются по всей системе Linux в соответствующие каталоги

(`/usr/share/man`, `/etc`, `/bin`, `/lib` и др.). Да, легко создать архивный файл и просто поместить набор программного обеспечения в систему Linux, однако этот метод установки затрудняет выполнение следующих действий.

- **Определение зависимости одних программ от других.** Чтобы устанавливаемое программное обеспечение заработало, нужно знать, зависит ли оно от другого. Если да, то придется отследить это программное обеспечение и установить его тоже (и оно также может зависеть от другого ПО).
- **Перечисление программ.** Даже если известно имя команды, вы можете не знать, где находятся ее документация или файлы конфигурации.
- **Удаление программ.** Если исходный тарбол не сохранен, вы не будете знать, где находятся все файлы, и не сможете удалить их все. Но даже если бы знали, вам пришлось бы удалять каждый из них по отдельности.
- **Обновление программ.** Тарболы не предназначены для хранения метаданных о своем содержимом. После установки содержимого тарбола нет возможности определить, какая версия программного обеспечения используется, что затрудняет поиск ошибок и обновление программ.

Чтобы устранить эти проблемы, перешли от простых тарболов, содержащих пакеты с программами, к более сложной упаковке. За немногими исключениями (Gentoo, Slackware и некоторые другие), в большинстве дистрибутивов Linux применяется один из двух форматов упаковки — DEB и RPM.

- **DEB (.deb).** Проект Debian GNU/Linux создал упаковку `deb`, которую используют Debian и другие дистрибутивы, основанные на Debian (Ubuntu, Linux Mint, KNOPPIX и т. д.). Работая с инструментами `apt-get`, `apt` и `dpkg`, дистрибутивы Linux могут устанавливать, обновлять и удалять программное обеспечение, а также управлять им.
- **RPM (.rpm).** Первоначально названный Red Hat Package Manager, но позже переименованный, RPM Package Manager является предпочтительным форматом пакетов для дистрибутивов SUSE, Red Hat (RHEL и Fedora) и тех, которые основаны на дистрибутивах Red Hat (CentOS, Oracle Linux и т. д.). Команда `rpm` стала первым инструментом для управления пакетами RPM. Позже для улучшения работы RPM был добавлен `yum`, а его в конечном итоге заменит `dnf`.

Для управления программным обеспечением на отдельных компьютерах подходит как RPM, так и DEB, что вызывает споры у некоторых пользователей. Хотя RPM — предпочтительный формат для обновления и обслуживания корпоративного программного обеспечения, а также управления им, DEB очень популярен у многих пользователей Linux. Эта глава охватывает как программу RPM (Fedora и Red Hat Enterprise Linux), так и в некоторой степени DEB, а также управление программным обеспечением.

Управление пакетами с помощью DEB

Программные пакеты Debian содержат несколько файлов и метаданных, относящихся к некоторому набору программного обеспечения в формате архива. Файлы могут быть исполняемыми файлами (командами), файлами конфигурации, документацией или другими программными элементами. Метаданные включают в себя зависимости, сведения о лицензировании, размеры пакетов, описания и другую информацию. Для работы с файлами DEB в Ubuntu, Debian и других дистрибутивах Linux существуют специальные консольные утилиты и инструменты с графическим интерфейсом, например такие.

- **Центр приложений Ubuntu (Ubuntu Software Center).** Перейдите в окно Software (Центр приложений) в Ubuntu через меню Acivities (Приложения) на рабочем столе GNOME. В появившемся окне можно искать нужные приложения и пакеты с помощью поиска, по ключевым словам или категориям.
- **Команда aptitude.** Команда `aptitude` — это установщик пакетов, который предоставляет экранное меню в оболочке. После выполнения команды используйте клавиши со стрелками, чтобы выделить нужный элемент, и нажмите клавишу Enter, чтобы выбрать его. Команда позволяет обновлять пакеты, устанавливать новые или просматривать установленные.
- **Набор команд apt*.** Это набор команд `apt*` (`apt-get`, `apt`, `apt-config`, `apt-cache` и т. д.), с помощью которых можно управлять программными пакетами.

Центр приложений Ubuntu интуитивно понятен и позволяет легко находить и устанавливать нужные пакеты программного обеспечения. Однако существуют команды, которые помогают установить пакеты и управлять ими с помощью команд `apt*`. В примере я хочу найти и установить пакет `vsftpd`:

- `$ sudo apt-get update` — обновить версию пакета;
- `$ sudo apt-cache search vsftpd` — найти пакет по ключевому слову (например, `vsftpd`);
- `$ sudo apt-cache show vsftpd` — отобразить информацию о пакете;
- `$ sudo apt-get install vsftpd` — установить пакет `vsftpd`;
- `$ sudo apt-get upgrade` — обновить установленные пакеты, если есть обновление;
- `$ sudo apt-cache pkgnames` — перечислить все установленные пакеты.

ПРИМЕЧАНИЕ

Обратите внимание на то, что в этих примерах командам apt* предшествует команда sudo. Это связано с тем, что администратор в Ubuntu может выполнять административные команды как обычный пользователь с правами sudo.

Для команд `apt*` существует множество других способов применения, которые стоит попробовать. Если у вас установлена система Ubuntu, я рекомендую ввести команду `man apt`, чтобы узнать о возможностях команд `apt`.

Управление пакетами с помощью RPM

Пакет RPM — это совокупность файлов, необходимых для работы функций, таких как текстовый процессор, средство просмотра фотографий или файловый сервер. В пакет RPM входят команды, файлы конфигурации и документация, составляющие программный компонент. Файл RPM включает в себя также метаданные, которые хранят информацию о содержимом этого пакета, о том, откуда он взялся, что ему нужно для запуска и др.

Что такое RPM

Даже не заглядывая в пакет RPM, можно многое понять по его названию. Чтобы узнать имя пакета RPM, установленного в данный момент в вашей системе (в данном примере браузер Firefox), введите из командной консоли в Fedora или Red Hat Enterprise Linux:

```
# rpm -q firefox
firefox-67.0-4.fc30.x86_64
```

Из этого понятно, что базовое имя пакета — `firefox`. Номер версии — 67.0 (присвоен разработчиком Firefox — проектом Mozilla). Релиз — 4 (назначается Fedora каждый раз, когда пакет обновляется под одним и тем же номером выпуска). Пакет `firefox` построен для Fedora 30 (`fc30`) и скомпилирован для 64-битной версии x86 (`x86_64`).

Когда пакет `firefox` был установлен, он, вероятно, был скопирован с установочного носителя (например, CD или DVD) или загружен из репозитория YUM (подробнее об этом позже). Имя файла RPM в локальном каталоге выглядело бы как `firefox-67.0-4.fc30.x86_64.RPM`, и можно было бы установить его оттуда. Независимо от того, откуда взялся пакет, после установки его имя и другая информация сохраняются в базе данных RPM на локальном компьютере.

Чтобы больше узнать о том, что находится внутри пакета RPM, используйте параметры, отличные от команды `rpm`, для запроса этой локальной базы данных RPM, как в этом примере:

```
# rpm -qi firefox
Name        : firefox
Version     : 67.0
Release     : 4.fc30
Architecture: x86_64
Install Date: Sun 02 Jun 2019 09:37:25 PM EDT
Group       : Unspecified
Size        : 266449296
License     : MPLv1.1 or GPLv2+ or LGPLv2+
Signature   : RSA/SHA256, Fri 24 May 2019 12:09:57 PM EDT, Key ID
ef3c111fcfc659b9
Source RPM  : firefox-67.0-4.fc30.src.rpm
Build Date  : Thu 23 May 2019 10:03:55 AM EDT
Build Host  : buildhw-08.phx2.fedoraproject.org
Relocations : (not relocatable)
Packager    : Fedora Project
```

```
Vendor      : Fedora Project
URL         : https://www.mozilla.org/firefox/
Bug URL     : https://bugz.fedoraproject.org/firefox
Summary     : Mozilla Firefox Web browser
Description :
Mozilla Firefox is an open-source web browser, designed for standards
compliance, performance and portability.
```

Помимо информации, полученной из самого имени пакета, параметр `-qi` (query information) позволяет увидеть, кто создал пакет (проект Fedora), когда он был собран и когда установлен. Перечисляются также группа, в которой находится пакет (не указано), его размер и лицензия. Чтобы узнать больше о пакете, URL-адрес указывает на его страницу в Интернете, а строки `Summary` и `Description` сообщают, для чего он используется.

Откуда берутся файлы RPM

Программное обеспечение для дистрибутивов Linux или созданное для работы с ними поставляется из тысяч проектов с открытым исходным кодом. Эти проекты называются *upstream software providers*, они обычно создают доступное для всех программное обеспечение на определенных условиях. Дистрибутив Linux берет исходный код и встраивает его в двоичные файлы. Затем он собирает документацию, файлы конфигурации, скрипты и другие компоненты от поставщика и соединяет с двоичными файлами. После этого пакет RPM собирается и подписывается (что позволяет пользователям проверить его), затем добавляется в хранилище RPM для конкретных дистрибутива и архитектуры (32-разрядная x86, 64-разрядная x86 и т. д.). Репозиторий размещается на установочном CD или DVD либо в каталоге, доступном в качестве FTP-сервера, веб-сервера или сервера NFS.

Установка файлов RPM

В начальную установку систем Fedora или Red Hat Enterprise Linux входит много пакетов RPM. После установки Linux можно добавлять пакеты с помощью окна Software (Центр приложений), как было описано ранее. (См. главу 9, чтобы узнать, как устанавливать систему Linux.)

Первым инструментом, разработанным для установки пакетов RPM, была команда `rpm`. С ее помощью можно устанавливать, обновлять, запрашивать, проверять и удалять пакеты RPM. Однако у команды есть серьезные недостатки.

- **Зависимости.** Большинство пакетов RPM в системе станут работать, только если установлено другое программное обеспечение (библиотека, исполняемые файлы и т. д.). Если такого ПО в системе нет, то при попытке установить пакет с помощью `rpm` установка не будет выполнена и система сообщит, какие компоненты ей для этого необходимы. После этого еще нужно покопаться, чтобы найти, какой пакет содержит нужный компонент. А этот пакет тоже может от чего-то зависеть, и потребуется это что-то установить, чтобы все заработало. Эту

ситуацию с любовью называют адом зависимостей и часто используют в качестве аргумента, когда говорят, что DEB-пакеты были лучше RPM. Инструменты DEB создавались для автоматического определения всех зависимостей пакетов задолго до того, как до этого дошли инструменты RPM.

- **Местоположение файлов RPM.** Команда `rpm` ожидает, что при установке файла RPM вы укажете его точное местоположение. Другими словами, вам пришлось бы предоставить команде параметр `firefox-67.0-4.fc30.x86_64.rpm`, если бы RPM находился в текущем каталоге, или `example.com/firefox-67.0-4.fc30.x86_64.rpm`, если бы он был на сервере.

По мере роста популярности Red Hat Linux и других приложений, основанных на RPM, стало очевидно, что необходимо сделать установку пакетов более удобной. Для этого была создана функция YUM.

Управление пакетами RPM с помощью YUM

Проект *YellowDog Updater Modified (YUM)* создавался для упрощения процесса управления зависимостями пакетов RPM. От него требовалось, чтобы пакеты RPM перестали считаться отдельными компонентами, а стали частями более крупных репозиториев программного обеспечения.

Что касается репозиториев, то решать проблему с зависимостями должен не пользователь, установивший программное обеспечение, а дистрибутив Linux или сторонний распространитель, который выкладывает его для всеобщего доступа. Так, например, проект Fedora должен удовлетвориться тем, что каждый компонент, необходимый каждому пакету в его дистрибутиве Linux, разрешен каким-либо другим пакетом в репозитории.

Репозитории также могут основываться друг на друге. Например, если пользователь работает в репозитории `rpmfusion.org`, значит, он уже имеет доступ к основному репозиторию Fedora. Если пакет, установленный из `rpmfusion.org`, требует библиотеки или команды из основного репозитория Fedora, то пакет Fedora можно загрузить и установить одновременно с установкой пакета из `rpmfusion.org`.

Репозитории `yum` можно разместить в каталоге на веб-сервере (`http://`), FTP-сервере (`ftp://`), локальном носителе, таком как CD или DVD, или в локальном каталоге (`file://`). Расположение этих репозиториев будет храниться в системе пользователя в файле `/etc/yum.conf` или чаще в отдельных файлах конфигурации в каталоге `/etc/yum.repos.d`.

Переход от yum к dnf

Интерфейс командной строки `dnf` представляет собой следующее поколение YUM. Пакетный менеджер DNF, или *Dandified YUM* (github.com/rpm-software-management/dnf/), является частью Fedora, начиная с 18-й версии, и недавно был добавлен в RHEL 8. Как и `yum`, `dnf` — это инструмент для поиска, установки и запроса пакетов

RPM, добавленных из удаленных репозиториев программного обеспечения в локальную систему Linux, и управления ими.

Хотя `dnf` поддерживает базовую совместимость командной строки с `yum`, одно из его главных отличий заключается в том, что он придерживается строгого API, который помогает разрабатывать расширения и плагины для `dnf`.

Для наших целей почти все команды `yum`, описанные в этой главе, можно заменить на `dnf` или использовать так, как они даны. Команда `yum` — это символическая ссылка на `dnf` в Fedora и RHEL, поэтому ввод любой команды приводит к одному и тому же результату. См. больше дополнительной информации о DNF на странице DNF: dnf.readthedocs.io.

Как работает команда yum

Основной синтаксис команды `yum` выглядит так:

```
# yum [параметры] команда
```

С помощью этого синтаксиса можно найти пакеты, просмотреть информацию о них, узнать о группах пакетов, обновить или удалить пакеты, и это еще не все функции команды. Из репозитория YUM и локальной конфигурации пользователь может установить пакет, просто набрав что-то вроде следующего:

```
# yum install firefox
```

Пользователю нужно знать только имя пакета (его можно запросить разными способами, как описано в разделе «Поиск программных пакетов» далее в этой главе). Функция YUM находит последнюю версию этого пакета, доступную в репозитории, загружает его в локальную систему и устанавливает.

Чтобы изучить возможности функции YUM и настроить ее для своей системы, воспользуйтесь планом установки YUM, описанным далее.

ПРИМЕЧАНИЕ

В последних версиях систем Fedora и RHEL файлы конфигурации YUM фактически являются ссылками на файлы DNF в каталоге /etc/dnf. Помимо основного файла конфигурации dnf (/etc/dnf/dnf.conf), этот каталог содержит в основном модули и плагины, которые расширяют возможности управления пакетами RPM.

1. **Проверка каталога /etc/yum.conf.** При запуске любой команды `yum` проверяет файл `/etc/yum.conf` на наличие настроек по умолчанию. Файл `/etc/yum.conf` — это основной файл конфигурации YUM. Он помогает определить местоположение репозиториев, хотя чаще всего идентификация репозиториев происходит через каталог `/etc/yum.repos.d`. Пример файла `/etc/yum.conf` в системе RHEL 8 с добавлением других файлов:

```
[main]
gpgcheck=1
installonly_limit=3
```

```
clean_requirements_on_remove=True
best=True

cachedir=/var/cache/yum/$basearch/$releasever
keepcache=0
debuglevel=2
logfile=/var/log/yum.log
exactarch=1
plugins=1
```

Параметр `gpgcheck` используется для проверки каждого пакета по ключу создателей пакета RPM. Значение установлено по умолчанию (`gpgcheck=1`). Для пакетов в Fedora или RHEL ключ для проверки всех пакетов поставляется вместе с дистрибутивом. Чтобы установить пакеты, не принадлежащие вашему дистрибутиву, нужно либо импортировать ключ для проверки этих пакетов, либо отключить эту функцию (`gpgcheck=0`).

Параметр `install_onlylimit=3` позволяет хранить в системе до трех версий одного и того же пакета (не устанавливайте меньшее значение, чтобы всегда были доступны по крайней мере два пакета ядра). Параметр `clean_requirements_on_remove=True` указывает команде `yum` удалить зависимые пакеты вместе с основным, если они не требуются. Параметр `best=True` ищет последнюю доступную версию для обновления пакета.

Другие параметры, которые можно добавить в файл `yum.conf`, сообщают YUM, где хранить файлы кэша (`/var/cache/yum`) и записи журнала (`/var/log/yum.log`) и хранить ли файлы кэша после установки пакета (`0` — хранить не нужно). Значение параметра `debuglevel` в файле `yum.conf` можно увеличить, чтобы получать более подробные описания в файлах журнала.

Далее видно, следует ли точно использовать архитектуру (x86, x86_64 и т. д.) при выборе пакетов для установки (`1` означает «да») и нужно ли использовать плагины (`1` означает «да») для черных и белых списков или для подключения пакетов к Red Hat Network.

Чтобы узнать о других функциях файла `yum.conf`, введите команду `man yum.conf`.

2. **Проверка файлов /etc/yum.repos.d/*.repo.** Репозитории программного обеспечения можно включить, перекинув файлы, заканчивающиеся на `.repo`, в каталог `/etc/yum.repos.d/`, который указывает на расположение одного или нескольких репозиториев. В Fedora даже базовые репозитории Fedora включаются из файлов `.repo` в этом каталоге. Пример простого файла конфигурации `yum` с именем `/etc/yum.repos.d/myrepo.repo`:

```
[myrepo]
name=My repository of software packages
baseurl=http://myrepo.example.com/pub/myrepo
enabled=1
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/MYOWNKEY
```

Каждая запись в репозитории начинается с имени репозитория в квадратных скобках. Строка `name` содержит удобочитаемое описание репозитория. Строка `baseurl` определяет каталог, содержащий файлы RPM, которые могут быть записаны как `httpd://`, `ftp://` или `file://`.

Строка `enabled` сообщает, активна ли запись: `1` — активна, `0` — неактивна. Если этой строки нет, то запись активна. Последние две строки в коде указывают, следует ли проверять подписи пакетов в этом репозитории. Строка `gpgkey` отображает расположение ключа, который используется для проверки пакетов в репозитории.

Вы можете включить столько репозиториев, сколько захотите. Однако имейте в виду, что с помощью команд `yum` проверяется каждый репозиторий и метаданные обо всех пакетах загружаются в локальную систему. Поэтому для более эффективной и быстрой работы не включайте все репозитории сразу.

3. **Загрузка пакетов RPM и метаданных из репозитория YUM.** После того как `yum` определит расположение репозиториев, метаданные каждого репозитория из каталога `repodata` загружаются в локальную систему. На самом деле именно наличие каталога `repodata` в каталоге RPM указывает на то, что это репозиторий YUM.

   Метаданные хранятся в локальной системе в каталоге `/var/cache/yum`. Любые дальнейшие запросы о пакетах, группах пакетов или другой информации из репозитория собираются из кэшированных метаданных до тех пор, пока не истечет период ожидания.

   После этого `yum` извлекает свежие метаданные, если выполняется команда `yum`. По умолчанию период ожидания составляет 6 часов для `yum` и 48 часов для `dnf`. Его можно изменить, установив параметр `metadata_expire` в файле `/etc/yum.conf`.

   Затем `yum` просматривает пакеты, которые нужно установить, и проверяет, нужны ли им какие-либо зависимые пакеты. Когда список пакетов создан, `yum` спрашивает, можно ли загрузить все эти пакеты. Если согласиться, пакеты будут загружены в каталоги кэша и установлены.

4. **Установка пакетов RPM в файловую систему.** После того как все необходимые пакеты будут загружены в каталоги кэша, `yum` и `rpm` дают команду установить каждый пакет. Если пакет содержит предустановленные скрипты (которые могут создавать специальную учетную запись пользователя или каталоги), эти скрипты запускаются. Содержимое пакетов (команды, файлы конфигурации, документы и т. д.) копируется в файловую систему, в места, указанные в метаданных RPM. Затем запускаются все сценарии, необходимые после установки (post install). (Такие сценарии запускают дополнительные команды, необходимые для настройки системы после установки каждого пакета.)

5. **Хранение метаданных репозитория YUM в локальной базе данных RPM.** Метаданные, содержащиеся в каждом установленном пакете RPM, в итоге копируются в локальную базу данных RPM. База данных RPM содержится

в файлах, хранящихся в каталоге `/var/lib/rpm`. После того как информация об установленных пакетах окажется в локальной базе данных RPM, вы сможете выполнять все виды запросов к этой базе данных. Сможете просмотреть, какие пакеты установлены, перечислить их компоненты, а также скрипты или журналы изменений, связанные с каждым пакетом. Вы даже сможете проверить установленные пакеты по базе данных RPM, чтобы узнать, не вмешивался ли кто-нибудь в установленные компоненты.

Команда `rpm` (описана в разделе «Установка, запрос и проверка программ с помощью команды rpm» далее в этой главе) — это лучший инструмент для запроса базы данных RPM. Вы можете запускать отдельные запросы с помощью команды `rpm`, или использовать ее в сценариях для создания отчетов, или выполнять общие повторяющиеся запросы.

Теперь вы знаете об основных функциях команды `yum` и можете настроить систему Fedora для подключения к основному репозиторию Fedora и репозиторию обновлений Fedora. А еще — опробовать работу команды `yum` и установить пакеты. И подключить сторонние репозитории YUM для установки программного обеспечения.

YUM и сторонние репозитории

Репозитории программного обеспечения Fedora и Red Hat Enterprise Linux проверяются на наличие только того программного обеспечения, которое соответствует критериям, делающим его открытым и распространяемым. Однако в некоторых случаях может понадобиться выйти за пределы этих репозиториев. При этом вы должны осознавать, что некоторые сторонние репозитории несколько ограничены.

- Они могут предъявлять менее жесткие требования к распространению и свободе от патентных ограничений, чем репозитории Fedora и RHEL.
- Они могут вызвать конфликты программного обеспечения.
- Они могут иметь программное обеспечение, которое не является открытым исходным кодом, им можно пользоваться бесплатно, но распространять его нельзя.
- Они могут замедлить процесс установки ваших пакетов, поскольку метаданные загружаются для каждого включенного репозитория.

По этим причинам я рекомендую либо не подключать никаких дополнительных репозиториев программного обеспечения, либо включить только репозитории RPM Fusion (rpmfusion.org) для Fedora и EPEL (fedoraproject.org/wiki/EPEL) для Red Hat Enterprise Linux. RPM Fusion представляет собой слияние нескольких популярных сторонних репозиториев Fedora (Freshrpms, Livna.org и Dribble). Дополнительные сведения см. в разделе FAQ репозитория (rpmfusion.org/FAQ). Чтобы подключить бесплатный репозиторий RPM Fusion в Fedora, выполните следующие действия.

1. Откройте окно Terminal (Терминал).
2. Введите команду `su` и пароль `root`, когда появится приглашение.

3. Введите приведенную далее команду одной строкой без пробела между косой чертой и `rpmfusion` (если не сработает, перейдите в каталог `fedora` и выберите RPM, подходящий для вашей версии Fedora):

```
# rpm -Uvh http://download1.rpmfusion.org/free/fedora/
rpmfusion-free-release-stable.noarch.rpm
```

Репозиторий RPM Fusion Nonfree содержит кодеки, необходимые для воспроизведения многих популярных мультимедийных форматов. Чтобы подключить его в Fedora, введите (вновь одной строкой без пробела между частями):

```
# rpm -Uhv http://download1.rpmfusion.org/nonfree/fedora/
rpmfusion-nonfree-release-stable.noarch.rpm
```

Большинство других сторонних репозиториев содержат программное обеспечение без открытого исходного кода. Например, чтобы установить плагин Adobe Flash в Linux, скачайте пакет с репозиторием YUM из Adobe, затем с помощью команды `yum` установите плагин и обновляйте его по мере необходимости командой `yum update`.

Управление приложениями командой yum

Команда `yum` имеет десятки подкоманд, которые можно использовать для работы с пакетами RPM. В следующих разделах приведены примеры полезных команд `yum` для поиска, установки, запроса и обновления пакетов, связанных с репозиториями YUM. В разделе «Установка и удаление пакетов» описано, как устанавливать и удалять пакеты с помощью команды `yum`.

ПРИМЕЧАНИЕ

Метаданные, описывающие содержимое репозиториев YUM, загружаются из каждого подключенного репозитория YUM при первом запуске команды yum. Метаданные загружаются снова по истечении времени, заданного в параметре metadata_expire. Чем больше YUM-репозиториев подключено и чем они обширнее, тем больше времени может занять загрузка. Время загрузки можно сократить, увеличив срок действия (в файле /etc/yum.conf) или не подключая ненужные репозитории.

Поиск программных пакетов

Различные подкоманды поиска находят пакеты по ключевым словам, содержимому или другим атрибутам.

Допустим, вы хотите попробовать другой текстовый редактор, но не можете вспомнить его название. Начните с подкоманды `search` и введите команду `editor` для поиска редактора в названии или описании:

```
# yum search editor
...
eclipse-veditor.noarch : Eclipse-based Verilog/VHDL plugin
ed.x86_64 : The GNU line editor
emacs.x86_64 : GNU Emacs text editor
```

В результате поиска появляется длинный список пакетов, содержащих слово `editor` в названии или описании. Допустим, нам нужен редактор `emacs`. Чтобы получить информацию об этом пакете, можно использовать подкоманду `info`:

```
# yum info emacs
Name         : emacs
Epoch        : 1
Version      : 26.2
Release      : 1.fc30
Architecture : x86_64
Size         : 3.2 M
Source       : emacs-26.2-1.fc30.src.rpm
Repository   : updates
Summary      : GNU Emacs text editor
URL          : http://www.gnu.org/software/emacs/
License      : GPLv3+ and CC0-1.0
Description  : Emacs is a powerful, customizable, self-documenting, modeless text
             : editor. Emacs contains special code editing features, a scripting
             : language (elisp), and the capability to read mail, news, and more
             : without leaving the editor.
```

Если вы знаете имена нужных команды, файла конфигурации или библиотеки, но не знаете, в каком пакете они находятся, используйте подкоманду `provides` для поиска пакета. В примере видно, что команда `dvdrecord` является частью пакета `wodim`:

```
# yum provides dvdrecord
wodim-1.1.11-41.fc30.x86_64 : A command line CD/DVD recording program
Repo        : fedora
Matched from:
Filename    : /usr/bin/dvdrecord
```

Подкоманда `list` применяется для перечисления имен пакетов различными способами. Используйте ее с базовым именем пакета, чтобы найти версию и репозиторий. Вы можете перечислить только пакеты, которые доступны (`available`), или установлены (`installed`), или все (`all`):

```
# yum list emacs
emacs.i686     1:26.2-1.fc30    updates
# yum list available
CUnit.i686     2.1.3-17.el8  rhel-8-for-x86_64-appstream-rpms
CUnit.x86_64     2.1.3-17.el8  rhel-8-for-x86_64-appstream-rpms
GConf2.i686      3.2.6-22.el8  rhel-8-for-x86_64-appstream-rpms
LibRaw.i686      0.19.1-1.el8  rhel-8-for-x86_64-appstream-rpm
...
# yum list installed
Installed Packages
GConf2.x86_64       3.2.6-22.el8     @AppStream
ModemManager.x86_64 1.8.0-1.el8      @anaconda
...
# yum list all
...
```

Чтобы посмотреть, от каких компонентов зависит пакет, используйте подкоманду `deplist`. Команда `deplist` отображает компоненты (зависимости), а также пакет, в который входит этот компонент (имя поставщика). Команда позволяет найти нужный компонент в других репозиториях, если в пакетах системы его нет, например:

```
# yum deplist emacs | less
package: emacs-1:26.1-8.fc30.x86_64
  dependency: /bin/sh
   provider: bash-5.0.7-1.fc30.i686
   provider: bash-5.0.7-1.fc30.x86_64
  dependency: /usr/sbin/alternatives
   provider: alternatives-1.11-4.fc30.x86_64
```

Установка и удаление пакетов

Подкоманда `install` позволяет установить один или несколько пакетов вместе с любыми зависимыми пакетами. Команда `yum install` ищет несколько репозиториев для загрузки необходимых зависимостей, например:

```
# yum install emacs
...
Package                Architecture Version             Repository Size
================================================================================
Installing:
 emacs                 x86_64       1:26.2-1.fc30       updates   3.2 M
Installing dependencies:
 emacs-common          x86_64       1:26.2-1.fc30       updates    38 M
 ImageMagick-libs      x86_64       1:6.9.10.28-1.fc30 fedora    2.2 M
 fftw-libs-double      x86_64       3.3.8-4.fc30        fedora    984 k
 ...

Transaction Summary
================================================================================
Install  7 Packages

Total download size: 45 M
Installed size: 142 M
Is this ok [y/N]: y
```

В примере видно, что `emacs` требует загрузки пакета `emacs-common` и еще нескольких и все они стоят в очереди на установку. Шесть пакетов вместе требуют 45 Мбайт для загрузки, но впоследствии будут использовать 142 Мбайт. Нажмите клавишу `y`, чтобы установить пакеты. Введите параметр `-y` в командную строку (сразу после команды `yum`), чтобы не нажимать клавишу `y` для установки пакетов. Лично я предпочитаю видеть все пакеты, которые будут установлены, прежде чем соглашусь на это.

Вы можете переустановить пакет, если по ошибке удалили компоненты уже установленного. Если попытаетесь выполнить обычную установку, система отве-

тит, что изменений не требуется. Вместо этого примените подкоманду `reinstall`. Предположим, вы установили пакет `zsh`, а затем случайно удалили файл `/bin/zsh`. Недостающие компоненты можно восстановить, набрав следующее:

```
# yum reinstall zsh
```

Можно удалить пакет вместе с его зависимостями, которые не требуются другим пакетам, с помощью подкоманды `remove`. Например, чтобы удалить пакет `emacs` и его зависимости, введите:

```
# yum remove emacs
Removing:
 emacs                  x86_64       1:26.2-1.fc30      updates    38 M

Removing unused dependencies:
 ImageMagick-libs       x86_64       1:6.9.10.28-1.fc30 fedora    8.9 M
 emacs-common           x86_64       1:26.2-1.fc30      updates    89 M
 fftw-libs-double       x86_64       3.3.8-4.fc30       fedora    4.2 M
 ...

Transaction Summary
================================================================================
Remove  7 Packages

Freed space: 142 M
Is this ok [y/N]: y
```

Обратите внимание на то, что используемая память, указанная для каждого пакета, является фактической (именно столько места пакет занимает в файловой системе), а не размером, указанным при загрузке (он значительно меньше).

Еще один способ удалить набор установленных пакетов — подкоманда `history`. С ее помощью можно увидеть все действия команд `yum` и отменить все транзакции. Другими словами, все установленные пакеты можно удалить, применив параметр `undo` подкоманды `history`, как показано в следующем примере:

```
# yum history
ID     | Command line   | Date and time    | Action(s)      | Altered
-------------------------------------------------------------------------------
    12 | install emacs  | 2019-06-22 11:14 | Install        |    7
...
# yum history info 12
Transaction ID : 12
...
Command Line    : install emacs
...
# yum history undo 12
Undoing transaction 12, from Sat 22 Jun 2019 11:14:42 AM EDT

Install emacs-1:26.2-1.fc30.x86_64                @updates
    Install emacs-common-1:26.2-1.fc30.x86_64         @updates
...
```

Прежде чем отменить транзакцию, ее можно просмотреть, чтобы точно узнать, какие пакеты были задействованы. Это позволит избежать ошибок и не удалить нужные пакеты. Отменив транзакцию 12, вы удаляете все пакеты, установленные во время нее. Чтобы отменить установку, которая включала десятки или даже сотни пакетов, используйте именно подкоманду `undo`.

Обновление программных пакетов

По мере появления новые версии пакетов помещаются в отдельные репозитории обновлений или просто добавляются в исходный репозиторий. Если доступны несколько версий пакета (в одном репозитории или в другом подключенном репозитории), `yum` предоставляет последнюю версию при установке. Для некоторых пакетов, таких как пакет ядра Linux, можно сохранить несколько версий.

Если новая версия пакета появится позже, ее можно загрузить и установить с помощью подкоманды `update`.

Подкоманда `check-update` проверяет наличие обновлений. Подкоманда `update` используется для обновления как одного пакета, так и нескольких сразу, для которых в данный момент имеется обновление. Или можно просто обновить один пакет (например, пакет `cups`), как в этом примере:

```
# yum check-update
...
file.x86_64       5.36-3.fc30    updates
file-libs.x86_64  5.36-3.fc30    updates
firefox.x86_64    67.0.4-1.fc30  updates
firewalld.noarch  0.6.4-1.fc30   updates
...
# yum update
Dependencies resolved.
================================================================
 Package              Arch   Version         Repository    Size
================================================================
Upgrading:
 NetworkManager       x86_64 1:1.16.2-1.fc30 updates      1.7 M
 NetworkManager-adsl x86_64 1:1.16.2-1.fc30 updates       25 k
...
Transaction Summary
================================================================
Install     7 Packages
Upgrade   172 Package(s)
Total download size: 50 M
Is this ok [y/N]: y
# yum update cups
```

Команда `check-update` запросила обновление пакета `cups`. Если для обновления `cups` необходимо обновить другие зависимые пакеты, то они также будут загружены и установлены.

Обновление групп пакетов

Чтобы упростить управление несколькими пакетами сразу, YUM позволяет использовать группы пакетов. Например, можно установить целиком GNOME Desktop Environment (весь рабочий стол) или Virtualization (все пакеты, необходимые для настройки компьютера в качестве хоста виртуализации). Начните с запуска подкоманды `grouplist`, чтобы увидеть список имен групп:

```
# yum grouplist | less
Available Environment Groups:
   Fedora Custom Operating System
   Minimal Install
   Fedora Server Edition
...
Installed Groups:
   LibreOffice
   GNOME Desktop Environment
   Fonts
...
Available Groups:
   Authoring and Publishing
   Books and Guides
   C Development Tools and Libraries
...
```

Допустим, вы решили попробовать другую среду рабочего стола и хотите узнать, что находится в группе LXDE. Для этого используйте подкоманду `groupinfo`:

```
# yum groupinfo LXDE
Group: LXDE
 Description: LXDE is a lightweight X11 desktop environment...
 Mandatory Packages:
...
   lxde-common
   lxdm
   lxinput
   lxlauncher
   lxmenu-data
...
```

В дополнение к описанию группы `groupinfo` показывает пакеты `Mandatory Packages` (обязательные, которые всегда устанавливаются вместе с группой), `Default Packages` (те, что установлены по умолчанию, но могут быть исключены) и `Optional Packages` (необязательные, которые являются частью группы, но не установлены по умолчанию). С помощью инструментов с графическим интерфейсом для установки групп пакетов можно отключить установку пакетов по умолчанию (`Default Packages`) или изменить установку дополнительных (`Optional Packages`).

Чтобы установить группу пакетов, используйте подкоманду `groupinstall`:

```
# yum groupinstall LXDE
```

В примере подкоманда `groupinstall` установила 101 новый пакет и обновила пять установленных. Чтобы удалить всю группу сразу, задействуйте подкоманду `groupremove`:

```
# yum groupremove LXDE
```

Базы данных и кэш пакетов RPM

Несколько подкоманд `yum` помогают выполнять задачи обслуживания, например проверку наличия проблем с базой данных RPM или очистку кэша. В YUM есть инструменты для обслуживания пакетов RPM и обеспечения эффективности и безопасности ПО системы.

Время от времени необходимо очищать кэш. Если сохранить загруженные пакеты после их установки (по умолчанию они удаляются благодаря параметру `keepcache=0` файла `/etc/yum.conf`), то каталоги с кэшем (в каталоге `/var/cache/yum`) могут заполниться. Метаданные, хранящиеся в каталогах кэша, можно очистить, что вызовет загрузку свежих метаданных из всех подключенных репозиториев YUM при следующем запуске `yum`. Варианты очистки кэша:

```
# yum clean packages
14 files removed
# yum clean metadata
Cache was expired
16 files removed
# yum clean all
68 files removed
```

Вполне возможно, хотя и маловероятно, что база данных RPM окажется повреждена. Это может случиться, если во время установки пакета произойдет что-то неожиданное, например внезапное отключение системы. Вы можете проверить базу данных RPM на наличие ошибок (`yum check`) или просто перестроить ее файлы. Пример поврежденной базы данных RPM и команда `rpm`, которая используется для ее исправления:

```
# yum check
error: db5 error(11) from dbenv->open: Resource temporarily
unavailable
error: cannot open Packages index using db5-Resource temporarily
unavailable(11)
error: cannot open Packages database in /var/lib/rpm
Error: Error: rpmdb open failed
# rpm --rebuilddb
# yum check
```

Здесь команда `yum clean` удаляет кэшированные данные из подкаталогов `/var/cache/yum`. Команда `rpm --rebuilddb` перестраивает базу данных. Команда `yum check` проверяет наличие проблем с локальным кэшем RPM и базой данных, но обрати-

те внимание на то, что для устранения этой проблемы в примере использовалась команда rpm.

В общем, для работы с локальной базой данных RPM лучше всего подходит команда rpm.

Загрузка RPM-пакетов из репозитория YUM

Если вы хотите просмотреть пакет, не устанавливая его, можете загрузить его с помощью команды dnf или yumdownloader (для более ранних версий). В любом случае пакет будет загружен из репозитория YUM и скопирован в ваш текущий каталог.

Например, чтобы загрузить последнюю версию пакета Firefox с помощью команды yumdownloader из репозитория YUM в текущий каталог, введите:

```
# yumdownloader firefox
firefox-67.0.4-1.fc30.x86_64.rpm    6.1 MB/s |  92 MB      00:14
```

Чтобы использовать команду dnf, введите следующее:

```
# dnf download firefox
firefox-60.7.0-1.el8_0.x86_64.rpm   6.1 MB/s |  93 MB      00:15
```

Когда загруженный пакет RPM добавлен в текущий каталог, можно применять для него различные команды rpm, как описано в следующем разделе.

Установка, запрос и проверка программ с помощью команды rpm

В локальной базе данных RPM содержится огромное количество информации об установленных пакетах. Команда rpm имеет десятки параметров, позволяющих найти сведения о каждом пакете, например о файлах, которые он включает, о том, кто его создал, когда он был установлен, его размере и многих других атрибутах. Поскольку база данных содержит скрытые данные (fingerprints (md5sums)) о каждом файле в каждом пакете, их можно запросить с помощью RPM, чтобы узнать, были ли файлы изменены.

Команда rpm все еще может выполнять основные действия по установке и обновлению, хотя большинство людей используют ее так, только когда в локальном каталоге находится готовый к установке пакет. Итак, опробуем работу команды. Для загрузки последней версии пакета zsh введите следующую команду:

```
# dnf download zsh
zsh-5.5.1-6.el8.x86_64.rpm       3.0 MB/s | 2.9 MB    00:00
```

Когда пакет zsh будет загружен в текущий каталог, выполните на нем несколько команд rpm.

Установка и удаление пакетов с помощью команды rpm

Чтобы установить пакет с помощью команды `rpm`, введите:

```
# rpm -i zsh-5.5.1-6.el8.x86_64.rpm
```

Обратите внимание на то, что для установки с помощью `rpm` задается полное имя пакета, а не только базовое. Если ранее пакет `zsh` был установлен, его можно обновить с помощью параметра `-U`. Во время обновления можно также использовать параметры `-h` и `-v` для вывода хештегов и более подробной информации:

```
# rpm -Uhv zsh-5.5.1-6.el8.x86_64.rpm
Verifying...          ########################## [100%]
Preparing...          ########################## [100%]
   1:zsh-5.5.1-6.el8  ########################## [100%]
```

Установка (`install`, параметр `-i`) загружает пакет только в том случае, если он еще не установлен, а вот обновление (`upgrade`, параметр `-U`) устанавливает его в любом случае. Третий тип установки (`freshen`, параметр `-F`) загружает пакет только в том случае, если на компьютере имеется более ранняя версия, как в следующем примере:

```
# rpm -Fhv *.rpm
```

Используйте параметр `freshen` (`-F`), если находитесь в каталоге, содержащем тысячи RPM, но хотите обновить только уже установленные (более ранней версии) и пропустить неустановленные.

К любому из вариантов установки можно добавить другие интересные параметры. Параметр `--replacepkgs` позволяет переустановить существующую версию пакета (если, например, вы ошибочно удалили некоторые компоненты), а параметр `--oldpackage` позволяет заменить пакет более ранней версией:

```
# rpm -Uhv --replacepkgs emacs-26.1-5.el8.x86_64.rpm
# rpm -Uhv --oldpackage zsh-5.0.2-25.el7_3.1.x86_64.rpm
```

Удалить пакет можно с помощью параметра `-e`. Для этого необходимо только его базовое имя, например:

```
# rpm -e emacs
```

Команда `rpm -e emacs` будет успешно выполнена, поскольку никакие другие пакеты не зависят от `emacs`. Однако для команды `emacs-common` это не сработает, так как пакет зависит от `emacs`. Если бы сначала попытаться удалить `emacs-common`, команда завершилась бы ошибкой с сообщением `"Failed dependencies"`.

Запрос информации с помощью команды rpm

После установки пакета можно запросить сведения о нем. С помощью параметра `-q` вы можете просмотреть информацию о пакете, включая описание (`-qi`), список файлов (`-ql`), документацию (`-qd`) и конфигурационные файлы (`-qc`):

```
# rpm -qi zsh
Name        : zsh
Version     : 5.5.1
Release     : 6.el8
...
# rpm -ql zsh
/etc/skel/.zshrc
/etc/zlogin
/etc/zlogout
...
# rpm -qd zsh
/usr/share/doc/zsh/BUGS
/usr/share/doc/zsh/CONTRIBUTORS
/usr/share/doc/zsh/FAQ
...
# rpm -qc zsh
/etc/skel/.zshrc
/etc/zlogin
/etc/zlogout
...
```

Используйте параметры для запроса любой части информации, содержащейся в RPM. Вы можете узнать, что требуется RPM для установки (`--requirements`), какую версию программного обеспечения пакет предоставляет (`--provides`), какие сценарии запускаются до и после установки или удаления RPM (`--scripts`) и какие изменения были внесены в RPM (`--changelog`):

```
# rpm -q --requires emacs-common
/bin/sh
/usr/bin/pkg-config
/usr/sbin/alternatives
...
# rpm -q --provides emacs-common
config(emacs-common) = 1:26.2-1.fc30
emacs-common = 1:26.2-1.fc30
emacs-common(x86-64) = 1:26.2-1.fc30
emacs-el = 1:26.2-1.fc30
pkgconfig(emacs) = 1:26.2
# rpm -q --scripts httpd
postinstall scriptlet (using /bin/sh):
if [ $1 -eq 1 ] ; then
        # Initial installation
        systemctl --no-reload preset httpd.service...
...
# rpm -q --changelog httpd | less
* Thu May 02 2019 Lubos Uhliarik <luhliari@redhat.com> - 2.4.39-4
- httpd dependency on initscripts is unspecified (#1705188)
* Tue Apr 09 2019 Joe Orton <jorton@redhat.com> - 2.4.39-3
...
```

В двух предыдущих примерах видно, что сценарии внутри пакета `httpd` используют команду `systemctl` для настройки службы `httpd`. Параметр `--changelog` позволяет увидеть причины изменений в каждой версии пакета.

С помощью функции `--queryformat` вы можете запрашивать различные теги информации и выводить их в любой понравившейся форме. Запустите параметр `--querytags`, чтобы увидеть все доступные теги:

```
# rpm --querytags | less
ARCH
ARCHIVESIZE
BASENAMES
BUGURL
...
# rpm -q binutils --queryformat "The package is %{NAME} \
     and the release is %{RELEASE}\n"
The package is binutils and the release is 29.fc30
```

Все запросы, сделанные ранее, были направлены в локальную базу данных RPM. Добавив параметр `-p` к другим параметрам, можно запросить файл RPM, находящийся в вашем локальном каталоге. Параметр `-p` позволяет заглянуть внутрь пакета и проверить его, прежде чем установить в свою систему.

Если вы еще не установили пакет `zsh`, то сделайте это и поместите его в свой локальный каталог (`dnf download zsh`). Затем выполните команды запроса в файле RPM:

```
# rpm -qip zsh-5.7.1-1.fc30.x86_64.rpm Просмотр информации о файле RPM
# rpm -qlp zsh-5.7.1-1.fc30.x86_64.rpm Перечисление всех файлов в файле RPM
# rpm -qdp zsh-5.7.1-1.fc30.x86_64.rpm Просмотр документации в файле RPM
# rpm -qcp zsh-5.7.1-1.fc30.x86_64.rpm Перечисление файлов конфигурации в файле RPM
```

Проверка RPM-пакетов

С помощью параметра `-V` можно проверить пакеты, имеющиеся в системе, чтобы увидеть, изменялись ли компоненты с момента их установки. Нормально, если файлы конфигурации изменяются с течением времени, а вот двоичные файлы (команды в `/bin`, `/sbin` и т. д.) изменяться не должны. Измененные двоичные файлы указывают на то, что система, вероятно, была взломана.

В приведенном далее примере я установлю пакет `zsh` и испорчу его. Если вы хотите попробовать сделать то же самое, обязательно удалите или переустановите пакет, когда закончите:

```
# rpm -i zsh-5.7.1-1.fc30.x86_64.rpm
# echo hello > /bin/zsh
# rm /etc/zshrc
# rpm -V zsh
missing   c  /etc/zshrc
S.5....T.    /bin/zsh
```

В примере видно, что файл `/bin/zsh` был изменен, а файл `/etc/shrch` удален. То, что на выходе после `rpm -V` вместо точки появляется буква или цифра, указывает на наличие изменений.

Значения букв, которые могут заменить точки (по порядку):

- `S` — изменен размер файла;
- `M` — изменен режим (включая права и тип файла);
- `5` — изменена контрольная сумма MD5;
- `D` — несоответствие главных и второстепенных номеров устройств;
- `L` — несоответствие пути `readLink(2)`;
- `U` — изменение прав пользователя;
- `G` — изменение прав группы;
- `T` — время модификации (`mTime`) изменено;
- `P` — привилегии изменены.

Эти показатели находятся в разделе Verify руководства `rpm`. В моем примере размер файла изменился (`S`), контрольная сумма `md5sum (5)` изменена (проверка по скрытым данным) и время модификации (`T`) в файле отличается.

Чтобы восстановить исходное состояние пакета, используйте `rpm` с параметром `--replacepkgs`, как показано далее. (Команда `yum reinstall zsh` будет работать так же.) Затем снова проверьте команду с помощью параметра `-V`. Отсутствие вывода из параметра `-V` означает, что все файлы вернулись в исходное состояние:

```
# rpm -i --replacepkgs zsh-5.7.1-1.fc30.x86_64.rpm
# rpm -V zsh
```

Рекомендуется создать резервную копию базы данных RPM (из файла `/var/lib/rpm`) и скопировать ее на какой-нибудь носитель только для чтения (например, CD). Так при проверке пакетов, которые могут быть взломаны, вы будете уверены в том, что база данных точно не взломана.

Управление программным обеспечением на предприятии

На этом этапе вы должны хорошо понимать, как устанавливать, запрашивать и удалять пакеты и иным образом манипулировать ими с помощью инструментов с графическим интерфейсом, а также команд `yum` и `rpm`. Для работы с файлами RPM на крупном предприятии эти знания нужно расширить.

Функции управления пакетами RPM на предприятии в Red Hat Enterprise Linux более сложные, но и более мощные. Вместо одного большого репозитория с программами, как в Fedora, RHEL предоставляет программное обеспечение

по платной подписке Red Hat с доступом к различным каналам и продуктам (RHEL, Red Hat Virtualization, Red Hat Software Collections и т. д.).

С точки зрения корпоративных вычислений одним из значительных преимуществ разработки пакетов RPM является то, что управление ими может быть автоматизировано. Другие схемы упаковки Linux позволяют пакетам останавливать свою установку и запрашивать информацию, например расположение каталога или имя пользователя. Пакеты RPM устанавливаются без остановки с помощью следующих инструментов.

- **Файлы kickstart.** Все вопросы, на которые вы отвечаете во время установки вручную, и все выбранные пакеты могут быть добавлены в файл, называемый *файлом kickstart*. При запуске установщика Fedora или Red Hat Enterprise Linux файл `kickstart` добавляется в командной строке загрузки. С этого момента процесс установки завершается самостоятельно. Можно внести любые изменения в установку пакетов, запустив скрипты до и после установки из файла `kickstart`, например, для добавления учетных записей пользователей или изменения файлов конфигурации.
- **Среда для загрузки PXE.** PXE-сервер можно настроить так, чтобы клиентские компьютеры могли загружать ядро Anaconda (инсталлятор) и выбранный файл `kickstart`. Совершенно пустой компьютер с картой сетевого интерфейса (NIC), поддерживающей загрузку PXE, может просто загрузиться с этой карты, чтобы запустить процесс новой установки. Другими словами, включите компьютер, и, если он попадает в порядок загрузки NIC, через несколько минут у вас будет свежеустановленная система, настроенная автоматически в соответствии с вашими точными указаниями.
- **Сервер Satellite (инструмент Spacewalk).** Системы Red Hat Enterprise Linux могут быть развернуты с использованием так называемого сервера *Satellite Server* — той же функции, которая есть у Red Hat CDN для управления и развертывания новых систем и обновлений. Системы RHEL могут быть настроены так, чтобы получать автоматические обновления программного обеспечения в определенное время со спутникового сервера. Наборы пакетов Errata могут быть быстро и автоматически развернуты в системах для исправления определенных проблем.
- **Образы контейнеров.** Вместо установки в системе отдельных RPM-пакетов можно упаковать множество пакетов в образ контейнера. Он подобен пакету RPM в том, что содержит набор программного обеспечения, но, в отличие от RPM, легче добавляется в систему, запускается и удаляется из нее, чем RPM.

Описание того, как использовать файлы `kickstart`, серверы Satellite Server, контейнеры и другие готовые к установке функции, не входит в книгу. Но для всех этих функций очень важно разбираться в YUM и RPM, чего мы пытались достичь в этой главе.

Резюме

Упаковка программного обеспечения в Fedora, Red Hat Enterprise Linux и связанных с ними системах обеспечивается с помощью программных пакетов, основанных на инструментах RPM Package Manager (RPM). Дистрибутивы Debian, Ubuntu и связанные с ними системы упаковывают программное обеспечение в файлы DEB. Вы можете опробовать простые в использовании инструменты с графическим интерфейсом, например Software (Центр приложений), для поиска и установки пакетов. Основные инструменты командной строки — команды `yum`, `dnf` и `rpm` для систем, связанных с Red Hat, а также `aptitude` `apt*`и `dpkg` для систем, связанных с Debian.

С помощью этих средств управления программным обеспечением можно устанавливать, запрашивать, проверять, обновлять и удалять пакеты. Вы также можете выполнять задачи обслуживания, такие как очистка файлов кэша и перестройка базы данных RPM. В этой главе описаны многие функции окна Software (Центр приложений), а также команды `yum`, `dnf` и `rpm`.

После установки системы и добавления необходимых пакетов программного обеспечения пришло время приступить к дальнейшей настройке систем Fedora, RHEL, Debian или Ubuntu. Если системой будут пользоваться несколько человек, то очередной задачей будет умение управлять учетными записями. В следующей главе описывается управление учетными записями в Fedora, RHEL и других системах Linux.

Упражнения

Эти упражнения позволяют проверить знание работы с программными пакетами RPM в Fedora или Red Hat Enterprise Linux. Для выполнения данных упражнений я рекомендую использовать систему Fedora с подключенным Интернетом. (Большинство упражнений сработает и в зарегистрированной системе RHEL.)

У вас должен быть доступ к репозиториям Fedora, которые настраиваются автоматически. Если затрудняетесь с решением заданий, воспользуйтесь ответами к упражнениям, приведенными в приложении Б (хотя Linux позволяет решать задачи разными способами).

1. Найдите Yum-репозиторий для пакета, который обеспечивает команду `mogrify`.
2. Отобразите информацию о пакете, содержащем команду `mogrify`, и определите домашнюю страницу для него (URL).
3. Установите пакет, содержащий команду `mogrify`.
4. Список всех файлов с документацией, находящихся в пакете, содержащем команду `mogrify`.

5. С помощью команды `changelog` просмотрите изменения пакета, содержащего команду `mogrify`.
6. Удалите команду `mogrify` из системы и проверьте ее пакет по базе данных RPM, чтобы убедиться, что команда действительно удалена.
7. Переустановите пакет, содержащий команду `mogrify`, и убедитесь, что он не поврежден.
8. Загрузите пакет, содержащий команду `mogrify`, в текущий каталог.
9. Отобразите общую информацию о только что загруженном пакете, запросив файл RPM-пакета в текущем каталоге.
10. Удалите пакет, содержащий команду `mogrify`, из своей системы.

11 Управление учетными записями

В этой главе

- Работа с учетными записями пользователей.
- Работа с учетными записями групп.
- Настройка централизованных учетных записей пользователей.

Добавление пользователей и управление ими — это обычные задачи системных администраторов Linux. *Учетные записи пользователей* поддерживают границы между пользователями системы и процессами, выполняемыми в системах. *Группы* — это способ назначения прав в системе, которые могут быть определены для нескольких пользователей одновременно.

В этой главе описывается создание не только нового пользователя, но и предопределенных параметров и файлов для настройки среды пользователя. С помощью инструментов, например команд `useradd` и `usermod`, можно назначить такие параметры, как расположение домашнего каталога, оболочка по умолчанию, группа по умолчанию, а также определенные значения идентификаторов (ID) пользователей и групп. С помощью Cockpit вы можете добавлять учетные записи пользователей и управлять ими через веб-интерфейс.

Создание учетных записей пользователя

Каждый пользователь системы Linux должен иметь отдельную учетную запись. В учетной записи пользователя можно безопасно хранить файлы, а также средства настройки пользовательского интерфейса (GUI, путь, переменные окружения и т. д.), соответствующие тому, как используется компьютер.

В большинстве систем Linux учетные записи пользователей можно добавлять несколькими способами. Системы Fedora и Red Hat Enterprise Linux предлагают

программу Cockpit, которая позволяет создавать учетные записи и управлять ими. Если программа Cockpit еще не установлена и не включена, сделайте это следующим образом:

```
# yum install cockpit -y
# systemctl enable --now cockpit.socket
```

Чтобы создать учетную запись пользователя через Cockpit, выполните следующие действия.

1. Откройте интерфейс Cockpit из своего браузера (`имя хоста:9090`).
2. Войдите в систему как суперпользователь или пользователь с правами root, установите флажок Reuse my password for privileged tasks (Использовать мой пароль для привилегированных задач), чтобы получить возможность запускать некоторые команды от имени суперпользователя, и выберите раздел Accounts (Учетные записи).
3. Нажмите кнопку Create New Account (Создать учетную запись).

   На рис. 11.1 показан пример всплывающего окна Create New Account (Создать учетную запись).

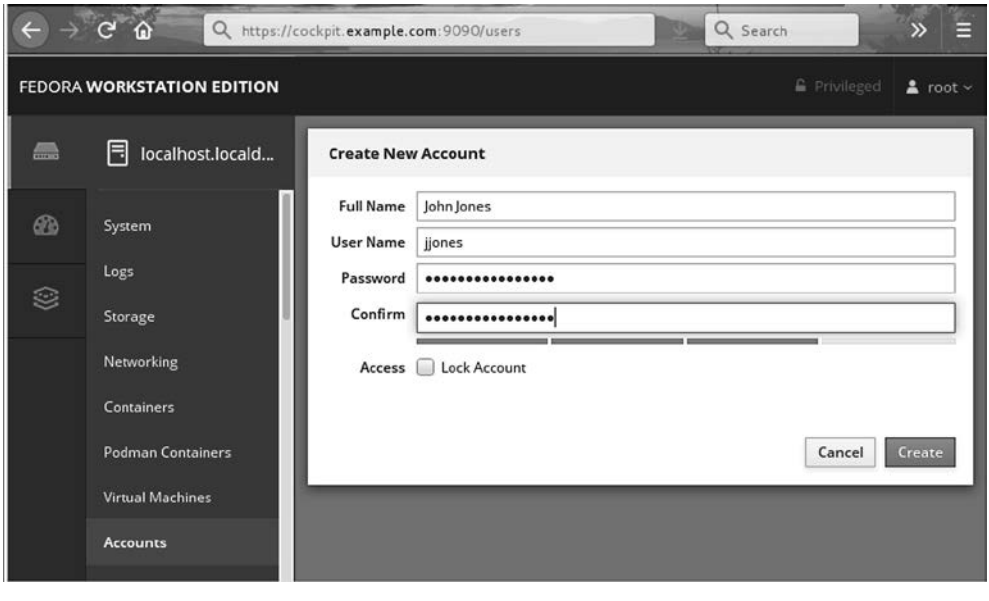


Рис. 11.1. Добавление и изменение учетных записей пользователей в Cockpit

4. Начните процесс добавления новой учетной записи пользователя в систему Linux. Рассмотрим поля, которые вам нужно заполнить.
   - Full Name (Полное имя). Используйте настоящее имя пользователя с прописными и строчными буквами, как в реальной жизни. Технически эта

информация хранится в комментарии к файлу `passwd`, и в связи с соглашением ожидается, что это поле будет содержать полное имя каждого пользователя.

- User Name (Имя пользователя). Это имя используется для входа в систему как данный пользователь. Выбирая имя пользователя, не начинайте его с цифры (например, 26jsmith). Кроме того, лучше применять только строчные буквы, никаких управляющих символов или пробелов и максимум восемь символов. Команда `useradd` допускает максимум 32 символа в имени, но часть приложений не могут работать с такими длинными именами. Некоторые инструменты, например `ps`, отображают идентификаторы пользователей (UID) вместо имен, если те слишком длинные. Имена пользователей Jsmith и jsmith могут привести к путанице с программами, такими как sendmail, которые не различают регистр.
- Password (Пароль), Confirm (Подтверждение). Введите пароль пользователя в поле Password (Пароль) и повторите его в поле Confirm (Подтверждение). Пароль должен состоять не менее чем из восьми символов и содержать прописные и строчные буквы, цифры и знаки препинания. В нем не должно быть реальных слов, повторяющихся букв или букв по порядку из строки на клавиатуре. С помощью этого интерфейса нужно установить пароль, который соответствует указанным критериям. (Если вы хотите добавить не соответствующий критериям пароль, примените команду `useradd`, которая будет описана далее в этой главе.) Полоски под полями паролей, сначала красные, становятся зелеными по мере повышения надежности пароля.
- Access (Доступ). Чтобы создать учетную запись, которую еще нельзя использовать, установите флажок Lock Account (Заблокировать учетную запись). В таком случае никто не сможет войти в данную учетную запись до тех пор, пока вы не снимете флажок или не измените эту информацию в файле `passwd`.

5. Нажмите кнопку Create (Создать), чтобы добавить пользователя в систему. Запись о новой учетной записи пользователя добавляется в файл `/etc/passwd`, а о новой учетной записи группы — в файл `/etc/group` (будут описаны позже в этой главе).

В Cockpit в окне Accounts (Учетные записи) можно изменить сведения об обычном пользователе после создания его учетной записи. Чтобы изменить эту информацию позже, выполните следующие действия.

1. Выберите учетную запись пользователя. Появится окно с возможными для него изменениями.
2. Имя пользователя нельзя изменить, его можно только удалить. А следующую информацию изменить можно.
   - Full Name (Полное имя). Поскольку полное имя пользователя — это всего лишь комментарий к файлу, его можно изменять по своему усмотрению.

- Roles (Роли). По умолчанию можно установить флажки Server Administrator (Администратор сервера), предоставляя пользователю права суперпользователя и добавляя его в группу wheel, и Image Builder (Создатель образов), позволяя пользователю создавать контейнеры и другие типы изображений через группу weldr. Другие роли могут быть добавлены в этот список с помощью иных компонентов Cockpit. Если пользователь уже в системе, он должен выйти из нее, чтобы получить эти права.
- Access (Доступ). Вы можете заблокировать учетную запись, установив флажок в поле Lock Account (Заблокировать учетную запись). Можно также задать срок действия учетной записи — определенную дату или же оставить учетную запись без срока действий и никогда не блокировать.
- Password (Пароль). Выберите пункт Set Password (Задать пароль), чтобы установить новый пароль для этого пользователя, или Force Change (Принудительно изменить), чтобы заставить пользователя изменить свой пароль при следующем входе в систему. По умолчанию срок действия паролей никогда не истекает. Это можно изменить и установить запрос на изменение пароля через заданное количество дней.
- Authorized Public SSH Keys (Авторизированные открытые SSH-ключи). Если у вас есть открытый SSH-ключ для пользователя, можете выбрать знак плюс (+), вставить этот ключ в текстовое поле и нажать кнопку Add (Добавить). Пользователь с соответствующим закрытым ключом с помощью него может войти в учетную запись пользователя через SSH, не вводя пароль.

3. Изменения вступают в силу сразу же, поэтому можете просто выйти из окна, когда закончите изменять учетную запись пользователя.

Раздел Accounts (Учетные записи) в Cockpit был разработан для упрощения процесса создания и изменения учетных записей пользователей. Дополнительные функции, связанные с учетными записями пользователей, можно добавлять или изменять из командной строки. Далее в этой главе описывается, как добавить учетные записи пользователей из командной строки с помощью `useradd` и изменить их командой `usermod`.

Добавление пользователей с помощью команды useradd

Иногда в системе Linux нет настольного инструмента или веб-интерфейса, с помощью которых можно добавлять пользователей. А иногда может показаться более удобным добавить сразу несколько пользователей с помощью скрипта оболочки или изменить функции учетной записи пользователя, которых нет в Cockpit. Для всего этого существуют специальные команды, позволяющие добавлять и изменять учетные записи пользователей из командной строки.

Самый простой способ создания нового пользователя из оболочки — команда `useradd`. Просто откройте окно Terminal (Терминал) с правами суперпользователя,

наберите useradd в командной строке и добавьте подробную информацию о новой учетной записи в качестве параметров.

Единственным обязательным параметром является логин пользователя, но вы, вероятно, захотите включить в него дополнительные сведения. Каждому элементу информации об учетной записи предшествует однобуквенный параметр с дефисом перед ним. Команда useradd работает со следующими параметрами.

- -c "*комментарий*" — комментарий к новой учетной записи пользователя. Обычно это полное имя пользователя. Замените *комментарий* именем учетной записи пользователя (-c Jake). Используйте кавычки для ввода нескольких слов (например, -c "Jake Jackson").
- -d *домашний_каталог* — устанавливает домашний каталог, который будет применяться для учетной записи. По умолчанию его название должно совпадать с именем учетной записи и располагаться в каталоге /home. Замените *домашний_каталог* именем нужного каталога, например -d /mnt/homes/jake.
- -D — вместо того чтобы создавать новую учетную запись, сохраняет предоставленную информацию в качестве новых параметров по умолчанию для всех вновь созданных учетных записей.
- -e *дата_устаревания* — назначает дату, когда учетная запись пользователя будет заблокирована, в формате ГГГГ-ММ-ДД. Замените *дата_устаревания* датой, например, для срока действия учетной записи до 5 мая 2022 года — -e 2022-05-05.
- -f -1 — устанавливает количество дней, которые должны пройти с момента устаревания пароля до блокирования учетной записи. По умолчанию значение -1 отключает этот параметр. Значение 0 блокирует учетную запись сразу же после истечения срока действия пароля. Замените -1 нужным числом.
- -g *группа* — устанавливает основную группу (она уже должна существовать в файле /etc/group), в которой будет находиться новый пользователь. Замените *группа* именем группы, например -g wheel. Без этого параметра создается новая группа, которая совпадает с именем пользователя и становится основной.
- -G *список_групп* — добавляет нового пользователя в предоставленный список дополнительных групп, разделенных запятыми (например, -G wheel,sales,tech,lunch). Если хотите использовать параметр -G позже с usermod, обязательно укажите -aG. Если ввести просто -G, существующие дополнительные группы будут удалены и представленные группы окажутся единственными назначенными.
- -k *skel_dir* — устанавливает каталог с шаблонами, который содержит файлы и каталоги для копирования в домашний каталог пользователя при его создании. Этот параметр можно использовать только в сочетании с параметром -m. Замените *skel_dir* именем нужного каталога. (Без этого параметра применяется каталог /etc/skel.)
- -m — автоматически создает домашний каталог пользователя и копирует в него файлы из каталога шаблонов (/etc/skel). (Это происходит по умолчанию для

Fedora и RHEL, поэтому действие не требуется. А в Ubuntu необходимо его выполнить.)

- `-M` — не создает домашний каталог нового пользователя, даже если для его создания задано поведение по умолчанию.
- `-n` — отключает поведение по умолчанию при создании новой группы, соответствующей имени и идентификатору нового пользователя. Этот параметр доступен в системах Fedora и RHEL. Другие системы Linux назначают нового пользователя в группу с тем же именем.
- `-o` — использует `-u uid` для создания учетной записи пользователя, которая имеет тот же UID, что и у другого пользователя. (Фактически это позволяет иметь два разных имени пользователя с правами на один и тот же набор файлов и каталогов.)
- `-p` *пароль* — добавляет пароль для новой учетной записи. Он должен быть зашифрованным. Можно добавить пароль не сразу, а применить команду *пароль пользователя* позже, чтобы задать его для пользователя. (Чтобы сгенерировать зашифрованный пароль MD5, введите команду `openssl` *пароль*.)
- `-s` *оболочка* — указывает командную оболочку, которая будет использоваться для этой учетной записи. Замените *оболочка* нужной командной оболочкой, например `-s /bin/csh`.
- `-u` *id_пользователя* — укажите идентификатор (ID) пользователя для учетной записи, например `-u 1793`. Без параметра `-u` будет реализовано поведение по умолчанию — автоматически назначен следующий доступный номер. Замените *id_пользователя* идентификатором пользователя. Автоматические идентификаторы для обычных пользователей начинаются с 1000, поэтому для таких пользователей нужно вводить идентификаторы, превышающие это число, чтобы не пересекаться с автоматическими назначениями.

Создадим учетную запись для нового пользователя. Его полное имя — Sara Green, а логин — `sara`. Для начала станьте суперпользователем и введите следующую команду:

```
# useradd -c "Sara Green" sara
```

Далее установите пароль, применив команду `passwd`. Его нужно ввести дважды:

```
# passwd sara
Changing password for user sara.
New password: **********
Retype new password: **********
```

ПРИМЕЧАНИЕ

Звездочки в примере — это пароль, который вы вводите. В действительности при вводе пароля звездочек не будет. Кроме того, имейте в виду, что запуск команды passwd от имени суперпользователя позволяет добавлять короткие или пустые пароли, которые обычные пользователи не могут добавить сами.

При создании учетной записи для sara команда useradd выполняет несколько действий.

- Считывает файлы /etc/login.defs и /etc/default/useradd, чтобы найти значения по умолчанию для использования при создании учетных записей.
- Проверяет параметры командной строки, чтобы узнать, какие значения по умолчанию следует переопределить.
- Создает новую запись пользователя в файлах /etc/passwd и /etc/shadow на основе значений по умолчанию и параметров командной строки.
- Создает все новые записи группы в файле /etc/group. (Fedora создает группу, используя имя нового пользователя.)
- Создает домашний каталог на основе имени пользователя в каталоге /home.
- Копирует все файлы, находящиеся в каталоге /etc/skel, в новый домашний каталог. Обычно это сценарии входа в систему и запуска приложений.

В предыдущем примере применены лишь несколько доступных параметров useradd. Большинство параметров учетной записи назначаются с использованием значений по умолчанию. При желании можно установить свои значения параметров.

Вот пример с несколькими вариантами:

```
# useradd -g users -G wheel,apache -s /bin/tcsh -c "Sara Green" sara
```

В этом случае useradd получает указание сделать users основной группой, к которой принадлежит sara (-g), добавляет ее в группы wheel и apache и назначает tcsh в качестве ее основной командной оболочки (-s). По умолчанию создается домашний каталог в /home под именем пользователя (/home/sara). Эта командная строка приводит к тому, что в файл /etc/passwd добавляется следующая строка:

```
sara:x:1002:1007:Sara Green:/home/sara:/bin/tcsh
```

Каждая строка в файле /etc/passwd представляет собой одну учетную запись пользователя. Каждое поле отделяется от следующего двоеточием (:). Положение поля в строке определяет его значение. Как видно в примере, первым идет имя пользователя. В поле с паролем содержится элемент x, поскольку здесь файл пароля shadow применяется для хранения зашифрованного пароля (в файле /etc/shadow).

Идентификатор (ID) пользователя выбран автоматически и равен 1002. Основной идентификатор группы — 1007, что соответствует частной группе в файле /etc/group. В поле комментария правильно введено полное имя пользователя Sara Green, домашний каталог был автоматически назначен как /home/sara, а командная оболочка — это /bin/tcsh, как и указано в параметрах useradd.

Многие параметры опускаются, как я сделал в первом примере с useradd, и в большинстве случаев значения устанавливаются по умолчанию. Например, если не использовать параметры -g sales или -G wheel, apache, имя группы будет назначено новому пользователю. Некоторые системы Linux (кроме Fedora и RHEL)

назначают имя группы по умолчанию. Аналогично исключение `-s /bin/tcsh` приводит к назначению `/bin/bash` в качестве оболочки по умолчанию.

Файл `/etc/group` содержит информацию о различных группах в системе Linux и пользователях в них. Группы позволяют нескольким пользователям совместно получать доступ к одним и тем же файлам. Так выглядит запись в файле `/etc/group`, созданная для `sara`:

```
sara:x:1007:
```

Каждая строка в файле группы содержит имя группы, ее пароль (обычно заполненный символом `x`), ее идентификационный номер и список пользователей, относящихся к этой группе. По умолчанию каждый пользователь добавляется в собственную группу, начиная с 1000 и со следующего доступного GID.

Пользовательские настройки по умолчанию

Команда `useradd` определяет значения по умолчанию для новых учетных записей, считывая файлы `/etc/login.defs` и `/etc/default/useradd`. Эти значения можно изменить, отредактировав файлы вручную с помощью стандартного текстового редактора. Хотя файл `login.def` в разных системах Linux различается, далее приведен пример, содержащий настройки, которые можно найти в `login.defs`:

```
PASS_MAX_DAYS     99999
PASS_MIN_DAYS     0
PASS_MIN_LEN      5
PASS_WARN_AGE     7
UID_MIN                  1000
UID_MAX                 60000
SYS_UID_MIN               200
SYS_UID_MAX               999
GID_MIN                  1000
GID_MAX                 60000
SYS_GID_MIN               201
SYS_GID_MAX               999
CREATE_HOME yes
```

Все раскомментированные строки содержат пару «ключевое слово — значение». Например, за ключевым словом `PASS_MIN_LEN` следует пустое пространство и значение `5`. Оно сообщает `useradd`, что пароль пользователя должен быть не менее пяти символов. Другие строки позволяют настроить диапазон автоматически назначаемых идентификаторов пользователей или групп. (Fedora начинает отсчет с UID 1000, более ранние системы начинали с UID 100.) Постоянные административные номера учетных записей пользователей и групп заняты для 199 и 200 человек соответственно. Таким образом, вы можете назначить собственные административные учетные записи с 200 и 201 соответственно вплоть до 999.

Перед каждым ключевым словом в поле с комментарием объясняется его значение (в примере я его отредактировал, чтобы занимало меньше места). Из-

менить значение по умолчанию так же просто, как отредактировать значение, связанное с ключевым словом, и сохранить его в файле перед запуском команды useradd.

Если вы хотите просмотреть другие настройки по умолчанию, перейдите в файл /etc/default/useradd. Можно также просмотреть настройки по умолчанию, введя команду useradd с параметром -D, как в примере:

```
# useradd -D
GROUP=100
HOME=/home
INACTIVE=-1
EXPIRE=
SHELL=/bin/bash
SKEL=/etc/skel
CREATE_MAIL_SPOOL=yes
```

Вы можете использовать параметр -D для изменения значений по умолчанию. При запуске с ним useradd не создает сразу новую учетную запись — вместо этого он сохраняет все дополнительные параметры в качестве новых значений по умолчанию в файле /etc/default/useradd. Не все параметры useradd можно применять с параметром -D, только следующие пять:

- -b *домашний_каталог* — задает каталог по умолчанию, в котором создаются домашние каталоги пользователей. Замените *домашний_каталог* именем каталога, например -b /garage. Обычно это каталог /home;
- -e *дата_устаревания* — устанавливает дату срока действия по умолчанию, когда учетная запись пользователя будет заблокирована. Замените значение *дата_устаревания* на дату в формате ГГГГ-ММ-ДД, например -e 2021-10-17;
- -f *дни* — устанавливает количество дней после истечения срока действия пароля до блокирования учетной записи. Замените *дни* числом, представляющим количество дней, например -f 7;
- -g *группа* — задает группу по умолчанию, в которую помещаются новые пользователи. Обычно useradd создает новую группу с теми же именем и идентификационным номером, что и у пользователя. Замените *группа* нужным именем группы, например -g bears;
- -s *оболочка* — устанавливает оболочку по умолчанию для новых пользователей. Обычно это /bin/bash. Замените *оболочка* полным путем к оболочке, который вы хотите использовать по умолчанию для новых пользователей, например -s /bin/ash.

Чтобы установить любое из значений по умолчанию, сначала задайте параметр -D, а затем добавьте значения по умолчанию. Например, чтобы установить расположение домашнего каталога по умолчанию в /home/everyone и оболочку по умолчанию в /bin/tcsh, введите следующее:

```
# useradd -D -b /home/everyone -s /bin/tcsh
```

В дополнение к определению пользовательских настроек администратор может создавать файлы по умолчанию, которые копируются в домашний каталог каждого пользователя. Эти файлы могут включать сценарии входа в систему и файлы конфигурации оболочки, например файл `.bashrc`. Имейте в виду, что по умолчанию настройкой этих типов файлов занимается каталог `/etc/skel`.

Существуют и другие полезные команды для работы с учетными записями пользователей — `usermod` для изменения настроек существующей учетной записи и `userdel` для удаления существующей учетной записи пользователя.

Изменение пользователей с помощью команды usermod

Команда `usermod` довольно проста, понятна и позволяет изменять параметры учетной записи. Для нее доступны многие параметры из тех, что используются с `useradd`, например:

- `-c` *`имя_пользователя`* — изменяет описание, связанное с учетной записью пользователя. Замените *`имя_пользователя`* именем учетной записи пользователя (`-c jake`). Применяйте кавычки для ввода нескольких слов, например `-c "Jake Jackson"`;
- `-d` *`домашний_каталог`* — изменяет домашний каталог, который будет использоваться для учетной записи. По умолчанию он должен носить имя пользователя и размещаться в `/home`. Замените *`домашний_каталог`* именем нужного каталога, например `-d /mnt/homes/jake`;
- `-e` *`дата_устаревания`* — назначает новую дату истечения срока действия для учетной записи в формате ГГГГ-ММ-ДД. Замените *`дата_устаревания`* нужной датой, например `-e 2022-10-15` для 15 октября 2022 года;
- `-f -1` — устанавливает количество дней, которое должно пройти после истечения срока действия пароля, прежде чем учетная запись будет заблокирована. По умолчанию значение `-1` отключает этот параметр. Значение `0` блокирует учетную запись сразу же по истечении срока действия пароля. Замените `-1` нужным значением;
- `-g` *`группа`* — изменяет основную группу (как указано в файле `/etc/group`), в которой будет находиться пользователь. Замените *`группа`* именем группы, например `-g wheel`;
- `-G` *`список_групп`* — добавляет дополнительные группы пользователя в предоставленный список групп, разделенных запятыми. Если пользователь уже находится по крайней мере в одной группе, кроме личной, нужно добавить параметр `-a` (`-Ga`). Если так не сделать, то пользователь будет назначен в новую группу, выйдя из всех предыдущих;
- `-l` *`новое_имя`* — изменяет имя учетной записи;

- `-L` — блокирует учетную запись, поставив восклицательный знак в начале зашифрованного пароля в файле `/etc/shadow`. Данное действие блокирует учетную запись, при этом не трогая пароль (параметр `-U` разблокирует ее);
- `-m` — доступен только совместно с параметром `-d`. Копирует содержимое домашнего каталога пользователя в новый каталог;
- `-o` — работает только с `-u uid` и снимает ограничения на уникальность идентификаторов (UID);
- `-s` *`оболочка`* — указывает другую командную оболочку, которая будет использоваться для этой учетной записи. Замените *`оболочка`* именем нужной командной оболочки, например `-s bash`;
- `-u` *`id_пользователя`* — изменяет идентификационный номер пользователя для учетной записи. Замените *`id_пользователя`* нужным идентификационным номером, например `-u 1474`;
- `-U` — снимает блокировку с учетной записи пользователя, удалив восклицательный знак в начале зашифрованного пароля.

Далее приведены примеры команды `usermod`:

```
# usermod -s /bin/csh chris
# usermod -Ga sales,marketing, chris
```

В первом примере оболочка заменяется оболочкой `csh` для пользователя с именем `chris`. Во втором примере пользователю `chris` назначают дополнительные группы. Параметр `-a` (`-Ga`) гарантирует, что дополнительные группы будут добавлены к уже существующим группам для пользователя `chris`. Если параметр `-a` не используется, существующие дополнительные группы для `chris` удаляются, а новый список групп включает только дополнительные группы, назначенные ему.

Удаление пользователей с помощью команды userdel

Команда `usermod` используется для изменения параметров пользователя, `useradd` — для создания учетных записей, а `userdel` — для удаления пользователей. Команда, приведенная далее, удаляет пользователя `chris`:

```
# userdel -r chris
```

Здесь пользователь `chris` удаляется из файла `/etc/password`. Параметр `-r` удаляет также домашний каталог пользователя. Если не применять параметр `-r`, как показано далее, домашний каталог для `chris` не будет удален:

```
# userdel chris
```

Имейте в виду, что простое удаление учетной записи пользователя ничего не меняет в файлах, которые пользователь оставляет в системе (за исключением

удаляемых в ходе работы с параметром `-r`). Однако при запуске `ls -l` для этих файлов права собственности принадлежат идентификатору предыдущего владельца.

Прежде чем удалить пользователя, запустите команду `find`, чтобы найти все его файлы. После удаления пользователя можно выполнить поиск по его идентификатору, чтобы найти оставшиеся файлы. Два варианта поиска:

```
# find / -user chris -ls
# find / -uid 504 -ls
```

Поскольку файлы, не привязанные к какому-либо имени пользователя, считаются небезопасными, рекомендуется найти их и назначить реальной учетной записи. Вот пример команды `find`, которая находит все файлы в файловой системе, не связанные ни с одним пользователем (файлы перечислены по UID):

```
# find / -nouser -ls
```

Учетные записи групп

Учетные записи групп полезны, если нужно поделиться набором файлов с несколькими пользователями. Можно создать группу и изменять набор связанных с ней файлов. Суперпользователь может назначить в эту группу пользователей, чтобы они получали доступ к файлам на основе прав группы. Рассмотрим пример файла и каталога:

```
$ ls -ld /var/salesdocs /var/salesdocs/file.txt
drwxrwxr-x. 2 root sales 4096 Jan 14 09:32 /var/salesstuff/
-rw-rw-r--. 1 root sales    0 Jan 14 09:32 /var/salesstuff/file.txt
```

Второй набор прав `rwx` для каталога `/var/salesdocs` (`rwxrwxr-x`) показывает, что любой член группы (`sales`) имеет разрешение на чтение файлов в этом каталоге (`r` — чтение), создание файлов и удаление их из этого каталога (`w` — запись) и внесение изменений в этот каталог (`x` — исполнение). Файл `file.txt` может быть прочитан и изменен членами группы `sales` на основе второго набора прав `rwx`.

Использование учетных записей групп

Каждый пользователь назначается в основную группу. В Fedora и RHEL по умолчанию эта группа является новой и носит имя пользователя: если имя пользователя `sara`, то группа, назначенная ему, также называется `sara`. Первичная группа обозначается номером в третьем поле каждой записи в файле `/etc/passwd`. Например, здесь применяется идентификатор группы `1007`:

```
sara:x:1002:1007:Sara Green:/home/sara:/bin/tcsh
```

Эта строка указывает на запись в файле `/etc/group`:

```
sara:x:1007:
```

Для примера обратимся к учетным записям пользователей и групп. Факты об использовании групп таковы.

- Когда пользователь `sara` создает файл или каталог, по умолчанию они назначаются основной группе `sara`, также называемой `sara`.
- Пользователь `sara` может принадлежать к нескольким дополнительным группам или не принадлежать ни к одной. Если бы пользователь `sara` был членом групп `sales` и `marketing`, записи об этом в файле `/etc/group` могли бы выглядеть следующим образом:

```
sales:x:1302:joe,bill,sally,sara
marketing:x:1303:mike,terry,sara
```

- Пользователь не может добавить себя в дополнительную группу, как и включить другого пользователя в свою. Назначать пользователей в группы может только обладающий правами суперпользователя.
- Любой файл, назначенный группе `sales` или `marketing`, доступен пользователю `sara` с правами, установленными для группы (в зависимости от того, у какого из пользователей наибольший доступ). Если пользователь хочет создать файл с назначенными ему группами `sales` или `marketing`, то он должен использовать команду `newgrp`. В этом примере `sara` задействует команду `newgrp`, чтобы группа `sales` временно стала основной, и создает файл:

```
[sara]$ touch file1
[sara]$ newgrp sales
[sara]$ touch file2
[sara]$ ls -l file*
-rw-rw-r--. 1 sara sara  0 Jan 18 22:22 file1
-rw-rw-r--. 1 sara sales 0 Jan 18 22:23 file2
[sara]$ exit
```

Кроме того, с помощью команды `newgrp` можно разрешить пользователям временно становиться членами другой группы. Для этого кто-то с правами суперпользователя может применить команду `gpasswd` и установить пароль группы, например `gpasswd sales`. После этого любой пользователь может ввести команду `newgrp sales` в оболочке и временно использовать группу `sales` в качестве своей основной, просто введя ее пароль при появлении запроса.

Создание учетных записей групп

Как суперпользователь, вы можете создавать новые группы из командной строки с помощью команды `groupadd`. Кроме того, как отмечалось ранее, группы образуются автоматически при создании учетной записи пользователя.

Идентификаторы групп от 0 до 999 присваиваются специальным административным группам. Например, у root-группы номер GID 0. В Red Hat Enterprise Linux и Fedora группы обычных пользователей носят номера начиная с 1000. В первых UNIX-системах присваивалось значение GID от 0 до 99. Сейчас в других системах

Linux административным группам присваивается значение GID от 0 до 500. Относительно новая функция, описанная ранее, резервирует административные учетные записи пользователей и групп до 199 и 200 соответственно и позволяет создавать собственные административные учетные записи между этими номерами и 999.

Вот несколько примеров создания учетной записи группы с помощью команды `groupadd`:

```
# groupadd kings
# groupadd -g 1325 jokers
```

В приведенных примерах группе с именем `kings` присваивается следующий доступный идентификатор группы. После этого группа `jokers` создается с идентификатором `1325`. Некоторым администраторам нравится брать неопределенный номер группы между 200 и 1000, чтобы созданная ими группа не смешивалась с группами с номерами выше 1000 (номера UID и GID могут использоваться параллельно).

Чтобы изменить группу позже, возьмите команду `groupmod`, как в следующем примере:

```
# groupmod -g 330 jokers
# groupmod -n jacks jokers
```

В первом примере идентификатор группы `jokers` изменен на `330`. Во втором случае название `jokers` меняется на `jacks`. Если вы затем хотите назначить пользователю какую-либо из групп в качестве дополнительной, возьмите команду `usermod`, как описано ранее в этой главе.

Управление пользователями в корпоративной среде

Основной метод Linux для работы с учетными записями пользователей и групп не изменился с тех пор, как были разработаны первые системы UNIX десятилетия назад. Однако по мере того, как способы использования систем Linux стали усложняться, функции управления пользователями, группами и связанными с ними разрешениями были добавлены в базовую модель пользователя/группы, чтобы она могла стать:

- *более гибкой*. В базовой модели каждому файлу можно назначить только одного пользователя и одну группу. Кроме того, обычные пользователи не имеют возможности назначать права другим пользователям или группам и совсем незначительно могут настроить совместные файлы/каталоги. Модель усовершенствована, и теперь обычные пользователи могут создавать специальные каталоги для совместной работы, применив функции `sticky bit` и `set GID`. Используя список контроля доступа (access control lists, ACL), любой пользователь может назначить определенные права на файлы и каталоги любым пользователям и группам;

- *более централизованной.* Если компьютер один, хранение информации обо всех пользователях в файле `/etc/passwd`, вероятно, не составляет труда. Но если нужно аутентифицировать один и тот же набор пользователей в тысячах систем Linux, централизация этой информации может сэкономить много времени и душевных сил. Red Hat Enterprise Linux включает в себя функции, позволяющие аутентифицировать пользователей с серверов LDAP или Microsoft Active Directory.

В следующем подразделе описывается, как использовать такие функции, как списки контроля доступа и общие каталоги (`sticky bit` и `set GID`), чтобы обеспечить управление обменом файлами и каталогами.

Настройка прав с помощью списка ACL

Функция *«список контроля доступа» (ACL)* была создана для того, чтобы обычные пользователи могли выборочно делиться своими файлами и каталогами с другими пользователями и группами. С помощью ACL пользователь может разрешить другим пользователям читать, записывать и исполнять файлы и каталоги, не открывая элементы файловой системы для всех и не требуя от суперпользователя изменить назначенных ему пользователя или группу.

Вот что нужно знать о списках ACL.

- Списки ACL должны быть включены в файловую систему при ее монтировании.
- В Fedora и Red Hat Enterprise Linux списки ACL автоматически включаются в любую файловую систему, созданную при установке системы.
- Если вы создаете файловую систему после установки (например, при добавлении жесткого диска), то должны убедиться, что параметр `acl` применяется при монтировании файловой системы (подробнее об этом позже).
- Чтобы добавить списки в файл, можно использовать команду `setfacl`, а чтобы просматривать списки ACL для файла — команду `getfacl`.
- Чтобы установить списки ACL для любого файла или каталога, вы должны быть его фактическим владельцем (назначенным ему пользователем). Другими словами, назначение прав пользователя или группы с помощью `setfacl` не дает вам права самостоятельно изменять списки ACL для этих файлов.
- Поскольку файлу/каталогу могут быть назначены несколько пользователей и групп, фактические права пользователя основаны на объединении всех назначений прав пользователям/группам, к которым он принадлежит. Например, если файл имеет разрешение только на чтение (`r--`) для группы `sales` и на чтение/запись/исполнение (`rwx`) для группы `marketing`, а пользователь `mary` принадлежит к обеим группам, то у `mary` будут права `rwx`.

ПРИМЕЧАНИЕ

Если списки ACL не включены в файловую систему, см. пункт «Включение списков ACL» далее в этом подразделе, чтобы узнать, как монтировать файловую систему с включенными списками ACL.

Настройка списков ACL с помощью команды setfacl

С помощью команды `setfacl` в списках ACL можно изменять права (`-m`) или удалить их (`-x`). Далее приведен пример синтаксиса команды `setfacl`:

```
setfacl -m u: имя_пользователя :rwx имя_файла
```

Здесь за параметром `modify` (`-m`) следует буква `u`, обозначающая, что вы устанавливаете права ACL для пользователя. После двоеточия (`:`) указываются имя пользователя, затем еще одно двоеточие и права, которые нужно назначить. Как и в случае с командой `chmod`, вы можете назначить пользователю или группе права на чтение (`r`), запись (`w`) и/или исполнение (`x`) (в данном примере даются полные права `rwx`). Последний аргумент заменяется фактическим именем файла.

Далее приведены примеры применения команды `setfacl`, где пользователь `mary` добавляет в файл права для других пользователей и групп:

```
[mary]$ touch /tmp/memo.txt
[mary]$ ls -l /tmp/memo.txt
-rw-rw-r--. 1 mary mary 0 Jan 21 09:27 /tmp/memo.txt
[mary]$ setfacl -m u:bill:rw /tmp/memo.txt
[mary]$ setfacl -m g:sales:rw /tmp/memo.txt
```

Здесь пользователь `mary` создал файл с именем `/tmp/memo.txt`. С помощью команды `setfacl` он изменил права (`-m`) для пользователя `bill`, так что теперь у него есть права на чтение/запись (`rw`) этого файла. Затем пользователь `mary` изменил права для группы `sales`, чтобы любой ее член также имел права на чтение и запись. Теперь посмотрите на вывод `ls -l` и `getfacl` в этом файле:

```
[mary]$ ls -l /tmp/memo.txt
-rw-rw-r--+ 1 mary mary 0 Jan 21 09:27 /tmp/memo.txt
[mary]$ getfacl /tmp/memo.txt
# file: tmp/memo.txt
# owner: mary
# group: mary
user::rwuser:
bill:rwgroup::
rwgroup:
sales:rwmask::
rwother::
r--
```

Обратите внимание на знак плюс (`+`) в выводе `rw-rw-r--+` после `ls -l`. Он указывает на то, что ACL установлены в файле, поэтому нужно запустить команду `getfacl`, чтобы увидеть, какие именно списки ACL там есть. В выводе `mary` — это и владелец, и группа (так же как после `ls -l`), права обычного пользователя — это `rw-`, а права в ACL для пользователя `bill` — это `rw-`. То же самое верно для прав для групп, в том числе группы `sales`. Другие права — это `r--`.

Строку `mask` (ближе к концу в предыдущем примере `getfacl`) нужно обсудить отдельно. Как только вы устанавливаете списки ACL для файла, для прав обычных

групп в файле задается маска максимальных привилегий, которые пользователь или группа ACL могут иметь для файла. Таким образом, даже если вы предоставляете пользователю больше прав ACL, чем позволяют права группы, его права не будут превышать прав группы, как в следующем примере:

```
[mary]$ chmod 644 /tmp/memo.txt
[mary]$ getfacl /tmp/memo.txt
# file: tmp/memo.txt
# owner: mary
# group: mary
user::rw-
user:bill:rw-   #effective:r--
group::rw-      #effective:r--
group:sales:rw- #effective:r--
mask::r--
other::r--
```

Обратите внимание на то, что, даже если пользователь `bill` и группа `sales` имеют права `rw-`, их действительные права — это `r--`. Таким образом, пользователь `bill` или кто-либо в группе `sales` не сможет изменить файл, если пользователь `mary` снова не откроет права, например набрав команду `chmod 664 /tmp/memo.txt`.

Настройка списков ACL по умолчанию

Настройка списков ACL по умолчанию в каталоге позволяет работать с ними в будущем. Это означает, что при создании новых файлов и каталогов в этом каталоге им будут назначаться одни и те же списки ACL. Чтобы назначить пользователю или группе права в ACL по умолчанию, добавьте параметр `d:` к пользователю или группе, например:

```
[mary]$ mkdir /tmp/mary
[mary]$ setfacl -m d:g:market:rwx /tmp/mary/
[mary]$ getfacl /tmp/mary/
# file: tmp/mary/
# owner: mary
# group: mary
user::rwx
group::rwx
other::r-x
default:user::rwx
default:group::rwx
default:group:sales:rwx
default:group:market:rwx
default:mask::rwx
default:other::r-x
```

Чтобы убедиться, что список ACL по умолчанию работает, создайте подкаталог. Затем запустите команду `getfacl`. Вы увидите, что строки по умолчанию

добавляются для пользователей, групп, масок и других объектов, которые берутся из списков ACL каталога:

```
[mary]$ mkdir /tmp/mary/test
[mary]$ getfacl /tmp/mary/test
# file: tmp/mary/test
# owner: mary
# group: mary
user::rwx
group::rwx
group:sales:rwx
group:market:rwx
mask::rwx
other::r-x
default:user::rwx
default:group::rwx
default:group:sales:rwx
default:group:market:rwx
default:mask::rwx
default:other::r-x
```

Обратите внимание на то, что при создании файла в этом каталоге передаваемые права различаются. Поскольку обычный файл создается без права `execute`, действующие права уменьшаются до `rw-`:

```
[mary@cnegus ~]$ touch /tmp/mary/file.txt
[mary@cnegus ~]$ getfacl /tmp/mary/file.txt
# file: tmp/mary/file.txt
# owner: mary
# group: mary
user::rw-
group::rwx          #effective:rw-
group:sales:rwx     #effective:rw-
group:market:rwx    #effective:rw-
mask::rw-
other::r--
```

Включение списков ACL

В современных системах Fedora и RHEL типы файловых систем xfs и ext (ext2, ext3 и ext4) создаются автоматически с поддержкой ACL. В других системах Linux или в файловых системах, созданных в них, для этого можно добавить параметр монтирования `acl` несколькими способами.

- Добавьте параметр `acl` в пятое поле строки файла `/etc/fstab`, который автоматически монтирует файловую систему при загрузке системы.
- Вставьте строку `acl` в поле `Default mount options` в суперблоке файловой системы, чтобы параметр `acl` использовался независимо от того, монтируется файловая система автоматически или вручную.

- Добавьте параметр `acl` в команду `mount` при монтировании файловой системы вручную.

Имейте в виду, что в системах Fedora и Red Hat Enterprise Linux вам нужно только добавить параметр монтирования `acl` к файловым системам, созданным в другом месте. Инсталлятор Anaconda автоматически добавляет поддержку ACL в каждую файловую систему, которую формирует во время установки, а инструмент `mkfs` добавляет `act` в каждую файловую систему, которую вы создаете с его помощью. Чтобы убедиться, что параметр `act` был добавлен в файловую систему `ext`, определите имя устройства, связанное с ней, и выполните команду `tune2fs -l` для просмотра имплантированных параметров монтирования, как в следующем примере:

```
# mount | grep home
/dev/mapper/mybox-home on /home type ext4 (rw)
# tune2fs -l /dev/mapper/mybox-home | grep "mount options"
Default mount options:    user_xattr acl
```

В примере я сначала набрал команду `mount`, чтобы увидеть список всех файловых систем, которые в данный момент монтируются, ограничив вывод словом `home` (потому что искал файловую систему, смонтированную на `/home`). Увидев имя устройства файловой системы, я взял его в качестве параметра для `tune2fs -l`, чтобы найти строку параметров монтирования по умолчанию. Далее я увидел, что параметры монтирования `user_xattr` (для расширенных атрибутов, таких как SELinux) и `act` имплантированы в суперблок файловой системы, чтобы их можно было использовать при монтировании файловой системы.

Если поле `Default mount options` пусто (например, когда файловая система только что создана), можно добавить параметр монтирования `act` с помощью команды `tune2fs -o`. Например, в другой системе Linux я создал файловую систему на съемном USB-накопителе, который был назначен в качестве устройства `/dev/sdc1`. Чтобы имплантировать параметр `acl` и проверить его наличие, я выполнил следующие команды:

```
# tune2fs -o acl /dev/sdc1
# tune2fs -l /dev/sdc1 | grep "mount options"
Default mount options:    acl
```

Чтобы проверить, что это сработало, перемонтируйте файловую систему и попробуйте использовать команду `getfacl` для файла в этой файловой системе.

Второй способ добавить поддержку `acl` в файловую систему — это ввести параметр `acl` в строку файла `/etc/fstab`, которая автоматически монтирует файловую систему во время загрузки. Далее приведен пример того, как будет выглядеть строка, которая монтирует файловую систему ext4, расположенную на устройстве `/dev/sdc1` в каталог `/var/stuff`:

```
/dev/sdc1     /var/stuff     ext4     acl     1 2
```

Вместо строки `defaults` в четвертом поле я указал параметр `acl`. Если в этом поле уже были заданы параметры, поставьте запятую после последнего параметра

и наберите `acl`. При следующем монтировании файловой системы включаются списки ACL. Если файловая система уже была смонтирована, я мог бы ввести следующую команду `mount` как суперпользователь, чтобы перемонтировать файловую систему с помощью `acl` или других значений, добавленных в файл `/etc/fstab`:

```
# mount -o remount /dev/sdc1
```

Третий способ добавить поддержку ACL в файловую систему — это монтировать файловую систему вручную и специально запрашивать параметр `acl`. Итак, если в файле `/etc/fstab` не было записи для файловой системы, то после создания точки монтирования (`/var/stuff`) введите следующую команду для монтирования файловой системы и включения поддержки ACL:

```
# mount -o acl /dev/sdc1 /var/stuff
```

Имейте в виду, что команда `mount` монтирует файловую систему временно. После перезагрузки системы файловая система не будет монтироваться, если только вы не добавите соответствующую строку в файл `/etc/fstab`.

Добавление каталогов для совместной работы пользователей

Специальный набор из трех битов прав обычно игнорируется при использовании команды `chmod` для изменения прав в файловой системе. Эти биты могут устанавливать специальные права для команд и каталогов. Основное внимание в этом разделе уделяется настройке битов, которые помогают создавать каталоги для совместной работы.

Как и биты чтения, записи и выполнения для `user`, `group` и `other`, эти специальные биты прав для файлов могут быть установлены с помощью команды `chmod`. При запуске, например, команды `chmod 775 /mnt/xyz` на самом деле подразумеваются права `0775`. Чтобы изменить права, замените число `0` любой комбинацией этих трех битов (4, 2 и 1) или возьмите вместо них буквенные значения. (Обратитесь к главе 4, если нужно освежить знания о том, как работают права.) Буквы и цифры приведены в табл. 11.1.

Таблица 11.1. Команды для создания и использования файлов

Название	Цифровое значение	Буквенное значение
Бит Set user ID	4	u + s
Бит Set group ID	2	g + s
Бит Sticky bit	1	o + t

Биты, которые подходят для создания совместных каталогов, — это бит `Set group ID (2)` и `Sticky bit (1)`. Если вас интересуют другие варианты использования битов `Set user ID` и `Set group ID`, обратитесь к врезке «Команды Set UID и Set GID Bit» далее.

Создание каталогов для групп (установка бита GID)

При создании каталога set GID все файлы, созданные в нем, назначаются соответствующей каталогу группе. Суть здесь в том, чтобы иметь каталог, в котором все члены группы могут обмениваться файлами, но при этом защищать их от других пользователей. Вот список действий для создания совместного каталога для всех пользователей в группе sales.

1. Создайте группу для совместной работы:

```
# groupadd -g 301 sales
```

2. Добавьте в нее пользователей, с которыми хотите обмениваться файлами (я взял пользователя mary):

```
# usermod -aG sales mary
```

3. Создайте каталог совместной работы:

```
# mkdir /mnt/salestools
```

Команды Set UID и Set GID Bit

Биты Set UID и Set GID используются в специальных исполняемых файлах, которые позволяют запускать команды несколько иначе. Обычно, когда пользователь применяет команду, она выполняется с правами этого пользователя. Другими словами, если я запускаю команду vi от имени chris, то она будет иметь права на чтение и запись файлов, которые пользователь chris может читать и записывать.

Команды с битами set UID или set GID отличаются друг от друга. Именно владелец и группа, назначенные команде, определяют права, которые она может применять к ресурсам на компьютере. Так, команда set UID принадлежит суперпользователю и будет работать с правами root, команда set GID, относящаяся к apache, будет иметь права группы apache.

Примерами с включенными UID являются команды su и newgrp. Обе они должны действовать как суперпользователь, чтобы выполнять свою работу. Однако, чтобы получить права суперпользователя, нужно ввести пароль. Можно сказать, что su — это команда set UID, по тому, где обычно находится первый бит выполнения (x):

```
$ ls /bin/su
-rwsr-xr-x. 1 root root  30092 Jan 30 07:11 su
```

4. Назначьте группу sales в каталог:

```
# chgrp sales /mnt/salestools
```

5. Измените права каталога на 2775. Будет подключен бит set group ID (2) и даны полные права доступа rwx для user (7), права доступа rwx для group (7) и p-x (5) для other:

```
# chmod 2775 /mnt/salestools
```

6. Зайдите от имени пользователя mary (запустите команду su – mary). Как пользователь mary создайте файл в общем каталоге и рассмотрите права. При перечислении прав видно, что каталог является каталогом set GID, потому что нижний регистр для параметра s появляется там, где должно быть право на выполнение для группы (rwxrwsr-x):

```
 # su - mary
 [mary]$ touch /mnt/salestools/test.txt
 [mary]$ ls -ld /mnt/salestools/ /mnt/salestools/test.txt
 drwxrwsr-x. 2 root sales 4096 Jan 22 14:32 /mnt/salestools/
-rw-rw-r--. 1 mary sales   0 Jan 22 14:32 /mnt/salestools/test.txt
```

Как правило, файл, созданный mary, будет иметь назначенную ему группу mary. Но поскольку файл test.tx был создан в каталоге set group ID, файл присваивается группе sales. Теперь любой, кто принадлежит к группе sales, может читать или записывать этот файл, основываясь на правах группы.

Создание каталогов с ограниченным удалением (бит sticky)

Каталог с ограниченным удалением создается после включения функции бит sticky. Чем каталог с ограниченным удалением отличается от других каталогов? Обычно, если для пользователя открыто право на запись в файле или каталоге, он может удалить этот файл или каталог. Однако вы не сможете удалить файлы другого пользователя в каталоге с ограниченным удалением, если не являетесь суперпользователем или владельцем каталога.

Как правило, каталог с ограниченным удалением используется там, где множество различных пользователей могут создавать файлы. Например, каталог /tmp — это каталог с ограниченным удалением:

```
$ ls -ld /tmp
drwxrwxrwt . 116 root root 36864 Jan 22 14:18 /tmp
```

Видно, что права открыты, но вместо x для бита execute для other устанавливается бит sticky. Далее приведен пример создания каталога с ограниченным удалением с файлом, который открыт для записи любым пользователем:

```
[mary]$ mkdir /tmp/mystuff
[mary]$ chmod 1777 /tmp/mystuff
[mary]$ cp /etc/services /tmp/mystuff/
[mary]$ chmod 666 /tmp/mystuff/services
[mary]$ ls -ld /tmp/mystuff /tmp/mystuff/services
drwxrwxrwt. 2 mary mary   4096 Jan 22 15:28 /tmp/mystuff/
-rw-rw-rw-. 1 mary mary 640999 Jan 22 15:28 /tmp/mystuff/services
```

Имея значение прав 1777 в каталоге /tmp/mystuff, вы можете видеть, что все права доступны, но вместо последнего бита выполнения появляется t. Если в файл

/tmp/mystuff/services разрешена запись, любой пользователь может открыть его и изменить содержимое. Но поскольку файл находится в каталоге sticky bit, только суперпользователь и пользователь mary могут удалить его.

Централизация учетных записей

По умолчанию аутентификация пользователей в Linux осуществляется сверкой информации о пользователе с файлом /etc/passwd и паролями из файла /etc/shadow, однако ее можно выполнять и другими способами. На большинстве крупных предприятий информация об учетных записях пользователей хранится на централизованном сервере аутентификации, поэтому каждый раз, когда вы устанавливаете новую систему Linux, она вместо добавления учетных записей пользователей в эту систему запрашивает сервер аутентификации при попытке входа в систему.

Как и при локальной аутентификации passwd/shadow, настройка централизованной аутентификации требует предоставления двух типов информации: об учетной записи (имя пользователя, идентификаторы пользователей/групп, домашний каталог, оболочка по умолчанию и т. д.) и методе аутентификации (различные типы зашифрованных паролей, смарт-карты, сканирование сетчатки глаза и т. д.). Linux позволяет настраивать эти типы информации.

Домены аутентификации, поддерживаемые с помощью команды authconfig, — это LDAP, NIS и Windows Active Directory.

Поддерживаемые типы централизованных баз данных таковы.

- **LDAP.** *Легковесный протокол доступа к каталогам* (Lightweight Directory Access Protocol) — популярный протокол для предоставления служб каталогов, таких как телефонные книги, адреса и учетные записи пользователей. Это открытый протокол, который настраивается во многих типах вычислительных сред.
- **NIS.** *Сетевая база данных* (Network Information Service) впервые была создана компанией Sun Microsystems для распространения сведений об учетных записях, конфигурации хоста и других типов системной информации во многих системах UNIX. Поскольку NIS передает информацию в виде открытого текста, большинство предприятий теперь используют более безопасные протоколы LDAP или Winbind для централизации аутентификации.
- **Winbind.** Выбор *Winbind* в окне Конфигурация аутентификации (Authentication Configuration) позволяет аутентифицировать пользователей на сервере Microsoft Active Directory (AD). Многие крупные компании расширяют свои настройки аутентификации на рабочем столе, чтобы выполнять настройку сервера, а также использовать сервер AD.

Если вы хотите создать собственные централизованные службы аутентификации и использовать проект с открытым исходным кодом, я рекомендую обратиться к серверу каталогов 389 Directory Server (directory.fedoraproject.org). Fedora и другие системы Linux работают с сервером LDAP.

Резюме

Отдельные учетные записи пользователей — это одно из основных средств обеспечения безопасных границ между пользователями системы Linux. Рядовые пользователи обычно могут управлять файлами и каталогами в собственных домашних каталогах, но за их пределами для этого нужны дополнительные права.

В этой главе вы узнали, как добавлять учетные записи пользователей и групп, изменять их и даже расширять учетные записи пользователей и групп за пределы локального файла `/etc/passwd`. Вы также узнали, что аутентификация может быть выполнена путем доступа к централизованным серверам LDAP.

В следующей главе вы познакомитесь еще с одной основной темой, необходимой системным администраторам Linux, — с управлением дисками. Вы узнаете, как разделять диски, добавлять файловые системы и монтировать их, чтобы содержимое разделов диска было доступно тем, кто использует вашу систему.

Упражнения

Решая эти упражнения, проверьте свои знания в сфере управления учетными записями пользователей и групп в Linux. Задачи подходят для систем Fedora или Red Hat Enterprise Linux (некоторые сработают и в других системах Linux). Если затрудняетесь с решением заданий, воспользуйтесь ответами к упражнениям, приведенными в приложении Б (хотя Linux позволяет решать задачи разными способами).

1. Добавьте в свою систему Linux локальную учетную запись с именем пользователя `jbaxter` и полным именем John Baxter, которая задействует `/bin/sh` в качестве оболочки по умолчанию. Пусть UID будет назначен по умолчанию. Установите для `jbaxter` пароль `My1N1te0ut!`.
2. Создайте учетную запись группы с именем `testing` с идентификатором `315`.
3. Добавьте пользователя `jbaxter` в группу `testing` и в группу `bin`.
4. Откройте оболочку как `jbaxter` (либо как новый сеанс входа в систему, либо с помощью текущей оболочки) и временно сделайте так, чтобы группа `testing` была вашей группой по умолчанию и при вводе команды `touch /home/jbaxter/file.txt` группа `testing` назначалась группой файла.
5. Обратите внимание на то, какой идентификатор пользователя был назначен для `jbaxter`, и удалите учетную запись пользователя, не удаляя домашний каталог, назначенный пользователю.
6. Найдите в каталоге `/home` (и любых подкаталогах) все файлы, назначенные идентификатору пользователя, который недавно принадлежал пользователю `jbaxter`.

7. Скопируйте файл `/etc/services` в каталог с шаблонами по умолчанию, чтобы он отображался в домашнем каталоге любого нового пользователя. Затем добавьте в систему нового пользователя с именем `mjones`, полным именем Mary Jones и домашним каталогом `/home/maryjones`.
8. Найдите в каталоге `/home` все файлы, принадлежащие пользователю `mjones`. Есть ли какие-нибудь файлы, принадлежащие `mjones`, которых вы не ожидали увидеть?
9. Войдите в систему как пользователь `mjones` и создайте файл с именем `/tmp/maryfile.txt`. С помощью списка ACL назначьте пользователю `bin` права на чтение/запись в этом файле. Затем назначьте этому файлу права на чтение и запись для группы `lp`.
10. Все еще как пользователь `mjones` создайте каталог с именем `/tmp/mydir`. Используя ACL, назначьте этому каталогу права по умолчанию, чтобы пользователь `adm` имел права на чтение/запись/выполнение этого каталога и любых созданных в нем файлов или каталогов. Создайте каталог `/tmp/mydir/testing/` и файл `/tmp/mydir/newfile.txt` и убедитесь, что пользователю `adm` также были назначены полные права на чтение/запись/выполнение. (Обратите внимание: несмотря на то что права `rwx` назначены пользователю `adm`, на файл `newfile.txt` разрешены права только `rw`. Что можно сделать, чтобы убедиться, что `adm` получает права также на выполнение файла?)

12 Управление дисками и файлами

В этой главе

- Работа со скриптами оболочки.
- Создание логических томов с помощью LVM.
- Добавление файловых систем.
- Монтирование файловых систем.
- Размонтирование файловых систем.

Операционная система, приложения и данные должны храниться в постоянном хранилище, чтобы при включении и выключении компьютера данные оставались на месте. Обычно хранилище — это жесткий диск компьютера. Чтобы упорядочить на нем информацию, его обычно разделяют на части, причем большинство разделов имеют структуру, называемую *файловой системой*.

В этой главе описано, как работать с жесткими дисками. Задачи жесткого диска включают в себя разбиение на разделы, добавление файловых систем и управление ими различными способами. Устройства хранения данных, подключенные к системам, например съемные устройства, включая жесткие диски (HDD) и твердотельные накопители (SSD), а также сетевые устройства, могут быть разделены, и ими можно управлять одним и тем же способом.

После описания основных способов разбиения дисков я расскажу, как можно использовать инструмент Logical Volume Manager (LVM), чтобы упростить управление файловыми системами и сделать его более эффективным.

Пространство диска

Основы работы хранилища данных одинаковы для большинства современных операционных систем. При установке операционной системы диск представляет собой один раздел или делится на несколько. Каждый раздел формирует файловую

систему. В Linux некоторые разделы могут быть специально отформатированы для области подкачки или логических томов LVM. Диски используются как постоянное хранилище, а *оперативная память* (*ОЗУ*, *RAM*) и подкачка применяются для временного хранения. Например, при выполнении команды она копируется с жесткого диска в оперативную память, чтобы процессор компьютера (CPU) мог быстрее получить к ней доступ.

Процессор может получить доступ к данным гораздо быстрее из оперативной памяти, чем с жесткого диска, хотя твердотельные накопители (SSD) сами по себе больше похожи на оперативную память, чем на жесткие диски. Однако пространство диска обычно намного больше, чем оперативная память, а оперативная память намного дороже и стирается при перезагрузке компьютера. Представьте, что офис — это оперативная память и диск. Тогда диск — это как картотека, где хранятся папки с нужной информацией. А оперативная память похожа на столешницу, куда кладут папку с бумагами во время работы. Когда папка не используется, она возвращается в картотеку.

Когда оперативная память заполняется из-за запуска слишком большого количества процессов или процесса с утечкой памяти, новые процессы не будут запускаться, если не расширить системную память компьютера. Вот тут-то и пригодится раздел подкачки. *Пространство подкачки* — это раздел подкачки жесткого диска или файл подкачки, через который компьютер может «выкачать» не используемые в данный момент данные из оперативной памяти, а затем «подкачать» их обратно в оперативную память, когда они снова понадобятся. Конечно, хорошо бы вообще никогда не превышать объем оперативной памяти (при подкачке падает производительность), но подкачка лучше, чем сбои в процессах.

Еще один специальный инструмент разбиения — это *менеджер логических томов* (Logical Volume Manager, LVM). Физические тома LVM позволяют создавать пулы дискового пространства, называемые *группами томов*. Группы томов дают большую гибкость при увеличении и уменьшении логических томов, чем при непосредственном изменении размера разделов диска.

Для Linux требуется по крайней мере один раздел диска, назначенный для root (`/`) всей файловой системы Linux. Однако чаще всего отдельные разделы назначаются определенным каталогам, таким как `/home`, `/var` и/или `/tmp`. Каждый из разделов подключается к более крупной файловой системе Linux, монтируя ее в точку файловой системы, где вы хотите использовать этот раздел. Любой файл, добавленный в каталог точки монтирования раздела или подкаталога, хранится в этом разделе.

ПРИМЕЧАНИЕ

Слово «монтировать» (mount) означает подключение файловой системы с жесткого диска, USB-накопителя или сетевого устройства хранения данных к определенной точке файловой системы. Это действие выполняется с помощью команды mount, а также параметров, указывающих команде, где находится запоминающее устройство и к какому каталогу в файловой системе его следует подключить.

Подключение разделов диска к файловой системе Linux выполняется автоматически и незаметно для конечного пользователя. Как же это происходит? Каждый раздел обычного диска, созданный при установке Linux, связан с именем устройства. Запись в файле `/etc/fstab` сообщает Linux имя устройства каждого раздела и место его монтирования (а также другие биты информации). Монтирование выполняется при загрузке системы.

Большая часть этой главы посвящена тому, как нужно разделять и подключать диск компьютера для формирования файловой системы Linux, а также как разбивать диски, форматировать файловые системы и раздел подкачки и использовать эти элементы при загрузке системы. Затем мы рассмотрим, как выполнить разбиение диска и создать файловую систему вручную.

Отличие от Windows

Файловые системы в Linux организованы иначе, чем в операционных системах Microsoft Windows. Вместо того чтобы обозначать диски, сетевые файловые системы, CD-ROM и другие типы носителей буквами (например, A:, B:, C:), все носители четко вписывают в структуру каталогов Linux.

Некоторые диски автоматически подключаются (монтируются) в файловую систему при добавлении съемных носителей. Например, CD может быть установлен в каталоге `/media/cdrom`. Если диск не монтируется автоматически, администратор должен создать в файловой системе точку монтирования, а затем подключить к ней диск.

Linux понимает файловые системы VFAT, которые часто являются форматом по умолчанию для USB-накопителей. Форматы USB-накопителя VFAT и exFAT позволяют обмениваться данными между системами Linux и Windows. Поддержка ядра Linux доступна для файловых систем NTFS, которые сейчас обычно используются в Windows. Однако NTFS, а иногда и exFAT требуют установки дополнительных драйверов ядра в Linux.

Файловые системы VFAT обычно применяются для обмена файлами между различными типами операционных систем. Поскольку VFAT использовалась в MS-DOS и ранних операционных системах Windows, ее можно считать хорошим способом передачи файлов между многими типами систем, включая Linux. NTFS — это тип файловой системы, наиболее часто задействуемый в современных системах Microsoft Windows.

Разбиение жесткого диска

В Linux существует несколько инструментов для управления разделами жесткого диска. Необходимо знать, как разбить диск на разделы, если вы хотите добавить диск в свою систему или изменить его конфигурацию.

В следующих разделах рассмотрим разбиение диска на разделы с помощью съемного USB-накопителя и фиксированного жесткого диска. Для безопасности я беру USB-накопитель, не содержащий никаких важных данных.

> **Разбиение диска может вызвать сбои в системе**
>
> Я не советую использовать основной жесткий диск вашей системы, чтобы попрактиковаться в разбиении диска, так как из-за любой ошибки система может перестать загружаться. Даже если вы возьмете для этого отдельный USB-накопитель, неправильная запись в файле `/etc/fstab` может привести к зависанию системы при перезагрузке. Если после разбиения диска на разделы система не загружается, обратитесь к главе 21, чтобы узнать, как устранить эту проблему.

Таблицы разделов

Настольные компьютеры традиционно используют *таблицы разделов Master Boot Record (MBR)* для хранения информации о размерах и расположении разделов жесткого диска. Существует множество хорошо известных инструментов для управления разделами MBR. Однако несколько лет назад новый стандарт — *таблицы разделов GUID* — начал применяться в системах как часть компьютерной архитектуры UEFI для замены старого метода загрузки системы BIOS.

Многие инструменты разбиения дисков Linux были обновлены для обработки новых *таблиц разделов GUID (gpt)*. Были добавлены и другие инструменты для обработки таблиц разделов GUID. Поскольку популярная команда `fdisk` не поддерживает gpt-разделы, вместо нее в примерах этой главы применяется команда `parted`.

Ограничения, наложенные спецификацией MBR, вызвали потребность в разделах GUID. В частности, размер MBR-разделов ограничен 2 Тбайт. Разбиение GUID может создавать разделы размером до 9,4 Збайт (зеттабайт).

Просмотр разделов диска

Для просмотра разделов диска используйте команду `parted` с параметром `-l`. Далее приведен пример разбиения на фиксированном жестком диске емкостью 160 Гбайт в системе Red Hat Enterprise Linux 8:

```
# parted -l /dev/sda
Disk /dev/sda: 160.0 GB, 160000000000 bytes, 312500000 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
```

```
Disk identifier: 0x0008870c
   Device Boot      Start         End      Blocks   Id   System
/dev/sda1   *        2048     1026047      512000   83   Linux
/dev/sda2         1026048   304281599   151627776   8e   Linux LVM
```

При добавлении USB-накопитель назначается следующему доступному устройству `sd`. В следующем примере показано разбиение на разделы жесткого диска (`/dev/sda`) и USB-накопителя из системы Fedora 30, где `/dev/sdb` назначается в качестве имени USB-устройства (второго диска в системе). Это новый USB-накопитель на 128 Гбайт:

```
# fdisk -l /dev/sdb
```

Хотя этот диск был назначен файлу `/dev/sdb`, ваш диск может быть назначен другому устройству. Как это понять?

- Запоминающее устройство SCSI или USB, представленное устройством `sd?` (например, `sda`, `sdb`, `sdc` и т. д.), может иметь до 16 второстепенных устройств (например, основное устройство `/dev/sdc` и `/dev/sdc1` через `/dev/sdc15`). Таким образом, всего может быть 15 разделов. Встроенный SSD, представленный устройством `nvme` (например, `nvme0`, `nvme1`, `nvme2` и т. д.), может быть одним пространством имен и разделов или разбит на несколько (большинство устройств просто использует первое пространство имен). Например, файл `/dev/nvme0n1p1` представляет первый раздел в первом пространстве имен на первом SSD-накопителе NVMe.
- На компьютерах x86 диски могут разделяться максимум на четыре части. Таким образом, чтобы иметь больше четырех полных разделов, один из них нужно расширить. Любые разделы, выходящие за пределы четырех основных разделов, являются логическими разделами, использующими пространство из расширенного раздела.
- Поле `id` указывает на тип раздела. Обратите внимание на то, что в первом примере есть раздел Linux LVM.

Ваш первый основной жесткий диск обычно отображается как `/dev/sda`. При установке RHEL и Fedora чаще всего добавляется по крайней мере один раздел LVM, созданный инсталлятором, из которого могут быть назначены другие разделы. Таким образом, вывод команды `fdisk` может быть таким же простым, как в примере:

```
# parted -l
Disk /dev/sda: 500.1 GB, 500107862016 bytes
```

Первый раздел имеет размер примерно 210 Мбайт и монтируется в каталоге `/boot/efi`. Второй раздел (1074 Мбайт) монтируется в разделе `/boot`. Для более старых таблиц разделов MBR существует только раздел `/boot`. В `boot` в колонке `Flags` указывается то, что это загрузочный раздел. Остальная часть диска задействуется разделом LVM, который в конечном итоге используется для создания логических томов.

Я рекомендую пока не использовать жесткий диск, а найти пустой USB-накопитель. На нем вы можете опробовать команды, описанные далее.

Создание диска с одним разделом

Чтобы добавить новый носитель информации (жесткий диск, USB-накопитель или аналогичное устройство) на компьютер и использовать его в Linux, сначала нужно подключить дисковое устройство к компьютеру, а затем разбить диск на разделы. Порядок действий такой.

1. Установите новый жесткий диск или вставьте новый USB-накопитель.
2. Разбейте диск на разделы.
3. Создайте на новом диске файловые системы.
4. Смонтируйте файловые системы.

Самый простой способ добавить диск или USB-накопитель в Linux — это отвести весь диск одному разделу Linux. Однако при желании можете иметь несколько разделов и назначать их для разных типов файловых систем и разных точек монтирования.

Далее рассмотрим порядок действий в Linux для создания USB-накопителя с одним разделом. Если у вас уже есть пустой USB-накопитель (любого размера), можете во время чтения выполнить описанные действия. (О том, как разбить диск на несколько разделов, поговорим позже.)

ПРЕДУПРЕЖДЕНИЕ

Если при разбиении диска на разделы с помощью команды parted вы допустили ошибку, убедитесь, что она исправлена. В отличие от команды fdisk, когда можно просто ввести команду q и выйти без сохранения, команда parted сохраняет и применяет изменения сразу же, поэтому просто выйти не получится.

1. Подключите USB-накопитель к доступному USB-порту. Я буду использовать 128-гигабайтный USB-накопитель, но вы можете взять его любой емкости.
2. Определите имя устройства для USB-накопителя. От имени суперпользователя из командной консоли введите команду `journalctl`, а затем вставьте USB-накопитель. Появятся сообщения, отображающие имя устройства, которое вы только что подключили (нажмите сочетание клавиш Ctrl+C, чтобы выйти из команды `tail`, когда закончите):

```
# journalctl -f
kernel: usb 4-1: new SuperSpeed Gen 1 USB device number 3 using
                 xhci_hcd
kernel: usb 4-1: New USB device found, idVendor=0781,
                 idProduct=5581, bcdDevice= 1.00
kernel: usb 4-1: New USB device strings: Mfr=1, Product=2,
                 SerialNumber=3
```

```
kernel: usb 4-1: Product: Ultra
kernel: usb 4-1: Manufacturer: SanDisk
...
kernel: sd 6:0:0:0: Attached scsi generic sg2 type 0
kernel: sdb: sdb1
kernel: sd 6:0:0:0: [sdb] Attached SCSI removable disk
udisksd[809]: Mounted /dev/sdb1 at /run/media/chris/7DEB-B010
              on behalf of uid 1000
```

3. Из выходных данных видно, что USB-накопитель был найден и назначен файлу `/dev/sdb`. (Имя вашего устройства может отличаться.) Он также содержит один форматированный раздел `sdb1`. Убедитесь, что вы точно определили нужный диск, иначе можете потерять все данные с других дисков.
4. Если USB-накопитель монтируется автоматически, размонтируйте его. Пример иллюстрирует, как найти разделы USB и размонтировать их:

```
# mount | grep sdb
/dev/sdb1 on /run/media...
# umount /dev/sdb1
```

5. С помощью команды `parted` создайте разделы на USB-накопителе. Например, если вы форматируете второй диск USB, SATA или SCSI (`sdb`), введите следующее:

```
# parted /dev/sdb
GNU Parted 3.2
Using /dev/sdb
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted)
```

   Теперь вы находитесь в режиме команды `parted`, в котором можно применять набор однобуквенных команд для работы с разделами.
6. У нового, чистого USB-накопителя может быть один раздел, полностью отведенный для файловой системы, совместимой с системой Windows, например VFAT или FAT32. Используйте команду `p` для просмотра всех разделов, чтобы удалить раздел. Вот как это выглядело у меня:

```
(parted) p
Model: SanDisk Ultra (scsi)
Disk /dev/sdb: 123GB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:

Number   Start   End    Size   Type    File system  Flags
1       16.4kB  123GB  123GB  primary  fat32        lba
(parted) rm
Partition number? 1
```

7. Переименуйте диск на диск с таблицами разделов gpt.

```
(parted) mklabel gpt
Warning: The existing disk label on /dev/sdb will be destroyed and all data
on this disk will be lost. Do you want to continue?
Yes/No? Yes
(parted)
```

8. Чтобы создать новый раздел, введите команду `mkpart`. Вам будет предложено задать тип файловой системы, а затем начало и конец раздела. В примере далее раздел называется `alldisk`, тип файловой системы — `xfs`, начинается с `1M` и заканчивается на `123GB`:

```
(parted) mkpart
Partition name?  []? alldisk
File system type?  [ext2]? xfs
Start? 1
End? 123GB
```

9. Дважды проверьте, что диск разделен так, как нужно, нажав клавишу `p`. (Ваши выходные данные будут отличаться — они зависят от размера диска.)

```
(parted) p
Model: SanDisk Ultra (scsi)
Disk /dev/sdb: 123GB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:

Number  Start   End    Size   File system  Name    Flags
 1      1049kB  123GB  123GB  xfs          alldisk
```

10. Разбиение диска выполнено, однако новый раздел еще не готов к применению. Далее нужно создать в нем файловую систему. Чтобы создать файловую систему в новом разделе диска, возьмите команду `mkfs`. По умолчанию она создает файловую систему `ext2`, которую можно использовать в Linux. Однако в большинстве случаев лучше задействовать файловую систему с возможностью ведения журнала, например, ext3, ext4 или xfs. Чтобы создать файловую систему xfs в первом разделе второго жесткого диска, введите следующее:

```
# mkfs -t xfs /dev/sdb1
```

СОВЕТ

Вы можете использовать различные команды или параметры этой команды для создания других типов файловых систем. Например, примените команду mkfs.exfat, чтобы создать файловую систему VFAT, mkfs.msdos — для создания DOS, mkfs.ext — для ext4. Файловая система VFAT или exFAT (доступная с Ubuntu) может понадобиться, если нужно обмениваться файлами между системами Linux, Windows и Mac.

11. Чтобы использовать новую файловую систему, нужно создать точку монтирования и подключить ее к разделу. Пример того, как это сделать, показан далее. Затем нужно убедиться, что монтирование прошло успешно:

```
# mkdir /mnt/test
# mount /dev/sdb1 /mnt/test
# df -h /mnt/sdb1
Filesystem         Size  Used Avail Use% Mounted on
/dev/sdb1          115G   13M  115G   1% /mnt/test
```

Команда `df` показывает, что `/dev/sdb1` монтируется на `/mnt/test` и предлагает около 115 Гбайт дискового пространства. Команда `mount` показывает все смонтированные файловые системы, но в примере указана только `sdb1`, чтобы показать, что она смонтирована. Любые файлы или каталоги, которые вы создадите позже в каталоге `/mnt/test`, а также в любом из его подкаталогов, хранятся на устройстве `/dev/sdb1`.

12. Когда закончите работать с диском, его можно размонтировать с помощью команды `unmount`, после чего безопасно удалить (см. описание команды `unmount` далее в этой главе на случай, если эта команда завершится неудачно):

```
# umount /dev/sdb1
```

13. Обычно USB-накопитель не настраивается для автоматического монтирования при каждой загрузке системы, потому что монтируется автоматически при подключении. Но при желании вы можете отредактировать файл `/etc/fstab` и добавить строку, описывающую, что и где монтировать. Вот пример такой строки:

```
/dev/sdb1     /mnt/test     xfs     defaults     0 1
```

В этом примере раздел `/dev/sdb1` монтируется в каталог `/mnt/test` как файловая система `xfs`. Ключевое слово `defaults` устанавливает, что раздел монтируется во время загрузки. Значение `0` указывает системе не создавать автоматически резервные копии файлов из этой файловой системы с помощью команды `dump` (теперь она используется редко, но это поле здесь есть). Значение `1` в последнем столбце указывает системе проверить раздел на наличие ошибок после определенного количества монтирований.

Теперь у вас есть рабочий постоянный раздел диска. В дальнейшем описывается, как разбить диск, содержащий несколько разделов.

Создание диска с несколькими разделами

Теперь, когда вы знакомы с основным процессом разбиения диска на разделы, добавления файловой системы и предоставления ей доступа (временно и постоянно), пришло время попробовать выполнить более сложный вариант разбиения. Возьмем тот же самый 128-гигабайтный USB-накопитель и проделаем все то, что описано далее, чтобы создать несколько разделов на одном диске.

В данном случае я настроил раздел Master Boot Record (MBR), чтобы проиллюстрировать работу расширенных разделов и применить более старую команду `fdisk`. Я создал два раздела по 5 Гбайт (`sdb1` и `sdb2`), два по 3 Гбайт (`sdb3` и `sdb5`) и один на 4 Гбайт (`sdb6`). Устройство `sdb4` будет использоваться как расширенный раздел, который занимает все оставшееся дисковое пространство. Пространство для разделов `sdb5` и `sdb6` берется из расширенного раздела. Таким образом, остается достаточно места для создания новых разделов.

Как и в предыдущем примере, вставьте USB-накопитель и определите имя устройства (в моем случае это `/dev/sdb`). Кроме того, обязательно отключите все разделы, которые монтируются автоматически при подключении USB-накопителя.

СОВЕТ

При указании размера каждого раздела введите знак плюс и количество мегабайт или гигабайт, которые вы хотите ему назначить. Например, укажите +1024M для создания 1024-мегабайтного раздела или +10G — для 10-гигабайтного. Обязательно используйте знак плюс (+) и букву M или G! Если забыть их указать, команда fdisk будет разделять диск на секторы, что может привести к неожиданным результатам.

1. Я начал процесс с перезаписи USB-накопителя командой `dd` (`dd if=/dev/zero of=/dev/sd<number> bs=1M count=100`). Это позволило начать с новой главной загрузочной записи. Будьте осторожны и указывайте правильный номер диска, иначе можно стереть всю операционную систему!
2. Создайте шесть новых разделов.

```
# fdisk /dev/sdb
Welcome to fdisk (util-linux 2.33.2).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Device does not contain a recognized partition table.
Created a new DOS disklabel with disk identifier 0x8933f665.
Command (m for help): n
Partition type
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-240254975, default 2048):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-240254975, default
240254975): +5G

Created a new partition 1 of type 'Linux' and of size 5 GiB.

Command (m for help): n
Partition type
   p   primary (1 primary, 0 extended, 3 free)
   e   extended (container for logical partitions)
```

```
Select (default p): p
Partition number (2-4, default 2): 2
First sector (10487808-240254975, default 10487808):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (10487808-240254975, default
240254975): +5G

Created a new partition 2 of type 'Linux' and of size 5 GiB.

Command (m for help): n
Partition type
   p   primary (2 primary, 0 extended, 2 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (3,4, default 3): 3
First sector (20973568-240254975, default 20973568):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (20973568-240254975, default
240254975): +3G

Created a new partition 3 of type 'Linux' and of size 3 GiB.

Command (m for help): n
Partition type
   p   primary (3 primary, 0 extended, 1 free)
   e   extended (container for logical partitions)
Select (default e): e
Selected partition 4
First sector (27265024-240254975, default 27265024):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (27265024-240254975, default
240254975): <ENTER>

Created a new partition 4 of type 'Extended' and of size 101.6 GiB.

Command (m for help): n
All primary partitions are in use.
Adding logical partition 5
First sector (27267072-240254975, default 27267072):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (27267072-240254975, default
240254975): +3G

Created a new partition 5 of type 'Linux' and of size 3 GiB.

Command (m for help): n
All primary partitions are in use.
Adding logical partition 6
First sector (33560576-240254975, default 33560576):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (33560576-240254975, default
240254975): +4G

Created a new partition 6 of type 'Linux' and of size 4 GiB.
```

3. Перед сохранением проверьте разбиение, введя параметр `p`. Обратите внимание на то, что существует пять разделов (`sdc1`, `sdc2`, `sdc3`, `sdc5` и `sdc6`) и между столбцами `Start` и `End` сектор `sdc4` поглощается секторами `sdc5` и `sdc6`:

```
 Command (m for help): p
 ...
Device    Boot    Start       End    Sectors   Size Id Type
/dev/sdb1          2048  10487807   10485760     5G 83 Linux
/dev/sdb2      10487808  20973567   10485760     5G 82 Linux
/dev/sdb3      20973568  27265023    6291456     3G 83 Linux
/dev/sdb4     27265024 240254975 212989952 101.6G 5 Extended
/dev/sdb5      27267072  33558527    6291456     3G 83 Linux
/dev/sdb6      33560576  41949183    8388608     4G 83 Linux
```

4. Тип раздела по умолчанию — это Linux. Я решил, что хочу использовать некоторые разделы для пространства подкачки (введите `82`), FAT32 (введите `x`) и Linux LVM (введите `8e`). Для этого набрал параметр `t` и указал нужный тип раздела. Введите параметр `L`, чтобы просмотреть список типов разделов:

```
Command (m for help): t
Partition number (1-6): 2
Hex code (type L to list codes): 82
Changed type of partition 'Linux' to 'Linux swap / Solaris'.

Command (m for help): t

Partition number (1-6): 5
Hex code (type L to list codes): c
Changed type of partition 'Linux' to 'W95 FAT32 (LBA)'.

Command (m for help): t

Partition number (1-6): 6
Hex code (type L to list codes): 8e
Changed type of partition 'Linux' to 'Linux LVM'.
```

5. Теперь следует проверить, соответствует ли таблица разделов тому, что нужно, а затем записать изменения:

```
   Command (m for help): p
 ...
Device    Boot    Start      End   Sectors   Size Id Type
/dev/sdb1          2048 10487807  10485760     5G 83 Linux
/dev/sdb2     10487808 20973567   10485760     5G 82 Linux swap / Solaris
/dev/sdb3      20973568 27265023   6291456     3G 83 Linux
/dev/sdb4     27265024 240254975 212989952 101.6G 5 Extended
/dev/sdb5     27267072 33558527    6291456     3G c W95 FAT32 (LBA)
/dev/sdb6    33560576 41949183     8388608     4G 8e Linux LVM

  Command (m for help): w
```

```
  The partition table has been altered!
  The kernel still uses the old partitions. The new table will be used at
the next reboot.
  Syncing disks
```

6. После завершения записи убедитесь, что ядро знает об изменениях в таблице разделов. Для этого выполните поиск `sdb` в каталоге `/proc/partitions`. Если новых устройств там нет, выполните команду `partprobe /dev/sdb` на диске или перезагрузите компьютер:

```
# grep sdb /proc/partitions
   8       16  120127488 sdb
   8       17  120125440 sdb1
# partprobe /dev/sdb
# grep sdb /proc/partitions
   8       16  120127488 sdb
   8       17    5242880 sdb1
   8       18    5242880 sdb2
   8       19    3145728 sdb3
   8       20          1 sdb4
   8       21    3145728 sdb5
   8       22    4194304 sdb6
```

7. Теперь разделы настроены для различных типов содержимого, но, чтобы сформировать из разделов файловые системы или области подкачки, требуются другие команды. Вот как это сделать с только что созданными разделами.
   - **`sdb1`**. Чтобы превратить его в обычную файловую систему Linux `ext4`, введите:

     ```
     # mkfs -t ext4 /dev/sdb1
     ```

   - **`sdb2`**. Чтобы превратить его в область подкачки, введите:

     ```
     # mkswap /dev/sdb2
     ```

   - **`sdb3`**. Чтобы превратить его в файловую систему `ext2` (по умолчанию), введите:

     ```
     # mkfs /dev/sdb3
     ```

   - **`sdb5`**. Чтобы превратить его в файловую систему VFAT (по умолчанию), введите:

     ```
     # mkfs -t vfat /dev/sdb5
     ```

   - **`sdb6`**. Чтобы превратить его в физический том LVM, введите:

     ```
     # pvcreate /dev/sdb6
     ```

Теперь разделы готовы к монтированию, использованию в качестве областей подкачки или добавлению в группу томов LVM. См. раздел «Использование разделов LVM», чтобы узнать, как физические тома LVM применяются для окончательного создания логических томов LVM из групп томов. См. раздел «Монтирование файловых систем» далее, чтобы узнать, как монтировать файловые системы и включать области подкачки.

Использование разделов LVM

Базовое разбиение диска в Linux имеет свои недостатки. Что произойдет, если закончится место на диске? Раньше распространенным решением было скопировать данные на более крупный диск, перезагрузить систему с новым диском и надеяться на то, что в ближайшее время место снова не закончится. А это означало простои и неэффективность.

Менеджер логических томов (Logical Volume Manager, LVM) гораздо более гибок и эффективен, когда в ходе работы постоянно увеличиваются потребности в хранении данных. С помощью LVM физические разделы диска добавляются в пулы пространства, называемые группами томов. Логическим томам назначается место в группах томов по мере необходимости. Что это дает?

- Добавляется больше места на логический том из группы томов, пока он используется.
- Добавляются дополнительные физические тома в группу томов, если в ней скоро закончится место. Физические тома могут быть компакт-дисками.
- Можно переносить данные из одного физического тома в другой, позволяя удалять диски меньшего размера и заменять их более крупными во время использования файловых систем, то есть без простоев.

С помощью LVM проще сжимать файловые системы и освобождать место на диске, хотя сжатие требует размонтирования логического тома (перезагрузка не требуется). LVM также поддерживает расширенные функции, такие как зеркалирование и работа в кластерах.

Проверка существующего LVM

Начнем с рассмотрения существующего менеджера LVM в системе Red Hat Enterprise Linux. Следующая команда отображает разделы на моем первом жестком диске:

```
# fdisk -l /dev/sda | grep /dev/sda
Disk /dev/sda: 160.0 GB, 160000000000 bytes
/dev/sda1   *        2048     1026047      512000   83   Linux
/dev/sda2   *     1026048   312498175   155736064   8e   Linux LVM
```

В этой системе RHEL жесткий диск объемом 160 Гбайт разбит на раздел Linux объемом 500 Мбайт (`sda1`) и раздел Linux LVM, который занимает остальную часть диска (`sda2`). Затем я ввел команду `pvdisplay`, чтобы увидеть, используется ли этот раздел в группе LVM:

```
# pvdisplay /dev/sda2
  --- Physical volume ---
  PV Name               /dev/sda2
  VG Name               vg_abc
  PV Size               148.52 GiB / not usable 2.00 MiB
```

```
  Allocatable           yes (but full)
  PE Size               4.00 MiB
  Total PE              38021
  Free PE               0
  Allocated PE          38021
  PV UUID               wlvuIv-UiI2-pNND-f39j-oH0X-9too-AOII7R
```

В примере видно, что физический том LVM, представленный `/dev/sda2`, занимает 148,52 Гбайт и все это пространство выделено группе томов с именем `vg_abc`. Наименьшая единица хранения, которая может быть использована из этого физического тома, составляет 4 Мбайт, что называется *физическим экстентом* (Physical Extent, PE).

ПРИМЕЧАНИЕ

Обратите внимание на то, что утилиты LVM отображают дисковое пространство в мебибайтах (МиБ) и гибибайтах (ГиБ). Один мегабайт составляет 1 000 000 байт (106), а 1 МиБ — 1 048 576 байт (220). Мебибайт — это более точная единица отражения того, как данные хранятся на компьютере. Но маркетологи, как правило, используют мегабайт, потому что так кажется, что жесткие диски, CD и DVD имеют большую емкость, чем на самом деле. Имейте в виду, что большинство инструментов в Linux отображают данные хранилища в мебибайтах и гибибайтах, хотя некоторые могут отображать также в мегабайтах и гигабайтах.

Далее рассмотрим данные о группе томов:

```
# vgdisplay vg_abc
  --- Volume group ---
  VG Name               vg_abc
  System ID
  Format                lvm2
  Metadata Areas        1
  Metadata Sequence No  4
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                3
  Open LV               3
  Max PV                0
  Cur PV                1
  Act PV                1
  VG Size               148.52 GiB
  PE Size               4.00 MiB
  Total PE              38021
  Alloc PE / Size       38021 / 148.52 GiB
  Free  PE / Size       0 / 0
  VG UUID               c2SGHM-KU9H-wbXM-sgca-EtBr-UXAq-UnnSTh
```

В примере видно, что выделен 38 021 PE (физический экстент). Используя команду `lvdisplay`, как показано далее, можно увидеть, где они были распределены (я сократил некоторые выходные данные):

```
# lvdisplay vg_abc
  --- Logical volume ---
  LV Name                /dev/vg_abc/lv_root
  VG Name                vg_abc
  LV UUID                33VeDc-jd0l-hlCc-RMuB-tkcw-QvFi-cKCZqa
  LV Write Access        read/write
  LV Status              available
  # open                 1
  LV Size                50.00 GiB
  Current LE             12800
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           253:0
  --- Logical volume ---
  LV Name                /dev/vg_abc/lv_home
  VG Name                vg_abc
  ...
  LV Size                92.64 GiB
  --- Logical volume ---
  LV Name                /dev/vg_abc/lv_swap
  VG Name                vg_abc
  ...
  LV Size                5.88 GiB
```


Существует три логических тома, определяющих пространство из `vg_abc`. Каждый логический том связан с именем устройства, которое включает имя группы томов и имя логического тома: `/dev/vg_abc/lv_root` (50 Гбайт), `/dev/vg_abc/lv_home` (92,64 Гбайт) и `/dev/vg_abc/lv_swap` (5,88 Гбайт). Другие устройства, связанные с этими именами, находятся в каталогах `/dev/mapper`: `vg_abc-lv_home`, `vg_abc-lv_root` и `vg_abc-lv_swap`. Чтобы ссылаться на эти логические тома, можно использовать любой набор имен.

Корневой и домашний логические тома форматируются как файловые системы `ext4`, а логический том подкачки — как раздел подкачки. Заглянем в файл `/etc/fstab`, чтобы увидеть, как применяются эти логические тома:

```
# grep vg_ /etc/fstab
/dev/mapper/vg_abc-lv_root /       ext4   defaults   1 1
/dev/mapper/vg_abc-lv_home /home   ext4   defaults   1 2
/dev/mapper/vg_abc-lv_swap swap    swap   defaults   0 0
```

На рис. 12.1 показано, как различные разделы, группы томов и логические тома соотносятся со всей файловой системой Linux. Устройство `sda1` форматируется как файловая система и монтируется в каталог `/boot`. Устройство `sda2` предоставляет место для группы томов `vg_abc`. Затем логические тома `lv_home` и `lv_root` монтируются в каталоги `/home` и `/` соответственно.

Если вам не хватает места в любом из логических томов, можно назначить больше места из группы томов. Если в группе томов не хватает места, можно добавить

другой жесткий диск или сетевой накопитель и объединить его с пространством группы томов.

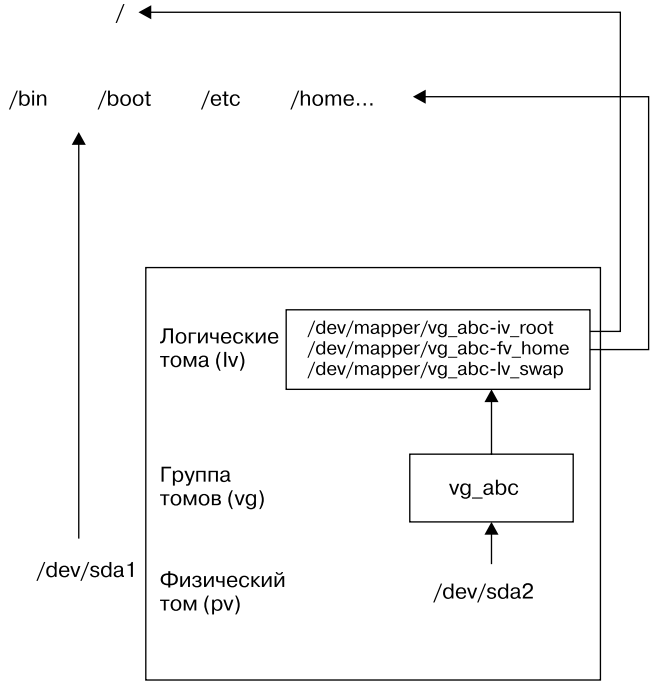


Рис. 12.1. Логические тома LVM можно монтировать как обычные разделы в файловой системе Linux

Теперь, поняв, как работает LVM, можно создавать логические тома LVM с нуля. Процесс будет описан далее.

Создание логических томов LVM

Логические тома LVM используются сверху вниз, но формируются снизу вверх. Как показано на рис. 12.1, сначала создается один или несколько физических томов (`pv`), затем они применяются для создания групп томов (`vg`), а группы томов образуют логические тома (`lv`).

Команды для работы с каждым компонентом LVM начинаются с букв `pv`, `vg` и `vl`. Например, команда `pvdisplay` показывает физические тома, `vgdisplay` — группы томов, а `lvdisplay` — логические тома.

Далее рассмотрим этапы создания томов LVM с нуля. Для этого можно использовать USB-накопитель и разделы, которые описаны ранее в этой главе.

1. Возьмите диск с некоторым количеством свободного пространства и создайте на нем раздел типа LVM (`8e`). Затем возьмите команду `pvcreate`, чтобы идентифицировать этот раздел как физический том LVM. Процесс с использованием устройства `/dev/sdb6` описан ранее в разделе «Создание диска с несколькими разделами».
2. Чтобы добавить этот физический том в новую группу томов, примените команду `vgcreate`. Как создать группу томов `myvg0` с помощью устройства `/dev/sdb6`, показывает следующая команда:

```
# vgcreate myvg0 /dev/sdc6
    Volume group "myvg0" successfully created
```

3. Чтобы просмотреть новую группу томов, введите:

```
# vgdisplay myvg0
--- Volume group ---
VG Name               myvg0
...
VG Size               <4.00 GiB
PE Size               4.00 MiB
Total PE              1023
Alloc PE / Size       0 / 0
Free  PE / Size       1023 / <4.00 MiB
```

4. Доступны все 1023 физических экстента (по 4 МиБ каждый). Далее приведен пример того, как создать логический том из части пространства в этой группе томов, а затем проверить, существует ли устройство для этого логического тома:

```
# lvcreate -n music -L 1G myvg0
  Logical volume "music" created
# ls /dev/mapper/myvg0*
/dev/mapper/myvg0-music
```

5. Как видно в примере, было создано устройство с именем `/dev/mapper/myvg0-music`. Теперь его можно использовать для установки и монтирования файловой системы, как и обычные разделы в начале этой главы, например:

```
# mkfs -t ext4 /dev/mapper/myvg0-music
# mkdir /mnt/mymusic
# mount /dev/mapper/myvg0-music /mnt/mymusic
# df -h /mnt/mymusic
Filesystem              Size Used Avail Use% Mounted on
/dev/mapper/myvg0-music 976M 2.6M  987M   1% /mnt/mymusic
```

6. Как и в случае с обычными разделами, логические тома можно монтировать постоянно, добавив запись в файл `/etc/fstab`, например:

```
/dev/mapper/myvg0-music /mnt/mymusic ext4 defaults 1 2
```

При следующей перезагрузке логический том автоматически монтируется в каталоге `/mnt/mymusic`. (Обязательно размонтируйте логический том и удалите эту строку, если хотите отсоединить USB-накопитель от компьютера.)

Расширение логических томов LVM

Если на логическом томе не хватает места, его можно добавить, и для этого не требуется размонтировать том. Нужно лишь иметь свободное место в группе томов, а также увеличить логический том и файловую систему. Основываясь на процессе, описанном в предыдущем разделе, вот как можно расширить логический том.

1. Обратите внимание на то, сколько места в данный момент на логическом томе, а затем проверьте, доступно ли оно в группе томов логического тома:

```
# vgdisplay myvg0
...
  VG Size               <4.00 MiB
  PE Size               4.00 MiB
  Total PE              1023
  Alloc PE / Size       256 / 1.00 GiB
  Free  PE / Size       767 / <3.00 GiB
# df -h /mnt/mymusic/
Filesystem                  Size Used Avail Use% Mounted on
/dev/mapper/myvg0-music     976M 2.6M  987M   1% /mnt/mymusic
```

2. Разверните логический том с помощью команды `vextend`:

```
# lvextend -L +1G /dev/mapper/myvg0-music
  Size of logical volume myvg0/music changed
        from 1.00GiB to 2.00 GiB (512 extents).
  Logical volume myvg0/music successfully resized
```

3. Измените размер файловой системы в соответствии с новым размером логического тома:

```
# resize2fs -p /dev/mapper/myvg0-music
```

4. Убедитесь, что файловая система изменена и включает в себя дополнительное пространство диска:

```
# df -h /mnt/mymusic/
Filesystem                Size   Used Avail Use% Mounted on
/dev/mapper/myvg0-music   2.0G   3.0M  1.9G   1% /mnt/mymusic
```

   Из примера видно, что файловая система теперь больше примерно на 1 Гбайт.

Монтирование файловых систем

Теперь, когда мы рассмотрели варианты разделения диска и файловых систем, я хочу сделать шаг назад и поговорить о том, как настраиваются файловые системы для постоянного подключения к системе Linux.

Большинство разделов жесткого диска, созданных при установке Linux, монтируются автоматически при загрузке системы. При установке Fedora, Ubuntu,

Red Hat Enterprise Linux и других систем Linux у вас есть возможность позволить инсталлятору автоматически настроить жесткий диск или самостоятельно создать разделы и указать точки монтирования для них.

Обычно при загрузке Linux все разделы на жестком диске перечисляются в файле `/etc/fstab` и монтируются. Поэтому в следующих подразделах описано, что можно найти в этом файле, а также каким образом можно монтировать другие разделы, чтобы они стали частью файловой системы Linux.

Команда `mount` используется для монтирования не только локальных устройств хранения данных, но и других типов файловых систем в системе Linux. Например, ее можно задействовать для монтирования каталогов (папок) по сети с серверов NFS или Samba, а также файловых систем с нового жесткого диска или USB-накопителя, который не настроен на автоматическое монтирование. Команда `mount` может монтировать и файлы изображений файловой системы с помощью петлевых loop-устройств.

ПРИМЕЧАНИЕ

С добавлением функций автоматического монтирования и внесением изменений в способ идентификации съемных носителей с ядром Linux 2.6 (использование таких функций, как Udev и Hardware Abstraction Layer) вам больше не нужно монтировать съемные носители вручную. Однако понимание того, как монтировать и размонтировать файловые системы вручную на сервере Linux, может быть очень полезным навыком, если вы хотите монтировать удаленные файловые системы или временно монтировать части системы.

Поддерживаемые файловые системы

Чтобы просмотреть типы файловых систем, в данный момент загруженных в ядро, введите `cat /proc/ filesystems`. В следующем списке приведены типы файловых систем, которые в настоящее время поддерживаются в Linux (они могут не использоваться или быть недоступными в вашем дистрибутиве Linux).

- **befs.** Файловая система, применяемая операционной системой BeOS.
- **btrfs.** Файловая система копирования при записи (copy-on-write), реализующая расширенные функции файловой системы. Отказоустойчива и проста в администрировании. Файловая система btrfs в последнее время набирает все большую популярность для корпоративных приложений.
- **cifs.** Общая файловая система Интернета (Common Internet Filesystem, CIFS) — виртуальная файловая система, используемая для получения доступа к серверам, соответствующим спецификации SNIA CIFS. CIFS — это попытка усовершенствовать и стандартизировать протокол SMB, задействуемый Samba и Windows для передачи данных.
- **ext4.** Преемница популярной файловой системы ext3. Включает в себя множество улучшений по сравнению с ext3, таких как поддержка томов до 1 ЭиБ (эксбибайт) и размеров файлов до 16 ТиБ (тебибайт). (Ранее заменяла ext3

в качестве файловой системы по умолчанию в Fedora и RHEL. Затем была заменена xfs в качестве системы по умолчанию в RHEL.)

- **ext3.** Файловые системы ext наиболее распространены в большинстве систем Linux. Если сравнивать с ext2, то файловая система ext3, также называемая третьей расширенной файловой системой, включает в себя функции ведения журнала, которые улучшают ее способность восстанавливаться после сбоев.
- **ext2.** Тип файловой системы по умолчанию для более ранних систем Linux. Функции такие же, как у ext3, за исключением функции ведения журнала.
- **ext.** Это первая версия ext3. Используется все реже.
- **iso9660.** Произошла из файловой системы High Sierra (оригинальный стандарт для CD-ROM). Расширения стандарта High Sierra, называемые Rock Ridge, позволяют файловым системам iso9660 поддерживать длинные имена файлов и информацию в стиле UNIX (например, имена файлов, сведения о праве собственности и ссылки). CD с данными обычно применяют этот тип файловой системы.
- **kafs.** Файловая система клиента AFS. Используется в распределенных вычислительных средах для обмена файлами с клиентами Linux, Windows и Macintosh.
- **minix.** Тип файловой системы Minix, первоначально используемый с версией MINIX UNIX. Поддерживает имена файлов длиной до 30 символов.
- **msdos.** Файловая система MS-DOS. Этот тип можно применять для монтирования носителей из старых операционных систем Microsoft.
- **vfat.** Расширенная файловая система Microsoft FAT (VFAT).
- **exfat.** Расширенная файловая система FAT (exFAT), оптимизированная для SD-карт, USB-накопителей и другой флеш-памяти.
- **umsdos.** Файловая система MS-DOS с расширениями, позволяющими использовать функции, аналогичные функциям UNIX, включая длинные имена файлов.
- **proc.** Не настоящая файловая система, а скорее интерфейс файловой системы к ядру Linux. Для ее настройки ничего особенного делать не нужно. Однако точка монтирования `/proc` должна быть файловой системой proc. Многие утилиты полагаются на `/proc` и получают из нее доступ к информации ядра Linux.
- **reiserfs.** Файловая система ReiserFS с ведением журнала. ReiserFS когда-то была распространенным типом файловой системы по умолчанию для нескольких дистрибутивов Linux. Однако сейчас файловые системы ext и xfs распространены шире.
- **swap.** Используется для разделов подкачки. Области подкачки применяются для временного хранения данных, когда заканчивается оперативная память. Данные перемещаются в область подкачки, а затем при необходимости возвращаются в оперативную память.

- **squashfs.** Сжатая файловая система, доступная только для чтения. Система Squashfs популярна на «живых» CD, где пространство ограничено и есть доступ только для чтения (например, CD или DVD).
- **nfs.** Сетевая файловая система (Network Filesystem, NFS). NFS применяется для монтирования файловых систем на других компьютерах Linux или UNIX.
- **hpfs.** Используется для монтирования файловой системы OS/2 HPFS и только для чтения.
- **ncpfs.** Файловая система для Novell NetWare. Файловые системы NetWare можно монтировать по сети.
- **ntfs.** Файловая система Windows NT. В зависимости от имеющегося дистрибутива может использоваться как файловая система только для чтения (чтобы смонтировать ее и скопировать файлы).
- **ufs.** Файловая система, популярная в операционных системах Sun Microsystems (то есть Solaris и SunOS).
- **jfs.** Шестидесятичетырехразрядная файловая система ведения журнала IBM, относительно легковесная для имеющегося количества функций.
- **xfs.** Высокопроизводительная файловая система, первоначально разработанная компанией Silicon Graphics, которая очень хорошо работает с большими файлами. Эта файловая система является системой по умолчанию для RHEL 7.
- **gfs2.** Файловая система с общим диском, которая позволяет нескольким машинам использовать один и тот же общий диск, не проходя через уровень сетевой файловой системы, такой как CIFS, NFS и т. д.

Чтобы просмотреть список файловых систем, которые поставляются вместе с ядром, введите команду `ls /lib/modules/kernelversion/kernel/fs/`. Реальные модули хранятся в подкаталогах этого каталога. Монтирование поддерживаемой файловой системы приводит к загрузке модуля файловой системы.

Введите команду `man fs`, чтобы просмотреть описания файловых систем Linux.

Подключение раздела подкачки

Раздел подкачки — это область диска, которая становится доступной для Linux, если в системе заканчивается оперативная память (ОЗУ). Если оперативная память заполнена и вы пытаетесь запустить другое приложение без раздела подкачки, действие не будет выполнено.

С помощью раздела подкачки Linux может временно помещать данные из оперативной памяти в раздел подкачки, а затем при необходимости возвращать их обратно. Производительность падает, но это лучше, чем полное невыполнение процессов.

Чтобы создать область подкачки из раздела или файла, используйте команду `mkswap`. Временно включить раздел подкачки можно командой `swapon`. В следующем

примере показано, как узнать доступное пространство подкачки, создать файл подкачки, включить его, а затем проверить, доступно ли это пространство в системе:

```
# free -m
             total       used       free     shared    buffers     cached
Mem:          1955        663       1291          0         42        283
-/+ buffers/cache:          337       1617
Swap:          819          0        819
# dd if=/dev/zero of=/var/tmp/myswap bs=1M count=1024
# mkswap /var/opt/myswap
# swapon /var/opt/myswap
# free -m
             total       used       free     shared    buffers     cached
Mem:          1955       1720        235          0         42       1310
-/+ buffers/cache:          367       1588
Swap:         1843          0       1843
```

Команда `free` показывает объем подкачки до и после создания и включения раздела подкачки с помощью команды `swapon`. Пространство подкачки будет доступно сразу же, но временно. Чтобы сделать эту область постоянной, нужно добавить ее в файл `/etc/fstab`, например:

```
/var/opt/myswap   swap      swap      defaults      0 0
```

Данная строка указывает, что файл подкачки с именем `/var/opt/myswap` должен быть включен во время загрузки. Поскольку для раздела подкачки нет точки монтирования, второе поле просто настроено на подкачку, как и тип раздела. Чтобы проверить работу файла подкачки перед перезагрузкой, можно сразу же включить его (`swapon -a`) и проверить, появляется ли дополнительная область подкачки:

```
# swapon -a
```

Отключение раздела подкачки

Чтобы отключить раздел подкачки, используйте команду `swapoff`. Причиной этого может стать, в частности, то, что раздел подкачки больше не нужен и вы хотите освободить занимаемое им место или удалить USB-накопитель, который поддерживает раздел подкачки.

Сначала убедитесь, что на устройстве подкачки не используется свободное пространство (с помощью команды `free`), а затем примените `swapoff`, чтобы отключить раздел подкачки и задействовать освободившееся место, например:

```
# free -m
             total       used       free     shared    buffers     cached
Mem:          1955       1720        235          0         42       1310
-/+ buffers/cache:          367       1588
Swap:         1843          0       1843
# swapoff /var/opt/myswap
# free -m
```

```
Mem:      1955    1720      235         0       42    1310
-/+ buffers/cache: 367     1588
Swap:      819       0      819
```

Обратите внимание: объем доступного места для подкачки был уменьшен после выполнения команды `swapoff`.

Определение монтируемых файловых систем с помощью файла fstab

Разделы жесткого диска на компьютере и удаленные файловые системы, которые используются ежедневно, скорее всего, настроены на автоматическое монтирование при загрузке Linux. Файл `/etc/fstab` содержит определения для каждого раздела, а также параметры, описывающие способ монтирования раздела. Пример применения файла `/etc/fstab`:

```
# /etc/fstab
/dev/mapper/vg_abc-lv_root     /        ext4   defaults       1 1
UUID=78bdae46-9389-438d-bfee-06dd934fae28 /boot ext4  defaults  1 2
/dev/mapper/vg_abc-lv_home     /home    ext4   defaults       1 2
/dev/mapper/vg_abc-lv_swap     swap     swap   defaults       0 0
# Mount entries added later
/dev/sdb1                      /win     vfat   ro 1 2
192.168.0.27:/nfsstuff         /remote  nfs    users,_netdev  0 0
//192.168.0.28/myshare         /share   cifs   guest,_netdev  0 0
# special Linux filesystems
tmpfs                          /dev/shm tmpfs  defaults       0 0
devpts                         /dev/pts devpts gid=5,mode=620 0 0
sysfs                          /sys     sysfs  defaults       0 0
proc                           /proc    proc   defaults       0 0
```

Здесь файл `/etc/fstab` взят из стандартной установки сервера Red Hat Enterprise Linux 6 с добавлением нескольких строк.

В данный момент вы можете игнорировать записи `tmpfs`, `devpts`, `sysfs` и `proc`. Это специальные устройства, связанные с общей памятью, терминальными окнами, информацией об устройстве и параметрами ядра соответственно.

В первом столбце файла `/etc/fstab` показано устройство или общий ресурс (что монтируется), а во втором — точка монтирования (где монтируется). За ними следуют тип файловой системы, любые параметры монтирования (или значения по умолчанию) и два числа, указывающие командам, таким как `dump` и `fsck`, что делать с файловой системой.

Первые три записи представляют разделы диска, назначенные корневому каталогу файловой системы (`/`), каталогу `/boot` и каталогу `/home`. Все они являются файловыми системами ext4. Четвертая строка — это устройство подкачки (применяется для хранения данных при переполнении оперативной памяти). Обратите внимание на то, что имена устройств для каталогов `/`, `/home` и `swap` начинаются с `/dev/mapper`. Это связано с тем, что они являются логическими томами LVM, которым назначается пространство из группы томов LVM (подробнее о LVM — в разделе «Использование разделов LVM» ранее в этой главе).

Раздел /boot находится в собственном физическом разделе /dev/sda1. Однако вместо /dev/sda1 устройство идентифицируется уникальным идентификатором (UUID). Зачем же применять UUID вместо /dev/sda1 для идентификации устройства? Предположим, вы подключили другой диск к компьютеру и загрузили систему. В зависимости от того, как ваш компьютер перебирает подключенные устройства при загрузке, возможно, новый диск будет идентифицирован как /dev/sda, что заставит систему искать содержимое /boot в первом разделе этого диска.

Чтобы просмотреть все UUID, назначенные устройствам хранения данных в своей системе, введите команду blkid:

```
# blkid
/dev/sda1:
  UUID="78bdae46-9389-438d-bfee-06dd934fae28" TYPE="ext4"
/dev/sda2:
  UUID="wlvuIv-UiI2-pNND-f39j-oH0X-9too-AOII7R" TYPE="LVM2_member"
/dev/mapper/vg_abc-lv_root:
  UUID="3e6f49a6-8fec-45e1-90a9-38431284b689" TYPE="ext4"
/dev/mapper/vg_abc-lv_swap:
  UUID="77662950-2cc2-4bd9-a860-34669535619d" TYPE="swap"
/dev/mapper/vg_abc-lv_home:
  UUID="7ffbcff3-36b9-4cbb-871d-091efb179790" TYPE="ext4"
/dev/sdb1:
  SEC_TYPE="msdos" UUID="75E0-96AA" TYPE="vfat"
```

Любое из названий устройств может быть заменено обозначением UUID в левой колонке в части /etc/fstab.

Следующие три строки я добавил в файл /etc/fstab, чтобы проиллюстрировать различные типы записей. Я подключил жесткий диск из старой системы Microsoft Windows и установил его в каталог /win. Затем добавил параметр ro, чтобы каталог монтировался только для чтения.

Последние две строки представляют удаленные файловые системы. В каталоге /remote каталог /nfsstuff монтируется для чтения/записи (rw) с узла 192.168.0.27 в качестве общего ресурса NFS. В каталоге /share общий ресурс Windows с именем myshare монтируется с узла 192.168.0.28. В обоих случаях я добавил параметр _netdev, который указывает Linux ждать появления сети, прежде чем пытаться смонтировать общие ресурсы. Дополнительные сведения о монтировании общих ресурсов CIFS и NFS см. в главах 19 и 20 соответственно.

Отличие от Windows

В разделе «Определение монтируемых файловых систем с помощью файла fstab» показано монтирование раздела жесткого диска из старой файловой системы VFAT, используемой в Windows. Большинство систем Windows сегодня работают с файловой системой NTFS. Эта файловая система поддерживается не во всех системах Linux. Для Fedora NTFS доступна в пакете ntfs-3g.

Чтобы разобраться в содержимом файла `/etc/fstab`, разделим его на поля.

- **Поле 1** — имя устройства, представляющего файловую систему. Это поле может иметь параметр `LABEL` или `UUID`, с помощью которых можно указать метку тома или универсальный уникальный идентификатор (UUID) вместо имени устройства. Преимущество этого подхода заключается в следующем: поскольку раздел идентифицируется по имени тома, его можно переместить на другое имя устройства, не изменяя при этом файл `fstab`. (См. описание команды `mkf` в разделе «Создание файловой системы с помощью mkf» далее в этой главе, чтобы узнать, как создавать файловые системы и ставить метки.)
- **Поле 2** — точка монтирования в файловой системе. Файловая система распространяет все данные из точки монтирования вниз по структуре дерева каталогов, если только другая файловая система не монтируется в какой-то точке под ней.
- **Поле 3** — тип файловой системы. Допустимые типы файловых систем описаны в разделе «Поддерживаемые файловые системы» ранее в этой главе (вы можете применять только те типы файловых систем, для которых в ядре есть драйверы).
- **Поле 4** — используйте команду `defaults` или список параметров, разделенных запятыми (без пробелов), которые хотите задействовать при монтировании записи. См. страницу руководства для команды `mount` (под параметром `-o`) для получения информации о других поддерживаемых параметрах.

СОВЕТ

Как правило, только суперпользователь может монтировать файловую систему с помощью команды mount. Чтобы разрешить любому пользователю монтировать файловую систему (например, на CD), можно добавить параметр user в поле 4 файла /etc/fstab.

- **Поле 5** — число в нем указывает, нужно ли сбрасывать файловую систему (то есть создавать резервные копии ее данных). Значение `1` означает, что файловая система должна быть сброшена, а `0` — что этого делать не нужно. (Сейчас это поле не особенно полезно, потому что большинство администраторов Linux применяют более сложные параметры резервного копирования, чем команда `dump`. Чаще всего используется значение `0`.)
- **Поле 6** — число в нем говорит о том, следует ли проверять указанную файловую систему с помощью `fsck`, когда придет время для ее проверки: `1` означает, что она должна быть проверена в первую очередь, `2` — что должна быть проверена после того, как все обозначенные `1` уже были проверены, а `0` — что ее проверять не нужно.

Если вы хотите больше узнать о параметрах монтирования, а также о других функциях файла `/etc/fstab`, обратитесь к справочным страницам `5 nfs` и `man 8 mount`.

Монтирование файловых систем с помощью команды mount

Системы Linux автоматически запускают команду `mount -a` (монтирование всех файловых систем из файла `/etc/fstab`) при каждой загрузке. По этой причине команда `mount` используется лишь в некоторых случаях. В частности, обычный пользователь или администратор применяют `mount` двумя способами:

- чтобы отобразить диски, разделы и удаленные файловые системы, смонтированные в данный момент;
- чтобы временно смонтировать файловую систему.

Любой пользователь может ввести `mount` без каких-либо параметров, чтобы увидеть, какие файловые системы в данный момент монтируются в локальной системе Linux. Далее приведен пример команды `mount`. Он показывает один раздел жесткого диска (`/dev/sda1`), содержащий корневую файловую систему (`/`) и типы файловых систем proc и devpts, установленные в `/proc` и `/dev` соответственно:

```
$ mount
/dev/sda3 on / type ext4 (rw)
/dev/sda2 on /boot type ext4 (rw)
/dev/sda1 on /mnt/win type vfat (rw)
/dev/proc on /proc type proc (rw)
/dev/sys on /sys type sysfs (rw)
/dev/devpts on /dev/pts type devpts (rw,gid=5,mode=620)
/dev/shm on /dev/shm type tmpfs (rw)
none on /proc/sys/fs/binfmt_misc type binfmt_misc (rw)
/dev/cdrom on /media/MyOwnDVD type iso9660 (ro,nosuid,nodev)
```

Наиболее распространенными устройствами для монтирования вручную являются съемные носители, такие как DVD или CD. Однако в зависимости от типа используемого рабочего стола CD и DVD могут монтироваться автоматически при присоединении к компьютеру. (В некоторых случаях при установке носителя также запускаются приложения. Например, могут начать работать музыкальный проигрыватель или фоторедактор, если на USB-носителе есть музыка или цифровые изображения.)

Бывают случаи, когда монтировать файловую систему вручную полезно. Например, вам нужно просмотреть содержимое старого жесткого диска, поэтому вы его устанавливаете в качестве второго диска на свой компьютер. Если разделы этого диска не монтируются автоматически, это можно сделать вручную. Так, чтобы смонтировать раздел диска `sdb1` с устаревшей файловой системой `ext3` и доступный только для чтения, введите следующее:

```
# mkdir /mnt/temp
# mount -t ext3 -o ro /dev/sdb1 /mnt/tmp
```

Команда `mount` также позволяет повторно монтировать раздел, чтобы изменить параметры его монтирования. Предположим, что вы хотите перемонтировать `/dev/`

sdb1 для чтения/записи, но не хотите его размонтировать (например, он в данный момент используется). Примените параметр remount, как показано далее:

```
# mount -t ext3 -o remount,rw /dev/sdb1
```

Монтирование образа диска с помощью loopback

Команда mount способна монтировать образы дисков. Это позволяет скачать файл образа SD-карты или DVD ISO из Интернета и посмотреть, что он содержит, не записывая его на DVD или другой носитель. С помощью образа на жестком диске создайте точку монтирования и используйте параметр -o loop для ее локального монтирования, например:

```
# mkdir /mnt/mydvdimage
# mount -o loop whatever-i686-disc1.iso /mnt/mydvdimage
```


В этом примере создается каталог /mnt/mydvdimage, а затем файл образа диска (whatever-i686-disc1.iso), находящийся в текущем каталоге, монтируется на нем. Теперь можно применить cd в этом каталоге, просмотреть его содержимое и скопировать или использовать его. Эта функция полезна для загруженных образов DVD, с которых нужно установить программное обеспечение, не записывая образ на DVD. Можно также совместно использовать эту точку монтирования через NFS, чтобы установить программное обеспечение с другого компьютера. Чтобы размонтировать образ, введите команду umount /mnt/mydvdimage.

Другие параметры команды mount доступны только для определенных типов файловых систем. Обратитесь к руководству для команды mount, чтобы узнать об этих и других полезных параметрах.

Команда umount

Когда закончите использовать временную файловую систему или захотите временно размонтировать постоянную, примените команду umount. Она отсоединяет файловую систему от точки монтирования в файловой системе Linux. Чтобы использовать команду umount, укажите для нее либо имя каталога, либо имя устройства, как показано в этом примере:

```
# umount /mnt/test
```

Команда размонтирует устройство из точки монтирования /mnt/test. Его можно также размонтировать с помощью такого варианта:

```
# umount /dev/sdb1
```

В целом, лучше использовать имя каталога (/mnt/test), поскольку команда umount не будет выполняться, если устройство смонтировано в нескольких местах сразу. (Все имена устройств начинаются с /dev.)

Если появляется сообщение, что устройство занято (device is busy), значит, запрос на команду umount не удался, потому что либо приложение имеет файл,

открытый на устройстве, либо на нем открыта оболочка с каталогом в качестве текущего каталога. Остановите процессы или перейдите в каталог за пределами устройства, чтобы выполнить запрос `umount`.

Еще один вариант размонтирования занятого устройства — это параметр `-l`. С помощью `umount -l` («ленивое» размонтирование) оно происходит, как только устройство освобождается. Чтобы размонтировать удаленную файловую систему NFS, которая теперь недоступна (например, сервер вышел из строя), можно использовать параметр `umount -f` и сделать это принудительно.

СОВЕТ

Чтобы определить, чем занимается устройство, которое нужно размонтировать, существует полезный инструмент — команда lsof. Введите ее с именем нужного раздела (например, lsof /mnt/test). Выходные данные покажут, какие команды удерживают файлы открытыми в этом разделе. Таким же образом можно использовать команду fuser-v /mnt/test.

Создание файловой системы с помощью mkf

Вы можете создать файловую систему любого поддерживаемого типа на выбранном диске или в разделе с помощью команды `mkfs`. Этот вариант наиболее полезен для создания файловых систем в разделах жесткого диска, но его можно использовать и чтобы создавать файловые системы на USB-накопителях или перезаписываемых DVD.

Перед созданием новой файловой системы убедитесь в следующем.

- Вы разбили диск на разделы так, как нужно (с помощью команды `fdisk`).
- Вы выбрали правильное имя устройства (чтобы не перезаписать свой жесткий диск по ошибке). Например, первый раздел на втором SCSI или USB флеш-накопителе в вашей системе — это `/dev/sdb1`, а третий диск — `/dev/sdc1`.
- Вы размонтировали раздел, если ранее он был смонтирован.

Далее приведены два примера использования команды `mkfs` для создания файловой системы в двух разделах USB-накопителя, расположенных как первый и второй разделы на третьем SCSI-диске (`/dev/sdc1` и `/dev/sdc2`). Первый создает файловую систему xfs, а второй — ext4:

```
# mkfs -t xfs /dev/sdc1
meta-data=/dev/sda3 isize=256            agcount=4, agsize=256825 blks
         =                  sectsz=512    attr=2, projid32bit=1
         =                  crc=0
data     =                  bsize=4096    blocks=1027300, imaxpct=25
         =                  sunit=0       swidth=0 blks
naming   =version 2         bsize=4096    ascii-ci=0 ftype=0
log      =internal log      bsize=4096    blocks=2560, version=2
         =                  sectsz=512    sunit=0 blks, lazy-count=1
realtime =none              extsz=4096    blocks=0, rtextents=0
```

```
# mkfs -t ext4 /dev/sdc2
mke2fs 1.44.6 (5-Mar-2019)
Creating filesystem with 524288 4k blocks and 131072 inodes
Filesystem UUID: 6379d82e-fa25-4160-8ffa-32bc78d410eee
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912
Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done
```

Теперь можете монтировать любую из этих файловых систем (например, `mkdir /mnt/myusb`, `mount /dev/sdc1 /mnt/myusb`), сделать `/mnt/myusb` текущим каталогом (`cd /mnt/myusb`) и создавать в нем файлы.

Управление хранилищем с помощью Cockpit

Большинство описанных в этой главе функций для работы с разделами диска и файловыми системами можно выполнить с помощью веб-интерфейса Cockpit. Запустите Cockpit, откройте веб-интерфейс (`имя хоста: 9090`) и выберите вкладку Storage (Хранилище). На рис. 12.2 показана вкладка Storage (Хранилище) в Cockpit в системе Fedora.

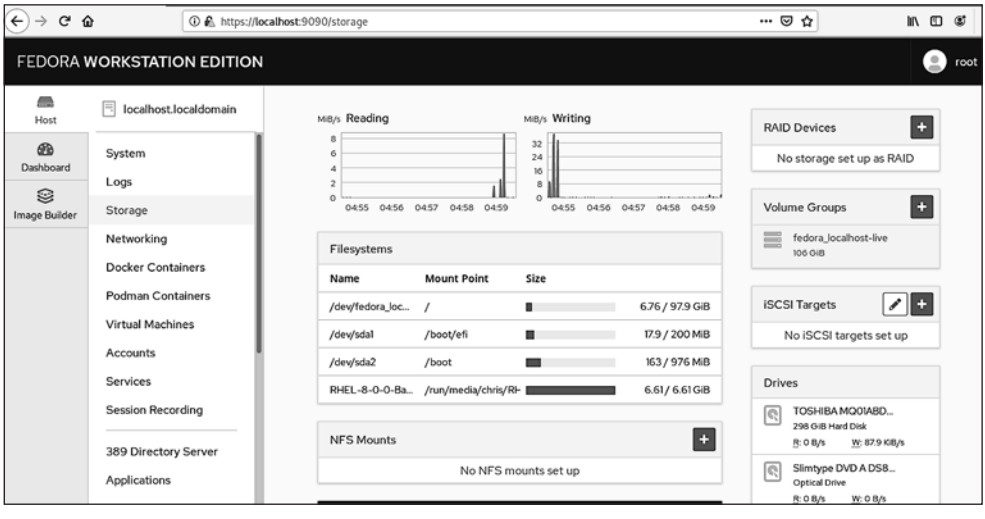


Рис. 12.2. На вкладке Storage (Хранилище) находятся устройства хранения данных, файловые системы и действия

Вкладка Storage (Хранилище) дает полный обзор хранилища вашей системы. Она отображает активность чтения и записи устройствами, обновляя данные каждую минуту, а также показывает локальные файловые системы, хранилища

(включая RAID-устройства и группы томов LVM), удаленно подключенные общие ресурсы NFS и цели iSCSI. Каждый жесткий диск, DVD и другое физическое запоминающее устройство отображаются на вкладке Storage (Хранилище).

Выбрав смонтированную файловую систему, можно увидеть и изменить разбиение на разделы в этой файловой системе. Например, выбрав автоматически смонтированную систему в `/run/media`, вы можете увидеть все разделы для устройства, на котором она находится (`/dev/sda1` и `/dev/sda2`). На рис. 12.3 показано, что в этих двух разделах имеются файловая система ext4 и btrfs — файловая система копирования при записи.

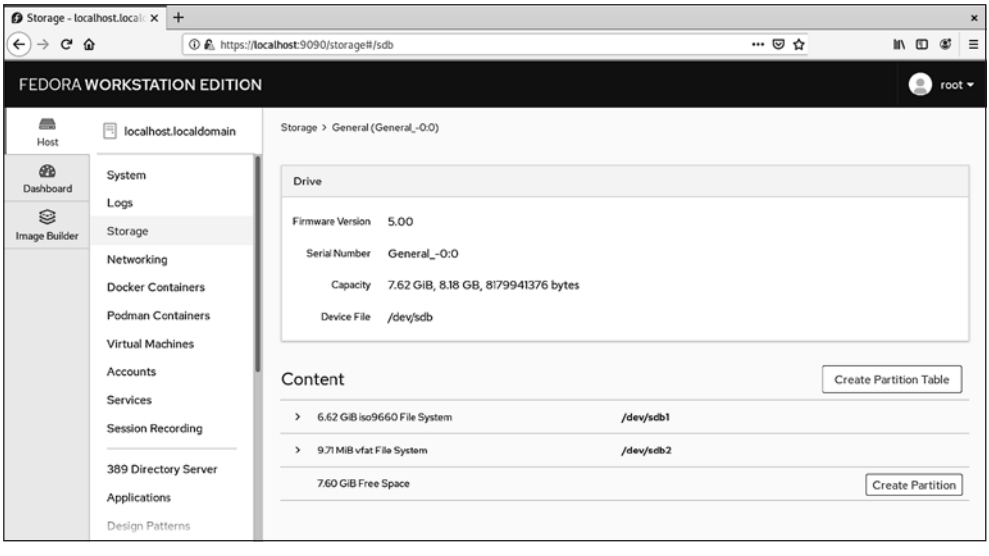


Рис. 12.3. Просмотр и изменение разделов диска для выбранного устройства

В окне информации об устройстве можно переформатировать все устройство хранения (кнопка Create Partition Table (Создать таблицу разделов)) или, если на нем имеется свободное место, добавить новый раздел (кнопка Create Partition (Создать раздел)). На рис. 12.4 показано окно Create Partition Table (Создать таблицу разделов).

Чтобы отформатировать диск или USB-накопитель, измените параметр Erase (Удаление) и разрешите перезапись всех данных на диске, а затем выберите тип разбиения. Нажмите кнопку Format (Форматировать), чтобы размонтировать все смонтированные разделы с диска и создать новую таблицу разделов. После этого можно добавить разделы на запоминающее устройство, выбрав размер, тип файловой системы и шифрование данных (по желанию). Вы даже можете выбрать, где в файловой системе операционной системы монтировать новый раздел. С помощью нескольких щелчков кнопкой мыши можно быстрее и интуитивно понятнее создать макеты дисков, чем используя командную строку.

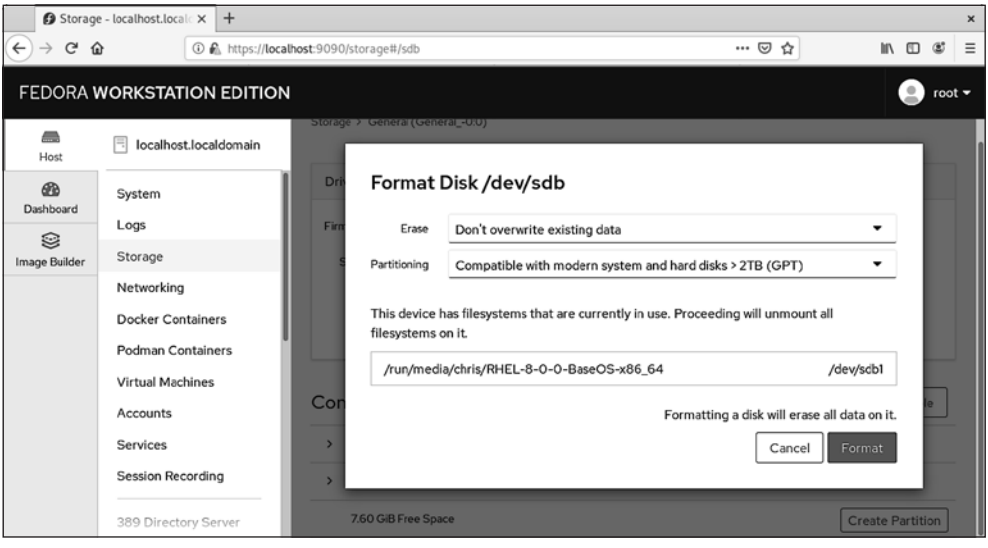


Рис. 12.4. Создание новой таблицы разделов

Резюме

Управление файловыми системами — это важная часть администрирования системы Linux. С помощью таких команд, как `fdisk`, можно просматривать и изменять разделы диска. Применив команду `mkfs`, можно добавить в разделы файловые системы. После создания файловые системы можно монтировать и размонтировать с помощью команд `mount` и `unmount` соответственно.

Менеджер логических томов (LVM) позволяет использовать более мощный и гибкий способ управления дисковыми разделами. С его помощью можно создавать пулы хранения, называемые группами томов, которые позволяют увеличивать и уменьшать логические тома, а также увеличивать размер групп томов добавлением физических томов.

Существует более интуитивно понятный способ работы с устройствами хранения — веб-интерфейс Cockpit, который позволяет просматривать и настраивать хранилища в системе Linux. С помощью веб-интерфейса вы можете увидеть как локальное, так и сетевое хранилище, а также переформатировать диски и изменить разделы на них.

В главе 13 представлены основные принципы, знать которые необходимо для того, чтобы стать системным администратором и расширить эти навыки до умения управлять сетевыми серверами. В этой главе содержатся сведения об установке и защите серверов, а также об управлении ими.

Упражнения

Используйте эти упражнения, чтобы проверить свои знания о создании разделов диска, управлении логическими томами и работе с файловыми системами. Вам понадобится пустой USB-накопитель объемом не менее 1 Гбайт.

Задачи подходят для систем Fedora или Red Hat Enterprise Linux (некоторые сработают и в других системах Linux). Если затрудняетесь с решением заданий, воспользуйтесь ответами к упражнениям, приведенными в приложении Б (хотя Linux позволяет решать задачи разными способами).

1. Просмотрите системный журнал в окне Terminal (Терминал) с помощью подходящей команды от имени суперпользователя и подключите USB-накопитель. Определите имя устройства.
2. Выполните команду, чтобы вывести список таблиц разделов для флеш-накопителя USB.
3. Удалите все разделы на USB-накопителе, сохраните изменения и убедитесь, что они были внесены как в таблицу разделов диска, так и в ядро Linux.
4. Добавьте три раздела на USB-накопитель: 100 Мбайт для раздела Linux, 200 Мбайт для раздела подкачки и 500 Мбайт для LVM-раздела. Сохраните изменения.
5. Добавьте файловую систему ext4 в раздел Linux.
6. Создайте точку монтирования с именем `/mnt/mypart` и смонтируйте в ней раздел Linux.
7. Подключите раздел подкачки и включите его, чтобы сразу же добавить дополнительное пространство.
8. Создайте группу томов `abc` из раздела LVM, создайте логический том объемом 200 Мбайт из группы `data`, добавьте раздел VFAT, а затем временно смонтируйте логический том в новом каталоге с именем `/mnt/test`. Проверьте, правильно ли он смонтирован.
9. Расширьте логический том с 200 до 300 Мбайт.
10. Правильно и безопасно удалите USB-накопитель из компьютера: размонтируйте раздел Linux, выключите раздел подкачки, размонтируйте логический том и удалите группу томов с USB-накопителя.

Часть IV

Администрирование серверов в Linux

В этой части

13 Администрирование серверов

В этой главе

- Администрирование серверов.
- Соединение серверов по сети.
- Ведение журнала удаленно и локально.
- Отслеживание серверных систем.
- Управление серверами на предприятии.

Некоторые функции системного администрирования необходимы даже в настольной системе (установка программного обеспечения, настройка принтеров и т. д.), однако при настройке системы Linux в качестве сервера появляется много новых задач. Особенно если настраиваемый сервер находится в открытом доступе в Интернете, где он может быть перегружен запросами от «хороших» пользователей, в то время как нужно постоянно быть начеку в ожидании атак «плохих».

Для систем Linux доступны десятки различных типов серверов. Большинство серверов обслуживают данные удаленных клиентов, но есть и такие, которые обслуживают локальную систему (например, те, что собирают сообщения журнала или запускают задачи обслуживания в установленное время с помощью функции `cron`). Многие серверы представлены процессами, которые непрерывно работают в фоновом режиме и отвечают на поступающие к ним запросы. Такие процессы называются *демонами*.

Серверы служат для выполнения определенных задач. Данные, которые они обслуживают, могут включать в себя веб-страницы, файлы, информацию о базе данных, электронную почту и множество других типов информации. Перечислю дополнительные навыки, которые должен иметь администратор сервера.

- **Удаленный доступ.** Чтобы задействовать настольную систему, пользователю необходимо находиться прямо перед ней. Серверные системы, напротив, как правило, размещаются под замком в специальных хранилищах с контролиру-

емыми температурой и влажностью. Чаще всего после установки физических компьютеров их администрирование осуществляется с помощью средств удаленного доступа. В этом случае графический интерфейс недоступен, поэтому для выполнения таких действий, как удаленный вход в систему, удаленное копирование и удаленное выполнение, приходится полагаться на средства командной строки или браузерные интерфейсы. Наиболее распространенные из этих инструментов построены на сетевом протоколе Secure Shell (SSH).

- **Усиленная защита.** Сервер должен иметь возможность принимать запросы от удаленных пользователей и систем. В отличие от настольных систем, которые могут просто закрыть все сетевые порты, разрешающие входящие запросы на доступ, сервер должен быть уязвимым, чтобы доступ к некоторым портам был свободен. Вот почему администратору сервера важно уметь открывать порты для необходимых служб и блокировать доступ другим. Для защиты служб применяются такие инструменты, как `iptables` и `firewalld` (инструменты брандмауэра), а также Security Enhanced Linux (ограничивает ресурсы, к которым служба может получить доступ из локальной системы).
- **Непрерывное наблюдение.** Обычно настольный компьютер или ноутбук выключен, если не используется, а вот серверы чаще всего остаются включенными 24 часа 7 дней в неделю 365 дней в году. Поскольку администратор не может находиться рядом с каждым сервером и постоянно контролировать их, он должен настроить инструменты для наблюдения за ними, сбора сообщений журнала и даже пересылки подозрительных сообщений на нужный адрес электронной почты. Вы можете включить отчеты активности системы для круглосуточного сбора данных о применении процессора, памяти, о сетевой активности и доступе к диску.

В этой главе я опишу основные инструменты и методы, которые необходимо знать и уметь использовать для администрирования удаленных серверов Linux. Вы научитесь применять инструменты SSH для безопасного доступа к серверу, передачи данных туда и обратно и даже запуска удаленных рабочих столов или графических приложений в вашей локальной системе. Вы также научитесь удаленно вести журнал и получать отчеты о работе системы, непрерывно наблюдая за ней.

Как работает администрирование сервера

Независимо от того, устанавливаете ли вы файловый сервер, веб-сервер или любой другой сервер, доступный в системах Linux, большинство действий для запуска сервера одинаковы. После запуска изменения в действиях появляются в областях конфигурации и настройки.

В последующих главах я опишу конкретные серверы и их различия. В каждой из глав, связанных с сервером, применяются те же основные шаги для запуска сервера и того, как сделать его доступным для других пользователей.

Шаг 1. Установка сервера

Большинство серверных программ не предустановлены в системе Linux, однако любая система Linux дает возможность установить пакеты программного обеспечения, необходимые для всех основных типов доступных серверов.

Бывает, что несколько пакетов программного обеспечения, связанных с определенным типом сервера, собирают в группы пакетов (еще их называют *коллекциями пакетов*). Если же нужное программное обеспечение не собрано в группу пакетов, нужно просто установить подходящие пакеты по отдельности. Типы серверных пакетов в Fedora и пакеты, доступные в каждой категории, таковы.

- **Сервер ведения логов.** Служба `rsyslog` позволяет локальной системе собирать сообщения журнала из различных компонентов системы. Он может выступать также в качестве удаленного сервера ведения журнала, собирая сообщения, отправленные с других серверов ведения логов. (Служба `rsyslog` будет описана далее в этой главе.) В последних версиях систем Ubuntu, Fedora и RHEL сообщения журнала собирают в журнале `systemd`, который может быть использован службой `rsyslog` или отображен локально командой `journalct`.
- **Сервер печати.** Общая служба печати UNIX (пакет `cups`) чаще всего применяется для предоставления функций сервера печати в системах Linux. При установке пакета CUPS доступно графическое администрирование (`system-config-printer`), драйверы принтеров (`foomatic`, `hpijs` и др.) также доступны при установке CUPS. (См. главу 16 «Настройка сервера печати».)
- **Веб-сервер.** Веб-сервер Apache (пакет `httpd` или `apache2`) — это программное обеспечение, которое чаще всего используется для обслуживания веб-страниц (HTTP-данных). Связанные пакеты включают модули, помогающие обслуживать определенные типы данных (Perl, Python, PHP- и SSL-соединения). Кроме того, существуют пакеты соответствующей документации (`httpd-manual`), инструменты для наблюдения за веб-данными (`webalizer`) и для предоставления прокси-служб (`squid`). (См. главу 17 «Настройка веб-сервера».)
- **Сервер FTP (протокол передачи файлов).** Демон Very Secure FTP (пакет `vsftpd`) — это FTP-сервер по умолчанию, используемый в системах Fedora и RHEL. Кроме того, FTP-сервер включает в себя пакеты `proftpd` и `pureftpd`. (См. главу 18 «Настройка FTP-сервера».)
- **Файловый сервер для Windows.** Сервер Samba (пакет `samba`) позволяет системе Linux выступать в качестве сервера файлов и печати для Windows. (См. главу 19 «Настройка Samba-сервера».)
- **Файловый сервер NFS.** Сетевая файловая система (Network File System, NFS) — это стандартная функция Linux и UNIX, которая предоставляет общие каталоги другим системам по сети. Пакет `nfs-utils` включает в себя службы NFS и связанные с ними команды. (См. главу 20 «Настройка NFS-сервера».)
- **Почтовый сервер.** Пакеты позволяют настраивать почтовый сервер, иногда называемый сервером агента доставки почты (Mail Transport Agent, MTA). Существует несколько вариантов почтовых серверов — `sendmail`, `postfix`

(по умолчанию в системах Fedora и RHEL) и `exim`. Связанные пакеты, например `dovecot`, позволяют почтовому серверу доставлять электронную почту клиентам.

- **Сервер каталогов.** Пакеты этой категории предоставляют удаленные и локальные службы аутентификации. К ним относятся Kerberos (`krb5-server`), LDAP (`openldap-server`) и NIS (`ypserv`).
- **Сервер DNS.** Доменная служба Berkeley Internet Name Domain (пакет `bind`) предоставляет программное обеспечение, необходимое для настройки сервера для преобразования имен узлов в IP-адреса.
- **Сервер NTP.** Пакеты `ntpd` и `chronyd` позволяют включить протокол сетевого времени для синхронизации системных часов с часами публичных или частных серверов NTP.
- **Сервер SQL.** Служба PostgreSQL (пакеты `postgresql` и `postgresql-server`) — это объектно-реляционная система управления базами данных. Связанные пакеты предоставляют документацию PostgreSQL и связанные с ней инструменты. Служба MySQL (пакеты `mysql` и `mysql-server`) — еще один популярный сервер баз данных SQL с открытым исходным кодом. Разработанная сообществом Linux на основе MySQL служба MariaDB вытеснила MySQL во многих дистрибутивах Linux.

Шаг 2. Настройка сервера

Большинство пакетов серверного программного обеспечения устанавливаются с настройками по умолчанию, которые больше ориентированы на безопасность, чем на немедленное и полное применение. Рассмотрим список того, о чем следует подумать перед настройкой сервера.

Использование файлов конфигурации

Традиционно серверы Linux настраиваются путем редактирования простых текстовых файлов в каталоге `/etc` (или в подкаталогах). Обычно есть первичный файл конфигурации, иногда появляется связанный каталог конфигурации, в котором файлы, заканчивающиеся на `.conf`, добавляют в основной файл конфигурации.

Пакет `httpd` (веб-сервер Apache) — это пакет сервера, который имеет основной файл конфигурации и каталог, в который можно поместить другие файлы конфигурации и добавить их в состав службы. Основным файлом конфигурации в системах Fedora и RHEL является файл `/etc/httpd/conf/httpd.conf`. Каталог конфигурации — это `/etc/httpd/conf.d/`.

После установки пакета `httpd` и связанных с ним пакетов вы увидите его файлы в каталоге `/etc/httpd/conf.d/`, они помещаются туда с помощью различных пакетов: `mod_ssl`, `mod_perl` и др. Таким образом, дополнительные пакеты могут иметь информацию о своей конфигурации, включенную в сервер `httpd`, и при этом пакет не будет запускать сценарии редактирования основного файла `httpd.conf`.

Единственным недостатком обычных текстовых файлов конфигурации является то, что нет возможности немедленно проверить все ошибки, как происходит при использовании графических инструментов администрирования. В таком случае необходимо либо запустить тестовую команду (если служба ее имеет), либо попытаться запустить саму службу, чтобы увидеть, нет ли каких-то проблем с отредактированным файлом конфигурации.

СОВЕТ

Вместо того чтобы редактировать файлы конфигурации с помощью редактора vi, возьмите редактор vim. Команда vim помогает находить ошибки в файле конфигурации прямо во время редактирования.

Команда vim распознает форматы многих файлов конфигурации (passwd, httpd.conf, fstab и др.). Если вы сделаете ошибку, введете недопустимый термин или параметр в один из этих файлов или каким-либо образом измените формат, изменится цвет текста. Например, если в файле /etc/fstab изменить параметр defaults на default, цвет слова изменится.

Проверка конфигураций по умолчанию

Большинство программных пакетов для серверов в системах Fedora и RHEL устанавливаются с минимальной конфигурацией и больше ориентированы на безопасность, чем на немедленное применение. При установке пакета программного обеспечения некоторые дистрибутивы Linux спрашивают, в какой каталог нужно установить пакет или какой учетной записью будет управляться сервер.

Поскольку пакеты RPM предназначены для автоматической установки, у пользователя нет особого выбора его настроек. Файлы устанавливаются в определенных местах, управлять ими могут определенные учетные записи, и при запуске службы доступ к ней вполне может быть ограничен. Пользователь может настроить программное обеспечение и сделать сервер полностью функциональным уже после установки пакета.

Почтовые серверы (пакеты `sendmail postfix`) и DNS-серверы (пакет `bind`) являются примерами серверов, которые устанавливаются с ограниченным функционалом. Оба они устанавливаются в конфигурации по умолчанию и запускаются при перезагрузке системы. Однако оба видят запросы только на локальном узле. Таким образом, пока вы не настроите эти серверы, пользователи, которые не вошли на ваш локальный сервер, не смогут отправлять почту на почтовый сервер или задействовать ваш компьютер в качестве общедоступного DNS-сервера.

Шаг 3. Запуск сервера

Большинство служб, устанавливаемых в Linux, настроены на запуск при загрузке системы, а затем на непрерывную обработку запросов, пока система не будет выключена. Существует два основных средства для управления службами: `systemd` (используется в настоящее время в дистрибутивах RHEL, Ubuntu и Fedora) и скрипты SystemVinit (применяются в системах Red Hat Enterprise Linux версии RHEL 6.x).

Независимо от того, какое средство используется в вашей системе Linux, ваша задача — установить, хотите ли вы, чтобы служба появлялась при загрузке системы, а также нужно ли запускать, останавливать и перезагружать службу по мере необходимости (возможно, при загрузке новых файлов конфигурации или временном запрете на доступ к службе). Команды для выполнения этих задач описаны в главе 15 «Запуск и остановка служб».

Бóльшая часть служб реализуется в качестве демон-процессов. Вот что нужно знать об этих процессах.

- **Права доступа пользователей и групп.** Демоны часто выполняются от имени пользователей и групп, отличных от суперпользователя. Например, демон `httpd` работает как пользователь `apache`, а демон `ntpd` — как пользователь `ntp`. Причина этого заключается в том, что, если кто-то взломает эти демоны, они все равно не получат прав на доступ к файлам за пределами данных служб.
- **Файлы конфигурации демонов.** Чаще всего у службы есть свой файл конфигурации для демона, который хранится в каталоге `/etc/sysconfig`. Он отличается от файла конфигурации службы тем, что его задача заключается в передаче аргументов самому серверному процессу, а не в настройке службы. Например, параметры, заданные в файле `/etc/sysconfig/rsyslogd`, передаются демону `rsyslogd` при его запуске. Допустим, можно указать демону выводить дополнительную отладочную информацию или принимать сообщения удаленного журнала. См. справочную страницу нужной службы (например, `man rsyslogd`), чтобы узнать, какие параметры для нее поддерживаются.
- **Номер порта.** Пакеты данных поступают в систему и из нее переходят по сетевым интерфейсам через порты для каждого поддерживаемого протокола (обычно UDP или TCP). Большинство стандартных служб имеют определенные номера портов, к которым подключаются демоны и клиенты. Если не нужно скрывать местоположение службы, то номера портов не изменяются. При переходе к защите службы необходимо убедиться, что порт к службе открыт в брандмауэре (см. главу 25 «Защита Linux в сети», чтобы узнать о брандмауэрах `iptables` и `firewalld`). Кроме того, если изменить порт, на котором прослушивается служба, а система SELinux при этом находится в принудительном режиме, она может запретить демону прослушивать этот порт (см. главу 24 «Повышенная безопасность с технологией SELinux», чтобы узнать о системе SELinux).

ПРИМЕЧАНИЕ

Причина изменения номеров портов в сервисе основывается на принципе «безопасность через неясность» (security by obscurity). Например, служба sshd чаще всего становится целью для тех, кто пытается взломать систему, угадывая логины и пароли на TCP-порте 22.

Я знаю, что некоторые пользователи меняли свой интернет-сервис sshd, чтобы задействовать другой номер порта (возможно, какой-то неиспользуемый порт с большим номером). Затем они просили друзей или коллег войти в их компьютер из SSH, указав новый номер порта. Суть заключается в том, что сканеры портов, желающие проникнуть в систему, с меньшей вероятностью просканируют неиспользуемый порт.

Не все службы работают непрерывно, как это делают демоны. Некоторые устаревшие службы UNIX запускались по требованию с помощью суперсервера `xinetd`. Другие службы просто запускаются один раз при каждом запуске. Третьи запускаются определенное количество раз, и демон `crond` видит, что служба была настроена для запуска в определенное время.

В последние годы службы `xinetd` в системах RHEL и Fedora, такие как `telnet` и `tftp`, были преобразованы в службы `systemd`. Ряд служб, включая `cockpit`, используют сокеты `systemd` для достижения тех же результатов.

Шаг 4. Защита сервера

Открытая система позволяет удаленным пользователям получить доступ к ней по сети, и это не то решение, которое принимается легко и просто. Взломщики по всему миру запускают программы для сканирования уязвимых серверов, которые они могут использовать, чтобы хранить свои данные или добавить ресурсы к своей вычислительной мощности. К счастью, в системах Linux существуют меры защиты серверов и служб от атак злоумышленников.

Общие методы обеспечения безопасности описаны в следующих разделах. О защите системы более подробно рассказано в части V «Методы обеспечения безопасности в Linux».

Защита пароля

Качественные надежные пароли — это первая линия обороны в защите системы Linux. Если пользователь может войти на ваш сервер через протокол `ssh` как суперпользователь с простейшим паролем, ожидайте, что сервер будет в итоге взломан. Метод защиты состоит в том, чтобы запретить прямой вход в систему с помощью пароля `root` и потребовать, чтобы все входили в систему как обычные пользователи, а затем применяли команды `su` или `sudo` для получения прав суперпользователя.

Можно также задействовать подключаемый модуль аутентификации (pluggable authentication module, PAM) для установки количества неудачных попыток входа в систему, которые может сделать пользователь, прежде чем будет заблокирован. Модули PAM включают в себя и другие функции для блокировки аутентификации на сервере Linux. (Описание модулей PAM см. в главе 23 «Продвинутые методы обеспечения безопасности».)

Конечно, можно полностью обойти пароли, требуя аутентификации с открытым ключом. Чтобы применить этот тип аутентификации, необходимо убедиться, что любой пользователь, который получает доступ к серверу, имеет открытый ключ, скопированный на сервер (например, через `ssh-copy-id`). Затем он может задействовать команды `ssh`, `scp` или связанные команды для доступа к этому серверу без ввода пароля. (См. подраздел «Аутентификация на основе ключей (без пароля)» далее в этой главе.)

Брандмауэры

Служба брандмауэра `iptables` может отслеживать каждый пакет, поступающий от сетевых интерфейсов компьютера и в обратную сторону, и реагировать на него. С помощью брандмауэра `iptables` можно отбросить или отклонить каждый пакет, запрашивающий службы в системе, за исключением подключенных. Кроме того, можно указать брандмауэру `iptables` разрешать запросы на обслуживание только с определенных IP-адресов («хорошие» пользователи) и не разрешать запросы с других («плохие» пользователи).

В последних версиях дистрибутивов RHEL и Fedora служба `firewalld` добавила больше функциональности в работу брандмауэров Linux. С помощью функции `firewalld` можно не только вставлять правила брандмауэра в ядро, но и создавать свои правила, разделяя их на зоны, и быстро изменять их в соответствии с событиями, происходящими в системе.

В каждой из глав, посвященных серверам, я расскажу, какие порты должны быть открыты для доступа к службам. Описание того, как работают `iptables` и `firewalld`, включено в главу 25 «Защита Linux в сети».

Средства TCP Wrappers

Средства *TCP Wrappers*, использующие файлы `/etc/hosts.allow` и `/etc/hosts.deny`, которые разрешают и запрещают доступ различными способами к выбранным службам, применялись в основном для защиты устаревших служб UNIX и теперь считаются недостаточно безопасными. Программа TCP Wrapper (каталог `/usr` на `/sbin/tcpd`) — это то единственное общее в системах, в которых задействуется служба `xinetd`, а файлы `/etc/hosts.allow` и `/etc/hosts.deny`, которые программа TCP Wrapper проверяла перед предоставлением доступа к сетевым службам, теперь проверяются специальными демонами. Параметр конфигурации в файлах конфигурации для этих демонов часто обозначается как поддержка TCP Wrapper.

Система SELinux

Fedora, Red Hat Enterprise Linux и другие дистрибутивы Linux поставляются с подключенной функцией Security Enhanced Linux (SELinux) в принудительном режиме (`Enforcing`). Хотя режим по умолчанию не оказывает большого влияния на большинство приложений, запускаемых в Linux, он значительно влияет на бóльшую часть основных служб.

Основная функция SELinux заключается в защите содержимого системы Linux от процессов, запущенных в системе. Другими словами, SELinux гарантирует, что веб-сервер, FTP-сервер, сервер Samba или DNS-сервер могут получить доступ только к ограниченному набору файлов в системе (определяется содержимым файлов), и разрешает лишь ограниченный набор функций (определяется логическими значениями и ограниченным доступом к порту).

Более подробно о том, как использовать SELinux, можно узнать в главе 24 «Повышенная безопасность с технологией SELinux».

Параметры безопасности в файлах конфигурации

В файлах конфигурации большинства служб содержатся значения, которые можно задать для дальнейшей защиты службы. Например, для файловых и веб-серверов можно ограничить доступ к определенным файлам или данным на основе имени пользователя, имени узла, IP-адреса или других атрибутов.

Шаг 5. Мониторинг работы сервера

Поскольку невозможно постоянно контролировать все службы, необходимо установить инструменты мониторинга, чтобы следить за работой серверов. Инструменты, которые можно применять для мониторинга серверов, описаны в следующих пунктах.

Настройка журналирования

С помощью службы `rsyslog` (демон `rsyslogd`) можно собирать важную информацию и ошибки в файлы журналов множества различных служб. По умолчанию в дистрибутиве RHEL сообщения журнала из приложений направляются в файлы журнала в каталоге `/var/log`. Для дополнительной безопасности и удобства сообщения журнала могут быть направлены также на централизованный сервер, в котором находятся все записи журнала.

Для работы со службой `rsyslog` и управления сообщениями журнала доступно несколько различных программных пакетов. Функция `logwatch` сканирует файлы журнала каждую ночь и отправляет важную информацию, собранную из них, на выбранный адрес электронной почты. Функция `logrotate` создает резервные копии файлов журналов в сжатые архивы, когда журналы достигают определенного размера или когда с момента предыдущего резервного копирования проходит заданное количество времени.

Функции настройки системного журналирования и управления им описаны в разделе «Настройка журнала системы» далее в этой главе.

Запуск отчетов активности системы

Средство `sar` (пакет `sysstat`) можно настроить для наблюдения за действиями в системе, например за использованием памяти и процессора, задержкой диска, сетевой активностью и другими утечками ресурсов. По умолчанию функция `sar` запускает программу `sadc` каждые несколько минут днем и ночью и собирает данные. Просмотр этих данных позже может помочь выяснить, где и когда повышается спрос на ресурсы в системе. Функция `sar` описана в разделе «Проверка системных ресурсов с помощью функции sar» далее в этой главе.

Просмотр текущей активности с помощью интерфейса Cockpit

С помощью программы Cockpit можно наблюдать за активностью системы в режиме реального времени. Откройте браузер, чтобы отобразить интерфейс Cockpit (localhost:9090). Он позволяет в режиме реального времени видеть процент использования процессора, потребление памяти и подкачки, сколько данных записывается на диск и с диска (дисковый ввод-вывод), а также отследить сетевой трафик по мере его сбора и отображения на экране. На рис. 13.1 показана вкладка Система (System) программы Cockpit, отображающая данные об активности.

Рис. 13.1. Вход в интерфейс Cockpit

Обновление программного обеспечения

При обнаружении и исправлении ошибок в системе безопасности необходимо убедиться, что на серверах установлены актуальные пакеты программного обеспечения, которые содержат исправления. Для чрезвычайно важных серверов самый безопасный и эффективный способ обновления — использовать подписку на системы Red Hat Enterprise Linux, а затем развертывать полученные обновления пакетов безопасности в системе.

Чтобы поддерживать персональные серверные и настольные системы в актуальном состоянии, существуют различные графические инструменты для добавления программного обеспечения и проверки наличия обновлений. Можно также применить команду `yum` для проверки и установки всех пакетов, доступных для систем RHEL или Fedora (введите `dnf update` или `yum update`).

Проверка системы на наличие взломов

Чтобы проверить файловую систему на наличие взломов, можно запустить команду `rpm-V`, чтобы узнать, не были ли в системе изменены какие-либо команды, файлы документов или файлы конфигурации. Чтобы больше узнать о команде `rpm -V`, обратитесь к главе 10 «Управление программами».

Теперь вы знаете, как выполняется настройка сервера Linux. В следующих разделах главы основное внимание будет уделено инструментам, необходимым для доступа к серверным системам Linux, их защиты и обслуживания.

Проверка и настройка серверов

Далее рассмотрим большинство аспектов, которые нужно проверять, будучи администратором сервера Linux. Имейте в виду, что сейчас многие серверы в крупных центрах обработки данных развертываются и управляются более крупными платформами. Поэтому, прежде чем вносить в сервер какие-либо изменения, узнайте, как им управлять. Изменения могут быть перезаписаны автоматически, если вы изменили определенное состояние этой системы.

Управление удаленным доступом с помощью службы Secure Shell

Инструменты Secure Shell — это набор клиентских и серверных приложений, которые позволяют реализовывать базовую связь между клиентскими компьютерами и сервером Linux. Эти инструменты включают в себя `ssh`, `scp`, `sftp` и многие другие команды. Поскольку связь между сервером и клиентами зашифрована, эти инструменты более безопасны, чем аналогичные, но устаревшие. Например, вместо старых команд удаленного входа, таких как `telnet` или `rlogin`, можно использовать команду `ssh`. Команда `ssh` также может заменить устаревшие команды удаленного выполнения, например `rsh`. Команды удаленного копирования, к примеру `rcp`, можно заменить безопасными командами, такими как `cp` и `rsync`.

С помощью инструментов Secure Shell и процесс аутентификации, и все последующие коммуникации шифруются. Коммуникации из команды `telnet` и более старой команды `"r"` свободно предоставляют пароли и все данные. Сегодня `telnet` и подобные команды должны применяться только для тестирования доступа к удаленным портам, предоставления общедоступных услуг, таких как загрузка PXE, или выполнения других задач, которые не раскрывают личные данные.

ПРИМЕЧАНИЕ

Для более глубокого изучения методов шифрования обратитесь к главе 23.

Большинство систем Linux обеспечивают безопасность оболочки клиента, а многие из них — и безопасность оболочки сервера. Например, если вы используете дистрибутив Fedora или RHEL, клиентские и серверные программные пакеты, содержащие инструменты `ssh`, — это пакеты `openssh`, `openssh-clients` и `openssh-server`. Выглядит это следующим образом:

```
# yum list installed | grep openssh
...
openssh.x86_64 7.9p1-5.fc30 @anaconda
openssh-clients.x86_64 7.9p1-5.fc30 @anaconda
openssh-server.x86_64 7.9p1-5.fc30 @anaconda
```

В дистрибутиве Ubuntu установлен только пакет `openssh-clients`. Он включает в себя все функции пакета `openssh`. Если нужно установить сервер, используйте команду `sudo apt-get install openssh-server`:

```
$ sudo dpkg --list | grep openssh
openssh-client/bionic-updates,bionic-security,now 1:7.6p1-4ubuntu0.3 amd64
[installed]
    secure shell (SSH) client, for secure access to remote machines
openssh-client-ssh1/bionic 1:7.5p1-10 amd64
    secure shell (SSH) client for legacy SSH1 protocol

openssh-sftp-server/bionic-updates,bionic-security,now 1:7.6p1-4ubuntu0.3
amd64 [installed]
    secure shell (SSH) sftp server module, for SFTP access from remote
machines
$ sudo apt-get install openssh-server
```

Запуск службы openssh-server

Системы Linux, которые поставляются с уже установленным пакетом `openssh-server`, не всегда настроены на его автоматический запуск. Управление службами Linux (см. главу 15 «Запуск и остановка служб») может различаться в зависимости от различных дистрибутивов. В табл. 13.1 показаны команды, определяющие, что `ssh`-демон сервера — `sshd` — запущен и работает.

Таблица 13.1. Команды, определяющие состояние демона sshd

Дистрибутив	Команда
RHEL 6	chkconfig --list sshd
Fedora и RHEL версии 7 и позже	systemctl status sshd.service
Ubuntu	systemctl status ssh.service

Если демон `sshd` в данный момент не запущен, его можно запустить, выполнив одну из команд, перечисленных в табл. 13.2. Для их работы требуются права суперпользователя.

Таблица 13.2. Команды для запуска sshd

Дистрибутив	Команда
RHEL 6	service sshd start
Fedora и RHEL версии 7 и позже	systemctl start sshd.service
Ubuntu	systemctl start ssh.service

Команды в табл. 13.2 запускают только службу `ssh` или `sshd`. Они не настраивают ее на автоматический запуск при загрузке. Чтобы убедиться, что служба сервера настроена на автоматический запуск, необходимо задействовать одну из команд из табл. 13.3 с правами суперпользователя.

Таблица 13.3. Команды для автоматического запуска службы sshd

Дистрибутив	Команда
RHEL 6	Chkconfig sshd on
Fedora и RHEL версии 7 и позже	systemctl enable sshd.service
Ubuntu	systemctl enable ssh.service

При установке пакета `openssh-server` на Ubuntu демон `sshd` настраивается на автоматический запуск при загрузке. Поэтому для сервера Ubuntu не обязательно выполнять команду из табл. 13.3.

Измените настройки брандмауэра, чтобы разрешить пакету `openssh-client` доступ к порту 22 (брандмауэры описаны в главе 25 «Защита Linux в сети»). После запуска службы и правильной настройки брандмауэра вы сможете использовать клиент `ssh` для доступа клиентов к вашей системе через `ssh`-сервер.

Любые дальнейшие настройки действий демона `sshd` обрабатываются в файле `/etc/ssh/sshd_config`. Как минимум можно установить параметр `PermitRootLogin` в значение `no`, чтобы никто не мог удаленно войти в систему как суперпользователь:

```
# grep PermitRootLogin /etc/ssh/sshd_config
PermitRootLogin no
```

После изменения файла `sshd_config` перезапустите службу `sshd`. Если вы задействуете `ssh` для входа в эту систему с удаленного клиента, необходимо сделать это от имени обычного пользователя, а затем применить команду `sudo`, чтобы стать суперпользователем.

Инструменты SSH-клиента

Для работы со службой SSH было создано много инструментов доступа к удаленным системам Linux. Наиболее часто используемым из них является команда `ssh` для удаленного входа в систему и удаленного выполнения других задач. Такие команды, как `scp` и `rsync`, могут копировать один или несколько файлов одновре-

менно между клиентскими и серверными системами SSH. Команда `sftp` предоставляет интерфейс, подобный FTP-серверу, для обхода удаленной файловой системы, а также получения файлов и их размещения между системами в интерактивном режиме.

По умолчанию все связанные с SSH инструменты аутентифицируются с помощью стандартных имен пользователей и паролей Linux через зашифрованные соединения. Поддерживает SSH и аутентификацию на основе ключей без ввода пароля между клиентами и серверами SSH, как описано в подразделе «Аутентификация на основе ключей (без пароля)» далее в этом разделе.

Удаленный вход в систему с помощью команды ssh

Используйте команду `ssh` с другого компьютера Linux, чтобы проверить, сможете ли вы войти в систему Linux, на которой работает служба `sshd`. Вы будете чаще всего применять именно эту команду, чтобы получить доступ к оболочке на серверах, которые настраиваете.

Войдите на свой сервер Linux из другой системы Linux с помощью команды `ssh`. (Если нет другой системы Linux, можете имитировать это действие, введя команду `localhost` вместо IP-адреса и войдя в систему как локальный пользователь.) Далее приведен пример удаленного входа в учетную запись `johndoe` по адресу 10.140.67.23:

```
$ ssh johndoe@10.140.67.23
The authenticity of host '10.140.67.23 (10.140.67.23)'
    can't be established.
RSA key fingerprint is
    a4:28:03:85:89:6d:08:fa:99:15:ed:fb:b0:67:55:89.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.140.67.23' (RSA) to the
    list of known hosts.
johndoe@10.140.67.23's password: *********
```

При первом входе в удаленную систему с помощью команды `ssh` система попросит вас подтвердить, действительно ли вы хотите подключиться. Введите `yes` и нажмите клавишу Enter. Когда получите приглашение командной строки, введите пароль пользователя.

При вводе варианта `yes` для продолжения вы принимаете открытый ключ удаленного хоста. В этот момент ключ загружается в файл клиента `~/.ssh/known_hosts`. Теперь данные, которыми обмениваются эти две системы, могут быть зашифрованы и расшифрованы с помощью асимметричного шифрования RSA (см. главу 23 «Продвинутые методы обеспечения безопасности»).

Войдя в удаленную систему, можно начинать вводить команды оболочки. Соединение будет работать так же, как и обычный вход в систему. Единственная разница заключается в том, что данные шифруются по мере их перемещения по сети.

По окончании введите `exit`, чтобы завершить удаленное соединение. Соединение прерывается, и вы возвращаетесь в командную строку в своей локальной

системе. (Если локальная оболочка не возвращается после выхода из удаленной оболочки, введите любой текст, это поможет прервать соединение.)

```
$ exit
logout
Connection to 10.140.67.23 closed
```

После прерывания удаленного подключения к системе в вашем локальном системном подкаталоге появится файл `~.ssh/known_hosts`. Он содержит открытый ключ удаленного хоста вместе с его IP-адресом. Открытый и закрытый ключи вашего сервера хранятся в каталоге `/etc/ssh`:

```
$ ls .ssh
known_hosts
$ cat .ssh/known_hosts
10.140.67.23 ssh-rsa
AAAAB3NzaC1yc2EAAAABIwAAAQEAoyfJK1YwZhNmpHE4yLPZAZ9ZNEdRE7I159f3I
yGiH21Ijfqs
NYFR10ZlBLlYyTQi06r/9O19GwCaJ753InQ8FWHW+OOYOG5pQmghhn
/x0LD2uUb6egOu6zim1NEC
JwZf5DWkKdy4euCUEMSqADh/WYeuOSoZ0pp2IAVCdh6
w/PIHMF1HVR069cvdv+OTL4vD0X8llSpw
0ozqRptz2UQgQBBbBjK1RakD7fY1TrWv
NQhYG/ugt gPaY4JDYeY6OBzcadpxZmf7EYUw0ucXGVQ1a
NP/erIDOQ9rA0YNzCRv
y2LYCm2/9adpAxc+UYi5UsxTw4ewSBjmsXYq//Ahaw4mjw==
```

СОВЕТ

При следующих подключениях этого пользователя к серверу по адресу 10.140.67.23 он будет аутентифицироваться с помощью сохраненного ключа. Если сервер изменяет свой ключ (это происходит, если операционная система переустановлена или изменены ключи), попытки подключиться к системе приведут к отказу в соединении и выводу предупреждения о том, что вы можете подвергнуться атаке. Если ключ действительно изменился, то чтобы снова получить доступ по ssh к этому адресу, просто удалите ключ хоста (всю строку) из файла known_hosts и скопируйте новый ключ.

Удаленное выполнение команд с помощью команды ssh

Помимо входа в удаленную оболочку, команда `ssh` может использоваться для удаленного выполнения команды и возврата выходных данных в локальную систему, например:

```
$ ssh johndoe@10.140.67.23 hostname
johndoe@10.140.67.23's password: **********
host01.example.com
```

В примере команда `hostname` выполняется от имени пользователя `johndoe` в системе Linux, расположенной по IP-адресу 10.140.67.23. Выходные данные команды — это имя удаленного хоста (в данном случае `host01.example.com`), который появляется на локальном экране.

Если вы запускаете команду удаленного выполнения `ssh`, которая включает параметры или аргументы, обязательно заключите всю удаленную командную строку в кавычки. Имейте в виду, что, если вы ссылаетесь на файлы или каталоги в удаленных командах, относительные пути интерпретируются по отношению к домашнему каталогу пользователя, как показано далее:

```
$ ssh johndoe@10.140.67.23 "cat myfile"
johndoe@10.140.67.23's password: **********
Contents of the myfile file located in johndoe's home directory.
```

В примере команда `ssh` переходит на удаленный хост, расположенный по адресу `10.140.67.23`, и запускает команду `cat myfile` от имени пользователя `johndoe`. Это приводит к отображению содержимого файла `myfile` из этой системы на локальном экране.

Другой тип удаленного выполнения, который можно реализовать с помощью `ssh`, — это переадресация X11 (X Window System, версия 11). Если на сервере включена переадресация X11 (в файле `/etc/sshd/sshd_config` установлен параметр `X11Forwarding yes`), можете безопасно запускать графические приложения с сервера по SSH-соединению с помощью команды `ssh -X`. Для нового администратора сервера это означает, что, если на сервере установлены графические средства администрирования, их можно запускать и не сидя за локальным компьютером, как в этом примере:

```
$ ssh -X johndoe@10.140.67.23 system-config-printer
johndoe@10.140.67.23's password: **********
```

После выполнения этой команды вам будет предложено ввести пароль администратора. Затем появится окно Printers (Принтеры), которое позволяет настроить принтеры. По окончании просто закройте окно, и локальное приглашение вернется. Это можно сделать для любого графического инструмента администрирования или для обычных X-приложений, таких как графический редактор `gedit` (тогда не придется применять `vi`).

Если вы хотите выполнить несколько команд `X`, но не хотите каждый раз повторно подключаться, используйте переадресацию X11 непосредственно из удаленной оболочки. Поместив команды в фоновый режим, вы сможете иметь несколько удаленных приложений X, работающих на вашем локальном рабочем столе одновременно, например:

```
$ ssh -X johndoe@10.140.67.23
johndoe@10.140.67.23's password: **********
$ system-config-printer &
$ gedit &
$ exit
```

После завершения работы с графическими приложениями закройте их как обычно. Затем введите команду `exit`, как показано в приведенном ранее коде, чтобы покинуть удаленную оболочку и вернуться в локальную.

Копирование файлов между системами с помощью команд scp и rsync

Команда `scp` похожа на устаревшую UNIX-команду `rcp` для копирования файлов в системы Linux и из них, только в данном случае все коммуникации зашифрованы. Файлы могут быть скопированы из удаленной системы в локальную или из локальной в удаленную. Есть также возможность рекурсивно копировать файлы через всю структуру каталогов.

Далее приведен пример применения команды `scp` для копирования файла, вызванного из домашнего каталога пользователя `chris`, в каталог `/tmp` на удаленном компьютере под именем пользователя `johndoe`:

```
$ scp /home/chris/memo johndoe@10.140.67.23:/tmp
johndoe@10.140.67.23's password: ***************
memo       100%|****************|  153   0:00
```

Вы должны ввести пароль для пользователя `johndoe`. После этого файл успешно копируется в удаленную систему.

Можно также делать рекурсивные копии с помощью команды `scp`, применив параметр `-r`. Вместо файла передайте имя каталога команде `scp`, и все файлы и каталоги ниже этой точки в файловой системе будут скопированы в другую систему:

```
$ scp johndoe@10.140.67.23:/usr/share/man/man1/ /tmp/
johndoe@10.140.67.23's password: ***************
volname.1.gz                                  100%   543   0.5KB/s   00:00
mtools.1.gz                                   100%  6788   6.6KB/s   00:00
roqet.1.gz                                    100%  2496   2.4KB/s   00:00
```

Пока пользователь `johndoe` имеет доступ к файлам и каталогам в удаленной системе, а локальный пользователь может записывать данные в целевой каталог (в данном случае оба значения верны), структура каталогов из `/usr/share/man/man1down` копируется в локальный каталог `/tmp`.

Команда `scp` может применяться для резервного копирования файлов и каталогов по сети. Однако, если сравнить команду `scp` с командой `rsync`, то видно, что команда `rsync` (которая также работает через SSH-соединения) — это лучший инструмент резервного копирования. Попробуйте выполнить команду `scp`, чтобы скопировать каталог `man1` (можете имитировать команду, используя `localhost` вместо IP-адреса, если у вас есть только одна доступная система Linux). В системе, в которую вы скопировали файлы, введите следующее:

```
$ ls -l /usr/share/man/man1/batch* /tmp/man1/batch*
-rw-r--r--.1 johndoe johndoe 2628 Apr 15 15:32 /tmp/man1/batch.1.gz
lrwxrwxrwx.1 root root 7 Feb 14 17:49 /usr/share/man/man1/batch.1.gz
    -> at.1.gz
```

Затем снова запустите команду `scp` и еще раз перечислите файлы:

```
$ scp johndoe@10.140.67.23:/usr/share/man/man1/ /tmp/
johndoe@10.140.67.23's password: ***************
$ ls -l /usr/share/man/man1/batch* /tmp/man1/batch*
```

```
-rw-r--r--.1 johndoe johndoe 2628 Apr 15 15:40 /tmp/man1/batch.1.gz
lrwxrwxrwx.1 root root 7 Feb 14 17:49 /usr/share/man/man1/batch.1.gz
    -> at.1.gz
```

Вывод команд в примере объясняет, как работает команда `scp`.

- **Потеря атрибутов.** Права или атрибуты даты и времени не сохранились при копировании файлов. Если вы используете команду `scp` в качестве инструмента резервного копирования, то, вероятно, захотите сохранить права и временные метки на файлах, чтобы восстановить файлы позже.
- **Потеря символических ссылок.** Файл `batch.1.gz` на самом деле является символической ссылкой на файл `at.1.gz`. Вместо того чтобы копировать ссылку, команда `scp` переходит по ссылке и на самом деле копирует файл. Опять же, если нужно восстановить этот каталог, файл `batch.1.gz` будет заменен реальным файлом `at.1.gz`, а не ссылкой на него.
- **Копирование повторяется без необходимости.** Второй вывод команды `scp` говорит о том, что все файлы были скопированы снова, несмотря на то что нужные файлы уже скопированы в целевое место назначения. Обновленная дата изменений подтверждает это. Команда `rsync`, напротив, может определить, что файл уже скопирован, и не будет копировать его снова.

Команда `rsync` — лучший инструмент резервного копирования по сети, поскольку не имеет перечисленных недостатков команды `scp`. Используйте команду `rsync`, чтобы выполнить то же действие, какое выполнялось командой `scp`, но с несколькими дополнительными параметрами:

```
$ rm -rf /tmp/man1/
$ rsync -avl johndoe@10.140.67.23:/usr/share/man/man1/ /tmp/
johndoe@10.140.67.23's password: ***************
sending incremental file list
man1/
man1/HEAD.1.gz
man1/Mail.1.gz -> mailx.1.gz
...
$ rsync -avl johndoe@10.140.67.23:/usr/share/man/man1/ /tmp/
johndoe@10.140.67.23's password: ***************
sending incremental file list
sent 42362 bytes received 13 bytes 9416.67 bytes/sec
total size is 7322223 speedup is 172.80
$ ls -l /usr/share/man/man1/batch* /tmp/man1/batch*
lrwxrwxrwx.1 johndoe johndoe 7 Feb 14 17:49 /tmp/man1/batch.1.gz
    -> at.1.gz
lrwxrwxrwx.1 root root 7 Feb 14 17:49 /usr/share/man/man1/batch.1.gz
    -> at.1.gz
```

После удаления каталога `/tmp/man1` запустите команду `rsync`, чтобы скопировать все файлы в каталог `/tmp/man1` с помощью параметров `-a` (рекурсивно архивировать), `-v` (подробно) и `-l` (копировать символические ссылки). Затем сразу же выполните команду еще раз и обратите внимание — ничего не скопируется.

Команда `rsync` знает, что все файлы уже скопированы, поэтому не будет делать это снова. Это позволяет экономить пропускную способность сети для каталогов с гигабайтами файлов, где изменяются лишь несколько мегабайт.

Обратите также внимание на выходные данные команды `ls -l`, где символические ссылки были сохранены в файле `batch.1.gz` и поэтому на файле появилась метка даты/времени. Если понадобится восстановить эти файлы позже, их можно вернуть к первоначальному состоянию.

В таком варианте команда `rsync` хорошо подходит для резервного копирования. Но что, если вы хотите зеркально отразить два каталога, сделав содержимое двух структур каталогов абсолютно одинаковым на двух компьютерах? Команды, приведенные далее, иллюстрируют, как создать точное зеркало структуры каталогов на обоих компьютерах, используя каталоги, показанные с предыдущими командами `rsync`.

Сначала в удаленной системе скопируйте новый файл в копируемый каталог:

```
# cp /etc/services /usr/share/man/man1
```

Затем в локальной системе запустите `rsync`, чтобы скопировать все новые файлы (в данном случае только каталог и новый файл `services`):

```
$ rsync -avl johndoe@10.140.67.23:/usr/share/man/man1 /tmp
johndoe@10.140.67.23's password:
***************
sending incremental file list
man1/
man1/services
```

После этого вернитесь в удаленную систему и удалите новый файл:

```
$ sudo rm /usr/share/man/man1/services
```

Теперь в локальной системе снова запустите команду `rsync` и обратите внимание на то, что в этот раз ничего не происходит. На этом этапе удаленный и локальный каталоги различаются, потому что локальная система имеет файл служб, а удаленная — нет. Это правильное поведение каталога резервных копий. (Резервные копии хранятся на случай, если что-то было удалено по ошибке.) Но если вы хотите, чтобы удаленные и локальные каталоги были зеркально отражены, вам придется добавить параметр `--delete`. В результате файл служб в локальной системе удаляется, что приводит к синхронизации удаленных и локальных структур каталогов:

```
$ rsync -avl /usr/share/man/man1 localhost:/tmp
johndoe@10.140.67.23's password: ***************
sending incremental file list
man1/
$ rsync -avl --delete johndoe@10.140.67.23:/usr/share/man/man1 /tmp
johndoe@10.140.67.23's password: ***************
sending incremental file list
deleting man1/services
```

Интерактивное копирование с помощью команды sftp

Если вы не знаете точно, что хотите скопировать в удаленную систему или из нее, примените команду `sftp` для создания интерактивного сеанса в стиле FTP через службу SSH. С помощью команды `sftp` можно подключиться к удаленной системе через службу SSH, изменить каталоги, перечислить содержимое каталога, а затем (при наличии соответствующих прав) получить нужные файлы и поместить их на сервер. Имейте в виду, что, несмотря на свое название, команда `sftp` не имеет ничего общего с протоколом FTP и не использует FTP-серверы. Она просто применяет стиль взаимодействия FTP между клиентом и сервером `sshd`.

В следующем примере показано, как пользователь `johndoe` подключается к `jd.example.com`:

```
$ sftp johndoe@jd.example.com
Connecting to jd.example.com
johndoe@jd.example.com's password: ***************
sftp>
```

На этом этапе можно начать интерактивный сеанс FTP. Вы можете использовать команды `get` и `put` для файлов, как и для любого FTP-клиента, зная при этом, что соединение безопасно и зашифровано. Поскольку протокол FTP передает имена пользователей, пароли и данные в виде открытого текста, применение команды `sftp` через службу SSH (по возможности) — лучшая альтернатива для того, чтобы позволить пользователям копировать файлы в интерактивном режиме из системы.

Аутентификация на основе ключей (без пароля)

Если вы применяете инструменты SSH для подключения к одним и тем же системам в течение всего дня, то введение своего пароля снова и снова может быстро надоесть. Вместо аутентификации на основе пароля SSH позволяет настроить аутентификацию на основе ключа. Вот как это работает.

- Вы создаете открытый и закрытый ключи.
- Вы защищаете закрытый ключ, но копируете открытый ключ в учетную запись пользователя на удаленном узле, для которого хотите выполнить аутентификацию.
- Когда ключ скопирован в нужное место, вы с помощью любых инструментов SSH подключаетесь к учетной записи пользователя на удаленном узле. Вместо того чтобы запрашивать пароль, удаленная служба SSH сравнивает открытый и закрытый ключи и разрешает доступ, если они совпадают.

При создании ключа можно добавить к закрытому ключу кодовую фразу. Если вы решили сделать это, то, даже если для аутентификации в удаленной системе не нужен пароль, необходимо вводить кодовую фразу, чтобы разблокировать закрытый ключ. Если не добавлять фразу, то подключаться можно с помощью пар

открытых и закрытых ключей таким образом, чтобы полностью исключить использование пароля. Но если кому-то нужно получить доступ к вашему секретному ключу, он может действовать от вашего имени при любом соединении, которое требует этого ключа.

Далее приведен пример того, как локальный пользователь с именем `chris` может настроить аутентификацию на основе ключа для удаленного пользователя с именем `johndoe` по IP-адресу 10.140.67.23. Если у вас нет двух систем Linux, можете имитировать действие, применив две учетные записи пользователей в локальной системе. Сначала я вошел в систему как локальный пользователь по имени `chris` и набрал следующую команду, чтобы сгенерировать локальную пару открытых и закрытых ключей:

```
$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/chris/.ssh/id_rsa): ENTER
Enter passphrase (empty for no passphrase): ENTER
Enter same passphrase again: ENTER
Your identification has been saved in /home/chris/.ssh/id_rsa.
Your public key has been saved in /home/chris/.ssh/id_rsa.pub.
The key fingerprint is:
bf:06:f8:12:7f:f4:c3:0a:3a:01:7f:df:25:71:ec:1d chris@abc.example.com
The key's randomart image is:
...
```

Я принял ключ RSA, предлагаемый по умолчанию (ключи DSA также разрешены), и дважды нажал клавишу Enter, чтобы добавить пустую фразу-пароль, связанную с ключом. В результате закрытый ключ (`id_rsa`) и открытый ключ (`id_rsa.pub`) копируются в каталог `.ssh` на моем локальном домашнем каталоге. Далее я скопировал этот ключ удаленному пользователю, чтобы можно было применять аутентификацию на основе ключа каждый раз, когда я подключаюсь к этой учетной записи с помощью инструментов `ssh`:

```
$ ssh-copy-id -i ~/.ssh/id_rsa.pub johndoe@10.140.67.23
johndoe@10.140.67.23's password:
***************
```

При появлении приглашения командной строки я ввел пароль пользователя `johndoe`. После этого открытый ключ, принадлежащий пользователю `chris`, копируется в файл `authorized_keys` в `ssh`-каталоге `johndoe` в удаленной системе. Когда `chris` в следующий раз попытается подключиться к учетной записи `johndoe`, SSH-соединение будет аутентифицировано с помощью этих ключей. Поскольку к закрытому ключу не добавлялась кодовая фраза, снимать блокировку с него не нужно.

Войдите в систему с помощью команды `ssh johndoe@10.140.67.23` и проверьте файл `$HOME/.ssh/authorized_keys`, чтобы убедиться, не добавлены ли дополнительные ненужные ключи:

```
[chris]$ ssh johndoe@10.140.67.23
Last login: Sun Apr 17 10:12:22 2016 from 10.140.67.22
[johndoe]$
```

Теперь, когда ключи сохранены, пользователь chris может применить ssh, scp, rsync или любую другую команду с поддержкой SSH для аутентификации по ключу. С помощью этих ключей, например, команда rsync может войти в скрипт cron и автоматически создавать резервную копию домашнего каталога johndoe каждую ночь.

Хотите еще больше обезопасить свою удаленную систему? После создания ключей для всех, кому можно входить в нее, можете настроить службу sshd в удаленной системе так, чтобы она не разрешала аутентификацию с помощью пароля, изменив параметр PasswordAuthentication в файле /etc/ssh/sshd_config на значение no, как здесь:

```
PasswordAuthentication no
```

Затем перезапустите службу sshd (systemctl restart sshd). После этого все, у кого есть действительный ключ, по-прежнему могут войти в систему. А тот, кто попытается войти в систему без ключа, получит следующее сообщение об ошибке (причем у него даже не будет возможности ввести имя пользователя и пароль):

```
Permission denied (publickey,gssapi-keyex,gssapi-with-mic).
```

Настройка журнала системы

Зная, как получить доступ к удаленному серверу с помощью инструментов SSH, вы можете войти на сервер и настроить некоторые службы, необходимые для обеспечения его бесперебойной работы. Ведение журнала системы — это одна из основных служб, настроенных для отслеживания того, что происходит в системе.

Служба rsyslog (демон rsyslogd) предоставляет функции для сбора сообщений журнала из программного обеспечения в системе Linux и их направления в локальные файлы журнала, на устройства или удаленные узлы ведения журнала. Конфигурация службы rsyslog аналогична конфигурации ее предшественницы, службы syslog. Однако служба rsyslog позволяет добавлять модули для управления сообщениями журнала и их направления.

В последних версиях систем Red Hat Enterprise Linux и Fedora функция rsyslog использует сообщения, которые собираются и хранятся в журнале systemd. Чтобы отобразить сообщения журнала непосредственно из журнала systemd, а не просматривать их из файлов в каталоге /var/log, задействуйте команду journalctl.

Подключение журнала системы с помощью службы rsyslog

Большинство файлов в каталоге /var/log заполняются сообщениями журнала, направленными из службы rsyslog. Демон rsyslogd — это демон ведения журнала системы. Он принимает сообщения журнала от множества других программ и записывает их в соответствующие файлы журнала. Такой вариант действий

предпочтительнее, чем когда каждая программа записывает данные непосредственно в собственный файл журнала, так как позволяет централизованно управлять обработкой файлов журнала.

Возможна настройка службы `rsyslogd` для фиксирования различных уровней детализации в файлах журнала. Например, можно настроить службу так, чтобы она игнорировала все, кроме самых важных сообщений, или, наоборот, записывала каждую деталь.

Демон `rsyslogd` может даже принимать сообщения от других компьютеров в сети. Функция удаленного ведения журнала достаточно удобна, так как позволяет централизовать управление файлами журналов и их просмотр из многих систем вашей сети. Кроме того, это дает важное преимущество в плане безопасности.

При удаленном ведении журнала, если система в вашей сети взломана, злоумышленник не сможет удалить или изменить файлы журнала, потому что они хранятся на отдельном компьютере. Однако важно помнить, что по умолчанию сообщения журнала не шифруются (хотя шифрование может быть включено). Любой, кто подключается к вашей локальной сети, может просматривать эти сообщения во время передачи их с одного компьютера на другой. Кроме того, хотя взломщики не могут изменять старые записи журнала, они способны повлиять на систему таким образом, что любое новое сообщение журнала не будет правдивым.

Запуск дополнительного хоста для журнала, компьютера, который будет служить только для записи сообщений журнала с других компьютеров в сети, — не редкость. Такая система не запускает никаких других служб, поэтому маловероятно, что она будет взломана. А это делает практически невозможным то, что взломщики смогут полностью стереть свои следы.

Работа файла rsyslog.conf

Файл `/etc/rsyslog.conf` является основным файлом конфигурации для службы `rsyslog`. В файле `/etc/rsyslog.conf` содержится раздел модулей, который позволяет добавлять или не удалять определенные функции в службе `rsyslog`. Далее приведен пример раздела модулей файла `/etc/rsyslog.conf` в системе RHEL 8:

```
module(load="imuxsock"
    # provides support for local system logging (e.g. via logger command)
    SysSock.Use="off") # Turn off message reception via local log socket;
                       # local messages are retrieved through imjournal now.
module(load="imjournal"
    # provides access to the systemd journal
        StateFile="imjournal.state") # File to store the position in the
journal
#module(load="imklog")
    # reads kernel messages (the same are read from journald)
#module(load="immark")
    # provides --MARK-- message capability
```

```
# Provides UDP syslog reception
# for parameters see http://www.rsyslog.com/doc/imudp.html
#module(load="imudp") # needs to be done just once
#input(type="imudp" port="514")

# Provides TCP syslog reception
# for parameters see http://www.rsyslog.com/doc/imtcp.html
#module(load="imtcp") # needs to be done just once
#input(type="imtcp" port="514")
```

Строки, начинающиеся с `module(load=`, загружают соответствующие модули. Перед модулями, которые в данный момент отключены, стоит знак фунта (`#`).

Модуль `imjournal` позволяет службе `rsyslog` получить доступ к журналу `systemd`. Модуль `imuxsock` необходим для приема сообщений из локальной системы. (Он не должен быть закомментирован, то есть перед ним должен стоять знак фунта (`#`).) Модуль `imklog` собирает сообщения ядра.

Модули, не включенные по умолчанию, включают модуль `immark`, который позволяет регистрировать сообщения с помощью параметра `--MARK--` (используется для указания того, что служба активна). Модули `imudp` и `imtcp` и связанные с ними записи номеров портов применяются для того, чтобы служба `rsyslog` могла принимать сообщения удаленного журнала, и более подробно рассматриваются в пункте «Настройка и использование узла ведения журнала с помощью службы rsyslogd» далее в этом разделе.

Бóльшая часть работы, выполняемой в файле конфигурации `/etc/rsyslog.conf`, заключается в изменении раздела `RULES`. Далее приведен пример правил из раздела `RULES` файла `/etc/rsyslog.conf` (обратите внимание на то, что в дистрибутиве Ubuntu эта информация хранится в каталоге `/etc/rsyslog.d`):

```
#### RULES ####
# Log all kernel messages to the console.

# Logging much else clutters up the screen.

#kern.*                                          /dev/console
# Log anything (except mail) of level info or higher.
# Don't log private authentication messages!
*.info;mail.none;authpriv.none;cron.none        /var/log/messages
# The authpriv file has restricted access.
authpriv.*                                       /var/log/secure
# Log all the mail messages in one place.
mail.*                                           -/var/log/maillog
# Log cron stuff
cron.* /var/log/cron
```

Записи с правилами делятся на две колонки. В левой приведена информация о том, какие сообщения совпадают, в правой показано, куда (имя каталога) переходят совпадающие сообщения. Сообщения сопоставляются на основе функции (`mail`, `cron`, `kern` и т. д.) и приоритета (начиная с `debug`, `info`, `notice` вплоть до `crit`, `alert`

и emerg) и разделяются точкой (.). Так, параметр mail.info соответствует всем сообщениям из почтовой службы, которые относятся к уровню информации и выше.

Что касается того, куда отправляются сообщения, то большинство из них направляется в файлы в каталоге /var/log. Можно, однако, направить сообщения с другого устройства (например, устройства /dev/) либо с удаленного хоста (например, @loghost.example.com). Знак @ указывает, что это имя удаленного хоста.

По умолчанию журналы ведутся только для локальных файлов в каталоге /var/log. Однако, если раскомментировать строку kern.*, вы сможете легко направлять сообщения ядра всех уровней на экран консоли вашего компьютера.

Первая рабочая запись в предыдущем примере показывает, что сообщения уровня информации от всех служб (*) соответствуют этому правилу, за исключением сообщений от служб mail, authpriv и cron (которые исключаются со словом none). Все совпадающие сообщения направляются в файл /var/log/messages.

Службы mail, authpriv (сообщения аутентификации) и cron (сообщения функции cron) имеют собственные файлы журналов, перечисленные в столбцах справа от них. Чтобы вы могли понять, как работают эти и другие файлы журнала, далее описывается файл /var/log/messages.

Работа файла журнала сообщений

Из-за того, что множество программ и служб записывают информацию в файл журнала сообщений, важно понимать формат работы этого файла. Изучив его, вы заранее можете получить предупреждение о проблемах, возникших в системе. Каждая строка файла — это одно сообщение, записанное какой-либо программой или службой. Вот фрагмент фактического файла журнала сообщений messages:

```
Feb 25 11:04:32 toys network: Bringing up loopback: succeeded
Feb 25 11:04:35 toys network: Bringing up interface eth0: succeeded
Feb 25 13:01:14 toys vsftpd(pam_unix)[10565]: authentication failure;
        logname= uid=0 euid=0 tty= ruser= rhost=10.0.0.5 user=chris
Feb 25 14:44:24 toys su(pam_unix)[11439]: session opened for
        user root by chris(uid=500)
```

Формат сообщений по умолчанию в файле /var/log/messages разделен на пять основных частей. Он определяется следующей записью в файле /etc/rsyslog.conf:

```
module(load="builtin:omfile" Template="RSYSLOG_TraditionalFileFormat")
```

При просмотре сообщений в файлах из каталога /var/log слева направо части сообщений выглядят следующим образом:

- дата и время регистрации сообщения;
- имя компьютера, с которого пришло сообщение;
- имя программы или службы, к которой относится сообщение;
- номер процесса (в квадратных скобках) программы, отправляющей сообщение;
- сам текст сообщения.

Еще раз взгляните на приведенный фрагмент файла. Из первых двух строк видно, что сеть была перезапущена. Следующая строка показывает, что пользователь `chris` пытался и не смог попасть на FTP-сервер в этой системе с компьютера по адресу `10.0.0.5`. (Он ввел неверный пароль, и аутентификация не удалась.) В последней строке показано, что `chris` применяет команду `su`, чтобы стать суперпользователем.

Периодически просматривая файлы `messages` и `secure`, вы можете отследить попытку взлома до того, как она будет реализована полностью. Если происходит чрезмерное количество попыток подключения к определенной службе, особенно если они исходят из систем, находящихся в Интернете, вы можете подвергнуться атаке.

Настройка и использование узла ведения журнала с помощью службы rsyslogd

Чтобы перенаправить файлы журнала вашего компьютера в службу `rsyslogd` другого компьютера, необходимо внести изменения как в локальный, так и в удаленный конфигурационный файл службы `rsyslog` — `/etc/rsyslog.conf`. Станьте суперпользователем с помощью команды `su`, а затем откройте файл `/etc/rsyslog.conf` в текстовом редакторе, например, `vi`.

На стороне клиента. Чтобы отправить сообщения на другой компьютер (удаленный хост) вместо файла, сначала замените имя файла журнала символом `@`, за которым следует имя удаленного хоста. Например, чтобы направить выходные данные сообщений, поступающие в файлы журналов `messages`, `secure` и `maillog`, на удаленный хост, добавьте строки, выделенные полужирным шрифтом, в файл сообщений:

```
# Log anything (except mail) of level info or higher.
# Don't log private authentication messages!
*.info;mail.none;news.none;authpriv.none;cron.none    /var/log/messages
*.info;mail.none;news.none;authpriv.none;cron.none @loghost
# The authpriv file has restricted access.
authpriv.*                                  /var/log/secure
authpriv.*                                  @loghost
# Log all the mail messages in one place.
mail.*                                      -/var/log/maillog
mail.*                                      @loghost
```

Теперь сообщения отправляются в службу `rsyslogd`, которая работает на компьютере с именем `loghost` (удаленный хост). Имя `loghost` не выбирается произвольно. Создание такого имени хоста и превращение его в псевдоним для реальной системы, действующей как удаленный хост, — обычное дело. Таким образом, если вам когда-нибудь понадобится переключить обязанности удаленного хоста на другой компьютер, нужно будет изменить только псевдоним `loghost`, не редактируя повторно файл `syslog.conf` на каждом компьютере.

На стороне удаленного хоста. Удаленный хост, настроенный на прием сообщений, должен прослушивать их на стандартных портах (514 UDP, хотя

его можно настроить на прием сообщений и через порт 514 TCP). Порядок действий при настройке удаленного хоста Linux, который также запускает службу `rsyslog`, таков.

- Отредактируйте файл `/etc/rsyslog.conf` в системе удаленного хоста и раскомментируйте строки, которые позволяют демону `rsyslogd` прослушивать удаленные сообщения журнала. Раскомментируйте первые две строки, чтобы включить входящие сообщения журнала UDP на порте 514 (по умолчанию), затем раскомментируйте две строки, чтобы разрешить сообщения, использующие протокол TCP (также порт 514):

```
module(load="imudp") # needs to be done just once
input(type="imudp" port="514")
module(load="imtcp") # needs to be done just once
input(type="imtcp" port="514")
```

- Откройте брандмауэр, чтобы разрешить отправку новых сообщений на ваш удаленный хост. (См. главу 25 «Защита Linux в сети», чтобы узнать, как открыть определенные порты и разрешить доступ к вашей системе.)
- Перезапустите службу `rsyslog` (команда `service rsyslog restart` или `systemctl restart rsyslog.service`).
- Если служба запущена, вы должны увидеть, что она прослушивает подключенные порты (UDP и/или TCP-порты 514). Выполните команду `netstat`, как показано далее, чтобы убедиться, что демон `rsyslogd` прослушивает порты IPv4 и IPv6 514 для служб UDP и TCP:

```
 # netstat -tupln | grep 514
tcp    0    0 0.0.0.0:514    0.0.0.0:*    LISTEN    25341/rsyslogd
tcp    0    0 :::514         :::*         LISTEN    25341/rsyslogd
udp    0    0 0.0.0.0:514    0.0.0.0:*              25341/rsyslogd
udp    0    0 :::514         :::*                   25341/rsyslogd
```

Просмотр журналов с помощью службы logwatch

Служба `logwatch` работает в большинстве систем Linux, которые ведут системный журнал с помощью службы `rsyslog`. Поскольку журналы в занятых системах со временем могут сильно увеличиться в размере, системному администратору может потребоваться много времени, чтобы просмотреть каждое сообщение во всех журналах. Чтобы запустить службу `logwatch`, введите следующую команду:

```
# yum install logwatch
```

Служба `logwatch` собирает сообщения, которые могут представлять проблему, один раз за ночь, помещает их в сообщение и отправляет его на указанный адрес электронной почты. Чтобы включить службу `logwatch`, нужно лишь установить пакет `logwatch`.

Служба `logwatch` запускается из команды `cron` (`0logwatch`), помещенной в файл `/etc/cron.daily`. Файл `/etc/logwatch/conf/logwatch.conf` содержит локальные на-

стройки. Параметры по умолчанию, используемые для сбора сообщений журнала, устанавливаются в файл `/usr/share/logwatch/default.conf/logwatch.conf`.

Некоторые настройки по умолчанию определяют расположение файлов журнала (`/var/log`), временного каталога (`/var/cache/logwatch`) и получателя ежедневного отчета на электронную почту `logwatch` (локальный суперпользователь). Если вы не собираетесь входить на сервер для чтения сообщений `logwatch`, то измените параметр `MailTo` в файле `/etc/logwatch/conf/ logwatch.conf`:

```
MailTo = chris@example.com
```

Другие параметры, которые можно изменить (например, уровень детализации или временной диапазон для каждого отчета), находятся в файле `/usr/share/logwatch/default.conf/logwatch.conf`. Как уже говорилось, вносите новые дополнения в файл `/etc/logwatch/conf/logwatch.conf`.

Когда служба включена (что происходит только при установке пакета `logwatch`), вы каждую ночь будете получать сообщение на почту суперпользователя. Войдя в систему как суперпользователь, можете применить старую команду `mail` для просмотра почтового ящика суперпользователя:

```
# mail
Heirloom Mail version 12.5 7/5/10. Type ? for help.
"/var/spool/mail/root": 2 messages 2 new
>N  1 logwatch@abc.ex  Sun Feb 15 04:02 45/664    "Logwatch for abc"
    2 logwatch@abc.ex  Mon Feb 16 04:02 45/664    "Logwatch for abc"
& 1
& x
```

Вы будете получать сообщения от `logwatch` на указанный адрес электронной почты ежедневно (в примере — в 04:02). Введите номер сообщения, которое хотите просмотреть, и пролистайте его с помощью клавиши Пробел или перейдите к нему построчно, нажимая клавишу Enter. Введите `x`, чтобы выйти.

Информация, которую вы видите, включает в себя ошибки ядра, установленные пакеты, ошибки аутентификации и неисправные службы. Пользователю также сообщается информация о задействовании дискового пространства, из чего видно, как заполняется хранилище. Просто взглянув на сообщение службы `logwatch`, вы получите представление о том, атакует ли кто-то вашу систему или происходят ли какие-то повторяющиеся сбои.

Проверка системных ресурсов с помощью функции sar

System Activity Reporter (`sar`) — одно из старейших средств мониторинга системы, созданных для ранних систем UNIX за несколько десятилетий до появления систем Linux. Сама команда может отображать на экране активность системы непрерывно или с заданными интервалами (каждую секунду или две). Она может отображать также собранные ранее данные о деятельности системы.

Команда `sar` является частью пакета `sysstat`. При установке пакета `sysstat` и подключении службы `sysstat` система немедленно начинает собирать данные о системной активности, которые можно просмотреть позже, используя определенные параметры команды `sar`:

```
# systemclt enable sysstat
# systemctl start sysstat
```

Для чтения данных в файлах `/var/log/sa/sa??` можно применять некоторые из следующих команд `sar`:

```
# sar -u  |  less
Linux 5.3.8-200.fc30.x86_64 (fedora30host) 11/28/2019     _x86_64_ (1 CPU)

23:27:46      LINUX RESTART (1 CPU)

11:30:05 PM  CPU    %user    %nice  %system    %iowait    %steal    %idle
11:40:06 PM  all     0.90     0.00     1.81       1.44      0.28    95.57
Average:     all     0.90     0.00     1.81       1.44      0.28    95.57
```

Параметр `-u` отображает загрузку процессора. По умолчанию вывод начинается в полночь текущего дня, а затем показывает, сколько времени обработки потребляется различными частями системы. Выходные данные продолжают показывать активность каждые 10 минут, пока не будет достигнуто текущее время.

Чтобы просмотреть выходные данные активности диска, выполните команду `sar -d`. Вывод также выполняется с 10-минутными интервалами, начиная с полуночи:

```
# sar -d | less
Linux 5.3.8-200.fc30.x86_64 (fedora30host) 11/28/2019 _x86_64_ (1 CPU)

23:27:46     LINUX RESTART  (1 CPU)

11:30:05 PM       DEV    tps    rkB/s  wkB/s  areq-sz  aqu-sz  await...
11:40:06 PM    dev8-0  49.31  5663.94  50.38   115.89    0.03   1.00
11:40:06 PM  dev253-0  48.99  5664.09   7.38   115.78    0.05   0.98
11:40:06 PM  dev253-1  10.84     0.01  43.34     4.00    0.04   3.29
Average:       dev8-0  49.31  5663.94  50.38   115.89    0.03   1.00
Average:     dev253-0  48.99  5664.09   7.38   115.78    0.05   0.98
Average:     dev253-1  10.84     0.01  43.34     4.00    0.04   3.29
```

Если вы хотите запускать отчеты о действиях в реальном времени, добавьте счетчики и временные интервалы в командную строку, как показано далее:

```
# sar -n DEV 5 2
Linux 5.3.8-200.fc30.x86_64 (fedora30host) 11/28/2019 _x86_64_ (1 CPU)
11:19:36 PM  IFACE  rxpck/s  txpck/s  rxkB/s  txkB/s  rxcmp/s  txcmp/s...
11:19:41 PM     lo     5.42     5.42    1.06    1.06   0.00       0.00...
11:19:41 PM   ens3     0.00     0.00    0.00    0.00   0.00       0.00...
...
```

```
Average:  IFACE  rxpck/s  txpck/s  rxkB/s  txkB/  rxcmp/s  txcmp/s  rxmcst/s
Average:     lo     7.21     7.21    1.42   1.42     0.00     0.00      0.00
Average:   ens3     0.00     0.00    0.00   0.00     0.00     0.00      0.00
Average:  wlan0     4.70     4.00    4.81   0.63     0.00     0.00      0.00

Average:   pan0     0.00     0.00    0.00   0.00     0.00     0.00      0.00
Average:   tun0     3.70     2.90    4.42   0.19     0.00     0.00      0.00
```

С помощью параметра `-n Dev` в примере показана активность различных сетевых интерфейсов системы. Теперь можно увидеть, сколько пакетов и сколько килобайт данных было передано и получено. В этом примере выборка данных производилась каждые 5 секунд и повторялась дважды.

Дополнительные сведения о том, как можно собирать и отображать данные `sar`, см. на справочных страницах `sar`, `sadc`, `sa1` и `sa2`.

Проверка пространства в системе

Хотя служба `logwatch` ежедневно отображает информацию об использовании пространства на системных дисках, команды `df` и `du` позволяют сразу увидеть, какой объем диска доступен. В следующих разделах эти команды будут описаны.

Отображение пространства в системе с помощью команды df

Вы можете отобразить объем, доступный в файловых системах, с помощью команды `df`. Чтобы увидеть свободное место во всех смонтированных файловых системах на компьютере Linux, введите команду `df` без каких-либо параметров:

```
$ df
Filesystem  1k-blocks     Used  Available  Use%  Mounted on
/dev/sda3    30645460  2958356   26130408   11%  /
/dev/sda2       46668     8340      35919   19%  /boot
...
```

В примере показано пространство, доступное в разделах жесткого диска, смонтированных в каталогах `/(root)` (`/dev/sda1`) и `/boot` (`/dev/sda2`). Дисковое пространство отображается в блоках размером 1 Кбайт. Чтобы отобразить выходные данные в более удобочитаемой форме, используйте параметр `-h`:

```
$ df -h
Filesystem         Size   Used   Avail   Use%   Mounted on
/dev/sda3           29G   2.9G     24G    11%   /
/dev/sda2           46M   8.2M     25M    19%   /boot
...
```

При вводе команды `df -h` выходные данные отображаются в виде более удобного списка с данными в мегабайтах или гигабайтах. Другие параметры команды `df` позволяют сделать следующее.

- Вывести только файловые системы определенного типа (`-t type`).
- Исключить файловые системы определенного типа (`-x type`). Например, введите `df-x tmpfs-x devtmpfs`, чтобы исключить временные типы файловых систем, ограничивая вывод файловыми системами, представляющими реальные области хранения.
- Включить в вывод файловые системы, в которых нет места, такие как `/proc` и `/dev/pts` (`-a`).
- Перечислить только доступные и используемые индексные узлы (`-i`).
- Отобразить пространство диска в определенных размерах блоков (`--block-size=#`).

Проверка использования диска с помощью команды du

Чтобы узнать, сколько места занимает конкретный каталог (и его подкаталоги), возьмите команду `du`. Без каких-либо параметров команда `du` перечисляет все каталоги ниже текущего каталога, а также объем, занимаемый каждым из них. И в конце команда `du` выводит общее дисковое пространство, используемое в этой структуре каталогов.

Команда `du` — хороший способ проверить, сколько места используется конкретным пользователем (`du /home/jake`) или определенным разделом файловой системы (`du /var`).

По умолчанию пространство диска отображается в блоках размером 1 Кбайт. Чтобы сделать вывод более удобочитаемым (в килобайтах, мегабайтах и гигабайтах), применяйте параметр `-h` следующим образом:

```
$ du  -h  /home/jake
114k      /home/jake/httpd/stuff
234k      /home/jake/httpd
137k      /home/jake/uucp/data
701k      /home/jake/uucp
1.0M      /home/jake
```

Выходные данные отображают пространство диска, занятое каждым каталогом домашнего каталога пользователя с именем `jake` (`/home/jake`). Потребляемое пространство диска отображается в килобайтах (`k`) и мегабайтах (`M`). Общее пространство, занимаемое каталогом `/home/jake`, отображается в последней строке. Добавьте параметр `-s`, чтобы увидеть общее пространство диска, занятое каталогом и его подкаталогами.

Отображение использования диска с помощью команды find

Команда `find` — это отличный способ поиска файлов на жестком диске с помощью различных критериев. Команда позволяет увидеть, где можно восстановить дисковое пространство, найдя файлы, которые превышают определенный размер или созданы конкретным человеком.

ПРИМЕЧАНИЕ

Вы должны быть суперпользователем, чтобы выполнить эту команду максимально эффективно, если только не проверяете личные файлы. Если вы им не являетесь, то не сможете проверять множество мест в файловой системе. Обычные пользователи чаще всего могут проверять только свои домашние каталоги.

В следующем примере команда `find` выполняет поиск в корневой файловой системе (`/`) любых файлов, принадлежащих пользователю с именем `jake` (`-user jake`), и выводит их имена. Выходные данные команды `find` организованы в виде длинного списка, отсортированного по размеру (`ls -ldS`). Эти данные отправляются в файл `/tmp/jake`. При просмотре этого файла (например, `less/tmp/jake`) вы найдете все файлы, принадлежащие пользователю `jake`, перечисленные по размеру. Так выглядит командная строка:

```
# find / -xdev -user jake -print | xargs ls -ldS > /tmp/jake
```

СОВЕТ

Параметр -xdev запрещает поиск в файловых системах, отличных от выбранной. Это позволяет не выводить ненужную информацию из файловой системы /proc. Это может также застраховать большие удаленно смонтированные файловые системы от поиска в них.

Вот еще один пример, только в этом случае вместо поиска файлов пользователя мы ищем файлы размером более 100 Кбайт (`-size +100M`):

```
# find / -xdev -size +100M | xargs ls -ldS > /tmp/size
```

Вы можете сэкономить много места на диске, просто удалив часть самых больших ненужных файлов. В примере видно, что большие файлы сортируются по размеру в файле `/tmp/size`.

Управление серверами на предприятии

Бóльшая часть процесса настройки сервера, описанного в этой книге, относится к установке системы вручную и работе непосредственно на удаленных хостах (компьютерах). Необходимость настраивать каждый хост индивидуально была бы

слишком неэффективной для современных центров обработки данных, состоящих из десятков, сотен или даже тысяч компьютеров. Чтобы сделать настройку серверов Linux в крупном дата-центре более эффективной, используют следующие методы.

- **Автоматическое развертывание.** Один из способов установки систем, не требующий работы вручную, — это загрузка PXE. Настроив PXE-сервер и загрузив компьютер в этой сети с помощью карты сетевого интерфейса с поддержкой PXE, можно начать полную установку этой системы, просто загрузив ее. После завершения установки система может перезагрузиться для запуска из установленной системы.
- **Универсальные хост-системы.** Сделав хост-системы максимально универсальными, можно значительно упростить индивидуальные установку, настройку и обновление. Процесс автоматизируется на уровнях, где базовая система устанавливается с помощью загрузки PXE, конфигурация выполняется с помощью таких функций, как `cloud-int` (средство настройки виртуальной машины), и приложения используют собственные зависимости при запуске. На уровне приложений это можно сделать, запустив приложение из виртуальной машины или контейнера. После завершения работы приложения его можно отключить, не оставляя зависимое программное обеспечение на хосте.
- **Разделение систем управления и рабочих систем.** Вместо индивидуального управления хост-системами отдельная платформа может управлять большими наборами систем. Для этого платформа, например OpenStack или OpenShift, использует узлы управления (их также называют компонентами *control plane* или *главными узлами*), управляющие машинами, на которых фактически выполняется рабочая нагрузка (иногда их называют *рабочими узлами* или просто *узлами*). Такое разделение задач по типу хоста позволяет развертывать приложения на любой доступной рабочей системе, которая отвечает потребностям приложения (например, имеет нужное количество доступной памяти или ресурсов процессора).

Имейте в виду, что понимание того, как настраиваются отдельные приложения и запускаются службы, по-прежнему является основой для этих более продвинутых способов управления ресурсами центра обработки данных. Подробное описание корпоративных средств развертывания и мониторинга выходит за рамки книги. Обратитесь к части VI «Работа с облачными вычислениями», чтобы получить представление о том, как различные облачные платформы на базе Linux справляются с этими проблемами.

Резюме

В системах Linux доступно множество различных типов серверов, однако основная процедура установки и настройки сервера, по существу, одинакова для всех.

Обычный список действий заключается в установке, настройке, запуске, защите и мониторинге серверов. Основные задачи, реализуемые всеми серверами, — это

использование сетевых инструментов (в частности, инструментов SSH) для входа в систему, копирование файлов или выполнение удаленных команд.

Поскольку администратор не может постоянно наблюдать за серверами, инструменты для сбора данных и последующего просмотра журнала очень важны при администрировании серверов Linux. Функцию `rsyslog` можно задействовать для локальных и удаленных журналов. Функция `sar` позволяет собрать текущие данные или воспроизвести данные, собранные ранее с интервалом 10 минут. Веб-интерфейс Cockpit позволяет наблюдать за процессором, памятью, диском и сетевой активностью в режиме реального времени. Чтобы следить за пространством диска, используйте команды `df` и `du`.

Навыки, описанные в этой главе, позволяют создать основу для управления системой на предприятии в будущем. Они полезны, однако, чтобы управлять множеством систем Linux одновременно, необходимо расширить их круг с помощью автоматизированных средств развертывания и мониторинга, как описано в разделе об облачных вычислениях в этой книге.

Можно легко настроить сеть для доступа к серверам по умолчанию, однако более сложная ее конфигурация требует знания файлов конфигурации сети и связанных с ними инструментов. В следующей главе описывается, как настроить и администрировать сетевую работу в Linux.

Упражнения

Упражнения в этом разделе охватывают основные инструменты для подключения к серверам Linux и наблюдения за ними. Как обычно, их можно решить несколькими способами. Поэтому не волнуйтесь, если выполните упражнения не так, как показано в ответах к ним, главное — получить те же результаты. Если затрудняетесь с решением заданий, посмотрите ответы к упражнениям, приведенные в приложении Б.

Некоторые упражнения предполагают, что у вас есть вторая доступная система Linux, в которую вы можете войти и попробовать различные команды. Нужно убедиться, что во второй системе служба `sshd` запущена, брандмауэр открыт и команда `ssh` разрешена в учетной записи пользователя, в которую нужно войти (суперпользователь часто блокируется службой `sshd`).

Если у вас есть только одна система Linux, создайте дополнительную учетную запись пользователя и имитируйте связь с другой системой, подключившись вместо этого к имени `localhost`, как показано в этом примере:

```
# useradd joe
# passwd joe
# ssh joe@localhost
```

1. С помощью команды `ssh` войдите на другой компьютер (или локальный компьютер), используя любую учетную запись, к которой у вас есть доступ. Введите пароль при появлении приглашения командной строки.

2. Применив функцию удаленного выполнения команд с помощью команды `ssh`, отобразите содержимое удаленного файла `/etc/system-release` в локальной системе.
3. Введите команду `ssh`, чтобы применить переадресацию X11 для отображения окна `gedit` в своей локальной системе, а затем сохраните файл в домашнем каталоге удаленного пользователя.
4. Рекурсивно скопируйте все файлы из каталога `/usr/share/selinux` в удаленной системе в каталог `/tmp` в своей локальной системе таким образом, чтобы все время изменения файлов обновилось до времени в локальной системе после копирования.
5. Рекурсивно скопируйте все файлы из каталога `/usr/share/logwatch` удаленной системы в каталог `/tmp` локальной системы таким образом, чтобы все время изменения файлов из удаленной системы сохранялось в локальной системе.
6. Создайте пару «открытый/закрытый ключ», чтобы использовать SSH-соединение (без ключевой фразы на ключе), скопируйте файл открытого ключа в учетную запись удаленного пользователя с помощью команды `ssh-copy-id` и примените аутентификацию на основе ключа для входа в эту учетную запись пользователя так, чтобы не требовалось вводить пароль.
7. Создайте запись в файле `/etc/rsyslog.conf`, который хранит все сообщения аутентификации (`authpriv`) информационного уровня и выше в файле с именем `/var/log/myauth`. Наблюдайте за файлом с одного терминала по мере поступления в него данных, а с другого терминала попытайтесь войти в свою локальную систему через `ssh` как пользователь с неверным паролем.
8. Примените команду `du`, чтобы определить самые большие структуры каталогов в каталоге `/usr/share`, отсортировать их от самых больших к самым маленьким и перечислить десять самых больших каталогов.
9. Задействуйте команду `df`, чтобы отобразить пространство, которое используется и доступно для всех файловых систем, подключенных в данный момент к локальной системе, но исключите любые файловые системы `tmpfs` или `devtmpfs`.
10. Найдите в каталоге `/usr` все файлы размером более 10 Мбайт.

14 Администрирование сети

В этой главе

- Автоматическое подключение Linux к сети.
- Просто подключение к сети с помощью программы NetworkManager.
- Настройка сети из командной строки.
- Работа с файлами конфигурации сети.
- Настройка маршрутизации, DHCP, DNS и других функций сети для предприятия.

Подключение настольного компьютера или ноутбука к сети, особенно к Интернету, стало настолько простым делом, что я решил отложить главу, описывающую весь процесс работы сети в Linux, до этого момента. В зависимости от того, какой сетевой интерфейс доступен, проводной или беспроводной, вы можете подключить Fedora, RHEL, Ubuntu или другую систему Linux к Интернету таким образом.

- **Проводное соединение.** Если в вашем доме или офисе есть проводной порт Ethernet, который обеспечивает доступ в Интернет, и если его можно подсоединить к компьютеру, используйте кабель Ethernet для подключения этих двух портов. После включения компьютера загрузите Linux и войдите в систему. Нажмите на значок NetworkManager на рабочем столе, который показывает, что вы подключены к Интернету, или позволяет подключиться одним щелчком кнопкой мыши.
- **Беспроводное соединение.** Для беспроводного подключения компьютера с Linux войдите в систему и щелкните на значке NetworkManager на рабочем столе. В появившемся списке беспроводных сетей выберите нужную и при появлении запроса введите нужный пароль. Каждый раз при входе в систему с этого компьютера из одного и того же места он будет автоматически подключаться к выбранной беспроводной сети.

Этой информации достаточно, если вас устраивает один из этих вариантов и вы не интересуетесь тем, как работает сеть в Linux. Но что делать, если ваша система Linux не подключается автоматически к Интернету? Что делать, если вы хотите соединить рабочий компьютер с частной сетью на работе (по VPN)? И что делать, если вы хотите заблокировать сетевые настройки на своем сервере или настроить систему Linux для работы в качестве маршрутизатора?

В этой главе темы охватывают работу сети для настольных компьютеров, серверов и корпоративных вычислений. Общий подход к настройке сети в этих трех типах систем Linux заключается в следующем.

- **Сеть для компьютеров/ноутбуков.** На настольных компьютерах или ноутбуках программа NetworkManager запускается по умолчанию и позволяет управлять сетевыми интерфейсами. С помощью этой программы вы также можете автоматически принимать адрес сервера и информацию о нем, чтобы подключиться к Интернету. Эти сведения можно задать и вручную. Вы можете настроить прокси-серверы и виртуальные частные сетевые подключения, позволяющие компьютеру работать от брандмауэра организации или подключаться через него соответственно.
- **Сеть для серверов.** Изначально программа NetworkManager предназначалась для работы на настольных компьютерах и ноутбуках, однако сейчас ее активно используют и для настройки сети на серверах. В программу теперь входят функции, полезные для настройки серверов, такие как соединение каналов Ethernet и настройка псевдонимов.
- **Сеть на предприятии.** Тема настройки сети на крупном предприятии может занять несколько томов. Однако я расскажу об основных сетевых технологиях, таких как DHCP- и DNS-серверы, которые позволяют настольным системам автоматически подключаться к Интернету и находить системы на основе имен, а не только IP-адресов.

Настройка сети настольных компьютеров

Независимо от того, подключаетесь ли вы к Интернету в системе Linux или Windows, через смартфон или любое другое сетевое устройство, чтобы соединение заработало, необходимо соблюсти определенные условия. Компьютер должен иметь сетевой интерфейс (проводной или беспроводной), IP-адрес, назначенный DNS-сервер и маршрут к Интернету (идентифицируемый устройством шлюза).

Прежде чем я расскажу о том, как изменить конфигурацию сети в Linux, рассмотрим, что происходит, когда Linux настроен на автоматическое подключение к Интернету с помощью программы NetworkManager.

- **Активация сетевых интерфейсов.** Утилита NetworkManager смотрит, какие сетевые интерфейсы (проводные или беспроводные) настроены. По умолча-

нию внешние интерфейсы обычно запускаются автоматически с помощью службы DHCP, хотя статические имена и адреса можно добавить и во время установки.

- **Запрос в службу DHCP.** Система Linux действует как DHCP-клиент и отправляет запрос в службу DHCP для каждого подключенного интерфейса. Она использует MAC-адрес сетевого интерфейса, идентифицируя себя в запросе.
- **Ответ от DHCP-сервера.** DHCP-сервер, работающий на беспроводном маршрутизаторе, кабельном модеме или другом устройстве, обеспечивающем доступ в Интернет, отвечает на запрос DHCP-клиента и предоставляет различные типы информации. Эта информация может содержать следующее.
  - **IP-адрес.** DHCP-сервер содержит диапазон IP-адресов, которые он может выдать любой системе в сети после запроса. В более безопасных средах или в случаях, когда необходимо обеспечить конкретные компьютеры конкретными адресами, DHCP-сервер предоставляет определенный IP-адрес для запросов с определенных MAC-адресов. (MAC-адреса должны быть уникальными среди всех карт сетевого интерфейса.)
  - **Маска подсети.** Когда DHCP-клиенту назначается IP-адрес, сопровождающая маска подсети сообщает этому клиенту, какая часть IP-адреса идентифицирует работу подсети, а какая — хост. Например, IP-адрес 192.168.0.100 и маска подсети 255.255.255.0 сообщают клиенту, что сеть — это 192.168.0, а хост — это 100.
  - **Срок аренды.** Когда DHCP-клиенту (в системе Linux) выделяется IP-адрес, ему назначается время аренды. Клиент не владеет этим адресом, он должен арендовать его снова и, когда истечет время, запросить еще раз после перезапуска сетевого интерфейса. Обычно DHCP-сервер запоминает клиента и назначает тот же адрес, когда система снова запускается или просит продлить аренду. Время аренды по умолчанию обычно составляет 86 400 секунд (24 часа) для IPV4-адресов. Для более избыточных IPv6-адресов аренда по умолчанию назначается на 2 592 000 секунд (30 дней).
  - **Сервер доменных имен.** Компьютеры думают цифрами (например, IP-адресами, такими как 192.168.0.100), а люди склонны думать именами (например, именем хоста www.example.com), поэтому компьютерам необходимо уметь переводить имена хостов в IP-адреса, а иногда и наоборот. Система доменных имен (Domain Name System, DNS) была разработана, чтобы решить эту проблему, предоставляя возможность сопоставления имен и адресов в Интернете с помощью иерархии серверов. Расположение одного или нескольких DNS-серверов (чаще всего двух или трех) обычно назначается DHCP-клиенту с хоста DHCP.
  - **Шлюз по умолчанию.** Интернет обладает одним уникальным пространством имен, но на самом деле он организован как ряд взаимосвязанных подсетей.

Чтобы сетевой запрос покинул локальную сеть, он должен знать, какой узел в сети обеспечивает маршрут к адресам за ее пределами. DHCP-сервер обычно предоставляет IP-адрес маршрута по умолчанию. Такой маршрут направляет данные к конечному пункту соединения, используя сетевые интерфейсы как в локальной подсети, так и в конечной.

- **Другая информация.** DHCP-сервер можно настроить так, чтобы он предоставлял все виды информации и помощь DHCP-клиенту. Например, он может указать местоположение NTP-сервера (для синхронизации времени между клиентами), сервера шрифтов (для получения шрифтов для X-дисплея), IRC-сервера (для онлайн-чатов) или сервера печати (для определения доступных принтеров).

- **Обновление параметров локальной сети.** После установки настроек для DHCP-сервера они соответствующим образом реализуются в локальной системе Linux. Например, IP-адрес задается в сетевом интерфейсе, записи DNS-сервера добавляются в локальный файл `/etc/resolv.conf` (NetworkManager), а время аренды хранится в локальной системе, чтобы она смогла запросить продление аренды.

Все только что описанные действия обычно происходят без вмешательства пользователя, нужно только включить систему Linux и войти в нее. Теперь предположим, что вы хотите иметь возможность проверить свои сетевые интерфейсы или изменить некоторые из перечисленных настроек. Можете сделать это с помощью инструментов, описанных далее.

Проверка сетевых интерфейсов

В Linux доступны как графические, так и командные инструменты для просмотра информации о сетевых интерфейсах. Для начала рассмотрим инструменты рабочего стола — программу NetworkManager и веб-интерфейс Cockpit.

Проверка сети с помощью программы NetworkManager

Самый простой способ проверить базовые настройки сетевого интерфейса — открыть раскрывающееся меню в правом верхнем углу рабочего стола и выбрать активный сетевой интерфейс. На рис. 14.1 показаны настройки Wi-Fi для активной сети на рабочем столе Fedora GNOME 3.

Как видите, интерфейсу назначены и IPv4-, и IPv6-адреса. IP-адрес 192.168.1.254 включает в себя как службу DNS, так и маршрут к внешним сетям.

Чтобы больше узнать о том, как настроена ваша система Linux, перейдите на одну из вкладок в верхней части окна. Например, на рис. 14.2 показана вкладка Security (Безопасность), на которой можно выбрать тип защищенного подключения к сети и установить пароль для подключения к ней.

Рис. 14.1. Проверка сетевых интерфейсов с помощью программы NetworkManager

Рис. 14.2. Просмотр сетевых настроек с помощью программы NetworkManager

Проверка сети с помощью веб-интерфейса Cockpit

Подключив веб-интерфейс Cockpit, вы сможете просматривать и изменять информацию о своих сетевых интерфейсах через браузер. В локальной системе в браузере введите `localhost:9090/network`, чтобы перейти непосредственно на страницу Networking (Сеть). На рис. 14.3 приведен пример окна Cockpit.

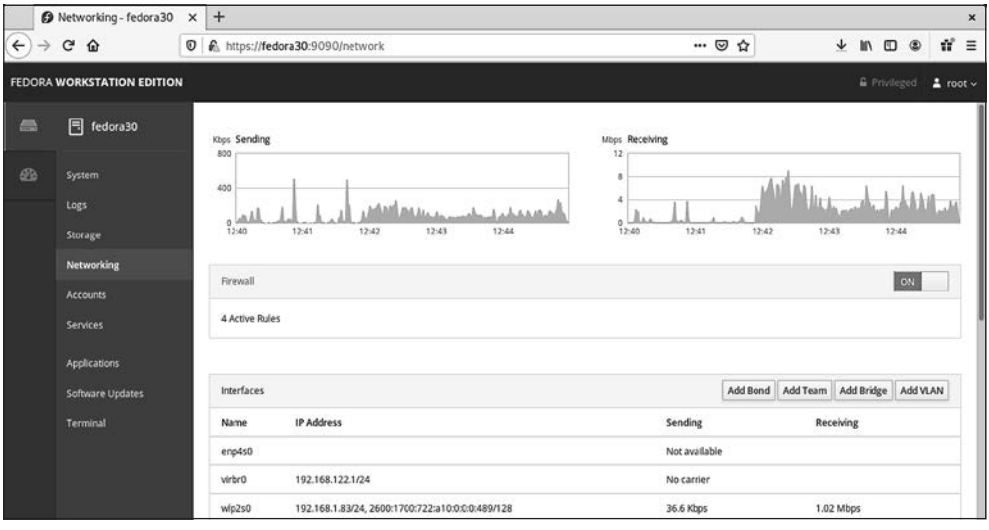


Рис. 14.3. Просмотр и изменение сетевых настроек из веб-интерфейса Cockpit

В разделе Networking (Сеть) показана информация обо всех ваших сетевых интерфейсах сразу. В данном случае представлены три сетевых интерфейса: `wlp2s0` (активный беспроводной интерфейс), `enp4s0` (неактивный проводной интерфейс) и `virbr0` (неактивный интерфейс в сети, подключенный к любым виртуальным машинам, работающим в локальной системе).

В верхней части раздела Networking (Сеть) вы можете увидеть данные, которые принимаются (Receiving (Прием)) и отправляются (Sending (Отправка)) в локальной системе. Выберите сетевой интерфейс, чтобы увидеть страницу, отображающую действия для этого конкретного интерфейса.

Выберите раздел Firewall (Брандмауэр), чтобы увидеть список служб, разрешенных в системе. Например, на рис. 14.4 показано, что порты UDP открыты для трех служб: клиента DHCPv6, многоадресного DNS и клиента Samba. Клиент DHCPv6 позволяет системе получать IPv6-адрес из сети. Многоадресные DNS-службы и клиентские службы Samba позволяют автоматически определять принтеры, общие файловые системы и ресурсы, а также различные устройства.

Единственная активная служба TCP в примере — это `ssh`. При активной службе `ssh` (TCP-порт 22) служба `sshd`, запущенная в локальной системе, позволяет удаленным пользователям войти в вашу локальную систему.

Тот факт, что эти порты открыты, не обязательно означает, что службы работают. Но если они запущены, межсетевой экран (брандмауэр) компьютера разрешит доступ к ним.

Более продвинутые функции раздела Networking (Сеть) в Cockpit позволяют добавлять связи, команды, мосты и VLAN к вашим локальным сетевым интерфейсам.

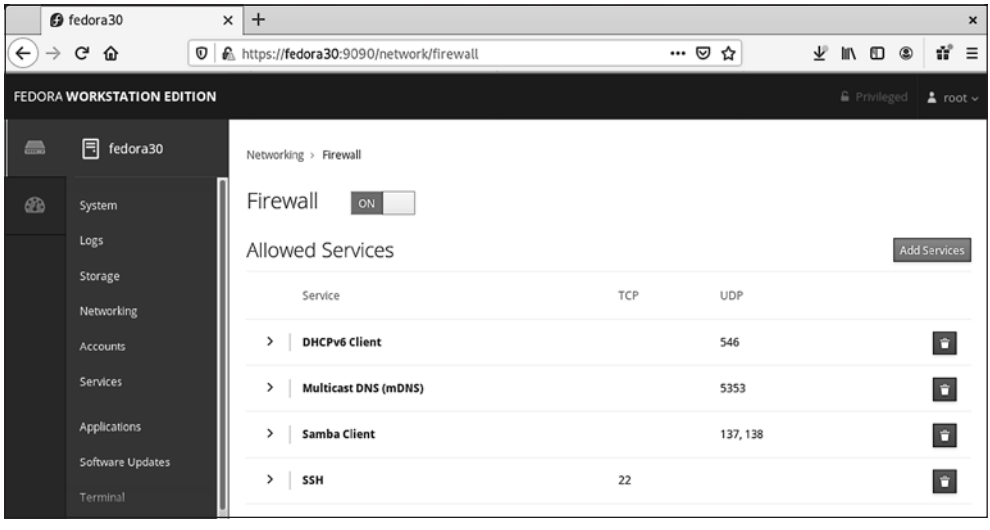


Рис. 14.4. Просмотр служб, доступных через межсетевой экран из веб-интерфейса Cockpit

Проверка сети из командной строки

Чтобы подробнее узнать о сетевых интерфейсах, попробуйте выполнить несколько команд. Существуют команды, которые могут отобразить информацию о сетевых интерфейсах, маршрутах, хостах и трафике в сети.

Просмотр сетевых интерфейсов. Чтобы просмотреть информацию о каждом сетевом интерфейсе в локальной системе Linux, введите следующее:

```
# ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue
        state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp4s0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500
        qdisc fq_codel state DOWN group default qlen 1000
    link/ether 30:85:a9:04:9b:f9 brd ff:ff:ff:ff:ff:ff
3: wlp2s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500
        qdisc mq state UP group default qlen 1000
    link/ether e0:06:e6:83:ac:c7 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.83/24 brd 192.168.1.255 scope global
        dynamic noprefixroute wlp2s0
        valid_lft 78738sec preferred_lft 78738sec
    inet6 2600:1700:722:a10::489/128 scope global dynamic
noprefixroute
        valid_lft 5529sec preferred_lft 5229sec
```

```
    inet6 fe80::25ff:8129:751b:23e3/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
4: virbr0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500
       qdisc noqueue state DOWN group default qlen 1000
    link/ether 52:54:00:0c:69:0a brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.1/24 brd 192.168.122.255 scope global virbr0
       valid_lft forever preferred_lft forever
```

Команда `ip addr show` выводит информацию о сетевых интерфейсах, в данном случае с ноутбука под управлением системы Fedora 30.

Запись `lo` в первой строке вывода показывает кольцевой интерфейс, который используется для подключения сетевых команд, выполняемых в локальной системе, к самой локальной системе. IP-адрес для локального хоста — это 127.0.0.1/8 (/8 — обозначение бесклассовой адресации CIDR, указывающее, что 127.0 — это номер сети, а 0.1 — номер хоста). Добавьте параметр `-s` (`ip-s addr show`), чтобы увидеть статистику пакетных передач и ошибок, связанных с каждым интерфейсом.

В этом случае проводной интерфейс Ethernet (`enp4s0`) отключен (нет кабеля), но беспроводной интерфейс включен (`wlp2s0`). MAC-адрес в беспроводном интерфейсе (`wlp2s0`) — это e0:06:e6:83:ac:c7 и интернет-адрес (IPv4) — это 192.168.1.8. IPv6-адрес также включен.

Более старые версии Linux применяются для назначения общих имен сетевых интерфейсов, таких как `eth0` и `wlan0`. Теперь интерфейсы называются по их расположению на шине компьютера. Например, первый порт на сетевой карте, размещенной на третьей шине PCI для системы Fedora, называется `p3p1`. Первым встроенным портом Ethernet будет `m1`. Бывает, что беспроводные интерфейсы используют имя беспроводной сети в качестве имени устройства.

Другая популярная команда для просмотра информации о сетевом интерфейсе — `ifconfig`. По умолчанию она отображает ту же информацию, что и команда `ip addr`, а еще, тоже по умолчанию, — количество пакетов, принятых (`RX`) и переданных (`TX`), а также количество данных, любые ошибки или потери пакетов:

```
# ifconfig wlp2s0
wlp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 192.168.1.83 netmask 255.255.255.0
            broadcast 192.168.1.255
        inet6 2600:1700:722:a10:b55a:fca6:790d:6aa6
            prefixlen 64 scopeid 0x0<global>
        inet6 fe80::25ff:8129:751b:23e3
            prefixlen 64 scopeid 0x20<link>
        inet6 2600:1700:722:a10::489
            prefixlen 128 scopeid 0x0<global>
        ether e0:06:e6:83:ac:c7 txqueuelen 1000 (Ethernet)
        RX packets 208402 bytes 250962570 (239.3 MiB)
        RX errors 0 dropped 4 overruns 0 frame 0
        TX packets 113589 bytes 13240384 (12.6 MiB)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
Checking connectivity to remote systems
```

Чтобы убедиться, что вы можете связаться с системами, доступными в сети, используйте команду `ping`. Она отправляет специальные пакеты в удаленную систему, ожидая от них ответа, например:

```
$ ping host1
PING host1 (192.168.1.15 ) 56(84) bytes of data.
64 bytes from host1 (192.168.1.15 ): icmp_seq=1 ttl=64 time=0.062 ms
64 bytes from host1 (192.168.1.15 ): icmp_seq=2 ttl=64 time=0.044 ms
^C
--- host1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1822ms
rtt min/avg/max/mdev = 0.044/0.053/0.062/0.009 ms
```

Команда `ping` в примере непрерывно пингует хост с именем `host1`. После нескольких пингов нажмите сочетание клавиш Ctrl+C, чтобы завершить процесс, и в последних нескольких строках вывода будет показано, сколько запросов команде удалось выполнить.

Можно было бы использовать и IP-адрес (в данном случае 192.168.1.15), чтобы проверить, есть ли возможность связаться с системой. Однако имя хоста дает вам дополнительное преимущество: вы знаете, что перевод этого имени в IP-адрес (выполняется DNS-сервером или файлом локального хоста) выполнен и работает правильно. Однако в этом случае хост `host1` появится в локальном файле `/etc/hosts`.

Проверка информации о маршруте. Маршрутизация — это следующий пункт, который можно проверить при настройке сетевых интерфейсов. В следующих примерах показано, как применить для этого команды `ip` и `route`:

```
# ip route show
default via 192.168.122.1 dev ens3 proto dhcp metric 20100
192.168.122.0/24 dev ens3 proto kernel scope link src 192.168.122.194
metric 100
```

В примере команда `ip route show` показывает, что адрес 192.168.122.1 обеспечивает маршрут к хосту из виртуальной машины RHEL 8 по сетевому интерфейсу `ens3`. Соединение с любым адресом в диапазоне 192.168.122.0/24 от виртуальной машины (192.168.122.194) проходит через этот интерфейс. Команда `route` предоставляет аналогичную информацию:

```
# route
Kernel IP routing table
Destination     Gateway     Genmask         Flags  Metric  Ref  Use  Iface
default         homeportal  0.0.0.0         UG     600     0      0  wlp2s0
192.168.1.0     0.0.0.0     255.255.255.0   U      600     0      0  wlp2s0
192.168.122.0   0.0.0.0     255.255.255.0   U      0       0      0  virbr0
```

Вывод таблицы маршрутизации осуществляется из системы Fedora с одним активным внешним сетевым интерфейсом. Карта беспроводного сетевого интерфейса находится на слоте PCI 2, порт 1 (`wlp2`).

Любые пакеты, предназначенные для сети 192.168.1, используют карту сетевого интерфейса `wlp2`. Пакеты, предназначенные для любого другого соединения, пересылаются в систему шлюза по адресу 192.168.1.0. Эта система представляет собой маршрутизатор для Интернета (в данном случае мой). Вот так выглядит более сложная таблица маршрутизации:

```
# route
Kernel IP routing table
Destination   Gateway       Genmask          Flags  Metric Ref  Use  Iface
default       gateway       0.0.0.0          UG     600    0      0  wlp3s0
10.0.0.0      vpn.example.  255.0.0.0        U      50     0      0  tun0
10.10.135.0   0.0.0.0       255.255.217.0    U      50     0      0  tun0
vpn.example.  gateway       255.255.255.255  UGH    600    0      0  wlp3s0
172.17.0.0    0.0.0.0       255.255.0.0      U      0      0      0  docker0
192.168.1.0   *             255.255.255.0    U      600    0      0  wlp3s0
```

В примере маршрута показан беспроводной интерфейс (`wlp3s0`), а также интерфейс, представляющий туннель виртуальной частной сети (Virtual Private Network, VPN). VPN обеспечивает зашифрованную частную связь между клиентом и удаленной сетью через небезопасную сеть, например Интернет. Здесь туннель идет от локальной системы по интерфейсу `wlan0` к хосту с именем `vpn.example.com` (часть имени сокращена).

Вся связь с сетью 192.168.1.0/24 по-прежнему идет непосредственно по беспроводной локальной сети. Однако пакеты, предназначенные для сетей 10.10.135.0/24 и 10.0.0.0/8, направляются непосредственно в `vpn.example.com` для связи с хостами на другой стороне VPN-соединения через туннель (`tun0`).

Для связи с контейнерами (`docker0`), работающими в локальной системе по сети 172.17.0.0, устанавливается специальный маршрут. Все остальные пакеты идут по стандартному маршруту (по умолчанию) через адрес 192.168.1.0. Что касается флагов, показанных в выходных данных, то `U` говорит, что маршрут открыт, `G` идентифицирует интерфейс как шлюз, а `H` говорит, что конечной целью является хост (как в случае с VPN-соединением).

До сих пор я показывал маршруты выхода из локальной системы. Чтобы проследить маршрут до хоста от начала до конца, используйте команду `traceroute` (`dnf install traceroute`). Например, чтобы проследить маршрут, по которому пакет идет от локальной системы до сайта google.com, введите следующую команду `traceroute`:

```
# traceroute google.com
traceroute to google.com (74.125.235.136), 30 hops max, 60 byte pkts
...
 7 rrcs-70-62-95-197.midsouth.biz.rr.com (70.62.95.197) ...
 8 ge-2-1-0.rlghncpop-rtr1.southeast.rr.com (24.93.73.62) ...
 9 ae-3-0.cr0.dca10.tbone.rr.com (66.109.6.80) ...
10 107.14.19.133 (107.14.19.133) 13.662 ms ...
11 74.125.49.181 (74.125.49.181) 13.912 ms ...
12 209.85.252.80 (209.85.252.80) 61.265 ms ...
```

```
13 66.249.95.149 (66.249.95.149) 18.308 ms ...
14 66.249.94.22 (66.249.94.22) 18.344 ms ...
15 72.14.239.83 (72.14.239.83) 85.342 ms ...
16 64.233.174.177 (64.233.174.177) 167.827 ms ...
17 209.85.255.35 (209.85.255.35) 169.995 ms ...
18 209.85.241.129 (209.85.241.129) 170.322 ms ...
19 nrt19s11-in-f8.1e100.net (74.125.235.136) 169.360 ms ...
```

Я сократил часть выходных данных, чтобы убрать начальные маршруты и количество времени (в миллисекундах), которое требовалось пакетам для прохождения каждого маршрута. С помощью команды `traceroute` можно увидеть, где произошла остановка, если сетевая связь перестала работать.

Просмотр хоста и доменных имен. Чтобы увидеть имя хоста, назначенное локальной системе, введите команду `hostname`. Чтобы увидеть доменную часть этого имени, используйте команду `dnsdomainname`:

```
# hostname
spike.example.com
# dnsdomainname
example.com
```

Настройка сетевых интерфейсов

Если вы не хотите, чтобы сетевые интерфейсы назначались автоматически с DHCP-сервера (или если DHCP-сервера нет), можете настроить их вручную. Это включает в себя назначение IP-адресов, размещение DNS-серверов и шлюзов, а также маршрутов, что можно настроить с помощью NetworkManager.

Настройка IP-адресов вручную

Чтобы изменить конфигурацию сети для проводного сетевого интерфейса с помощью NetworkManager, выполните следующие действия.

1. Выберите значок настроек в раскрывающемся меню в правом верхнем углу рабочего стола, а затем пункт Network (Сеть).
2. Предположим, у вас есть карта сетевого интерфейса, которая еще не используется. В этом случае нажмите кнопку настроек (маленький значок шестеренки) рядом с тем сетевым интерфейсом, который хотите изменить.
3. Выберите раздел IPv4 и измените метод IPv4 с варианта Automatic (Автоматический) (DHCP) на вариант Manual (Вручную).
4. Введите следующую информацию (требуются только Address (Адрес) и Netmask (Маска сети)).
   - Address (Адрес). IP-адрес, который вы хотите назначить своему локальному сетевому интерфейсу, например 192.168.100.100.

- Netmask (Маска сети). Маска подсети, которая определяет, какая часть IP-адреса представляет сеть, а какая — хост. Например, сетевая маска 255.255.255.0 идентифицирует сетевую часть предыдущего адреса как 192.168.100, а хост-часть — как 100.
- Gateway (Шлюз). IP-адрес компьютера или устройства в сети, который действует как маршрут по умолчанию. Он направляет пакеты из локальной сети на любой адрес, недоступный в локальной сети, или по какому-то другому пользовательскому маршруту.
- DNS servers (Серверы DNS). Заполните IP-адреса системы, которая предоставляет службу DNS вашему компьютеру. Если существует более одного DNS-сервера, добавьте остальные в список серверов, разделяя их запятыми.

5. Нажмите кнопку Apply (Применить). Новая информация сохранится, и сеть перезапустится с использованием новой информации. На рис. 14.5 показан пример сетевых настроек.

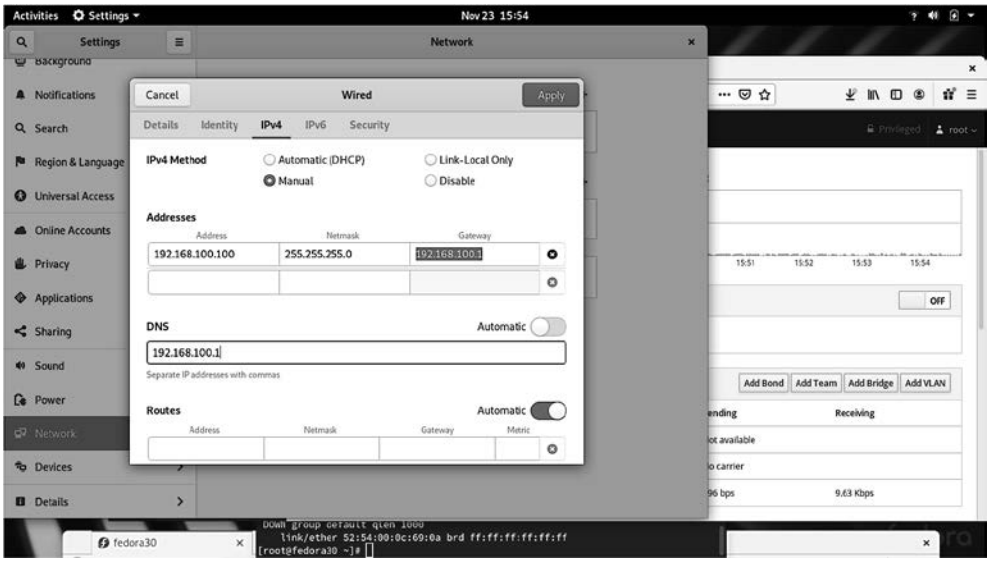


Рис. 14.5. Изменение сетевых настроек с помощью программы NetworkManager

Настройка псевдонимов IP-адресов

Вы можете присоединить несколько IP-адресов к одному сетевому интерфейсу. Для этого на том же экране NetworkManager нужно заполнить поля последующих адресов в области Addresses (Адреса) и добавить новую информацию об IP-адресе. Вот что следует знать о добавлении псевдонимов к адресам.

- Для каждого адреса требуется маска сети, шлюз при этом не нужен.
- Кнопка Apply (Применить) остается неактивной до тех пор, пока вы не введете в поля актуальную информацию.
- Новый адрес не обязательно должен находиться в той же подсети, что и исходный адрес, хотя он считывает трафик в той же физической сети.

После того как я добавил адрес 192.168.100.103 к своему проводному интерфейсу, команда `ip addr show enp4s0` отобразила следующие два IP-адреса на интерфейсе:

```
2: enp4s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel
state UP group default qlen 1000
    link/ether 30:85:a9:04:9b:f9 brd ff:ff:ff:ff:ff:ff
    inet 192.168.100.100/24 brd 192.168.100.255 scope
            global noprefixroute enp4s0
        valid_lft forever preferred_lft forever
    inet 192.168.100.103/24 brd 192.168.100.255 scope
            global secondary noprefixroute enp4s0
        valid_lft forever preferred_lft forever
```

Дополнительные сведения о настройке псевдонимов непосредственно в файлах конфигурации см. в подразделе «Настройка псевдонимов сетевых интерфейсов» далее в этой главе.

Установка маршрутов

При запросе подключения к IP-адресу ваша система просматривает таблицу маршрутизации, чтобы определить путь, по которому можно подключиться к данному адресу. Информация передается в виде пакетов. В зависимости от назначения пакет маршрутизируется следующими способами.

- Локальная система отправляется на интерфейс `lo`.
- Система в локальной сети направляется через вашу карту сетевого интерфейса непосредственно на карту сетевого интерфейса предполагаемой системы-получателя.
- Любая другая система отправляется на шлюз (маршрутизатор), который затем направляет пакет по назначенному адресу в Интернете.

Конечно, то, что я только что описал, — это самый простой вариант развития событий. На самом деле у вас может быть несколько карт сетевого интерфейса со множеством интерфейсов различных сетей. А еще может быть несколько маршрутизаторов в локальной сети, которые обеспечивают доступ к другим частным сетям. Например, у вас есть маршрутизатор (или другая система, действующая в качестве маршрутизатора) в локальной сети, вы можете добавить пользовательский маршрут к этому маршрутизатору через утилиту NetworkManager. Задействуя

предыдущий пример с помощью этой утилиты, прокрутите страницу вниз, чтобы увидеть раздел Routes (Маршруты). Затем добавьте следующее.

- Address (Адреса). Сетевой адрес подсети, к которой вы хотите подключиться. Например, если маршрутизатор (шлюз) предоставит вам доступ ко всем системам в сети 192.168.200, добавьте адрес 192.168.200.0.
- Netmask (Маска сети). Добавьте сетевую маску, необходимую для идентификации подсети. Например, если маршрутизатор предоставляет доступ к адресу класса C 192.168.200, вы можете использовать сетевую маску 55.255.255.0.
- Gateway (Шлюз). Добавьте IP-адрес маршрутизатора (шлюза), который обеспечит доступ к новому маршруту. Например, если маршрутизатор имеет IP-адрес 192.168.1 в вашей сети 192.168.1.199, добавьте этот адрес в поле.

Нажмите кнопку Apply (Применить), чтобы применить новые данные. Возможно, вам придется перезапустить интерфейс, чтобы изменения вступили в силу (например, командой `ifup enp4s0`). Введите команду `route -n`, чтобы убедиться, что новый маршрут был применен:

```
# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags  Metric  Ref  Use  Iface
0.0.0.0         192.168.100.1   0.0.0.0         UG     1024    0      0  p4p1
192.168.100.0   0.0.0.0         255.255.255.0   U      0       0      0  p4p1

192.168.200.0   192.168.1.199   255.255.255.0   UG     1       0      0  p4p1
```

В примере видно, что шлюз по умолчанию — это 192.168.100.1. Однако все пакеты, предназначенные для сети 192.168.200, маршрутизируются через узел шлюза по IP-адресу 192.168.1.199. Предположительно, этот хост имеет сетевой интерфейс сети 192.168.200 и настроен так, чтобы другие хосты могли маршрутизировать через него в эту сеть.

См. подраздел «Настройка пользовательских маршрутов» далее в этой главе, чтобы узнать, как задавать маршруты непосредственно в файлах конфигурации.

Настройка сетевого прокси

Если ваша настольная система работает через корпоративный брандмауэр, вы можете не иметь прямого доступа к Интернету. В этом случае, возможно, придется заходить в Интернет через прокси-сервер. Вместо того чтобы предоставить полный доступ к Интернету, прокси-сервер позволяет делать запросы только для определенных служб за пределами локальной сети. Затем прокси-сервер передает эти запросы в Интернет или другую сеть.

Прокси-серверы обычно предоставляют доступ к веб-серверам (http:// и https://) и FTP-серверам (ftp://). Однако прокси-сервер, поддерживающий протокол SOCKS (SOCKet Secure), может разрешать использование прокси-службы для различных

протоколов за пределами локальной сети. (SOCKS — это сетевой протокол, позволяющий компьютерам-клиентам получать доступ к Интернету через брандмауэр.) Чтобы идентифицировать прокси-сервер в программе NetworkManager и связать выбранные протоколы, проходящие через этот сервер, в окне Settings (Параметры) выберите вкладку Network (Сеть), а затем перейдите в настройки раздела Network Proxy (Сетевой прокси).

Вместо того чтобы идентифицировать прокси-сервер для сетевых интерфейсов (через программу NetworkManager), вы можете настроить свой браузер на прямое использование прокси-сервера, изменив предпочтения в браузере Firefox. Переключаться на прокси-сервер из окна браузера нужно так.

1. В окне браузера Firefox выберите раздел Preferences (Настройки) из выпадающего списка справа. Появится окно настроек для браузера.
2. В окне настроек прокрутите информацию вниз до пункта Network Settings (Параметры сети) и нажмите кнопку Settings (Настроить).
3. В появившемся окне можно попробовать автоматически определить настройки прокси-сервера или, если вы установили прокси-сервер в NetworkManager, выбрать вариант Use system proxy settings (Использовать системные настройки прокси). Можно также выбрать ручную настройку прокси-сервера, заполнить информацию, приведенную далее, и нажать кнопку OK.
   - HTTP Proxy (HTTP прокси). IP-адрес компьютера, предоставляющего прокси-службу. Пересылает все запросы на веб-страницах (http://protocol) в прокси-сервер.
   - Port (Порт). Порт, связанный с прокси-службой. По умолчанию номер порта 3128, но может быть и другим.
   - Use this proxy server for all protocols (Использовать этот прокси для FTP и HTTPS). Установите флажок в этом поле, чтобы задействовать те же прокси-сервер и порт для всех других запросов на обслуживание. Это приведет к тому, что другие настройки прокси-сервера станут неактивными. (Устанавливать этот флажок не обязательно, можно настроить каждую прокси-службу отдельно.)
   - No Proxy (Без прокси). Добавьте имя хоста или IP-адрес для любой системы, с которой хотите напрямую связаться через браузер Firefox, не используя прокси-сервер. В таком случае не нужно указывать в этом поле соединение с `localhost` и локальный IP-адрес (127.0.0.1), так как эти адреса уже будут перенаправляться.

На рис. 14.6 показан пример окна Configure Proxy Access to the Internet (Настройка прокси-доступа в Интернет). Здесь указана информация для настройки подключения к прокси-серверу, расположенному по IP-адресу 192.168.1.1, для всех протоколов. Нажмите кнопку OK, и все запросы из браузера Firefox в места за пределами локальной системы будут направлены через прокси на соответствующий сервер.

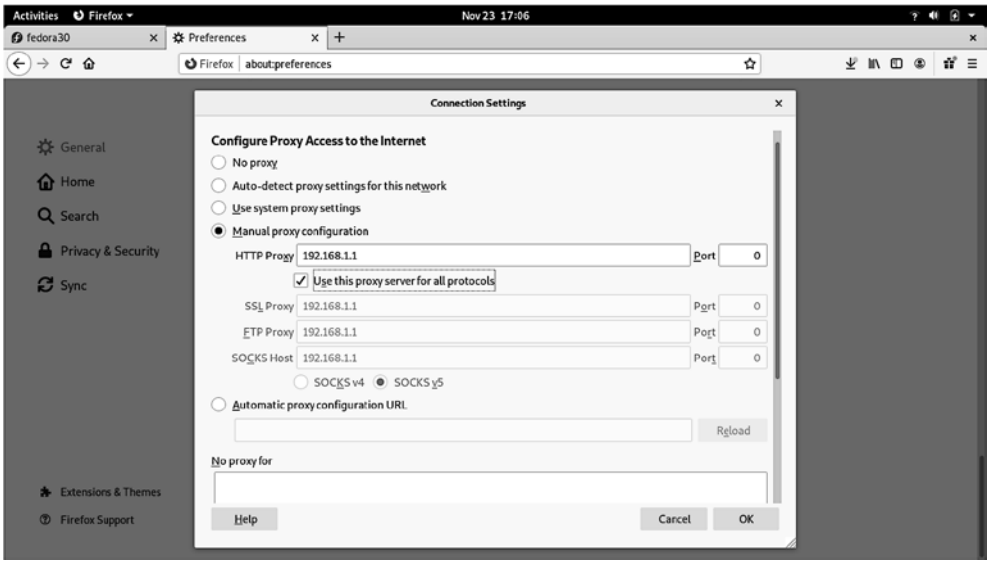


Рис. 14.6. Настройка использования прокси-сервера в браузере Firefox

Настройка сети из командной строки

Утилита NetworkManager отлично справляется с автоматическим обнаружением проводных сетей и предоставляет списки беспроводных. Случается, что нужно отказаться от графического интерфейса NetworkManager и Cockpit, чтобы настроить необходимые функции. Вот часть сетевых функций систем RHEL и Fedora, которые будут описаны в следующих разделах.

- **Базовая конфигурация.** Вы узнаете, как использовать команду `nmtui`, чтобы настроить основные сети через интерфейс меню в оболочке. Этот инструмент обеспечивает интуитивно понятный интерфейс для построения сетей на серверах, не имеющих графического интерфейса.
- **Файлы конфигурации.** Вы разберетесь в файлах конфигурации, связанных с сетью Linux, и в том, как их настроить.
- **Объединение каналов Ethernet.** Настройка объединения каналов Ethernet (несколько сетевых карт прослушивают один и тот же IP-адрес).

Настройка сети с помощью команды nmtui

Многие серверы не имеют графических интерфейсов. Поэтому, если вы хотите настроить работу сети, необходимо иметь возможность сделать это из оболочки. Один из способов — напрямую редактировать сетевые файлы конфигурации.

Другой способ — использовать команды меню, которые позволяют осуществлять навигацию по меню и настраивать сетевые интерфейсы с помощью клавиши Tab и клавиш со стрелками.

Команда `nmtui` (`yum install NetworkManager-tui`) предоставляет интерфейс на основе меню, который работает в оболочке. От имени суперпользователя введите команду `nmtui`, чтобы увидеть такой экран, как на рис. 14.7.

Рис. 14.7. Настройка сетевых параметров с применением утилиты NetworkManager TUI

С помощью клавиш ←, ↑, ↓ и → и Tab можно перемещаться по интерфейсу. Выделите нужный элемент меню и нажмите клавишу Enter, чтобы выбрать его. Интерфейс ограничен и позволяет выбрать следующие параметры: Edit a connection (Изменить соединение), Activate a connection (Подключиться), используя карты сетевого интерфейса, Set system hostname (Изменить имя хоста) с помощью имени хоста и конфигурации DNS.

Редактирование соединения в NetworkManager TUI

На экране NetworkManager TUI отображаются варианты действий с текущим соединением.

1. Edit a connection (Изменить соединение). Выделите этот вариант и нажмите клавишу Enter. Отобразятся список сетевых устройств (обычно проводных или беспроводных карт Ethernet), а также все беспроводные сети, к которым вы подключались в прошлом.
2. Network devices (Устройства сети). Выделите одно из сетевых устройств (я выбрал проводной интерфейс Ethernet) и нажмите клавишу Enter.
3. IPv4 Configuration (Конфигурация IPv4). Нажмите кнопку Show (Показать) напротив конфигурации IPv4 и клавишу Enter. Появится окно Edit Connection (Изменить соединение), которое позволяет отредактировать информацию, относящуюся к выбранному сетевому устройству.
4. Manual (Вручную). Вы можете оставить поля Profile Name (Имя профиля) и Device (Устройство) без изменений. По умолчанию включен вариант Automatic (Автоматически). Он позволяет сетевому интерфейсу автоматически появляться в сети, если доступна служба DHCP. Чтобы самостоятельно ввести адрес и другую информацию, используя клавишу Tab, выделите поле Automatic (Автоматически) и нажмите клавишу Пробел, затем с помощью клавиш ←, ↑, ↓ и → выделите вариант Manual (Вручную) и нажмите клавишу Enter.

5. Address (Адреса). Теперь введите информацию об адресе (IP-адрес и сетевую маску), например 192.168.0.150/24, где 24 — это бесклассовая междоменная маршрутизация (CIDR), которая является эквивалентом сетевой маски 255.255.255.0.
6. Gateway (Шлюз). Введите IP-адрес компьютера или маршрутизатора, который передаст маршрут в Интернет.
7. DNS servers (Серверы DNS). Введите IP-адреса одного или двух DNS-серверов, чтобы сообщить системе, куда направляться, и чтобы перевести имена запрашиваемых хостов в IP-адреса.
8. Search domains (Поиск доменов). Поиск доменов применяется при запросе хоста из приложения без использования полного доменного имени. Например, если вы добавите `ping host1` к пути поиска `example.com`, команда будет пытаться отправить пинг-пакеты на `host1.example.com`.
9. Routing (Маршрутизация). Вы можете создавать собственные маршруты, перейдя к пункту Routing (Маршрутизация) и выбрав его клавишей Enter. Заполните поля Destination/Prefix (Назначение/префикс) и Next Hop (Следующий переход) и нажмите кнопку OK, чтобы сохранить новый пользовательский маршрут.
10. Другие варианты. Из других вариантов на экране можно выбрать Never use this network for default route (Не использовать эту сеть для текущего маршрута), если сеть не подключается к более широким сетям, и Ignore automatically obtained routes (Игнорировать автоматически полученные маршруты), если вы не хотите, чтобы данные функции автоматически устанавливались из сети. На рис. 14.8 показан экран после выбора варианта настроек Manual (Вручную).

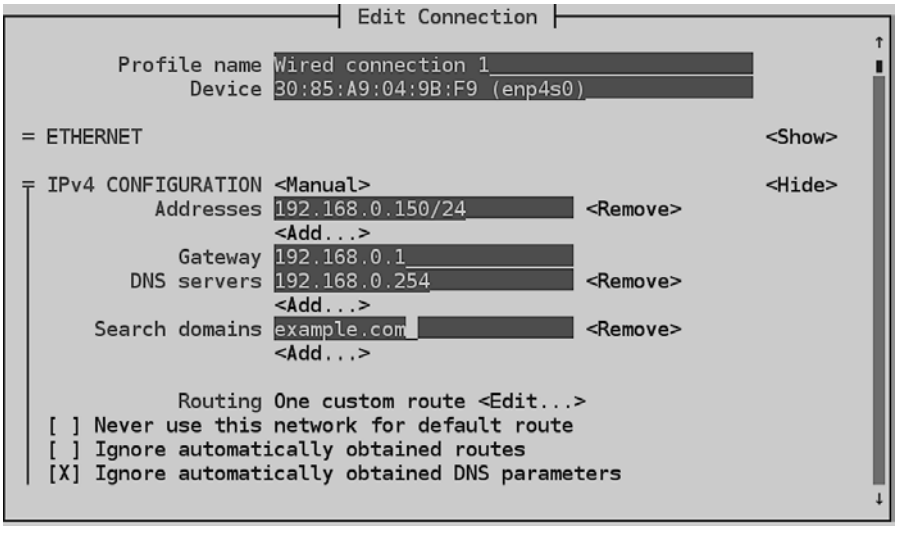


Рис. 14.8. Установите постоянные IP-адреса, выбрав пункт Manual (Вручную) на экране Edit Connection (Изменить соединение)

Перейдите к кнопке OK с помощью клавиши Tab и нажмите клавишу Пробел. Затем нажмите кнопку Quit (Выйти), чтобы выйти из утилиты.

Сетевые файлы конфигурации

Вне зависимости от того, что применяется для изменения сетевых настроек — NetworkManager или `nmtui`, большинство одинаковых файлов конфигурации обновляется. В системах Fedora и RHEL сетевые интерфейсы и пользовательские маршруты задаются в файлах каталога `/etc/sysconfig/network-scripts`.

Откройте файл `/usr/share/doc/initscripts/sysconfig.txt`, чтобы увидеть описания сетевых конфигурационных файлов (доступны в пакете `initscripts`).

Необходимо быть осторожными, так как программа NetworkManager считает, что она контролирует файлы в каталоге сетевых скриптов. Поэтому имейте в виду: если вы вручную зададите адреса в интерфейсе, который программа NetworkManager использует для службы DHCP, она может перезаписать изменения, внесенные вручную в файл.

Файлы сетевого интерфейса

Файлы конфигурации для каждого проводного, беспроводного, ISDN, коммутируемого или другого типа сетевого интерфейса представлены файлами в каталоге `/etc/sysconfig/network-scripts`, которые начинаются с `ifcfg-interface`. Обратите внимание на то, что слово `interface` заменяется именем сетевого интерфейса.

Если установить сетевой интерфейс проводной карты NIC как `enp4s0`, то вот так будет выглядеть пример файла `ifcfg-enp4s0` для этого интерфейса, настроенного на использование службы DHCP:

```
DEVICE=enp4s0
TYPE=Ethernet
BOOTPROTO=dhcp
ONBOOT=yes
DEFROUTE=yes
UUID=f16259c2-f350-4d78-a539-604c3f95998c
IPV4_FAILURE_FATAL=no
IPV6INIT=yes
IPV6_AUTOCONF=yes
IPV6_DEFROUTE=yes
IPV6_FAILURE_FATAL=no
NAME="System enp4s0"
PEERDNS=yes
PEERROUTES=yes
IPV6_PEERDNS=yes
IPV6_PEERROUTES=yes
```

В этом примере файла `ifcfg-enp4s0` первые две строчки устанавливают имя устройства и тип интерфейса для подключения к сетям Ethernet. Переменная `BOOTPROTO` имеет значение `dhcp`, что заставляет ее запрашивать адреса

с DHCP-сервера. Если имеется параметр ONBOOT=yes, то интерфейс запускается автоматически во время загрузки системы. Настройки для IPV6 указывают, что нужно инициализировать IPV6 и использовать их, но при этом интерфейс будет продолжать инициализироваться, даже если сеть IPV6 недоступна. Другие параметры настраивают применение параметра Use Peer DNS (статический IP-адрес) в автоматическом режиме и направление обнаруженных значений.

Вот как может выглядеть простой файл ifcfg-enp4s1 для проводного интерфейса Ethernet, использующего статические IP-адреса:

```
DEVICE=enp4s1
HWADDR=00:1B:21:0A:E8:5E
TYPE=Ethernet
BOOTPROTO=none
ONBOOT=yes
USERCTL=no
IPADDR=192.168.0.140
NETMASK=255.255.255.0
GATEWAY=192.168.0.1
```

В этом примере ifcfg-enp4s1 переменная BOOTPROTO имеет значение none, так как адрес и другая информация установлены статически. Другие изменения необходимы для задания информации об адресе, которую обычно собирают с DHCP-сервера. В этом случае устанавливается IP-адрес 192.168.0.140 с маской сети 255.255.255.0. Параметр GATEWAY=192.168.0.1 идентифицирует адрес маршрутизатора в Интернете.

Список других интересных настроек.

- PEERDNS. Установка настройки PEERDNS=n не дает службе DHCP перезаписывать файл /etc/resolv.conf. Она позволяет установить, какие DNS-серверы использует ваша система, не опасаясь, что эта информация будет стерта данными от DHCP-сервера.
- DNS?. Если программа NetworkManager управляет файлом ifcfg, то он устанавливает адрес DNS-сервера с помощью записей службы DNS. Например, DNS1=192.168.0.2 приводит к тому, что этот IP-адрес записывается в файл /etc/resolv.conf в качестве первого DNS-сервера системы. У вас может быть несколько строк DNS? (DNS2=, DNS3= т. д.).

Помимо настройки основных сетевых интерфейсов, вы можете в каталоге /etc/sysconfig/network-scripts создавать файлы, которые используются для установки псевдонимов (несколько IP-адресов для одного и того же интерфейса), связанных интерфейсов (несколько сетевых карт, прослушивающих один и тот же адрес) и пользовательских маршрутов. Мы рассмотрим эту тему позднее в данной главе.

Другие сетевые файлы

Помимо файлов сетевого интерфейса, существуют и другие файлы конфигурации сети, которые можно редактировать непосредственно для настройки сети Linux. Рассмотрим их подробнее.

Файл /etc/sysconfig/network. Общесистемные настройки, связанные с вашей локальной сетью, могут быть включены в файл `/etc/sysconfig/network`. В версиях системы до RHEL 6 в этом файле обычно устанавливалось имя хоста системы, но в него можно добавить и другие настройки. Пример содержимого файла `/etc/sysconfig/network`:

```
GATEWAY=192.168.0.1
```

Здесь параметр `GATEWAY` по умолчанию установлен как `192.168.0.1`. Разные интерфейсы могут использовать различные адреса `GATEWAY`. Чтобы узнать о других параметрах, которые могут появиться в сетевых файлах, изучите файл `sysconfig.txt` в каталоге `/usr/share/doc/initscripts`.

Файл /etc/hostname. В дистрибутивах RHEL и Fedora имя хоста системы хранится в файле `/etc/hostname`. Например, если файл содержит имя хоста `host1.example.com`, это имя будет устанавливаться каждый раз при загрузке системы. Чтобы проверить, какое имя хоста задано, введите команду `hostname`.

Файл /etc/hosts. До создания службы DNS преобразование имен хостов в IP-адреса осуществлялось передачей одного хост-файла. Пока в Интернете было всего несколько десятков, а затем и несколько сотен хостов, этот вариант довольно хорошо работал. Однако по мере роста сети Интернет одного хост-файла стало недостаточно, и была изобретена служба DNS.

Файл `/etc/hosts` все еще существует в системах Linux, и его все еще можно применять для сопоставления IP-адресов с именами хостов. Файл `/etc/hosts` — это способ настроить имена и адреса для небольшой локальной сети или просто создать псевдонимы, чтобы облегчить доступ к постоянно используемым системам.

Пример файла `/etc/hosts`:

```
127.0.0.1  localhost localhost.localdomain
::1        localhost localhost.localdomain
192.168.0.201  node1.example.com node1 joe
192.168.0.202  node2.example.com node2 sally
```

Первые две строки (`127.0.0.1` и `::1`) задают адреса для локальной системы. IPv4-адрес для локального хоста — это `127.0.0.1`, IPv6-адрес для локального хоста — `::1`. В примере также есть строки с двумя IP-адресами. Вы можете прийти к первому IP-адресу (192.168.0.201) с помощью команд `node1.example.com`, `node1` или `joe`. Например, введите `ping joe`, и тогда пинг-пакеты будут отправлены на адрес 192.168.0.201.

Файл /etc/resolv.conf. DNS-серверы и поисковые домены задаются в файле `/etc/resolv.conf`. Если утилита NetworkManager включена и запущена, не нужно редактировать этот файл напрямую. При использовании команды `DNS?=` из файлов `ifcfg -*` NetworkManager перезапишет файл `/etc/resolv.conf` и вы потеряете уже добавленные в него данные. Пример файла `/etc/resolv.conf`, который был изменен утилитой NetworkManager:

```
# Generated by NetworkManager
nameserver 192.168.0.2
nameserver 192.168.0.3
```

Каждая запись с директивой `nameserver` (имя сервера) идентифицирует IP-адрес DNS-сервера. Порядок записей определяет порядок проверки DNS-серверов. Если первая запись недоступна, вполне нормально, что отображаются две или три дополнительные. Более того, проверка недопустимого имени хоста для каждого сервера может занять слишком много времени.

Другой тип записи, которую можно добавить в этот файл, — *запись поиска*. Она позволяет указывать домены для поиска, когда имя хоста запрашивается по базовому имени, а не полному доменному имени. Записей поиска может быть несколько при идентификации одного или нескольких доменных имен после ключевого слова поиска, как в следующем примере:

```
search example.com example.org example.net
```

Параметры поиска разделены пробелами или отступами.

Файл /etc/nsswitch.conf. В отличие от предыдущих файлов, файлом `/etc/nsswitch.conf` управляет команда `authselect` и его нельзя изменять вручную. Чтобы внести изменения, отредактируйте файл `//etc/authselect/user-nsswitch.conf` и запустите команду `authselect apply-changes`.

Настройки в файле `/etc/nsswitch.conf` определяют, что преобразование имени хоста выполняется поиском сначала локального файла (файлов) `/etc/hosts`, а затем DNS-серверов, перечисленных в файле `/etc/resolv.conf` (`dns`). Значение переменной `myhostname` используется для того, чтобы гарантировать, что адрес всегда будет возвращаться хосту. Вот как выглядит запись `hosts` в файле `/etc/resolv.conf` в дистрибутиве Red Hat Enterprise Linux:

```
hosts:      files dns myhostname
```

Вы можете добавлять и другие местоположения, такие как базы данных Network Information Service (`nis` или `nisplus`), чтобы запросить перевод имени хоста в IP-адрес. Можно изменить порядок, в котором запрашиваются различные службы, а также проверить правильность преобразования имени хоста в IP-адрес с помощью различных команд.

Если вы хотите проверить, правильно ли запрашиваются ваши DNS-серверы, используйте команды `host` или `dig`, например:

```
$ host redhat.com
redhat.com has address 209.132.183.105
redhat.com mail is handled by 10 us-smtp-inbound-1.mimecast.com.
redhat.com mail is handled by 10 us-smtp-inbound-2.mimecast.com.
$ dig redhat.com
; <<>> DiG 9.11.11-RedHat-9.11.11-1.fc30 <<>> redhat.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 9948
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;redhat.com.            IN  A
...
```

```
;; ANSWER SECTION:
redhat.com.     3600   IN  A  09.132.183.105
;; Query time: 49 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Sat Nov 23 19:16:14 EST 2019
```

По умолчанию команда `host` выполняет более простой вывод для DNS-запросов. Она показывает IP-адрес для `redhat.com` и имена почтовых серверов (MX-записей), которые относятся к `redhat.com`. Команда `dig` отображает информацию, аналогичную той, что появляется в файлах, содержащих записи DNS. Часть вывода `QUESTION SECTION` указывает, что в разделе адреса запрашивается адрес `redhat.com` и в части вывода `ANSWER` был дан ответ (`209.132.183.105`). Вы также можете увидеть адрес запрошенного DNS-сервера.

Команды `host` и `dig` используются только для запроса DNS-серверов. Они не проверяют файл `nsswitch.conf`, чтобы найти другие запросы, например, в файле `hosts`. Для этого нужно взять команду `getent`:

```
# getent hosts node1
192.168.0.201  node1
```

Здесь команда `getent` находит хост с именем `node1`, который был добавлен в мой локальный файл `/etc/hosts`. Команду `getent` можно применять для запроса любой информации, настроенной в файле `nsswitch.conf`. Например, введите команду `getent passwd root`, чтобы увидеть учетную запись суперпользователя в локальном файле. Команда может также запросить удаленную базу данных LDAP, чтобы получить информацию о пользователе, если такая функция настроена. О ее настройке написано в главе 11 «Управление учетными записями».

Настройка псевдонимов сетевых интерфейсов

Рассмотрим случай, когда необходимо, чтобы ваша карта сетевого интерфейса прослушивала несколько IP-адресов. Например, если вы настраиваете веб-сервер, который обслуживает защищенные данные (https) для нескольких доменов (example.com, example.org и т. д.), то для каждого домена потребуется отдельный IP-адрес, связанный с отдельным сертификатом. В таком случае вместо того, чтобы добавлять дополнительные карты сетевого интерфейса, можно создать несколько псевдонимов на одной карте.

Чтобы получить псевдоним сетевого интерфейса в RHEL 6 и более ранних версиях системы Fedora, нужно создать еще один файл `ifcfg-`. Как и в интерфейсе `eth0` в системе RHEL, вы можете создать интерфейс `eth0:0`, связанный с той же картой сетевого интерфейса.

Для этого создайте в каталоге `/etc/sysconfig/network-scripts` файл с именем `ifcfg-eth0:0`, содержащий следующую информацию:

```
DEVICE=eth0:0
ONPARENT=yes
IPADDR=192.168.0.141
NETMASK=255.255.255.0
```

Здесь код создает псевдоним для сетевого интерфейса `eth0` и называет его `eth0:0`. Не параметр `ONBOOT`, а параметр `ONPARENT` приказывает вызвать этот интерфейс, если родительский процесс (`eth0`) запущен и прослушивает адрес 192.168.0.141.

Вы можете вызвать этот интерфейс, набрав команду `ifup eth0: 0`. Затем, чтобы проверить, что интерфейс появился, используйте команду `ip`:

```
$ ip addr show eth0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
        pfifo_fast state UP qlen 1000
    link/ether f0:de:f1:28:46:d9 brd ff:ff:ff:ff:ff:ffinet
    192.168.0.140/24 brd 192.168.0.255 scope global
    eth0inet 192.168.0.141/24 brd 192.168.0.255 scope global secondary
    eth0:0inet6 fe80::f2de:f1ff:fe28:46d9/64 scope link
        valid_lft forever preferred_lft forever
```

В выводе видно, что карта сетевого интерфейса `eth0` теперь прослушивает два адреса: 192.168.0.140 (`eth0`) и 192.168.0.141 (`eth0:0`). Таким образом, система будет реагировать на пакеты, предназначенные для любого из этих двух адресов. Вы можете добавить больше IP-адресов к этому интерфейсу, создав больше файлов `ifcfg-eth0:?` (`ifcfg-eth0:1`, `ifcfg-eth0:2` т. д.).

В более поздних системах RHEL и Fedora псевдонимы можно создавать непосредственно в основном файле `ifcfg`. Например, основной адрес (192.168.0.187) и адрес-псевдоним (192.168.99.1) для карты сетевого интерфейса с именем `p4p1` могут быть представлены следующими настройками в файле `ifcfg-p4p1`:

```
IPADDR=192.168.0.187
PREFIX=24
IPADDR1=192.168.99.1
PREFIX1=24
```

Объединение каналов Ethernet

Объединение каналов Ethernet позволяет иметь более одной сетевой интерфейсной карты на компьютере, связанном с одним IP-адресом. Есть несколько причин, почему эта функция может пригодиться.

- **Высокая доступность.** Несколько сетевых карт на одном и том же IP-адресе могут гарантировать, что если одна подсеть выйдет из строя или одна сетевая карта испортится, то доступ к адресу все еще будет возможен на другой сетевой карте, подключенной к другой подсети.
- **Производительность.** Если сетевого трафика слишком много, чтобы карта сетевого интерфейса смогла его обработать, трафик можно распределить по нескольким сетевым картам.

В системах Red Hat Enterprise Linux и Fedora на компьютере с несколькими сетевыми картами можно настроить объединение каналов Ethernet, создав несколько

файлов ifcfg и загрузив необходимый модуль. Вы можете начать с одного файла объединения (например, ifcfg-bond0), а затем добавить еще несколько файлов ifcfg-eth?. Затем загрузите модуль объединения.

В зависимости от типа объединения можете установить различные режимы своего интерфейса объединения. Используя переменную BONDING_OPTS, выберите режим и другие параметры объединения (все они передаются в модуль объединения). Можете прочитать о модуле объединения, введя команду modinfo bonding или установив пакет kernel-doc и изучив файл bonding.txt из каталога /usr/share/doc/kernel-doc*/Documentation/networking/.

Далее приведен пример файла, который определяет интерфейс объединения. Файл здесь — это etc/sysconfig/network-scripts/ifcfg-bond0:

```
DEVICE=bond0
ONBOOT=yes
IPADDR=192.168.0.50
NETMASK=255.255.255.0
BOOTPROTO=none
BONDING_OPTS="mode=active-backup"
```

Интерфейс bond0 в этом примере использует IP-адрес 192.168.0.50. Запускается при загрузке системы. Параметр Bonding_opts устанавливает режим активного резервного копирования в файл объединения. Это означает, что одновременно активна только одна карта сетевого интерфейса, а следующая карта вступает в действие только при сбое предыдущей (реализована функция отказоустойчивости). С интерфейсом bond0 еще не связана ни одна карта сетевого интерфейса. Для этого необходимо добавить отдельные параметры для файла ifcfg. Например, создайте файл /etc/sysconfig/network-scripts/ifcfg-eth0, который выглядит как в примере далее, а затем — eth1, eth2, eth3 и т. д. для карты сетевого интерфейса, которую нужно добавить в интерфейс объединения:

```
DEVICE=eth0
MASTER=bond0
SLAVE=yes
BOOTPROTO=none
ONBOOT=yes
```

При использовании интерфейса eth0 в качестве части интерфейса bond0 IP-адрес не назначается. Это происходит потому, что интерфейс eth0 применяет IP-адрес из интерфейса bond0, определяя себя подчиненным (SLAVE=yes) для bond0 (MASTER=bond0).

В конце нужно убедиться, что интерфейс bond0 задействует модель объединения. Для этого создайте файл /etc/modprobe.d/bonding.conf, который содержит следующую строку:

```
alias bond0 bonding
```

Поскольку все интерфейсы настроены на параметр ONBOOT=yes, запускается интерфейс bond0 и все интерфейсы становятся eth?-доступными по мере необходимости.

Настройка пользовательских маршрутов

При простой настройке сети сообщения, предназначенные для локальной сети, направляются на соответствующий интерфейс в ней, а сообщения для хостов вне локальной сети — на шлюз по умолчанию для отправки на удаленные хосты. В качестве альтернативы можно настроить пользовательские маршруты, чтобы обеспечить дополнительные пути к определенным сетям.

Чтобы задать пользовательский маршрут в системах Fedora и RHEL, нужно создать файл конфигурации в каталоге `/etc/sysconfig/network-scripts`. Для маршрута необходимо определить:

- переменную шлюза `GATEWAY?`. Установите IP-адрес узла в локальной сети, который обеспечивает маршрут к подсети, представленной статическим маршрутом;
- переменную адресов `ADDRESS?`. Установите IP-адрес, представляющий сеть, до которой можно добраться по статическому маршруту;
- переменную маски сети `NETMASK?`. Установите маску сети, которая определяет, какая часть переменной `ADDRESS?` является сетью, и представляет хосты в ней.

Имя файла каждого пользовательского маршрута начинается с `route-interface`. Например, маршрут через интерфейс `eth0` будет называться `route-eth0`. У вас может быть несколько пользовательских маршрутов в этом файле, причем каждая запись маршрута заменит вопросительный знак (`?`) номером интерфейса:

```
ADDRESS0=192.168.99.0
NETMASK0=255.255.255.0
GATEWAY0=192.168.0.5
```

В примере любой пакет, предназначенный для хоста в сети 192.168.99, будет отправлен через локальный интерфейс `eth0` и направлен на узел шлюза 192.168.0.5. Предположительно, этот узел обеспечит маршрут к другой сети, содержащей хосты в диапазоне адресов 192.168.99. Данный маршрут вступит в силу после перезапуска сетевого интерфейса `eth0`. Чтобы проверить, работает ли маршрут после перезагрузки сетевого интерфейса, введите следующее:

```
# route
Kernel IP routing table
Destination   Gateway      Genmask         Flags  Metric  Ref  Use  Iface
default       192.168.0.1  0.0.0.0         UG     0       0      0  eth0
192.168.0.0   *            255.255.255.0   U      1       0      0  eth0
192.168.99.0  192.168.0.5  255.255.255.0   UG     0       0      0  eth0
```

Выходные данные команды `route -n` показывают, что маршрут по умолчанию (все, что не предназначено для локальной сети 192.168.0 или сети 192.168.99) проходит через адрес 192.168.0.1. Любые пакеты, предназначенные для сети 192.168.99, направляются по адресу 192.168.0.5.

Если нужно создать больше пользовательских маршрутов, добавьте их в этот же файл `route-eth0`. Для следующих маршрутов наборы информации будут называться `ADDRESS1`, `NETMASK1`, `GATEWAY1` и т. д.

Настройка сети на предприятии

До сих пор мы рассматривали настройку сети для отдельных систем. Функции, доступные в Linux, выходят за рамки этого процесса, предоставляя программное обеспечение, способное поддерживать полноценную сетевую инфраструктуру, необходимую хост-компьютерам для связи.

В следующих разделах разберем некоторые типы служб сетевой инфраструктуры, доступные в системах Linux. Проверка работы этих функций выходит за рамки книги, но, если вам понадобится управлять функциями сетевой инфраструктуры, в следующих разделах дано представление о том, как они реализуются в Linux.

Настройка Linux в качестве маршрутизатора

Если у вас на компьютере есть несколько сетевых интерфейсов (две и более карты сетевого интерфейса), можете настроить Linux в качестве маршрутизатора. Для этого нужно лишь изменить один параметр ядра, который позволяет пересылать пакеты. Чтобы временно включить параметр `ip_forward`, от имени суперпользователя введите следующее:

```
# cat /proc/sys/net/ipv4/ip_forward
0
# echo 1 > /proc/sys/net/ipv4/ip_forward
# cat /proc/sys/net/ipv4/ip_forward
1
```

Переадресация пакетов (маршрутизация) по умолчанию отключена, а значение `ip_forward` равно `0`. При значении `1` переадресация пакетов сразу же будет включена. Чтобы сделать ее постоянной, необходимо добавить это значение в файл `/etc/sysctl.conf` следующим образом:

```
net.ipv4.ip_forward = 1
```

Если изменить этот файл так, как показано здесь, то каждый раз при перезагрузке системы значение `ip_forward` будет сбрасываться к `1`. (Обратите внимание на то, что `net.ipv4.ip_forward` отражает фактическое местоположение файла `ip_forward` без `/proc/sys` и с точками, которые заменяют косую черту. Таким образом вы можете изменить любые параметры ядра, заданные в структуре каталогов `/proc/sys`.)

В качестве маршрутизатора система Linux часто используется как брандмауэр между частной сетью и публичной сетью, например Интернетом. В таком случае можно задействовать эту же систему в качестве брандмауэра, который

преобразует сетевые адреса (NAT) и предоставляет службу DHCP, чтобы системы в частной сети могли маршрутизироваться через систему Linux с помощью частных IP-адресов. (См. главу 25 «Защита Linux в сети», чтобы узнать о работе с правилами брандмауэра Linux с помощью функции `iptables`.)

Настройка Linux в качестве DHCP-сервера

Система Linux не только способна использовать DHCP-сервер для получения своего IP-адреса и другой информации, но и может быть настроена на работу в качестве DHCP-сервера. В своей самой простой форме DHCP-сервер может выдавать IP-адреса из пула адресов любой системе, которая запрашивает IP-адрес. Однако обычно DHCP-сервер также распределяет местоположения DNS-серверов и шлюза по умолчанию.

Настройка DHCP-сервера не так проста и требует предварительной подготовки. Не добавляйте DHCP-сервер в сеть, которую вы не контролируете и в которой уже есть работающий DHCP-сервер. Многие клиенты получают информацию об адресах от любого DHCP-сервера, который ее раздает.

Служба DHCP предоставляется пакетом `dhcp-server` в системах Fedora и RHEL. Сама служба называется `dhcpd`. Основным файлом конфигурации является файл `/etc/dhcp/dhcpd.conf` для сетей IPv4 (в том же каталоге находится файл `dhcpd6.conf` для сетей IPv6). По умолчанию демон `dhcpd` прослушивает UDP-порт 67, поэтому его нужно оставить открытым на локальном брандмауэре.

Чтобы настроить DHCP-сервер, можно скопировать файл `dhcpd.conf.example` из каталога `/usr/share/doc/dhcp-server`, заменить им файл `/etc/dhcp/dhcpd.conf`, а затем изменить его по своему усмотрению. Однако перед применением этого файла нужно изменить параметры доменного имени, чтобы домен и диапазоны IP-адресов соответствовали тем, которые вы используете. Комментарии в файле помогут сделать это.

При установке некоторых виртуальных и облачных служб в системе Linux DHCP-сервер настраивается по умолчанию под пользователя. Например, когда вы устанавливаете программу KVM и запускаете службу `libvirtd` в системе RHEL или Fedora, она автоматически настраивает частную сеть по умолчанию в диапазоне адресов 192.168.122.0/24. При запуске виртуальных машин им присваиваются IP-адреса из данного диапазона. После установки и запуска службы Docker в этих дистрибутивах она настраивает частную сеть и раздает IP-адреса контейнерам Docker, запущенным в этой системе.

Настройка Linux в качестве DNS-сервера

В Linux большинство профессиональных серверов системы доменных имен (DNS) реализовано с помощью службы Berkeley Internet Name Domain (BIND). В системах Fedora и RHEL для этого необходимо установить пакеты `bind`, `bind-utils` и `bind-libs`. Для дополнительной безопасности можно установить пакет `bind-chroot`.

По умолчанию пакет `bind` настраивается путем редактирования файла `/etc/named.conf`. Сопоставление имени хоста с IP-адресом выполняется в файлах, расположенных в каталоге `/var/named`. Если установить пакет `bind-chroot`, файлы конфигурации `bindc` перемещаются в каталог `/var/named/chroot`, который попытается скопировать из каталогов `/etc` и `/var` файлы, необходимые для настройки `bind`, чтобы именованный демон (который предоставляет службу) был подключен к структуре каталогов `/etc/named/chroot`.

Если вы хотите опробовать работу пакета `bind`, я рекомендую начать с настройки DNS-сервера для небольшой домашней сети, не выходя за границы брандмауэра. Вы можете заблокировать IP-адреса компьютеров в своем доме, подключив MAC-адреса карты сетевого интерфейса каждого компьютера к определенным IP-адресам на DHCP-сервере, а затем сопоставив эти имена с адресами на DNS-сервере.

ВНИМАНИЕ

Имейте в виду, что прежде, чем создавать общедоступный DNS-сервер, необходимо правильно защитить свой DNS-сервер. Публичный DNS-сервер можно взломать и перенаправлять с него трафик на любой другой сервер. Таким образом, при использовании такого сервера вы рискуете нарваться на нежелательные и опасные для системы сайты.

Настройка Linux в качестве прокси-сервера

Прокси-сервер позволяет ограничивать сетевой трафик из частной сети в общедоступную, например в Интернет. Такие серверы позволяют ограничивать доступ к определенным сайтам, что полезно, например, на школьных компьютерах или компьютерах сотрудников предприятия.

Настройка Linux в качестве маршрутизатора и прокси-сервера в домашней сети позволит контролировать доступ в Интернет только по определенным протоколам и лишь после фильтрации трафика.

Используя прокси-сервер Squid, который поставляется с большинством систем Linux (пакет `squid` в Fedora и RHEL), вы можете включить систему для приема запросов к веб-серверам (HTTP и HTTPS), файловым серверам (FTP) и другим протоколам. Можете определить, какие системы будут задействовать ваш прокси-сервер (ограничив по имени хоста или IP-адресу) и даже какие сайты они могут посещать (задав определенный адрес, диапазон адресов, имя хоста или доменные имена).

Настройка прокси-сервера `squid` так же проста, как установка пакета `squid`, редактирование файла `/etc/squid/squid.conf` и запуск службы `squid`. Файл поставляется с минимальными рекомендуемыми настройками. Однако вы можете определить хосты (на основе IP-адреса или имени), которым нужно разрешить использовать эту службу. Существуют черные списки (находятся в службе `squid`), которые позволяют запретить, например, детям доступ к целым наборам сайтов.

Резюме

Большинство сетевых подключений с настольного компьютера Linux или ноутбука реализуются практически без вмешательства пользователя. Если вы задействуете утилиты NetworkManager по проводному или беспроводному Ethernet-соединению, адрес и информация о сервере, необходимые для запуска, автоматически поступают с DHCP-сервера.

С помощью графического интерфейса NetworkManager можно настроить сеть под свои нужды. Вы можете установить статические IP-адреса и использовать компьютеры сервера имен и шлюза. Чтобы выполнить вручную сложную настройку сети, перейдите к непосредственной работе с файлами конфигурации сети.

Файлы конфигурации сети в Linux можно применять для настройки более продвинутых функций, таких как объединение каналов Ethernet.

Помимо основных настроек сетевого подключения, в Linux доступны функции, которые позволяют предоставлять типы служб для сетевой инфраструктуры. В этой главе говорится о службах и функциях, таких как маршрутизация, DHCP- и DNS-серверы. Их необходимо знать, чтобы работать с продвинутыми сетевыми функциями в Linux.

Настроив свою сеть, можно перейти к настройке служб для работы по сети. В главе 15 «Запуск и остановка служб» описываются инструменты, необходимые для включения, отключения, запуска, остановки и проверки состояния служб в системе Linux.

Упражнения

Упражнения этого раздела позволяют изменить сетевые интерфейсы в вашей системе Linux, а также понять, как настроить более продвинутые сетевые функции. Выполните их в системе Linux, у которой есть активное сетевое соединение и все в порядке с сетевой активностью.

Советую выполнять эти упражнения непосредственно из консоли компьютера (другими словами, не подключаясь к компьютеру через службу `ssh`). Некоторые из выполняемых вами команд могут прервать подключение к сети, а ошибки в некоторых настройках могут привести к тому, что компьютер будет временно недоступен для сети.

Как всегда, Linux позволяет решать задачи несколькими способами и не только так, как указано в ответах. Если затрудняетесь с решением заданий, воспользуйтесь ответами к упражнениям, приведенными в приложении Б.

1. С рабочего стола проверьте, успешно ли утилита NetworkManager запустила сетевой интерфейс (проводной или беспроводной) в сеть. Если нет, то попробуйте запустить свой сетевой интерфейс.
2. Выполните команду для проверки активных сетевых интерфейсов, доступных на вашем компьютере.

3. Попробуйте связаться с сайтом google.com из командной строки таким образом, чтобы служба DNS работала правильно.
4. Выполните команду для проверки маршрутов, используемых для связи за пределами вашей локальной сети.
5. Проследите маршрут, по которому осуществляется соединение с сайтом google.com.
6. Просмотрите сетевую активность своей системы Linux из веб-интерфейса Cockpit.
7. Создайте запись хоста, которая позволит связаться с вашей локальной хост-системой с помощью имени `myownhost`.
8. Определите адреса DNS-серверов, которые используются для преобразования имен хостов в IP-адреса в системе, а затем проверьте, какие из них запрашиваются из вашей системы, чтобы найти нужный IP-адрес для соединения с сайтом google.com.
9. Создайте пользовательский маршрут, который направляет трафик, предназначенный для сети 192.168.99.0/255.255.255.0, на IP-адрес в вашей локальной сети, например на 192.168.0.5 (сначала убедитесь, что сеть 192.168.99 не задействуется).
10. Проверьте, настроена ли в вашей системе маршрутизация пакетов IPv4 между сетевыми интерфейсами.

15 Запуск и остановка служб

В этой главе

- Службы инициализации Linux.
- Проверка служб, управляемых демонами Linux.
- Остановка и запуск служб.
- Изменение уровня выполнения сервера Linux.
- Удаление служб.

Основная задача серверной системы Linux — предоставлять службы локальным или удаленным пользователям. Сервер может давать доступ к веб-страницам, файлам, информации баз данных, потоковой музыке или другим типам данных. Серверы имен могут предоставлять доступ к спискам хост-компьютеров или именам пользователей. Сотни этих и других типов служб можно настроить в системах Linux.

Локальные службы в системах Linux, такие как доступ к принтеру или вход в систему, обычно реализуются с помощью *демон*-процессов. Большинство систем Linux позволяют управлять каждым демоном как *службой* с помощью одной из нескольких популярных систем инициализации, также называемых init-системами. Преимущества использования системы инициализации позволяют следующее.

- **Определить уровень выполнения.** Объедините наборы служб в так называемые *целевые* уровни выполнения.
- **Установить зависимости.** Установите зависимости служб таким образом, чтобы, например, служба, требующая сетевых интерфейсов, не запускалась до тех пор, пока все службы запуска сети не будут успешно активизированы.
- **Установить уровень выполнения по умолчанию.** Выберите, какой уровень выполнения будет запускаться при загрузке системы *(по умолчанию)*.

- **Управлять службами.** Запускайте команды, которые указывают отдельным службам запускать, останавливать, приостанавливать, перезапускать или даже перезагружать файлы конфигурации.

Сейчас в системах Linux применяются несколько различных систем инициализации. Какая из них используется, зависит от дистрибутива Linux и его версии. В этой главе я расскажу о системах инициализации, которые работали в Fedora, Red Hat Enterprise Linux, Ubuntu и многих других дистрибутивах Linux.

- **SysVinit.** Эта система инициализации была создана для систем UNIX System V в начале 1980-х годов. Она использует простой для понимания метод запуска и остановки служб на основе уровня выполнения. Еще несколько лет назад большинство систем UNIX и Linux работали с SysVinit.
- **Systemd.** Последние версии Fedora и RHEL применяют систему инициализации `systemd`. Это и самая сложная, и наиболее гибкая по конфигурации из init-систем. `Systemd` не только предлагает функции для запуска и работы со службами, но и позволяет управлять сокетами, устройствами, точками монтирования, областями подкачки и другими типами устройств.

ПРИМЕЧАНИЕ

В более старой версии дистрибутива Ubuntu в качестве системы инициализации использовалась система Upstart. Начиная с версии Ubuntu 15.04 (выпущена 28 апреля 2015 года), она была заменена демоном инициализации системы. Поэтому в данной книге мы не будем рассматривать систему Upstart.

В этой главе описываются системы инициализации SysVinit и `systemd`. В процессе использования init-системы, соответствующей вашему дистрибутиву Linux, вы узнаете, как идет процесс загрузки для запуска служб, как запускать и останавливать службы по отдельности, а также как включать и отключать службы.

Демон инициализации (init или systemd)

Перед тем как перейти к управлению службами, необходимо понять, как работает демон инициализации. Демон инициализации можно рассматривать как «отца всех процессов». Он является первым процессом, который запускается ядром на сервере Linux. Демон инициализации для дистрибутивов Linux, использующих SysVinit, так и называется — `init`. В системе `systemd` демон инициализации называется `systemd`.

Ядро Linux имеет идентификатор процесса (PID) 0. Таким образом, демон процесса инициализации (`init` или `systemd`) имеет PPID (идентификатор родительского процесса) 0 и PID 1. После запуска `init` отвечает за запуск процессов во время загрузки сервера, таких как оболочка входа в систему (процессы `getty` или `mingetty`). Он также отвечает за управление службами.

Демон `init` в Linux был основан на демоне `init` из UNIX System V. Именно поэтому он называется демоном SysVinit. Однако это не единственный классический

init-демон. Демон init не является частью ядра Linux, поэтому он может быть абсолютно разным, и дистрибутивы Linux выбирают, какой тип использовать. Второй классический init-демон был основан в системе Berkeley UNIX (BSD).

Классические демоны init работали без проблем в течение многих лет. Однако они были созданы для работы в статической среде. По мере появления нового оборудования, например USB-устройств, у классических демонов init возникли проблемы в работе со съемными устройствами. Компьютерное оборудование вместо статического стало событийным. Для работы с этими средами требовались новые демоны init. Кроме того, по мере появления новых служб классическим демонам init приходилось запускать все больше и больше дополнительных служб. В итоге процесс инициализации системы стал менее эффективным и в конечном счете сильно замедлился.

Современные демоны инициализации решают подобные проблемы неэффективной загрузки системы и нестатических сред. Самый популярный из новых демонов инициализации — systemd. Дистрибутивы Ubuntu, RHEL и Fedora уже перешли на демон systemd, сохраняя обратную совместимость с классическими пакетами SysVinit, Upstart или BSD.

Демон systemd был написан Леннартом Пёттерингом, разработчиком Red Hat. Описание демона доступно по адресу docs.fedoraproject.org/en-US/quick-docs/understanding-and-administering-systemd. Сейчас он используется всеми последними версиями Fedora, RHEL, OpenSUSE и Ubuntu.

Чтобы правильно управлять своими службами, необходимо знать, какой демон инициализации применяется на сервере. Выяснить это может быть довольно проблематично. Процесс инициализации SysVinit или Upstart называется одинаково — init. Демон для первых систем инициализации systemd также назывался init, но в последних версиях переименован в systemd. Запустите команду ps -e, чтобы узнать, использует ли ваша система демон systemd:

```
# ps -e | head
  PID TTY          TIME CMD
    1 ?        00:04:36 systemd
    2 ?        00:00:03 kthreadd
    3 ?        00:00:15 ksoftirqd/0
```

Если в вашей системе PID 1 является демоном init, загляните на страницу об init в «Википедии» (wikipedia.org/wiki/Init), в раздел Other implementations. Это поможет понять, является ли ваш init-демон SysVinit, Upstart или какой-либо другой системой инициализации.

Классические демоны инициализации

Классические демоны инициализации SysVinit и BSD заслуживают внимания, даже если ваш Linux-сервер задействует другой демон init. Новые демоны init не только используют обратную совместимость с классическими — многие даже основаны на них.

Классические демоны `init` SysVinit и BSD работают схожим образом. Изначально они сильно отличались друг от друга, но со временем существенных различий осталось очень мало. К примеру, более старый демон BSD `init` раньше получал информацию о конфигурации из файла `/etc/ttytab`. Теперь, как и SysVinit, демон BSD `init` берет информацию о конфигурации во время загрузки из файла `/etc/inittab`. Далее приведен пример классического файла SysVinit `/etc/inittab`:

```
# cat /etc/inittab
# inittab This file describes how the INIT process should set up
# Default runlevel. The runlevels used by RHS are:
#   0 – halt (Do NOT set initdefault to this)
#   1 – Single user mode
#   2 – Multiuser, no NFS (Same as 3, if you do not have networking)
#   3 – Full multiuser mode
#   4 – unused
#   5 – X11
#   6 – reboot (Do NOT set initdefault to this)
#
id:5:initdefault:

# System initialization.
si::sysinit:/etc/rc.d/rc.sysinit

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

# Trap CTRL-ALT-DELETE
ca::ctrlaltdel:/sbin/shutdown -t3 -r now
pf::powerfail:/sbin/shutdown -f -h +2
"Power Failure; System Shutting Down"

# If power was restored before the shutdown kicked in, cancel it.
pr:12345:powerokwait:/sbin/shutdown -c
"Power Restored; Shutdown Cancelled"

# Run gettys in standard runlevels
1:2345:respawn:/sbin/mingetty tty1
2:2345:respawn:/sbin/mingetty tty2
3:2345:respawn:/sbin/mingetty tty3
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6

# Run xdm in runlevel 5
x:5:respawn:/etc/X11/prefdm -nodaemon
```

Файл `/etc/inittab` сообщает демону `init`, какой уровень выполнения является уровнем по умолчанию.

Уровень выполнения — это номер классификации, который определяет, какие службы запускаются, а какие останавливаются. В предыдущем примере уровень выполнения по умолчанию, 5, устанавливается с идентификатором строки: `5:initdefault:`. В табл. 15.1 показаны семь стандартных уровней запуска Linux.

Таблица 15.1. Стандартные уровни выполнения Linux

Уровень выполнения	Название	Описание
0	Завершение работы компьютера	Все службы отключены, сервер остановлен
1 или S	Однопользовательский режим	Учетная запись суперпользователя автоматически регистрируется на сервере. Другие пользователи не могут войти на сервер. Доступен только интерфейс командной строки. Сетевые службы не запускаются
2	Многопользовательский режим без поддержки сети	Пользователи могут войти на сервер, но им доступен только интерфейс командной строки. В некоторых системах сетевые интерфейсы и службы запускаются, в других — нет. Первоначально этот уровень выполнения применялся для запуска неинтеллектуальных терминалов, чтобы пользователи могли войти в систему (но сетевые службы не запускались)
3	Многопользовательский режим без поддержки сети	Пользователи могут войти на сервер, но им доступен только интерфейс командной строки. Запускаются сетевые интерфейсы и службы. Это общий уровень выполнения для серверов
4	Пользовательский уровень	Пользователи могут настроить этот уровень выполнения самостоятельно
5	Многопользовательский режим с поддержкой сети и графической оболочки	Пользователи могут войти на сервер. Доступны командная строка и графический интерфейс. Запускаются сетевые службы. Это общий уровень выполнения для настольных систем
6	Перезагрузка компьютера	Сервер перезагружается

Дистрибутивы Linux могут немного различаться в определении каждого уровня выполнения, а также в том, какие уровни применяются.

ВНИМАНИЕ!

Единственные уровни выполнения, которые могут использоваться в файле /etc/inittab, — это уровни от 2 до 5. Другие уровни могут вызвать проблемы. Например, если поместить уровень 6 в файл /etc/inittab по умолчанию, то при перезагрузке сервера он войдет в цикл и будет перезагружаться снова и снова.

Уровни выполнения используются не только в качестве уровня по умолчанию в файле /etc/inittab. Их можно вызвать и с помощью самого демона init. Поэтому, если нужно немедленно остановить свой сервер, введите команду init в командной строке:

```
# init 0
...
System going down for system halt NOW!
```

Команда init принимает любые уровни выполнения из табл. 15.1, что позволяет быстро переключать сервер из одной категории уровней в другую. Например, если нужно устранить неполадки, которые требуют отключения графического интерфейса, введите команду init 3 в командной строке:

```
# init 3
INIT: Sending processes the TERM signal
starting irqbalance:                       [ OK ]
Starting setroubleshootd:
Starting fuse: Fuse filesystem already available.
...
Starting console mouse services:           [ OK ]
```

Чтобы увидеть текущий уровень выполнения своего Linux-сервера, просто введите команду runlevel. Первый элемент вывода — это предыдущий уровень выполнения сервера, который в примере далее равен 5. Второй элемент вывода отображает текущий уровень выполнения сервера, который в примере равен 3:

```
$ runlevel
5 3
```

Кроме команды init, можно применять команду telinit, которая работает точно так же. В следующем примере команда telinit используется для перезагрузки сервера, переводя его на уровень выполнение 6:

```
# telinit 6
INIT: Sending processes the TERM signal
Shutting down smartd:                      [ OK ]
Shutting down Avahi daemon:                [ OK ]
Stopping dhcdbd:                           [ OK ]
Stopping HAL daemon:                       [ OK ]
...
Starting killall:
Sending all processes the TERM signal...   [ OK ]
Sending all processes the KILL signal...   [ OK ]
...
Unmounting filesystems                     [ OK ]
Please stand by while rebooting the system
...
```

На только что запущенном сервере Linux текущий номер уровня выполнения должен совпадать с номером уровня по умолчанию в файле /etc/inittab. Однако обратите внимание на то, что предыдущий уровень выполнения в следующем

примере равен N. Это значение (Nonexistent) говорит, что сервер был недавно загружен на текущий уровень выполнения:

```
$ runlevel
N 5
```

Как же сервер узнает, какие службы остановить и какие запустить, когда выбран определенный уровень выполнения? При выборе уровня выполнения запускаются скрипты, расположенные в каталоге /etc/rc.d/rc#.d, где # — выбранный уровень выполнения. Эти скрипты запускаются независимо от того, выбран ли уровень выполнения через загрузку сервера и используется ли параметр /etc/inittab initdefault или команда init либо telinit. Например, если выбран уровень выполнения 5, то все скрипты в каталоге /etc/rc.d/rc5.d будут запущены. Ваш список будет отличаться от приведенного далее в зависимости от того, какие службы вы установили и включили:

```
# ls /etc/rc.d/rc5.d
K01smolt                      K88wpa_supplicant   S22messagebus
K02avahi-dnsconfd             K89dund             S25bluetooth
K02NetworkManager             K89netplugd         S25fuse
K02NetworkManagerDispatcher   K89pand             S25netfs
K05saslauthd                  K89rdisc            S25pcscd
K10dc_server                  K91capi             S26hidd
K10psacct                     S00microcode_ctl    S26udev-post
K12dc_client                  S04readahead_early  S28autofs
K15gpm                        S05kudzu            S50hplip
K15httpd                      S06cpuspeed         S55cups
K20nfs                        S08ip6tables        S55sshd
K24irda                       S08iptables         S80sendmail
K25squid                      S09isdn             S90ConsoleKit
K30spamassassin               S10network          S90crond
K35vncserver                  S11auditd           S90xfs
K50netconsole                 S12restorecond      S95anacron
K50tux                        S12syslog           S95atd
K69rpcsvcgssd                 S13irqbalance       S96readahead_later
K73winbind                    S13mcstrans         S97dhcdbd
K73ypbind                     S13rpcbind          S97yum-updatesd
K74nscd                       S13setroubleshoot   S98avahi-daemon
K74ntpd                       S14nfslock          S98haldaemon
K84btseed                     S15mdmonitor        S99firstboot
K84bttrack                    S18rpcidmapd        S99local
K87multipathd                 S19rpcgssd          S99smartd
```

Обратите внимание на то, что некоторые скрипты в каталоге /etc/rc.d/rc5.d начинаются с буквы K, а некоторые — с S. Значение K относится к скрипту, который немедленно останавливает процесс. Значение S относится к скрипту, который запускает процесс. Кроме того, каждый K- и S-скрипт имеет номер службы или демона, которыми они управляют. Это позволяет останавливать или запускать службы в определенном порядке. Например, благодаря этому сетевые службы Linux-сервера не будут запускаться до запуска самой сети.

Каталог /etc/rc.d/rc#.d существует для всех стандартных уровней запуска Linux. Каждый из каталогов содержит скрипты для запуска и остановки служб своего конкретного уровня выполнения:

```
# ls -d /etc/rc.d/rc?.d
/etc/rc.d/rc0.d  /etc/rc.d/rc2.d  /etc/rc.d/rc4.d  /etc/rc.d/rc6.d
/etc/rc.d/rc1.d  /etc/rc.d/rc3.d  /etc/rc.d/rc5.d
```

На самом деле файлы в каталогах /etc/rc.d/rc#.d являются не скриптами, а символическими ссылками на скрипты в каталоге /etc/rc.d/init.d. Таким образом, нет необходимости в копировании определенных скриптов.

```
# ls -l /etc/rc.d/rc5.d/K15httpd
lrwxrwxrwx 1 root root 15 Oct 10 08:15
 /etc/rc.d/rc5.d/K15httpd -> ../init.d/httpd
# ls /etc/rc.d/init.d
anacron              functions   multipathd                rpcidmapd
atd                  fuse        netconsole                rpcsvcgssd
auditd               gpm         netfs                     saslauthd
autofs               haldaemon   netplugd                  sendmail
avahi-daemon         halt        network                   setroubleshoot
avahi-dnsconfd       hidd        NetworkManager            single
bluetooth            hplip       NetworkManagerDispatcher  smartd
btseed               hsqldb      nfs                       smolt
bttrack              httpd       nfslock                   spamassassin
capi                 ip6tables   nscd                      squid
ConsoleKit           iptables    ntpd                      sshd
cpuspeed             irda        pand                      syslog
crond                irqbalance  pcscd                     tux
cups                 isdn        psacct                    udev-post
cups-config-daemon   killall     rdisc                     vncserver
dc_client            kudzu       readahead_early           winbind
dc_server            mcstrans    readahead_later           wpa_supplicant
dhcdbd               mdmonitor   restorecond               xfs
dund                 messagebus  rpcbind                   ypbind
firstboot            microcode   rpcgssd                   yum-updatesd
```

Обратите внимание на то, что каждая служба имеет один скрипт в файле /etc/rc.d/init.d. Не существует отдельных скриптов для остановки и запуска службы. Эти скрипты останавливают или запускают службу в зависимости от того, какой параметр передается им демоном init.

Каждый скрипт в файле /etc/rc.d/init.d делает все, что необходимо для запуска или остановки конкретной службы на сервере. Далее приведена часть скрипта httpd в системе Linux, использующей демон SysVinit. Пример содержит оператор для обработки переданного ему параметра ($1), к примеру start, stop, status т. д.:

```
# cat /etc/rc.d/init.d/httpd
#!/bin/bash
#
# httpd         Startup script for the Apache HTTP Server
#
# chkconfig: -  85 15
```

```
# description:  Apache is a World Wide Web server.
#               It is used to serve \
#               HTML files and CGI.
# processname: httpd
# config: /etc/httpd/conf/httpd.conf
# config: /etc/sysconfig/httpd
# pidfile: /var/run/httpd.pid

# Source function library.
. /etc/rc.d/init.d/functions
...
# See how we were called.
case "$1" in
  start)
        start
        ;;
  stop)
        stop
        ;;
  status)
        status $httpd
        RETVAL=$?
        ;;
...
esac

exit $RETVAL
```

После отработки скриптов уровня выполнения из соответствующего каталога `/etc/rc.d/rc#.d` процесс запуска демона SysVinit завершается. Последнее, что делает процесс `init` на этом этапе, — выполняет какие-либо дополнительные указания из файла `/etc/inittab` (например, создать процессы `mingetty` для виртуальных консолей и запустить интерфейс рабочего стола, если вы находитесь на уровне запуска 5).

Система инициализации systemd

Демон инициализации `systemd` — это замена демонам SysVinit и Upstart. Этот современный демон инициализации был добавлен в дистрибутивы Fedora 15 и RHEL 7 и используется до сих пор. Он имеет обратную совместимость с SysVinit и Upstart. Время инициализации системы демоном `systemd` сокращается, поскольку он может запускать службы параллельно.

Основы работы демона systemd

С помощью демона SysVinit службы останавливаются и запускаются на основе уровней выполнения. Служба `systemd` занимается уровнями выполнения, но реализует их по-другому — с помощью так называемых *юнитов целей* (target

units). Хотя основная задача функции `systemd` заключается в запуске и остановке служб, она может управлять также другими типами объектов, называемых юнитами. *Юнит* — это группа, состоящая из имени, типа и файла конфигурации и реализующая определенную службу или действие. Существует 12 типов юнитов `systemd`:

- `automount`;
- `device`;
- `mount`;
- `path`;
- `service`;
- `snapshot`;
- `socket`;
- `target`;
- `timer`;
- `wap`;
- `slice`;
- `cope`.

Два основных юнита `systemd`, с которыми вы будете иметь дело в ходе работы со службами, — это сервисные и целевые юниты. *Сервисные юниты* используются для управления демонами на сервере Linux. *Целевой юнит* — это просто группа других юнитов.

В следующем примере показаны несколько сервисных и целевых юнитов `systemd`. Имена сервисных юнитов похожи на имена демонов, например `cups` и `sshd`. Обратите внимание на то, что каждое имя сервисного юнита заканчивается на `.service`. Целевые юниты имеют такие имена, как `sysinit` (`sysinit` используется для запуска служб при инициализации системы). Имена целевых юнитов заканчиваются на `.target`:

```
# systemctl list-units | grep .service
...
cups.service           loaded active running CUPS Printing Service
dbus.service           loaded active running D-Bus Message Bus
...
NetworkManager.service loaded active running Network Manager
prefdm.service         loaded active running Display Manager
remount-rootfs.service loaded active exited  Remount Root FS
rsyslog.service        loaded active running System Logging
...
sshd.service           loaded active running OpenSSH server daemon
systemd-logind.service loaded active running Login Service
...
# systemctl list-units | grep .target
basic.target           loaded active active  Basic System
```

```
cryptsetup.target      loaded active active  Encrypted Volumes
getty.target           loaded active active  Login Prompts
graphical.target       loaded active active  Graphical Interface
local-fs-pre.target    loaded active active  Local File Systems (Pre)
local-fs.target        loaded active active  Local File Systems
multi-user.target      loaded active active  Multi-User
network.target         loaded active active  Network
remote-fs.target       loaded active active  Remote File Systems
sockets.target         loaded active active  Sockets
sound.target           loaded active active  Sound Card
swap.target            loaded active active  Swap
sysinit.target         loaded active active  System Initialization
syslog.target          loaded active active  Syslog
```

Файлы конфигурации системы Linux находятся в каталогах `/lib/systemd/system` и `/etc/systemd/system`. Вы можете использовать команду `ls` для просмотра этих каталогов, но предпочтительнее делать это с помощью параметров команды `systemctl` следующим образом:

```
# systemctl list-unit-files --type=service
UNIT FILE                                   STATE
...
cups.service                                enabled
...
dbus.service                                static
...
NetworkManager.service                      enabled
...
poweroff.service                            static
...
sshd.service                                enabled
sssd.service                                disabled
...
276 unit files listed.
```

Все файлы конфигурации юнита, показанные в предыдущем примере, связаны с сервисным юнитом. Файлы конфигурации для целевых юнитов отображаются следующим образом:

```
# systemctl list-unit-files --type=target
UNIT FILE                  STATE
anaconda.target            static
basic.target               static
bluetooth.target           static
cryptsetup.target          static
ctrl-alt-del.target        disabled
default.target             enabled
...
shutdown.target            static
sigpwr.target              static
smartcard.target           static
```

```
sockets.target             static
sound.target               static
swap.target                static
sysinit.target             static
syslog.target              static
time-sync.target           static
umount.target              static
43 unit files listed.
```

Обратите внимание на то, что в обоих примерах файлов конфигурации юнитов сами юниты отображаются со значением статуса `static`, `enabled` или `disabled`. Статус `enabled` означает, что юнит в данный момент включен. Статус `disabled` означает, что юнит в данный момент отключен. Тут все просто. А со статусом `static` может возникнуть недопонимание. Он говорит о том, что юнит «статически включен», то есть включен по умолчанию и не может быть отключен даже суперпользователем.

Файлы конфигурации сервисного юнита содержат множество информации, например, какие дополнительные службы должны быть запущены и когда какой файл среды использовать и т. д. В следующем примере показан файл конфигурации юнита демона `sshd`:

```
# cat /lib/systemd/system/sshd.service
[Unit]
Description=OpenSSH server daemon
Documentation=man:sshd(8) man:sshd_config(5)
After=network.target sshd-keygen.target

[Service]
Type=notify
EnvironmentFile=-/etc/crypto-policies/back-ends/opensshserver.config
EnvironmentFile=-/etc/sysconfig/sshd
ExecStart=/usr/sbin/sshd -D $OPTIONS $CRYPTO_POLICY
ExecReload=/bin/kill -HUP $MAINPID
KillMode=process
Restart=on-failure
RestartSec=42s

[Install]
WantedBy=multi-user.target

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

Этот базовый файл конфигурации сервисного юнита обладает следующими параметрами.

- **Description** — описание службы в свободной форме (строка комментария).
- **Documentation** — список справочных страниц для демона `sshd` и файла конфигурации.

- **After** — устанавливает порядок. Другими словами, параметр перечисляет, какие устройства должны быть активизированы до запуска определенной службы.
- **Environment File** — файлы конфигурации службы.
- **ExecStart** — команда для запуска этой службы.
- **ExecReload** — команда для перезагрузки этой службы.
- **WantedBy** — целевой юнит, к которому относится данная служба.

Обратите внимание на то, что целевой юнит `multi-user.target` используется в файле конфигурации сервисного юнита `sshd`. А это значит, что сервисный юнит `sshd` относится к целевому юниту `multi-user.target`. Другими словами, когда активизируется целевой юнит `multi-user.target`, запускается сервисный юнит `sshd`.

Вы можете просмотреть список юнитов, которые активизируют целевой юнит, с помощью следующей команды:

```
# systemctl show --property "Wants" multi-user.target
Wants=irqbalance.service firewalld.service plymouth-quit.service
systemd-update-utmp-runlevel.service systemd-ask-password-wall.path...
(END) q
```

К сожалению, команда `systemctl` не форматирует эти выходные данные. Она пробегает по списку так быстро, что не успеваешь увидеть все результаты. Кроме того, необходимо ввести команду `q`, чтобы вернуться в командную строку. Чтобы устранить эту проблему, передайте выходные данные в команды форматирования, которые отсортируют содержимое по алфавиту, как показано в следующем примере:

```
# systemctl show --property "Wants" multi-user.target \
    | fmt -10 | sed 's/Wants=//g' | sort
atd.service
auditd.service
avahi-daemon.service
chronyd.service
crond.service
...
```

В этом примере отображаются все службы и другие юниты, которые будут активизированы (запущены), включая `sshd`, когда активизируется юнит `multi-user.target`. Помните, что целевой юнит — это просто группа других юнитов, как показано ранее. Обратите внимание на то, что в этой группе не все юниты сервисные. В выходных данных присутствуют также юниты пути и другие целевые юниты.

Целевой юнит также имеет параметры `Wants` и `Requires`. Параметр `Wants` запускает все перечисленные юниты после активизации устройства. Если они терпят неудачу или не могут быть запущены, целевой юнит продолжает функционировать. В предыдущем примере представлен только параметр `Wants`.

Параметр `Requires` действует гораздо строже, чем `Wants`. Параметр `Requires` означает, что все перечисленные юниты активизируются (запускаются). Если они выходят из строя или не могут запуститься, то весь юнит (группа юнитов) перестает функционировать.

Вы можете просмотреть список различных юнитов, которые относятся к целевому юниту `Requires` (их необходимо активизировать, иначе юнит выйдет из строя), используя команду, приведенную в следующем примере. Обратите внимание на то, что вывод `Requires` намного короче, чем вывод `Wants` для юнита `multi-user.target`, поэтому никакого дополнительного форматирования выходных данных не требуется:

```
# systemctl show --property "Requires" multi-user.target
Requires=basic.target
```

Целевые юниты, как и сервисные, также имеют файлы конфигурации. В следующем примере показано содержимое файла конфигурации `multi-user.target`:

```
# cat /lib/systemd/system/multi-user.target
# This file is part of systemd.
#
...

[Unit]
Description=Multi-User
Documentation=man:systemd.special(7)
Requires=basic.target
Conflicts=rescue.service rescue.target
After=basic.target rescue.service rescue.target
AllowIsolate=yes
```

Этот базовый файл конфигурации целевого юнита имеет следующие параметры.

- `Description` — описание службы в свободной форме.
- `Documentation` — перечисляет соответствующие справочные страницы `systemd`.
- `Requires` — если файл `multi-user.target` активизируется, то вместе с ним активизируется и указанный целевой юнит. Если целевой юнит деактивизирован или выходит из строя, то и `multi-user.target` перестает функционировать. Если нет параметров `After` и `Before`, то одновременно активизируются как файл `multi-user.target`, так и указанный целевой юнит.
- `Conflicts` — позволяет избежать конфликтов во время работы служб. Запуск `multi-user.target` останавливает все перечисленные в нем целевые и сервисные юниты и наоборот.
- `After` — устанавливает порядок. Другими словами, параметр перечисляет, какие устройства должны быть активизированы до запуска определенной службы.
- `AllowIsolate` — представлен логическим значением `yes` или `no`. Если этот параметр установлен со значением `yes`, то целевой юнит `multi-user.target` активизируется вместе со своими зависимостями, а все остальные отключаются.

Чтобы получить более подробную информацию об этих файлах конфигурации и их параметрах, введите команды `man sys emd.service`, `systemd.target` и `man systemd.unit` в командной строке.

Понимание концепции целевых юнитов `systemd` упрощает понимание процесса загрузки Linux-сервера, использующего службу `systemd`. При загрузке службы `systemd` активизирует юнит `default.target`. Он является псевдонимом либо `multi-user.target`, либо `graphical.target`. Таким образом, в зависимости от установленных псевдонимов `alias` запускаются службы, на которые ориентирован целевой юнит.

Чтобы получить дополнительную информацию о демоне `systemd`, введите команду `man -k systemd` в командной строке.

Обратная совместимость служб systemd с SysVinit

Демон `systemd` поддерживает обратную совместимость с демоном SysVinit. Это позволяет дистрибутивам Linux постепенно переходить на использование демона `systemd`.

Хотя уровни выполнения не являются частью `systemd`, инфраструктура `systemd` создавалась для того, чтобы обеспечить совместимость с концепцией уровней выполнения. Существует семь файлов конфигурации целевого юнита, специально созданных для обратной совместимости с SysVinit:

- `runlevel0.target`;
- `runlevel1.target`;
- `runlevel2.target`;
- `runlevel3.target`;
- `runlevel4.target`;
- `runlevel5.target`;
- `runlevel6.target`.

Как вы, вероятно, уже поняли, для каждого из семи классических уровней запуска SysVinit существует файл конфигурации целевого юнита. Эти файлы символически связаны с файлами конфигурации целевого юнита, наиболее точно соответствующими исходному уровню выполнения. В следующем примере отображены символические ссылки для целевых юнитов уровня выполнения. Обратите внимание на то, что целевые юниты уровней выполнения 2, 3 и 4 символически связаны с юнитом `multi-user.target`, который наследует расширенный многопользовательский режим:

```
# ls -l /lib/systemd/system/runlevel*.target
lrwxrwxrwx. 1 root root 15 Apr 9 04:25 /lib/systemd/system/runlevel0.target
    -> poweroff.target
lrwxrwxrwx. 1 root root 13 Apr 9 04:25 /lib/systemd/system/runlevel1.target
    -> rescue.target
lrwxrwxrwx. 1 root root 17 Apr 9 04:25 /lib/systemd/system/runlevel2.target
    -> multi-user.target
lrwxrwxrwx. 1 root root 17 Apr 9 04:25 /lib/systemd/system/runlevel3.target
    -> multi-user.target
```

```
lrwxrwxrwx. 1 root root 17 Apr 9 04:25 /lib/systemd/system/runlevel4.target
    -> multi-user.target
lrwxrwxrwx. 1 root root 16 Apr 9 04:25 /lib/systemd/system/runlevel5.target
    -> graphical.target
lrwxrwxrwx. 1 root root 13 Apr 9 04:25 /lib/systemd/system/runlevel6.target
    -> reboot.target
```

Файл `/etc/inittab` все еще существует, но содержит только комментарии, указывающие на то, что он не используется, а лишь передает базовую информацию в `systemd`. Файл `/etc/inittab` больше не имеет истинного функционального применения. Далее приведен пример файла `/etc/inittab` на сервере Linux, задействующем `systemd`:

```
# cat /etc/inittab
# inittab is no longer used.
#
# ADDING CONFIGURATION HERE WILL HAVE NO EFFECT ON YOUR SYSTEM.
#
# Ctrl-Alt-Delete is handled by
# /etc/systemd/system/ctrl-alt-del.target
#
# systemd uses 'targets' instead of runlevels.
# By default, there are two main targets:
#
# multi-user.target: analogous to runlevel 3
# graphical.target: analogous to runlevel 5
#
# To view current default target, run:
# systemctl get-default
#
# To set a default target, run:
# systemctl set-default TARGET.target
```

В файле `/etc/inittab` объясняется, что, если вы хотите получить что-то похожее на классический уровень выполнения 3 или 5 в качестве уровня по умолчанию, нужно запустить команду `systemctl default.target` и установить нужный уровень выполнения. Чтобы проверить, с чем `default.target` в данный момент символически связан (или, используя устаревшие термины, проверить уровень выполнения по умолчанию), примените команду, показанную далее. Видно, что на этом сервере Linux по умолчанию запускается на уровне выполнения 3:

```
# ls -l /etc/systemd/system/default.target
lrwxrwxrwx. 1 root root 36 Mar 13 17:27
    /etc/systemd/system/default.target ->
        /lib/systemd/system/runlevel3.target
```

По-прежнему доступна возможность переключения уровней выполнения с помощью команды `init` или `telinit`. Любая из этих команд преобразуется в запрос активизации целевого юнита `systemd`. Поэтому, введя команду `init 3` в командной строке, в действительности вы получите команду `systemctl isolate multi-user.target`. Кроме того, все еще можно использовать команду `runlevel`

для определения текущего устаревшего уровня выполнения, но делать это я настоятельно не рекомендую.

Классический файл SysVinit `/etc/inittab` обрабатывал появление процессов `getty` или `mingetty`. `Systemd init` обрабатывает их через юнит `getty.target`. Данный юнит активизируется целевым юнитом `multi-user.target`. С помощью следующей команды можно увидеть, как эти два целевых юнита связаны между собой:

```
# systemctl show --property "WantedBy" getty.target
WantedBy=multi-user.target
```

Теперь, когда у вас появилась основа для понимания классических и современных демонов инициализации, пришло время перейти к практическим действиям администратора сервера, связанным с демоном `init`.

Проверка статуса службы

Как администратор Linux, вы должны проверять состояние служб на сервере. По соображениям безопасности необходимо отключить и удалить все неиспользуемые системные службы, обнаруженные в ходе проверки. Для устранения неполадок самое главное — иметь возможность быстро узнать, что должно и что не должно работать на сервере Linux.

Конечно, для начала нужно узнать, какую службу инициализации применяет ваш сервер Linux. Как это определить, описано в разделе «Демон инициализации (init или systemd)» ранее в этой главе. Следующие разделы посвящены различным демонам инициализации.

Проверка служб систем SysVinit

Чтобы просмотреть все службы сервера Linux, использующего классический демон SysVinit, примените команду `chkconfig`. В следующем примере показаны службы, доступные на классическом сервере с демоном SysVinit. Обратите внимание на то, что для всех служб каждый уровень выполнения (0–6) отображается со статусом `on` или `off`. Статус указывает, запущена конкретная служба (`on`) или нет (`off`) для этого уровня выполнения:

```
# chkconfig --list
ConsoleKit      0:off  1:off  2:off  3:on   4:on   5:on   6:off
NetworkManager  0:off  1:off  2:off  3:off  4:off  5:off  6:off
...
crond           0:off  1:off  2:on   3:on   4:on   5:on   6:off
cups            0:off  1:off  2:on   3:on   4:on   5:on   6:off
...
sshd            0:off  1:off  2:on   3:on   4:on   5:on   6:off
syslog          0:off  1:off  2:on   3:on   4:on   5:on   6:off
tux             0:off  1:off  2:off  3:off  4:off  5:off  6:off
```

```
udev-post       0:off  1:off  2:off  3:on   4:on   5:on   6:off
vncserver       0:off  1:off  2:off  3:off  4:off  5:off  6:off
winbind         0:off  1:off  2:off  3:off  4:off  5:off  6:off
wpa_supplicant  0:off  1:off  2:off  3:off  4:off  5:off  6:off
xfs             0:off  1:off  2:on   3:on   4:on   5:on   6:off
ypbind          0:off  1:off  2:off  3:off  4:off  5:off  6:off
yum-updatesd    0:off  1:off  2:off  3:on   4:on   5:on   6:off
```

Некоторые службы никогда не запускаются, например `vncserver`. Другие службы, например демон `cups`, запускаются на уровнях выполнения со второго по пятый.

С помощью команды `chkconfig` нельзя определить, запущена ли служба в данный момент. Для этого необходимо воспользоваться командой `service`. Чтобы отделить только запущенные в данный момент службы, команда `service` передается по конвейеру в команду `grep`, а затем сортирует вывод следующим образом:

```
# service --status-all | grep running... | sort
anacron (pid 2162) is running...
atd (pid 2172) is running...
auditd (pid 1653) is running...
automount (pid 1952) is running...
console-kit-daemon (pid 2046) is running...
crond (pid 2118) is running...
cupsd (pid 1988) is running...
...
sshd (pid 2002) is running...
syslogd (pid 1681) is running...
xfs (pid 2151) is running...
yum-updatesd (pid 2205) is running...
```

Можно использовать обе команды, `chkconfig` и `service`, чтобы просмотреть параметры каждой службы. С помощью обеих команд вы сможете просмотреть настройки демона `cups`:

```
# chkconfig --list cups
cups            0:off  1:off  2:on   3:on   4:on   5:on   6:off
#
# service cups status
cupsd (pid 1988) is running...
```

Здесь видно, что демон `cupsd` настроен на запуск на каждом уровне выполнения, кроме 0, 1 и 6, а команда `service` показывает, что он в данный момент работает. Кроме того, для демона задается номер идентификатора процесса (PID).

Чтобы просмотреть все службы демона `systemd` для сервера Linux, используйте следующую команду:

```
# systemctl list-unit-files --type=service | grep -v disabled
UNIT FILE                                   STATE
abrt-ccpp.service                           enabled
abrt-oops.service                           enabled
abrt-vmcore.service                         enabled
abrtd.service                               enabled
```

```
alsa-restore.service                          static
alsa-store.service                            static
anaconda-shell@.service                       static
arp-ethers.service                            enabled
atd.service                                   enabled
auditd.service                                enabled
avahi-daemon.service                          enabled
bluetooth.service                             enabled
console-kit-log-system-restart.service        static
console-kit-log-system-start.service          static
console-kit-log-system-stop.service           static
crond.service                                 enabled
cups.service                                  enabled
...
sshd-keygen.service                           enabled
sshd.service enabled
system-setup-keyboard.service                 enabled
...
134 unit files listed.
```

Помните, что для службы `systemd` доступно три варианта состояния: `enabled` (включен), `disabled` (отключен) и `static` (статически включен). Нет необходимости включать состояние `disabled`, чтобы увидеть, какие службы будут активны. Для этого можно использовать команду `grep` с параметром `-v`, как показано в предыдущем примере. Состояние `static` по умолчанию активно и, значит, должно быть включено в вывод.

Чтобы узнать, запущена ли конкретная служба, задействуйте следующую команду:

```
# systemctl status cups.service
cups.service – CUPS Scheduler
  Loaded: loaded (/lib/systemd/system/cups.service; enabled)
  Active: active (running) since Wed 2019-09-18 17:32:27 EDT; 3 days ago
    Docs: man:cupsd(8)
Main PID: 874 (cupsd)
  Status: "Scheduler is running..."
   Tasks: 1 (limit: 12232)
  Memory: 3.1M
  CGroup: /system.slice/cups.service
          └─874 /usr/sbin/cupsd -l
```

Команда `systemctl` может использоваться для отображения состояния одной или нескольких служб. В предыдущем примере была выбрана служба печати. Обратите внимание на то, что имя службы — это `cups.service`. В примере приводится много полезной информации о службе, например, что она включена и активна, а также время ее запуска и идентификатор процесса (PID).

Теперь, когда вы можете проверить состояние служб и узнать что-то о них, необходимо научиться выполнять запуск, остановку и перезагрузку служб на вашем сервере Linux.

Запуск и остановка служб

Запуск, остановка и перезапуск служб обычно относятся к необходимым задачам, другими словами, задачам управления службами без перезагрузки сервера. Например, если вы хотите временно остановить службу, подойдет этот раздел книги. Но если хотите остановить службу и не позволять ей перезапускаться при перезагрузке сервера, то действительно нужно отключить ее, как описано в разделе «Подключение постоянных служб» далее в этой главе.

Остановка и запуск служб SysVinit

Основная команда для остановки и запуска служб SysVinit — это команда `service`. С ее помощью имя нужной службы ставится на второе место в командной строке. В конце указывается подходящее действие для службы: `stop`, `start`, `restart` и т. д. В следующем примере показано, как остановить службу `cups`. Обратите внимание на значение `OK` — оно дает понять, что служба `cupsd` была успешно остановлена:

```
# service cups status
cupsd (pid 5857) is running...
# service cups stop
Stopping cups:            [ OK ]
# service cups status
cupsd is stopped
```

Чтобы запустить службу, используйте параметр `start` вместо параметра `stop` в конце команды `service`, как показано далее:

```
# service cups start
Starting cups:            [ OK ]
# service cups status
cupsd (pid 6860) is running...
```

Для перезапуска службы SysVinit используется параметр `restart`. Он останавливает службу и затем сразу же запускает ее снова:

```
# service cups restart
Stopping cups:            [ OK ]
Starting cups:            [ OK ]
# service cups status
cupsd (pid 7955) is running...
```

Когда служба уже остановлена, параметр `restart` генерирует состояние `FAILED` при попытке остановить ее еще раз. Однако, как показано в следующем примере, служба успешно запускается при попытке перезапуска:

```
# service cups stop
Stopping cups:            [  OK  ]
# service cups restart
Stopping cups:            [FAILED]
```

```
Starting cups:           [  OK  ]
# service cups status
cupsd (pid 8236) is running...
```

Перезагрузка службы отличается от перезапуска службы. При перезагрузке сама служба не останавливается — снова загружаются только ее файлы конфигурации. В следующем примере показано, как перезагрузить демон `cups`:

```
# service cups status
cupsd (pid 8236) is running...
# service cups reload
Reloading cups:          [ OK ]
# service cups status
cupsd (pid 8236) is running...
```

Если служба SysVinit остановлена в процессе перезагрузки, появится статус `FAILED`. Это показано в следующем примере:

```
# service cups status
cupsd is stopped
# service cups reload
Reloading cups: [FAILED]
Stopping and starting systemd services
```

Для демона `systemd` команда `systemctl` позволяет останавливать, запускать, перезагружать и перезапускать службы. Параметры команды `systemctl` похожи.

Остановка службы с помощью демона systemd

В следующем примере состояние демона `cups` проверяется, а затем он останавливается с помощью команды `systemctl stop cups.service`:

```
# systemctl status cups.service
cups.service – CUPS Printing Service
      Loaded: loaded (/lib/systemd/system/cups.service; enabled)
      Active: active (running) since Mon, 20 Apr 2020 12:36:3...
    Main PID: 1315 (cupsd)
      CGroup: name=systemd:/system/cups.service
              1315 /usr/sbin/cupsd -f
# systemctl stop cups.service
# systemctl status cups.service
cups.service – CUPS Printing Service
      Loaded: loaded (/lib/systemd/system/cups.service; enabled)
      Active: inactive (dead) since Tue, 21 Apr 2020 04:43:4...
      Process: 1315 ExecStart=/usr/sbin/cupsd -f
   (code=exited, status=0/SUCCESS)
      CGroup: name=systemd:/system/cups.service
```

Обратите внимание на то, что при проверке статуса после остановки демона `cups` видно, что служба неактивна (`dead`), но все еще считается включенной. Это означает, что демон `cups` будет запускаться при загрузке сервера.

Запуск службы с помощью демона systemd

Запустить демон cups так же просто, как и остановить его. В следующем примере это показано:

```
# systemctl start cups.service
# systemctl status cups.service
cups.service – CUPS Printing Service
      Loaded: loaded (/lib/systemd/system/cups.service; enabled)
      Active: active (running) since Tue, 21 Apr 2020 04:43:5...
    Main PID: 17003 (cupsd)
      CGroup: name=systemd:/system/cups.service
              └ 17003 /usr/sbin/cupsd -f
```

После запуска демона cups использование команды systemctl с параметром status говорит о том, что служба активна (запущена), кроме того, показан ее номер идентификатора процесса (PID) — 17003.

Перезапуск службы с помощью демона systemd

Перезапуск службы означает, что служба останавливается, а затем запускается снова. Если в данный момент она не запущена, перезапуск просто запустит ее:

```
# systemctl restart cups.service
# systemctl status cups.service
cups.service – CUPS Printing Service
      Loaded: loaded (/lib/systemd/system/cups.service; enabled)
      Active: active (running) since Tue, 21 Apr 2020 04:45:2...
    Main PID: 17015 (cupsd)
      CGroup: name=systemd:/system/cups.service
              └ 17015 /usr/sbin/cupsd -f
```

Вы также можете выполнить условный перезапуск службы с помощью команды systemctl. Условный перезапуск перезапускает службу только в том случае, если она в данный момент запущена. Любая служба в неактивном состоянии не запускается:

```
# systemctl status cups.service
cups.service - CUPS Printing Service
     Loaded: loaded (/lib/systemd/system/cups.service; enabled)
     Active: inactive (dead) since Tue, 21 Apr 2020 06:03:32...
    Process: 17108 ExecStart=/usr/sbin/cupsd -f
     (code=exited, status=0/SUCCESS)
     CGroup: name=systemd:/system/cups.service
# systemctl condrestart cups.service
# systemctl status cups.service
cups.service – CUPS Printing Service
     Loaded: loaded (/lib/systemd/system/cups.service; enabled)
     Active: inactive (dead) since Tue, 21 Apr 2020 06:03:32...
    Process: 17108 ExecStart=/usr/sbin/cupsd -f
    (code=exited, status=0/SUCCESS)
     CGroup: name=systemd:/system/cups.service
```

Обратите внимание на то, что в примере демон `cups` находился в неактивном состоянии. При условном перезапуске сообщения об ошибках не генерировались! Демон `cups` не был запущен, поскольку условные перезапуски влияют лишь на активные службы. Таким образом, всегда полезно проверять состояние службы после остановки, запуска, условного перезапуска и т. д.

Перезагрузка службы с помощью демона systemd

Перезагрузка службы отличается от ее перезапуска. При перезагрузке служба не останавливается, вновь загружаются только ее файлы конфигурации. В следующем примере показано, как перезагрузить демон `cups`:

```
# systemctl status sshd.service
sshd.service – OpenSSH server daemon
      Loaded: loaded (/usr/lib/systemd/system/sshd.service; enabled)
      Active: active (running) since Wed 2019-09-18 17:32:27 EDT; 3 days ago
    Main PID: 1675 (sshd)
      CGroup: /system.slice/sshd.service
              └─1675 /usr/sbin/sshd -D
# systemctl reload sshd.service
# systemctl status sshd.service
sshd.service – OpenSSH server daemon
      Loaded: loaded (/lib/systemd/system/sshd.service; enabled)
      Active: active (running) since Wed 2019-09-18 17:32:27 EDT; 3 days ago
     Process: 21770 ExecReload=/bin/kill -HUP $MAINPID (code=exited, status=0/
SUCCESS)
     (code=exited, status=0/SUCCESSd)
    Main PID: 1675 (sshd)
      CGroup: /system.slice/sshd.service
              └─1675 /usr/sbin/sshd -D ...
```

Перезагрузка службы с помощью команды `reload` вместо перезапуска предотвращает прерывание любых ее отложенных действий. Перезагрузка — это лучший вариант для используемого сервера Linux.

Теперь, когда вы знаете, как останавливать и запускать службы, чтобы устранять неполадки и аварийные ситуации, вы должны научиться включать и отключать их.

Подключение постоянных служб

Параметры `stop` и `start` применяются для неотложных задач, а не для постоянных служб. *Постоянная служба* — это служба, которая запускается во время загрузки сервера или на определенном уровне выполнения. Службы, которые должны быть постоянными, — это обычно новые службы сервера Linux.

Настройка постоянных служб для демона SysVinit

Одна из приятных особенностей классического демона SysVinit заключается в том, что сделать определенную службу постоянной или отменить это состояние очень легко. Рассмотрим следующий пример:

```
# chkconfig --list cups
cups            0:off  1:off  2:off  3:off  4:off  5:off  6:off
```

На этом сервере Linux служба `cups` не запускается ни на одном из уровней выполнения, что показано с помощью команды `chkconfig`. Вы также можете проверить, установлены ли какие-либо символьные ссылки запуска (`S`) в каждом из семи каталогов уровня выполнения `/etc/rc.d/rc?d`. Помните, что демон SysVinit хранит в этом каталоге символические ссылки для запуска и остановки различных служб на определенных уровнях выполнения. Каждый каталог представляет определенный уровень выполнения, например, `rc5.d` предназначен для уровня выполнения 5. Обратите внимание на то, что в списке перечислены только файлы, начинающиеся с буквы `K`, поэтому появились ссылки для немедленного завершения демона `cups`. Ни один из них не обозначен буквой `S`, что согласуется с командой `chkconfig`, поскольку демон `cups` не запускается ни на одном из уровней выполнения на этом сервере:

```
# ls /etc/rc.d/rc?.d/*cups
/etc/rc.d/rc0.d/K10cups   /etc/rc.d/rc3.d/K10cups
/etc/rc.d/rc1.d/K10cups   /etc/rc.d/rc4.d/K10cups
/etc/rc.d/rc2.d/K10cups   /etc/rc.d/rc5.d/K10cups
/etc/rc.d/rc6.d/K10cups
```

Чтобы сделать службу постоянной на определенном уровне выполнения, снова используйте команду `chkconfig`. Вместо параметра `--list` введите параметр `--level`, как показано в следующем примере:

```
# chkconfig --level 3 cups on
# chkconfig --list cups
cups            0:off  1:off  2:off  3:on   4:off  5:off  6:off
# ls /etc/rc.d/rc3.d/S*cups
/etc/rc.d/rc3.d/S56cups
```

Постоянство службы на уровне выполнения 3 проверяется с помощью команды `chkconfig --list` и просмотра каталога `rc3.d` для любых файлов, начинающихся с буквы `S`. Чтобы сделать службу постоянной на нескольких уровнях выполнения, можно выполнить следующие действия:

```
# chkconfig --level 2345 cups on
# chkconfig --list cups
cups            0:off  1:off  2:on   3:on   4:on   5:on   6:off
# ls /etc/rc.d/rc?.d/S*cups
/etc/rc.d/rc2.d/S56cups   /etc/rc.d/rc4.d/S56cups
/etc/rc.d/rc3.d/S56cups   /etc/rc.d/rc5.d/S56cups
```

Отключить службу так же просто, как и включить ее с помощью демона SysVinit. Для этого нужно изменить значение `on` в команде `chkconfig` на `off`. В следующем примере показано использование команды `chkconfig` для отключения службы `cups` на уровне выполнения 5:

```
# chkconfig --level 5 cups off
# chkconfig --list cups
cups            0:off  1:off  2:on   3:on   4:on   5:off  6:off
# ls /etc/rc.d/rc5.d/S*cups
ls: cannot access /etc/rc.d/rc5.d/S*cups: No such file or directory
```

Как и ожидалось, теперь нет символической ссылки, начинающейся с буквы `S`, для службы `cups` в каталоге `/etc/rc.d/rc5.d`. Для демона `systemd` снова используется команда `systemctl`. С ее помощью вы можете отключать и включать службы на сервере Linux.

Включение службы с помощью демона systemd

Параметр `enable` в команде `systemctl` настраивает запуск служб при загрузке системы (делает службы постоянными). Далее показано, как именно это сделать:

```
# systemctl status cups.service
cups.service – CUPS Printing Service
      Loaded: loaded (/lib/systemd/system/cups.service; disabled)
      Active: inactive (dead) since Tue, 21 Apr 2020 06:42:38 ...
    Main PID: 17172 (code=exited, status=0/SUCCESS)
      CGroup: name=systemd:/system/cups.service
# systemctl enable cups.service
Created symlink /etc/systemd/system/printer.target.wants/cups.service
    → /usr/lib/systemd/system/cups.service.
Created symlink /etc/systemd/system/sockets.target.wants/cups.socket
    → /usr/lib/systemd/system/cups.socket.
Created symlink /etc/systemd/system/multi-user.target.wants/cups.path
    → /usr/lib/systemd/system/cups.path.
# systemctl status cups.service
cups.service – CUPS Printing Service
      Loaded: loaded (/lib/systemd/system/cups.service; enabled)
      Active: inactive (dead) since Tue, 21 Apr 2020 06:42:38...
    Main PID: 17172 (code=exited, status=0/SUCCESS)
      CGroup: name=systemd:/system/cups.service
```

Обратите внимание на то, что статус службы `cups.service` изменился после использования параметра `enable` в команде `systemctl`. Кроме того, отметьте для себя, что параметр `enable` просто создает несколько символических ссылок. Может возникнуть искушение создать эти ссылки самостоятельно. Однако предпочтительнее применять для этого именно команду `systemctl`.

Отключение службы с помощью демона systemd

Вы можете использовать параметр `disable` в команде `systemctl`, чтобы предотвратить запуск службы при загрузке. Однако это действие не сразу останавливает службу. Для этого нужен параметр `stop`, который описан в пункте «Остановка службы с помощью демона systemd». В следующем примере показано, как отключить включенную в данный момент службу:

```
# systemctl disable cups.service
rm '/etc/systemd/system/printer.target.wants/cups.service'
rm '/etc/systemd/system/sockets.target.wants/cups.socket'
rm '/etc/systemd/system/multi-user.target.wants/cups.path'
# systemctl status cups.service
cups.service – CUPS Printing Service
      Loaded: loaded (/lib/systemd/system/cups.service; disabled )
      Active: active (running) since Tue, 21 Apr 2020 06:06:41...
    Main PID: 17172 (cupsd)
      CGroup: name=systemd:/system/cups.service
              17172 /usr/sbin/cupsd -f
```

Параметр `disable` просто удаляет несколько файлов с помощью команды `systemctl`. Обратите также внимание на то, что в предыдущем примере, хоть служба `cups` и отключена, демон `cups` все еще активен (работает) и должен быть остановлен вручную. Некоторые службы нельзя отключить с помощью демона `systemd`. Таковы статические службы. Рассмотрим пример со службой `dbus.service`:

```
# systemctl status dbus.service
dbus.service – D-Bus System Message Bus
      Loaded: loaded (/lib/systemd/system/dbus.service; static)
      Active: active (running) since Mon, 20 Apr 2020 12:35:...
    Main PID: 707 (dbus-daemon)
...
# systemctl disable dbus.service
# systemctl status dbus.service
dbus.service – D-Bus System Message Bus
      Loaded: loaded (/lib/systemd/system/dbus.service; static)
      Active: active (running) since Mon, 20 Apr 2020 12:35:...
    Main PID: 707 (dbus-daemon)
...
```

Когда команда `systemctl disable` задействуется для службы `dbus.service`, ничего не происходит. Помните: значение `static` означает, что служба включена по умолчанию и не может быть отключена даже суперпользователем. Иногда отключения службы недостаточно, чтобы точно понять, что она не работает. Например, вам нужно, чтобы служба `network.service` заменила службу `NetworkManager.service` для запуска сетевых интерфейсов.

Отключение службы NetworkManager не позволит ей запуститься самостоятельно. Но если какая-то другая служба перечислит NetworkManager в качестве зависимости, то при запуске она попытается запустить и NetworkManager.

Отключить службу так, чтобы она совсем перестала работать в системе, можно с помощью параметра `mask`. Например, чтобы настроить службу NetworkManager так, чтобы она никогда не запускалась, введите следующее:

```
# systemctl mask NetworkManager.service
ln -s '/dev/null' '/etc/systemd/system/NetworkManager.service'
```

Как видно из выходных данных, файл `NetworkManager.service` в файле `/etc` связан с файлом `/dev/null`. Поэтому, даже если кто-то попытается запустить эту службу, ничего не произойдет. Чтобы снова задействовать эту службу, введите команду `systemctl unmask NetworkManager.service`.

Теперь, когда вы знаете, как сделать отдельные службы постоянными (и как отключить или замаскировать службы), необходимо взглянуть на группы служб в целом. Далее я расскажу, как запускать группы служб во время загрузки.

Настройка уровня выполнения по умолчанию (целевого юнита)

Постоянная служба — это служба, запускаемая во время загрузки сервера, а постоянный (по умолчанию) уровень выполнения, или целевой юнит, — это группа служб, запускаемых во время загрузки. Как классический SysVinit, так и Upstart определяют эти группы служб как уровни выполнения, а `systemd` называет их целевыми юнитами.

Настройка уровня выполнения по умолчанию демона SysVinit

Постоянный уровень запуска для сервера Linux с помощью SysVinit устанавливается в файле `/etc/inittab`. Часть этого файла показана далее:

```
# cat /etc/inittab
#
# inittab         This file describes how the INIT process should
#                 set up the system in a certain run-level.
...
id:5:initdefault:
...
```

Строка `initdefault` в примере показывает, что текущий уровень выполнения по умолчанию — это уровень выполнения 5. Чтобы изменить это, просто отредактируйте файл `/etc/inittab` с помощью текстового редактора и измените 5 на один из уровней запуска: 2, 3 или 4. Не используйте уровни запуска 0 или 6 в этом файле! Это приведет к тому, что ваш сервер либо остановится, либо перезагрузится при запуске.

Для демона systemd *целевые юниты* относятся к группам служб, которые должны быть запущены. Далее показаны различные целевые юниты, которые можно сделать постоянными, а также их обратно совместимые целевые юниты, подходящие для уровня выполнения:

- multi-user.target =;
  - runlevel2.target;
  - runlevel3.target;
  - runlevel4.target;
- graphical.target = runlevel5.target.

Постоянный целевой юнит устанавливается с помощью символической ссылки на файл default.target unit. Рассмотрим следующий пример:

```
# ls -l /etc/systemd/system/default.target
lrwxrwxrwx. 1 root root 36 Mar 13 17:27
    /etc/systemd/system/default.target ->
    /lib/systemd/system/runlevel5.target
# ls -l /lib/systemd/system/runlevel5.target
lrwxrwxrwx. 1 root root 16 Mar 27 15:39
    /lib/systemd/system/runlevel5.target ->
    graphical.target
```

Здесь видно, что текущим постоянным целевым юнитом на этом сервере является runlevel5.target, поскольку default.targe — это символическая ссылка на файл юнита runlevel5.target. Но обратите внимание на то, что runlevel5.target также является символической ссылкой и указывает на graphical.target. Таким образом, текущий постоянный целевой юнит этого сервера — graphical.target.

Чтобы сделать другой целевой юнит постоянным, нужно просто изменить символическую ссылку на default.target.

Последовательно придерживайтесь целевых юнитов уровня выполнения, если они используются на вашем сервере. В следующем примере команда systemctl изменяет постоянный целевой юнит сервера с graphical.target на multi-user.target:

```
# systemctl get-default
graphical.target
#
    systemctl set-default runlevel3.target
    Removed /etc/systemd/system/default.target.
    Created symlink /etc/systemd/system/default.target →
/usr/lib/systemd/system/multi-user.target.
# systemctl get-default
    multi-user.target
```

Когда сервер перезагружается, multi-user.target становится постоянным целевым юнитом. В это время все службы в юните multi-user.target запускаются (активизируются).

Добавление новых или пользовательских служб

Иногда требуется добавить новую службу для вашего сервера Linux. И возможно, вам придется настроить определенную службу самостоятельно. С этой целью необходимо выполнить определенные шаги для демона инициализации вашего Linux-сервера, то есть управлять службой либо распознать ее пользовательские настройки.

Добавление новых служб в SysVinit

При добавлении новой или пользовательской службы к серверу с помощью демона SysVinit необходимо выполнить определенные действия и позволить SysVinit управлять этой службой.

1. Создать скрипт-файл для новой или пользовательской службы.
2. Переместить скрипт новой или пользовательской службы в нужное место для управления демоном SysVinit.
3. Установить соответствующие права на скрипт.
4. Добавить службу на определенный уровень выполнения.

Шаг 1. Создать скрипт-файл для новой или пользовательской службы

Если вы настраиваете существующий скрипт службы, просто сделайте копию исходного файла юнита из файла `/etc/rc.d/init.d` и добавьте любые необходимые настройки.

Если вы создаете новый скрипт, убедитесь, что применяете все необходимые параметры, которые команда `service` должна принимать, такие как `start`, `stop`, `restart` и т. д.

Создавая новый скрипт, особенно если это происходит в первый раз, было бы разумно скопировать текущий скрипт службы из файла `/etc/rc.d/init.d` и изменить его под нужды службы. Рассмотрим пример скрипта службы `cupsd`:

```
# cat /etc/rc.d/init.d/cups
#!/bin/sh
#
...
#    chkconfig: 2345 25 10
...
start () {
        echo -n $"Starting $prog: "
        # start daemon
        daemon $DAEMON
        RETVAL=$?
```

```
        echo
         [ $RETVAL = 0 ] && touch /var/lock/subsys/cups
        return $RETVAL
}
stop () {
        # stop daemon
        echo -n $"Stopping $prog: "
        killproc $DAEMON
        RETVAL=$?
        echo         [ $RETVAL = 0 ] && rm -f /var/lock/subsys/cups
}
restart() {
        stop
        start
}
case $1 in
...
```

Скрипт службы `cups` начинается с создания функций для каждого из параметров `start`, `stop` и `restart`. Если у вас возникли проблемы с написанием скриптов оболочки, обратитесь к главе 7 «Простейшие скрипты оболочки».

Строка, которую обязательно нужно проверить и, возможно, изменить в новом скрипте, — это закомментированная строка `chkconfig`, например:

```
# chkconfig: 2345 25 10
```

При добавлении скрипта службы на более позднем этапе команда `chkconfig` считывает эту строку, чтобы установить уровни выполнения для служб `start` (2, 3, 4 и 5), порядок выполнения, когда скрипт настроен на службы `start` (25), и порядок завершения, когда он установлен на службы `stop` (10). Проверьте порядок загрузки на уровне выполнения по умолчанию, прежде чем добавлять собственный скрипт, как показано в примере:

```
# ls /etc/rc5.d
...
/etc/rc5.d/S22messagebus
/etc/rc5.d/S23NetworkManager
/etc/rc5.d/S24nfslock
/etc/rc5.d/S24openct
/etc/rc5.d/S24rpcgssd
/etc/rc5.d/S25blk-availability
/etc/rc5.d/S25cups
/etc/rc5.d/S25netfs
/etc/rc5.d/S26acpid
/etc/rc5.d/S26haldaemon
/etc/rc5.d/S26hypervkvpd
/etc/rc5.d/S26udev-post
...
```

В этом случае строка `chkconfig` в скрипте `S25My_New_Service` приведет к добавлению скрипта после `S25cups` и перед `S25netfs` в порядке загрузки. По желанию

можете изменить строку hkconfig в скрипте службы так, чтобы служба запускалась раньше (используйте меньшее число) или позже (возьмите большее число) в списке скриптов службы.

Шаг 2. Добавьте скрипт службы в файл /etc/rc.d/init.d

После того как вы изменили или создали и протестировали файл скрипта службы, можете переместить его в нужное место — в файл /etc/rc.d/init.d:

```
# cp My_New_Service /etc/rc.d/init.d
# ls /etc/rc.d/init.d/My_New_Service
/etc/rc.d/init.d/My_New_Service
```

Шаг 3. Установите соответствующие права на скрипт

Скрипт должен иметь права на выполнение:

```
# chmod 755 /etc/rc.d/init.d/My_New_Service
```

Шаг 4. Добавьте службу в каталоги уровней выполнения

Последний шаг настраивает скрипты службы для запуска и остановки на разных уровнях выполнения и проверяет, работают ли они.

1. Чтобы добавить скрипт на основе строки chkconfig в скрипт службы, введите следующее:

   ```
   # chkconfig --add My_New_Service
   # ls /etc/rc?.d/*My_New_Service
   /etc/rc0.d/K10My_New_Service   /etc/rc4.d/S25My_New_Service
   /etc/rc1.d/K10My_New_Service   /etc/rc5.d/S25My_New_Service
   /etc/rc2.d/S25My_New_Service   /etc/rc6.d/K10My_New_Service
   /etc/rc3.d/S25My_New_Service
   ```

   Как и в предыдущем примере (chkconfig: 2345 25 10), символические ссылки на скрипт устанавливают запуск службы в позиции 25 (S25) для уровней выполнения 2, 3, 4 и 5. Кроме того, ссылки останавливаются (или не запускаются) на уровнях выполнения 0, 1 и 6.

2. После того как вы создали символическую ссылку (ссылки), проверьте, что новая или измененная служба работает должным образом, прежде чем выполнять перезагрузку сервера.

   ```
   # service My_New_Service start
   Starting My_New_Service:           [ OK ]
   # service My_New_Service stop
   ```

Когда все готово, новая или измененная служба запускается на каждом уровне запуска, выбранном в системе. Кроме того, вы можете запустить или остановить ее вручную с помощью команды service.

Добавление новых служб к демону systemd

При добавлении новой или пользовательской службы `systemd` на сервер Linux необходимо выполнить три шага, чтобы передать управление службой демону `systemd`.

1. Создайте файл конфигурации для нового или пользовательского юнита службы.
2. Переместите скрипт новой или пользовательской службы в нужное место для управления демоном `systemd`.
3. Добавьте службу к параметру `Wants` целевого юнита, чтобы новая или пользовательская служба автоматически запускалась вместе с другими службами.

Шаг 1. Создайте файл конфигурации для нового или пользовательского юнита службы

Если вы изменяете файл конфигурации юнита службы, просто сделайте копию исходного файла юнита из файла `/lib/systemd/system` и добавьте все необходимые настройки.

Для новых файлов нужно создать файл конфигурации юнита службы с нуля. Рассмотрим базовый шаблон файла юнита службы. Как минимум вам понадобятся описание и параметры `ExecStar` для файла конфигурации юнита службы:

```
# cat My_New_Service.service
[Unit]
Description=My New Service
[Service]
ExecStart=/usr/bin/My_New_Service
```

Для получения дополнительной информации о настройке или создании нового файла конфигурации юнита и нужных параметров воспользуйтесь справочными страницами. В командной строке введите `man systemd service`, чтобы больше узнать о различных параметрах файла юнита службы.

Шаг 2. Переместите файл конфигурации юнита службы

Перед перемещением нового или измененного файла конфигурации службы необходимо знать, что существует два возможных места хранения таких файлов. То место, которое вы выберете, определяет, вступят ли настройки в силу и останутся ли они постоянными при обновлении программного обеспечения.

Поместите файл конфигурации юнита службы в одно из следующих двух мест.

- Файл `/etc/systemd/system`.
  - Это место расположения применяется для хранения пользовательских файлов конфигурации юнитов службы.
  - Здесь файлы не перезаписываются установкой или обновлениями программного обеспечения. Файлы используются системой, *даже* если в каталоге `/lib/systemd/system` есть файл с таким же именем.

- Файл `/lib/systemd/system`.
  - Это место расположения применяется для хранения файлов конфигурации юнитов служб.
  - Файлы здесь перезаписываются установкой и обновлениями программного обеспечения. Файлы используются системой только в том случае, если в каталоге `/etc/systemd/system` нет файла с таким же именем.

Таким образом, лучшее место для хранения нового или пользовательского файла конфигурации службы — это файл `/etc/systemd/system`.

СОВЕТ

Чтобы при создании новой или пользовательской службы изменение вступило в силу без перезагрузки сервера, необходимо выполнить специальную команду. В командной строке введите systemctl daemon-reload.

Шаг 3. Добавьте службу в каталог параметра Wants

Последний шаг необязателен. Его нужно делать только в том случае, если вы хотите, чтобы новая служба начиналась с определенного целевого юнита `systemd`. Чтобы служба была активизирована (запущена) конкретным целевым юнитом, она должна находиться в его каталоге `Wants`.

Вначале добавьте строку `WantedBy=desired.target` в нижнюю часть файла конфигурации юнита службы. В следующем примере видно, что желаемым целевым юнитом для новой службы является юнит `multi-user.target`:

```
# cat /etc/systemd/system/My_New_Service.service
[Unit]
Description=My New Fake Service
[Service]
ExecStart=/usr/bin/My_New_Service
[Install]
WantedBy=multi-user.target
```

Чтобы добавить новый сервисный юнит в целевой, необходимо создать символическую ссылку. В следующем примере показаны файлы, расположенные в каталоге `Wants` юнита `multi-user.target`. Ранее в подразделе «Система инициализации systemd» команда `systemctl` использовалась для перечисления содержимого каталога `Wants`, и она по-прежнему является предпочтительным вариантом для этого. Обратите внимание на то, что в этом каталоге файлы являются символическими ссылками, указывающими на файлы конфигурации юнитов служб в каталоге `/lib/systemd/system`:

```
# ls /etc/systemd/system/multi-user.target.wants
abrt-ccpp.service     cups.path            remote-fs.target
abrtd.service         fcoe.service         rsyslog.service
abrt-oops.service     irqbalance.service   sendmail.service
```

```
abrt-vmcore.service   lldpad.service       sm-client.service
atd.service           mcelog.service       sshd-keygen.service
auditd.service        mdmonitor.service    sshd.service
...
# ls -l /etc/systemd/system/multi-user.target.wants
total 0
lrwxrwxrwx. 1 root root 37 Nov 2 22:29 abrt-ccpp.service ->
    /lib/systemd/system/abrt-ccpp.service
lrwxrwxrwx. 1 root root 33 Nov 2 22:29 abrtd.service ->
    /lib/systemd/system/abrtd.service
...
lrwxrwxrwx. 1 root root 32 Apr 26 20:05 sshd.service ->
    /lib/systemd/system/sshd.service
```

Далее представлен процесс добавления файла символической ссылки для `My_New_Service`:

```
# ln -s /etc/systemd/system/My_New_Service.service
/etc/systemd/system/multi-user.target.wants/My_New_Service.service
```

Символическая ссылка создается в каталоге `multi-user.target.wants`. Теперь новая служба `My_New_Service` активизируется (запускается) при активизации юнита `multi-user.target`.

СОВЕТ

Если вы хотите изменить целевой юнит systemd для службы, необходимо изменить символическую ссылку так, чтобы она указывала на новое целевое местоположение каталога Wants. Примените команду ln -sf, чтобы принудительно разорвать любую текущую символическую ссылку и принудительно назначить новую.

Приведенные ранее шаги добавят новую или пользовательскую службу на сервер Linux `systemd`. Помните, что новая служба не запустится до перезагрузки сервера. Чтобы запустить новую службу перед перезагрузкой, задействуйте команды, рассмотренные в подразделе «Остановка и запуск служб».

Резюме

Способ запуска и остановки служб зависит от того, какой демон инициализации применяет ваш Linux-сервер: SysVinit, Upstart или Systemd. Прежде чем приступить к управлению службами, обязательно воспользуйтесь примерами из этой главы, чтобы определить демон инициализации вашего Linux-сервера.

Принципы запуска и остановки служб сочетаются с другими принципами управления службами, такими как: создание постоянных служб, запуск определенных служб во время загрузки сервера, перезагрузка службы и перезапуск службы. Понимание этого очень полезно, поскольку в следующей главе мы будем рассматривать процесс настройки сервера печати Linux и управления им.

Упражнения

Используйте материалы этой главы, чтобы выполнить следующие упражнения. Если затрудняетесь с решением заданий, посмотрите ответы к упражнениям, приведенные в приложении Б (хотя Linux позволяет решать задачи разными способами). Выполните все упражнения, прежде чем переходить к ответам. Задачи подходят для систем Fedora или Red Hat Enterprise Linux (некоторые сработают и в других системах Linux).

1. Определите, какой демон инициализации в данный момент использует ваш сервер.
2. Какую команду можно применить для проверки состояния демона `sshd` в зависимости от демона инициализации, используемого на сервере Linux?
3. Определите предыдущий и текущий уровни выполнения своего сервера.
4. Как вы можете изменить уровень выполнения по умолчанию или целевой юнит на своем сервере Linux?
5. Какие команды перечисляют службы, запущенные (или активные) на вашем сервере для каждого демона инициализации?
6. Перечислите запущенные (или активные) службы на своем Linux-сервере.
7. Какие команды показывают текущее состояние конкретной службы для каждого демона инициализации?
8. Отобразите состояние демона `cups` на своем Linux-сервере.
9. Попробуйте перезапустить демон `cups` на своем Linux-сервере.
10. Попробуйте перезагрузить демон `cups` на своем Linux-сервере.

16 Настройка сервера печати

В этой главе

- Печать в Linux.
- Настройка принтеров.
- Команды печати.
- Управление печатью документов.
- Удаленные принтеры.

Вы можете настроить свою систему Linux для использования принтеров, подключенных непосредственно к ней (через USB-порт) или доступных по сети. Аналогично любой принтер, настроенный в локальной системе, может совместно применяться пользователями других систем Linux, Windows или macOS в качестве сервера печати.

Вы настраиваете принтер как собственный принтер Linux в Fedora, RHEL, Ubuntu и других системах Linux с помощью *общей системы печати UNIX* (Common UNIX Printing System, CUPS). Чтобы настроить принтер в качестве сервера печати в стиле Microsoft Windows, можно использовать службу Samba в Linux.

В этой же главе мы рассмотрим сервер печати CUPS. В частности, он предоставляет графический интерфейс, называемый Print Settings (Настройки принтера), который имеется в Fedora, Red Hat Enterprise Linux и других дистрибутивах Linux. С помощью окна Print Settings (Настройки принтера), вы также можете настроить свои принтеры в качестве серверов печати, чтобы другие пользователи имели возможность печатать на них документы со своих компьютеров.

Если у вас нет рабочего стола или вы хотите печатать из скрипта оболочки, в этой главе найдете информацию о том, как применять команды печати. Из командной строки доступны команды печати, например `lp`. Есть также команды для запроса печати в очереди (`LPQ`), управления очередями печати (`cupsenable`, `cupsdisable` и `cupsreject`) и удаления очередей печати (`lprm`).

Общая система печати UNIX

Система CUPS стала стандартом для печати в Linux и других UNIX-подобных операционных системах. Она была разработана, чтобы удовлетворить современные потребности в стандартизированных определениях принтеров и совместном использовании по сети на основе интернет-протоколов (как и большинство компьютерных сетей сегодня).

В настоящее время практически любой дистрибутив Linux включает в себя систему CUPS для обслуживания печати. Перечислю некоторые особенности этой службы.

- **Протокол IPP (Internet Printing Protocol).** Служба CUPS основана на протоколе межсетевой печати (www.pwg.org/ipp). Это стандарт, созданный для упрощения совместного использования принтеров по IP-сетям. Через протокол IPP серверы принтеров и клиенты, которые хотят печатать, могут обмениваться информацией о модели и функциях принтера с помощью протокола HTTP (то есть веб-содержимого). Сервер также может отображать, доступен ли принтер, чтобы клиент печати мог легко найти список локально доступных принтеров без дополнительных настроек.
- **Драйверы.** Служба CUPS стандартизировала процесс создания драйверов принтера. Идея состояла в наличии общего формата, который могли бы использовать производители принтеров, чтобы драйвер был способен работать во всех типах UNIX-систем. Таким образом, производитель мог создать только один драйвер, который годился бы и для Linux, и для macOS X, и для других производных UNIX.
- **Классы принтеров.** Классы принтеров можно использовать для создания нескольких серверов печати, применяющих один и тот же принтер, или одного сервера печати, задействующего несколько принтеров. В первом случае несколько серверов могут иметь разные параметры (например, вывод на определенный лоток для бумаги или печать с определенными размерами символов или полями). Во втором варианте вы можете иметь пул принтеров, чтобы распределить печать. В этом случае неисправный принтер или принтер, работающий с очень большими документами, не остановит весь процесс печати. Служба CUPS также поддерживает *неявные классы*, которые формируются путем автоматического слияния идентичных сетевых принтеров.
- **Обзор принтеров.** При просмотре принтеров клиентские компьютеры могут видеть все доступные принтеры CUPS в вашей локальной сети. В результате клиенты могут выбрать принтеры, которые будут использовать, среди доступных в сети, причем им нет необходимости заранее знать, как на самом деле называются принтеры и куда они подключены. Вы можете отключить эту функцию, чтобы другие пользователи локальной сети не могли видеть принтер.

- **Команды печати UNIX.** Для интеграции в Linux и другие среды UNIX служба CUPS позволяет задействовать стандартные версии команд для печати и управления принтерами, которые обычно использовались в системах UNIX.

Вы можете настроить печать CUPS и без помощи окна Print Settings (Настройки принтера) — другими способами.

- **Настройка службы CUPS из браузера.** Проект службы CUPS предлагает веб-интерфейс для добавления принтеров и управления ими. При запущенной службе `cupsd` введите `localhost:631` в браузере на компьютере, чтобы управлять печатью. (См. подраздел «Веб-администрирование службы CUPS» далее в этой главе.)
- **Настройка службы CUPS вручную.** Вы можете настроить CUPS вручную, то есть отредактировать файлы конфигурации и запустить демон `cupsd` из командной строки.

Файлы конфигурации для CUPS содержатся в каталоге `/etc/cups`. В частности, нам нужны файл `cupsd.conf`, который определяет права, аутентификацию и другую информацию для демона принтера, а также файл `printers.conf`, определяющий адреса и параметры настроенных принтеров. Файл `classes.conf` определяет локальные классы принтера.

Печать из системы Windows с помощью службы CUPS

Выполнить печать через CUPS можно и из систем, отличных от UNIX. Например, вы можете использовать драйвер принтера PostScript для печати непосредственно из системы Windows на сервер CUPS. Или применять CUPS без изменений, настроив компьютер Windows с помощью драйвера PostScript, который будет задействовать `printservername:631/printers/targetPrinter` как печатный порт.

Плюс ко всему вы можете использовать собственные драйверы принтера Windows для принтера вместо драйвера PostScript. Если собственный драйвер Windows не работает в очереди печати CUPS, можно включить печать без обработки данных Raw Print Queue. Такая печать непосредственно использует данные из собственного драйвера печати Windows и выводит ее на принтер.

Чтобы использовать службу CUPS, необходимо установить пакет `cups` в дистрибутиве Fedora или RHEL. Большинство настольных дистрибутивов Linux подключают службу CUPS во время начальной установки системы. Если служба не установлена в Fedora или RHEL, сделайте это, набрав следующую команду:

```
# yum install cups cups-client
```

Настройка принтеров

Обычно лучше всего пользоваться инструментами администрирования принтера, специально разработанными для вашего дистрибутива, однако многие системы Linux полагаются на инструменты, входящие в состав программного пакета CUPS.

В следующих разделах описано, как применять веб-инструменты администрирования CUPS. Затем будет рассмотрен инструмент настройки печати `system-config-printer`, доступный в системах Fedora. В некоторых случаях настройка не требуется, поскольку система может автоматически обнаруживать и настраивать подключенные принтеры. Чтобы установить инструмент Print Settings (Настройки принтера) в Fedora от имени суперпользователя, введите следующую команду `dnf` (или `yum`):

```
# yum install system-config-printer
```

Добавление принтера автоматически

Принтер CUPS можно настроить и автоматически транслировать в сеть, чтобы компьютер-клиент мог его обнаружить и использовать без настройки. Подключите USB-принтер к компьютеру, и он будет автоматически обнаружен и настроен.

На самом деле, если вы подключаете локальный принтер в дистрибутиве Fedora и драйвер печати еще не установлен, система предложит установить пакеты программного обеспечения, необходимые для использования принтера.

После первого перехода к печати документа или применения инструмента Print Settings (Настройки принтера) принтеры будут готовы к работе. Дальнейшую настройку можно выполнить с помощью веб-инструмента администрирования CUPS или окна Print Settings (Настройки принтера).

Веб-администрирование службы CUPS

Служба CUPS имеет собственный веб-инструмент администрирования для добавления, удаления и изменения конфигураций принтеров на вашем компьютере. Служба печати CUPS (с помощью демона `cupsd`) прослушивает порт 631 и обеспечивает доступ к веб-административному интерфейсу CUPS и совместному использованию принтеров.

Если служба CUPS уже запущена на вашем компьютере, можете сразу же задействовать веб-администрирование CUPS из своего браузера. Чтобы узнать, работает ли служба, и начать настройку принтеров, откройте браузер на локальном компьютере и введите следующее: `http://localhost:631/`.

Появится запрос на ввод имени пользователя и пароля для тех функций, которые этого требуют. В таком случае введите имя суперпользователя и пароль и нажмите кнопку OK. Появится такое окно, как на рис. 16.1.

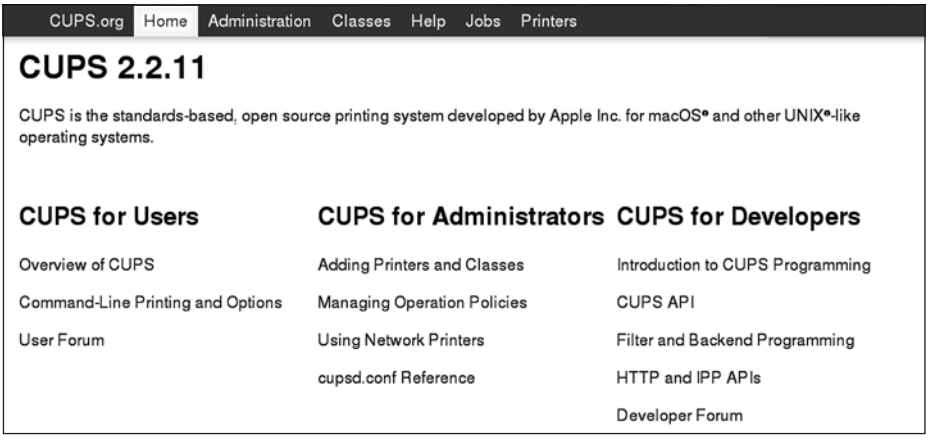


Рис. 16.1. Веб-администрирование CUPS

Удаленное администрирование печати

По умолчанию веб-администрирование CUPS доступно только с локального хоста. Чтобы получить доступ к веб-администрированию CUPS с другого компьютера, перейдите на главную страницу CUPS.

1. Выберите вкладку Administration (Администрирование).
2. Установите флажок Allow remote administration (Разрешить удаленное администрирование).
3. Нажмите кнопку Change Settings (Сохранить).

Затем откройте брандмауэр своего компьютера, чтобы разрешить подключение к TCP-порту 631 для доступа к службе. После этого из любого браузера в локальной сети вы можете получить доступ к странице администрирования CUPS, перейдя на порт 631 на сервере CUPS (например, `host.example.com:631`).

Возможно, потребуется перезапустить службу CUPS, чтобы изменения вступили в силу, командой `systemctl restart cups service`. Чтобы работать с браузером от имени суперпользователя, необходимо ввести имя суперпользователя и пароль.

Добавление принтеров

Чтобы настроить принтер, который не обнаруживается автоматически, добавьте его в окне Administration (Администрирование). Добавить принтер можно следующим образом.

1. Нажмите кнопку Add Printer (Добавить принтер). Появится соответствующее окно.

2. Выберите устройство, к которому подключен принтер. Он может быть подключен локально к параллельному порту, SCSI, последовательному порту или USB-порту непосредственно на компьютере. Кроме того, можете выбрать тип сетевого подключения для принтеров Apple (AppSocket или HP JetDirect), протокола интернет-печати (`http`, `https`, `ipps` или `ipp`) или Windows Printer (с использованием Samba или SMB).
3. Откроется окно, в котором необходимо ввести дополнительную информацию о подключении к принтеру. Например, может понадобиться указать сетевой адрес принтера IPP или Samba.
4. Введите название, расположение и описание принтера, выберите, хотите ли вы предоставить общий доступ к нему, и нажмите кнопку Continue (Продолжить).
5. Выберите марку принтера в списке Create (Создать). Если вы не видите в списке производителя вашего принтера, выберите PostScript для принтера PostScript или HP для принтера PCL. Для данного производителя можете указать конкретную модель.
6. Настройте принтер. Если нужно установить определенные параметры печати, сделайте это. Затем выберите пункт Set Printer Options (Сохранить параметры), чтобы продолжить.
7. Теперь новый принтер должен быть доступен. Если он успешно добавлен, щелкните на его имени, чтобы открылась новая страница. На ней можно выбрать вкладку Maintenance (Обслуживание), чтобы распечатать пробную страницу, или Administration (Администрирование), чтобы изменить параметры принтера.

Выполнив базовую настройку принтера, можно продолжать работу с принтерами. Варианты того, что можно сделать, следующие.

- **Просмотреть активные задания принтера.** Нажмите кнопку Show All Jobs (Показать активные задания), чтобы увидеть, какие задания печати в данный момент активны на любом из принтеров, настроенных для этого сервера. Нажмите кнопку Show Completed Jobs (Показать завершенные задания), чтобы просмотреть информацию об уже готовых заданиях.
- **Создать группу принтеров.** Перейдите на вкладку Administration (Администрирование), нажмите кнопку Add Class (Добавить группу) и укажите название, описание и расположение группы принтеров. В списке Printers (Members) (Состав группы) выберите принтеры на сервере, которые будут входить в нее.
- **Отменить или переместить задания.** Если вы по ошибке начали печать 100 страниц или принтер засорился, очень пригодится функция отмены заданий. А если отправили задание не на тот принтер, поможет функция перемещения заданий. На вкладке Administration (Администрирование) выберите пункт Manage Jobs (Управление заданиями), а затем нажмите кнопку Show Active Jobs (Показать все задания), чтобы увидеть, какие задания в данный момент находятся в очереди. Нажмите кнопку Cancel Job (Отменить задание) рядом с заданием, чтобы отменить его. Выберите кнопку Move Job (Переместить задание), чтобы переместить его на другой принтер.

- **Обзор принтеров.** Перейдите на вкладку Printers (Принтеры) в верхней части любой веб-страницы CUPS, чтобы увидеть все настроенные принтеры. Для каждого принтера можно выбрать задачи Maintenance (Обслуживание) или Administrative (Администрирование). В разделе Maintenance (Обслуживание) выберите вариант Pause Printer (Приостановить принтер), чтобы принтер остановил печать, но по-прежнему мог принимать задания в очередь. Вариант Reject Jobs (Не принимать задания) запрещает принтеру принимать новые задания в данный момент. Вариант Move All Jobs (Переместить все задания) перемещает все задания на другой принтер. Вариант Cancel All Jobs (Отменить все задания) удаляет все задания печати. Вариант Print Test Page (Печать пробной страницы) задает распечатку пробной страницы. На рис. 16.2 показана вкладка Printers (Принтеры) для конкретного принтера.

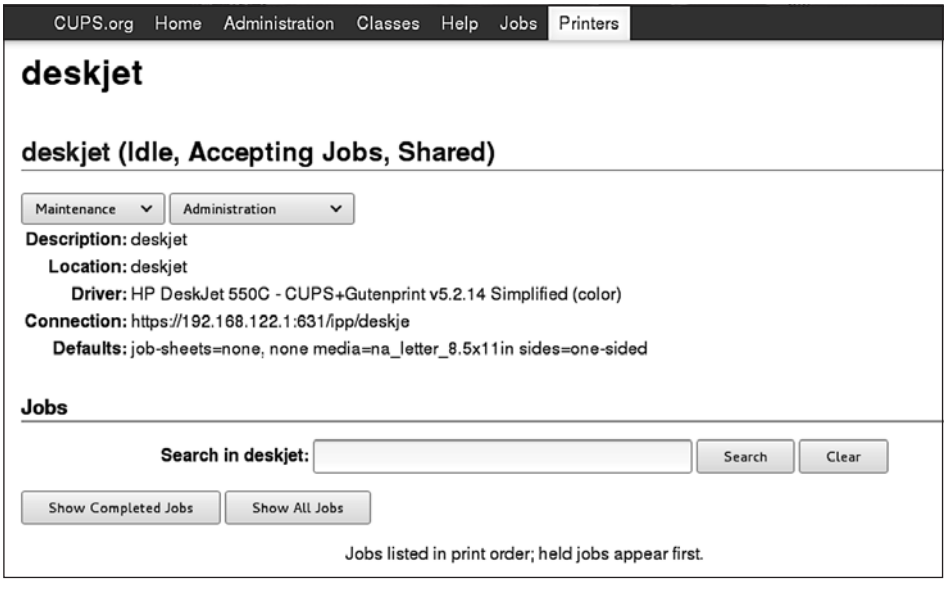


Рис. 16.2. На вкладке Принтеры производится управление принтером

Инструмент Настройки принтера (Print Settings)

В дистрибутиве Fedora настраивать принтеры можно в окне Print Settings (Настройки принтера). Я рекомендую использовать этот инструмент вместо веб-администрирования CUPS, поскольку файлы конфигурации принтера адаптируются для работы с тем, как служба CUPS запускается в системах в Fedora. После установки пакета (`dnf install system-config-printer`), чтобы установить принтер с рабочего стола GNOME, откройте окно Print Settings (Настройки принтера), введя Print Settings в строке поиска окна Acivities (Приложения) или, как суперпользователь, набрав команду `system-config-printer`. Этот инструмент

позволяет добавлять и удалять принтеры и редактировать их свойства. Он также позволяет отправлять на эти принтеры пробные страницы, чтобы убедиться, что они работают правильно.

Суть здесь заключается в том, что вы настраиваете принтеры, управляемые вашим демоном печати (`cupsd` для службы CUPS). После настройки принтера пользователи локальной системы могут работать с ним. Обратитесь к разделу «Настройка сервера печати», чтобы узнать, как сделать сервер доступным для пользователей с других компьютеров в вашей сети.

Принтеры могут быть подключены непосредственно к вашему компьютеру (например, через USB-порт) или к другому компьютеру в сети (например, из другой системы UNIX или Windows).

Настройка локальных принтеров в окне Настройки принтера

Добавьте локальный принтер (другими словами, принтер, подключенный непосредственно к компьютеру) в окно Printers (Принтеры), выполнив следующие действия.

Добавление локального принтера. Для добавления локального принтера с рабочего стола GNOME в последней версии системы Fedora проделайте следующее.

1. Чтобы открыть окно Print Settings (Настройки принтера), введите следующую команду:

   ```
   # system-config-printer &
   ```

   Появится соответствующее окно.
2. Нажмите кнопку Add (Добавить). (Нажмите кнопку Adjust Firewall (Настроить брандмауэр), чтобы разрешить доступ к порту принтера 631, если появится окно с запросом.) Появится окно New Printer (Новый принтер).
3. Если нужный принтер обнаружен автоматически, просто выберите его и нажмите кнопку Forward. Если же не обнаружен, выберите устройство, к которому подключен принтер (LPT #1 и последовательный порт #1 — это первые параллельный и последовательный порты соответственно), и нажмите кнопку Forward. (Введите команду `/usr/sbin/lpinfo-v | less` в оболочке, чтобы увидеть типы подключений принтера.) Окно предложит идентифицировать драйвер принтера.
4. Чтобы использовать установленный драйвер для принтера, выберите пункт Select Printer From Database (Выберите принтер из базы данных), а затем — производителя принтера. В качестве альтернативы можете выбрать вариант Provide PPD File (Предоставить PPD-файл) и добавить собственный PPD-файл (например, если у вас есть принтер, который не поддерживается в Linux, и есть драйвер, поставляемый вместе с принтером). PPD расшифровывается PostScript Printer Description. Нажмите кнопку Forward, чтобы просмотреть список моделей принтеров.

СОВЕТ

Если ваш принтер не отображается в списке, но поддерживает PCL (язык управления принтером HP), попробуйте выбрать один из принтеров HP (например, HP LaserJet). Если ваш принтер поддерживает PostScript, выберите PostScript printer из списка. Вариант Raw Print Queue позволяет отправлять уже отформатированные для конкретного принтера документы на определенный принтер.

5. Выбрав модель принтера, укажите драйвер, который будете использовать, а затем нажмите кнопку Forward, чтобы продолжить.
6. Добавьте следующую информацию и нажмите кнопку Forward:
   - Printer Name (Имя принтера). Добавьте имя, которое вы хотите присвоить принтеру. Имя должно начинаться с буквы, после нее могут стоять буквы, цифры, дефисы (-) и подчеркивания (_). Например, принтер HP на компьютере с именем `maple` можно назвать `hp-maple`;
   - Description (Описание). Добавьте несколько слов, описывающих принтер, например его функции (допустим, «HP LaserJet 2100M с поддержкой PCL и PS»);
   - Location (Расположение). Добавьте несколько слов, описывающих, где находится принтер (например, «В комнате 205 под кофеваркой»).
7. Добавив принтер, нажмите кнопку No (Нет) или Yes (Да), отвечая на предложение напечатать пробную страницу. Новый принтер появится в окне Print Settings (Настройки принтера). Дважды щелкните на нем, чтобы открыть окно его свойств Printer Properties (рис. 16.3).

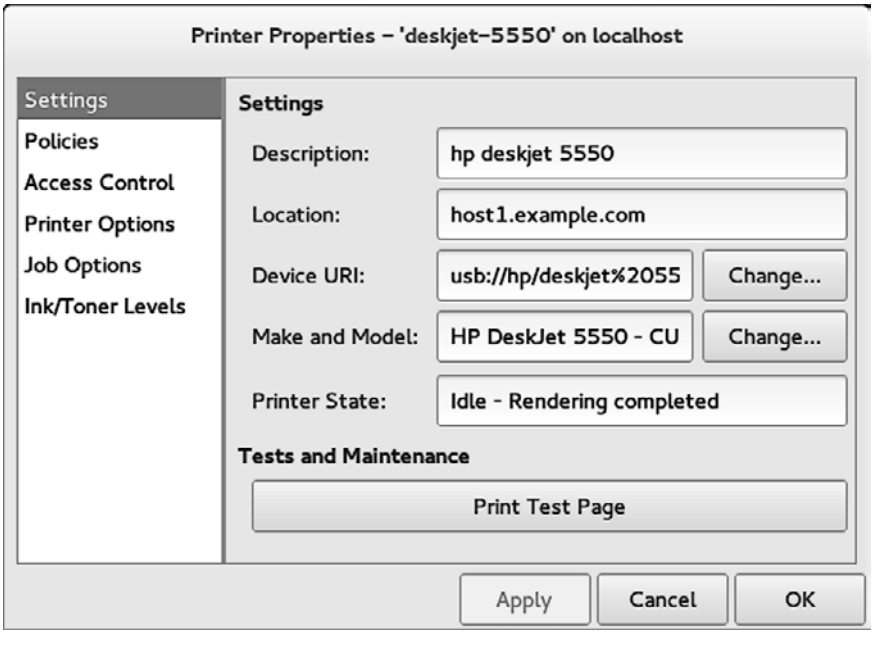


Рис. 16.3. Окно Printer Properties

8. Чтобы сделать устройство принтером по умолчанию, щелкните на нем правой кнопкой мыши и выберите Set As Default (Использовать по умолчанию). При добавлении других принтеров можете изменить принтер по умолчанию, выбрав нужный и снова указав Set As Default (Использовать по умолчанию).
9. Проверьте, работает ли принтер. Откройте окно Terminal (Терминал) и примените команду `lp` для печати файла (например, `lp /etc/hosts`). Чтобы узнать, как использовать этот принтер совместно с другими компьютерами в своей сети, обратитесь к разделу «Настройка сервера печати» далее в этой главе.

Настройка локального принтера. Дважды щелкнув на нужном принтере, выберите один из следующих пунктов меню, чтобы изменить его настройку.

- Settings (Параметры). В этом диалоговом окне отображаются настройки Description (Описание), Location (Размещение), Device URI (URI устройства), а также сведения Make and Model (Марка и модель), добавленные ранее.
- Policies (Политики). Щелкните на вкладке Policies (Политики), чтобы задать следующие настройки:
  - State (Состояние). Установите флажок Enabled (Разрешен), чтобы указать, будет ли принтер печатать задания, находящиеся в очереди. Пункт Accepting Jobs (Прием заданий) разрешает принтеру принимать новые задания для печати. Пункт Shared (Общий доступ) позволяет принтеру быть доступным для совместного применения другими компьютерами. Необходимо выбрать вариант Server Settings (Настройки сервера) и установить флажок Share Published printers connected to this system (Совместно использовать общедоступные принтеры, подключенные к этой системе), прежде чем принтер будет принимать задания печати с других компьютеров;
  - Policies (Политики). При ошибке можно выбрать вариант остановки принтера. Есть также вариант abort-job (отменить задание) или retry-job (повторить) в случае возникновения ошибки;
  - Banner (Заголовок). По умолчанию для принтера нет начальных и завершающих заголовков. Выберите нужный заголовок, который может содержать текст Classified, Confidential, Secret и др.
- Access Control (Управление доступом). Если ваш принтер является общим, перейдите в это окно, чтобы создать список пользователей, которым разрешен доступ к принтеру (всем остальным запрещен) либо он запрещен (всем остальным разрешен).
- Printer Options. Выберите вкладку Printer Options, чтобы установить значения по умолчанию параметров, связанных с драйвером принтера. Для разных принтеров доступны различные параметры. Многие из них могут быть переопределены при печати документа. Вот примеры вариантов, возможных в этом окне:
  - Watermark (Водяной знак). Вы можете добавлять и изменять водяные знаки на печатных страницах. По умолчанию водяной знак отключен. Выбрав Watermark (Водяной знак) (за текстом) или Overlay (Наложение) (над тек-

стом), можете установить дополнительные параметры. Водяные знаки могут быть на каждой странице — вариант All (Все) или только на первой странице — вариант First Only (Только первая страница). Выберите пункт Watermark Text (Текст водяного знака), чтобы выбрать текст для водяного знака или наложения (Draft (Черновик), Copy (Копия), Confidential (Конфиденциально), Final (Финальный) и др.). Затем можете выбрать тип шрифта, размер, стиль и непрозрачность водяного знака или наложения;

- Resolution Enhancement (Повышение разрешения). Можете оставить текущие настройки принтера или включить либо выключить повышение разрешения;
- Page Size (Размер страницы). По умолчанию используется американский размер страницы, но можно настроить принтер так, чтобы он печатал страницы в других вариантах из раскрывающегося списка;
- Media Source (Источник бумаги). Выберите лоток для печати. Укажите Tray 1 (Верхний лоток), чтобы вставить бумагу вручную;
- Levels of Gray (Оттенки серого). Выберите текущий, повышенный или стандартный уровень серого;
- Resolution (Разрешение). Выберите разрешение печати по умолчанию (например, 300, 600 или 1200 точек на дюйм). Более высокое разрешение обеспечивает лучшее качество печати, но занимает больше времени;
- EconoMode (Экономичный режим). Для принтера доступны три режима: текущий, экономный (экономия тонера) и режим максимального качества.

- Job Options (Параметры задания). Нажмите эту кнопку, чтобы задать общие параметры по умолчанию, которые будут использоваться для этого принтера. К ним относятся Common Options (Общие настройки) — количество копий, ориентация, масштаб, количество страниц на одном листе; Image Options (Параметры изображения) — масштаб, насыщенность, цветовой тон и гамма; Text Options (Параметры текста) — количество знаков на дюйм, линий на дюйм и параметры полей.
- Ink/Toner Levels (Уровни чернил/тонера). Выберите эту вкладку, чтобы просмотреть информацию о том, сколько чернил или тонера осталось в принтере. (Не все принтеры сообщают эти значения.)

Нажмите кнопку Apply, внося нужные изменения.

Настройка удаленных принтеров

Чтобы использовать принтер, доступный в сети, вы должны добавить его к своей системе Linux. Система поддерживает удаленные подключения принтера: Networked CUPS (IPP), принтеры, принтеры Networked UNIX (LPD), принтеры Networked Windows (Samba) и принтеры JetDirect. (Конечно, и CUPS, и Unix-серверы печати могут быть запущены как из систем Linux, так и из других UNIX-подобных систем.)

Во всех случаях необходимо сетевое подключение вашей системы Linux к серверам, к которым подключены эти принтеры. Для использования удаленного принтера требуется, чтобы он был настроен на удаленном сервере. См. раздел «Настройка сервера печати» далее в этой главе, чтобы узнать, как это сделать на вашем сервере Linux.

С помощью окна Print Settings (Настройки принтера) (`system-config-printer`) можно настроить подключение к удаленным принтерам. Вот как это делается.

1. В окне Acivities (Приложения) в GNOME 3 в поле поиска введите `Print Settings` и нажмите клавишу Enter.
2. Нажмите кнопку Add (Добавить). Появится окно New Printer (Новый принтер).
3. В зависимости от типов портов, поддерживаемых на вашем компьютере, выберите один из вариантов:
   - LPT #1. Используйте этот вариант для принтера, подключенного к параллельному порту;
   - Serial Port #1. Применяйте для принтера, подключенного к последовательному порту;
   - Network Printer. В этом варианте можно выполнить поиск сетевых принтеров (по имени хоста или IP-адресу) или ввести URI для различных типов принтеров:
     а) Find Network Printer. Вместо ввода URI принтера можно указать имя хоста или IP-адрес системы, где установлен принтер, через который нужно выполнить печать. Все принтеры, найденные на этом хосте, появятся в окне;
     б) AppleSocket/HP JetDirect. Вариант для принтера JetDirect;
     в) Протокол интернет-печати (Internet Printing Protocol, IPP). Используйте этот вариант для принтера CUPS или другого IPP-принтера. Большинство принтеров Linux и Mac OS X относятся к этой категории;
     г) Протокол интернет-печати (Internet Printing Protocol, HTTPS). Применяйте этот вариант для CUPS или другого IPP-принтера, используемого по защищенному соединению (потребуются действительные сертификаты);
     д) Хост или принтер LPD/LPR (LPD/LPR Host or Printer). Вариант для принтера UNIX;
     е) Windows Printer via SAMBA. Вариант для системного принтера Windows.

Продолжайте выполнять действия в любом из подходящих разделов.

Добавление удаленного принтера CUPS

Если вы решили добавить принтер CUPS (IPP), доступный по локальной сети из окна Print Settings (Настройки принтера), то в появившемся окно необходимо добавить следующую информацию.

- Host (Сервер). Это имя хоста компьютера, к которому подключен принтер (или к которому можно получить доступ другим способом). Это может быть IP-адрес или имя хоста TCP/IP для компьютера. Имя TCP/IP доступно в файле `/etc/hosts` или через DNS-сервер имен.
- Queue (Очередь). Имя принтера на удаленном сервере печати CUPS. CUPS поддерживает варианты принтеров, что позволяет каждому из них иметь несколько наборов параметров. Если удаленный принтер CUPS настроен таким образом, можете выбрать определенный путь к нему, например `hp/300dpi` или `hp/1200dpi`. Символ косой черты (`/`) отделяет имя очереди печати от варианта принтера.

Выполните остальные настройки так же, как и для локального принтера (см. подпункт «Добавление локального принтера» ранее в этой главе).

Добавление удаленного принтера UNIX (LDP/LPR)

Если вы решили добавить принтер UNIX (LPD/LPR) из окна Print Settings (Настройки принтера), то в появившемся окне необходимо указать следующую информацию.

- Host (Сервер). Это имя хоста компьютера, к которому подключен принтер (или к которому можно получить доступ другим способом). Это может быть IP-адрес или имя хоста TCP/IP для компьютера. Имя TCP/IP доступно в файле `/etc/hosts` или через DNS-сервер имен. Нажмите кнопку Probe (Датчик), чтобы найти сервер.
- Queue (Очередь). Имя принтера на удаленном компьютере UNIX.

Выполните остальные настройки так же, как и для локального принтера (см. подпункт «Добавление локального принтера» ранее в этой главе).

СОВЕТ

Если задание печати, отправленное для проверки принтера, отклонено, сервер печати, возможно, не разрешил вам доступ к принтеру. Попросите администратора удаленного компьютера добавить ваше имя хоста в файл /etc/lpd.perms. (Введите lpstat-d printer, чтобы увидеть состояние задания печати.)

Добавление принтера Windows (SMB)

Подключение компьютера для доступа к принтеру SMB (служба печати Windows) включает в себя добавление записи для принтера в окне Select Connection (Выбор подключения).

При добавлении принтера Windows (через службу Samba) в окне Print Settings (Настройки принтера) нажмите кнопку Browse (Обзор), чтобы просмотреть

список компьютеров в вашей сети, которые относятся к службе SMB (файловые и/или печатные службы). В этом окне можно настроить принтер следующим образом.

1. Введите URI принтера без `smb://`. Например, можно набрать `/host1/myprinter` или `/mygroup/host1/myprinter`.
2. Выберите один из вариантов: Prompt user if authentication is required (Запрашивать данные аутентификации пользователя) или Set authentication details now (Ввести данные аутентификации сейчас).
3. При выборе Set authentication details now (Ввести данные аутентификации сейчас) введите данные о пароле и имени пользователя, затем нажмите кнопку Verify (Проверить), чтобы проверить, сможете ли войти на сервер.
4. Нажмите кнопку Forward.

Кроме того, вы можете определить сервер, который не отображается в списке серверов. Введите информацию, необходимую для создания принтера URI SMB.

- Workgroup (Рабочая группа). Имя рабочей группы Samba, в которую входит общий принтер. Использовать рабочую группу не обязательно.
- Server (Сервер). Имя NetBIOS или IP-адрес компьютера, который может совпадать или не совпадать с его именем TCP/IP. Чтобы перевести это имя в адрес, необходимый для доступа к SMB-хосту, Samba проверяет определенные файлы, в которых это имя может быть назначено IP-адресу. Samba проверяет следующее (в указанном порядке), пока не найдет совпадение: локальный файл `/etc/hosts`, локальный файл `/etc/lmhosts`, WINS-сервер в сети — и отвечает на широковещательные трансляции на каждом локальном сетевом интерфейсе.
- Share (Общий ресурс). Имя общего принтера, который используется совместно с удаленным компьютером. Оно может отличаться от имени, применяемого локальными пользователями.
- User name (Имя пользователя). Чтобы получить доступ к принтеру SMB, серверу SMB требуется имя пользователя. Оно не нужно, если вы аутентифицируете принтер на основе общего доступа, а не на уровне пользователя. С помощью уровня общего доступа можете добавить пароль для каждого общего принтера или файловой системы.
- Password (Пароль). Применяйте пароль, связанный с именем пользователя SMB или общим ресурсом, в зависимости от типа контроля доступа.

ВНИМАНИЕ!

Когда вы вводите имя пользователя и пароль для SMB, информация сохранится в незашифрованном виде в файле /etc/cups/printers.conf. Убедитесь, что файл доступен только суперпользователю.

Далее приведен пример URI SMB, который можно добавить в поле SMB://:

```
jjones:my9passswd@FSTREET/NS1/hp
```

URI, показанный здесь, идентифицирует имя пользователя (jjones), пароль пользователя (my9passswd), рабочую группу (FSTREET), сервер (NS1) и имя очереди принтера (hp).

Выполните остальную часть настройки так же, как и для локального принтера (см. раздел «Добавление локального принтера» ранее в этой главе).

Если все настроено правильно, вы можете использовать стандартную команду lp для печати файла на принтере. Задействуйте для печати следующую команду:

```
$ cat file1.ps | lp -P NS1-PS
```

СОВЕТ

Получив сообщение об ошибке, убедитесь, что компьютер, на котором выполняется печать, доступен. Например, для принтера NS1 hp введите команду smbclient -L NS1 -U jjones, а затем пароль (в данном случае my9passswd). Параметр -L запрашивает информацию о сервере, параметр -U jjones просит войти в систему от имени пользователя jjones. Получив положительный ответ на запрос имени после ввода пароля, вы должны увидеть список общих принтеров и файлов с этого сервера. Проверьте имена и повторите печать.

Печать с помощью CUPS

Такие инструменты, как веб-администрирование CUPS и окно Print Settings (Настройки принтера), эффективно скрывают все возможности службы CUPS. Однако может потребоваться обратиться непосредственно к инструментам и файлам конфигурации службы. В следующих разделах описано, как использовать специальные функции CUPS.

Настройка сервера CUPS (cupsd.conf)

Демон cupsd прослушивает запросы к вашему серверу печати CUPS и отвечает на них, основываясь на настройках, сохраненных в файле /etc/cups/cupsd.conf. Переменные конфигурации в файле cupsd.conf имеют ту же форму, что и в файле конфигурации Apache (httpd.confor apache2.conf). Введите команду man cupsd.conf, чтобы просмотреть подробную информацию о любых настройках.

Окно Print Settings (Настройки принтера) добавляет информацию о доступе к файлу cupsd.conf. Для других систем Linux или при отсутствии рабочего стола на сервере вам может потребоваться настроить файл cupsd.conf вручную. Перейдите к файлу cupsd.conf, чтобы настроить свой сервер CUPS. Большинство настроек

необязательны или могут быть оставлены применяемые по умолчанию. Рассмотрим настройки, которые можно использовать в файле `cupsd.conf`.

По умолчанию классификация не задается. С классификацией, `topsecret`, вы можете отображать надпись `Top Secret` на всех страницах, проходящих через сервер печати:

```
Classification topsecret
```

Варианты классификации, кроме `topsecret`: `classified`, `confidential`, `secret` и `unclassified`.

Термин *browsing* относится к процессам передачи информации о вашем принтере в локальной сети и прослушивания информации других серверов печати. Параметр `cups-browsed` используется для просмотра общих удаленных принтеров. Просмотр включен по умолчанию для всех локальных сетей (`@LOCAL`). Вы можете разрешить передавать информацию браузера CUPS (`BrowseAllow`) на другие адреса. Информация о просмотре по умолчанию передается по адресу 255.255.255.255. Вот примеры нескольких настроек просмотра:

```
Browsing On
BrowseProtocols cups
BrowseOrder Deny,Allow
BrowseAllow from @LOCAL
BrowseAddress 255.255.255.255
Listen *:631
```

Чтобы включить веб-администрирование CUPS и использовать принтеры совместно с другими пользователями сети, демон `cupsd` настраивается на прослушивание через порт 631 всех сетевых интерфейсов вашего компьютера на основе строки `Listen *:631`. По умолчанию демон прослушивает локальные интерфейсы только в системах Linux (`Listen localhost:631`). В системе Fedora служба CUPS по умолчанию прослушивает все интерфейсы.

Это хороший способ дать пользователям нескольких подключенных локальных сетей возможность находить и применять принтеры в соседних локальных сетях.

Вы можете разрешить или запретить доступ к различным функциям сервера CUPS. Настройка доступа для принтера CUPS в окне `Print Settings` (Настройки принтера) может выглядеть следующим образом:

```
<Location /printers/ns1-hp1>
Order Deny,Allow
Deny From All
Allow From 127.0.0.1
AuthType None
</Location>
```

В примере печать на принтере `ns1-hp1` разрешена только пользователям локального хоста (127.0.0.1). Пароль не требуется (`AuthType None`). Чтобы разрешить доступ к инструменту администрирования, служба CUPS должна запрашивать пароль (`AuthType Basic`).

Запуск сервера CUPS

Для систем Linux, применяющих скрипт запуска в стиле System V (например, более ранние версии дистрибутивов Fedora и RHEL), запуск и выключение службы печати CUPS довольно просты. Используйте команду `chkconfig`, чтобы служба запускалась при каждой перезагрузке. Введите скрипт `cups`, чтобы служба CUPS запустилась автоматически. В дистрибутиве RHEL 6. x или более ранней версии введите от имени суперпользователя следующее:

```
# chkconfig cups on
# service cups start
```

Если служба CUPS уже запущена, возьмите параметр `restart` вместо `start`. Параметр `restart` — это хороший способ прочитать все параметры конфигурации, измененные в файле `cupsd.conf` (хотя, если служба CUPS уже запущена, команда `service cups reload` перечитывает файлы конфигурации без перезапуска).

В дистрибутивах Fedora 30 и RHEL 8 для запуска и остановки служб вместо `service` используется команда `systemctl`:

```
# systemctl status cups.service

* cups.service – CUPS Printing Service
  Loaded: loaded (/usr/lib/systemd/system/cups.service; enabled)
  Active: active (running) since Sat 2016-07-23 22:41:05 EDT; 18h ago
Main PID: 20483 (cupsd)
  Status: "Scheduler is running..."
  CGroup: /system.slice/cups.service
          ├─ 20483 /usr/sbin/cupsd -f
```

В примере видно, что служба CUPS запущена, так как статус показывает, что демон `cupsd` активен с идентификатором PID 20483. Запустить службу CUPS можно следующим образом (если она не запущена):

```
# systemctl start cups.service
```

Дополнительную информацию о командах `systemctl` и `service` для работы со службами см. в главе 15 «Запуск и остановка служб».

Настройка принтера CUPS вручную

Если в вашем дистрибутиве Linux нет графических средств настройки CUPS, можете редактировать файлы конфигурации напрямую. Так, добавленный в окне Print Settings (Настройки принтера) принтер определяется в файле `/etc/cups/printers.conf`. Вот как выглядит запись о принтере:

```
<DefaultPrinter printer>
Info HP LaserJet 2100M
Location HP LaserJet 2100M in hall closet
DeviceURI parallel:/dev/lp0
```

```
State Idle
Accepting Yes
Shared No
JobSheets none none
QuotaPeriod 0
PageLimit 0
KLimit 0
</Printer>
```

Это пример локального принтера, который является принтером по умолчанию для локальной системы. В поле `Shared No` значений нет, так как он доступен только в локальной системе. Наиболее интересная информация находится в поле `DeviceURI`, которое отображает, что принтер подключен к параллельному порту `/dev/lp0`. Состояние неактивно — `Idle` (готов принимать задания), а в поле `Accepting` установлено значение `Yes` (принтер принимает задания печати по умолчанию).

Поле `DeviceURI` может определить имя устройства несколькими способами, отображая местоположение принтера. Вот несколько примеров, перечисленных в файле `printers.conf`:

```
DeviceURI parallel:/dev/plp
DeviceURI serial:/dev/ttyd1?baud=38400+size=8+parity=none+flow=soft
DeviceURI scsi:/dev/scsi/sc1d6l0
DeviceURI usb:// имя_устройства:порт
DeviceURI socket:// имя_устройства:порт
DeviceURI tftp:// имя_устройства/путь
DeviceURI ftp:// имя_устройства/путь
DeviceURI http:// имя_устройства [:порт]/путь
DeviceURI ipp:// имя_устройства/путь
DeviceURI smb:// имя_устройства/принтер
```

Первые четыре примера показывают варианты для локальных принтеров (`parallel`, `serial`, `scsi` и `usb`). Другие примеры относятся к удаленным хостам. Во всех случаях имя хоста может быть именем хоста или IP-адресом. Номера портов или пути определяют место расположения каждого принтера на хосте. Например, параметр `hostname` может быть `myhost.example.com:63`, а параметр `path` может быть заменен на любое имя, например `printers/myprinter`.

Команды печати

Чтобы сохранить обратную совместимость с устаревшими средствами печати UNIX и Linux, служба CUPS поддерживает многие команды для работы с печатью. Бóльшую часть функций службы CUPS из командной строки можно выполнить с помощью команды `lp`. Приложения обработки текстов, такие как LibreOffice, OpenOffice и AbiWord, настроены на применение этой команды для печати.

Используйте окно Print Settings (Настройки принтера), чтобы определить необходимые для каждого принтера фильтры, благодаря которым текст можно правильно отформатировать. Параметры команды `lp` могут добавлять фильтры

для правильной обработки текста. Другие команды для управления печатными документами — это `lpq` (для просмотра содержимого очередей печати), `lprm` (для удаления заданий печати из очереди) и `lpstat -t` (для управления принтерами).

Печать с помощью команды lp

Используйте команду `lp` для печати документов как на локальных, так и на удаленных принтерах (при условии, что принтеры настроены локально). Файлы документов могут быть либо добавлены в конец командной строки `lp`, либо направлены в команду `lp` с помощью канала (`|`). Пример простой команды `lp`:

```
$ lp doc1.ps
```

Если указан только файл документа с командой `lp`, вывод будет направлен на принтер по умолчанию. Как обычный пользователь, вы можете изменить принтер по умолчанию, установив новое значение для переменной `PRINTER`. Обычно переменная `PRINTER` добавляется в один из загрузочных файлов, например `$HOME/.bashrc`. Если добавить приведенную далее строку в файл `.bashrc`, то принтеру по умолчанию будет установлена команда `lp3`:

```
export PRINTER=lp3
```

Чтобы переопределить принтер по умолчанию, укажите конкретный принтер в командной строке `lp`. В следующем примере используется параметр `-P` для выбора другого принтера:

```
$ lp -P canyonps doc1.ps
```

Команда `lp` имеет множество параметров, которые позволяют ей интерпретировать и форматировать несколько различных типов документов. К ним относятся параметр `-# num`, где `num` заменяется количеством копий для печати (от 1 до 100), и параметр `-l`, который отправляет документ необработанным, предполагая, что тот уже отформатирован. Чтобы узнать, какие еще параметры подходят для команды `lp`, введите команду `man lp`.

Вывод состояний с помощью команды lpstat -t

Используйте команду `lpstat -t` для отображения состояния ваших принтеров, например:

```
$ /usr/sbin/lpstat -t
printer hp disabled since Wed 10 Jul 2019 10:53:34 AM EDT
printer deskjet-555 is idle. enabled since Wed 10 Jul 2019
10:53:34 AM EDT
```

В выводе примера показаны два активных принтера. Принтер `hp` сейчас отключен (неактивен). Принтер `deskjet-555` включен.

Удаление задач печати с помощью команды lprm

Пользователи могут удалять свои задания печати из очереди с помощью команды `lprm`. Применяемая только в командной строке, команда `lprm` удаляет все задания печати пользователя с принтера по умолчанию. Чтобы удалить задания с определенного принтера, применяйте параметр `-P` следующим образом:

```
$ lprm -P lp0
```

Чтобы удалить все задания печати текущего пользователя, введите следующее:

```
$ lprm -
```

Суперпользователь может удалить все задания печати конкретного пользователя, указав его в командной строке `lprm`. Например, чтобы удалить все задания печати пользователя `mike`, суперпользователь вводит следующие данные:

```
# lprm – U mike
```

Чтобы удалить определенное задание печати из очереди, укажите его номер в командной строке `lprm`. Чтобы найти номер задания, введите команду `lpq`. Вот как может выглядеть вывод этой команды:

```
# lpq
printer is ready and printing
Rank    Owner               Job Files                 Total Size Time
active  root                133 /home/jake/pr1          467
2       root                197 /home/jake/mydoc      23948
```

Здесь видны два задания на печать, ожидающих в очереди. (Принтер готов и печатает задание, указанное в списке активных.) В столбце `Job` есть номер задания, связанный с каждым документом. Чтобы удалить первое задание на печать, введите следующее:

```
# lprm 133
```

Настройка сервера печати

Итак, вы настроили принтер таким образом, чтобы и сами, и другие пользователи вашего компьютера могли печатать на нем. А теперь хотите поделиться этим принтером с другими пользователями в доме, школе или офисе. Время переходить к настройке принтера в качестве сервера печати.

Принтеры, настроенные в системе Linux, могут совместно использоваться другими компьютерами, относящимися к одной сети. Ваш компьютер может выступать не только в качестве сервера печати Linux (настройка службы CUPS), но и как сервер печати SMB (Windows) для клиентских компьютеров. После подсоединения локального принтера к системе Linux и подключения компьютера к локальной сети можно настроить систему для совместного использования принтера с клиентскими

компьютерами с помощью интерфейса Linux (UNIX) или SMB. Как это сделать, будет описано далее в этой главе.

Настройка общего принтера CUPS

Сделать локальный принтер доступным для других компьютеров в общей сети довольно просто. Если между компьютерами, совместно использующими принтер, существует сетевое соединение TCP/IP, нужно просто предоставить права всем хостам, отдельным хостам или пользователям с удаленных хостов на доступ к службе печати вашего компьютера.

Чтобы настроить принтер на прием заданий печати с других компьютеров вручную в файле `/etc/cups/printers.conf`, убедитесь, что в нем есть строка `Shared Yes`. В следующем примере из записи `printers.conf`, приведенной ранее в этой главе, создана новая:

```
<DefaultPrinter printer>
Info HP LaserJet 2100M
Location HP LaserJet 2100M in hall closet
DeviceURI parallel:/dev/lp0
State Idle
Accepting Yes
Shared Yes
JobSheets none none
QuotaPeriod 0
PageLimit 0
KLimit 0
</Printer>
```

В системах Linux, использующих окно Print Settings (Настройки принтера), описанное ранее в этой главе, эффективнее всего настроить принтер в качестве общего с его помощью.

Вот как это сделать в дистрибутиве Fedora 30.

1. На экране Acivities (Приложения) на рабочем столе GNOME 3 в Fedora в поле поиска введите `Print Settings` и нажмите клавишу Enter. Появится окно Print Settings (Настройки принтера).
2. Чтобы разрешить общий доступ ко всем принтерам, выберите Server ▸ Settings (Сервер ▸ Параметры). Если вы находитесь в системе как обычный пользователь, понадобится ввести пароль суперпользователя. Появится окно Basic Server Settings (Основные настройки сервера).
3. Установите флажок Share Published printers connected to this system (Показывать общие принтеры, подключенные к этой системе) и нажмите кнопку OK. После этого может появиться диалоговое окно с предложением изменить настройки межсетевого экрана.
4. Чтобы разрешить или ограничить печать для конкретного принтера, дважды щелкните на имени принтера, к которому хотите предоставить общий доступ.

(Если принтер еще не настроен, обратитесь к разделу «Настройка принтеров» ранее в этой главе.)

5. Перейдите на вкладку Policies (Политики) и установите флажок Shared (Общий доступ).
6. Если вы хотите ограничить доступ к принтеру конкретным пользователям, перейдите на вкладку Access Control (Управление доступом) и выберите один из следующих параметров:
   - Allow Printing for Everyone Except These Users (Разрешить печатать всем, кроме указанных пользователей). Если выбран этот параметр, доступ к принтеру разрешен всем пользователям. Введя имена пользователей в поле Users (Пользователи) и нажав кнопку Add (Добавить), вы запретите им доступ к принтеру;
   - Deny Printing for Everyone Except These Users (Запретить печатать всем, кроме этих пользователей). Если выбран этот параметр, всем пользователям запрещен доступ к принтеру. Введя имена пользователей в поле Users (Пользователи) и нажав кнопку Add (Добавить), вы разрешите им доступ к принтеру.

Теперь можете настроить другие компьютеры для доступа к своему принтеру, как описано в разделе «Настройка принтеров» в этой главе. Если попытка выполнить печать с другого компьютера окажется неудачной, воспользуйтесь следующими советами по устранению неполадок.

- **Откройте брандмауэр.** Ограничительный межсетевой экран может не разрешить печать. Необходимо включить доступ к TCP-порту 631, чтобы разрешить доступ к печати на вашем компьютере.
- **Проверьте имена и адреса устройств.** Убедитесь, что вы правильно ввели имя своего компьютера и очередь на печать при настройке его на другом компьютере. Попробуйте использовать IP-адрес вместо имени хоста. (Если это сработает, значит, проблема была в DNS-именах.) Запустите инструмент `tcpdump`, который позволяет увидеть, где именно происходит сбой.
- **Проверьте, какие адреса прослушивает cupsd.** Демон `cupsd` должен прослушивать данные за пределами локального хоста, чтобы удаленные системы могли печатать через него. Примените команду `netstat` (от имени суперпользователя), как показано далее, чтобы проверить это. Первый пример показывает, что демон `cupsd` прослушивает данные только на локальном хосте (27.0.0.1:631), второй отображает прослушивание на всех сетевых интерфейсах (0 0.0.0.0:631):

```
# netstat -tupln | grep 631
tcp        0      0 127.0.0.1:631     0.0.0.0:*         LISTEN
6492/cupsd
# netstat -tupln | grep 631
tcp        0      0 0.0.0.0:631       0.0.0.0:*         LISTEN
6492/cupsd
```

Изменения доступа к общему принтеру вносятся в файлы `cupsd.conf` и `printers.conf` в каталоге `/etc/cups`.

Настройка общего принтера Samba

Принтеры Linux можно настроить как общие SMB-принтеры, которые будут доступны из систем Windows. Чтобы предоставить общий доступ к принтеру Samba (SMB), просто настройте основные параметры сервера, как описано в главе 19 «Настройка Samba-сервера». По умолчанию все ваши принтеры будут совместно использоваться в локальной сети. В следующем разделе показано, как выглядят настройки и как их можно изменить.

Настройка файла печати smb.conf

При настройке сервера Samba создается файл `/etc/samba/smb.conf`, позволяющий совместно использовать все настроенные принтеры. Вот пример нескольких строк из файла `smb.conf`, относящихся к общему доступу к принтерам:

```
[global]
        ...
    load printers = yes
    cups options = raw
    printcap name = /etc/printcap
    printing = cups
        ...
[printers]
            comment = All Printers
            path = /var/spool/samba
            browseable = yes
            writeable = no
            printable = yes
```

Прочитайте комментарии в строках, чтобы больше узнать о содержимом файла. Строки, начинающиеся с точки с запятой (`;`), указывают на значение параметра по умолчанию в строке комментария. Удалите точку с запятой, чтобы изменить настройку.

Строки из примера показывают, что принтеры из файла `/etc/printcap` загружены и используется служба CUPS. Если для параметра `cup soptions` задано значение `raw`, Samba предполагает, что файлы печати будут отформатированы к тому времени, как достигнут сервера печати. Это позволяет клиентам Linux или Windows применять собственные драйверы печати.

Последние несколько строк — это фактическое определение принтеров. Изменив параметр `browseable` с `no` на `yes`, вы даете пользователям возможность печатать на всех принтерах (`printable = yes`). Можете также хранить собственные драйверы печати Windows на своем сервере Samba. Когда клиент Windows задействует ваш принтер, нужный драйвер подключается автоматически — нет необходимости загружать драйвер с сайта поставщика. Чтобы включить общий ресурс драйвера принтера, добавьте общий ресурс Samba с именем `print$`, который выглядит следующим образом:

```
[print$]
comment = Printer Drivers
path = /var/lib/samba/drivers
```

```
browseable = yes
guest ok = no
read only = yes
write list = chris, dduffey
```

После того как общий ресурс станет доступным, можно начать копирование драйверов печати Windows в каталог `/var/lib/samba/drivers`.

Настройка SMB-клиентов

Настраивая принтер Samba на своем компьютере Linux, вы, вероятно, захотите поделиться им с клиентами системы Windows. Если служба Samba правильно настроена на вашем компьютере и клиентские компьютеры могут связаться с ней по сети, у пользователей не должно возникнуть проблем с поиском и применением вашего принтера.

В системах Windows 10 из меню Пуск (Start) перейдите в настройки Принтеры и сканеры (Printers and Scanners) и выберите принтер из списка, чтобы настроить его.

В системе Windows Vista щелкните на ярлыке Сеть (Network). Имя вашего хост-компьютера (имя NetBIOS, которое, вероятно, является также именем TCP/IP) появится на экране или в папке рабочей группы на нем. Щелкните на значке, представляющем ваш компьютер. В открывшемся окне отобразятся общие принтеры и папки.

СОВЕТ

Если значок вашего компьютера не отображается в разделе Сетевое окружение (Network Neighborhood или My Network Places), попробуйте воспользоваться поиском. В системе Windows XP выберите Пуск (Start) ▸ Найти программы и файлы (Search) ▸ Компьютеры или пользователи (Computer or People) ▸ Компьютер в сети (A Computer on the Network). Введите имя своего компьютера в поле Имя компьютера (Computer Name) и нажмите кнопку Поиск (Search). Дважды щелкните на значке компьютера в результатах поиска. Появится окно с общими принтерами и папками, имеющимися на вашем компьютере.

После того как общий принтер появится в окне, настройте указатель на него, дважды щелкнув на значке принтера. Появится сообщение о том, что перед использованием принтера его необходимо настроить. Нажмите кнопку Да (Yes), чтобы продолжить настройку принтера для локального применения. Появится мастер Add Printer Wizard. Ответьте на вопросы о том, как вы планируете использовать принтер, и добавьте соответствующие драйверы. По окончании настройки принтер появится в окне принтеров.

Еще один способ настроить SMB-принтер из операционной системы Windows XP — это перейти в меню Start (Пуск) и найти раздел Printers and Faxes (Принтеры и факсы). В появившемся окне щелкните на значке Add a Printer (Добавить принтер) в левой верхней части окна и выберите вариант Network Printer (Сетевой принтер) в первом окне. Там вы можете просмотреть и/или настроить свой SMB-принтер.

Резюме

Печать по сети имеет важное значение в современных бизнес-сетях. Используя несколько подключенных к сети устройств, вы можете сосредоточиться на нескольких высококачественных принтерах, с которыми могут работать несколько человек, а не на многочисленных более дешевых устройствах. Кроме того, принтер, с которым работают несколько пользователей, облегчает собственное обслуживание, в то же время позволяя всем выполнять свои задачи в сфере печати.

Сегодня почти в каждом крупном дистрибутиве Linux служба печати по умолчанию — это общая система печати UNIX (CUPS). Любая система Linux, использующая службу CUPS, имеет веб-интерфейс администрирования CUPS для настройки печати CUPS, а также файлы конфигурации в каталоге `/etc/cups` для настройки принтеров и службы CUPS (демон `cupsd`).

В RHEL, Fedora, Ubuntu и других системах Linux можно настроить принтер с помощью окон конфигурации печати, доступных как на настольных компьютерах KDE, так и на настольных компьютерах GNOME. Разнообразие драйверов позволяет печатать на различных типах принтеров, а также на принтерах, подключенных к компьютерам в сети.

Вы можете настроить свой компьютер как сервер печати Linux, а также эмулировать сервер печати SMB (Windows). Когда сеть настроена правильно и локальный принтер установлен, можно совместно использовать его по сети в качестве сервера печати UNIX или SMB.

Упражнения

Выполните эти упражнения, чтобы проверить свои знания о настройке принтеров в Linux. Задачи подходят для систем Fedora или Red Hat Enterprise Linux (некоторые сработают и в других системах Linux). Если затрудняетесь с решением заданий, воспользуйтесь ответами к упражнениям, приведенными в приложении Б (хотя Linux позволяет решать задачи разными способами).

1. Используя окно Print Settings (Настройки принтера) из пакета `system-config-printer`, добавьте новый принтер с именем `myprinter` к вашей системе. (Новый принтер не обязательно подключать, чтобы настроить для него очередь печати.) Подключите принтер PostScript к локальному последовательному порту, LPT или другому.
2. С помощью команды `lpstat -t` просмотрите состояние всех ваших принтеров.
3. С помощью команды `lp` распечатайте файл `/etc/hosts` на новом принтере.
4. Проверьте очередь печати для этого принтера, чтобы отобразить задание печати.
5. Удалите задание печати из очереди (отмените его).

6. В окне Printing (Печать) задайте параметры базового сервера, который отображает ваши принтеры, чтобы другие системы в вашей локальной сети могли печатать на них.
7. Разрешите удаленное администрирование своей системы с помощью браузера.
8. Выполните удаленное администрирование своей системы из другой, открыв браузер на порте 631, где находится ваш сервер печати.
9. Используйте команду `netstat`, чтобы узнать, по каким адресам демон `cupsd` прослушивает данные (порт печати 631).
10. Удалите принтер `myprinter` из своей системы.

17 Настройка веб-сервера

В этой главе

- Установка веб-сервера Apache.
- Настройка веб-сервера Apache.
- Защита веб-сервера Apache с помощью iptables и SELinux.
- Создание виртуальных хостов.
- Создание защищенного (HTTPS) сайта.
- Проверка веб-сервера Apache на наличие ошибок.

Веб-серверы отвечают за обслуживание содержимого, которое вы просматриваете в Интернете каждый день. Однозначно, самый популярный веб-сервер — это Apache (файл `httpd`), веб-сервер, который спонсируется фондом Apache Software Foundation (apache.org). Поскольку Apache является проектом с открытым исходным кодом, он доступен во всех основных дистрибутивах Linux, включая Fedora, RHEL и Ubuntu.

При этом вы можете использовать свой веб-сервер Apache для решения множества задач. Например, для обслуживания содержимого нескольких доменов (виртуальный хостинг), обеспечения зашифрованной связи (HTTPS) и защиты некоторых или всех сайтов с помощью различных видов аутентификации.

В этой главе описывается процесс установки и настройки веб-сервера Apache. Он включает в себя действия для защиты сервера, а также использование различных модулей, чтобы можно было подключить разные методы аутентификации и скриптовые языки к своему веб-серверу. Затем я опишу, как генерировать сертификаты для создания сайта HTTPS Secure Sockets Layer (SSL).

Веб-сервер Apache

Сервер *Apache HTTPD*, известный также как *Apache HTTPD Server*, предоставляет службу, с которой взаимодействуют клиентские браузеры. Демон (`httpd`) работает в фоновом режиме на основном сервере и ожидает запросов от веб-клиентов. Браузеры предоставляют эти соединения демону HTTP и отправляют запросы, которые он интерпретирует, отсылая обратно соответствующие данные, например веб-страницу или что-то другое.

Сервер Apache HTTPD включает в себя интерфейс, который позволяет модулям подключаться к процессу для обработки определенных частей запроса. Помимо них в сервере доступны модули для обработки скриптовых языков, таких как Perl или PHP, в веб-документах, которые позволяют добавить шифрование к соединениям между клиентами и сервером.

Веб-сервер Apache изначально являлся набором патчей и улучшений для демона HTTP, созданного Национальным центром суперкомпьютерных приложений (National Center for Supercomputing Applications, NCSA) Иллинойсского университета в Урбана-Шампейне. HTTP-демон NCSA был самым популярным HTTP-сервером в то время, но после того, как его автор Роберт Маккул в середине 1994 года покинул NCSA, начал сдавать свои позиции.

ПРИМЕЧАНИЕ

Mosaic — еще один проект NCSA. Большинство современных браузеров происходят от Mosaic.

В начале 1995 года группа разработчиков сформировала Apache Group и начала вносить значительные изменения в кодовую базу HTTPD NCSA. Apache вскоре сменил NCSA HTTPD на позиции самого популярного веб-сервера и сохраняет ее по сей день.

Позже Apache Group сформировала фонд Apache Software Foundation (ASF) для содействия разработке Apache и других программ с открытым кодом. С началом работы над новыми проектами в ASF сервер Apache стал известен как Apache HTTPD, и эти два названия взаимозаменяемы до сих пор. В настоящее время фонд ASF поддерживает более 350 инициатив с открытым исходным кодом, в том числе Tomcat (включает в себя технологии Java Servlet и JavaServer Pages с открытым исходным кодом), Hadoop (проект, обеспечивающий высокодоступные вычисления) и SpamAssassin (программа фильтрации электронной почты).

Установка веб-сервера

Хотя сервер Apache доступен в каждом крупном дистрибутиве Linux, часто он упакован по-разному. В большинстве случаев для запуска простого веб-сервера Apache нужен лишь пакет, содержащий сам демон Apache (`/usr/sbin/httpd`) и связанные с ним файлы. В Fedora, RHEL и других дистрибутивах веб-сервер Apache поставляется в пакете `httpd`.

Пакет httpd

Чтобы изучить пакет `httpd` в системах Fedora или RHEL перед установкой, загрузите его с помощью команды `yumdownloader` и выполните несколько команд `rpm` для просмотра содержимого пакета:

```
# yumdownloader httpd
# rpm -qpi httpd-*rpm
Name        : httpd
Version     : 2.4.41
Release     : 1.fc30
Architecture: x86_64
Install Date: (not installed)
Group       : Unspecified
Size        : 5070831
License     : ASL 2.0
Signature   : RSA/SHA256, Mon 19 Aug 2019 06:06:09 AM EDT, Key ID
ef3c111fcfc659b9
Source RPM  : httpd-2.4.41-1.fc30.src.rpm
Build Date  : Thu 15 Aug 2019 06:07:29 PM EDT
Build Host  : buildvm-30.phx2.fedoraproject.org
Relocations : (not relocatable)
Packager    : Fedora Project
Vendor      : Fedora Project
URL         : http://httpd.apache.org/
Bug URL     : https://bugz.fedoraproject.org/httpd
Summary     : Apache HTTP Server
Description :
The Apache HTTP Server is a powerful, efficient, and extensible
web server.
```

Команда `yumdownloader` загружает последнюю версию пакета `httpd` в текущий каталог. Команда `rpm -qpi` запрашивает информацию из только что загруженного пакета `httpd` RPM. В примере видно, что пакет был создан проектом Fedora и это действительно пакет HTTP-сервера Apache. Затем загляните внутрь пакета, чтобы увидеть следующие файлы конфигурации:

```
# rpm -qpc httpd-*rpm
/etc/httpd/conf.d/autoindex.conf
/etc/httpd/conf.d/userdir.conf
/etc/httpd/conf.d/welcome.conf
/etc/httpd/conf.modules.d/00-base.conf
/etc/httpd/conf.modules.d/00-dav.conf
...
/etc/httpd/conf/httpd.conf
/etc/httpd/conf/magic
/etc/logrotate.d/httpd
/etc/sysconfig/htcacheclean
```

Основной файл конфигурации Apache — это `/etc/httpd/conf/httpd.conf`. Файл `welcome.conf` определяет домашнюю страницу по умолчанию, пока не будет добавлено другое содержимое. Файл `Magic` создает правила, которые сервер может использовать для определения типа файла при попытке его открыть.

Файл `etc/logrotate.d/httpd` определяет, как изменены файлы журнала Apache. Файл `/usr/lib/tmpfiles.d/httpd.conf` определяет каталог, содержащий временные файлы (этот файл изменять не нужно).

Некоторые модули Apache переносят файлы конфигурации (`*.conf`) в каталог `/etc/httpd/conf.modules.d/`. Любой файл в этом каталоге, заканчивающийся на `.conf`, помещается в основной файл `httpd.conf` и используется для настройки Apache. Большинство пакетов модулей, которые поставляются с файлами конфигурации, помещают их в каталог `/etc/httpd/conf.d`. Например, модули `mod_ssl` (для защищенных веб-серверов) и `mod_python` (для интерпретации кода python) имеют связанные файлы конфигурации в каталоге `/etc/httpd/conf.d` с именами `ssl.conf` и `python.conf` соответственно.

Чтобы начать настройку веб-сервера, можно просто установить пакет `httpd`. Однако при желании можно добавить дополнительные пакеты, связанные с пакетом `httpd`. Один из способов — установить весь пакет Web Server (в системе Fedora) или основную группу Basic Web Server (в системе RHEL), как показано в следующем примере:

```
# yum groupinstall "Web Server"
```

Помимо пакетов, связанных с пакетом `httpd` (например, `rsyslogd`, `irqbalance` и др.), в группе Web Server в Fedora существуют и другие пакеты, которые по умолчанию поставляются вместе с `httpd`.

- **httpd-manual.** Заполняет каталог `/var/www/manual` документацией по работе с сервером Apache. После запуска службы `httpd` (как показано далее) доступ к этому набору руководств можно получить из браузера на локальном компьютере, набрав `localhost/manual` в адресной строке.

  Вместо `localhost` можно использовать полное доменное имя или IP-адрес системы. Затем появится страница Apache Documentation (рис. 17.1).

- **mod_ssl.** Содержит модуль и файл конфигурации, необходимые веб-серверу для обеспечения безопасных подключений к клиентам с использованием протоколов Secure Sockets Layer (SSL) и Transport Layer Security (TLS). Эта функция необходима для шифрования различных данных, например, при онлайн-покупках. Файл конфигурации находится в каталоге `/etc/httpd/conf.d/ssl.conf`.

- **crypto-utils.** Содержит команды для генерирования ключей и сертификатов, необходимых для безопасной связи с веб-сервером Apache.

- **mod_perl.** Содержит модуль Perl (`mod_perl`), файл конфигурации и связанные с ним файлы, необходимые для того, чтобы веб-сервер Apache мог напрямую выполнять любой код Perl.

- **php.** Содержит модуль PHP и файл конфигурации, необходимые для запуска PHP-скриптов непосредственно в сервере Apache. Связанные пакеты включают

пакеты `php-ldap` (запуск PHP-кода, который должен получить доступ к базам данных LDAP) и `php-mysql` (поддержка баз данных на сервере Apache).

- **php-ldap.** Обеспечивает поддержку облегченного протокола доступа к каталогам (LDAP) в модуль PHP, разрешая доступ к службе каталогов по сети.
- **squid.** Предоставляет прокси-службы для определенных протоколов, таких как HTTP, как описано в главе 14 «Администрирование сети». Хотя пакет сам по себе не предоставляет HTTP-содержимое, прокси-сервер Squid обычно пересылает запросы от прокси-клиентов в Интернет или другую сеть, в которой есть веб-содержимое. Такой подход обеспечивает контроль или фильтрацию содержимого, доступ к которому клиенты могут получить из дома, школы или с места работы.
- **webalizer.** Содержит инструменты для анализа данных веб-сервера.

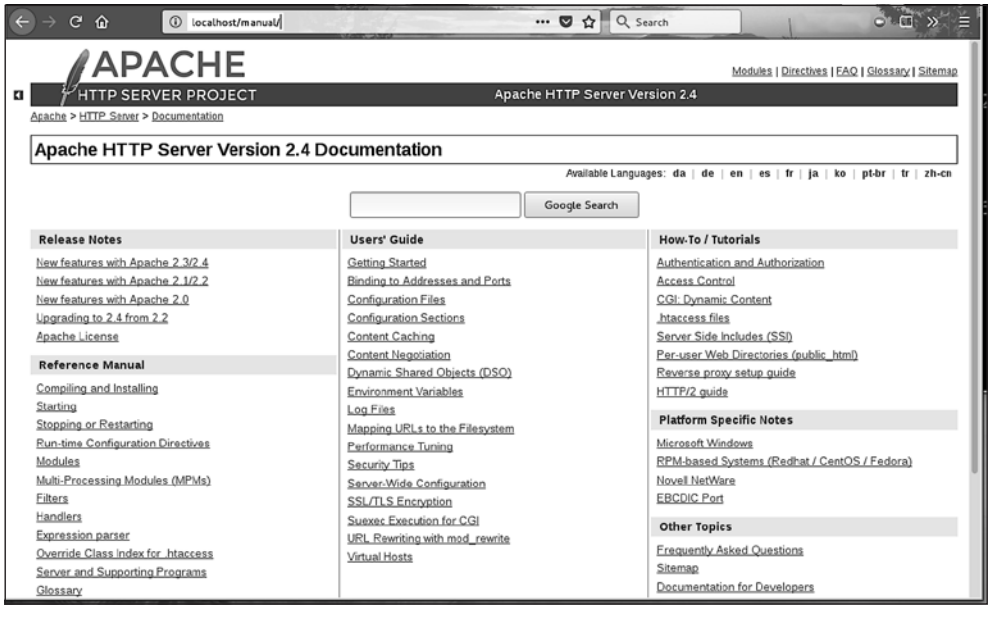


Рис. 17.1. Страница Apache Documentation

Дополнительные пакеты в группе Web Server берутся из подгруппы веб-серверов. Запустите команду `yum groupinfo web-server`, чтобы отобразить перечисленные ранее пакеты. Некоторые из них позволяют применять разные варианты предоставления содержимого, например вики (`moin`), системы управления содержимым (`drupal7`) и блоги (`wordpress`). Другие включают инструменты для построения графиков веб-статистики (`awstats`) или используют легкие альтернативы веб-серверам Apache (`lighttpd` и `cherokee`).

Установка веб-сервера Apache

Несмотря на то что для установки веб-сервера Apache нужен только пакет `httpd`, в начале изучения Apache следует установить руководство (`httpd-manual`). Если вы хотите создать защищенный (SSL) сайт и, возможно, использовать для него статистику, можно просто установить всю группу пакетов, содержащуюся в дистрибутиве Fedora 30:

```
# yum groupinstall "Web Server"
```

При наличии интернет-соединения с репозиторием Fedora (или репозиторием RHEL, если вы используете систему RHEL) все обязательные и стандартные пакеты из этой группы будут установлены. В таком случае у вас будет все программное обеспечение, необходимое для выполнения примеров и упражнений, описанных в этой главе.

Запуск веб-сервера Apache

Чтобы запустить веб-сервер Apache, нужно задать запуск службы при каждой перезагрузке. В Red Hat Enterprise Linux (до RHEL 6) и более старых дистрибутивах Fedora от имени суперпользователя можно ввести следующее:

```
# chkconfig httpd on
# service httpd start
Starting httpd:                  [ OK ]
```

В системах Fedora 30 и RHEL 8 включение и запуск пакета `httpd` осуществляются с помощью команды `systemctl`:

```
# systemctl enable httpd.service
# systemctl start httpd.service
# systemctl status httpd.service
• httpd.service – The Apache HTTP Server
  Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled;
    vendor preset: disabled)
 Drop-In: /usr/lib/systemd/system/httpd.service.d
          └── php-fpm.conf
  Active: active (running) since Mon 2019-09-02 16:16:56 EDT;
    21min ago
    Docs: man:httpd.service(8)
Main PID: 11773 (/usr/sbin/httpd)
  Status: "Total requests: 14; Idle/Busy workers 100/0;Requests/sec:
    0.0111; Bytes served/s>
   Tasks: 214 (limit: 2294)
  Memory: 24.6M
  CGroup: /system.slice/httpd.service
          ├── 11773 /usr/sbin/httpd -DFOREGROUND
          ├── 11774 /usr/sbin/httpd -DFOREGROUND
          ├── 11775 /usr/sbin/httpd -DFOREGROUND
```

```
           ├── 11776 /usr/sbin/httpd -DFOREGROUND
           ├── 11777 /usr/sbin/httpd -DFOREGROUND
           └── 11778 /usr/sbin/httpd -DFOREGROUND
...
```

При запуске службы `httpd` по умолчанию запускаются пять или шесть демонов `httpd` (в зависимости от вашей системы Linux), которые обрабатывают запросы веб-сервера. Для обработки запросов можно настроить больше или меньше демонов `httpd` на основе настроек в файле `httpd.conf` (они описаны в подразделе «Файлы конфигурации веб-сервера Apache» далее в этой главе).

Чтобы настроить поведение демона `httpd`, вы можете изменить службу `httpd`, запустив команду `systemctl edit httpd`. Так как существуют различные версии службы `httpd`, проверьте ее справочную страницу (`man httpd`), чтобы узнать, какие параметры можно передать демону `httpd`. Например, выполните команду `systemctl edit httpd` и добавьте следующую запись:

```
[Service]
Environment=OPTIONS='-e debug'
```

Сохраните изменения (сочетания клавиш Ctrl+O, Ctrl+X). Добавьте параметр `-e debug`, чтобы увеличить уровень журнала, отправляя максимальное количество сообщений Apache в файлы журнала. Перезапустите службу `httpd`, чтобы изменения вступили в силу. Введите команду `ps`, чтобы убедиться в этом:

```
$ ps -ef | grep httpd
root   14575  1      0  08:49  ?  00:00:01  /usr/sbin/httpd -e debug
-DFOREGROUND
apache 14582  14575  0  08:49  ?  00:00:00  /usr/sbin/httpd -e debug
-DFOREGROUND
```

Если вы добавили параметр отладки (`-e debug`), не забудьте убрать его, снова запустив команду `systemctl edit httpd` и удалив запись после окончания отладки Apache, затем перезапустите службу. Если оставить отладку включенной, то файлы журнала быстро заполнятся.

Защита веб-сервера Apache

Чтобы защитить Apache, необходимо знать о стандартных функциях безопасности Linux (права, владение, брандмауэры, Security Enhanced Linux), а также о функциях безопасности, специфичных для Apache. В следующих разделах описываются функции безопасности, относящиеся к Apache.

Права доступа к файлам веб-сервера Apache и владения ими

Процесс демона `httpd` выполняется от имени пользователя `apache` и группы `apache`. По умолчанию HTML-содержимое хранится в каталоге `/var/www/html` (определяется значением переменной `DocumentRoot` в файле `httpd.conf`).

Чтобы демон `httpd` получил доступ к этому содержимому, применяются стандартные права Linux. Если у других пользователей нет прав на чтение, они должны быть включены для пользователя или группы `apache`, чтобы файлы считывались и обслуживались клиентами. Аналогично любой каталог, через который демон `httpd` должен пройти, чтобы получить доступ к содержимому, должен иметь права на выполнение для пользователя `apache`, группы `apache` или другого пользователя (`other`).

Хотя вы не можете войти в систему как пользователь `apache` (`/sbin/nologin` — это оболочка по умолчанию), но можете создать содержимое как суперпользователь и изменить его владельца (команда `chown`) или права (команда `chmod`). При этом часто отдельные учетные записи пользователей или групп добавляются для создания содержимого, которое могут читать все (`other`), но которое доступно только для записи конкретными пользователем или группой.

Веб-сервер Apache и брандмауэры

Если вы заблокировали свой брандмауэр (межсетевой экран) в Linux, необходимо открыть несколько портов, чтобы клиенты могли общаться с веб-сервером Apache через него. Стандартный веб-сервис (HTTP) доступен через TCP-порт 80, безопасный веб-сервис (HTTPS) доступен через TCP-порт 443. (Порт 443 появляется только в том случае, если вы установили пакет `mod_ssl`, как описано далее.)

Чтобы проверить, какие порты использует сервер `httpd`, примените команду `netstat`:

```
# netstat -tupln | grep httpd
tcp6   0      0 :::80              :::*              LISTEN     29169/httpd
tcp6   0      0 :::443             :::*              LISTEN     29169/httpd
```

Выходные данные в примере показывают, что демон `httpd` (идентификатор процесса ID 29169) прослушивает все адреса портов 80 (`:::80`) и 443 (`:::443`). Оба порта связаны с протоколом TCP (`tcp6`). Чтобы открыть эти порты в дистрибутивах Fedora или Red Hat Enterprise Linux, необходимо добавить правила брандмауэра.

В текущей системе Fedora 30 или RHEL 7 либо 8 откройте окно Firewall (Брандмауэр), введя `Firewall` и нажав клавишу Enter на экране Acivities (Приложения) на рабочем столе GNOME 3. Далее выберите вариант Permanent (Постоянная) в поле Configuration (Конфигурация). Затем, выбрав общедоступную зону, установите флажки рядом с полями служб http и https. Эти порты немедленно откроются.

Для RHEL 6 или более старых версий системы Fedora добавьте правила в файл `/etc/sysconfig/iptables` (где-то перед конечным параметром `DROP` или `REJECT`), например:

```
-A INPUT -m state --state NEW -m tcp -p tcp --dport 80 -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 443 -j ACCEPT
```

Перезапустите службу `iptables` (`service iptables restart`), чтобы новые правила вступили в силу.

Защита веб-сервера Apache с помощью SELinux

Если система *Security Enhanced Linux* (SELinux) установлена (по умолчанию в системах Fedora и Red Hat Enterprise Linux), то она обеспечивает дополнительный уровень безопасности вашей службе `httpd`. По сути, SELinux защищает систему от повреждений со стороны, к примеру от того, кто взломал демон `httpd`. SELinux создает политику, которая:

- запрещает доступ к файлам, которые не заданы в нужных контекстах файлов. Для службы `httpd` в SELinux существуют различные контексты файлов содержимого, файлов конфигурации, файлов журналов, скриптов и других файлов, связанных со службой. Любой файл, который не установлен в надлежащий контекст, недоступен для демона `httpd`;
- предотвращает использование небезопасных функций, таких как загрузка файлов и аутентификация с открытым текстом, устанавливая логические типы (booleans) для таких функций в положение `off`. Можете выборочно включать логические типы по мере необходимости, если они соответствуют вашим требованиям безопасности;
- не допускает демон `httpd` к нестандартным функциям, например к порту за пределами портов по умолчанию, которые служба может использовать.

Полное описание функции SELinux содержится в главе 24 «Повышенная безопасность с технологией SELinux». Сейчас перечислим некоторые моменты, о которых необходимо знать при использовании SELinux с веб-сервером Apache и службой `httpd`.

- **Отключите SELinux.** Нет необходимости задействовать SELinux постоянно. Вы можете оставить его в разрешительном режиме (`permissive`), если не хотите создавать политику SELinux, необходимую для того, чтобы веб-сервер работал с SELinux. Можете изменить принудительный режим на разрешительный, отредактировав файл `/etc/sysconfig/selinux` так, чтобы значение `SELINUX` было установлено, как в примере далее. После этого в следующий раз, когда перезагрузите систему, система будет переведена в разрешительный режим. Это означает, что если вы нарушаете политику SELinux, то это событие будет зарегистрировано, но не прервано системой (как было бы в принудительном режиме):

  ```
  SELINUX=permissive
  ```

- **Прочитайте справочную страницу httpd_selinux.** Введите команду `man httpd_selinux` из оболочки. На этой справочной странице показаны соответствующие контексты файлов и доступные логические типы. (Если справочной страницы в системе нет, установите ее с помощью команды `yum install selinux-policy-doc`.)
- **Используйте стандартные места расположения для файлов.** При создании новые файлы наследуют контексты файлов каталогов, в которых они хранятся. Поскольку файл `/etc/httpd` установлен в правильный контекст для файлов

конфигурации, файл `/var/www/html` подходит для файлов содержимого и т. д. Правильная настройка контекстов файлов предполагает простое копирование файлов или создание новых файлов там же.

- **Измените настройки SELinux, чтобы разрешить нестандартные функции.** Возможно, вы захотите использовать веб-содержимое из каталога `/mystuff` или поместить файлы конфигурации в каталог `/etc/whatever`. Кроме того, можете разрешить пользователям вашего сервера загружать файлы, запускать скрипты или включать другие функции, которые по умолчанию отключены в SELinux. В этих случаях можете задействовать команды SELinux для установки контекстов файлов и логических типов, необходимых для того, чтобы SELinux работал так, как нужно вам.

Обязательно прочтите главу 24 «Повышенная безопасность с технологией SELinux», чтобы узнать больше о системе SELinux.

Файлы конфигурации веб-сервера Apache

Файлы конфигурации для Apache HTTPD невероятно гибки, а это означает, что сервер можно настроить практически полностью под свои нужды. Такая гибкость достигается за счет повышенной сложности в виде большого количества вариантов конфигурации, называемых *директивами*. На практике, однако, в большинстве случаев активно используются лишь несколько директив.

ПРИМЕЧАНИЕ

На сайте httpd.apache.org/docs/current/mod/directives.html представлен полный список директив, поддерживаемых веб-сервером Apache. Если у вас установлен пакет httpd-manual, то описания этих директив и других функций Apache будут доступны в руководстве на сервере, на котором работает Apache: localhost/manual/.

В дистрибутивах Fedora и RHEL основной файл конфигурации базового сервера Apache находится в файле `/etc/httpd/conf/httpd.conf`. Помимо него, любой файл, заканчивающийся на `.conf` в каталоге `/etc/httpd/conf.d`, также применяется для настройки сервера Apache (на основе строки `Include` в файле `httpd.conf`). В дистрибутиве Ubuntu конфигурация Apache хранится в текстовых файлах, считываемых сервером Apache, начиная с `/etc/apache2/apache2.conf`. Конфигурация считывается с начала до конца, причем большинство директив обрабатываются в том порядке, в котором они были прочитаны.

Использование директив

Область действия многих директив конфигурации может быть изменена в зависимости от контекста. Другими словами, некоторые параметры могут быть установлены на общем уровне, а затем изменены для конкретного файла, каталога или виртуального хоста. Некоторые директивы всегда носят общий характер, например

определяющие, какие IP-адреса прослушивает сервер. Другие активны только при применении к определенному месту.

Места настраиваются в виде начального тега, содержащего тип места и местоположение ресурса, затем перечисляются параметры конфигурации для него, и все это заканчивается конечным тегом. Эту форму часто называют *блоком конфигурации*, и она очень похожа на HTML-код. Специальный тип блока конфигурации, известный как *блок location*, используется для ограничения области действия директив определенными файлами или каталогами. Эти блоки имеют следующий вид:

```
<locationtag specifier>
(options specific to objects matching the specifier go within this block)
</locationtag>
```

Существуют различные типы тегов места, которые выбираются в зависимости от типа указанного местоположения ресурса. Спецификатор, включенный в начальный тег, обрабатывается в зависимости от типа тега местоположения. Теги места, которые вы обычно используете и с которыми сталкиваетесь, — это `Directory`, `Files` и `Location`, они ограничивают область действия директив определенными каталогами, файлами или местоположениями соответственно.

- `Directory` применяются для указания пути, основанного на расположении в файловой системе. Например, `<Directory/>` относится к корневому каталогу на компьютере. Каталоги наследуют настройки из каталогов, расположенных над ними, причем наиболее специфичный блок `Directory` переопределяет менее специфичные независимо от порядка их появления в файлах конфигурации.
- `File` используются для указания файлов по имени. Они могут содержаться внутри блока `Directory` и применяться только с файлами из этого каталога. Настройки в блоке `File` переопределяют настройки в блоках `Directory`.
- `Location` используются для указания URI, применяемого для доступа к файлу или каталогу. Они отличаются от блока `Directory` тем, что относятся к адресу, содержащемуся в запросе, а не к реальному местоположению файла на диске. Теги `Location` обрабатываются последними и переопределяют настройки в блоках `Directory` и `Files`.

Сравните варианты этих тегов `DirectoryMatch`, `FilesMatch` и `LocationMatch`. У них одна и также функция, однако они могут содержать регулярные выражения в спецификации ресурса. Блоки `FilesMatch` и `LocationMatch` обрабатываются одновременно вместе с блоками `File` и `Location` соответственно. Блоки `DirectoryMatch` обрабатываются после блоков `Directory`.

Сервер Apache можно настроить также для обработки параметров конфигурации, содержащихся в файлах с именем, указанным в директиве `AccessFileName` (которая обычно имеет значение `.htaccess`). Директивы в файлах конфигурации доступа применяются ко всем объектам в каталоге, который они содержат, включая подкаталоги и их содержимое. Файлы конфигурации доступа обрабатываются одновременно с блоками `Directory`, используя аналогичный порядок наиболее специфического соответствия.

ПРИМЕЧАНИЕ

Файлы управления доступом полезны тем, что позволяют пользователям изменять определенные настройки, не имея доступа к файлам конфигурации сервера. Директивы конфигурации, разрешенные в файле конфигурации доступа, определяются параметром AllowOverride в своем каталоге. Некоторые директивы не имеют смысла на этом уровне и при попытке перейти к URl обычно выводят сообщение о внутренней ошибке сервера. Параметр AllowOverride подробно описан на сайте httpd.apache.org/docs/mod/core.html#allowoverride.

Три директивы, обычно встречающиеся в блоках расположения и файлах управления доступом, — это `DirectoryIndex`, `Options` и `ErrorDocument`:

- `DirectoryIndex` сообщает серверу Apache, какой файл следует загрузить, если URI содержит каталог, но не имя файла. Эта директива не работает в блоках `Files`.
- Параметр `Options` используется для настройки того, как Apache обрабатывает файлы в каталоге. Параметр `ExecCGI` сообщает Apache, что файлы в этом каталоге могут запускаться как CGI-скрипты, а параметр `Includes` — что включения на стороне сервера (SSIs) разрешены. Другим распространенным параметром является `Indexes`, который приказывает Apache генерировать список файлов, если одно из имен файлов, найденных в настройках `DirectoryIndex`, отсутствует. Можно указать абсолютный список параметров или изменить его, добавив перед именем параметры. См. страницу httpd.apache.org/docs/mod/core.html#options для получения дополнительной информации.
- Директивы `ErrorDocument` можно использовать для указания файла, содержащего сообщения веб-клиентам при возникновении конкретной ошибки. Файл располагается в каталоге `/var/www`. Директива должна указывать код ошибки и полный URI в документе об ошибке. Возможные коды ошибок — 403 (отказано в доступе), 404 (файл не найден) и 500 (внутренняя ошибка сервера). Больше информации о директиве `ErrorDocument` имеется на странице httpd.apache.org/docs/mod/core.html#errordocument. Например, когда клиент запрашивает URL-адрес с ненайденного сервера, следующая строка `ErrorDocument` приводит к тому, что код ошибки `Includes` отправляет клиенту сообщение об ошибке, которая указана в файле `/var/www/error/HTTP_NOT_FOUND.html.var`:

```
ErrorDocument 404 /error/HTTP_NOT_FOUND.html.var
```

Еще один распространенный вариант применения блоков расположения и файлов управления доступом — ограничение или расширение доступа к ресурсу. Директива `Allow` используется для разрешения доступа к соответствующим хостам, а директива `Deny` — для его запрета. Оба этих параметра могут встречаться более одного раза в блоке и обрабатываются на основе параметра `Order`. Установив параметр `Order` в директиве `Deny`, директива `Allow` разрешает доступ к любому хосту, который не указан в директиве `Deny`. После настройки директивы `Allow` директива `Deny` запрещает доступ к любому хосту, не разрешенному в директиве `Allow`.

Как и в большинстве других случаев, для хоста чаще используются наиболее специфичные параметры `Allow` или `Deny`, что означает, что вы можете запретить доступ (`Deny`) к диапазону и разрешить доступ (`Allow`) к подмножествам этого диапазона. Добавив параметр `Satisfy` и некоторые дополнительные параметры, вы можете задействовать аутентификацию паролем. Для получения дополнительной информации о директивах `Allow`, `Deny`, `Satisfy` и др. обратитесь к Apache Directive Index по адресу httpd.apache.org/docs/current/mod/directives.html.

Настройки по умолчанию

Начать использовать веб-сервер Apache можно сразу после его установки, потому что файл `httpd.conf` содержит настройки по умолчанию, которые сообщают серверу, где искать веб-контент, скрипты, файлы журналов и другие элементы, необходимые для работы. Он также включает в себя настройки, которые сообщают серверу, сколько серверных процессов должно выполняться одновременно и как отображается содержимое каталога.

Если вы хотите разместить один сайт (например, на домене example.com), просто добавьте в каталог `/var/www/html` его содержимое и адрес вашего сайта на DNS-сервере, чтобы другие пользователи могли просматривать его. Затем по мере необходимости можно изменить описанные ранее директивы.

Чтобы помочь вам разобраться с настройками, которые входят в стандартный файл `httpd.conf`, я далее привел некоторые из них вместе с описанием. Чтобы было яснее, часть комментариев удалил и переставил некоторые настройки.

Настройки в следующем примере отображают места, где сервер `httpd` получает и куда помещает содержимое по умолчанию:

```
ServerRoot "/etc/httpd"
Include conf.d/*.conf
ErrorLog logs/error_log
CustomLog "logs/access_log" combined
DocumentRoot "/var/www/html"
ScriptAlias /cgi-bin/ "/var/www/cgi-bin/"
```

Директива `ServerRoot1` определяет файл `/etc/httpd` как место, где хранятся файлы конфигурации.

В таком случае там, где появляется строка `Include`, все файлы, оканчивающиеся на `.conf` в каталоге `etc/httpd/conf.d`, включены в файл `httpd.conf`. Файлы конфигурации часто связаны с модулями Apache (и включены в программный пакет вместе с модулем) или блоками виртуальных хостов (которые вы можете добавить сами при настройке виртуальных хостов в отдельных файлах). См. раздел «Виртуальный хост в веб-сервере Apache» далее в этой главе.

Сообщения об обнаружении ошибок или обработки содержимого помещаются в файлы, указанные строками `ErrorLog` и `CustomLog`. Как видно в приведенном примере, все эти журналы хранятся в каталогах `/etc/httpd/logs/error_log` и `etc/httpd/logs/access_log` соответственно. Эти журналы жестко связаны с каталогом `/var/log/httpd`, поэтому оттуда можно получить доступ к тому же файлу.

Директивы DocumentRoot и ScriptAlias определяют, где хранится содержимое, обслуживаемое httpd-сервером. Обычно файл Index.htm размещается в каталоге DocumentRoot (/var/www/html по умолчанию) в качестве домашней страницы. Добавлять содержимое можно по мере необходимости. Директива ScriptAlias сообщает демону httpd, что все скрипты, запрошенные из каталога cgi-bin, должны быть найдены в каталоге /var/www/cgi-bin. Например, клиент может получить доступ к скрипту, расположенному в файле /var/www/cgi-bin/script.cgi, введя URL-адрес, допустим, example.com/cgi-bin/script.cgi.

Помимо расположения файлов, в файле httpd.conf можно найти и другую информацию, например:

```
Listen 80
User apache
Group apache
ServerAdmin root@localhost
DirectoryIndex index.html index.php
AccessFileName .htaccess
```

Директива Listen 80 указывает службе httpd прослушивать входящие запросы на порте 80 (порт по умолчанию для протокола HTTP). По умолчанию она прослушивает все сетевые интерфейсы, хотя ее можно ограничить выбранными интерфейсами по IP-адресу (например, Listen 192.168.0.1:80).

Директивы User и Group говорят службе httpd работать от имени apache как для пользователя, так и для группы. Значение директивы ServerAdmin (root@localhost по умолчанию) публикуется на некоторых веб-страницах, чтобы сообщить пользователям, куда обращаться по электронной почте при возникновении проблем с сервером.

Директива DirectoryIndex перечисляет файлы, которые служба httpd будет обслуживать при запросе каталога. Например, если браузер запросил каталог http://host/whatever/, то служба httpd увидит, существует ли каталог /var/www/html/whatever/index.html, и, если он есть, начнет его обслуживать. Если бы каталога не существовало, то служба httpd искала бы файл index.php. Если этот файл не найден, содержимое каталога отобразится. Чтобы указать службе httpd использовать содержимое файла .htaccess, можно добавить директиву AccessFileName, если он существует в каталоге. Например, файл можно использовать для того, чтобы потребовать защитить каталог паролем или указать, что содержимое каталога должно отображаться определенным образом. Однако для того, чтобы этот файл работал, контейнер директивы Directory (описан далее) должен отключить параметр AllowOverride. (По умолчанию параметр AllowOverride None запрещает применение файла .htaccess для любых директив.)

Далее показаны контейнеры директив Directory, которые определяют поведение при доступе к корневым каталогам (/) /var/www и /var/www/html:

```
<Directory/>
    AllowOverride none
    Require all denied
```

```
</Directory>
<Directory "/var/www">
    AllowOverride None
    # Allow open access:
    Require all granted
</Directory>
<Directory "/var/www/html">
    Options Indexes FollowSymLinks
    AllowOverride None
    Require all granted
</Directory>
```


Первый контейнер `Directory` (`/`) указывает, что, если служба `httpd` попытается получить доступ к любым файлам в файловой системе Linux, он будет запрещен. Директива `AllowOverride none` запрещает файлам `.htaccess` переопределять настройки для этого каталога. Эти параметры применяются ко всем подкаталогам, которые не определены в других контейнерах директивы `Directory`.

В каталоге `/var/www` можно настроить доступ к содержимому, то есть предоставить доступ к содержимому, добавленному в него. Однако переопределение параметров недопустимо.

Контейнер директивы `Directory/var/www/html` следует за символическими ссылками и не допускает переопределений. При установке параметра `Require all granted` служба `httpd` никак не препятствует доступу к серверу.

Если все только что описанные настройки сработали, можно начать добавлять нужное содержимое в каталоги `/var/www/html` и `/var/www/cgi-bin` `html`. Одна из причин, по которым настройка по умолчанию здесь не подходит, заключается в том, что может понадобиться обслуживать содержимое для нескольких доменов (например, example.com, example.org и example.net). Для этого необходимо настроить виртуальные хосты. Виртуальные хосты, более подробно описанные в следующем подразделе, — это удобный (и почти необходимый) инструмент для доставки клиентам различной информации на основе адреса или имени сервера, на который направляется запрос. К виртуальным хостам применяются большинство общих параметров конфигурации, но они могут быть переопределены директивами внутри блока `VirtualHost`.

Виртуальный хост в веб-сервере Apache

Веб-сервер Apache поддерживает создание отдельных сайтов на одном сервере для сохранения различного содержимого. Отдельные сайты настраиваются на одном сервере в так называемых *виртуальных хостах*.

На самом деле *виртуальные хосты* — это простой способ добавить содержимое для нескольких доменов, доступных с одного и того же сервера Apache. Вместо того чтобы физически обслуживать содержимое для каждого домена из отдельной системы, можно обслуживать его для нескольких доменов из одной операционной системы.

Сервер Apache, выполняющий виртуальный хостинг, может иметь несколько доменных имен, разрешенных на IP-адресе сервера. Информация, которая отсылается веб-клиенту, зависит от доменного имени, используемого для доступа к серверу.

Например, если клиент попал на сервер, запросив имя www.example.com, то он будет направлен к контейнеру виртуального хоста, для которого директива `ServerName` применяет www.example.com. Контейнер предоставит место расположения содержимого и, возможно, различные журналы ошибок или директивы `Directory` из общих настроек. Таким образом, каждым виртуальным хостом можно управлять так, как если бы он находился в отдельной системе.

Чтобы использовать виртуальный хостинг на основе имен, добавьте столько контейнеров `VirtualHost`, сколько захотите. Порядок настройки виртуального хоста таков.

ПРИМЕЧАНИЕ

После подключения первого блока VirtualHost стандартная директива DocumentRoot (/var/www/html) больше не используется, если кто-то обращается к серверу по IP-адресу или какому-то имени, которое не задано в контейнере VirtualHost. Вместо этого первый контейнер VirtualHost служит местом расположения сервера по умолчанию.

1. В системах Fedora и RHEL создайте файл с именем `/etc/httpd/conf.d/example.org.conf` с помощью следующего шаблона:

```
<VirtualHost *:80>

    ServerAdmin     webmaster@example.org
    ServerName      www.example.org
    ServerAlias     web.example.org
    DocumentRoot    /var/www/html/example.org/
DirectoryIndex index.php index.html index.htm
</VirtualHost>
```

   В примере показаны следующие настройки.

   - Спецификация `*: 80` в блоке `VirtualHost` указывает, к какому адресу и порту относится этот виртуальный хост. Если IP-адресов, связанных с вашей системой Linux, несколько, то символ * заменяется определенным IP-адресом. Значение порта необязательно для спецификаций `VirtualHost`, но всегда должно использоваться для предотвращения ошибок для виртуальных хостов SSL (по умолчанию они применяют порт 443).
   - Строки `ServerName1` и `ServerAlias1` сообщают веб-серверу Apache, какие имена должен распознавать виртуальный хост, поэтому замените их именами, соответствующими вашему сайту. Не используйте строку `ServerAlias`, если у вас нет дополнительных имен для сервера. Задействуйте строку `ServerAlias` с несколькими именами, если есть альтернативные имена.
   - Директива `DocumentRoot` указывает, где хранятся веб-документы (содержимое, обслуживаемое для этого сайта). Несмотря на то что она отображается

как подкаталог, который создается в каталоге `DocumentRoot` по умолчанию (`/var/www/html`), сайты прикрепляются к домашним каталогам определенных пользователей (например, `/home/chris/public_html`), так что различными сайтами могут управлять разные пользователи.

2. При включенном хосте примените команду `apachectl` для проверки конфигурации, а затем выполните перезагрузку с помощью команды `graceful`:

```
# apachectl configtest
Syntax OK
# apachectl graceful
```

При условии, что вы зарегистрировали систему на DNS-сервере, браузер должен иметь возможность получить доступ к этому сайту с помощью www.example.org или web.example.org. Если это сработает, можете начать добавлять в систему другие виртуальные хосты.

Существует еще один способ расширить возможности вашего сайта — разрешить нескольким пользователям делиться содержимым собственных ресурсов на вашем сервере. Можете разрешить им добавлять то, чем они хотят поделиться через ваш веб-сервер, в подкаталог своих домашних каталогов, как описано в следующем разделе.

ПРИМЕЧАНИЕ

Удобный способ управления виртуальными хостами — хранить каждый в отдельном файле. Однако необходимо сохранить ваш основной виртуальный хост в файле, который будет прочитан раньше других, потому что первый виртуальный хост получает запросы на имена сайтов, не совпадающие ни с одним из них в вашей конфигурации. В коммерческой среде веб-хостинга обычно создается специальный виртуальный хост по умолчанию, содержащий сообщение об ошибке, говорящее, что сайт с таким именем не настроен.

Разрешение пользователям публиковать свой веб-контент

В ситуациях, когда у вас нет возможности настроить виртуальный хост для каждого пользователя, которому необходимо предоставить веб-пространство, задействуйте модуль `mod_userdir` в веб-сервере Apache. Если этот модуль включен (по умолчанию отключен), для каждого пользователя в Интернете доступен каталог `public_html` в домашнем каталоге по адресу http://servername/~username/.

Например, пользователь с именем `wtucker` на хосте www.example.org хранит веб-контент в каталоге `/home/wtucker/public_html`. Он доступен по адресу www.example.org/~wtucker.

Внесите изменения в файл `/etc/httpd/conf/httpd.conf`, чтобы пользователи могли публиковать контент из собственных домашних каталогов. Не все версии Apache имеют подобные блоки в файле `httpd.conf`, поэтому, возможно, придется создавать их с нуля.

1. Создайте блок `<IfModule mod_userdir.c>`. Измените `chris` на любое имя пользователя, чтобы разрешить пользователям создать собственный каталог `public_html`. Можно добавить несколько имен:

```
<IfModule mod_userdir.c>
    UserDir enabled chris
    UserDir public_html
</IfModule>
```

2. Создайте блок директивы `<Directory /home/*/public_html>` и измените настройки по своему усмотрению. Пример того, как блок будет выглядеть:

```
<Directory "/home/*/public_html">
    Options Indexes Includes FollowSymLinks
    Require all granted
</Directory>
```

3. Пусть ваши пользователи создадут каталоги `public_html` в своих домашних каталогах:

```
$ mkdir $HOME/public_html
```

4. Установите права на выполнение (как суперпользователь), чтобы демон `httpd` мог получить доступ к домашнему каталогу:

```
# chmod +x /home /home/*
```

5. Если система SELinux находится в принудительном режиме (по умолчанию в дистрибутивах Fedora and RHEL), то правильный контекст файла SELinux (`httpd_user_content_t`) должен быть установлен в каталогах `/home/*/www`, `/home/*/web` и `/home/*/public_html`, чтобы SELinux позволял демону `httpd` автоматически получать доступ к содержимому. Если по какой-то причине контекст не задан, установите его следующим образом:

```
ttpd_user_content_t to /home/*/
# chcon -R --reference=/var/www/html/ /home/*/public_html
```

6. Установите логический тип SELinux, чтобы пользователи могли обмениваться HTML-содержимым из своих домашних каталогов:

```
# setsebool – P httpd_enable_homedirs true
```

7. Перезапустите или перезагрузите службу `httpd`. На этом этапе появляется возможность получить доступ к содержимому каталога `public_html` пользователя, набрав в браузере `//hostname/~user`.

Защита веб-трафика с помощью технологии SSL/TLS

Все данные, которыми вы делитесь со своего сайта с помощью стандартного протокола HTTP, отправляются в виде открытого текста. Это означает, что любой, кто наблюдает за трафиком в сети между сервером и клиентом, может просматривать ваши незащищенные данные. Чтобы обезопасить эту информацию, можно добавить

сертификаты на сайт (чтобы клиент мог проверить, кто вы) и зашифровать данные (чтобы никто не мог войти в вашу сеть и увидеть данные).

Приложения коммерческой среды, например, интернет-магазины и те, кто оказывает банковские услуги, всегда должны быть зашифрованы с помощью спецификаций Secure Sockets Layer (SSL) или Transport Layer Security (TLS). Спецификация TLS основана на версии 3.0 спецификаций SSL, поэтому они очень похожи по своей природе. Из-за этого сходства, а также потому, что SSL старше, последняя используется чаще. Для веб-соединений сначала устанавливается SSL-соединение, а затем через него «туннелируется» обычная HTTP-связь.

ПРИМЕЧАНИЕ

Поскольку спецификация SSL действует раньше любого HTTP-соединения, работа виртуального хостинга на основе имен (который выполняется на уровне HTTP) с SSL проблематична. Как следствие, каждый виртуальный хост SSL, который вы настраиваете, должен иметь уникальный IP-адрес. (См. сайт Apache для получения дополнительной информации: httpd.apache.org/docs/vhosts/name-based.html.)

При соединении SSL-клиента и SSL-сервера для проверки идентификационных данных и установления параметров сеанса и ключа сеанса используется асимметричная криптография (открытый ключ). Затем с согласованным ключом применяется симметричный алгоритм шифрования, чтобы зашифровать данные, передаваемые во время сеанса. Асимметричное шифрование во время фазы «рукопожатия» позволяет обеспечить безопасную связь, не задействуя предварительный ключ, а симметричное шифрование является более быстрым и практичным для использования с данными сеанса.

Чтобы клиент мог проверить сервер, последний должен иметь заранее сгенерированный закрытый ключ и сертификат, содержащий открытый ключ и информацию о сервере. Этот сертификат проверяется с помощью открытого ключа, известного клиенту.

Сертификаты обычно подписаны цифровой подписью, выданной *сторонним центром сертификации* (certificate authority, CA), который проверил личность создателя запроса и действительность запроса. В большинстве случаев CA — это компания, которая договорилась с поставщиком браузера об установке своего сертификата. Затем CA взимает плату с оператора сервера за свои услуги.

Коммерческие центры сертификации различаются по цене, функциям и поддержке браузеров, но помните, что цена не всегда является показателем качества. Популярные CA — InstantSSL (www.instantssl.com), Let's Encrypt (www.letsencrypt.org) и DigiCert (www.digicert.com).

У вас также есть возможность создавать самоподписанные сертификаты, хотя их стоит применять только для тестирования или для небольшого количества пользователей сервера и если вы не планируете добавлять эти сертификаты в несколько систем. Инструкции по созданию самоподписанного сертификата приведены в пункте «Создание SSL-ключа и самоподписанного сертификата» далее в этом подразделе.

И наконец, последний вариант — запустить собственный центр сертификации. Возможно, стоит делать это только в том случае, если у вас есть ожидающие сертификата пользователи и средства для распространения сертификата CA среди них (включая помощь в установке его в своих браузерах). Процесс создания CA слишком сложен, чтобы описывать его в этой книге, но это достойная альтернатива самоподписанным сертификатам.

В следующих пунктах рассказано, как настраивается по умолчанию HTTPS-связь в дистрибутивах Fedora и RHEL при установке пакета `mod_ssl`. После этого я опишу, как лучше настроить SSL-связь, создав собственные SSL-ключи и сертификаты для веб-сервера (в системе Fedora или RHEL), который мы настраивали ранее в этой главе.

Настройка SSL-сертификата

Установите пакет `mod_ssl` в системе Fedora или RHEL (это происходит по умолчанию, если вы установили группу пакетов Basic Web Server), после этого будут созданы самоподписанный сертификат и закрытый ключ, что позволяет сразу же использовать протокол HTTPS для связи с веб-сервером.

Хотя настройка по умолчанию пакета `mod_ssl` реализует зашифрованную связь между вашим веб-сервером и клиентами, но из-за того, что сертификат самоподписанный, клиент, обратившийся к вашему сайту, будет предупрежден о том, что сертификат ненадежен. Перед началом изучения настройки сертификата SSL для веб-сервера Apache, убедитесь, что пакет `mod_ssl` установлен на сервере, на котором запущена служба `httpd`:

```
# yum install mod_ssl
```

Пакет `mod_ssl` включает в себя модуль, необходимый для реализации SSL на веб-сервере (`mod_ssl.so`), и файл конфигурации для SSL-хостов `/etc/httpd/conf.d/ssl.conf`. В этом файле содержится множество комментариев, которые помогут понять, что нужно изменить. Строки без комментариев определяют начальные настройки и виртуальный хост по умолчанию. Пример нескольких таких строк:

```
Listen 443 https
...
<VirtualHost _default_:443>
ErrorLog logs/ssl_error_log
TransferLog logs/ssl_access_log
LogLevel warn
SSLEngine on
...
SSLCertificateFile /etc/pki/tls/certs/localhost.crt
SSLCertificateKeyFile /etc/pki/tls/private/localhost.key
...
</VirtualHost>
```

Служба SSL настроена на прослушивание стандартного SSL-порта 443 на всех сетевых интерфейсах системы.

Создается блок `VirtualHost`, который заставляет сообщения об ошибках и сообщения о доступе регистрироваться в файлах журналов, отделенных от стандартных журналов, используемых сервером (`ssl_error_log` и `ssl_access_login` в каталоге `/var/log/httpd/`). Уровень сообщений журнала установлен на `warn`, и команда `SSLEngine` включена.

В предыдущем примере две записи, связанные с SSL-сертификатами в блоке `VirtualHost`, отображают информацию о ключе и сертификате. Как упоминалось ранее, ключ генерируется при установке пакета `mod_ssl` и помещается в файл `/etc/pki/tls/private/localhost.key`. С помощью этого ключа создается самоподписанный сертификат `/etc/pki/tls/certs/localhost.crt`. Когда вы создадите собственный ключ и сертификат, вам нужно будет заменить значения `SSLCertificateFile` и `SSLCertificateKeyFile` в этом файле.

После установки пакета `mod_ssl` и перезагрузки файла конфигурации проверьте следующим образом, работает ли сертификат по умолчанию.

1. Установите соединение с сайтом из браузера с помощью протокола HTTP. Например, если вы используете браузер Firefox в системе, где работает веб-сервер, введите `https://localhost` в строке поиска и нажмите клавишу Enter. На рис. 17.2 показан пример страницы.
2. Эта страница предупреждает о том, что проверить подлинность этого сайта невозможно, так как нельзя узнать, кто создал данный сертификат.

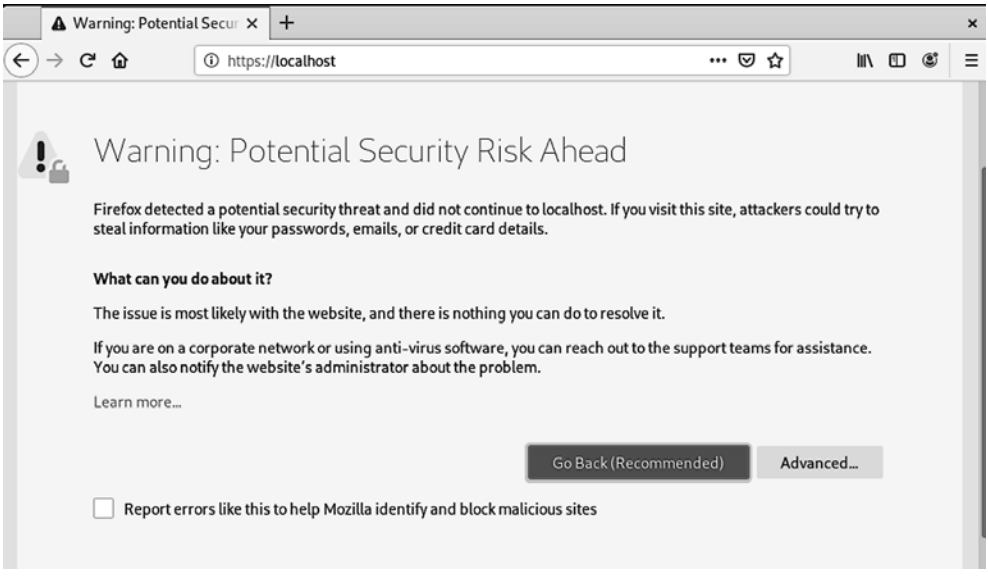


Рис. 17.2. Попытка перехода к сайту SSL с сертификатом по умолчанию

3. Поскольку вы получаете доступ к сайту через браузер на локальном хосте, нажмите кнопку Advanced (Дополнительно), а затем View (Просмотр), чтобы просмотреть сгенерированный сертификат. Сертификат включает в себя ваше имя хоста, информацию о том, когда он был выдан и когда истекает срок его действия, а также множество других сведений об организации.
4. Нажмите кнопку Accept the Risk and Continue (Принять риск и продолжить), чтобы разрешить подключение к этому сайту.
5. Закройте это окно, а затем выберите Confirm Security Exception (Добавить исключение безопасности), чтобы принять соединение. После этого откроется ваша веб-страница по умолчанию, использующая протокол HTTPS. Отныне ваш браузер будет принимать HTTPS-соединения с веб-сервером с помощью этого сертификата и шифровать все сообщения между собой и сервером.

Вы не хотите, чтобы ваш сайт отпугивал пользователей, поэтому стоит установить для него действительный сертификат. Лучше всего создать самоподписанный сертификат, который по крайней мере содержит достоверную информацию о ваших сайте и организации. В следующем пункте рассказывается, как это сделать.

Создание SSL-ключа и самоподписанного сертификата

Чтобы начать настройку SSL и создать открытый и закрытый ключи, примените команду `openssl`, которая является частью пакета `openssl`. После этого сгенерируйте самоподписанный сертификат, чтобы протестировать сайт или использовать его внутри частной системы.

1. Если пакет `openssl` еще не установлен, сделайте это следующим образом:

```
# yum install openssl
```

2. Сгенерируйте закрытый ключ `2048-bit RSA` и сохраните его в файл:

```
# cd /etc/pki/tls/private
# openssl genrsa -out server.key 2048
# chmod 600 server.key
```

ПРИМЕЧАНИЕ

Используйте имя файла, отличное от server.key, если планируете иметь на своем компьютере более одного SSL-хоста (для чего требуется более одного IP-адреса). В дальнейшем убедитесь, что указали правильное имя файла в конфигурации веб-сервера Apache.

В средах с более высоким уровнем безопасности используйте шифрование ключа, добавив аргумент `-des3` после аргумента `genrsa` в командной строке `openssl`. При появлении запроса на ввод ключевой фразы нажмите клавишу Enter:

```
# openssl genrsa -des3 -out server.key 1024
```

3. Если не планируете подписывать сертификат или хотите протестировать настройки, создайте самоподписанный сертификат и сохраните его в файле `server.crt` в каталоге `/etc/pki/tls/certs`:

```
# cd /etc/pki/tls/certs
# openssl req -new -x509 -nodes -sha1 -days 365 \
    -key /etc/pki/tls/private/server.key \
    -out server.crt
Country Name (2 letter code) [AU]: US
State or Province Name (full name) [Some-State]: NJ
Locality Name (eg, city) [Default City]: Princeton
Organization Name (eg, company) [Default Company Ltd
Ltd]:TEST USE ONLY
Organizational Unit Name (eg, section) []:TEST USE ONLY
Common Name (eg, YOUR name) []:secure.example.org
Email Address []:dom@example.org
```

4. Отредактируйте файл `/etc/httpd/conf.d/ssl.conf`, чтобы изменить местоположение ключа и сертификата на только что созданные, например:

```
SSLCertificateFile /etc/pki/tls/certs/server.crt
SSLCertificateKeyFile /etc/pki/tls/private/server.key
```

5. Перезапустите или перезагрузите службу `httpd`.
6. Снова откройте https://localhost в локальном браузере и повторите процедуру проверки, приняв новый сертификат.

Для внутреннего использования или тестирования подойдет и самоподписанный сертификат. Однако для общедоступных сайтов следует применять сертификат, проверенный центром сертификации (CA). Далее описано, как это сделать.

Создание запроса на получение сертификата

Если вы планируете получить сертификат, подписанный центром сертификации (включая тот, который запускаете сами), используйте свой закрытый ключ для создания запроса на получение сертификата (CSR).

1. Создайте каталог, в котором будет храниться CSR:

```
# mkdir /etc/pki/tls/ssl.csr
# cd /etc/pki/tls/ssl.csr/
```

2. С помощью команды `openssl` сгенерируйте CSR. В результате в текущем каталоге будет создан CSR-файл с именем `server.csr`. При вводе этой информации строка `Common Name` должна совпадать с именем, которое клиенты будут использовать для доступа к вашему серверу. Обязательно уточните остальные детали, чтобы проверить их сторонним CA. Кроме того, если вы задали кодовую фразу для своего ключа, необходимо ввести ее, чтобы применить ключ:

```
# openssl req -new -key ../private/server.key -out server.csr

Country Name (2 letter code) [AU]:US
```

```
State or Province Name (full name) [Some-State]:Washington
Locality Name (eg, city) []:Bellingham
Organization Name (eg, company) [Internet Widgits Pty
Ltd]:Example Company, LTD.
Organizational Unit Name (eg, section) []:Network
 Operations
Common Name (eg, YOUR name) []:secure.example.org
Email Address []:dom@example.org

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

3. Перейдите на сайт выбранного центра подписи сертификатов и запросите подпись для сертификата. Сайт CA, скорее всего, попросит вас скопировать содержимое своего CSR (в данном примере файл `server.csr`) и вставить в форму.
4. Когда CA отправит вам сертификат (возможно, по электронной почте), сохраните его в каталоге `/etc/pki/tls/certs/`, используя имя своего сайта, например, example.org.crt.
5. Измените значение `SSLCertificateFile` в файле `/etc/httpd/conf.d/ssl.conf` и укажите новый CRT-файл. Если у вас несколько хостов SSL, можете создать отдельную запись (возможно, в отдельном файле `.conf`), которая выглядит следующим образом:

```
Listen 192.168.0.56:443
<VirtualHost *:443>
    ServerName      secure.example.org
    ServerAlias     web.example.org
    DocumentRoot    /home/username/public_html/
    DirectoryIndex  index.php index.html index.htm
    SSLEngine       On
    SSLCertificateKeyFile /etc/pki/tls/private/server.key
    SSLCertificateFile /etc/pki/tls/certs/example.org.crt
</VirtualHost>
```

IP-адрес, показанный в директиве `Listen`, следует заменить общедоступным IP-адресом, представляющим обслуживаемый вами SSL-хост. Помните, что каждый SSL-хост должен иметь собственный IP-адрес.

Диагностика веб-сервера

В любой достаточно сложной среде всегда возникают проблемы. В следующих пунктах приведены советы по выявлению и устранению наиболее распространенных ошибок, с которыми вы можете столкнуться в ходе работы с веб-сервером.

Проверка ошибок конфигурации

Вы можете столкнуться с ошибками конфигурации или проблемами в скриптах, которые мешают запуску веб-сервера Apache или препятствуют доступу к определенным файлам. Большинство этих проблем можно обнаружить и решить с помощью двух инструментов, предоставляемых Apache: программы `apachectl` и журнала системных ошибок.

При возникновении проблемы сначала используйте программу `apachectl` с параметром `configtest` для проверки конфигураций. На самом деле стоит взять за правило выполнять это действие и запускать программу каждый раз, когда вы изменяете настройки:

```
# apachectl configtest
Syntax OK
# apachectl graceful
/usr/sbin/apachectl graceful: httpd gracefully restarted
```


В случае синтаксической ошибки программа `apachectl` указывает, где она возникает, и делает все возможное, чтобы дать информацию о характере проблемы. Затем используйте параметр перезапуска `graceful` (`apachectl graceful`), чтобы перезагрузить конфигурацию сервера Apache без отключения каких-либо активных клиентов.

ПРИМЕЧАНИЕ

Параметр перезапуска graceful в программе apachectl автоматически тестирует конфигурацию перед отправкой сигнала перезагрузки в apache. Но все же стоит обзавестись привычкой вручную тестировать конфигурации веб-сервера.

Некоторые из проблем конфигурации проходят синтаксические тесты программы `apachectl`, но выводят демон `http` из строя сразу же после перезагрузки его конфигурации. Если это произойдет, используйте команду `tail`, чтобы проверить журнал ошибок Apache на наличие полезной информации. В системах Fedora и RHEL журнал ошибок находится в файле `/var/log/httpd/error.log`. Чтобы найти местоположение журнала ошибок в других системах, обратитесь к директиве `ErrorLog` в вашей конфигурации Apache.

Вы можете столкнуться с сообщением об ошибке, которое выглядит примерно так:

```
[crit] (98)Address already in use: make_sock: could not bind to port 80
```

Эта ошибка указывает на то, что что-то еще соединено с портом 80, или другой процесс Apache уже запущен (`apachectl` перехватывает эту информацию), или Apache привязал один и тот же IP-адрес и комбинацию портов в нескольких местах сразу.

Используйте команду `netstat` для просмотра списка программ (включая Apache) с TCP-портами в состоянии `LISTEN`:

```
# netstat -nltp
Active  Internet connections (only servers)
Proto   Local Address  Foreign Address  State   PID/Program name
tcp6    :::80          :::*             LISTEN  2105/httpd
```

Выходные данные команды `netstat` (в примере сокращены) показывают, что экземпляр процесса `httpd` с идентификатором процесса ID 2105 прослушивает (как указано состоянием `LISTEN`) соединения с любым локальным IP-адресом (обозначен как `:::80`) на порте 80 (стандартный HTTP-порт). Если другая программа прослушивает порт 80, она тоже отображается. Используйте команду `kill` для немедленного завершения процесса, но если дело в чем-то другом, для начала стоит выяснить причину.

Если вы не видите никаких других процессов, прослушивающих порт 80, возможно, Apache по ошибке прослушивает одни и те же IP-адрес и комбинацию портов в нескольких местах сразу. Чтобы это исправить, можно применить три директивы конфигурации — `BindAddress`, `Port` и `Listen`.

- Директива `BindAddress` позволяет указать один IP-адрес для прослушивания или все IP-адреса с помощью шаблона поиска (wildcard). В файле конфигурации не должно быть больше одного оператора `BindAddress`.
- Директива `Port` указывает, на каком TCP-порте следует прослушивать, но не позволяет указать IP-адрес. Директива `Port`, как правило, не применяется более одного раза в конфигурации.
- Директива `Listen` позволяет указать как IP-адрес, так и порт для привязки. IP-адрес может представлять собой шаблон поиска, и в файле конфигурации могут находиться несколько операторов `Listen`.

Чтобы избежать путаницы, стоит задействовать только одну из этих директив. Из трех вариантов `Listen` наиболее гибкая, поэтому, вероятно, с ней вы будете работать чаще всего. Распространенная ошибка при использовании директивы `Listen` заключается в том, что указание порта на всех IP-адресах (`*: 80`) и того же порта на определенном IP-адресе (`1.2.3.4:80`) вызывает ошибку `make_sock`.

Ошибки конфигурации, связанные с SSL, обычно приводят к неправильному запуску веб-сервера Apache. Убедитесь, что все файлы ключей и сертификатов существуют и имеют правильный формат (проверяйте их с помощью команды `openssl`).

Получив другие сообщения об ошибках, попробуйте узнать в Интернете, не сталкивался ли кто-то еще с подобной проблемой. В большинстве случаев решение можно найти в нескольких первых ссылках в поиске.

Если в директиве `ErrorLog` недостаточно информации, настройте ее для регистрации дополнительной информации с помощью директивы `LogLevel`. Параметры, доступные для нее: `emerg`, `alert`, `crit`, `error`, `warn`, `notice`, `info` и `debug`. Выберите только один из них.

Любое сообщение, такое же важное, как и выбранный `LogLevel`, хранится в `ErrorLog`. На обычном сервере `LogLevel` имеет значение `warn`. Не нужно устанавливать его на любое значение ниже `crit`, а также оставлять его со значением `debug`, так как это может замедлить работу сервера и увеличить файл `ErrorLog`.

В крайнем случае попробуйте запустить службу `httpd -X` вручную, чтобы проверить наличие сбоев или других сообщений об ошибках. Параметр `-X` запускает службу `httpd` так, чтобы она выводила на экран сообщения отладки и более высокого уровня.

Запрещенный доступ и внутренние ошибки сервера

Два распространенных типа ошибок, с которыми можно столкнуться при просмотре определенных страниц на вашем сервере, — это ошибки прав доступа и внутренние ошибки сервера. Оба типа ошибок обычно можно выявить, используя информацию в журнале ошибок. После внесения любых изменений, описанных в списке далее, для решения одной из этих проблем повторите попытку запроса и проверьте журнал ошибок, чтобы увидеть, изменилось ли сообщение (например, чтобы увидеть, что операция успешно завершена).

ПРИМЕЧАНИЕ

Ошибки типа File not found (Файл не найден) можно проверить так же, как и ошибки Access forbidden (Доступ запрещен) и Server internal errors (Внутренние ошибки сервера). Вы можете обнаружить, что веб-сервер Apache ищет конкретный файл не там, где нужно. Как правило, весь путь к файлу отображается в журнале ошибок. Убедитесь, что используете правильный виртуальный хост, и проверьте все настройки параметра Alias, которые могут изменить направление поиска.

- **Права файлов.** Ошибка File permissions prevent access (Права файлов запрещают доступ) указывает на то, что процесс `apache` работает от имени пользователя, который не может открыть запрошенный файл. По умолчанию служба `httpd` запускается пользователем и группой Apache. Убедитесь, что у учетной записи есть права на выполнение каталога и всех каталогов более высокого уровня, а также права на чтение самих файлов. Права на чтение каталога необходимы и в том случае, если вы хотите, чтобы Apache генерировал индексы файлов. Дополнительные сведения о том, как просматривать и изменять права, см. на странице руководства команды `chmod`.

ПРИМЕЧАНИЕ

Права на чтение не требуются для скомпилированных двоичных файлов, например написанных на языках C и C++, но и их можно добавлять на сервер, если нет необходимости хранить содержимое программы в секрете.

- **Доступ запрещен.** Ошибка Client denied by server configuration (Доступ клиента запрещен конфигурацией сервера) указывает на то, что веб-сервер Apache настроен на отказ в доступе к файлу. Проверьте разделы `Location` и `Directory`, которые могли повлиять на файл, в доступе к которому отказано. Помните, что настройки пути применяются и к любым путям низшего уровня. Вы можете переопределить их, изменив права только для конкретного пути, к которому хотите разрешить доступ.
- **Индекс не найден.** Ошибка Directory index forbidden by rule (Индекс каталога запрещен правилами) указывает на то, что веб-сервер Apache не смог найти индексный файл с именем, указанным в директиве `DirectoryIndex`, и не будет создавать индекс, содержащий список файлов в каталоге. Убедитесь, что индексная страница, если она у вас есть, имеет одно из имен, указанных в соответствующей директиве `DirectoryIndex`, или добавьте строку `Options Indexes` в соответствующий раздел каталога или местоположения для нее.
- **Ошибка скрипта.** Ошибка Premature end of script headers (Преждевременное завершение заголовков скрипта) может указывать на сбой скрипта до его завершения. Иногда подобные ошибки появляются и в журнале ошибок. При использовании команд `suexec` или `suPHP` она может быть вызвана ошибкой с владением файлом или правами доступа. Такие ошибки появляются в файлах журналов в каталоге `/var/log/httpd`.
- **Ошибки SELinux.** Если права доступа к файлам открыты, но в файлах журналов появляются сообщения об отказе в доступе, то причиной проблемы может быть SELinux. Временно установите SELinux в разрешительный режим (`setenforce 0`) и снова попробуйте получить доступ к файлу. Если файл теперь доступен, вновь установите SELinux в принудительный режим (`setenforce 1`) и проверьте контексты файлов и логические типы. Контексты файлов должны быть указаны правильно, чтобы служба `httpd` могла получить доступ к файлу. Логические типы могут препятствовать обслуживанию файла из удаленно смонтированного каталога, или отправке страницы по электронной почте, или загрузке файла. Введите команду `httpd_selinux` для получения подробной информации о параметрах конфигурации SELinux, связанных со службами `httpd`. (Установите пакет `selinux-policy-devel`, чтобы добавить эту справочную страницу в свою систему.)

Резюме

Проект Apache с открытым исходным кодом — это самый популярный веб-сервер в мире. Хотя он невероятно гибок, безопасен и сложен, базовый веб-сервер Apache можно настроить всего за несколько минут в Fedora, RHEL и большинстве других дистрибутивов Linux.

В этой главе описаны шаги по установке, настройке, защите и устранению неполадок базового веб-сервера Apache. Вы узнали, как настроить виртуальный хо-

стинг и защитить SSL-хосты. Прочитали, как настроить веб-сервер Apache, чтобы позволить любой учетной записи пользователя в системе публиковать содержимое из собственного каталога `public_html`.

Продолжая тему конфигурации сервера, в главе 18 «Настройка FTP-сервера» я расскажу, как настроить FTP-сервер в системе Linux. В приведенных примерах будет показано, как сделать это с помощью пакета `vsftpd`.

Упражнения

Упражнения, приведенные далее, охватывают темы, связанные с установкой и настройкой веб-сервера Apache. Как обычно, я рекомендую использовать отдельную систему Fedora или Red Hat Enterprise Linux для их выполнения. Не выполняйте их на рабочем компьютере, так как они изменяют файлы конфигурации Apache и служб и могут повредить уже настроенные службы. Примените для тренировки виртуальную машину или другой компьютер.

Предполагается, что при выполнении упражнений вы начнете с установки системы Fedora или RHEL, на которой еще нет сервера Apache (пакет `httpd`).

Если затрудняетесь с решением заданий, воспользуйтесь ответами к ним, приведенными в приложении Б (хотя Linux позволяет решать задачи разными способами).

1. Из системы Fedora установите все пакеты, связанные с группой Basic Web Server.
2. Создайте файл с именем `index.html` в каталоге, назначенном директивой `DocumentRoot` в главном файле конфигурации Apache. В файле должна быть фраза `My Own Web Server`.
3. Запустите веб-сервер Apache и настройте его на автоматический запуск во время загрузки. Убедитесь, что он доступен из браузера на вашем локальном хосте. (Если он работает правильно, вы увидите фразу: `My Own Web Server`.)
4. Используйте команду `netstat`, чтобы увидеть, на каких портах прослушивается `httpd`-сервер.
5. Попробуйте подключиться к веб-серверу Apache из браузера, находящегося за пределами локальной системы. Если это не удается, исправьте все появившиеся проблемы, изучив настройки межсетевого экрана (брандмауэра), SELinux и других функций безопасности.
6. С помощью команды `openssl` или аналогичной создайте собственный закрытый ключ RSA и самоподписанный SSL-сертификат.
7. Настройте веб-сервер Apache так, чтобы он использовал ваши ключ и самоподписанный сертификат для обслуживания защищенного содержимого (HTTPS).
8. Используйте браузер, чтобы создать HTTPS-соединение со своим веб-сервером и просмотреть содержимое созданного вами сертификата.

9. Создайте файл с именем `/etc/httpd/conf.d/example.org.conf`, который включает виртуальный хостинг на основе имен и создает виртуальный хост, который должен:
   - прослушивать все интерфейсы на порте 80;
   - иметь администратора сервера `joe@example.org`;
   - иметь имя сервера `joe@example.org`;
   - иметь директиву `DocumentRoot` в каталоге `/var/www/html/example.org`;
   - иметь директиву `DirectoryIndex`, которая включает в себя как минимум файл `index.html`.
10. В каталоге `DocumentRoot` создайте файл `Index.html`, содержащий фразу: `Welcome to the House of Joe`. Добавьте текст `joe.example.org` в конец строки `localhost` в файле `/etc/hosts` на компьютере, на котором запущен веб-сервер. Затем введите в браузере `joe.example.org`. На первой странице вы должны увидеть фразу `Welcome to the House of Joe`.

18 Настройка FTP-сервера

В этой главе

- Как работает сервер FTP.
- Установка пакета `vsftpd`.
- Настройки безопасности в пакете `vsftpd`.
- Настройка файлов конфигурации в пакете `vsftpd`.
- Запуск клиентов сервера FTP.

Протокол передачи файлов (File Transfer Protocol, FTP) — один из старейших протоколов обмена файлами по сетям. Несмотря на то что существуют более безопасные протоколы для передачи файлов по сети, FTP по-прежнему довольно часто используется, обеспечивая свободный доступ к файлам в Интернете.

Сейчас в системах Linux доступны несколько проектов FTP-серверов. Однако чаще всего с Fedora, Red Hat Enterprise Linux, CentOS, Ubuntu и другими дистрибутивами Linux применяется безопасный демон FTP (пакет `vsftpd`). В этой главе рассказывается, как установить, настроить, использовать и защитить FTP-сервер с помощью пакета `vsftpd`.

Сервер FTP

Протокол FTP задействует модель «клиент/сервер». Демон FTP-сервера прослушивает входящие запросы (через TCP-порт 21) от FTP-клиентов. Клиент предъявляет логин и пароль. Если сервер принимает эту информацию, клиент может перемещаться по файловой системе, перечислять файлы и каталоги и загружать файлы.

Небезопасным FTP делает то, что между FTP-клиентом и сервером все передается открытым текстом. Этот протокол был создан в то время, когда большинство

компьютерных коммуникаций осуществлялось по частным линиям или коммутируемому каналу, в шифровании которых не было необходимости. Если вы используете сервер FTP через общедоступную сеть, в любом месте между клиентом и сервером злоумышленник сможет увидеть не только передаваемые данные, но и данные аутентификации (информацию о логине и пароле).

Таким образом, сервер FTP не подходит для частного обмена файлами (используйте SSH-команды, такие как `sftp`, `scp` или `rsync`, чтобы зашифровать частную передачу файлов). Но если вы делитесь общедоступными документами, репозиториями программного обеспечения с открытым исходным кодом или другими данными, сервер FTP отлично подойдет. Независимо от операционной системы, которую применяют другие пользователи, у них будет приложение для получения файлов FTP с вашего FTP-сервера.

Когда пользователи проходят аутентификацию на FTP-сервере в Linux, их имена и пароли проверяются на соответствие стандартным учетным записям и паролям в Linux. Существует также специальная неаутентифицированная учетная запись anonymous для FTP-сервера. Доступ к анонимной учетной записи может получить любой желающий, поскольку для нее не требуется пароль. На самом деле термин *«анонимный FTP-сервер»* часто используется для описания общедоступного FTP-сервера, который не требует аутентификации с помощью учетной записи или даже не разрешает ее.

ПРИМЕЧАНИЕ

Возможность входа на сервер vsftpd с помощью обычной учетной записи пользователя Linux включена по умолчанию в системах Fedora и Red Hat Enterprise Linux, но, если система SELinux установлена в принудительный режим, она предотвращает вход и передачу файлов. Чтобы сохранить SELinux в принудительном режиме, но попрежнему разрешать вход, измените логические типы (см. раздел «Настройка SELinux для FTP-сервера» далее в этой главе).

После аутентификации (на управляющем порте TCP 21) между клиентом и сервером устанавливается второе соединение. FTP поддерживает как активные, так и пассивные типы соединений. При активном FTP-соединении сервер отправляет данные со своего порта TCP 20 на какой-то случайный порт, номер которого выбирает выше порта 1023 на клиенте. При пассивном FTP-соединении клиент запрашивает пассивное соединение, а затем случайный порт у сервера.

Многие браузеры поддерживают пассивный режим FTP, поэтому, если у клиента есть брандмауэр (межсетевой экран), он не блокирует порт данных, который FTP-сервер может использовать в активном режиме. Поддержка пассивного режима требует дополнительной настройки брандмауэра сервера, которая разрешит случайные подключения к портам выше 1023-го на сервере. В разделе «Настройка брандмауэра для FTP-сервера» далее в этой главе описывается, что нужно сделать с брандмауэром Linux, чтобы заставить работать как пассивные, так и активные FTP-соединения.

После установки соединения между клиентом и сервером устанавливается текущий каталог клиента. Для анонимного пользователя каталог `/var/ftp` является

домашним каталогом в системах Fedora или RHEL, а `/srv/ftp` — в системе Ubuntu и большинстве дистрибутивов на базе Debian. Анонимный пользователь не может выйти за пределы структуры каталогов `/var/ftp`. Если обычный пользователь, скажем, `joe`, входит на FTP-сервер, то `/home/joe` — это его текущий каталог, но он может перейти в любую часть файловой системы, права на которые у него есть.

Командно-ориентированные FTP-клиенты (например, команды `lftp` и `ftp`) переходят в интерактивный режим после подключения к серверу. Из командной строки можете запустить множество команд, похожих на те, которые вы использовали в оболочке. Можно задействовать команду `pwd`, чтобы увидеть текущий каталог, `ls`, чтобы просмотреть содержимое каталога, и `cd`, чтобы изменить каталоги. Когда вы увидите нужный файл, примените команды `get` и `put` для загрузки файлов с сервера или на сервер соответственно.

Используя графические инструменты для доступа к FTP-серверам (например, браузер), введите URL-адрес сайта, на который хотите перейти (допустим, ftp://docs.example.com). Если не вводить имя пользователя и пароль, то будет установлено анонимное соединение и отобразится содержимое домашнего каталога сайта. Щелкните на ссылках с каталогами, чтобы перейти к ним. Нажмите на ссылки с файлами, чтобы отобразить или загрузить эти файлы в локальную систему.

Поняв, как работает сервер FTP, можно приступить к установке FTP-сервера (пакет `vsftpd`) в своей системе Linux.

Установка сервера FTP с помощью пакета vsftpd

Для настройки безопасного FTP-сервера в Fedora, RHEL и других дистрибутивах Linux требуется только один пакет — `vsftpd`. Если у вас есть доступ к репозиторию программного обеспечения Fedora или RHEL, просто введите следующую команду от имени суперпользователя, чтобы установить `vsftpd`:

```
# yum install vsftpd
```

Если вы работаете с дистрибутивом Ubuntu (или другим дистрибутивом Linux, основанным на Debian), для установки `vsftpd` введите следующее:

```
$ sudo apt-get install vsftpd
```

Далее представлен пример команд, которые можно запустить после установки пакета `vsftpd`, чтобы ознакомиться с его содержимым. Из дистрибутива Fedora или RHEL выполните следующую команду:

```
# rpm -qi vsftpd
...
Packager    : Fedora Project
Vendor      : Fedora Project
URL         : https://security.appspot.com/vsftpd.html
Summary     : Very Secure Ftp Daemon
Description : vsftpd is a Very Secure FTP daemon. It was written
              completely from scratch.
```

Если хотите получить более подробную информацию о пакете vsftpd, перейдите на сайт security.appspot.com/vsftpd.html. Здесь есть также дополнительная документация и информация о последних версиях vsftpd.

С помощью команд можно просмотреть полное содержимое пакета vsftpd (rpm-ql vsftpd), только файлы документации (-qd) или файлы конфигурации (-qc). Для просмотра файлов документации используйте следующую команду:

```
# rpm -qd vsftpd
/usr/share/doc/vsftpd/EXAMPLE/INTERNET_SITE/README
...
/usr/share/doc/vsftpd/EXAMPLE/PER_IP_CONFIG/README
...
/usr/share/doc/vsftpd/EXAMPLE/VIRTUAL_HOSTS/README
/usr/share/doc/vsftpd/EXAMPLE/VIRTUAL_USERS/README
...
/usr/share/doc/vsftpd/FAQ
...
/usr/share/doc/vsftpd/vsftpd.xinetd
/usr/share/man/man5/vsftpd.conf.5.gz
/usr/share/man/man8/vsftpd.8.gz
```

В структуру каталогов /usr/share/doc/vsftpd/EXAMPLE включены примеры файлов конфигурации, которые помогут настроить пакет vsftpd способами, подходящими для интернет-сайта, сайта с несколькими IP-адресами и виртуальных хостов. Основной каталог /usr/share/doc/vsftpd содержит FAQ (часто задаваемые вопросы), советы по установке и информацию о версии.

На справочных страницах пакета vsftpd, возможно, найдется самая полезная информация о настройке сервера. Введите команду man vsftpd.conf, чтобы прочитать о файле конфигурации, и man vsftpd, чтобы прочитать о процессе демона и о том, как управлять им в качестве службы systemd.

Чтобы перечислить файлы конфигурации, введите следующее:

```
# rpm -qc vsftpd
/etc/logrotate.d/vsftpd
/etc/pam.d/vsftpd
/etc/vsftpd/ftpusers
/etc/vsftpd/user_list
/etc/vsftpd/vsftpd.conf
```

Главный файл конфигурации — это /etc/vsftpd/vsftpd.conf (в RHEL и Fedora) и /etc/vsftpd.conf (в Ubuntu). Файлы ftpusers и user_list (в Fedora и RHEL, но не в Ubuntu) в одном каталоге хранят информацию об учетных записях пользователей, которым ограничен доступ к серверу. Файл /etc/pam.d/vsftpd задает способ аутентификации на FTP-сервере. Файл /etc/logrotate.d/vsftpd настраивает способ ротации файлов журналов с течением времени.

Вы установили пакет vsftpd и быстро ознакомились с его содержимым. Следующий шаг — запустить и протестировать службу vsftpd.

Запуск службы vsftpd

Для запуска службы `vsftpd` не требуется никаких настроек, если вам подходят настройки по умолчанию. Запустив службу `vsftpd` с настройками по умолчанию в дистрибутиве Fedora, вы получите следующее.

- Служба `vsftpd` запускает демон `vsftpd`, который работает в фоновом режиме.
- Стандартный порт, который демон `vsftpd` прослушивает, — это порт TCP 21. По умолчанию данные передаются пользователю после установки соединения через порт TCP 20. Порт TCP 21 должен быть открыт в брандмауэре, чтобы новые соединения получили доступ к службе. По умолчанию доступны как IPv4-, так и IPv6-соединения. Эта процедура изменяется на службу TCP IPv4. (См. раздел «Защита FTP-сервера» далее в этой главе, чтобы получить подробную информацию об открытии портов, отслеживании соединений, необходимых для пассивного FTP, и настройке других правил брандмауэра, связанных с FTP.)
- Демон `vsftpd` считывает файл `vsftpd.conf`, чтобы определить, какие функции служба разрешает.
- Учетные записи пользователей Linux (за исключением администраторов) могут получить доступ к FTP-серверу. Можно подключить анонимную учетную запись пользователя (пароль не требуется). (Если SELinux находится в принудительном режиме, вам нужно установить логический тип, чтобы разрешить обычным пользователям входить на FTP-сервер. Подробнее см. раздел «Защита FTP-сервера».)
- Анонимный пользователь имеет доступ только к каталогу `/var/ftp` и его подкаталогам. Обычный пользователь в качестве текущего каталога применяет свой домашний каталог, но может получить доступ к любому доступному каталогу через обычный логин или сеанс SSH. Списки пользователей в файлах `/etc/vsftpd/user_list` и `/etc/vsftpd/ftpusers` определяют администраторов и специальных пользователей, которые не имеют доступа к FTP-серверу (`root`, `bin`, `daemon` и др.).
- По умолчанию анонимный пользователь может загружать файлы с сервера, но не загружать их в сервер. Обычный пользователь может загружать или скачивать файлы, основываясь на обычных правах Linux.
- Сообщения журнала с деталями загрузки файлов записываются в файл `/var/log/xferlogs`. Эти сообщения хранятся в стандартном формате `xferlog`.

Чтобы запустить сервер, используя описанные настройки по умолчанию, следуйте приведенной далее инструкции. Если вы хотите сначала изменить дополнительные настройки, перейдите в раздел «Настройка FTP-сервера» далее в этой главе, завершите настройку параметров, а затем вернитесь к инструкции.

1. Проверьте службу vsftpd. Прежде чем запустить службу vsftpd, посмотрите, не работает ли она. В системе Fedora или Red Hat Enterprise Linux 7 или 8 выполните следующее:

```
# systemctl status vsftpd.service
vsftpd.service – Vsftpd ftp daemon
       Loaded: loaded (/usr/lib/systemd/system/vsftpd.service; disabled)
       Active: inactive (dead)
```

   В дистрибутиве Red Hat Enterprise Linux 6, чтобы увидеть ту же информацию, нужны две команды:

```
# service vsftpd status
vsftpd is stopped
# chkconfig --list vsftpd
vsftpd 0:off 1:off 2:off 3:off 4:off 5:off 6:off
```

   В приведенных примерах команды service, chkconfig и systemctl отображают состояние службы — stopped. Видно также, что она совсем отключена (в Fedora и RHEL 7 или 8) и выключена на каждом уровне запуска (в RHEL 6). Значение off означает, что служба не будет включаться автоматически при запуске системы.

2. Чтобы запустить и включить службу vsftpd в дистрибутиве Fedora или RHEL 7 или 8 (и проверить состояние), введите следующее:

```
# systemctl start vsftpd.service
# systemctl enable vsftpd.service
ln -s '/lib/systemd/system/vsftpd.service'
    '/etc/systemd/system/multi-user.target.wants/vsftpd.service'
# systemctl status vsftpd.service
vsftpd.service – Vsftpd ftp daemon
        Loaded: loaded (/usr/lib/systemd/system/vsftpd.service;
            enabled vendor preset: disabled))
        Active: active (running) since Wed, 2019-09-18 00:09:54 EDT; 22s
ago
        Main PID: 4229 (vsftpd)
            Tasks: 1 (limit: 12232)
        Memory: 536.0K
            CGroup: /system.slice/vsftpd.service
         └ 4229 /usr/sbin/vsftpd /etc/vsftpd/vsftpd.conf
```

   В дистрибутиве Red Hat Enterprise Linux 6 запустите и включите (on) службу vsftpd (затем проверьте состояние) следующим образом:

```
# service vsftpd start
Starting vsftpd for vsftpd:                          [ OK ]
# chkconfig vsftpd on ; chkconfig --list vsftpd
vsftpd       0:off    1:off    2:on     3:on    4:on    5:on    6:off
```

3. Теперь в любой системе проверьте, работает ли служба, используя команду netstat:

```
# netstat -tupln | grep vsftpd
tcp  0   0 0.0.0.0:21   0.0.0.0:*   LISTEN     4229/vsftpd
```

Из выходных данных команды `netstat` видно, что процесс `vsftpd` (идентификатор процесса ID 4229) прослушивает (`LISTEN`) все IP-адреса для входящих соединений на порте 21 (0.0.0.0:21) для протокола TCP (`tcp`).

4. Быстрый способ проверить, что служба `vsftpd` работает, — это поместить файл в каталог `/var/ftp` и попытаться открыть его из вашего браузера на локальном хосте:

```
# echo "Hello From Your FTP Server" > /var/ftp/hello.txt
```

В браузере локальной системы (в браузере Firefox или другом) введите следующий адрес:

```
ftp://localhost/hello.txt
```

Если в браузере появляется текст Hello From Your FTP Server, значит, сервер `vsftpd` работает и доступен из вашей локальной системы. Затем подключитесь из браузера в другой системе, заменив `localhost` IP-адресом вашего хоста или полным именем хоста.

Если это сработает, сервер `vsftpd` станет общедоступным. Если нет, что вполне возможно, прочитайте раздел «Защита FTP-сервера». В нем рассказывается, как открыть брандмауэры и изменить другие функции безопасности, чтобы разрешить доступ и иным способом защитить ваш FTP-сервер.

Защита FTP-сервера

Несмотря на то что запустить FTP-сервер `vsftpd` легко, это не означает, что он сразу же будет полностью доступен. Если в вашей системе есть брандмауэр, он, вероятно, заблокирует доступ ко всем службам в системе, за исключением тех, которые были разрешены вручную.

Если вы решите, что настройка службы `vsftpd` по умолчанию (описана в предыдущем разделе) вам подходит, то можете просто настроить брандмаэур, разрешив доступ и обеспечив безопасность для службы `vsftpd`. В следующих разделах мы рассмотрим, как настроить брандмауэр и SELinux (логические типы и файловые контексты), чтобы защитить сервер `vsftpd`.

Настройка брандмауэра для FTP-сервера

Если в системе есть брандмауэр, необходимо установить для него правила, которые разрешают входящие запросы на FTP-сервер и позволяют пакетам возвращаться в систему по установленным соединениям. Брандмауэры реализуются с использованием правил `iptables` и управляются с помощью службы `iptables` или `firewalld` (подробнее о службах брандмауэра см. главу 25 «Защита Linux в Сети»).

В дистрибутивах Fedora и Red Hat Enterprise Linux правила брандмауэра традиционно хранятся в файле `/etc/sysconfig/iptables`, а базовая служба — это

`iptables` (в RHEL 6) и `iptables.service` (в Fedora). Модули загружаются в брандмауэр из файла `/etc/sysconfig/iptables-config`. В системах RHEL 7 и Fedora 21 или более поздних версиях новая служба `firewalld` управляет правилами, а сами они хранятся в каталоге `/etc/firewalld/zones`.

ПРИМЕЧАНИЕ

Работать с брандмауэром лучше всего непосредственно из системной консоли, а не через удаленный вход (например, с помощью ssh), потому что одна небольшая ошибка может немедленно отрезать вас от вашего сервера. После этого нужно будет переходить к консоли, чтобы вернуться на сервер и устранить проблему. Добавьте правила для брандмауэра, чтобы он разрешал доступ к FTP-серверу, не открывая доступ к другим службам. Сначала нужно разрешить вашей системе принимать запросы на TCP-порт 21, затем убедиться, что модуль отслеживания соединений загружен.

В системах RHEL 7 и Fedora 20 или более поздних версиях можно использовать окно Firewall Configuration (Конфигурации брандмауэра), чтобы включить брандмауэр и открыть доступ к службе FTP. Установите пакет `firewall-config` и запустите команду `firewall-config`, чтобы открыть окно Firewall Configuration (Конфигурации брандмауэра) (рис. 18.1).

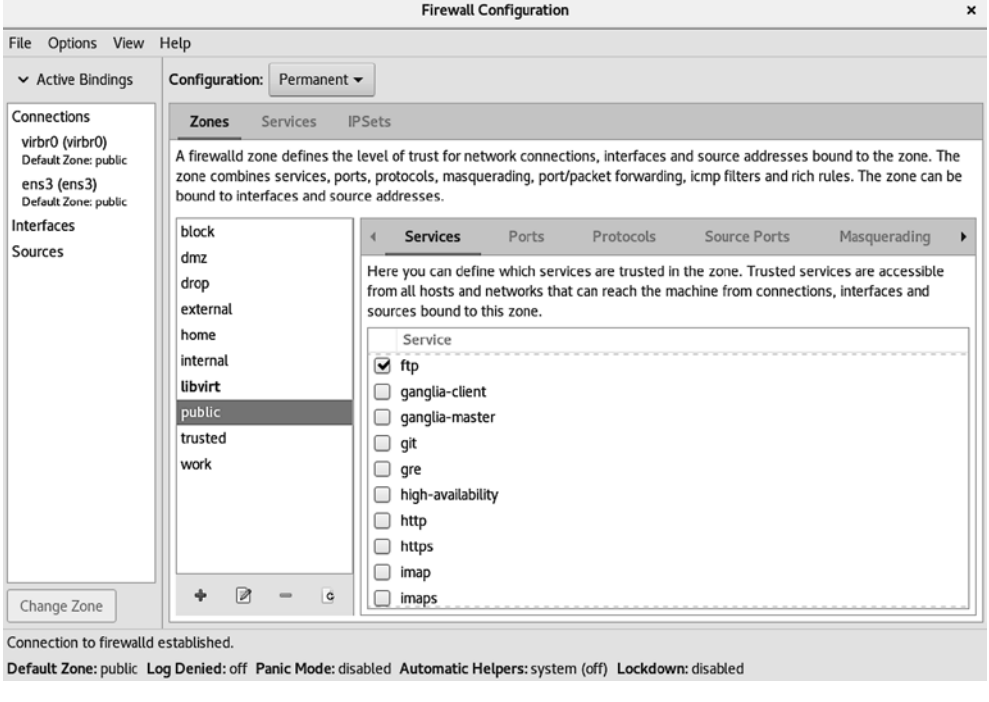


Рис. 18.1. Откройте доступ к FTP-службе в окне Firewall Configuration (Конфигурации брандмауэра)

Чтобы открыть постоянный доступ к FTP-службе, щелкните на поле Configuration (Конфигурация) и выберите вариант Permanent (Постоянная). Затем установите флажок ftp на вкладке Services (Службы). В брандмауэре автоматически откроется TCP-порт 21 (FTP) и загрузит модули ядра, необходимые для обеспечения доступа к пассивной службе FTP. Выберите Options ▸ Reload Firewalld (Параметры ▸ Перезагрузить Firewalld), чтобы раз и навсегда применить правило брандмауэра.

Для RHEL 6 и более ранних систем добавьте правила непосредственно в файл `/etc/sysconfig/iptables`. Если вы используете брандмауэр с настройками по умолчанию, первые правила открывают доступ к запросам на любые службы, поступающие с локального хоста, и позволяют получать пакеты, связанные с установленными соединениями или относящиеся к ним. В середине находятся правила, открывающие порты для запросов на обслуживание, которые уже разрешены, например, служба secure shell (`sshd` на TCP-порте 22). В конце находится последнее правило, которое обычно отбрасывает (`DROP`) или отклоняет (`REJECT`) любой неразрешенный запрос.

Чтобы дать публичный доступ на ваш FTP-сервер тому, кто его запрашивает, необходимо разрешить новые запросы к TCP-порту 21. Обычно это правило добавляется где-то перед окончательным правилом `DROP` или `REJECT`. В следующем примере показана часть содержимого файла `/etc/sysconfig/iptables` с правилом, разрешающим доступ к FTP-серверу (выделено полужирным):

```
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 22 -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 21 -j ACCEPT
...
-A INPUT -j REJECT --reject-with icmp-host-prohibited
COMMIT
```

В примере видно, что для таблицы фильтров брандмауэр принимает пакеты от установленных подключений, подключений от локальных хостов и любых новых запросов на TCP-порт 22 (служба SSH). Строка, которую мы только что добавили (`--dport 21`), позволяет принимать любые пакеты на новых соединениях с TCP-портом 21.

ПРИМЕЧАНИЕ

Важно установить строку ESTABLISHED, RELATED в правилах брандмауэра iptables. Без нее пользователи смогли бы подключиться к службам SSH (порт 22) и FTP (порт 21), то есть пройти аутентификацию, но не смогли бы передавать данные.

Следующее, что нужно сделать в RHEL 6 и более ранних системах, — настроить модуль отслеживания FTP-соединений, который будет загружаться каждый

раз при запуске брандмауэра. Отредактируйте эту строку в начале файла `/etc/sysconfig/iptables-config`, чтобы она выглядела следующим образом:

```
IPTABLES_MODULES="nf_conntrack_ftp"
```

На этом этапе можно перезапустить брандмауэр, помня о том, что ошибка может заблокировать вас, если вы вошли в систему удаленно. Используйте одну из двух следующих команд для перезапуска брандмауэра в зависимости от того, применяет ли ваша система более старую службу `iptables` или новую `firewalld`:

```
# service iptables restart
```

или

```
# systemctl restart firewalld.service
```

Попробуйте еще раз получить доступ к FTP-серверу из удаленной системы с помощью браузера или другого FTP-клиента.

Настройка SELinux для FTP-сервера

Если служба SELinux установлена в разрешительный режим или отключена, то она никоим образом не будет блокировать доступ к службе `vsftpd`. Однако, если SELinux находится в принудительном режиме, это может привести к тому, что ваш сервер `vsftpd` будет вести себя не так, как хотелось бы. Используйте следующие команды для проверки состояния SELinux в своей системе:

```
# getenforce
Enforcing
# grep ^SELINUX= /etc/sysconfig/selinux
SELINUX=enforcing
```

Команда `getenforce` показывает, в какой режим установлена служба SELinux. (В примере она находится в принудительном режиме.) Переменная `SELINUX=` в файле `/etc/sysconfig/selinux` показывает, в какой режим устанавливается SELinux при запуске системы. Если она находится в принудительном режиме, как в примере, проверьте справочную страницу `ftpd_selinux` для получения информации о настройках SELinux, которые могут повлиять на работу вашей службы `vsftpd`. Установите пакет `selinux-policy-doc`, чтобы получить справочную страницу `ftpd_selinux`, а также справочные страницы для других служб, связанных с политикой SELinux.

Примеры контекстов файлов, которые следует установить для SELinux, чтобы разрешить доступ к файлам и каталогам `vsftpd`.

- Чтобы поделиться содержимым так, чтобы его можно было загрузить на FTP-клиенты, он должен быть помечен контекстом файла `public_content_t`. Файлы, созданные в каталоге `/var/ftp` или в его подкаталогах, автоматически наследуют контекст файла `public_content_t`. (Обязательно создайте новое содержимое или скопируйте существующее в каталоги `/var/ftp`. Если файлы туда просто переместить, контекст может не измениться должным образом.)

- Чтобы разрешить загрузку файлов анонимным пользователям, контекст файла в каталоге, в который вы загружаете файлы, должен быть установлен в значение `public_content_rw_t`. (Для этого должны быть установлены также другие разрешения, логические типы SELinux и настройки `vsftpd.conf`.)

Если у вас в структуре каталогов `/var/ftp` есть файлы с неправильными контекстами (что может произойти, если переместить туда файлы из других каталогов вместо их копирования), можете изменить или восстановить в них контекст файлов, чтобы они могли быть общими. Например, для рекурсивного изменения контекста файла каталога `/var/ftp/pub/stuff` так, чтобы содержимое можно было читать с FTP-сервера через SELinux, введите следующее:

```
# semanage fcontext -a -t public_content_t "/var/ftp/pub/stuff(/.*)?"
# restorecon -F -R -v /var/ftp/pub/stuff
```

Если вы хотите разрешить пользователям записывать файлы в каталог и читать из него, нужно назначить контекст файла `public_content_rw_t` каталогу загрузки. В примере далее команда указывает SELinux разрешить загрузку файлов в каталог `/var/ftp/pub/uploads`:

```
# semanage fcontext -a -t public_content_rw_t\
   "/var/ftp/pub/uploads(/.*)?"
# restorecon -F -R -v /var/ftp/pub/uploads
```

Функции FTP-сервера, которые SELinux считает небезопасными, имеют логические типы, позволяющие разрешить или запретить эти функции, например.

- Чтобы SELinux разрешал анонимным пользователям читать и записывать файлы и каталоги, необходимо включить логический тип `allow_ftpd_anon_write` (в RHEL 6) или `ftpd_anon_write` (в RHEL 7 или более поздней версии):

  ```
  # setsebool -P ftpd_anon_write on
  ```

- Чтобы иметь возможность монтировать удаленные общие файловые системы NFS или CIFS (Windows) и совместно использовать их из вашего сервера `vsftpd`, необходимо включить следующие два логических типа соответственно:

  ```
  # setsebool -P allow_ftpd_use_nfs on
  # setsebool -P allow_ftpd_use_cifs on
  ```

Если вы обнаружите, что не можете получить доступ к файлам или каталогам с вашего FTP-сервера, которые должны быть доступны, попробуйте временно отключить службу SELinux:

```
# setpenforce 0
```

Если сможете получить доступ к файлам или каталогам с SELinux в разрешительном режиме, верните систему в принудительный режим (`setenforce 1`). Таким образом можно выяснить, что проблема заключается в настройках SELinux. (См. главу 24 «Повышенная безопасность с технологией SELinux» для получения дополнительной информации о SELinux.)

Права доступа к файлам в службе vsftpd

Сервер `vsftpd` применяет стандартные права Linux для разрешения или запрета доступа к файлам и каталогам. Анонимному пользователю, чтобы просмотреть или загрузить файл, нужно по крайней мере право на чтение. Оно должно быть открыто для `other` (`------r--`). Чтобы получить доступ к каталогу, необходимы права на выполнение `execute`, также в значении для `other` (`--------x`).

Для обычных пользователей общим правилом является то, что если они могут получить доступ к файлу из оболочки, то могут получить и доступ к тому же файлу с FTP-сервера. Таким образом, как правило, обычные пользователи должны иметь возможность загружать (`get`) и помещать (`put`) файлы в собственные домашние каталоги и из них. После установки прав и принятия иных мер безопасности для FTP-сервера можно переходить к другим параметрам конфигурации FTP-сервера.

Настройка FTP-сервера

Большая часть настройки службы `vsftpd` выполняется в файле `/etc/vsftpd/vsftpd.conf`. Варианты `vsftpd.conf` для различных типов сайтов включены в каталог `/usr/share/doc/vsftpd`. В зависимости от того, как вы хотите использовать FTP-сервер, в следующих разделах описано несколько способов его настройки. Не забудьте перезапустить службу `vsftpd` после внесения каких-либо изменений в конфигурацию.

Настройка доступа для пользователей

Сервер `vsftpd` могут применять все локальные пользователи Linux (перечисленные в файле `/etc/passwd`), а доступ анонимных пользователей запрещен. Это основывается на следующих настройках `vsftpd.conf`:

```
anonymous_enable=NO
local_enable=YES
```

Некоторые компании разрешают пользователям применять FTP для загрузки содержимого на свои собственные веб-серверы. В некоторых случаях пользователи имеют учетные записи только для FTP-сервера, что означает: они не могут войти в оболочку, но могут войти через сервер FTP и управлять своими данными. Создание учетной записи пользователя, у которой нет оболочки по умолчанию (на самом деле `/sbin/nologin`), — это способ не дать пользователю войти в оболочку, но при этом разрешить доступ по FTP. Например, строка `/etc/passwd` для учетной записи FTP-пользователя `bill` может выглядеть примерно так:

```
bill:x:1000:1000:Bill Jones:/home/bill:/sbin/nologin
```

С учетной записью пользователя, установленной с `/sbin/nologin` в качестве оболочки по умолчанию, любые попытки войти в систему с консоли или через `ssh` в качестве учетной записи пользователя `bill` будут отклонены. Однако до тех пор, пока у пользователя `bill` есть пароль и включен доступ локальной учетной записи к FTP-серверу, он сможет войти на сервер через FTP-клиент.

Не каждый пользователь с учетной записью в системе Linux имеет доступ к FTP-серверу. Параметр `userlist_enable=YES` в файле `vsftpd.conf` сообщает об отказе в доступе к FTP-серверу всем учетным записям, перечисленным в файле `/etc/vsftpd/user_list`. В этот список входят административные пользователи `root`, `bin`, `daemon`, `adm`, `lp` и др. Можете добавить в этот список других пользователей, которым хотели бы запретить доступ.

Если вы измените параметр `userlist_enable` на значение `NO`, то файл `user_list` станет списком только тех пользователей, которые имеют доступ к серверу. Другими словами, установка значения параметру `userlist_enable=NO`, удаление всех имен пользователей из файла `user_list` и добавление имен пользователей `chris`, `joe` и `mary1` в этот файл приводят к тому, что сервер разрешает только этим трем пользователям входить на него.

Независимо от того, как задано значение `userlist_enable`, файл `/etc/vsftpd/ftpusers` всегда включает в себя пользователей, которым отказано в доступе к серверу. Как и файл `userlist_enable`, файл `ftpusers` содержит список пользователей. Вы можете добавить в него пользователей, если хотите, чтобы им было отказано в доступе по FTP.

Один из способов ограничить доступ в систему пользователям с обычными учетными записями — это применить настройки команды `chroot`. Примеры некоторых параметров `chroot`:

```
chroot_local_user=YES
chroot_list_enable=YES
chroot_list_file=/etc/vsftpd/chroot_list
```

С помощью этих настроек вы можете создать список локальных пользователей и добавить их в файл `/etc/vsftpd/chroot_list`. После того как один из этих пользователей войдет в систему, ему будет запрещено переходить в те места в ней, которые находятся за пределами структуры его домашнего каталога.

Если загрузка на FTP-сервер разрешена, каталоги, в которые пользователь хочет загрузить данные, должны быть доступны для записи этим пользователем. Однако загруженные файлы могут храниться под именем пользователя, отличным от имени пользователя, загрузившего файл. Это одна из функций, которая будет описана далее, в пункте «Права на загрузку».

Права на загрузку

Чтобы разрешить любую форму записи на сервер `vsftpd`, необходимо установить параметр `write_enable=YES` в файле `vsftpd.conf` (по умолчанию установлен). Поэтому, если включены локальные учетные записи, пользователи могут войти

в систему и сразу же начать загрузку файлов в свои домашние каталоги. Однако анонимные пользователи по умолчанию лишены возможности загружать файлы.

Чтобы разрешить анонимную загрузку с помощью службы `vsftpd`, необходимо добавить первый параметр из примера, приведенного далее, а также вторую строку кода (оба варианта можно подключить, раскомментировав их из файла `vsftpd.conf`). Первый параметр позволяет анонимным пользователям загружать файлы, второй — создавать каталоги:

```
anon_upload_enable=YES
anon_mkdir_write_enable=YES
```

Следующим шагом является создание каталога, в котором анонимные пользователи могут записывать файлы. Любой каталог в каталоге `/var/ftp`, имеющий права на запись для пользователя `ftp`, группы `ftp` или `other`, может быть записан анонимным пользователем. Чаще всего создается каталог загрузок с правами на запись. Далее приведены примеры команд, выполняемых на сервере:

```
# mkdir /var/ftp/uploads
# chown ftp:ftp /var/ftp/uploads
# chmod 775 /var/ftp/uploads
```

Если брандмауэр открыт и логические типы SELinux установлены правильно, анонимный пользователь может перейти в каталог загрузок и поместить в него файл из локальной системы пользователя. На сервере файл будет находиться в собственности пользователя `ftp` и группы `ftp`. Права, установленные в каталоге (`775`), позволят вам видеть загруженные файлы, но не изменять и не перезаписывать их.

Одна из причин, почему нужно разрешить анонимным пользователям загружать файлы на сервер, заключается в том, чтобы позволить людям, которых вы не знаете, сбрасывать файлы в вашу папку загрузок. Поскольку любой, кто может найти сервер, способен записывать файлы в этот каталог, для сервера должна быть какая-то форма безопасности. Нужно запретить анонимному пользователю видеть файлы, загруженные другими пользователями, а также принимать или удалять файлы, загруженные другими анонимными пользователями FTP. Одной из форм безопасности является функция `chown` FTP.

Установив следующие два значения, вы можете разрешить анонимную загрузку. В результате, когда анонимный пользователь загружает файл, последний сразу же присваивается другому пользователю. Далее приведен пример настроек `chown`, которые вы можете поместить в файл `vsftpd.conf` для использования со своим каталогом анонимной загрузки:

```
chown_uploads=YES
chown_username=joe
```

Если анонимный пользователь загрузит файл после перезапуска службы `vsftpd` с этими настройками, этот файл будет принадлежать пользователю `Joe` и группе `ftp`. Для владельца установлены права на чтение/запись, а для остальных права не установлены (`rw-------`).

Ранее мы рассмотрели варианты конфигурации отдельных функций на сервере vsftpd. Некоторые наборы переменных vsftp.conf могут работать вместе с помощью подходящих для определенных типов FTP-сайтов способов. В следующем разделе описан один из этих примеров, представленный файлом конфигурации vsftpd.conf, который входит в пакет vsftpd. Этот файл можно скопировать из каталога с примерами файлов в файл /etc/vsftpd/vsftpd.conf и использовать на FTP-сервере, доступном в Интернете.

Настройка службы vsftpd для Интернета

Чтобы безопасно обмениваться файлами с FTP-сервера в Интернете, можно заблокировать сервер, ограничив его только разрешением на загрузку и только от анонимных пользователей. Чтобы настроить безопасный обмен файлами vsftpd через Интернет, создайте резервную копию текущего файла /etc/vsftpd/vsftpd.conf и скопируйте этот файл, чтобы перезаписать файл vsftpd.conf:

```
/usr/share/doc/vsftpd/EXAMPLE/INTERNET_SITE/vsftpd.conf
```

Далее описывается содержание этого файла. Настройки в первом разделе задают права доступа к серверу:

```
# Access rights
anonymous_enable=YES
local_enable=NO
write_enable=NO
anon_upload_enable=NO
anon_mkdir_write_enable=NO
anon_other_write_enable=NO
```

Включите параметр anonymous_enable (YES) и отключите параметр local_enable (NO), чтобы никто не мог войти на FTP-сервер с помощью обычной учетной записи пользователя Linux. Все должны входить через анонимный аккаунт. Никто не может загружать файлы (write_enable=NO). Затем установите, что анонимный пользователь не может загружать файлы (anon_upload_enable=NO), создавать каталоги (anon_mkdir_write_enable=NO) или как-то иначе записывать на сервер (anon_other_write_enable=NO). Вот так выглядят настройки безопасности:

```
# Security
anon_world_readable_only=YES
connect_from_port_20=YES
hide_ids=YES
pasv_min_port=50000
pasv_max_port=60000
```

Поскольку демон vsftpd может читать файлы, назначенные пользователю и группе ftp, параметр anon_world_readable_only=YES гарантирует, что анонимные пользователи могут видеть файлы, где включен только бит разрешения на чтение для other (------r--). Параметр connect_from_port_20=YES дает демону vsftpd

немного больше прав на отправку данных так, как может запросить клиент, разрешив передачу данных в стиле PORT.

Параметр hide_ids=YES скрывает реальные права, установленные для файлов, поэтому пользователю, получающему доступ к FTP-сайту, видно только то, что все файлы принадлежат пользователю ftp. Два параметра pasv ограничивают диапазон портов, которые могут применяться с пассивным FTP (где сервер выбирает порт с более высоким номером для отправки данных), значениями от 50 000 до 60 000.

В следующей части содержатся функции сервера vsftpd:

```
# Features
xferlog_enable=YES
ls_recurse_enable=NO
ascii_download_enable=NO
async_abor_enable=YES
```

С помощью параметра xferlog_enable=YES все передачи файлов на сервер и с сервера регистрируются в файле /var/log/xferlog. Параметр ls_recurse_enable=NO запрещает пользователям рекурсивно перечислять содержимое FTP-каталога (другими словами, предотвращает вывод списка, который можно получить с помощью команды ls -R), поскольку на большом сайте это может привести к большому потреблению ресурсов. Отключение загрузки ASCII заставляет все загрузки находиться в двоичном режиме (предотвращая перевод файлов в ASCII, что не подходит для двоичных файлов). Параметр async_abor_enable=YES гарантирует, что FTP-клиенты при прерывании передачи не будут зависать.

Следующие параметры влияют на производительность:

```
# Performance
one_process_model=YES
idle_session_timeout=120
data_connection_timeout=300
accept_timeout=60
connect_timeout=60
anon_max_rate=50000
```

С помощью параметра one_process_model=YES производительность может улучшиться, поскольку служба vsftpd запускает один процесс на каждое соединение. Уменьшение параметра idle_session_timeout с 300 секунд по умолчанию до 120 секунд приводит к отключению FTP-клиентов, которые простаивают более 2 минут. Таким образом, затрачивается меньше времени на управление сеансами FTP, которые уже не используются. Если передача данных останавливается более чем на несколько секунд data_connection_timeout (здесь 300 секунд), соединение с клиентом прерывается.

Параметр accept_timeout, равный 60 секундам, позволяет удаленному клиенту принять PASV-соединение за 1 минуту. Параметр connect_timeout задает время, в течение которого удаленный клиент должен ответить на запрос об установлении соединения для передачи данных в стиле PORT. Ограничение скорости передачи до 50 000 байт/с с помощью параметра anon_max_rate может улучшить общую производительность сервера, ограничив пропускную способность, которую может потреблять каждый клиент.

Подключение FTP-клиентов к серверу

Многие клиентские программы поставляются вместе с Linux. Их можно использовать для подключения к FTP-серверу. Если вы просто хотите сделать анонимную загрузку некоторых файлов с FTP-сервера, браузер Firefox — подходящий инструмент для этого с простым интерфейсом. Для более сложных взаимодействий между FTP-клиентом и сервером можно задействовать FTP-клиенты командной строки. В следующих разделах описаны некоторые из этих инструментов.

Доступ к FTP-серверу из браузера Firefox

Браузер Firefox обеспечивает быстрый и простой способ проверить доступ к вашему FTP-серверу или к любому общедоступному FTP-серверу. В своей системе введите `ftp://localhost` в адресной строке. Браузер предложит войти в систему, что можно сделать от имени обычного или анонимного пользователя, если сервер доступен через анонимный FTP. Как анонимный пользователь вы увидите нечто похожее как на рис. 18.2.

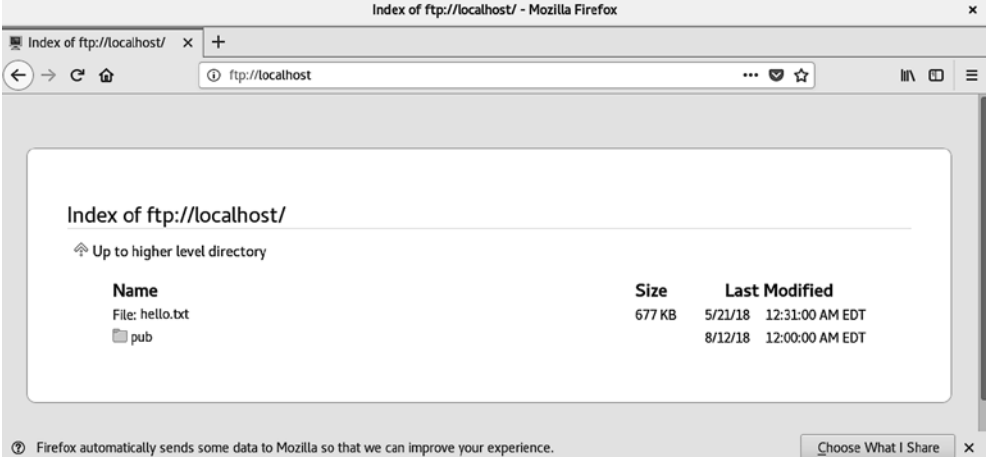


Рис. 18.2. Доступ к FTP-серверу из браузера Firefox

Чтобы войти на FTP-сервер как определенный пользователь из браузера Firefox, вставьте `имяпользователя:password@` перед `localhost`, как показано в следующем примере:

```
ftp://chris:MypassWd5@localhost
```

Указав правильное имя пользователя и пароль, вы сразу же увидите содержимое своего домашнего каталога. Щелкните на папке, чтобы открыть ее. Щелкните на файле, чтобы загрузить или просмотреть его.

Доступ к FTP-серверу с помощью команды lftp

Для проверки своего FTP-сервера из командной строки примените команду `lftp`. Чтобы установить команду `lftp` в системах Fedora или RHEL, введите в командной строке следующее:

```
# yum install lftp
```

Если вы применяете команду `lftp` только с именем FTP-сервера, к которому хотите получить доступ, команда пытается подключиться к FTP-серверу как анонимный пользователь. Добавив параметр `u` *имя_пользователя*, можете ввести пароль пользователя при появлении запроса и получить доступ к FTP-серверу от имени пользователя, под которым вошли в систему.

После ввода информации о пользователе и пароле вы получите приглашение `lftp`, готовое к вводу команд. При вводе первой команды устанавливается соединение с сервером. Вы можете задействовать команды для перемещения по FTP-серверу, а затем команды `get` и `put` для загрузки и выгрузки файлов.

В следующем примере показано, как применять команды подобным образом. Предполагается, что FTP-сервер (и связанные с ним меры безопасности) был настроен таким образом, чтобы позволить локальным пользователям подключаться, а также читать и записывать файлы:

```
# lftp -u chris localhost
Password:
********
lftp chris@localhost:~> pwd
ftp://chris@localhost/%2Fhome/chris
lftp chris@localhost:~> cd stuff/state/
lftp chris@localhost:~/stuff/state> ls
-rw-r--r--    1 13597     13597         1394 Oct 23 2014
enrolled-20141012
-rw-r--r--    1 13597     13597         514 Oct 23 2014
enrolled-20141013
lftp chris@localhost:~/stuff/state> !pwd
/root
lftp chris@localhost:~/stuff/state> get survey-20141023.txt
3108 bytes transferred
lftp chris@localhost:~/stuff/state> put /etc/hosts
201 bytes transferred
lftp chris@localhost:~/stuff/state> ls
-rw-r--r--    1 13597     13597         1394 Oct 23 2014
enrolled-20141012
-rw-r--r--    1 13597     13597         514 Oct 23 2014
enrolled-20141013
-rw-r--r--    1 0         0             201 May 03 20:22 hosts
lftp chris@localhost:~/stuff/state> !ls
anaconda-ks.cfg          bin             install.log
dog                      Pictures        sent
Downloads                Public          survey-20141023.txt
lftp chris@localhost:~/stuff/state> quit
```

После ввода имени пользователя (`-u chris`) команда `lftp` запрашивает пароль пользователя `chris`. Команда `pwd` показывает, что пользователь `chris` вошел в систему на локальном хосте и что `/home/chris` — это текущий каталог. Так же как и в обычной оболочке командной строки Linux, вы можете применить команду `cd` для перехода в другой каталог и `ls` для перечисления его содержимого.

Чтобы команда, которую вы запускаете, интерпретировалась клиентом, просто поставьте перед ней восклицательный знак (`!`). Например, команда `!pwd` показывает, что текущий каталог в системе, инициировавшей `lftp`, — это `/root`. Эта информация полезна, потому что, если вы получаете файл с сервера без указания его назначения, он переходит в текущий каталог клиента (в данном случае `/root`). Другие команды, которые будут интерпретироваться клиентской системой, — это `!cd` (для изменения каталогов) и `!ls` (для перечисления файлов).

Если у вас есть права на чтение файла на сервере и права на запись, используйте команду `get` для загрузки файла с сервера (`get survey-20141023.txt`). Если у вас есть права на запись и загрузку в текущий каталог на сервере, применяйте команду `put` для копирования файла на сервер (`put /etc/hosts`).

Команда `ls` показывает, что файл `/etc/hosts` был загружен на сервер. Команда `!ls` говорит о том, что файл `survey-20141023.txt` был загружен с сервера в инициирующую систему.

Использование gFTP-клиента

В системах Linux доступно и множество других FTP-клиентов. Еще один FTP-клиент, который стоит попробовать, — это gFTP. Клиент gFTP предоставляет интерфейс, который позволяет видеть как локальную, так и удаленную стороны FTP-сеанса. Чтобы установить gFTP в системе Fedora, выполните следующую команду:

```
# yum install gftp
```

Чтобы открыть gFTP, запустите его из меню приложений или выполните команду `gftp &` из оболочки. Для использования этого клиента введите URL-адрес FTP-сервера, к которому хотите подключиться, введите имя пользователя (например, `anonymous`) и нажмите клавишу Enter. На рис. 18.3 показан пример применения gFTP для подключения к сайту gnome.org — ftp.gnome.org.

Чтобы перейти на FTP-сайт из gFTP, просто дважды щелкните на папке (как в окне файлового менеджера). Полные пути к локальному каталогу (слева) и удаленному каталогу (справа) показаны над списками файлов и папок, расположенных ниже.

Чтобы перенести файл с удаленной стороны на локальную, выберите нужный файл справа и нажмите на стрелку влево в середине экрана. Следите за ходом передачи файлов с помощью сообщений в нижней части экрана. Когда передача завершится, файл появится на панели слева.

Рис. 18.3. FTP-клиент gFTP позволяет видеть обе стороны FTP-сеанса

Можете добавить в закладки адреса, необходимые для подключения к FTP-сайту. Адрес добавляется к набору закладок, уже хранящихся в меню закладок. Выберите сайты из списка, чтобы опробовать gFTP. Большинство сайтов предназначены для дистрибутивов Linux и других ресурсов с открытым исходным кодом.

Резюме

FTP-сервер — это простой способ обмена файлами по сети TCP. Довольно безопасный демон FTP (пакет `vsftpd`) доступен для Fedora, Red Hat Enterprise Linux, Ubuntu и других систем Linux.

Сервер `vsftpd` по умолчанию позволяет анонимным пользователям загружать файлы с сервера, а обычным пользователям Linux — загружать или скачивать файлы (при условии правильных настроек безопасности). Перемещение по FTP-серверу аналогично перемещению по файловой системе Linux, то есть нужно переходить вверх и вниз по структуре каталогов, чтобы найти нужный контент.

Существуют как графические, так и текстовые FTP-клиенты. Популярным текстовым клиентом для Linux является `lftp`. Что касается графических FTP-клиентов, то можно задействовать обычный браузер, например Firefox, или специальные FTP-клиенты, такие как gFTP.

FTP-серверы — это не единственный способ обмена файлами по сети из систем Linux. Служба Samba предоставляет способ совместного использования файлов по сети, чтобы общий каталог Linux выглядел как общий каталог из системы Windows. Глава 19 «Настройка Samba-сервера» описывает, как применять Samba для предоставления общего доступа к файлам в стиле Windows.

Упражнения

Упражнения в этом разделе связаны с настройкой FTP-сервера в RHEL или Fedora и подключением к этому серверу с помощью FTP-клиента. Если затрудняетесь с решением, воспользуйтесь ответами к упражнениям, размещенными в приложении Б (хотя Linux позволяет решать задачи разными способами).

Не выполняйте эти упражнения в системе Linux, работающей на общедоступном FTP-сервере, потому что они почти наверняка помешают его работе.

1. Определите, какой пакет предоставляет демон Very Secure FTP.
2. Установите пакет Very Secure FTP в своей системе и найдите в нем файлы конфигурации.
3. Включите анонимный доступ к FTP-серверу и отключите вход локальным учетным записям в службе Very Secure FTP.
4. Запустите службу Very Secure FTP Daemon и настройте ее для запуска при загрузке системы.
5. В системе, на которой работает ваш FTP-сервер, создайте в анонимном каталоге FTP файл с именем `test`, который содержит фразу `Welcome to your vsftpd server`.
6. В браузере системы, на которой работает ваш FTP-сервер, откройте файл `test` из анонимного домашнего каталога FTP. Убедитесь, что вы видите содержимое этого файла.
7. Из браузера за пределами системы, на которой работает FTP-сервер, получите доступ к файлу `test` в анонимном домашнем каталоге FTP. Если не можете получить доступ к файлу, убедитесь, что ваш брандмауэр, SELinux и TCP-оболочки настроены так, чтобы доступ был разрешен.
8. Настройте свой сервер `vsftpd` так, чтобы разрешить анонимным пользователям загрузку файлов в каталог `in`.
9. Установите FTP-клиент `lftp` (если нет второй системы Linux, сделайте это на том же хосте, на котором работает FTP-сервер). Если не можете загрузить файлы в каталог `in`, убедитесь, что ваш брандмауэр, SELinux и TCP-оболочки настроены так, чтобы доступ был разрешен.
10. Используя любой FTP-клиент, посетите каталог `/pub/debian-meetings` на сайте ftp.gnome.org и перечислите его содержимое.

19 Настройка Samba-сервера

В этой главе

- Загрузка и установка Samba-сервера.
- Функции безопасности Samba-сервера.
- Редактирование файла конфигурации `smb.conf`.
- Доступ к Samba-серверу для клиентов Linux и Windows.
- Samba-сервер на предприятии.

Samba-сервер — это проект, реализующий открытые исходные версии протоколов, применяемых для обмена файлами между системами Windows и принтерами, а также для аутентификации пользователей и ограничения хостов. Samba предлагает разные способы обмена файлами между системами Windows, Linux и MacOS, известные и доступные их пользователям.

В этой главе вы ознакомитесь с процессом установки и настройки сервера Samba. Здесь описываются функции безопасности, необходимые для совместного использования ресурсов файлов и принтеров, а также способы доступа к этим ресурсам из систем Linux и Windows.

Что такое Samba

Samba (samba.org) — это набор программ, который позволяет Linux, UNIX и другим системам взаимодействовать с протоколами общего доступа к файлам и принтерам Microsoft Windows. Windows, macOS и другие клиентские системы могут получить доступ к серверам Samba для совместного использования файлов и принтеров таким же образом, как и с файловых серверов Windows и серверов печати.

С помощью Samba вы можете задействовать стандартную сеть TCP/IP для связи с клиентами. Для службы имен Samba-сервер поддерживает обычные име-

на хостов TCP/IP, а также имена NetBIOS. По этой причине Samba не требует протокола NetBEUI (Microsoft Raw NetBIOS frame). Общий доступ к файлам осуществляется с помощью протокола Server Message Block (SMB), который иногда называют *Common Internet File System* (единая файловая система Интернета, CIFS).

При разработке проекта Samba было предпринято множество усилий, чтобы сделать программное обеспечение безопасным и надежным. На самом деле многие пользователи предпочитают серверы Samba вместо файловых серверов Windows именно из-за дополнительной безопасности, присущей службам обмена файлами в стиле Windows в Linux или в других UNIX-подобных операционных системах.

Помимо всей технической всячины, конечным результатом является то, что Samba позволяет легко обмениваться файлами между серверами Linux и настольными системами Windows. Для настройки сервера требуется всего несколько файлов конфигурации и инструментов для управления Samba. Для клиентов общие ресурсы просто отображаются в разделе Network (Сеть) (меню Пуск), а в старых системах Windows — в Проводнике Windows или окне Network Neighborhood (Сетевое окружение).

Для настройки службы Samba необходимо непосредственно отредактировать файлы конфигурации Samba (в частности, `smb.conf`) и запустить несколько команд. Графические и веб-интерфейсы, такие как `system-config-samba` и Samba SWAT, больше не входят в состав новейших систем Fedora и RHEL.

Чтобы начать использовать сервер Samba в системе Linux, необходимо установить несколько пакетов программного обеспечения, как описывается в следующем разделе.

Установка Samba-сервера

В Red Hat Enterprise Linux и Fedora для настройки файла Samba и сервера печати необходимо установить пакет `samba`. В числе других компонентов пакет `samba` включает демон службы Samba (`/usr/sbin/smbd`) и демон сервера имен NetBIOS (`/usr/sbin/nmbd`). Пакет `samba` включает в себя пакет `samba-common`, который содержит файлы конфигурации сервера (`smb.conf`, `lmhost` и др.) и команды добавления паролей и тестирования файлов конфигурации, а также другие функции Samba.

Функции из других пакетов также упоминаются в этой главе, поэтому я опишу, как установить и эти пакеты.

- Пакет `samba-client`. Содержит инструменты командной строки, такие как `smbclient` (для подключения к общим ресурсам Samba или Windows), `nmblookup` (для поиска адресов хостов) и `findsmb` (для поиска хостов SMB в сети).
- Пакет `samba-winbind`. Включает компоненты, которые позволяют серверу Samba в Linux стать полноправным членом домена Windows, в том числе применение учетных записей пользователей и групп Windows в Linux.

Чтобы установить все только что упомянутые пакеты (`samba-common` устанавливается как зависимый пакет общего пакета `samba`, так что не нужно делать это отдельно), как суперпользователь введите из командной строки в дистрибутиве Fedora или RHEL следующее:

```
# yum install samba samba-client samba-winbind
...
Last metadata expiration check: 0:01:44 ago on Sun 24 Jan 2020
11:35:37 AM EST.
Dependencies resolved.
================================================================================
Package       Architecture      Version  Repository
Size
================================================================================
Installing:
 samba         x86_64   4.10.4-101.el8_1  rhel-8-for-x86_64-baseosrpms-739 k
 samba-winbind x86_64   4.10.4-101.el8_1  rhel-8-for-x86_64-baseosrpms-570 k
Installing dependencies:
 samba-common-tools
               x86_64   4.10.4-101.el8_1  rhel-8-for-x86_64-baseosrpms-469 k
 samba-libs    x86_64   4.10.4-101.el8_1  rhel-8-for-x86_64-baseosrpms-185 k
 samba-winbind-modules
               x86_64   4.10.4-101.el8_1  rhel-8-for-x86_64-baseosrpms-122 k
 samba-client  x86_64   4.10.4-101.el8_1  rhel-8-for-x86_64-baseosrpms-658 k

Transaction Summary
================================================================================
Install  6 Packages

Total download size: 2.5 M
Installed size: 6.8 M
Is this ok [y/d/N]: y
```

Чтобы просмотреть файлы конфигурации после того, как вы установили пакеты Samba, введите:

```
# rpm -qc samba-common
/etc/logrotate.d/samba
/etc/samba/lmhosts
/etc/samba/smb.conf
/etc/sysconfig/samba
```

Файлы `/etc/logrotate.d/samba` и `/etc/sysconfig/samba` обычно не изменяются. Первый устанавливает, как файлы в каталоге `/var/log/samba` обрабатываются (копируются в другие файлы и удаляются) с течением времени. Второй — это файл, в который можно поместить параметры, передаваемые демону `smbd`, `nmbd` или `winbindd`, чтобы отключить различные функции, например отладку.

Большинство файлов конфигурации, которые можно изменить для сервера Samba, находятся в каталоге `/etc/samba`. Файл `smb.conf` — это основной файл конфигурации, в котором содержатся основные настройки сервера, а также ин-

формация об общем доступе к отдельным файлам и принтерам (подробнее об этом позже). Файл `lmhosts` позволяет сопоставлять имя хоста Samba NetBIOS с IP-адресами.

Хотя по умолчанию файла `/etc/samba/smbusers` не существует, вы можете создать его, чтобы сопоставлять имена пользователей Linux и Windows. Информация о том, как настроить Samba-сервер, находится на справочной странице `smb.conf` (`man smb.conf`). Существуют также справочные страницы для команд Samba, таких как `smbpasswd` (смена паролей), `smbclient` (подключение к серверу Samba) и `nmblookup` (поиск информации NetBIOS).

После установки пакетов Samba и быстрого обзора того, что в них содержится, запустите службу Samba и посмотрите, какие в ней будут настройки.

Запуск и остановка службы Samba

Установив пакеты `samba` и `samba-common`, вы можете запустить сервер и исследовать, как он работает с настройками по умолчанию. С сервером Samba связаны две основные службы, каждая из которых имеет собственный демон службы.

- `smb`. Эта служба управляет процессом демона `smbd`, который предоставляет службы общего доступа к файлам и принтерам, доступные клиентам Windows.
- `nmb`. Эта служба управляет демоном `nmbd`. Обеспечивая сопоставление имени службы имен NetBIOS с адресом, `nmbd` может сопоставлять запросы от клиентов Windows с именами NetBIOS, разрешенными в IP-адресах.

Для обмена файлами с другими системами Linux и принтерами с помощью Samba требуется только служба `smb`. В следующем разделе описывается, как запустить и включить ее.

Запуск службы Samba (smb)

Служба `smb` запускает сервер `smbd` и открывает доступ к файлам и принтерам вашей системы для других компьютеров общей сети. Как обычно, в разных системах Linux службы включаются и запускаются по-разному. Для запуска демона `smbd` в различных системах Linux вам нужно найти имя службы и правильный инструмент.

Чтобы в системах Fedora и RHEL подключить сервер Samba и немедленно запускать его при их загрузке, от имени суперпользователя введите из командной строки следующее:

```
# systemctl enable smb.service
# systemctl start smb.service
# systemctl status smb.service
smb.service – Samba SMB Daemon
    Loaded: loaded (/usr/lib/systemd/system/smb.service; enabled)
```

```
    Active: active (running) since Fri 2020-01-31 07:23:37 EDT; 6s ago
       Docs: man:smbd(8)
             man:samba(7)
             man:smb.conf(5)
    Status:  "smbd: ready to serve connections..."
     Tasks: 4 (limit: 12216)
    Memory: 20.7M
  Main PID: 4838 (smbd)
CGroup: /system.slice/smb.service
 ├ 4838 /usr/sbin/smbd --foreground --no-process-group
 └ 4840 /usr/sbin/smbd --foreground --no-process-group
```

Первая команда `systemctl` включает службу, вторая сразу же запускает ее, а третья отображает состояние. Обратите внимание на то, что файл службы располагается по адресу `/usr/lib/systemd/system/smb.service`. Посмотрите на содержимое этого файла:

```
# cat /usr/lib/systemd/system/smb.service
[Unit]
Description=Samba SMB Daemon
Documentation=man:smbd(8) man:samba(7) man:smb.conf(5)
Wants=network-online.target
After=network.target network-online.target nmb.service winbind.
service
[Service]
Type=notify
NotifyAccess=all
PIDFile=/run/smbd.pid
LimitNOFILE=16384
EnvironmentFile=-/etc/sysconfig/samba
ExecStart=/usr/sbin/smbd --foreground --no-process-group $SMBDOPTIONS
ExecReload=/bin/kill -HUP $MAINPID
LimitCORE=infinity
Environment=KRB5CCNAME=FILE:/run/samba/krb5cc_samba
[Install]
WantedBy=multi-user.target
```

Процесс демона Samba (`smbd`) запускается после целевых `network`, `network-online`, `nmb` и `winbind`. Файл `/etc/sysconfig/samba` содержит переменные, которые передаются в качестве аргументов демонам `smbd`, `nmbd` и `winbindd` при их запуске. Ни для одного из этих демонов по умолчанию не задано никаких параметров. Строка `WantedBy` указывает, что служба `smb.service` должна запускаться при загрузке системы в многопользовательский режим (многопользовательская цель), что происходит по умолчанию.

В RHEL 6 и более ранних версиях системы сервер Samba можно запустить следующим образом:

```
# service smb start
Starting SMB services:     [ OK ]
# chkconfig smb on
```

```
# service smb status
smbd (pid 28056) is running...
# chkconfig --list smb
smb             0:off  1:off  2:on  3:on  4:on  5:on  6:off
```

Независимо от того, используете ли вы сервер Samba в RHEL, Fedora или другой системе Linux, проверить доступ к нему можно с помощью команды `smbclient` (из клиентского пакета `samba`). Основную информацию с сервера Samba можно получить с помощью следующей команды:

```
# smbclient -L localhost
Enter SAMBA\root's password: <ENTER>
Anonymous login successful

    Sharename  Type    Comment
    ---------  ----    -------
    print$     Disk    Printer Drivers
    IPC$       IPC     IPC Service
  (Samba Server Version 4.10.10)
    deskjet    Printer deskjet
Reconnecting with SMB1 for workgroup listing.
Anonymous login successful
    Server               Comment
    ---------            -------
    Workgroup            Master
    ---------            -------
```

Вывод команды `smbclient` позволяет увидеть, какие службы доступны с сервера. По умолчанию анонимный вход разрешен при запросе сервера (поэтому я просто нажал клавишу Enter, когда было предложено ввести пароль). Из этого вывода по умолчанию вы можете узнать о настройке сервера Samba следующее.

- Все принтеры, совместно используемые через сервер CUPS в вашей системе Linux, доступны также с сервера Samba, работающего в этой системе.
- Никакие каталоги сервера еще не являются общими.
- Служба имен NetBIOS еще не запущена с сервера Samba.

После этого вы можете решить, хотите ли запустить службу имен NetBIOS на своем сервере Samba.

Запуск сервера имен NetBIOS (nmbd)

Если в сети не работает сервер домена Windows, как в данном случае, можете запустить службу `nmb` на хосте Samba, чтобы подключить ее. Чтобы запустить службу `nmb` (демон `nmbd`) в Fedora или RHEL 7, введите следующее:

```
# systemctl enable nmb.service
# systemctl start nmb.service
# systemctl status nmb.service
```

В RHEL 6 и более ранних версиях системы для запуска службы nmb необходимо ввести следующее:

```
# service nmb start
# service nmb status
# chkconfig nmb on
# chkconfig --list nmb
```

Независимо от того, как была запущена служба NetBIOS, демон nmbd теперь запустится и будет сопоставлять имена NetBIOS с адресами. Снова выполните команду smbclient L, а затем введите IP-адрес сервера.

На этот раз последние несколько строк вывода должны отображать информацию, полученную от сервера NetBIOS, который теперь работает на сервере Samba. В этом случае последние несколько строк выглядят так:

```
# smbclient -L localhost
    ...
    Workgroup   Master
    ---------   -------
    SAMBA       FEDORA30
```

Видно, что новый сервер NetBIOS называется FEDORA30 и является главным сервером для рабочей группы. Чтобы сделать запрос к серверу nmbd для IP-адреса FEDORA30, необходимо ввести следующую команду:

```
# nmblookup -U localhost FEDORA30
querying FEDORA30 on 127.0.0.1
192.168.122.81 FEDORA30<00>
```

Теперь видно, что сервер Samba работает из локальной системы. Имя хоста (в данном случае FEDORA30) назначается системе по умолчанию.

Но если у вас настроен брандмауэр или включена система SELinux, возможно, вы не сможете полностью получить доступ к серверу Samba из удаленной системы. В следующем разделе будет описано, как открыть сервер Samba для систем за пределами локальной системы, а также разрешить некоторые функции сервера, которые могли быть отключены в SELinux.

Остановка служб Samba (smb) и NetBIOS (nmb)

Чтобы остановить службы smb и nmb в Fedora или RHEL, вы можете использовать ту же команду systemctl, что и для их запуска. Ее же можно применить и для отключения служб, чтобы они не запускались снова при загрузке системы. Вот пример того, как немедленно остановить службы smb и nmb:

```
# systemctl stop smb.service
# systemctl stop nmb.service
```

В RHEL 6 и более ранних версиях систем Linux, чтобы остановить службы smb и nmb, введите следующие команды:

```
# service smb stop
# service nmb stop
```

Чтобы в системах Fedora или RHEL предотвратить запуск служб `smb` и `nmb` при следующей перезагрузке системы, введите следующие команды:

```
# systemctl disable smb.service
# systemctl disable nmb.service
```

В системе Red Hat Enterprise Linux 6 и более ранних версиях введите следующие команды, чтобы отключить службы `smb` и `nmb`:

```
# chkconfig smb off
# chkconfig nmb off
```

Конечно, останавливать и отключать службы `smb` и `nmb` необходимо только в том случае, если вы больше не хотите работать со службой Samba. Продолжайте конфигурировать службу, настроив функции безопасности Linux, чтобы открыть доступ к службе Samba для других пользователей вашей сети.

Защита сервера Samba

Если вы не можете получить доступ к своему серверу Samba сразу после его запуска, вероятно, придется изменить настройки безопасности. Многие установки Linux по умолчанию предотвращают, а не разрешают доступ к системе, а настройки службы Samba направлены на обеспечение скорее ее доступности, а не безопасности.

Перечислю функции безопасности, о которых вы должны знать при настройке системы Samba.

- **Брандмауэры.** Брандмауэр по умолчанию для Fedora, RHEL и других систем Linux предотвращает любой доступ к локальным службам из внешних систем. Таким образом, чтобы разрешить пользователям доступ с других компьютеров к вашей службе Samba, вы должны создать правила брандмауэра, которые открывают один или несколько портов для выбранных протоколов (в частности, TCP).
- **Система SELinux.** Многие функции Samba система SELinux воспринимает как потенциально небезопасные. Поскольку логические типы SELinux по умолчанию (включение/выключение определенных функций) настроены так, чтобы обеспечить наименьший необходимый доступ, вам необходимо включить логические типы для доступа пользователей к своим домашним каталогам с помощью Samba. Другими словами, вы можете настроить Samba для совместного применения домашних каталогов пользователей, но система SELinux запретит эту функцию, если ее не перенастроить.
- **Ограничения для хоста и пользователя.** В самих файлах конфигурации Samba можно указать, какие хосты и пользователи могут иметь доступ к серверу Samba в целом или к отдельным общим каталогам.

В следующих разделах описывается, как настроить вышеупомянутые функции безопасности для сервера Samba.

Настройка брандмауэра для сервера Samba

Если брандмауэр `iptables` или `firewalld` настроен для системы при ее установке, он обычно разрешает любые запросы на услуги от локальных, но не от внешних пользователей. Вот почему при установке (как сказано в посвященном ей разделе в этой главе) необходимо проверить, подключается ли служба Samba с помощью команды `smbclient` из локальной системы. Но если бы запрос исходил из другой системы, он был бы отклонен.

Настройка правил брандмауэра для Samba в основном состоит из открытия входящих портов, на которых прослушиваются демоны `smbd` и `nmbd`. Перечислю порты, которые необходимо открыть, чтобы получить работающий сервис Samba в вашей системе Linux.

- **Порт TCP 445.** Это основной порт, на котором слушает демон Samba `smbd`. Брандмауэр должен поддерживать входящие на этот порт пакетные запросы, чтобы служба Samba работала.
- **Порт TCP 139.** Демон `smbd` также прослушивает порт TCP 139 для того, чтобы обрабатывать данные, связанные с именами хостов NetBIOS. Службу Samba можно использовать через порт TCP, не открывая порт, но это не рекомендуется.
- **Порты UDP 137 и 138.** Демон `nmbd` использует эти два порта для входящих запросов NetBIOS. Если вы применяете демон `nmbd`, они должны быть открыты для новых пакетных запросов для разрешения имен NetBIOS.

Для систем Fedora и RHEL разрешить входящий доступ к этим четырем портам очень просто. Откройте окно Firewall Configuration (Брандмауэр) и на вкладке Services (Службы) установите флажки samba и sambaclient. Эти порты сразу становятся доступными (без перезапуска службы `firewalld`).

Для более ранних систем Fedora и RHEL, которые используют службу `iptables` вместо службы `firewalld`, открытие брандмауэра — это ручной процесс. Рассмотрим брандмауэр по умолчанию дистрибутива Fedora, который разрешает входящие пакеты от локального хоста и от установленных соединений и связанных с установленными соединениями пакетов, но запрещает все остальные входящие пакеты. В следующем примере представлен набор правил брандмауэра в файле `/etc/sysconfig/iptables` с четырьмя новыми правилами (выделены):

```
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -i lo -j ACCEPT
-I INPUT -m state --state NEW -m udp -p udp --dport 137 -j ACCEPT
-I INPUT -m state --state NEW -m udp -p udp --dport 138 -j ACCEPT
-I INPUT -m state --state NEW -m tcp -p tcp --dport 139 -j ACCEPT
-I INPUT -m state --state NEW -m tcp -p tcp --dport 445 -j ACCEPT
-A INPUT -j REJECT --reject-with icmp-host-prohibited
-A FORWARD -j REJECT --reject-with icmp-host-prohibited
COMMIT
```

Ваш брандмауэр может включать дополнительные правила, разрешающие входящие пакетные запросы для других служб, таких как Secure Shell (`sshd`) или web (`httpd`). Их можно не изменять. Главное, чтобы ваши правила Samba располагались где-то перед окончательными правилами `REJECT`.

Если брандмауэр `iptables` включен, можете перезапустить его, чтобы новые правила вступили в силу. Для этого введите команду `systemctl restart iptables.service` (в старых системах Fedora) или `service restart iptables` (в RHEL 6 или более ранней версии). Снова подключитесь к серверу Samba с помощью команды `smbclient` или с помощью других методов, описанных в разделе «Доступ к общим ресурсам Samba» далее в этой главе.

Дополнительные сведения об использовании брандмауэра `iptables` см. в главе 25 «Защита Linux в сети».

Настройка системы SELinux для сервера Samba

Существуют как контекст файла, так и логические типы, связанные с использованием Samba с SELinux в принудительном режиме. Контексты файлов должны быть правильно установлены в каталоге, общем для службы Samba. Логические типы позволяют переопределить подход с точки зрения безопасности по умолчанию к определенным функциям Samba.

Информацию о том, как SELinux ограничивает Samba, вы можете найти на справочной странице `samba_selinux` (`man samba_selinux`). Чтобы получить ее, необходимо установить пакет `selinux-policy-doc`. Лучше понять SELinux поможет глава 24.

Настройка логических типов в SELinux для Samba-сервера

Простой способ перечислить и изменить логические типы SELinux для Samba — использовать командную строку. Чтобы задействовать команду `emanage` для перечисления логических типов, связанных с Samba, введите следующее:

```
# semanage boolean -l | egrep "smb|samba"
```

Далее приведен список логических типов SELinux, применимых к Samba, с описанием. Большинство логических типов позволяет установить, какие файлы и каталоги сервер может читать и записывать от имени пользователей Samba. Другие типы дают возможность разрешить потенциально небезопасные функции.

- `samba_run_unconfined`. Позволяет серверу запускать неограниченные скрипты из общих ресурсов Samba.
- `smbd_anon_write`. Дает возможность разрешить анонимным пользователям изменять общедоступные файлы, применяемые для общедоступных служб передачи файлов. Файлы и каталоги должны быть помечены как `public_content_rw_t`.
- `samba_enable_home_dirs`. Позволяет серверу совместно задействовать домашние каталоги пользователей.

- `samba_export_all_ro`. Позволяет Samba совместно применять любые файлы и каталоги только для чтения.
- `use_samba_home_dirs`. Позволяет удаленному серверу Samba получить доступ к домашним каталогам на локальном компьютере.
- `samba_create_home_dirs`. Позволяет серверу создавать новые домашние каталоги (например, с помощью PAM).
- `samba_export_all_rw`. Позволяет серверу совместно использовать любые файлы или каталоги для чтения/записи.

Следующие логические типы влияют на способность сервера Samba совместно применять каталоги, которые самостоятельно монтируются из других удаленных служб (например, NFS), или выступать в качестве контроллера домена Windows.

- `samba_share_fusefs`. Позволяет серверу экспортировать тома `ntfs/fusefs`.
- `samba_share_nfs`. Позволяет серверу экспортировать тома NFS.
- `samba_domain_controller`. Позволяет Samba выступать в качестве контроллера домена, добавлять пользователей и группы, а также изменять пароли.

Команда `setsebool` используется для включения или выключения логических типов SELinux. С помощью параметра `-P` она устанавливает логический тип, применяемый на постоянной основе.

Например, чтобы разрешить Samba совместно использовать любой файл или каталог с правами только на чтение с сервера, в качестве суперпользователя введите следующие команды:

```
# setsebool -P samba_export_all_ro on
# getsebool samba_export_all_ro
samba_export_all_ro --> on
```

В этом случае команда `setsebool` устанавливает логический тип в режим `on`. Команда `getsebool` позволяет увидеть значение логического типа.

Установка контекстов для файлов Samba в SELinux

SELinux устанавливает ограничения на файлы, к которым может получить доступ служба Samba. Вместо того чтобы разрешить серверу Samba общий доступ к любому файлу с соответствующими правами на чтение и запись, SELinux (когда система находится в принудительном режиме) требует, чтобы файлы и каталоги имели правильные контексты файлов, прежде чем служба Samba сможет даже увидеть, что файлы существуют.

Чтобы служба Samba могла сразу же работать с SELinux, некоторые файлы и каталоги предустановлены с соответствующими контекстами файлов. Например, файлы конфигурации Samba (`/etc/ samba/*`), файлы журналов (`/var/log/ samba/*`) и библиотеки (`/var/lib/ samba/*`) имеют правила, обеспечивающие правильные контексты файлов. Чтобы найти файлы и каталоги, связанные со службой

Samba и демоном smbd, которые имеют предустановленные контексты файлов, выполните следующие команды:

```
# semanage fcontext -l | grep -i samba
# semanage fcontext -l | grep -i smb
```

Часть нужных контекстов заканчивается на _t, например, samba_etc_t, samba_log_t и samba_var_t для каталогов /etc/samba, /var/log/samba и /var/lib/samba соответственно.

Вы можете обнаружить, что следует изменить контекст файла, например, при перемещении файлов в нестандартные места (допустим, перемещая файл smb.conf в /root/smb.conf) или когда нужно поделиться каталогом (за исключением домашних каталогов, которые можно включить, установив логические типы). В отличие от серверов vsftpd (FTP) и httpd (web), которые поставляются вместе с Linux, Samba не имеет общих каталогов содержимого по умолчанию (упоминавшиеся ранее используют каталоги /var/ftp и /var/www/html).

Вы можете навсегда изменить контекст файла, создав новое правило контекста, а затем применив его к нужному файлу или каталогу. Сделать это можно с помощью команд semanage, чтобы создать правило, и restorecon, чтобы его применить. Например, если вы хотите поделиться каталогом /mystuff, то должны создать его с соответствующими правами и выполнить приведенную далее команду, чтобы сделать его доступным для чтения/записи с сервера:

```
# semanage fcontext -a -t samba_share_t "/mystuff(/.*)?"
# restorecon -v /mystuff
```

После выполнения этих команд каталог /mystuff вместе с любыми расположенными ниже файлами и каталогами будет иметь контекст файла samba_share_t. Затем необходимо назначить правильного владельца Linux и правильные права доступа к файлам, чтобы разрешить доступ к выбранным вами пользователям. В разделе «Настройка сервера Samba» приведен пример создания общего ресурса, а также показано, как добавить разрешения и права собственности в общий каталог с помощью стандартных команд Linux.

Настройка прав хоста/пользователя сервера Samba

В самом файле smb.conf вы можете разрешить или ограничить доступ ко всему серверу Samba или к определенным общим ресурсам в зависимости от хостов или пользователей, пытающихся получить доступ. Можете также ограничить доступ к серверу Samba, предоставив службу только определенным интерфейсам.

Например, если одна карта сетевого интерфейса подключена к Интернету, а другая — к локальной сети, вы можете указать серверу Samba, чтобы он обслуживал запросы только на локальном сетевом интерфейсе. В следующем разделе описано, как настроить Samba, в том числе как определить, какие хосты, пользователи или сетевые интерфейсы могут получить доступ к серверу.

Настройка сервера Samba

В файле `/etc/samba/smb.conf` находятся настройки для установки сервера Samba, определения общих принтеров, задания способа аутентификации и создания общих каталогов. Файл состоит из следующих предопределенных разделов:

- `[global]`. Здесь размещены настройки, применимые к серверу Samba в целом. Здесь вы устанавливаете описание сервера, его рабочую группу (домен), расположение файлов журнала, тип безопасности по умолчанию и другие параметры;
- `[homes]`. Этот раздел определяет, могут ли пользователи с учетными записями на сервере Samba видеть свои домашние каталоги (доступные для просмотра) или писать в них;
- `[printers]`. Настройки этого раздела сообщают Samba, следует ли открывать доступ к принтерам печати Linux (CUPS) через сервер;
- `[print$]`. Здесь каталог настраивается как общая папка с драйверами принтера.

В файле `smb.conf` строки, начинающиеся со знаков фунта (`#`) или точки с запятой (`;`), являются комментариями. Удаление точек с запятой позволяет быстро настроить различные виды общей информации. Знак `#` можно применять и для комментирования строки.

Перед редактированием файла `smb.conf` сделайте резервную копию, которой сможете воспользоваться, если что-то пойдет не так. Для начала скопируйте файл `smb.conf.example` в `smb.conf`.

Настройка раздела [global]

Пример раздела `[global]` файла `smb.conf`:

```
[global]
        workgroup = SAMBA
        security = user
        passdb backend = tdbsam
        printing = cups
        printcap name = cups
        load printers = yes
        cups options = raw

;       netbios name = MYSERVER
;       interfaces = lo eth0 192.168.12.2/24 192.168.13.2/24
;       hosts allow = 127. 192.168.12. 192.168.13.
```

В этом примере рабочая группа (применяемая также в качестве доменного имени) имеет значение `SAMBA`. Когда клиент взаимодействует с сервером Samba, это имя сообщает клиенту, в какой рабочей группе находится сервер.

Тип безопасности по умолчанию установлен на значение `user` (имена пользователей и пароли Samba).

Значение `passdb backend = tdbsam` указывает на использование базы данных Samba backend для хранения паролей. Вы можете применить команду `smbpasswd` для установки пароля каждого пользователя, как описано далее.

Настройки `printing = cups` и `printcap name = cups` указывают на использование команды `printcap`, созданной службой печати CUPS. Когда вы устанавливаете значение `load printers = yes`, Samba знает, что нужно совместно задействовать все принтеры, настроенные вашей локальной службой печати CUPS из сервера.

Настройка `cups options` позволяет передавать любые параметры принтерам CUPS, обслуживаемым сервером Samba. По умолчанию установлена только настройка `raw`, что позволяет клиентам Windows использовать собственные драйверы печати. Принтеры на вашем сервере Samba печатают страницы, представленные в необработанном виде.

По умолчанию DNS-имя хоста вашего сервера (введите команду `hostname`, чтобы узнать, что это) используется также в качестве NetBIOS-имени на сервере. Вы можете изменить эту настройку и установить отдельное имя NetBIOS, раскомментировав строку `netbios name` и добавив нужное имя сервера. Например, `netbios name = myownhost.localhost` задействуется в качестве имени NetBIOS, если оно не было задано иначе.

Если требуется ограничить доступ к серверу Samba так, чтобы он отвечал только на определенные интерфейсы, можете раскомментировать строку `interfaces` и добавить либо IP-адрес, либо имена (`lo`, `eth0`, `eth1` и т. д.) нужных сетевых интерфейсов.

Вы также можете ограничить доступ к серверу Samba определенными хостами. Раскомментируйте строку `hosts allow` (удалите точку с запятой) и вставьте IP-адреса хостов, которые нужно разрешить. Чтобы ввести диапазон адресов, просто завершите часть адреса подсети, после которой стоит точка. Например, `127.` связан с IP-адресами, которые указывают на локальный хост. Строка `192.168.12.` соответствует всем IP-адресам от 192.168.12.1 до 192.168.12.254.

Настройка раздела [homes]

Раздел `[homes]` по умолчанию настроен таким образом, чтобы любая учетная запись пользователя сервера Samba могла получить доступ к собственному домашнему каталогу через сервер. Вот как выглядит раздел `[homes]` по умолчанию:

```
[homes]
       comment = Home Directories
       valid users = %S, %D%w%S
       browseable = No
       read only = No
       inherit acls = Yes
```

Установка `valid users` в значение `%S` заменяет текущее имя службы, что позволяет любым пользователям службы получить доступ к своим домашним каталогам.

Допустимые пользователи также идентифицируются доменом или рабочей группой (`%D`), разделителем `winbind` (`%w`) и именем текущей службы (`%S`).

Параметр `browseable = No` запрещает серверу Samba отображать доступные общие домашние каталоги. Пользователи, которые могут предоставить собственные имена пользователей и пароли Samba, могут читать и писать в своих домашних каталогах (`read only = no`). Если для параметра `inherit acls` установлено значение `Yes`, списки контроля доступа можно наследовать, чтобы добавить еще один уровень безопасности к общим файлам.

Если после запуска службы `smb` вы не можете войти в систему с помощью действительной учетной записи пользователя, необходимо изменить некоторые функции безопасности в системе. В частности, в системах Fedora и RHEL нужно изменить функции SELinux, чтобы пользователи могли получить доступ к своим домашним каталогам, если система SELinux находится в принудительном режиме.

Например, если вы попытаетесь применить команду `smbclient` для входа в свой домашний каталог, то войдете успешно, но при попытке перечислить содержимое домашнего каталога появится следующее сообщение:

```
NT_STATUS_ACCESS_DENIED listing \*
```

Чтобы настроить SELinux и разрешить пользователям Samba доступ к своим домашним каталогам в качестве общих ресурсов Samba, включите логический тип `samba_enable_home_dirs`, введя следующее как суперпользователь из оболочки:

```
# setsebool -P samba_enable_home_dirs on
```

Команда `setsebool` разрешает совместное применение домашних каталогов (что по умолчанию отключено). Сначала создайте пароль для пользователя с помощью команды `smbpasswd`, а затем войдите в систему посредством команды `smbclient`. Команда `smbclient` для проверки доступа к домашнему каталогу пользователя `chris` будет выглядеть так (замените IP-адреса именем или адресом вашего сервера Samba):

```
$ smbpasswd -a chris
New SMB password: *********
Retype new SMB password: *********
Added user chris.

$ smbclient -U chris //192.168.0.119/chris

Enter SAMBA\chris's password:
Try "help" to get a list of possible commands.
smb: \> ls file.txt
    file.txt 149946368  Sun Jan  4 09:28:53 2020
            39941 blocks of size 524288. 28191 blocks available
smb:\>  quit
```

Главное, что нужно помнить: даже если общий ресурс недоступен для просмотра, вы можете запросить его, указав имя хоста или IP-адрес сервера Samba, а затем имя пользователя (в данном случае `chris`), чтобы получить доступ к домашнему каталогу пользователя.

Настройка раздела [printers]

Любой принтер, настроенный для печати CUPS в системе Linux, автоматически передается другим пользователям через сервер на основе раздела `[printers]`, добавленного по умолчанию. Параметр `global cups options = raw` определяет все принтеры как необработанные (это означает, что клиент Windows должен предоставить соответствующий драйвер принтера для каждого общего принтера).

Вот как выглядит раздел `[printers]` по умолчанию в файле `smb.conf`:

```
[printers]
        comment = All Printers
        path = /var/tmp
        printable = Yes
        create mask = 0600
        browseable = No
```

Параметр `path` указывает серверу хранить временные файлы печати в файле `/var/tmp`. Строка `printable = Yes` определяет то, что все принтеры CUPS в локальной системе будут совместно использоваться сервером Samba. Принтеры доступны для записи и по умолчанию разрешают гостевую печать. Параметр `create mask = 0600`, применяемый в примере, удаляет биты записи и выполнения для группы и других в списках доступа, когда файлы создаются в каталоге `path`.

Чтобы убедиться, что локальные принтеры доступны, можете запустить команду `smbclient -L` из системы Linux, как было показано ранее. В системе Windows в окне File Explorer (Проводник) выберите Network (Сеть) и значок сервера Samba. В этом окне отображаются все общие принтеры и папки. (Подробнее о просмотре и использовании общих принтеров см. раздел «Доступ к общим ресурсам Samba» далее в этой главе.)

Создание папки общего доступа для сервера Samba

Нельзя добавить то, чего не существует, поэтому прежде всего общую папку (или каталог) нужно создать и присвоить ей соответствующие права. В приведенном далее примере каталог `/var/salesdata` общий. К примеру, вы хотите, чтобы данные были доступны для записи пользователем с именем `chris`, но видны всем в вашей сети. Чтобы создать этот каталог и установить соответствующие права и контексты файлов SELinux, введите от имени суперпользователя следующее:

```
# mkdir /var/salesdata
# chmod 775 /var/salesdata
# chown chris:chris /var/salesdata
# semanage fcontext -a -t samba_share_t /var/salesdata
# restorecon -v /var/salesdata
# touch /var/salesdata/test
# ls -lZ /var/salesdata/test
-rw-r--r--. 1 root root
    unconfined_u:object_r:samba_share_t:s0 0 Dec 24 14:35
        /var/salesdata/test
```

Добавление папки общего доступа на сервер

Если каталог `/var/salesdata` создан и правильно настроен для совместного использования Samba, то вот как может выглядеть общая папка (называемая `salesdata`) в файле `smb.conf`:

```
[salesdata]
        comment = Sales data for current year
        path = /var/salesdata
        read only = no
;       browseable = yes
        valid users = chris
```

Перед формированием общего ресурса был создан каталог `/var/salesdata` с пользователем `chris`, назначенным в качестве пользователя и группы, и каталог был настроен на чтение и запись для него. (Контекст файла SELinux также должен быть установлен, если SELinux находится в принудительном режиме.) Имя пользователя Samba `chris` должно быть представлено вместе с паролем для доступа к общему ресурсу. После того как пользователь `chris` подключен к общему ресурсу, он может его читать и записывать в нем (`read only = no`).

Теперь, когда вы ознакомились с настройками Samba по умолчанию и настройками простого общего каталога (папки), изучите следующие несколько разделов, чтобы узнать, как еще настроить общие ресурсы. В частности, в примерах показано, как сделать общие ресурсы доступными для конкретных пользователей, хостов и сетевых интерфейсов.

Проверка общего доступа на сервере

Чтобы изменения в конфигурации Samba вступили в силу, необходимо перезапустить службу `smb`. После этого убедитесь, что созданный вами общий ресурс Samba открыт и любой пользователь, назначенный этому ресурсу, может получить к нему доступ. Чтобы сделать это, введите от имени суперпользователя из оболочки на сервере Samba следующие команды:

```
# systemctl restart smb.service
# smbclient -L localhost -U chris
Enter SAMBA\chris's password: *******
    Sharename       Type      Comment
    ---------       ----      -------
    salesdata       Disk      Sales data for current year
    print$          Disk      Printer Drivers
    IPC$            IPC       IPC Service (Samba 4.10.4)
    chris           Disk      Home Directories
Reconnecting with SMB1 for workgroup listing.
    Server               Comment
    ---------            -------
```

```
    Workgroup            Master
    ---------            -------
    SAMBA                FEDORA30
...
```

Здесь вы можете увидеть имя общего ресурса (`salesdata`), домен, установленный на имя рабочей группы `SAMBA`, и описание (`Sales data for current year`). Быстрый способ проверить доступ к общему ресурсу — применить команду `smbclient`. Вы можете использовать имя хоста или IP-адрес с командой `smbclient` для доступа к общему ресурсу. Поскольку в этом примере я нахожусь в локальной системе, то просто задействую имя `localhost` и добавленного пользователя (`chris`):

```
# smbclient -U chris //localhost/salesdata
Enter SAMBA\chris's password: ********
Try "help" to get a list of possible commands.
smb: \> lcd /etc
smb: \> put hosts
putting file hosts as \hosts (43.5 kb/s) (average 43.5 kb/s)
smb: \> ls
    .                                   D        0  Sun Dec 29 09:52:51 2020
    ..                                  D        0  Sun Dec 29 09:11:50 2020
    Hosts                               A       89  Sun Dec 29 09:52:51 2020
          39941 blocks of size 524288. 28197 blocks available
smb: \> quit
```

Общие ресурсы Samba — это `//host/share` или `\\host\share`. Но когда вы идентифицируете общий ресурс Samba из оболочки Linux как второй вариант, обратные косые черты должны быть экранированы. То есть в качестве аргумента первый пример должен был бы выглядеть как `\\\\localhost\\salesdata`. Таким образом, первая форма проще в применении.

ПРИМЕЧАНИЕ

Поставив обратную косую черту (\) перед символом, вводимым из оболочки, вы экранируете его. Это приказывает оболочке использовать символ, следующий за обратной косой чертой, буквально, вместо того чтобы придавать ему особое значение. (* и ? — это примеры символов с особым значением.) Поскольку сама обратная косая черта имеет особое значение для оболочки, то если вы хотите применять ее в буквальном значении, нужно поставить перед ней обратную косую черту. Вот почему в адресе Samba, который включает в себя две обратные косые черты, их нужно набрать четыре.

При появлении запроса введите пароль Samba для этого пользователя (он может отличаться от пароля пользователя Linux). Пароль пользователя Samba устанавливается с помощью команды `smbpasswd`. После этого появится приглашение `smb:\>`.

На этом этапе открывается сеанс для хоста Samba, который похож на сеанс `ftp` для обхода FTP-сервера. Команда `lcd /etc` назначает `/etc` текущим каталогом в локальной системе. Команда `put hosts` загружает файл `hosts` из локальной системы в общий каталог. Команда `ls` показывает, что файл существует на сервере. Команда `quit` завершает сеанс.

Ограничение доступа к серверу Samba по сетевому интерфейсу

Чтобы ограничить доступ ко всем общим ресурсам, установите параметр `global interfaces` в файле `smb.conf`. Сервер Samba предназначен скорее для локального обмена файлами, чем для обмена по широким сетям. Если у компьютера есть сетевой интерфейс, подключенный к локальной сети, и интерфейс, подключенный к Интернету, то стоит открывать доступ только к локальной сети.

Чтобы установить, какие интерфейсы прослушивает сервер Samba, раскомментируйте строку `interfaces` из предыдущего примера в разделе `[global]` файла `smb.conf`. Затем добавьте имена интерфейсов или диапазоны IP-адресов тех компьютеров, которым вы хотите разрешить доступ к вашему компьютеру, например:

```
interfaces = lo 192.168.22.15/24
```

Такая строка интерфейсов позволяет получить доступ к службе Samba всем пользователям локальной системы (`lo`). Она также позволяет получить доступ к любым системам в сети 192.168.22. См. описание различных способов идентификации хостов и сетевых интерфейсов на справочной странице `smb.conf`.

Ограничение доступа к серверу Samba по хосту

Доступ хоста к серверу Samba может быть установлен для всей службы или для отдельных общих ресурсов.

Вот примеры строк `hosts allow` и `hosts deny`:

```
hosts allow = 192.168.22. EXCEPT 192.168.22.99
hosts allow = 192.168.5.0/255.255.255.0
hosts allow = .example.com market.example.net
hosts deny = evil.example.org 192.168.99.
```

Эти строки можно поместить в раздел `[global]` или в любой раздел общего каталога. Первый пример разрешает доступ к любому хосту в сети 192.168.22., за исключением 192.168.22.99. Обратите внимание на то, что в конце номера сети требуется точка. В примере с 192.168.5.0/255.255.255.0 используется нотация маски сети для идентификации 192.168.5 как набора разрешенных адресов.

В третьей строке примера кода любой хост из сети `example.com` разрешен, как и отдельный хост `market.example.net`. Пример `hosts deny` показывает, что вы можете применять одну и ту же форму для идентификации имен и IP-адресов, чтобы предотвратить доступ с определенных хостов.

Ограничение доступа пользователя к серверу Samba

Определенным пользователям и группам Samba можно разрешить доступ к части общих ресурсов Samba, идентифицировав их в файле `smb.conf`. Помимо гостевых пользователей, доступ которым можно разрешить или не разрешить, аутентификация пользователя по умолчанию для Samba требует добавления учетной записи пользователя Samba (Windows), которая сопоставляется с локальной учетной записью пользователя Linux.

Чтобы разрешить пользователю доступ к серверу Samba, необходимо создать для него пароль. Вот пример того, как добавить пароль Samba для пользователя `jim`:

```
# smbpasswd -a jim
New SMB password: *******
Retype new SMB password: *******
```

После выполнения команды `smbpasswd` пользователь `jim` может задействовать это имя пользователя и пароль для доступа к серверу Samba. Файл `/var/lib/samba/private/passdb.tdb` содержит только что введенный пароль для `jim`. Последний может изменить пароль, просто набрав команду `smbpasswd` при входе в систему. Суперпользователь также может изменить пароль, повторно выполнив команду, показанную в примере, но не применяя параметр `-a`.

Если вы хотите предоставить пользователю `jim` доступ к общему ресурсу, добавьте строку `valid users` к общему блоку в файле `smb.conf`. Например, чтобы предоставить доступ к общему ресурсу как пользователю `chris`, так и пользователю `jim`, можно добавить следующую строку:

```
valid users = jim, chris
```

Если для общего ресурса установлен параметр `read only`, то оба пользователя потенциально могут записывать файлы в общий ресурс (в зависимости от прав доступа к файлам). Если значение `read only` равно `yes`, вы все равно можете разрешить доступ пользователям `jim` и `chris` для записи файлов, добавив строку `write list` следующим образом:

```
write list = jim, chris
```

Строка `Write list` может содержать группы (то есть группы Linux, содержащиеся в файле `/etc/group`), чтобы разрешить запись любому пользователю Linux, принадлежащему к определенной группе Linux. Вы можете добавить права на запись для группы, поставив знак плюс (+) перед именем. Например, следующая строка дает группе `market` доступ на запись к общему ресурсу, с которым связана эта строка:

```
write list = jim, chris, +market
```

Существует множество способов изменить и расширить возможности общих ресурсов Samba. Для получения дополнительной информации о настройке Samba обязательно ознакомьтесь с самим файлом `smb.conf`, который включает в себя множество полезных заметок, и справочной страницей `smb.conf`.

Доступ к общим ресурсам Samba

После создания несколько общих каталогов в Samba многие инструменты для клиентов, дающие доступ к этим общим папкам, становятся доступны как в Linux, так и в Windows. Инструменты командной строки в Linux включают команду `smbclient`, описанную ранее в этой главе. Как графическое средство доступа к общим ресурсам можно использовать файловые менеджеры, имеющиеся в Windows (Проводник) и Linux (Файлы с рабочим столом GNOME).

Доступ к общим ресурсам Samba в Linux

Как только общий ресурс Samba будет создан, к нему можно обратиться из удаленных систем Linux и Windows с помощью файловых менеджеров или команд удаленного монтирования.

Доступ к общим ресурсам Samba через файловый менеджер Linux

Файловый менеджер в Linux предоставляет доступ к общим каталогам из Linux (Samba) и Windows (SMB). Файловые менеджеры различаются на разных рабочих столах Linux. В GNOME 3 можете нажать на значок Files (Файлы). На других рабочих столах откройте домашнюю папку (Home).

В окне Nautilus (Файлы) выберите слева вкладку Other Locations (Другие места). Появятся доступные сети (например, сеть Windows). В нижней части окна вы увидите поле Connect to Server (Подключение к серверу), введите в него местоположение доступного общего ресурса Samba. Учитывая предыдущие примеры, используйте один из следующих вариантов:

```
smb://192.168.122.119/chris
smb://192.168.122.119/salesdata
```

Появится окно, изображенное на рис. 19.1.

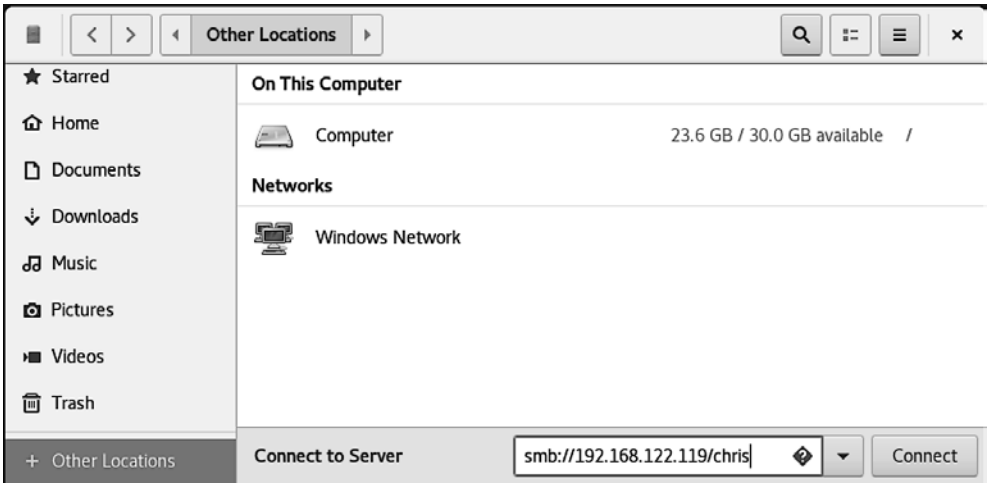


Рис. 19.1. Общий ресурс Samba из окна Nautilus — Connect to Server

Нажмите кнопку Connect (Присоединиться). В появившемся окне (рис. 19.2) вы можете подключиться в качестве зарегистрированного пользователя. Для этого нужно ввести свое имя пользователя, доменное имя Samba и пароль пользователя. Вы также можете выбрать, сохранить ли введенный пароль на будущее.

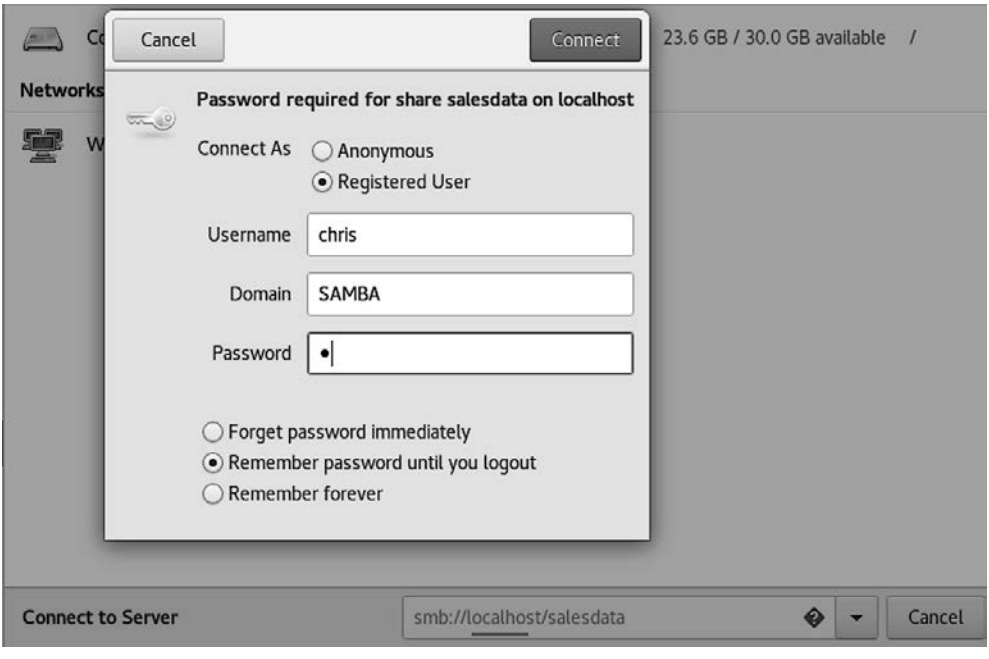


Рис. 19.2. Добавьте учетные данные на сервер Samba

Нажмите кнопку Connect (Присоединиться). Если пользователь и пароль верны, вы увидите содержимое удаленного каталога. Если у вас есть право на запись в общий ресурс, можете открыть другое окно Nautilus (Файлы) и перетащить файлы между двумя системами. На рис. 19.3 показан пример окна Nautilus (Файлы) после подключения к общему ресурсу `salesdata`.

Рис. 19.3. Общий ресурс Samba в окне Nautilus

Монтирование общего ресурса Samba из командной строки Linux

Поскольку общий каталог Samba можно рассматривать как удаленную файловую систему, подключайте ресурс Samba (временно или постоянно) к своей системе Linux с помощью общих инструментов Linux. Используя стандартную команду монтирования (с установленным параметром `cifs-utils`), вы можете монтировать удаленный общий ресурс Samba в качестве файловой системы CIFS в Linux. В этом примере общий ресурс `salesdata` монтируется с хоста по IP-адресу 192.168.0.119 в локальном каталоге `/mnt/sales`:

```
# yum install cifs-utils -y
# mkdir /mnt/sales
# mount -t cifs -o user=chris \
      //192.168.0.119/salesdata /mnt/sales
Password for chris@//192.168.122.119/salesdata: *******
# ls /mnt/sales
hosts   memos  test  whitepapers
```

При появлении запроса введите пароль Samba для пользователя `chris`. Учитывая, что он в этом примере имеет права на чтение из общего каталога и запись в него, пользователи вашей системы должны иметь возможность читать из подключенного каталога и записывать в него. Независимо от того, кто сохраняет файлы в общем каталоге, на сервере эти файлы принадлежат пользователю `chris`.

Монтирование длится до тех пор, пока система не будет перезагружена или вы не выполните команду `umount` в каталоге. Если хотите, чтобы общий ресурс монтировался постоянно (то есть каждый раз, когда система загружается) в одном и том же месте, выполните дополнительную настройку. Сначала откройте файл `/etc/fstab` и добавьте запись, как в примере далее:

```
//192.168.0.119/salesdata /mnt/sales cifs credentials=/root/cif.txt 0 0
```

Затем создайте файл учетных данных (в этом примере `/root/cif.txt`). В него введите имя пользователя и его пароль, который нужно задействовать, когда система попытается смонтировать файловую систему. Вот пример содержимого этого файла:

```
user=chris
pass=mypass
```

Перед перезагрузкой, чтобы проверить правильность записи, установите ее из командной строки. Команда `mount -a` попытается смонтировать любую еще не смонтированную файловую систему, указанную в файле `/etc/fstab`. Команда `df` отображает информацию о дисковом пространстве для подключенного каталога, как в следующем примере:

```
# mount -a
# df -h /mnt/sales
Filesystem                  Size  Used  Avail   Ues%  Mounted on
//192.168.0.119/salesdata    20G  5.7G    14G    30%  /mnt/sales
```

Теперь можете использовать общий каталог Samba, как и любой другой каталог в локальной системе.

Доступ к общим ресурсам Samba в системе Windows

Как и в Linux, вы можете получить доступ к общим ресурсам Samba из окна файлового менеджера, в данном случае File Explorer (Проводник) в системе Windows. Для этого откройте любую папку в Windows и на панели слева выберите пункт Network (Сеть). На экране должен появиться значок, представляющий сервер Samba. Нажмите на него и, если необходимо, введите пароль. В окне появятся все общие принтеры и папки с этого сервера (рис. 19.4).

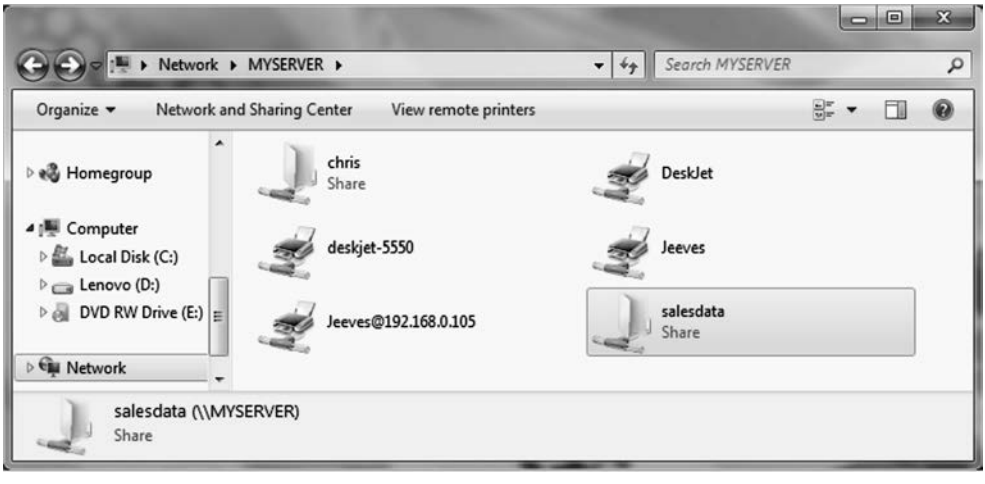


Рис. 19.4. Общий доступ Samba в системе Windows

На рис. 19.4 видно, что существуют две общие папки (каталоги): `chris` и `salesdata`. Есть также несколько общих принтеров. Чтобы работать с папками, дважды щелкните на них и введите необходимые данные аутентификации. Поскольку принтеры по умолчанию настроены на использование необработанных драйверов, вам необходимо применять драйверы Windows для любого из принтеров Samba.

Сервер Samba на предприятии

Данная тема выходит за рамки данной книги, однако общий доступ к файлам и принтерам Windows через серверы Samba часто встречается на крупных предприятиях. Несмотря на то что Linux начал доминировать на рынке корпоративных серверов, системы Microsoft Windows по-прежнему преобладают на настольных компьютерах.

Основные функции, необходимые для внедрения серверов Samba на крупном предприятии со множеством настольных компьютеров Microsoft Windows, связаны с аутентификацией. Большинство крупных предприятий используют серверы Microsoft Active Directory Services (ADS) для проверки подлинности. На стороне

Linux это означает настройку Kerberos в системе Linux и применение ADS (вместо `user`) для реализации безопасности в файле `smb.conf`.

Преимущество централизованной аутентификации заключается в том, что пользователи должны помнить только один набор учетных данных для всего предприятия, а системные администраторы — управлять меньшим количеством учетных записей пользователей.

Резюме

Из-за популярности настольных компьютеров Windows серверы Samba стали чаще всего использоваться для обмена файлами и принтерами между системами Windows и Linux. Сервер Samba предоставляет способ взаимодействия с системами Windows путем реализации протокола Server Message Block (SMB) или Common Internet File (CIFS) для совместного применения ресурсов по сети.

В этой главе описаны установка, запуск, защита, настройка и получение доступа к серверам Samba в системе Linux. Используя инструменты командной строки, я продемонстрировал, как настроить сервер Samba. Я также показал как инструменты командной строки, так и инструменты рабочего стола для открытия доступа к общим ресурсам Samba из систем Linux и Windows.

В следующей главе описывается Network File System (NFS, сетевая файловая система). NFS — это функция Linux, созданная для совместного использования и монтирования файловых систем с другими системами Linux и UNIX с помощью сетей.

Упражнения

Упражнения в этом разделе относятся к задачам, связанным с настройкой сервера Samba в Linux и получения доступа к нему с помощью клиента Samba. Как и ранее, большинство упражнений можно решить несколькими способами. Поэтому не волнуйтесь, если вы получили те же результаты, какие показаны в ответах, но выполнили упражнения не так. В приложении Б представлены варианты решения упражнений.

Не выполняйте задания в системе Linux, работающей на сервере Samba, потому что они могут навредить работе сервера. Эти упражнения протестированы на системе Fedora. Для других систем шаги выполнения могут отличаться.

1. Установите пакеты `samba` и `samba-client`.
2. Запустите и подключите службы `smb` и `nmb`.
3. Установите рабочую группу сервера Samba в значение `TESTGROUP`, имя `netbios` — в значение `MYTEST`, а строку сервера — в значение `Samba Test System`.
4. Добавьте в свою систему пользователя Linux с именем `phil`, а также пароль Linux и пароль Samba для него.

5. Установите раздел [homes] таким образом, чтобы домашние каталоги были доступны для просмотра (yes) и записи (yes), а phil стал единственным допустимым пользователем.
6. Установите любой логический тип SELinux, необходимый для того, чтобы пользователь phil мог получить доступ к своему домашнему каталогу через клиент Samba, а затем перезапустите службы smb и nmb.
7. В локальной системе применяйте команду smbclient, чтобы указать, что домашний общий ресурс доступен.
8. Из окна Nautilus (Файлы) в локальной системе подключитесь к домашнему общему ресурсу пользователя phil на локальном сервере Samba таким образом, чтобы можно было перетаскивать файлы в эту папку.
9. Откройте брандмауэр, чтобы любой имеющий доступ к серверу мог получить доступ к службе Samba (демоны smbd и nmbd).
10. Из другой системы в вашей сети (Windows или Linux) снова откройте общий ресурс [homes] от имени пользователя phil и убедитесь, что можете перетаскивать в него файлы.

20 Настройка NFS-сервера

В этой главе

- Программное обеспечение для сервера NFS.
- Подключение и запуск сервера NFS.
- Экспорт каталогов сервера NFS.
- Настройка функций безопасности для сервера NFS.
- Монтирование удаленных общих каталогов сервера NFS.

Вместо обозначения устройств хранения данных буквами (A, B, C и т. д.), как происходит в операционных системах Microsoft, системы Linux объединяют файловые системы с нескольких жестких дисков, USB-накопителей, CD-ROM и других локальных устройств в единую файловую систему Linux. Протокол Network File System (NFS) позволяет расширить файловую систему Linux для подключения файловых систем других компьютеров к локальной структуре каталогов.

Файловый сервер NFS обеспечивает простой способ обмена большими объемами данных между пользователями и компьютерами на предприятии. Администратор системы Linux, настроенной для совместного применения своих файловых систем с помощью сервера NFS, должен выполнить следующие настройки NFS.

1. **Настроить сеть.** Протокол NFS обычно задействуется в частных сетях, а не в публичных, таких как Интернет.
2. **Запустить службу NFS.** Чтобы полноценно подключить службу NFS, необходимо активизировать несколько ее демонов. В системах Fedora и Red Hat Enterprise Linux можно запустить службу командой `nfsserver`.
3. **Выбрать, чем делиться на сервере.** Решите, какие каталоги (папки) на своем сервере Linux NFS сделать доступными для других компьютеров. Вы можете выбрать в файловой системе любую точку и сделать все файлы и каталоги ниже ее доступными для других компьютеров.

4. **Обеспечить безопасность сервера.** Применяйте различные функции безопасности, чтобы установить наиболее подходящий уровень безопасности. Безопасность на уровне монтирования позволяет ограничить компьютеры, которые могут монтировать данные на сервере, и для тех, кому разрешено монтировать его, — указать, может ли он быть смонтирован только для чтения/записи или только для чтения. Безопасность на уровне пользователя реализуется сопоставлением пользователей из клиентских систем с пользователями на сервере NFS (на основе UID, а не имени пользователя), чтобы они могли полагаться на стандартные разрешения Linux на чтение/запись/выполнение, владение файлами и групповые права доступа к файлам и их защиты.
5. **Смонтировать файловую систему на клиенте.** Каждый клиентский компьютер, которому разрешен доступ к общей файловой системе NFS сервера, может монтировать ее в любом месте по выбору. Например, можете смонтировать файловую систему с компьютера под названием `oak` в каталоге `/mnt/oak` своей локальной файловой системы. После монтирования вы сможете просмотреть содержимое этого каталога, набрав команду `ls /mnt/oak`.

Хотя этот сервер часто применяется в качестве файлового (или другого типа), Linux является операционной системой общего назначения, поэтому любая система Linux может совместно использовать или экспортировать файловые системы в качестве сервера или задействовать файловые системы другого компьютера (монтировать) в качестве клиента. Фактически системы Red Hat Enterprise Linux 8 и Fedora 30 Workstation включают в себя службу `nfs-server` по умолчанию.

ПРИМЕЧАНИЕ

Файловая система обычно представляет собой структуру файлов и каталогов, которая существует на одном устройстве (например, на жестком диске или CD). Термин «файловая система Linux» относится ко всей структуре каталогов (которая может включать файловые системы из нескольких разделов диска, NFS или различных сетевых ресурсов), начиная с каталога root (/) на одном компьютере. Общий каталог в NFS может представлять собой всю файловую систему компьютера или ее часть, которые могут быть присоединены (из общего каталога вниз по дереву каталогов) к файловой системе другого компьютера.

Если в вашей системе уже запущены службы NFS и Cockpit, вы можете монтировать общие ресурсы NFS и просматривать смонтированные общие ресурсы из веб-интерфейса Cockpit. Вот как это сделать.

1. Войдите в свой аккаунт в интерфейсе Cockpit (порт 9090) через браузер и перейдите на вкладку Storage (Хранилище). URL-адрес для доступа к хранилищу в службе Cockpit в локальной системе должен быть примерно таким: https://host1.example.com:9090/storage.
2. Если в вашей системе есть смонтированные общие ресурсы NFS, они должны появиться в разделе монтирования NFS. На рис. 20.1 показаны два подключенных общих ресурса NFS.

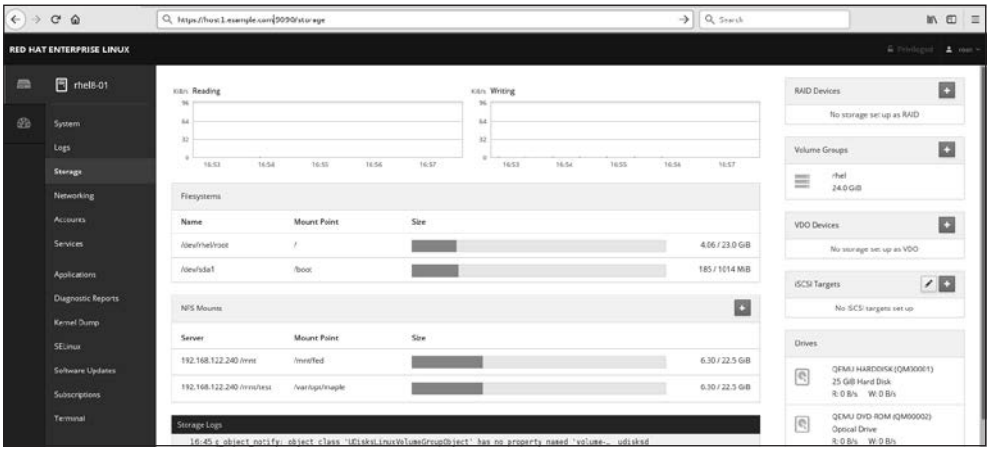


Рис. 20.1. Общие ресурсы NFS монтируются локально через веб-интерфейс Cockpit

3. Чтобы смонтировать удаленный общий ресурс NFS, выберите знак плюс (+) в строке NFS Mounts (Монтирование NFS). Введите адрес или имя хоста сервера NFS, общий каталог на общем ресурсе NFS и точку в локальной файловой системе, где будете монтировать общий ресурс. Затем нажмите кнопку Add (Добавить) (рис. 20.2).

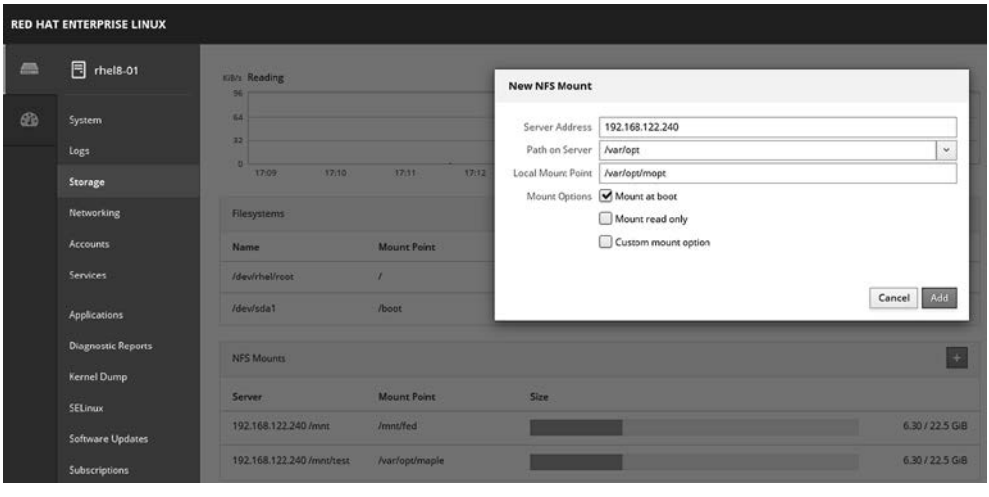


Рис. 20.2. Добавление новой точки монтирования NFS через веб-интерфейс Cockpit

На этом этапе вы получаете доступ к содержимому удаленного общего ресурса NFS из точки монтирования в локальной файловой системе. По умолчанию информация о монтировании NFS добавляется в файл `/etc/fstab`, поэтому общий

ресурс NFS будет доступен при каждой перезагрузке системы. Теперь, после того как мы рассмотрели простейший способ применения NFS, поговорим о том, как настраивать и использовать NFS с нуля.

Установка сервера NFS

Чтобы запустить сервер NFS, вам нужны набор модулей ядра (они поставляются вместе с самим ядром), а также инструменты пользовательского уровня для настройки службы, запуска демонов и запроса службы различными способами.

Для более ранних версий дистрибутивов Fedora и RHEL необходимые компоненты, которых нет в ядре, можно добавить, установив пакет `nfs-utils`. В RHEL 8 и Fedora 30 необходимые компоненты включены в установку по умолчанию следующей командой:

```
# yum install nfs-utils
```

Помимо нескольких документов в каталоге `/usr/share/doc/nfs-utils`, большая часть документации, содержащейся в пакете `nfs-utils`, включает в себя справочные страницы различных его компонентов. Чтобы перечислить список документов, введите команду:

```
# rpm -qd nfs-utils | less
```

Существуют инструменты и справочные страницы как для сервера NFS (совместная с другими пользователями работа с каталогом), так и для клиентов NFS (локальное монтирование удаленного каталога NFS). Чтобы настроить сервер, вы можете обратиться к справочной странице экспорта (чтобы настроить файл `/etc/exports` для совместного применения ваших каталогов). На справочной странице команды `exportfs` описано, как совместно использовать и просматривать список каталогов, которыми вы делитесь из файла `/etc/exports`. Справочная страница `nfsd` описывает параметры, которые можно передать демону сервера `rpc.nfsd`, что позволит запускать сервер в режиме отладки.

Справочные страницы на стороне клиента включают страницу `mount.nfs`, чтобы узнать, какие параметры можно задействовать при монтировании удаленных каталогов NFS в локальной системе. Существует также справочная страница `nfsmount.conf`, которая описывает, как использовать файл `/etc/nfsmount.conf` для настройки поведения вашей системы при локальном монтировании удаленных ресурсов. На справочной странице `showmount` описано, как применять команду `showmount` для просмотра общих каталогов, доступных с серверов NFS.

Чтобы просмотреть информацию и больше узнать о пакете `nfs-utils`, его файлах конфигурации и командах, введите соответственно команды:

```
# rpm -qi nfs-utils
# rpm -qc nfs-utils
# rpm -ql nfs-utils | grep bin
```

Запуск службы NFS

Запуск сервера NFS включает в себя запуск нескольких демонов. Базовая служба NFS в системах Fedora и RHEL 8 называется `nfs-server`. Чтобы запустить эту службу, включите ее (чтобы она запускалась каждый раз при загрузке системы) и проверьте состояние, выполнив следующие три команды:

```
# systemctl start nfs-server.service
# systemctl enable nfs-server.service
# systemctl status nfs-server.service
• nfs-server.service – NFS server and services
     Loaded: loaded (/lib/systemd/system/nfs-server.service; enabled
             vendor preset: disabled)
     Active: active (exited) since Mon 2019-9-02 15:15:11 EDT; 24s ago
   Main PID: 7767 (code=exited, status=0/SUCCESS)
      Tasks: 0 (limit: 12244)
     Memory: 0B
     CGroup: /system.slice/nfs-server.service
```

Из состояния строки видно, что служба `nfs-server` включена и активна. Служба NFS требует также, чтобы служба RPC была запущена (`rpcbind`). Служба `nfs-server` автоматически запускает службу `rpcbind`, если это еще не было сделано.

В системе Red Hat Enterprise Linux 6 для проверки, запуска и включения службы NFS (`nfs`) понадобятся команды `service` и `chkconfig`. Следующие команды показывают, что в данный момент служба `nfs` не работает и отключена:

```
# service nfs status
rpc.svcgssd is stopped
rpc.mountd is stopped
nfsd is stopped
# chkconfig --list nfs
nfs  0:off  1:off  2:off  3:off  4:off  5:off  6:off
```

Как упоминалось ранее, для работы NFS должна быть запущена служба `rpcbind`. В системе RHEL 6 можно использовать следующие команды для запуска и постоянного включения служб `rpcbind` и `nfs`:

```
# service rcpbind start
Starting rpcbind:                [  OK  ]
# service nfs start
Starting NFS services:           [  OK  ]
Starting NFS quotas:             [  OK  ]
Starting NFS daemon:             [  OK  ]
Starting NFS mountd:             [  OK  ]
# chkconfig rpcbind on
# chkconfig nfs on
```

Команды (`mount`, `exportfs` и т. д.) и файлы (`/etc/exports`, `/etc/fstab` и т. д.) настройки службы в основном одинаковы для всех систем Linux. Итак, после того, как вы установили и запустили NFS, просто следуйте инструкциям, приводимым в этой главе, чтобы начать применять службу.

Совместное использование файловых систем NFS

Чтобы совместно работать с файловой системой NFS из системы Linux, необходимо экспортировать ее с сервера. Экспорт осуществляется в Linux путем добавления записей в файл `/etc/exports`. Каждая запись определяет каталог в локальной файловой системе, который нужно задействовать совместно с другими компьютерами. Эта запись также определяет другие компьютеры, которые могут получить доступ к ресурсу (или открывают его для всех компьютеров), и включает другие параметры с правами на каталог.

Помните: когда вы делитесь каталогом, то делитесь также всеми файлами и подкаталогами, расположенными ниже этого каталога (по умолчанию). Вы должны быть уверены, что хотите поделиться всем в этой структуре каталогов. И все еще можете ограничить доступ к этой структуре каталогов несколькими способами, они будут описаны далее в этой главе.

Настройка файла /etc/exports

Чтобы сделать каталог из системы Linux доступным для других систем, нужно экспортировать его. Экспорт осуществляется на постоянной основе путем добавления информации об этом каталоге в файл `/etc/exports`.

Вот как это должно выглядеть:

```
Directory    Host (Options ...)    Host ( Options ...)    # Comments
```

В этом примере *`Directory`* — имя каталога, к которому предоставляется общий доступ, а *`Host`* — клиентский компьютер, на котором ограничен общий доступ к этому каталогу. *`Options`* может включать в себя множество параметров определения мер безопасности, прикрепленных к общему каталогу для хоста. (Пары *`Host`* и *`Option`* можно повторять.) *`Comments`* — это любые необязательные комментарии (после знака `#`).

Справочная страница `exports` (`man exports`) содержит подробную информацию о синтаксисе файла `/etc/exports`. В частности, в ней находятся параметры, которые можно использовать для ограничения доступа к каждому общему каталогу и его защиты.

Как суперпользователь задействуйте любой текстовый редактор для настройки файла `/etc/exports` и изменения записей общего каталога или добавления новых. Пример файла `/etc/exports`:

```
/cal     *.linuxtoys.net(rw)              # События компании
/pub     *(ro,insecure,all_squash)        # Общедоступный каталог
/home    maple(rw,root_squash) spruce(rw,root_squash)
```

Строка `/cal` представляет собой каталог, содержащий информацию о событиях, связанных с компанией. Любой компьютер в домене компании (`*.linuxtoys.net`)

может монтировать этот общий ресурс NFS. Пользователи могут записывать файлы в каталог, а также читать их (об этом говорит параметр `rw`). Комментарий (`# События компании`) просто служит напоминанием о том, что находится внутри каталога.

Строка `/pub` представляет собой общедоступный каталог. Он позволяет любому компьютеру и пользователю читать файлы из каталога, указанного параметром, но не записывать файлы. Параметр `insecure` позволяет любому компьютеру, даже тому, который не задействует безопасный порт NFS, получить доступ к каталогу. Параметр `all_squash` сопоставляет всех пользователей (`UID`) и все группы (`gid`) с пользователем `nobody` (`UID 65534`), предоставляя им минимальные права доступа к файлам и каталогам.

Запись `/home` позволяет группе пользователей иметь один и тот же каталог `/home` на разных компьютерах. Предположим, что вы совместно применяете каталог `/home` с компьютера с именем `oak`. Компьютеры по имени `maple` и `pruce` могут смонтировать эту папку на свой каталог `/home`. Если бы вы дали всем пользователям одинаковое имя `/UID` на всех компьютерах, у вас мог бы быть один и тот же каталог `/home/`, доступный для любого пользователя независимо от того, с какого компьютера он вошел. Параметр `Root_squash` применяется для исключения суперпользователя с другого компьютера из права доступа root к общему каталогу.

Это всего лишь примеры, вы можете совместно задействовать любые каталоги, которые выберете, включая всю файловую систему (`/`). Конечно, существуют и последствия для безопасности совместного использования всей файловой системы или ее частей, таких как каталог `/etc`. Параметры безопасности, которые можно добавить в файл `/etc/exports`, описаны в следующих разделах.

Имена хостов в файле /etc/exports

Вы можете указать в файле `/etc/exports`, какие хост-компьютеры будут иметь доступ к вашему общему каталогу. Если хотите связать несколько имен хостов или IP-адресов с определенным общим каталогом, обязательно оставьте пробел перед каждым именем хоста. Однако не добавляйте пробелы между именем хоста и его параметрами, например:

```
/usr/local maple(rw) spruce(ro,root_squash)
```

Обратите внимание на то, что после (`rw`) есть пробел, а после `maple` его нет. Хосты можно идентифицировать несколькими способами.

- **Отдельный хост.** Введите одно или несколько имен хостов TCP/IP или IP-адресов. Если хост находится в локальном домене, вы можете просто указать имя хоста. В противном случае используйте полный формат `host.domain`. Допустимые формы указания отдельных хост-компьютеров:

  ```
  maple
  maple.handsonhistory.com
  10.0.0.11
  ```

- **IP-адрес.** Разрешите доступ ко всем хостам с определенного сетевого адреса, указав номер и маску сети, разделенные косой чертой (/). Допустимые способы обозначения сетевых номеров:

```
10.0.0.0/255.0.0.0 172.16.0.0/255.255.0.0
192.168.18.0/255.255.255.0
192.168.18.0/24
```

- **Домен TCP/IP.** Используя подстановочные знаки, вы можете включить доступ ко всем или некоторым хост-компьютерам с определенного уровня домена. Вот некоторые из допустимых вариантов применения подстановочных знаков * и ?:

```
*.handsonhistory.com
*craft.handsonhistory.com
???.handsonhistory.com
```

  Первый пример соответствует всем хостам в домене handsonhistory.com. Второй соответствует woodcraft, basketcraft или любым другим именам хостов, оканчивающимся на craft, в домене handsonhistory.com. Последний пример соответствует любому трехбуквенному имени хоста в домене.

- **Группы Network Information Service (NIS, информационная служба сети).** Вы можете разрешить доступ к узлам, содержащимся в группе NIS. Чтобы указать группу NIS, перед именем группы поставьте знак «эт» (@) (например, @group).

Параметры доступа в файле /etc/exports

Нет необходимости просто открывать доступ к своим файлам и каталогам при экспорте каталога с помощью NFS. В разделе параметров каждой строки в файле /etc/exports можно добавить параметры, разрешающие или ограничивающие доступ, установив права на чтение/запись. Параметры, которые передаются в NFS:

- ro. Клиент может монтировать эту экспортированную файловую систему только для чтения. По умолчанию файловая система монтируется для чтения/записи;
- rw. Запрашивает общий каталог с правами на чтение и запись. (Если клиент выбирает этот вариант, он все равно может смонтировать каталог только для чтения.)

Параметры отображения пользователя в файле /etc/exports

В дополнение к параметрам, определяющим, как обрабатываются права в целом, можно применять параметры для установки прав, которые имеют определенные пользователи, для общих файловых систем NFS.

Один из методов, упрощающих этот процесс, состоит в том, чтобы каждый пользователь с несколькими учетными записями имел одни и те же имя пользователя

и UID на всех компьютерах. Это упрощает сопоставление пользователей, и они будут иметь одни и те же права доступа к подключенной файловой системе и к файлам, хранящимся на их локальных жестких дисках. Если этот метод неудобен, ID пользователей можно сопоставить со множеством других способов. Вот вариант настройки прав пользователя и параметра `/etc/exports`, который применяется для каждого метода.

- **root-пользователь.** Суперпользователь клиента по умолчанию сопоставляется с именем пользователя `nobody` (UID 65534). Это не позволяет суперпользователю клиента изменять все файлы и каталоги в общей файловой системе. Если вы хотите, чтобы он имел права root на сервере, примените параметр `no_root_squash`.

СОВЕТ

Имейте в виду, что даже если права суперпользователя ограничены, суперпользователь из клиента все равно может стать любой другой учетной записью пользователя и получить доступ к файлам этих учетных записей на сервере. Поэтому убедитесь, что вы доверяете суперпользователю все свои пользовательские данные, прежде чем делиться ими для чтения/записи с клиентом.

- **Группа/пользователь nfsnobody или nobody.** Указывая ID пользователя 65534 и ID группы, вы, по сути, создаете пользователя/группу с правами, которые не разрешают доступ к файлам, принадлежащим каким-либо реальным пользователям на сервере, если только они не открывают доступ всем. Однако файлы, созданные пользователем или группой 65534, доступны любому пользователю или группе 65534. Чтобы установить всех удаленных пользователей на ID 65534, примените параметр `all_squash`.

  С помощью UID и GID 65534 предотвращают слияние идентификатора с действительным идентификатором пользователя или группы. Взяв параметры `anonuid` и `anongid`, вы можете изменить пользователя или группу 65534 соответственно. Например, `anonuid=175` устанавливает для всех анонимных пользователей UID 175, а `anongid=300` задает для GID значение 300. (При указании прав файла отображается только номер, если вы не добавляете строки с именами новых UID и GID в файлы `/etc/passwd` и `/etc/group`.)

- **Сопоставление пользователей.** Если у пользователя есть учетные записи входа на нескольких компьютерах (и один и тот же идентификатор), сервер NFS по умолчанию сопоставляет этот идентификатор. Это означает, что, если пользователь с именем `mike` (UID 110) на компьютере `maple` имеет учетную запись на `pine` (`mike`, UID 110), он может работать с собственными удаленно смонтированными файлами на любом компьютере.

  Если клиентский пользователь, не настроенный на сервере, создает файл в смонтированном каталоге NFS, этот файл присваивается UID и GID удаленного клиента. (То есть команда `ls -l` на сервере показывает UID владельца.)

Экспорт общих файловых систем

После добавления строк в файл `/etc/exports` выполните команду `exportfs`, чтобы экспортировать эти каталоги (сделать их доступными для других компьютеров в сети). Перезагрузите компьютер или перезапустите службу NFS, и команда `exportfs` автоматически запустится и начнет экспортировать каталоги. Если вы хотите экспортировать их немедленно, запустите `exportfs` из командной строки (от имени суперпользователя).

СОВЕТ

Запустите команду exportfs после изменения файла экспорта. Если в файле есть какие-либо ошибки, команда их найдет и идентифицирует.

Пример вывода команды `exportfs`:

```
# /usr/sbin/exportfs -a -r -v
exporting maple:/pub
exporting spruce:/pub
exporting maple:/home
exporting spruce:/home
exporting *:/mnt/win
```

Параметр `-a` указывает на то, что все каталоги, перечисленные в файле `/etc/exports`, должны быть экспортированы. Параметр `-r` повторно синхронизирует все действия экспорта с текущим файлом `/etc/exports`, отключая действия, которых больше нет в файле. Параметр `-v` делает подробный вывод после выполнения команды. В этом примере каталоги `/pub` и `/home` с локального сервера сразу же становятся доступными для монтирования клиентскими компьютерами, которые имеют имена (`maple` и `pruce`). Каталог `/mnt/win` доступен для всех клиентских компьютеров.

Защита сервера NFS

Функция NFS была создана в то время, когда шифрование и другие меры безопасности обычно не встраивались в сетевые службы, такие как удаленный вход в систему, общий доступ к файлам и удаленное выполнение. Поэтому сервер NFS (даже до версии 3) имеет несколько недопустимых проблем с безопасностью.

Проблемы с безопасностью NFS сделали службу неподходящим средством для использования в общедоступных сетях и даже затруднили безопасное применение внутри предприятия. Перечислю некоторые из этих проблем.

- **Удаленные суперпользователи.** Даже с помощью команды `root_squash` по умолчанию, которая запрещает суперпользователям иметь корневой доступ к удаленным общим ресурсам, суперпользователь любого компьютера, которому

предоставлен общий доступ к каталогам NFS, может получить доступ к любой другой учетной записи пользователя. Поэтому, если вы обмениваетесь домашними каталогами с правами на чтение/запись, суперпользователь на любом подключенном компьютере имеет полный доступ к содержимому этих домашних каталогов.

- **Незашифрованные сообщения.** Поскольку трафик NFS не зашифрован, любой, кто лазит по вашей сети, может видеть передаваемые данные.
- **Сопоставление пользователей.** По умолчанию права для общих ресурсов NFS отображаются по ID пользователя. Так, например, пользователь с UID 1000 на клиенте NFS имеет доступ к файлам, принадлежащим UID 1000 на сервере NFS. Это происходит независимо от применяемых имен пользователей.
- **Открытая структура файловой системы.** До версии NFS 3, если вы совместно задействовали каталог через службу NFS, вы раскрывали местоположение этого каталога в файловой системе сервера. (Другими словами, если вы поделились каталогом `/var/stuff`, клиенты будут знать его точное местоположение на вашем сервере.)

Это были плохие новости. Хорошей новостью является то, что большинство этих проблем решаются в версии NFS 4, но требуют дополнительных настроек. Интегрируя поддержку Kerberos, NFS 4 позволяет настраивать доступ пользователей, предоставляя каждому из них билет Kerberos. Однако для этого нужно настроить сервер Kerberos. Что касается предоставления доступа к общему расположению NFS, то с помощью NFS 4 вы можете привязать общие каталоги к каталогу `/exports`, поэтому, когда они станут общими, точное местоположение этих каталогов не будет раскрыто.

Посетите сайт help.ubuntu.com/community/NFSv4Howtofor, чтобы подробно узнать о функциях NFS 4 в дистрибутиве Ubuntu.

Что касается стандартных функций безопасности Linux, связанных с NFS, то брандмауэры `iptables`, TCP-оболочки и система SELinux играют определенную роль в обеспечении безопасности и предоставлении доступа к серверу NFS с удаленных клиентов. В частности, могут быть полезными функции брандмауэра, работающие с NFS, однако их настройка может оказаться особенно сложной. Эти функции безопасности описаны в следующих подразделах.

Настройка брандмауэра для NFS-сервера

Служба NFS полагается на несколько разных демонов, причем большинство из них прослушивают различные порты для доступа к службе. Для версии NFS 4, по умолчанию, используемой в системе Fedora, порты TCP и UDP 2049 (`nfs`) и 111 (`rpcbind`) должны быть открыты. Сервер должен открыть также порты TCP и UDP 20048 для команды `show-mount`, чтобы иметь возможность запрашивать доступные общие каталоги NFS из команды `rpc.mountd` на сервере.

Для RHEL 8, Fedora 30 и других систем, использующих службу `firewalld`, вы можете применить окно Firewall (Межсетевой экран) (`yum install firewall-config`),

чтобы открыть брандмауэр для службы NFS. Введите `firewall-config`, затем убедитесь, что в окне установлены флажки mountd, nfs и rpcbind, которые открывают соответствующие порты службе NFS. На рис. 20.3 показан пример этого окна.

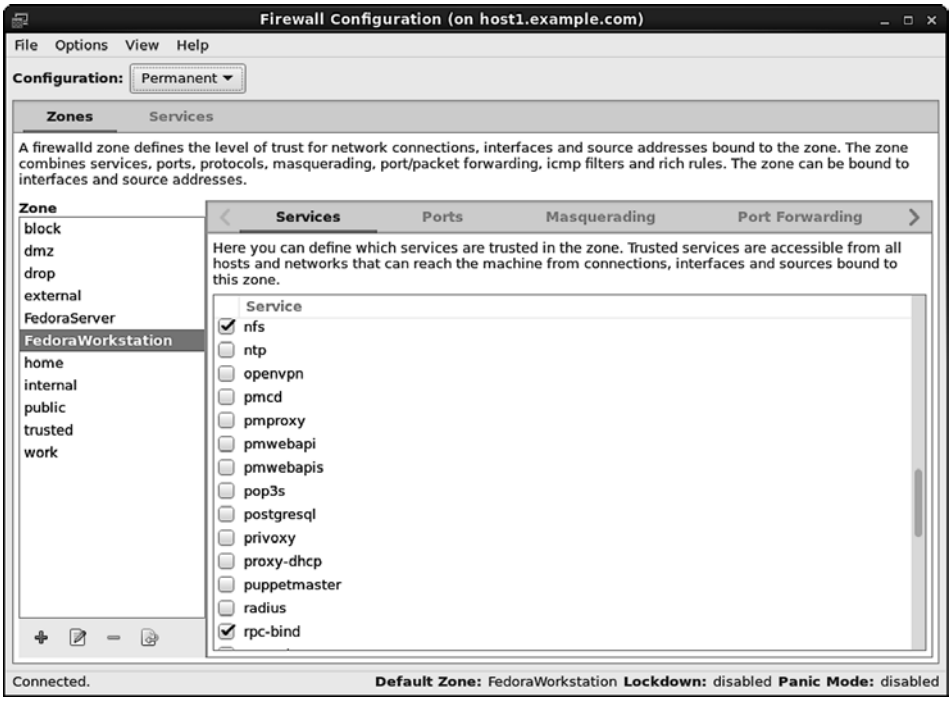


Рис. 20.3. Используйте окно Firewall, чтобы открыть брандмауэру доступ к службе

Для RHEL 6 и других систем, задействующих службу `iptables` напрямую (до появления `firewalld`), чтобы открыть порты на брандмауэре сервера NFS, убедитесь, что `iptables` включен и запущен с правилами брандмауэра, аналогичными следующим, добавленным в файл `/etc/sysconfig/iptables`:

```
-A INPUT -m state --state NEW -m tcp -p tcp --dport 111 -j ACCEPT
-A INPUT -m state --state NEW -m udp -p udp --dport 111 -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 2049 -j ACCEPT
-A INPUT -m state --state NEW -m udp -p udp --dport 2049 -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 20048 -j ACCEPT
-A INPUT -m state --state NEW -m udp -p udp --dport 20048 -j ACCEPT
```

В Red Hat Enterprise Linux 6.x и более ранних версиях настройка брандмауэра немного сложнее. Проблема с брандмауэрами заключается в том, что существует несколько связанных с NFS служб, которые прослушивают разные порты, назначаемые случайным образом. Чтобы обойти эту проблему, нужно заблокировать номера портов, использующих эти службы, и открыть брандмауэр, чтобы эти порты были доступными.

Для упрощения процесса блокировки портов сервера NFS можно добавить строку в файл `/etc/sysconfig/nfs` и назначить службам определенные номера портов. Далее приведены примеры параметров, содержащихся в файле `/etc/sysconfig/nfs`, с заданными статическими номерами портов:

```
RQUOTAD_PORT=49001
LOCKD_TCPPORT=49002
LOCKD_UDPPORT=49003
MOUNTD_PORT=49004
STATD_PORT=49005
STATD_OUTGOING_PORT=49006
RDMA_PORT=49007
```

Установив эти порты, я перезапустил службу `nfs` (`service nfs restart`). Используя команду `netstat`, в выводе вы увидите процессы, которые прослушивают назначенные порты:

```
tcp  0  0 0.0.0.0:49001  0.0.0.0:*   LISTEN   4682/rpc.rquotad
tcp  0  0 0.0.0.0:49002  0.0.0.0:*   LISTEN   -
tcp  0  0 0.0.0.0:49004  0.0.0.0:*   LISTEN   4698/rpc.mountd
tcp  0  0 :::49002       :::*        LISTEN   -
tcp  0  0 :::49004       :::*        LISTEN   4698/rpc.mountd
udp  0  0 0.0.0.0:49001  0.0.0.0:*            4682/rpc.rquotad
udp  0  0 0.0.0.0:49003  0.0.0.0:*            -
udp  0  0 0.0.0.0:49004  0.0.0.0:*            4698/rpc.mountd
udp  0  0 :::49003       :::*                 -
udp  0  0 :::49004       :::*                 4698/rpc.mountd
```

Теперь, когда эти номера портов установлены и применяются различными службами, можете добавить правила `iptables`, как было сделано с портами 2049 и 111 для настройки базовой службы NFS.

Доступ к службе NFS из TCP-оболочек

Для таких служб, как `vsftpd` и `sshd`, TCP-оболочки в Linux позволяют добавлять информацию в файлы `/etc/hosts.allow` и `/etc/hosts.deny`, чтобы указать, какие хосты могут или не могут получить доступ к службе. Сам демон сервера `nfsd` не включен для TCP-оболочек, но служба `rpcbind` включена.

Для NFS 3 и более ранних версий просто добавьте в файл `/etc/hosts.deny` следующую строку, которая будет запрещать доступ к службе `rpcbind`, а также к службе NFS:

```
rpcbind: ALL
```

Однако для серверов под управлением версии NFS 4 по умолчанию строка `rpcbind: ALL` не позволяет внешним хостам получать информацию о службах RPC (например, NFS) с помощью таких команд, как `showmount`. Но это не помешает монтировать общий каталог NFS.

Настройка системы SELinux для сервера NFS

Если система SELinux находится в разрешающем или отключенном режиме, она не блокирует доступ к службе NFS. Чтобы система работала в принудительном режиме, необходимо понимать, как работают логические типы. Чтобы проверить состояние SELinux в системе, введите следующее:

```
# getenforce
Enforcing
# grep ^SELINUX= /etc/sysconfig/selinux
SELINUX=enforcing
```

Если система находится в принудительном режиме, как в примере, откройте справочную страницу `nfs_selinux`, чтобы получить больше информации о настройках SELinux, которые могут повлиять на работу службы `vsftpd`. Вот несколько контекстов файлов SELinux, связанных с NFS, о которых необходимо знать.

- `nfs_export_all_ro`. С помощью этого логического типа SELinux позволяет обмениваться файлами с правами только на чтение на серверах NFS. Общий доступ к файлам NFS только для чтения разрешен при этом независимо от контекста файла SELinux, установленного для общих файлов и каталогов.
- `nfs_export_all_rw`. С помощью этого логического типа SELinux позволяет обмениваться файлами с правами на чтение/запись с помощью NFS. Как и предыдущий логический тип, этот работает независимо от контекста файла, установленного для общих файлов и каталогов.
- `use_nfs_home_dirs`. Чтобы NFS-сервер мог делиться вашим домашним каталогом, установите этот логический тип.

Первые два из данных логических типов включены по умолчанию. Логический тип `use_nfs_home_dirs` отключен. Чтобы включить каталог `use_nfs_home_dirs`, можно ввести следующее:

```
# setsebool -P use_nfs_home_dirs on
```

Однако вы можете игнорировать все логические типы, связанные с общим доступом к файлам NFS, изменив контексты файлов в файлах и каталогах, которыми хотите поделиться через NFS. Контексты файлов `public_content_t` и `public_content_rw_t` могут быть установлены в любом каталоге с доступом через NFS или другие протоколы общего доступа к файлам, такие как HTTP, FTP и др. Например, чтобы установить правило, позволяющее совместно использовать каталог `/whatever` и его подкаталоги для чтения/записи через NFS, а затем применить это правило, введите следующее:

```
# semanage fcontext -a -t public_content_rw_t "/whatever(/.*)?"
# restorecon -F -R -v /whatever
```

Если вы хотите, чтобы пользователи могли просто читать файлы из каталога, но не записывать в него, вместо этого назначьте каталогу контекст `public_content_t`.

Файловые системы NFS

После того как сервер экспортирует каталог по сети через NFS, клиентский компьютер подключает этот каталог к собственной файловой системе с помощью команды `mount`. Это та же команда, которая применяется для монтирования файловых систем с локальных жестких дисков, DVD и USB-накопителей, но с немного другими параметрами.

Команда `mount` позволяет клиенту автоматически монтировать каталоги NFS, добавленные в файл `/etc/fstab`, точно так же, как это происходит с локальными дисками. Каталоги NFS можно добавить в файл `/etc/fstab` таким образом, чтобы они не монтировались автоматически (поэтому вы можете монтировать их вручную). С параметром `noauto` каталог NFS, указанный в файле `/etc/fstab`, останется неактивным до тех пор, пока команда `mount` не будет использована после запуска системы для монтирования файловой системы.

Кроме работы с файлом `/etc/fstab` вы можете установить параметры монтирования с помощью файла `/etc/nfs-mount.conf`. В нем можно задать параметры монтирования, применимые к любому каталогу NFS, который вы монтируете, или только к тем, которые связаны с определенными точками монтирования или серверами NFS.

Однако, прежде чем приступить к монтированию общих каталогов NFS, вы, вероятно, захотите проверить, какие общие каталоги доступны через NFS, с помощью команды `showmount`.

Просмотр общих ресурсов NFS

В клиентской системе Linux можно использовать команду `showmount`, чтобы увидеть, какие общие каталоги доступны с выбранного компьютера, как в примере:

```
$ showmount -e server.example.com
/export/myshare client.example.com
/mnt/public
```

Вывод команды `showmount` показывает, что общий каталог с именем `/export/myshare` доступен только хосту `client.example.com`. Однако общий каталог `/mnt/public` доступен всем.

Монтирование файловой системы NFS вручную

После того как каталог с компьютера в вашей сети был экспортирован (то есть стал доступен для монтирования), можете монтировать его вручную с помощью команды `mount`. Это хороший способ убедиться в том, что каталог доступен и работает, прежде чем вы настроите его на постоянное монтирование. Далее приведен

пример монтирования каталога /stuff с компьютера с именем maple на локальном компьютере:

```
# mkdir /mnt/maple
# mount maple:/stuff /mnt/maple
```

Первая команда (mkdir) создает каталог точки монтирования. (Каталог /mnt — это общее место размещения временно смонтированных дисков и файловых систем NFS.) Команда mount идентифицирует удаленный компьютер и общую файловую систему, разделенную двоеточием (maple:/stuff), а далее указан каталог локальной точки монтирования (/mnt/maple).

ПРИМЕЧАНИЕ

Если монтирование завершится неудачно, убедитесь, что служба NFS запущена на сервере и правила брандмауэра сервера не запрещают доступ к ней. На сервере введите команду ps ax | grep nfsd, чтобы просмотреть список процессов сервера nfsd. Если вы не видите списка, запустите демоны NFS, как описано ранее в этой главе. Чтобы просмотреть правила брандмауэра, введите iptables -vnL. По умолчанию демон nfsd прослушивает запросы NFS на порте 2049. Ваш брандмауэр должен принимать UDP-запросы на портах 2049 (NFS) и 111 (RPC). В Red Hat Enterprise Linux 6 и более ранних версиях Fedora может потребоваться установить статические порты для связанных служб, а затем открыть порты для них в брандмауэре. См. раздел «Защита сервера NFS» ранее в данной главе, чтобы узнать, как преодолеть эти проблемы безопасности.

Чтобы убедиться, что монтирование NFS завершено, введите команду mount -t nfs4. Она перечисляет все смонтированные файловые системы NFS. Пример вывода команды mount (файловые системы, не относящиеся к этому обсуждению, удалены):

```
# mount -t nfs4
192.168.122.240:/mnt on /mnt/fed type nfs4
(rw,relatime,vers=4.2,rsize=262144,wsize=262144,namlen=255,hard,
proto=tcp,timeo=600,retrans=2,sec=sys,clientaddr=192.168.122.63,
local_lock=none,addr=192.168.122.240)
```

Выходные данные команды mount -t nfs4 отображают только файловые системы, смонтированные с файловых серверов NFS. Файловая система NFS — это каталог /mnt с IP-адреса 192.168.122.240 (192.168.122.240:/mnt). Он монтируется в каталоге /mnt/fed, и его тип монтирования — это nfs4. Файловая система была смонтирована для чтения/записи (rw), а IP-адрес maple — это 92.168.122.240 (addr=192.168.122.240). Приведены и другие параметры, связанные с монтированием, такие как размеры пакетов для чтения и записи и номер версии NFS.

Показанные ранее действия монтирования реализуют временное монтирование файловой системы NFS в локальной системе. В следующем разделе описывается, как сделать монтирование постоянным (с помощью файла /etc/fstab) и выбрать различные варианты монтирования NFS.

Монтирование файловой системы NFS во время загрузки

Чтобы настроить файловую систему NFS на автоматическое монтирование в заданной точке при каждом запуске системы Linux, необходимо добавить соответствующую строку для этой файловой системы в `/etc/fstab`. Этот файл содержит информацию обо всех типах монтируемых (и доступных для монтирования) файловых систем вашей системы.

Формат добавления файловой системы NFS в локальную систему:

```
host:directory   mountpoint   nfs   options   0   0
```

Первый элемент (*host:directory*) идентифицирует компьютер сервера NFS и общий каталог, *mountpoint* — это локальная точка монтирования, в которую монтируется каталог NFS. После них указывается тип файловой системы (`nfs`). Все параметры, связанные с монтированием, отображаются рядом в списке, разделенном запятыми. (Последние два нуля настраивают систему на то, чтобы не сбрасывать содержимое файловой системы и не запускать в ней команду `fsck`.)

Далее приведены примеры строк NFS в файле `/etc/fstab`:

```
maple:/stuff   /mnt/maple nfs   bg,rsize=8192,wsize=8192   0 0
oak:/apps      /oak/apps  nfs   noauto,ro                 0 0
```

В первом примере удаленный каталог `/stuff` компьютера `maple` (`maple:/stuff`) монтируется в локальный каталог `/mnt/maple` (он уже должен существовать). Если монтирование завершается неудачно из-за недоступности общего ресурса, команда `bg` заставляет процесс монтирования перейти в фоновый режим и повторить попытку позже.

Тип файловой системы — `nfs`, а размеры буферов чтения (`rsize`) и записи (`wsize`) (описаны в пункте «Параметры монтирования» далее в этом подразделе) устанавливаются равными `8192` скорости передачи данных, связанной с этим соединением. Во втором примере удаленным каталогом является `/apps` на компьютере с именем `oak`. Он настроен как файловая система NFS (`nfs`), которая может быть смонтирована в каталоге `/oak/apps` локально. Однако эта файловая система не монтируется автоматически (`noauto`) и может быть смонтирована только для чтения (`ro`) с помощью команды `mount` после запуска системы.

СОВЕТ

По умолчанию файловая система NFS монтируется на чтение/запись. Однако она экспортируется по умолчанию с доступом только для чтения. Если вы не можете выполнить запись в файловую систему NFS, убедитесь, что она была экспортирована с сервера с доступом на чтение/запись.

Монтирование файловых систем noauto

Файл `/etc/fstab` также может содержать устройства для других файловых систем, которые не монтируются автоматически. Например, это могут быть несколько разделов на жестком диске или общая файловая система NFS, которую вы хотите

монтировать периодически. Файловая система noauto может быть смонтирована вручную. Преимущество этого способа заключается в том, что при вводе команды mount можно ввести меньше информации, а остальное заполнить содержимым файла /etc/fstab. Например, можете ввести:

```
# mount /oak/apps
```

Команда mount знает, что нужно проверить файл /etc/fstab, чтобы получить файловую систему для монтирования (oak:/apps), тип файловой системы (nfs) и параметры для использования с монтированием (в данном случае ro — только для чтения). Вместо того чтобы вводить локальную точку монтирования (/oak/apps), можно ввести имя удаленной файловой системы (oak:/apps) и указать другую информацию.

СОВЕТ

Если, давая имя точке монтирования, вы включите в него имя удаленного сервера NFS, что поможет запомнить, где на самом деле хранятся файлы. Это невозможно, если вы совместно используете домашние (/home) или почтовые каталоги (/var/spool/mail). Например, вы можете смонтировать файловую систему с компьютера duck в каталоге /mnt/duck.

Параметры монтирования

Чтобы повлиять на способ монтирования файловой системы, вы можете добавить несколько параметров монтирования в файл /etc/fstab (или в саму командную строку монтирования). Добавляя параметры в файл /etc/fstab, разделяйте их запятыми. Например, при монтировании каталога oak:/apps используются параметры noauto, ro и hard:

```
oak:/apps    /oak/apps    nfs    noauto,ro,hard    0 0
```

Далее приведены параметры, полезные для монтирования файловых систем NFS. Вы можете прочитать об этих и других параметрах монтирования NFS, которые можно поместить в файл /etc/fstab, на справочной странице nfs (man 5 nfs).

- hard. Если применяется этот параметр и сервер NFS выключается или отключается, пока процесс ожидает доступа к нему, процесс зависает до тех пор, пока сервер не заработает снова. Полезная функция, если очень важно, чтобы данные, с которыми вы работаете, оставались синхронизированными с программами, использующими их. (Это поведение по умолчанию.)
- soft. Если сервер NFS отключается или выходит из строя, процесс, пытающийся получить доступ к данным с него, заканчивается через заданный период времени, когда этот параметр включен. Ошибка ввода-вывода передается процессу, который пытается получить доступ к серверу NFS.
- rsize. Это размер блоков данных (в байтах), которые клиент NFS запрашивает при чтении данных с сервера NFS. Значение по умолчанию — 1024. Использование большего значения (например, 8192) обеспечивает лучшую

производительность в быстрой (например, локальной) и относительно безошибочной (с небольшим количеством шумов или ошибок) сети.

- `wsize`. Это размер блоков данных (в байтах), которые клиент NFS запрашивает при записи данных на сервер NFS. Значение по умолчанию — `1024`. Проблемы с производительностью те же, что при использовании параметра `rsize`.
- `timeo=#`. Устанавливает время ожидания перед повторной попыткой передачи, представляет собой число в десятых долях секунды, Значение по умолчанию — 0,7 секунды. Каждый последующий тайм-аут удваивает его (максимум до 60 секунд). Увеличьте это значение, если считаете, что тайм-ауты происходят из-за медленного ответа сервера или малой скорости сети.
- `retrans=#`. Устанавливает количество незначительных тайм-аутов и повторных передач, которые должны произойти до длительного тайм-аута.
- `retry=#`. Устанавливает, в течение скольких минут нужно продолжать повторять неудачные запросы монтирования, где символ # заменяется количеством минут для повторной попытки. Значение по умолчанию — 10 000 минут (что составляет около одной недели).
- `bg`. Если время первой попытки монтирования истекло, выполните все последующие в фоновом режиме. Этот параметр очень полезен, если вы монтируете медленную или не всегда доступную файловую систему NFS. При размещении запросов на монтирование в фоновом режиме ваша система может продолжать монтировать другие файловые системы, вместо того чтобы ждать завершения текущей.

ПРИМЕЧАНИЕ

Если вложенная точка монтирования отсутствует, происходит тайм-аут для того, чтобы ее добавить. Например, если смонтировать /usr/trip и /usr/trip/extra в файловой системе NFS, когда точка /usr/trip еще не установила попытку монтирования /usr/trip/extra, то /usr/trip/extra перейдет в тайм-аут. Если повезет, то /usr/trip возобновит работу и /usr/trip/extra смонтируется при следующей попытке.

- `fg`. Если время первой попытки монтирования истекло, выполните последующие не в фоновом режиме. Это поведение по умолчанию. Используйте данный параметр, если необходимо, чтобы монтирование прошло успешно, прежде чем продолжить (например, если вы монтировали `/usr`).

Не все параметры монтирования NFS нужно добавлять в файл `/etc/fstab`. На стороне клиента файл `/etc/nfsmount.conf` может быть настроен в разделах `Mount`, `Server` и `Global`. В разделе `Mount` можно указать, какие параметры монтирования используются при монтировании файловой системы NFS в определенную точку. Раздел `Server` позволяет добавлять параметры в любую файловую систему NFS, смонтированную с определенного сервера NFS. Параметры из раздела `Global` применяются к каждому монтированию NFS с этого клиента.

Следующая строка в файле `/etc/nfsmount.conf` устанавливает размер блока чтения и записи 32 Кбайт для всех каталогов NFS, смонтированных из системы `thunder.example.com`:

```
[ Server "thunder.example.com" ]
  rsize=32k
  wsize=32k
```

Чтобы установить параметры по умолчанию для каждого монтирования NFS в ваших системах, раскомментируйте блок `NFSMount_Global_Options`. В нем можно установить протоколы и версии NFS, а также скорость передачи и настройки повторных попыток. Пример блока `NFSMount_Global_Options`:

```
[ NFSMount_Global_Options ]
# This sets the default version to NFS 4
Defaultvers=4
# Sets the number of times a request will be retried before
# generating a timeout
Retrans=2
# Sets the number of minutes before retrying a failed
# mount to 2 minutes
Retry=2
```

Здесь версия NFS по умолчанию равна `4`. Данные передаются дважды (`2`) перед генерацией тайм-аута. Время ожидания перед повторной попыткой передачи составляет 2 минуты. Вы можете переопределить любое из этих значений по умолчанию, добавив параметры монтирования в файл `/etc/fstab` или в командную строку монтирования при монтировании каталога NFS.

Функция autofs для монтирования файловых систем NFS по требованию

Совершенствование области автоматического обнаружения и монтирования съемных устройств означает, что вы можете просто вставить или подключить эти устройства и система сама их обнаружит, смонтирует и отобразит. Однако, чтобы автоматизировать процесс обнаружения и монтирования удаленных файловых систем NFS, все равно нужно использовать функцию `autofs` (сокращение от *automatically mounted filesystems* — «автоматически монтируемые файловые системы»).

Функция `autofs` монтирует сетевые файловые системы по требованию, когда кто-то пытается их применить. Если функция `autofs` настроена и включена, вы можете вызвать монтирование любых доступных общих каталогов NFS по требованию. Для работы с функцией `autofs` необходимо установить пакет `autofs`. (Чтобы установить пакет из сети, для систем Fedora и RHEL можете ввести команду `yum install autofs`, а для Ubuntu или Debian — `apt-get install autofs`.)

Автоматическое подключение к каталогу /net

При включенной функции `autofs`, если вы знаете имя хоста и каталог, совместно используемый другим хост-компьютером, просто измените (`cd`) его на каталог монтирования `autofs` (`/net` или `/var/autofs` по умолчанию). Это приведет к тому, что общий ресурс автоматически смонтируется и станет доступным.

Включение функции `autofs` в системах Fedora или RHEL производится так.

1. В системе Fedora или RHEL из окна Terminal (Терминал) откройте как суперпользователь файл `/etc/auto.master` и найдите следующую строку:

   ```
   /net   -hosts
   ```

   Это приведет к тому, что каталог `/net` будет действовать как точка монтирования для общих каталогов NFS, к которым вы хотите получить доступ в сети. (Если в начале этой строки есть символ комментария, удалите его.)

2. Чтобы запустить службу `autofs` в системе Fedora 30, RHEL 7 или более поздней версии, введите от имени суперпользователя следующее:

   ```
   # systemctl start autofs.service
   ```

3. В системе Fedora 30, RHEL 7 или более поздней версии настройте службу `autofs` для перезапуска при каждой загрузке системы:

   ```
   # systemctl enable autofs
   ```

Хотите верьте, хотите нет, но это все, что нужно сделать. Если у вас есть сетевое подключение к серверам NFS, с которых вы хотите поделиться каталогами, получите доступ к общему каталогу NFS. Например, если знаете, что каталог `/usr/local/share` используется совместно с компьютера с именем `huttle` из вашей сети, введите следующую команду:

```
$ cd /net/shuttle/
```

Если на нем есть какие-то доступные вам общие каталоги, можете успешно перейти в этот каталог. Можно также ввести следующее:

```
$ ls
usr
```

Теперь видно, что каталог является частью пути к общему каталогу. Если бы существовали общие каталоги из других каталогов верхнего уровня (например, `/var` или `/tmp`), вы бы их увидели. Конечно, доступность любого из этих каталогов зависит от того, как настроена система безопасности на сервере.

Перейдите прямо в общий каталог, как показано в этом примере:

```
$ cd /net/shuttle/usr/local/share
$ ls
info man music television
```

На этом этапе команда `ls` должна отобразить содержимое каталога `/usr/local/share` на компьютере с именем `shuttle`. Варианты применения этого содержимого

зависят от того, как был настроен доступ к нему для совместного использования сервером.

То, что вы не увидите никаких файлов и каталогов, пока не начнете работать с ними, допустим не перейдете в сетевой каталог, может немного сбить с толку. Например, команда `ls` ничего не отображает в сетевом каталоге до тех пор, пока каталог не будет смонтирован, то есть иногда он виден, а иногда его нет. Просто перейдите в сетевой каталог или получите доступ к файлу в таком каталоге, а функция `autofs` позаботится обо всем остальном.

В приведенном примере `shuttle` — это имя хоста. Однако вы можете использовать любые имя или IP-адрес, который определяет местоположение компьютера сервера NFS. Так, вместо `shuttle` можете взять `shuttle.example.com` или IP-адрес, например 192.168.0.122.

Автоматическое монтирование домашних каталогов

Вместо того чтобы просто монтировать файловую систему NFS в каталоге `/net`, можете настроить службу `autofs` для монтирования определенного каталога NFS в определенном месте. Например, настроить домашний каталог пользователя с централизованного сервера, который может автоматически монтироваться с другого компьютера при входе пользователя в систему. Аналогично можете применить централизованный механизм аутентификации, такой как LDAP (как описано в главе 11 «Управление учетными записями»), чтобы предложить централизованные учетные записи пользователей.

Следующий вывод иллюстрирует, как настроить учетную запись пользователя на сервере NFS и предоставить общий доступ к домашнему каталогу пользователя с именем `joe` с этого сервера, чтобы его можно было автоматически подключить при входе `joe` на другой компьютер. В этом примере вместо применения центрального сервера аутентификации в каждой системе создаются соответствующие учетные записи.

1. На сервере NFS (mynfs.example.com), который предоставляет централизованный домашний каталог пользователя для `joe`, создайте учетную запись пользователя `joe` с домашним каталогом `/home/shared/joe` в качестве имени. Найдите также ID пользователя `joe` из файла `/etc/passwd` (третье поле), чтобы сопоставить его при настройке учетной записи пользователя для `joe` в другой системе.

```
# mkdir /home/shared
# useradd -c "Joe Smith" -d /home/shared/joe joe
# grep joe /etc/passwd
joe:x:1000:1000:Joe Smith:/home/shared/joe:/bin/bash
```

2. На сервере NFS экспортируйте каталог `/home/shared/` в любую систему своей локальной сети (в примере `192.168.0.*`), чтобы можно было поделиться

домашним каталогом для Joe и любых других пользователей, которых вы создадите, добавив эту строку в файл /etc/exports:

```
# /etc/exports file to share directories under /home/shared
# only to other systems on the 192.168.0.0/24 network:
/home/shared 192.168.0.*(rw,insecure)
```

ПРИМЕЧАНИЕ

В приведенном ранее примере файла экспорта параметр insecure позволяет клиентам использовать порты выше порта 1024 для выполнения запросов на монтирование. Некоторые клиенты NFS требуют этого, потому что у них нет доступа к занятым портам NFS.

3. На сервере NFS перезапустите службу nfs-server, а если она уже запущена, просто экспортируйте общий каталог следующим образом:

```
# exportfs -a -r -v
```

4. На NFS-сервере убедитесь, что соответствующие порты должны быть открыты в брандмауэре. Дополнительные сведения см. в разделе «Защита сервера NFS» ранее в этой главе.
5. В клиентской системе NFS добавьте в файл /etc/auto.master запись, определяющую точку монтирования, куда вы хотите монтировать удаленный каталог NFS, и файл (по своему выбору), в котором будете определять местоположение удаленного каталога NFS. Я добавил эту запись в файл auto.master:

```
/home/remote /etc/auto.joe
```

6. В клиентской системе NFS добавьте в только что отмеченный файл (я использовал /etc/auto.joe) запись, которая содержит следующую строку:

```
joe    -rw    mynfs.example.com:/home/shared/joe
```

7. В клиентской системе NFS перезапустите службу autofs:

```
# systemctl restart autofs.service
```

8. В клиентской системе NFS создайте пользователя с именем joe с помощью команды useradd. Для нее вам нужно получить UID для joe на сервере (507 в данном примере), чтобы клиентская система joe владела файлами из домашнего каталога joe NFS. При выполнении следующей команды создается учетная запись пользователя joe, но вы увидите сообщение об ошибке, указывающее, что домашний каталог уже существует (так и должно быть):

```
# useradd -u 507 -c "Joe Smith" -d /home/remote/joe joe
# passwd joe
Changing password for user joe.
New password: ********
Retype new password: ********
```

9. Войдите в клиентскую систему NFS под именем joe. Если все работает правильно, то, когда joe входит в систему и пытается получить доступ к своему

домашнему каталогу (`/home/remote/joe`), каталог `/home/share/joes` должен быть смонтирован из сервера `mynfs.example.com`. Каталог NFS был как общим, так и смонтированным на чтение/запись с владением UID `507` (`joe` в обеих системах), поэтому пользователь `joe` в локальной системе должен иметь возможность добавлять, удалять, изменять и просматривать файлы в нем.

После того как пользователь `joe` выходит из системы (фактически когда он прекращает доступ к каталогу) на время тайм-аута (по умолчанию 10 минут), каталог размонтируется.

Размонтирование файловых систем NFS

Смонтированную файловую систему NFS размонтировать очень просто. Команда монтирования используется либо с локальной точкой монтирования, либо с именем удаленной файловой системы. Вот два способа размонтировать `maple:/stuff` из локального каталога `/mnt/maple`:

```
# umount maple:/stuff
# umount /mnt/maple
```

Любой из них сработает. Если `maple:/stuff` монтируется автоматически (из списка в файле `/etc/fstab`), то каталог монтируется заново при следующей загрузке Linux. Если это было временное монтирование (или указано как `noauto` в файле `/etc/fstab`), то оно не монтируется повторно во время загрузки.

СОВЕТ

Используйте команду umount, а не unmount.

Если при попытке размонтировать файловую систему вы получаете сообщение, что устройство занято (`device is busy`), это означает, что размонтировать файловую систему не удалось, поскольку в данный момент она используется. Скорее всего, один из каталогов в файловой системе NFS является текущим каталогом вашей оболочки (или оболочки кого-то еще в вашей системе). Другим вариантом может быть то, что команда удерживает файл открытым в файловой системе NFS (например, в текстовом редакторе). Проверьте свои окна Terminal (Терминал) и другие оболочки, а затем введите команду `cd` из каталога или просто закройте окна Terminal (Терминал).

Если файловая система NFS не размонтируется, можно заставить ее сделать это сейчас (`umount -f /mnt/maple`) или размонтировать и очистить позже (`umount -l /mnt/maple`). Лучше всего применять параметр `-l`, поскольку принудительное размонтирование способно нарушить процесс изменения файла. Другой вариант — запустить команду `fuser-v mountpoint`, чтобы увидеть, какие пользователи работают со смонтированным общим ресурсом NFS, а затем командой `fuser-k mountpoint` моментально завершить все эти процессы.

Резюме

Сетевая файловая система (NFS) — один из старейших компьютерных файлообменных серверов. Она по-прежнему наиболее популярна для обмена каталогами файлов между системами UNIX и Linux. NFS позволяет серверам назначать определенные каталоги, предоставляя доступ к ним назначенным хостам, а клиентским системам — подключаться к этим каталогам, локально их монтируя.

Сервер NFS можно защитить с помощью правил брандмауэра (`iptables`), оболочек TCP (разрешить и запретить доступ к хосту) и SELinux (ограничить то, как протоколы общего доступа к файлам могут совместно задействовать ресурсы NFS). Хотя служба NFS изначально была небезопасна (данные передаются в незашифрованном виде, а доступ пользователям открыт), функции NFS 4-й версии повысили ее общую безопасность.

Эта глава — последняя из посвященных серверам глав этой книги. Глава 21 «Диагностика Linux» охватывает широкий спектр тем, относящихся к настольным компьютерам и серверам, и помогает понять процессы диагностики системы Linux и устранения неполадок в ней.

Упражнения

Упражнения этой главы относятся к задачам, связанным с настройкой и использованием сервера NFS в Linux. Для их решения желательно иметь две системы Linux, подключенные к локальной сети. Одна из них будет действовать как сервер NFS, а другая — как клиент NFS.

Чтобы получить максимальную пользу от выполнения этих упражнений, рекомендую не применять сервер Linux, на котором уже работает служба NFS. В таком случае вы не сможете выполнить все упражнения, не нарушив работу службы NFS, которая уже активна и совместно использует ресурсы.

В приложении Б представлены варианты решения упражнений.

1. В системе Linux, которая действует в качестве сервера NFS, установите пакеты, необходимые для настройки службы NFS.
2. На сервере NFS перечислите входящие в пакет файлы документации, которые предоставляет программное обеспечение для сервера NFS.
3. На сервере NFS определите имя службы NFS и запустите ее.
4. На сервере NFS проверьте состояние только что запущенной службы NFS.
5. На сервере NFS создайте каталог `/var/mystuff` и поделитесь им со своего сервера NFS, применяя следующие атрибуты: «доступно всем», «доступно только для чтения» и «суперпользователь на клиенте имеет корневой доступ к общему ресурсу».
6. На сервере NFS убедитесь, что созданный вами общий ресурс доступен всем хостам, открыв TCP-оболочки, `iptables` и SELinux.

7. Во второй системе Linux (клиент NFS) просмотрите общие ресурсы, доступные с сервера NFS. (Если у вас нет второй системы, сделайте это из той же системы.) Если не видите общий каталог NFS, вернитесь к предыдущему упражнению и повторите попытку.
8. На клиенте NFS создайте каталог `/var/remote` и временно смонтируйте каталог `/var/mystuff` с сервера NFS в этой точке монтирования.
9. На клиенте NFS размонтируйте каталог `/var/remote`, добавьте строку, чтобы то же самое монтирование выполнялось автоматически при перезагрузке (с параметром `bgmount`), и проверьте, что созданная вами строка работает правильно.
10. С сервера NFS скопируйте некоторые файлы в каталог `/var/mystuff`. Из клиента NFS убедитесь, что видите файлы, только что добавленные в этот каталог, и что не можете записывать файлы в этот каталог из клиента.

21 Диагностика Linux

В этой главе

- Диагностика загрузчика системы.
- Диагностика инициализации системы.
- Решение проблем с пакетами программного обеспечения.
- Проверка сетевого интерфейса.
- Диагностика проблем с памятью.
- Режим восстановления.

В любой сложной операционной системе многое может пойти не так. Вы не сможете сохранить файл, потому что не хватает места на диске. Приложение выйдет из строя из-за нехватки памяти в системе. Система не загрузится должным образом по множеству причин.

В Linux, открытой и сосредоточенной на максимальной эффективности работы программного обеспечения, есть поразительное количество инструментов для устранения всех мыслимых проблем. В конце концов, если операционная система работает не так, как вам хотелось бы, можете переписать код самостоятельно (хотя в книге я не рассказываю, как это сделать).

В этой главе рассматриваются некоторые наиболее распространенные проблемы, с которыми вы можете столкнуться в системе Linux, и описываются инструменты и действия, с помощью которых эти проблемы решаются. Темы разбиты по областям диагностики, таким как процесс загрузки, пакеты программного обеспечения, сеть, проблемы с памятью и режим восстановления.

Диагностика загрузки системы

Прежде чем приступать к устранению неполадок системы Linux, ее нужно загрузить. Загрузка системы Linux, установленной непосредственно на компьютере с архитектурой ПК, состоит из таких этапов, как:

- включение питания;
- загрузка жесткого диска (из BIOS или UEFI);
- поиск загрузчика системы и его запуск;
- выбор загрузчика операционной системы;
- запуск ядра и начального диска оперативной памяти для выбранной операционной системы;
- запуск процесса инициализации (`init` или `systemd`);
- запуск всех служб, связанных с выбранным уровнем активности (уровень выполнения или цель по умолчанию).

Конкретные действия в каждой из этих точек в последние годы претерпели трансформацию. Загрузчики меняются, чтобы можно было приспособить новые виды оборудования. Процесс инициализации тоже меняется так, что службы могут запускаться более эффективно, основываясь на зависимостях и реагируя на состояние системы (например, какое оборудование подключено или какие файлы существуют), а не на статический порядок загрузки.

Диагностика загрузки Linux начинается после включения компьютера и заканчивается, когда все службы запущены. В этот момент в консоли обычно появляется графическое или текстовое приглашение войти в систему.

После прочтения кратких описаний методов запуска перейдите к разделу «Запуск загрузчика (BIOS или UEFI)», чтобы понять, что происходит на каждом этапе загрузки и где может потребоваться диагностика. Поскольку общая структура процесса загрузки Linux одинакова для всех трех представленных здесь систем (Fedora, RHEL и Ubuntu), я пройду через него только один раз, а различия между ними опишу по ходу действий.

Методы запуска системы

Запуск служб, связанных с работающей системой Linux, зависит от конкретного дистрибутива Linux. После того как загрузчик запускает ядро, все остальные действия (монтирование файловых систем, настройка параметров ядра, запуск служб и т. д.) выполняются в процессе инициализации.

Описывая загрузку, я сосредоточусь на двух различных типах инициализации: System V `init` и `systemd`.

Скрипты инициализации System V init

Средство инициализации System V состоит из процесса инициализации (первого процесса, запускаемого после самого ядра), файла `/etc/inittab`, который направляет все действия запуска, и набора скриптов оболочки, запускающих отдельные службы. Первые версии Fedora вплоть до RHEL 5 использовали процесс инициализации System V `init`.

Система System V `init` была разработана для UNIX System V компанией AT&T в середине 1980-х годов, когда системы UNIX впервые включили запуск сетевых

интерфейсов и подключенных к ним сервисов. Только в последнее десятилетие она была вытеснена системами Upstart и `systemd`, которые лучше соответствуют требованиям современных операционных систем.

В System V `init` наборы служб назначаются так называемым *уровням выполнения*. Например, многопользовательский уровень выполнения может запускать базовые системные службы, сетевые интерфейсы и сетевые службы. Однопользовательский режим просто запускает базовую систему Linux, чтобы кто-то мог войти в систему из системной консоли, не запуская сетевые интерфейсы или службы.

После запуска System V `init` вы можете использовать команды `reboot`, `shutdown` и `init` для изменения уровней запуска. Можно также применять `service` и `chkconfig`, чтобы запускать/останавливать отдельные службы или включать/отключать службы соответственно.

Скрипты инициализации System V настроены на выполнение в определенном порядке, причем каждый из них должен быть завершен до запуска следующего. Если служба выходит из строя, то нет никаких условий для ее автоматического перезапуска. Система `systemd` была разработана для устранения этих и других недостатков System V `init`.

Запуск с помощью системы systemd

Система `systemd` быстро становится самым популярным и подходящим способом инициализации для многих систем Linux. Она была принята в системах Fedora 15 и RHEL 7 и заменила Upstart в Debian и Ubuntu 15.04. Хотя `systemd` сложнее, чем System V `init`, но она предоставляет гораздо больше функций.

- **Цели.** Вместо уровней выполнения `systemd` фокусируется на целях. *Цель* может запускать набор служб, создавать или запускать другие типы устройств (например, монтирование каталогов, сокеты, области подкачки и таймеры).
- **Совместимость System V.** Существуют цели, совпадающие с уровнями запуска System V, для тех, кто привык с ними работать. Например, цель `graphical.target` совпадает с уровнем запуска 5, а цель `multi-user.target`, по существу, является уровнем запуска 3. Однако целей гораздо больше, чем уровней выполнения, что дает вам возможность более точно управлять наборами единиц. Кроме того, `systemd` поддерживает скрипты и команды инициализации System V, такие как `chkconfig` и `service`, для управления этими службами, если они установлены.
- **Запуск на основе зависимостей.** При запуске системы может запуститься любая служба в целевом объекте по умолчанию (`graphical.target` для настольных компьютеров и `multi-user.target` для большинства серверов), которая имеет соответствующие зависимости. Эта функция может ускорить процесс загрузки, гарантируя, что одна несработавшая служба не остановит запуск других служб, если она не нужна им для запуска.

- **Использование ресурсов.** С помощью `systemd` вы можете применять команду `cgroups` для ограничения объема ресурсов системы, потребляемых службой. Например, если вы ограничите объем памяти, процессора или других ресурсов, которые может потреблять вся служба, то запущенный процесс или служба, запускающая необоснованное количество дочерних процессов, не сможет потреблять больше ресурсов, чем ей разрешено.

При запуске Linux-системы с поддержкой `systemd` первым запущенным процессом (PID 1) является демон `systemd` (а не демон `init`). Основная команда для управления службами `systemd` — это команда `systemctl`. Управление сообщениями журнала `systemd` осуществляется с помощью команды `journalctl`. Вы также можете использовать старые команды System V `init`, такие как `init`, `poweroff`, `reboot`, `runlevel` и `shutdown` для управления службами.

Запуск загрузчика (BIOS или UEFI)

Когда вы физически включаете компьютер, загружается прошивка для инициализации оборудования и поиска операционной системы для загрузки. На ПК эта прошивка традиционно называется *BIOS (Basic Input Output System)*. В последние годы в качестве замены BIOS на некоторых компьютерах стал доступен новый тип прошивки под названием *UEFI (Unified Extensible Firmware Interface)*. Они взаимно исключают друг друга.

UEFI была разработана для обеспечения безопасной загрузки, чтобы в ходе нее могли использоваться только операционные системы с официальными компонентами. И все же UEFI можно задействовать с неподписанными операционными системами, отключив функцию безопасной загрузки.

Для дистрибутива Ubuntu поддержка безопасной загрузки впервые была реализована в версии 12.04.2. Дистрибутив RHEL 7 и его более поздние версии также официально поддерживают функцию безопасной загрузки. Основная задача прошивок BIOS и UEFI состоит в том, чтобы инициализировать аппаратное обеспечение, а затем передать управление процессом загрузки загрузчику. Затем *загрузчик* находит и запускает операционную систему. После установки операционной системы, как правило, нужно просто позволить прошивке делать свою работу и не прерывать ее.

Однако бывают случаи, когда нужно прервать работу прошивки. Для обсуждения этого мы разберем обычный ход работы BIOS. Сразу после включения питания вы увидите экран BIOS, который обычно включает в себя информацию о том, как перейти в режим настройки и изменить порядок загрузки. Вот что вы сможете сделать, если нажмете нужную функциональную клавишу (часто `F1`, `F2` или `F12`), чтобы выбрать один из следующих пунктов.

- **Программа установки (setup utility).** Программа установки позволяет изменять настройки в BIOS. Эти настройки можно использовать для включения

или отключения определенных компонентов либо включения или выключения выбранных функций.

- **Порядок загрузки (boot order).** Компьютеры способны запускать операционную систему или, точнее, загрузчик, который запускает операционную систему, с различных устройств, подключенных к компьютеру. Это могут быть CD, DVD, жесткий диск, USB-драйвер или карта сетевого интерфейса. Порядок загрузки определяет порядок проверки этих устройств. Изменив порядок загрузки, вы можете указать компьютеру временно игнорировать порядок загрузки по умолчанию и попытаться загрузиться с выбранного устройства.

На своей рабочей станции Dell после загрузки экрана BIOS я сразу же нажимаю функциональную клавишу F2, чтобы перейти в настройки, или F12, чтобы временно изменить порядок загрузки. В следующих разделах описано, какие проблемы можно устранить с помощью экранов Setup (Настройки) и Boot Order (Порядок загрузки).

Диагностика настроек BIOS

Как я отмечал ранее, обычно вы даете BIOS запускаться без перерыва до загрузчика по умолчанию (например, жесткого диска). Однако перечислю случаи, когда может понадобиться перейти в режим настройки и изменить BIOS.

- **Чтобы увидеть все ваше оборудование.** Если проблема диагностики связана с аппаратным обеспечением, лучше всего начать изучение системы с настройки BIOS. Экран Setup (Настройки) сообщает о типе системы, версии BIOS, процессорах, слотах памяти и типах памяти (32- или 64-разрядные), о том, какие устройства находятся в каждом слоте, а также множество подробностей о типах устройств, подключенных к системе.

  Если вообще не получается загрузить операционную систему, настройки BIOS могут стать единственным способом определить модель системы, тип процессора и получить другие сведения, необходимые для поиска нужной информации или вызова службы поддержки.

- **Чтобы отключить/подключить устройство.** Большинство устройств, присоединенных к компьютеру, включены и доступны для операционной системы. Чтобы устранить неполадки, может потребоваться отключить устройство.

  Предположим, компьютер имеет две карты сетевого интерфейса (NIC). Вы хотите использовать вторую карту NIC для установки Linux по сети, но инсталлятор продолжает пытаться задействовать первую карту NIC для подключения к сети. Вы можете отключить первую карту, чтобы инсталлятор даже не видел ее при попытке подключиться к сети. Или оставить первую карту видимой для компьютера, но отключить ее способность к PXE-загрузке.

  Возможно, у вас есть аудиокарта и вы хотите отключить встроенное аудио на материнской плате. Это можно сделать и в BIOS. И наоборот, иногда нужно включить устройство, которое было отключено. Возможно, вам предоставили

компьютер с отключенным устройством в BIOS. Из операционной системы, например, может показаться, что у вас нет передних USB-портов или CD-накопителя. Глядя на настройки BIOS, вы узнаете, почему эти устройства недоступны.

- **Чтобы изменить настройки устройства.** Иногда настройки по умолчанию, которые входят в BIOS, в вашей ситуации не сработают. В таком случае может понадобиться изменить следующие настройки в BIOS:
  - **сетевые параметры загрузки PXE.** Большинство современных сетевых карт способны загружаться с серверов, находящихся в сети. Если при загрузке из сети вы обнаружите, что карта сетевого интерфейса не отображается как загрузочное устройство на экране Boot Order (Порядок загрузки), вам, возможно, придется включить эту функцию в BIOS;
  - **настройки виртуализации.** Если вы хотите запустить систему Linux в качестве виртуального хоста, процессор компьютера должен поддерживать технологии Intel Virtual Technology или AMD Secure Virtual Machine (SVM). Но возможно, даже если процессор поддерживает данные технологии, они отключены в BIOS. Чтобы включить поддержку, перейдите на экран настройки BIOS и найдите пункт Virtualization (Виртуализация) (возможно, он относится к категории Performance (Производительность)). Убедитесь, что виртуализация включена.

Диагностика порядка загрузки

В зависимости от оборудования, подключенного к вашему компьютеру, типичный порядок загрузки может включать сначала дисковод CD/DVD, затем жесткий диск, затем USB-устройство и, наконец, карту сетевого интерфейса. BIOS будет обращаться к каждому устройству и искать загрузчик в его главной загрузочной записи. Найдя загрузчик, он запускает его. Если загрузчик не найден, BIOS переходит к следующему устройству до тех пор, пока устройства не закончатся. Если загрузчик не найден, компьютер не загрузится.

Одна из возможных проблем с порядком загрузки заключается в том, что устройство, которое нужно загрузить, может вообще не отображаться в порядке загрузки. В этом случае перейдите на экран Setup (Настройки), как описано в предыдущем разделе, и либо включите устройство, либо измените настройки, чтобы сделать его загрузочным.

Если устройство, с которого вы хотите загрузиться, отображается в порядке загрузки, нужно просто переместить выделение клавишей со стрелкой, чтобы отметить нужное устройство, и нажать клавишу Enter. Далее приведены причины выбора собственного устройства для загрузки.

- **Режим восстановления.** Если Linux не загружается с жесткого диска, загрузка с CD или USB-накопителя позволяет сделать это в режиме восстановления (будет описан далее в этой главе), который может помочь восстановить жесткий

диск в незагружаемой системе. Дополнительную информацию см. в разделе «Диагностика в режиме восстановления» далее в этой главе.

- **Новая установка.** Иногда в порядке загрузки сначала указывается жесткий диск. Если вы решите заново установить операционную систему, то потребуется выбрать загрузочное устройство, на котором находится инсталлятор (CD, DVD, USB-накопитель или сетевая карта).

Предположим, что вы преодолели все проблемы в настройках BIOS. Теперь перейдем к следующему шагу — запуску загрузчика BIOS.

Диагностика загрузчика операционной системы GRUB

Как правило, BIOS находит главную загрузочную запись на первом жестком диске и начинает поэтапную загрузку этого загрузчика. Глава 9 «Установка Linux» описывает загрузчик GRUB, который используется с большинством современных систем Linux, включая RHEL, Fedora и Ubuntu. Загрузчик GRUB в RHEL 6, описанный далее, — это более ранняя версия загрузчика GRUB 2, входящего в состав RHEL 7 и более поздних версий Fedora и Ubuntu. (Далее в этой главе я познакомлю вас и с GRUB 2.)

Сейчас нас интересует функция загрузчика с точки зрения того, что делать, если он выходит из строя, или какими способами можно его прервать, чтобы изменить протекание загрузки.

Загрузчик операционной системы GRUB Legacy

Перечислим несколько причин выхода загрузчика из строя в RHEL 6 и способы преодоления этих сбоев.

- **Не удалось найти активный раздел.** Когда загрузчик устанавливается на носитель, раздел для него обычно помечается как загрузочный. Если появляется сообщение о том, что раздел не найден, значит, возникла ошибка. Если вы уверены, что загрузчик находится на диске, используйте команду `fdisk` (возможно, с восстановительного носителя), чтобы сделать раздел загрузочным, и повторите попытку. Дополнительная информация о команде `fdisk` есть в разделе о разбиении жестких дисков в главе 12 «Управление дисками и файлами».
- **Выбранное загрузочное устройство недоступно.** Вы можете увидеть подобное сообщение, когда главная загрузочная запись была удалена с жесткого диска или содержимое жесткого диска ожидает загрузки с другого загрузчика, например с загрузочного CD. Вначале посмотрите, будет ли система загружаться с других носителей. Если окажется, что главная загрузочная запись удалена, загрузите аварийный носитель, чтобы попытаться восстановить содержимое диска. Но если главная загрузочная запись утрачена, возможно, другие данные на диске тоже стерты или для их восстановления потребуется специальная экспертиза. Если главная загрузочная запись была просто перезаписана (что

может произойти, если вы установили другую операционную систему в другой раздел диска), возможно, удастся ее переустановить из режима восстановления (как это делается, говорится в разделе «Диагностика в режиме восстановления» далее в этой главе).

- **Появляется текстовое приглашение загрузчика GRUB.** BIOS может запустить загрузчик GRUB и сразу перейти к его приглашению, не выбирая операционную систему. Вероятно, это означает, что основная часть загрузочной записи GRUB найдена, но, когда загрузчик изучил жесткий диск, чтобы найти следующий этап процесса загрузки и меню операционных систем для загрузки, он не смог их найти. Иногда это происходит, когда BIOS обнаруживает диски в неверном порядке и ищет файл `grub.conf` в неправильном разделе.

  Один из вариантов решения проблемы (при этом файл `grub.conf` находится в первом разделе первого диска), состоит в том, чтобы перечислить содержимое этого файла и ввести строки `root`, `kerner` и `initrd` вручную. Чтобы перечислить содержимое файла, введите команду `cat (hd0, 0)/grub/grub.conf`. Если это не сработает, введите `hd0, 1`, чтобы получить доступ к следующему разделу на этом диске и т. д., или `hd1,0`, чтобы использовать первый раздел следующего диска и т. д. Когда найдете строки, представляющие файл `grub.conf`, вручную введите строки `root`, `kerner` и `initrd` для нужной строки (заменив расположение жесткого диска, найденного в корневой строке), а затем — команду `boot`. Система должна запуститься, и вы сможете вручную исправить файлы загрузчика. Дополнительные сведения о загрузчике GRUB см. в главе 9.

Если BIOS находит загрузчик в главной загрузочной записи диска, а тот находит файлы конфигурации GRUB на диске, он начинает обратный отсчет времени в секундах, как определено значением строки `timeout` в файле `/boot/grub/grub.conf` (в RHEL 6 это 5 секунд). Во время обратного отсчета можно прервать работу загрузчика (до того, как он загрузит операционную систему по умолчанию), нажав любую клавишу.

Прервав работу загрузчика, вы увидите меню доступных строк загрузки. Они представляют различные доступные ядра загрузки. Но могут быть и совершенно разными операционными системами (например, Windows, BSD или Ubuntu).

Причины прерывания процесса загрузки из меню загрузки для диагностики Linux таковы.

- **Необходимость загрузиться на другом уровне выполнения.** Системы RHEL 6 обычно запускаются на уровне выполнения 3 (загрузка до текстового приглашения) или 5 (загрузка до графического интерфейса). Вы можете переопределить уровень выполнения по умолчанию, выделив строку ядра в меню загрузки и поставив в конце другое значение уровня. Для этого выделите нужную строку операционной системы, введите `e`, выделите ядро, введите `e` и укажите в конце строки новый уровень выполнения (например, поставьте пробел и цифру `1`, чтобы перейти в однопользовательский режим). Затем нажмите клавишу Enter и введите `b`, чтобы загрузить новую строку.

В чем смысл загрузки на разных уровнях выполнения для диагностики неполадок? Уровень выполнения 1 обходит аутентификацию, поэтому загрузка происходит непосредственно в корневую строку. Это полезно, если вы забыли пароль `root` и его нужно изменить (для этого введите команду `passwd`). Уровень выполнения 3 обходит запуск интерфейса рабочего стола. Перейдите на уровень выполнения 3, если возникли проблемы с видеодрайвером и вы хотите отладить его без автоматического запуска графического интерфейса.

- **Необходимость выбрать другое ядро.** Когда система RHEL устанавливает новое ядро, она всегда сохраняет по крайней мере одно старое. Если новое ядро выходит из строя, вы всегда можете вернуть предыдущее. Чтобы загрузить другое ядро из меню загрузчика GRUB, используйте клавиши ←, ↑, ↓ и →, чтобы выделить нужное ядро, и нажмите клавишу Enter, чтобы запустить процесс.
- **Желание выбрать другую операционную систему.** Если на вашем жестком диске установлена другая операционная система, можете загрузить ее вместо RHEL. Например, если на одном компьютере есть системы Fedora и RHEL и последняя не работает, вы можете загрузиться в Fedora, смонтировать нужные файловые системы RHEL и попытаться устранить проблему.
- **Необходимость изменить параметры загрузки.** Обратите внимание на то, что в ядро передается множество параметров. Они должны содержать как минимум имя ядра (например, `vmlinuz-2.6.32.el6.x86_64`) и раздел с корневой файловой системой (например, `/dev/mapper/abc-root`). По желанию в строку ядра можно добавить и другие параметры.

  Вы можете изменить параметры ядра и добавить функции в ядро или временно отключить поддержку конкретного компонента. Например, добавление строки `init=/ bin/bash` приводит к тому, что система обходит процесс `init` и переходит прямо к оболочке (аналогично запуску `init 1`). В дистрибутиве RHEL 7 добавление `1` в качестве параметра ядра не поддерживается, поэтому к однопользовательскому режиму можно перейти с помощью команды `init=/bin/bash`. Строка `nousb` временно отключит USB-порты (чтобы убедиться, что все подключенное к ним также будет отключено).

Предположим, нужное ядро выбрано. Дальше загрузчик попытается запустить его, включая содержимое начального диска оперативной памяти, который содержит драйверы и другое программное обеспечение, необходимое для загрузки конкретного оборудования.

Загрузчик операционной системы GRUB 2

Методы диагностики Linux из командной строки загрузчика GRUB 2 аналогичны методам, используемым в устаревшей командной строке загрузчика GRUB. Следуйте приведенным далее инструкциям, чтобы прервать процесс загрузки GRUB в самых последних версиях систем Fedora, RHEL и Ubuntu.

1. После включения компьютера и сразу после того, как появится экран BIOS, нажмите любую клавишу (например, ↑). На экране появятся несколько пунктов меню, представляющих различные ядра.
2. Из доступных строк по умолчанию загружается последнее доступное ядро (оно выделено и готово к загрузке). Однако можете выбрать другое ядро, если верно одно из следующих условий.
   - Текущее ядро повреждено, и нужно выбрать более старое работающее ядро.
   - Вы хотите запустить ядро, которое представляет собой совершенно другую операционную систему, установленную на диске.
   - Вы хотите запустить восстановительное ядро.
3. Для запуска ядра Linux выделите нужное с помощью клавиш ↑ и ↓ и введите `e`. Появятся команды, которые запускают систему (рис. 21.1).

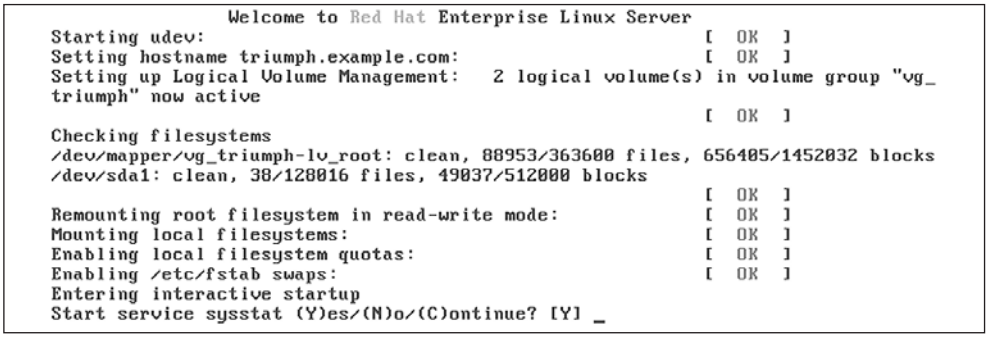


Рис. 21.1. Прервите загрузчик GRUB, чтобы изменить процесс загрузки

4. Чтобы добавить аргументы в ядро, переместите курсор в конец строки, начинающейся с `linux`, и введите нужные аргументы. (См. список параметров ядра на странице kernel.org/doc/Documentation/admin-guide/kernel-parameters.txt). Приведу два примера:
   - `selinux`. Если проблема на стороне системы SELinux, которая блокирует использование системы, отключите ее следующим образом:

     ```
     selinux=0
     ```

   - `smt`. Одновременная многопоточность (`smt`) позволяет одному ядру процессора выполнять несколько потоков. Чтобы исправить некоторые недостатки микропроцессора, нужно отключить эту функцию. Можно сделать это во время загрузки, введя аргумент `smt=1` или `nosmt` в командной строке ядра.
5. Когда закончите добавлять аргументы, нажмите сочетание клавиш Ctrl+x, чтобы загрузить систему с добавленными аргументами ядра.

Запуск ядра

После запуска ядра остается только следить за вероятными проблемами. В системе RHEL появится экран Red Hat Enterprise Linux с медленно вращающимся значком. Если хотите просмотреть сообщения с подробным описанием хода загрузки, нажмите клавишу Esc.

В этот момент ядро пытается загрузить драйверы и модули, необходимые для использования оборудования на компьютере. Основное, что следует искать на этом этапе, — сбои оборудования, которые могут помешать какой-либо функции работать должным образом (хотя они могут быстро прокручиваться). Сейчас это происходит гораздо реже, чем раньше, но, возможно, нет нужного для оборудования драйвера или загружен неправильный драйвер, который вызывает ошибки.

Прокручиваемые на экране сообщения, которые создаются при загрузке ядра, копируются в *кольцевой буфер* ядра. Как следует из названия, кольцевой буфер хранит сообщения ядра, удаляя старые сообщения после того, как заполнится. После полной загрузки компьютера вы можете войти в систему и ввести следующую команду, чтобы записать эти сообщения ядра в файл (а позже просмотреть их с помощью команды `less`):

```
# dmesg > /tmp/kernel_msg.txt
# less /tmp/kernel_msg.txt
```

Я чаще всего направляю сообщения ядра в файл (выберите любое имя), чтобы можно было изучить их позже или перенаправить кому-то, кто может помочь отладить любые проблемы. Сообщения появляются по мере обнаружения компонентов, таких как процессор, память, сетевые карты, жесткие диски и т. д.

В системах Linux, поддерживающих службу `systemd`, сообщения ядра хранятся в журнале `systemd`. Таким образом, помимо команды `dmesg` вы можете запустить команду `journalctl`, чтобы увидеть сообщения ядра с момента загрузки до настоящего времени. Пример сообщений ядра системы RHEL 7:

```
# journalctl -k
-- Logs begin at Sat 2019-11-23 10:36:31 EST,
      end at Sun 2019-12-08 08:09:42 EST. --
Nov 23 10:36:31 rhel81 kernel: Linux version 4.18.0-147.0.3.el8_1.x86_64
    (mockbuild@x86-vm-09.build.eng.bos.redhat.com)
       (gcc version 8.3.1 20190507 (Red Hat 8.3.1-4)
          (GCC)) #1 SMP Mon Nov 11 12:58:36 UTC 2019
Nov 23 10:36:31 rhel81 kernel: Command line:
  BOOT_IMAGE=(hd0,msdos1)/vmlinuz-4.18.0-147.0.3.el8_1.x86_64
    root=/dev/mapper/rhel-root ro resume=/dev/mapper/rhel-swap
       rd.lvm.lv=rhel/root rd.lvm.lv=rhel/swap rhgb quiet
...
Nov 23 10:36:31 rhel81 kernel: Hypervisor detected: KVM
Nov 23 10:36:31 rhel81 kernel: kvm-clock: Using msrs 4b564d01 ...
```

Нужно найти драйверы, которые не загружаются, или сообщения, показывающие, что определенные функции оборудования не включены. Например, однажды

у меня была карта ТВ-тюнера (для просмотра телеканалов на экране компьютера), которая установила неправильный тип тюнера для обнаруженной карты. Используя информацию о номере модели ТВ-карты и типе неисправности, я обнаружил, что настройки карты драйвера позволяют пробовать различные варианты, пока я не найду ту, которая соответствует моей карте тюнера.

Описывая, как просматривать сообщения о запуске ядра, я немного забежал вперед. Прежде чем вы сможете войти в систему и просмотреть сообщения ядра, оно должно закончить вывод системы. Только завершив первоначальное обнаружение аппаратного обеспечения и загрузку драйверов, ядро передает системе инициализации контроль над всем остальным, что необходимо сделать для загрузки системы.

Диагностика системы инициализации

Первый процесс системы, в котором ядро только что запустилось, зависит от используемого средства инициализации. Для системы System V `init` это процесс `init`, для `systemd` — процесс `systemd`. В зависимости от того, что работает в вашей системе (для проверки введите `ps-ef | head`), следуйте описаниям System V или `systemd`, приведенным далее. В дистрибутиве RHEL 6, в котором работают и Upstart, и System V `init`, применяется инициализация System V.

Диагностика системы инициализации System V

Еще несколько лет назад большинство систем Linux использовали System V `init` для инициализации служб в системе Linux. В дистрибутиве RHEL 6, когда ядро передает управление ходом загрузки процессу `init`, последний проверяет файл `/etc/inittab` на наличие указаний о том, как загрузить систему.

Файл `inittab` сообщает процессу `init`, какой уровень выполнения установлен по умолчанию, а затем указывает на файлы в каталоге `/etc/init`, чтобы переназначить некоторые клавиши (например, Ctrl+Alt+Delete для перезагрузки системы), запустить виртуальные консоли и определить местоположение скрипта инициализации основных служб в системе `/etc/rc.sysinit`.

При диагностике неполадок Linux, возникающих после того, как процесс `init` начался, двумя вероятными виновниками оказываются обработка файла `rc.sysinit` и скрипты уровня выполнения.

Диагностика файла rc.sysinit

Как следует из названия, скрипт `/etc/rc.sysinit` инициализирует множество основных функций системы. Когда этот файл запускается с помощью команды `init rc.sysinit`, устанавливаются имя хоста системы, файлы `/proc` и `/sysfilesystems`, SELinux, параметры ядра и выполняются десятки других действий.

Одна из наиболее важных функций `rc.sysinit` — настройка хранилища в системе. На самом деле, если процесс загрузки завершается неудачно во время обработки

файла rc.sysinit, по всей вероятности, скрипт не смог найти, смонтировать или расшифровать локальные или удаленные устройства хранения данных, необходимые для запуска системы.

Далее приведен список распространенных сбоев, которые могут возникнуть при выполнении задач из файла sysinit, и даны варианты их устранения.

- **Ошибка локального монтирования.** Если запись в файле /etc/fstab не монтируется, процесс загрузки завершается до запуска служб уровня выполнения. Обычно это происходит, когда вы добавляете запись в файл /etc/fstab, в которой есть ошибка. Когда файл fstab не срабатывает, вы попадаете в оболочку суперпользователя с установленной корневой файловой системой только для чтения. Чтобы устранить эту проблему, нужно перемонтировать корневую файловую систему, исправить файл fstab, смонтировать запись файловой системы, чтобы убедиться, что теперь она работает, и перезагрузиться. Вот как выглядит нужная последовательность команд:

```
# mount -o remount,rw /
# vim /etc/fstab
# mount -a
# reboot
```

ПРИМЕЧАНИЕ

Команда vim применяется, в частности, при редактировании файла /etc/fstab, поскольку она поддерживает его формат. При использовании команды vim столбцы окрашиваются в соответствующий цвет и выполняется проверка ошибок. Например, записи в поле Mount Options становятся зелеными, если они допустимы, и черными, если нет.

- **Не установлено имя хоста.** Если имя хоста установлено неправильно, проверьте обработку файла rc.sysinit, чтобы увидеть ошибки. Чтобы задать имя хоста системы, скрипт rc.sysinit использует значение HOSTNAME= в файле /etc/sysconfig/network. Если этот параметр не задан, то вместо него применяется имя localhost. Значение имени хоста также может быть получено с DHCP-сервера.
- **Не удается расшифровать файловую систему.** Скрипт rc.sysinit ищет в файле /etc/crypttab информацию, необходимую для расшифровки зашифрованных файловых систем. Если этот файл будет поврежден, может потребоваться найти его резервную копию, чтобы расшифровать файловую систему. Не повезет, если вам будет предложено ввести пароль, а вы его не знаете.

Другие особенности также задаются в скрипте rc.sysinit. Он устанавливает режим SELinux и загружает аппаратные модули. А еще создает программные RAID-массивы и настраивает группы томов и систему управления логическими томами. Проблемы, возникающие в любой из этих областей, отражаются в сообщениях об ошибках, которые появляются на экране после загрузки ядра и перед запуском процессов уровня выполнения.

Диагностика процессов уровней выполнения

В Red Hat Enterprise Linux 6.x и ранних версиях, когда система загружается, службы запускаются на основе уровня выполнения по умолчанию. Существует семь уровней выполнения, от 0-го до 6-го. Уровень выполнения по умолчанию обычно равен 3 (для сервера) или 5 (для рабочего стола). Далее приведены описания уровней запуска в системах Linux вплоть до RHEL 6.

0. Завершение работы компьютера. Все службы отключены, работа компьютера остановлена.

1. Однопользовательский режим. Запускаются только процессы, необходимые для загрузки компьютера, включая монтирование всех файловых систем, и имеющие доступ к системе с консоли. Сеть и сетевые службы не запускаются. Этот уровень обходит обычную проверку подлинности и загружает в корень запрос пользователя — так называемый `sulogin`. Вы можете применить этот режим, чтобы немедленно стать суперпользователем и изменить пароль. (Можете также применить слово `single` вместо `1`, чтобы перейти на однопользовательский режим выполнения. Разница между `single` и `1` заключается в том, что `single` не запускает скрипты в каталоге `/etc/rc1.d`.)

2. Многопользовательский режим без поддержки сети. Этот уровень выполнения используется редко. Первоначальный смысл его утерян. Ранние системы UNIX применяли этот уровень для запуска процессов `tty` в системах, где было несколько неинтеллектуальных терминалов, подключенных к системе. Это позволяло многим людям одновременно получать доступ к системе с символьных терминалов (многие работали из оболочки без графического интерфейса). Сетевые интерфейсы, как правило, не запускались, потому что постоянно работающие сетевые интерфейсы не были распространены. В наши дни уровень выполнения 2 обычно запускает сетевые интерфейсы, но не все.

3. Многопользовательский режим без поддержки сети. Этот уровень выполнения обычно применяется на серверах Linux, которые загружаются не с графическим интерфейсом, а с простым текстовым приглашением в консоли. Сеть запущена, как и все сетевые службы. Графическая среда рабочего стола может быть установлена или не установлена (как правило, нет) на машинах, загружающихся на уровень выполнения 3, но и графические среды должны быть запущены после загрузки.

4. Пользовательский уровень. Этот уровень запускает те же службы, что и уровень выполнения 3. Его можно применять, если вы хотите установить различные службы, доступные с уровней выполнения 3 и 4. Этот уровень выполнения обычно не используется. Вместо этого для загрузки задействуется уровень выполнения 3 или 5, при этом администратор по мере необходимости просто включает или выключает службы для работающей системы.

5. Многопользовательский режим с поддержкой сети и графической оболочки. Это уровень выполнения обычно применяется в настольных системах Linux. Он запускает сеть и все сетевые службы, а кроме того, графическое приглашение

входа в систему в консоли. Когда пользователи входят в систему, они видят графическую среду рабочего стола.

6. Перезагрузка компьютера. Этот уровень похож на уровень выполнения 0 в том смысле, что завершает все службы и останавливает все процессы. Однако затем уровень 6 запускает систему.

Уровни выполнения предназначены для установки степени активности в системе Linux. Уровень выполнения по умолчанию задан в файле `/etc/inittab`, но вы можете изменить его в любое время с помощью команды `init`. Например, от имени суперпользователя введите команду `init 0` для завершения работы, `init 3`, если хотите завершить графический интерфейс (с уровня выполнения 5), но оставить все остальные службы активными, или `init 6` для перезагрузки.

Уровни выполнения по умолчанию (другими словами, уровень выполнения, на который вы загружаетесь) — это 3 (для сервера) и 5 (для рабочего стола). Часто серверы не имеют установленных рабочих столов, поэтому загружаются на уровень 3, чтобы не затрачивать ресурсы на обработку или не рисковать безопасностью из-за того, что рабочий стол работает на веб-серверах или файловых серверах.

Вы можете увеличивать или уменьшать уровень выполнения. Например, администратор, выполняющий техническое обслуживание системы, может загрузиться на уровень 1, а затем ввести `init 3`, чтобы загрузить все необходимые службы на сервере. Для настройки рабочего стола можно загрузиться на уровень 5, затем спуститься на уровень 3, чтобы исправить рабочий стол (например, установить новый драйвер или изменить разрешение экрана), а потом ввести `init 5`, чтобы вернуться на рабочий стол.

Уровень служб на каждом уровне выполнения определяется сценариями последних, настроенных на запуск. Есть каталоги для каждого уровня запуска: файлы `/etc/rc0.d/`, `/etc/rc1.d/`, `/etc/rc2.d/`, `/etc/rc3.d/` и т. д. Когда с приложением связан скрипт выполнения, он помещается в каталог `/etc/init.d/`, а затем символически связывается с файлом в каждом каталоге `/etc/rc?.d/`.

Скрипты, связанные с каждым каталогом `/etc/rc?.d`, начинаются с буквы `K` или `S`, за которыми следуют две цифры и имя службы. Скрипты, начинающиеся с буквы `K`, указывают, что служба должна быть остановлена, а скрипты, начинающиеся с буквы `S`, — что она должна быть запущена. Два следующих числа указывают порядок запуска службы. Вот примеры файлов, которые можно найти в каталоге `/etc/rc3.d/` и которые настроены на запуск:

- `S01sysstat` — начинает собирать системную статистику;
- `S08iptables` — запускает брандмауэр `iptables`;
- `S10network` — запускает сетевые интерфейсы;
- `S12rsyslog` — запускает системный журнал;
- `S28autofs` — запускает автоматическое монтирование;
- `S50bluetooth` — запускает службу Bluetooth;
- `S55sshd` — запускает службу `secure shell`;

- `S58ntpd` — запускает службу синхронизации времени NTP;
- `S85httpd` — запускает веб-сервер Apache;
- `S90crond` — запускает службу `crond`;
- `S91smb` — запускает службу Samba;
- `S97rhnsd` — запускает службу Red Hat Network;
- `S99local` — запускает определенные пользователем локальные команды.

Эти службы, запущенные из каталога `/etc/rc3.d`, дают представление о порядке загрузки процессов при входе на уровень выполнения 3. Обратите внимание на то, что службы `sysstat` (собирает системную статистику) и `iptables` (создает брандмауэр системы) запускаются до запуска сетевых интерфейсов. За ними следуют `rsyslog` (служба логирования системы) и различные сетевые службы.

К моменту запуска скриптов уровня выполнения у вас уже должна быть работающая система. В отличие от некоторых других систем Linux, которые запускают все скрипты для уровня выполнения 1, затем 2, затем 3 и т. д., RHEL переходит прямо в каталог, представляющий уровень выполнения, сначала останавливая все службы, которые начинаются с него, и запуская все те, которые начинаются с этого каталога.

При запуске каждого скрипта `S` вы увидите сообщение о том, запущена ли служба. Вот что может пойти не так на этом этапе запуска системы.

- **Запуск службы не удался.** Для правильной загрузки службе может потребоваться доступ к сетевым интерфейсам или дисковому разделу, который не установлен. Большинство служб просто приостанавливаются, и запускается следующий скрипт. Войдя в систему, вы сможете настроить службу. Некоторые методы настройки служб включают изменения в процессе демона, чтобы он перенаправлял больше данных в файл журнала, или запуск демона вручную, чтобы сообщения об ошибках приходили прямо на экран. Дополнительную информацию о запуске служб вручную см. в главе 15 «Запуск и остановка служб».
- **Служба может зависнуть.** Некоторые службы, не получившие то, что им нужно для запуска, могут бесконечно зависать, не позволяя вам войти в систему и устранить проблему. Некоторые процессы занимают больше времени при первом появлении после свежей установки, поэтому нужно подождать несколько минут, чтобы увидеть, что скрипт по-прежнему работает, а не просто завис.

Если не получается пройти мимо зависшей службы, перезагрузитесь в *интерактивный режим запуска*, который предлагается перед запуском каждой службы. Чтобы войти в этот режим в RHEL, перезагрузите и прервите работу загрузчика (нажмите любую клавишу, когда увидите 5-секундный обратный отсчет). Выделите строку, которую хотите загрузить, и введите `e`. Выделите строку ядра и введите `e`. Затем добавьте слово `confirm` в конец строки ядра, нажмите клавишу Enter и введите `b`, чтобы загрузить новое ядро.

На рис. 21.2 показан пример сообщений, которые появляются при загрузке RHEL в интерактивном режиме запуска.

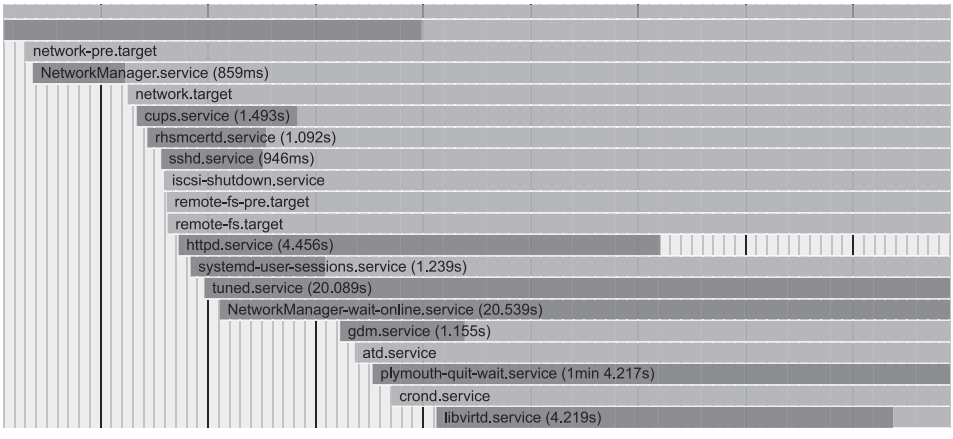


Рис. 21.2. Подтвердите службу в интерактивном режиме запуска RHEL

Большинство сообщений, показанных на рис. 21.2, генерируются из скрипта `rc sysinit`.

После приветственного сообщения запускается `udev`, чтобы следить за новым оборудованием, подключенным к системе, и загружать драйверы по мере необходимости. Задается имя хоста, активизируется управление логическими томами (LVM), проверяются все файловые системы (с добавленными томами LVM), монтируются все еще не смонтированные файловые системы, корневая файловая система перемонтируется на чтение и запись, и включаются все разделы подкачки LVM. Дополнительную информацию о LVM и других типах разделов и файловых систем см. в главе 12 «Управление дисками и файлами».

Последнее сообщение `Entering interactive startup` сообщает вам, что процесс скрипта `rc.sysinit` завершен и службы для выбранного уровня выполнения готовы к запуску. Поскольку система находится в интерактивном режиме, появляется сообщение с вопросом, хотите ли вы запустить первую службу (`sysstat`). Введите `Y`, чтобы запустить ее, и перейдите к следующей. Найдя неисправную службу, запрашивающую запуск, введите `N`, чтобы она не запускалась. Если решите, что остальные службы можно запустить, введите `C` для продолжения запуска остальных служб. После того как система обнаружит, что неисправные службы не запущены, вернитесь назад и исправьте их.

Последний комментарий о скриптах запуска. Файл `/etc/rc.local` — это одна из последних служб, запускаемых на каждом уровне выполнения. На уровне выполнения 5 он связан с файлом `/etc/rc5.d/S99local1`. Любую команду, которую нужно запускать при каждой загрузке системы, можно поместить в файл `rc.local`.

Вы можете использовать файл `rc.local`, чтобы отправить сообщение по электронной почте или запустить быстрое правило брандмауэра `iptables` при запуске системы. В общем, лучше применить существующий скрипт запуска или создать

новый самостоятельно, чтобы можно было управлять командой или командами как службой. Помните, что файл `rc.local` — это быстрый и простой способ заставить команды запускаться при каждой загрузке системы.

Диагностика системы инициализации systemd

Последние версии систем Fedora, RHEL и Ubuntu используют `systemd` вместо System V `init` в качестве системы инициализации. И хотя `systemd` более сложна, чем System V `init`, она предлагает больше вариантов анализа того, что происходит во время инициализации.

Процесс загрузки systemd

Когда демон `systemd` (`/usr/lib/systemd/systemd`) запускается после запуска ядра, он приводит в движение все другие настроенные для запуска службы. В частности, отключает содержимое файла `/etc/systemd/system/default.target`, как в примере:

```
# cat /etc/systemd/system/default.target
...
[Unit]
Description=Graphical Interface
Documentation=man:systemd.special(7)
Requires=multi-user.target
Wants=display-manager.service
Conflicts=rescue.service rescue.target
After=multi-user.target rescue.service rescue.target display-manager.service
AllowIsolate=yes
```

Файл `default.target` на самом деле является символической ссылкой на файл в каталоге `/lib/systemd/system`. Для сервера он может быть связан с файлом `multi-user.target`, для рабочего стола — связан с файлом `graphical.target` (как показано в примере).

В отличие от инициализации System V `init`, которая просто запускает служебные скрипты в алфавитно-цифровом порядке, служба `systemd` должна работать в обратном направлении от файла `default.target`, чтобы определить, какие службы и целевые объекты запускаются. В примере `default.target` — это символическая ссылка на файл `graphical.target`. Перечисляя содержимое этого файла, вы можете увидеть следующее:

- требуется, чтобы файл `multi-user.target` был загружен первым;
- после него загружается служба `display-manager.service`.

Продолжив изучать, что требуется этим двум единицам, вы можете найти то, что нужно еще. Например, `multi-user.target` требует, чтобы `basic.target` (запускает различные базовые службы) и `display-manager.service` (запускает диспетчер отображения, `gdm`) запускали графический экран входа в систему.

Чтобы просмотреть службы, запускаемые файлом `multi-user.target`, перечислите содержимое каталога `/etc/systemd/system/multi-user.target.wants`, как в примере далее:

```
# ls /etc/systemd/system/multi-user.target.wants/
atd.service            ksm.service              rhsmcertd.service
auditd.service         ksmtuned.service         rpcbind.service
avahi-daemon.service   libstoragemgmt.service   rsyslog.service
chronyd.service        libvirtd.service         smartd.service
crond.service          mcelog.service           sshd.service
cups.path              mdmonitor.service        sssd.service
dnf-makecache.timer    ModemManager.service     tuned.service
firewalld.service      NetworkManager.service   vdo.service
irqbalance.service     nfs-client.target        vmtoolsd.service
kdump.service          remote-fs.target
```

Эти файлы являются символическими ссылками на файлы, которые определяют, что начинать для каждой из этих служб. В системе они могут включать удаленную оболочку (`sshd`), печать (`cups`), аудит (`auditd`), сеть (NetworkManager) и др. Ссылки добавляются в этот каталог либо при установке пакета для службы, либо при включении службы с помощью команды `systemctl enable`.

Имейте в виду, что в отличие от System V `init` система `systemd` может запускать или останавливать единичные файлы, которые представляют собой нечто большее, чем просто службы, и иным образом управлять ими. Она может управлять устройствами, автоматическим монтированием, путями, розетками и т. д. После того как `systemd` все запустила, вы можете войти в систему, чтобы предотвратить любые потенциальные проблемы.

После входа в систему команда `systemctl` позволяет увидеть каждый unit-файл, который `systemd` пытается запустить, например:

```
# systemctl
UNIT                                        LOAD   ACTIVE SUB
      DESCRIPTION
proc-sys-fs-binfmt_misc.automount           loaded active waiting
      Arbitrary Executable File Formats File System
sys-devices-pc...:00:1b.0-sound-card0.device loaded active plugged
      631xESB/632xESB High Definition Audio Control
sys-devices-pc...:00:1d.2-usb4-4\x2d2.device loaded active plugged
      DeskJet 5550
...

-.mount                                  loaded active mounted  Root Mount
boot.mount                               loaded active mounted  /boot
...
autofs.service                            loaded active running
      Automounts filesystems on demand
cups.service                              loaded active running
      CUPS Scheduler
httpd.service                             loaded failed failed
      The Apache HTTP Server
```

Из вывода `systemctl` можно понять, произошел ли сбой какого-либо единичного файла. В примере видно, что запустить `httpd.service` (ваш веб-сервер) не удалось. Продолжая исследование, используйте команду `journalctl -u` для этой службы, чтобы увидеть, были ли добавлены какие-то сообщения об ошибках:

```
# journalctl -u httpd.service
...
Dec 08 09:30:36 rhel81-bible systemd[1]: Starting The Apache HTTP
Server...
Dec 08 09:30:36 rhel81-bible httpd[16208]: httpd: Syntax error
    on line 105 of /etc/httpd/conf/httpd.conf:
    /etc/httpd/conf/httpd.conf:105: <Directory> was not closed.
Dec 08 09:30:36 rhel81-bible systemd[1]: httpd.service: Main process exited,
    code=exited, status=1/FAILURE
Dec 08 09:30:36 rhel81-bible systemd[1]: httpd.service:
    Failed with result 'exit-code'.
Dec 08 09:30:36 rhel81-bible systemd[1]:
    Failed to start The Apache HTTP Server.
```

Из выходных данных видно, что в файле `httpd.conf` обнаружилось несоответствие директив (не удалось закрыть раздел каталога). После того как ошибка была исправлена, я смог запустить службу (`systemctl start httpd`). Если другие unit-файлы отображаются как невыполненные, можно снова запустить команду `journalctl -u`, используя эти имена unit-файлов в качестве аргументов.

Анализ процесса загрузки systemd

Чтобы точно увидеть, что произошло в ходе загрузки системы, использующей службу `systemd`, система `systemd` применяет инструмент `systemd-analyze`. Чтобы узнать, остановились ли службы, или чтобы найти место для собственной службы `systemd`, задействуйте эту команду для анализа всего процесса загрузки. Вот несколько примеров:

```
# systemd-analyze time
Startup finished in 1.775s (kernel) + 21.860s (initrd)
    + 1min 42.414s (userspace) = 2min 6.051s
graphical.target reached after 1min 42.121s in userspace
```

Параметр `time` позволяет увидеть, сколько времени заняла каждая фаза процесса загрузки, от старта ядра до конца загрузки целевого объекта по умолчанию. Вы можете использовать команду `plot` для создания SVG-графики каждого компонента процесса запуска (в примере применяется команда `eog` для отображения выходных данных):

```
# systemd-analyze plot > /tmp/systemd-plot.svg
# eog /tmp/systemd-plot.svg
```

На рис. 21.3 показан небольшой фрагмент вывода из гораздо более крупного графика.

Рис. 21.3. Фрагмент вывода из команды systemd-analyze startup plot

Здесь видны службы, которые запускаются после запуска службы `NetworkManager.service`. Темно-красный цвет показывает время, в течение которого запускается служба или цель. Если служба продолжает работать, это отображается светло-красным цветом. В этом примере служба `httpd.service` не запустилась после попытки запуска в течение 4,456 секунды. Это видно по белой полосе справа, обозначающей, что служба не работает. На этом этапе вы можете использовать команду `journalctl`, как описано ранее, чтобы устранить проблему.

В следующем разделе говорится о том, как устранить неполадки, которые могут возникнуть с программными пакетами.

Диагностика пакетов программного обеспечения

Пакеты программного обеспечения (такие как `yum` для RPM и `apt-get` для DEB) предназначены для облегчения управления системным программным обеспечением. (См. главу 10 «Управление программами» для получения основных сведений о том, как управлять пакетами программного обеспечения.) Однако, несмотря на все настройки, иногда пакет программного обеспечения может выйти из строя.

ПРИМЕЧАНИЕ

Команда dnf заменила yum в последних системах Fedora и RHEL. В этом разделе часто описывается команда yum, так как в большинстве случаев она будет работать как в старых, так и в новых системах Fedora и RHEL, поскольку у нее есть псевдоним dnf. Команда dnf используется, когда это требуется.

В следующих разделах описываются распространенные проблемы, которые могут произойти с пакетами RPM в системе RHEL или Fedora, и способы их решения.

Иногда при попытке установить или обновить пакет с помощью команды `yum` сообщения об ошибках говорят о том, что зависимые пакеты, необходимые для установки, недоступны. Это может произойти и в малом масштабе (когда вы пытаетесь установить один пакет), и в большом (когда пробуете обновить всю систему).

Из-за частых обновлений и больших репозиториев Fedora и Ubuntu несоответствия в зависимостях пакетов возникают чаще, чем в более стабильных выборочных репозиториях (например, как в системе Red Hat Enterprise Linux). Вот что можно сделать, чтобы избежать сбоев зависимостей.

- **Используйте обновленные, хорошо проверенные репозитории.** В дистрибутиве Fedora есть тысячи программных пакетов. Если вы задействуете основные репозитории Fedora для установки программного обеспечения текущей версии, то с ними редко возникают проблемы.

  После того как пакеты будут добавлены в репозиторий, проверены ответственными за репозиторий и настроены (и вы при этом не работаете с внешними репозиториями), все, что требуется для установки выбранного пакета, окажется доступно. Однако при использовании сторонних репозиториев проявляются ошибки в зависимостях, которые они не могут контролировать. Например, если репозиторий создает новую версию собственного программного обеспечения, для которой требуются более поздние версии базового программного обеспечения (допустим, библиотеки), необходимые версии могут быть недоступны из репозитория Fedora.
- **Постоянно обновляйте свою систему.** Каждую ночь запускайте команду `dnf update` (или `yum update` в старых системах), чтобы снизить вероятность столкновения с серьезными проблемами зависимостей, что может произойти при обновлении системы лишь раз в несколько месяцев. В системах с рабочим столом GNOME окно программного обеспечения позволяет проверять наличие обновлений и применять их к установленным пакетам программного обеспечения. В системах Fedora 22 и RHEL 8 (и более поздних версиях) появилась возможность добавлять автообновления (fedoraproject.org/wiki/AutoUpdates). Программа AutoUpdates автоматически загружает обновленные пакеты, если они доступны, а если они настроены, может и установить их. Еще один подход заключается в создании задания для проверки или запуска ночных обновлений. Подробнее о том, как это сделать, см. во врезке «Обновление программного обеспечения с помощью команды cron».
- **Время от времени обновляйте дистрибутивы.** Дистрибутивы Fedora и Ubuntu выпускают новые версии каждые шесть месяцев. Fedora прекращает предоставлять обновленные пакеты для каждой версии через 13 месяцев после ее выпуска. Таким образом, нет необходимости обновляться до новой версии каждые шесть месяцев, однако вы должны делать это минимум раз в год, иначе можете

столкнуться с проблемами зависимостей и безопасности, когда обслуживание версии Fedora прекратится.

Чтобы обновиться до последней версии этих дистрибутивов (например, с Fedora 30 до Fedora 31), выполните следующие действия:

1) обновите до последней версии программное обеспечение:

```
# dnf upgrade --refresh -y
```

2) установите плагин `dnf-plugin-system-upgrade`:

```
# dnf install dnf-plugin-system-upgrade -y
```

3) начните обновление до новой версии:

```
# dnf system-upgrade download --releasever=31
```

4) перезагрузите систему в процессе обновления:

```
# dnf system-upgrade reboot
```

Если хотите использовать стабильную систему, Red Hat Enterprise Linux — лучший вариант, потому что он предоставляет обновления для каждой основной версии в течение семи лет или дольше.

ПРИМЕЧАНИЕ

Если вы в системе Ubuntu обновляете свои пакеты с помощью команды apt-get, имейте в виду, что параметры update и upgrade в Ubuntu с командой apt-get имеют иные значения, чем с командами dnf или yum (Fedora и RHEL).

В системе Ubuntu команда apt-get update вызывает загрузку в локальную систему последних метаданных пакета (имен пакетов, номеров версий и т. д.). Команда apt-get upgrade приводит к тому, что система обновляет все установленные пакеты, у которых есть новые доступные версии, на основе последних загруженных метаданных.

А каждый раз, когда вы запускаете команду dnf или yum в Fedora или RHEL, загружаются последние метаданные о новых пакетах для текущей версии. Затем команда yum update загружает последние пакеты, доступные для текущей версии Fedora или RHEL. Чтобы перейти к следующей версии, необходимо запустить команду dnf system-upgrade, как описано ранее.

Если вы столкнулись с проблемой зависимостей, вот несколько действий, которые можно предпринять, чтобы попытаться решить эту проблему.

- **Используйте проверенные репозитории.** В последних версиях хорошо известных дистрибутивов (например, RHEL, Fedora или Ubuntu) проблемы с зависимостями встречаются редко и быстро устраняются. Но если вы применяете репозитории для более старых версий или находящиеся в разработке (например, Fedora Rawhide), скорее всего, проблем с зависимостями будет больше. Переустановите или обновите пакеты, чтобы устранить эти проблемы.
- **Используйте сторонние приложения и репозитории только в случае необходимости.** Чем дальше вы находитесь от ядра дистрибутива Linux, тем больше

вероятность того, что когда-нибудь у вас возникнут проблемы с зависимостями. Всегда просматривайте основные репозитории своего дистрибутива, прежде чем искать пакеты в другом месте или пытаться создать их самостоятельно.

Даже если пакет изначально работает, возможно, в дальнейшем он не будет обновлен. Пакет из стороннего репозитория может перестать работать, если создатели не предоставят новую версию при изменении зависимых пакетов.

- **Изучите связанные с ядром зависимости.** Если вы загружаете сторонние пакеты RPM для такого оборудования, как видеокарты или беспроводные сетевые карты, содержащие драйверы ядра, и устанавливаете более позднее ядро, эти драйверы перестанут работать. В результате может случиться, что графический экран входа в систему не будет запускаться при загрузке системы или сетевая карта не станет загружаться, поэтому у вас не будет беспроводной сети.

  Поскольку большинство систем Linux сохраняют два самых последних ядра, вы можете перезагрузиться, прервать загрузчик GRUB и выбрать предыдущее (все еще работающее) ядро, с которого сможете загрузиться. Это заставит систему работать со старым ядром и действующими драйверами, пока вы исправляете неполадки.

  Основное решение состоит в том, чтобы загрузить новый драйвер, перестроенный для текущего ядра. Такие сайты, как rpmfusion.org, создают сторонние пакеты драйверов с открытым исходным кодом и обновляют их при появлении нового ядра. С помощью репозитория rpmfusion.org ваша система будет получать свежие драйверы при добавлении нового ядра.

  В качестве альтернативы сайту rpmfusion.org вы можете пойти прямо на сайт производителя и загрузить его драйверы Linux (Nvidia предлагает драйверы Linux для своих видеокарт) или, если для драйвера доступен исходный код, попробовать создать его самостоятельно.

- **Исключите часть пакетов из обновления.** Если вы обновляете множество пакетов одновременно, исключите те, которые влияют на другие, поскольку вы ищете проблемные. Вот пример того, как обновить все пакеты, нуждающиеся в обновлении, за исключением пакета `somepackage` (замените `somepackage` именем пакета, который хотите исключить):

```
# yum -y --exclude=somepackage update
```

Обновление программного обеспечения с помощью команды cron

Команда `cron` настраивает выполнение команд в заданное время и с заданными интервалами. Вы можете установить точные минуту, час, день или месяц выполнения команды. Можете настроить выполнение команды каждые пять минут, каждый третий час или в определенное время в пятницу днем.

Чтобы применить команду для ночных обновлений программного обеспечения, настройте ее от имени суперпользователя, выполнив команду `crontab -e`. Откроется

файл с помощью редактора по умолчанию (команда `vi`), который настраивает файл `crontab`. Вот пример того, как может выглядеть созданный вами файл `crontab`:

```
#  min   hour   day/month   month   day/week command
   59    23     *           *       *        dnf -y update | mail root@localhost
```

Файл `crontab` состоит из пяти полей, обозначающих день и время, и поля, содержащего командную строку для запуска. Я добавил строку комментария, чтобы обозначить поля. В примере задействуется команда `dnf -y update`, а ее выходные данные отправляются по почте пользователю root@localhost. Команда выполняется в 23:59. Звездочки (`*`) необходимы в качестве заполнителей, указывающих на выполнение команды в каждый день месяца, месяц и день недели.

Создавая новую запись, убедитесь, что вы либо направляете вывод в файл, либо передаете его в команду, которая может работать с выводом. Если этого не сделать, все выходные данные будут отправлены пользователю, выполнившему команду `crontab -e` (в данном случае суперпользователю).

В файле `crontab` можете назначить диапазон чисел или список чисел либо вообще их пропустить. Например, `1`, `5` или `17` в первом поле вызывает выполнение команды через 1, 5 и 17 минут в тот час, который указывается в поле `hour`. Значение `*/3` во втором поле заставляет команду выполняться каждые три часа (полночь, 3 часа ночи, 6 часов утра и т. д.). Значение `1-3` в четвертом поле указывает `cron` выполнить команду в январе, феврале и марте. Дни недели и месяцы могут задаваться в виде цифр или слов.

Чтобы узнать больше о формате файла `crontab`, введите команду `man 5 crontab`. Чтобы прочитать о команде `crontab`, примените команду `man 1 crontab`.

Исправление ошибок баз данных RPM и кэша

Информация обо всех пакетах RPM в вашей системе хранится в локальной базе данных RPM. Хотя и гораздо реже, чем в более ранних версиях Fedora и RHEL, но база данных RPM может быть повреждена. Из-за этого нельзя устанавливать, удалять или перечислять пакеты RPM.

Если вы обнаружили, что команды `rpm` и `yum` зависают и не выполняются, выводя сообщение `rpmdb open fails`, попробуйте перестроить базу данных RPM. Чтобы убедиться, что в базе данных RPM есть проблема, запустите команду `yum check`.

Вот пример того, как выглядит вывод этой команды при поврежденной базе данных:

```
# yum check
error: db4 error(11) from dbenv->open: Resource temporarily
unavailable
error: cannot open Packages index using db4 – Resource temporarily
    unavailable (11)
error cannot open Packages database in /var/lib/rpm
CRITICAL:yum.main:
Error: rpmdb open fails
```

База данных RPM и другая информация об установленных пакетах RPM хранятся в каталоге `/var/lib/rpm`. Вы можете удалить файлы базы данных, начинающиеся с `__db*`, и перестроить их из метаданных, хранящихся в других файлах этого каталога.

Перед началом работы рекомендуется создать резервную копию каталога `/var/lib/rpm`. Затем нужно удалить старые файлы `__db*` и перестроить их заново. Для этого введите следующие команды:

```
# cp -r /var/lib/rpm /tmp
# cd /var/lib/rpm
# rm __db*
# rpm --initdb
```

Новые файлы `__db*` появятся в этом каталоге через несколько секунд. Выполните простую команду `rpm` или `yum`, чтобы убедиться, что базы данных теперь в порядке.

Точно так же, как RPM имеет базы данных локально установленных пакетов, так и функция `Yum` хранит информацию, связанную с репозиториями Yum, в локальном каталоге `/var/cache/yum`. С появлением команды `dnf` каталог кэша теперь называется `/var/cache/dnf`. Кэшированные данные включают метаданные, заголовки, пакеты и данные подключаемых модулей `yum`.

Если когда-нибудь возникнут проблемы с данными, кэшированными с помощью команды `yum`, вы можете их очистить. В следующий раз при запуске команды `yum` необходимые данные будут загружены снова. Причины для очистки кэша `yum` таковы.

- **Метаданные устарели.** При первом подключении к репозиторию Yum (путем загрузки пакета или запроса репозитория) метаданные загружаются в систему. Они состоят из информации обо всех доступных пакетах из репозитория.

  По мере добавления и удаления пакетов из репозитория метаданные должны обновляться, иначе система будет работать со старой информацией. По умолчанию запущенная команда `dnf` проверяет наличие новых метаданных, если старые метаданные старше 48 часов (или на значение минут `metadata_expire=`, установленное в файле `/etc/dnf/dnf.conf`).

  Если метаданные устарели, но срок их действия еще не истек, запустите команду `dnf clean metadata`, чтобы удалить все метаданные и принудительно загрузить новые метаданные при следующей загрузке. Кроме того, можете запустить команду `dnf makecache`, чтобы загрузить обновленные метаданные из всех репозиториев.

- **Не хватает места на диске.** Обычно `yum` может кэшировать до нескольких сотен мегабайт данных в каталогах `/var/cache/dnf`. Однако в зависимости от настроек файла `/etc/dnf/dnf.conf` (например, `keepcache=1`, который сохраняет все загруженные RPM даже после их установки) каталоги кэша могут содержать несколько гигабайт данных.

  Чтобы очистить все пакеты, метаданные, заголовки и другие данные, хранящиеся в каталоге `/var/cache/dnf`, введите следующее:

  ```
  # yum clean all
  ```

На этом этапе ваша система получает актуальную информацию из репозиториев при следующем запуске команды `yum`.

Далее говорится о диагностике неполадок в сети.

Диагностика сети

Поскольку все больше и больше информации, изображений, видео и другого контента, который мы используем каждый день, теперь доступно за пределами наших локальных компьютеров, рабочее сетевое соединение требуется почти в каждой компьютерной системе. Поэтому, если вы отключаете сетевое соединение или не можете связаться с другими системами, полезно узнать, что в Linux есть много инструментов для решения этой проблемы.

Для клиентских компьютеров (ноутбуков, настольных компьютеров и портативных устройств) требуется подключение к сети для доступа к другим компьютерным системам. На сервере необходимо, чтобы клиенты могли связаться с вами по сети. В следующих разделах описываются различные инструменты устранения неполадок сетевого подключения для клиентских и серверных систем Linux.

Диагностика исходящих соединений

Допустим, вы открываете браузер, но не можете попасть ни на один сайт. Скорее всего, вы не подключены к сети. Возможно, проблема заключается в разрешении имен, но она может быть связана и с подключением за пределами вашей локальной сети.

Чтобы проверить, работают ли исходящие сетевые соединения, можно воспользоваться множеством команд, описанных в главе 14 «Администрирование сети». Проверьте подключение с помощью простой команды `ping`. Чтобы проверить, работает ли перенос имен в адрес, примените `host` и `dig`.

В следующих разделах рассматриваются проблемы, с которыми вы можете столкнуться при подключении к сети для исходящих подключений, и инструменты для их устранения.

Обзор сетевых интерфейсов

Чтобы увидеть состояние ваших сетевых интерфейсов, используйте команду `ip`. В приведенном далее выводе показано, что кольцевой интерфейс (`lo`) включен (так что вы можете запускать сетевые команды в локальной системе), но карта `eth0` (ваша первая проводная сетевая карта) отключена (состояние `DOWN`). Если бы интерфейс был включен, строка `inet` показывала бы его IP-адрес. В примере же только кольцевой интерфейс имеет адрес `inet` (127.0.0.1):

```
# ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
```

```
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 state DOWN qlen
1000
    link/ether f0:de:f1:28:46:d9 brd ff:ff:ff:ff:ff:ff
```

По умолчанию в системах RHEL 8 и Fedora называния сетевых интерфейсов зависят от того, как они подключены к физическому оборудованию. Например, в RHEL 8 можно встретить сетевой интерфейс `enp11s0`. Это означает, что сетевой адаптер представляет собой проводную карту Ethernet (`en`) на плате PCI 11 (`p11`) и слоте 0 (`s0`). Беспроводной адаптер обозначается `wl`, а не `en`. Суть состоит в том, чтобы сделать имена сетевых карт более предсказуемыми, потому что при перезагрузке системы не гарантируется, как операционная система назовет сетевые интерфейсы `eth0`, `eth1` и т. д.

Проверка физического соединения

При проводном подключении убедитесь, что компьютер подключен к порту сетевого коммутатора. Если у вас несколько сетевых адаптеров, проверьте, чтобы кабель был подключен правильно. Если знаете имя сетевого интерфейса (`eth0`, `p4p1` или другое), которое понадобится, чтобы определить, какой сетевой адаптер связан с интерфейсом, введите команду `ethtool -p eth0` и посмотрите, какой сетевой адаптер на задней части компьютера мигает (нажмите сочетание клавиш Ctrl+C, чтобы остановить мигание). Подключите кабель к нужному порту.

Если вместо того, чтобы отобразить неработающий интерфейс, команда `ip` не показывает вообще никакого интерфейса, проверьте, не отключено ли оборудование. Возможно, карта проводного сетевого адаптера неплотно установлена в свой слот или сетевой адаптер отключен в BIOS.

При беспроводном соединении вы можете щелкнуть на значке программы NetworkManager и не увидеть доступных беспроводных интерфейсов. Они могут быть отключены в BIOS. На ноутбуке проверьте, не отключен ли небольшой переключатель, который отключает сетевую карту. Я встречал ситуации, когда пользователи полностью перекраивали сетевые настройки, чтобы найти проблему, а все дело было именно в этом маленьком переключателе.

Проверка маршрутов

Если сетевой интерфейс включен, но вы все еще не можете добраться до нужного хоста, проверьте маршрут к нему. Начните с проверки маршрута по умолчанию. Затем свяжитесь с устройством шлюза локальной сети в следующую сеть. Наконец, пропингуйте систему в Интернете:

```
# ip route show
default via 192.168.122.1 dev ens3 proto dhcp metric 100
192.168.122.0/24 dev ens3 proto kernel scope link src 192.168.122.194
metric 100
```

Строка `default` показывает, что шлюз по умолчанию находится по адресу 192.168.122.1 и туда можно попасть через карту `ens3`. Поскольку в примере используется только интерфейс `ens3` и показан лишь маршрут к сети 192.168.122.0, вся связь, не адресованная хосту в сети 192.168.122.0/24, передается через шлюз по умолчанию (192.168.122.1). Его правильнее называть маршрутизатором.

Чтобы добраться до своего маршрутизатора, попробуйте применить команду `ping`, как в этом примере:

```
# ping -c 2 192.168.122.1
PING 192.168.122.1 (192.168.122.1) 56(84) bytes of data.
64 bytes from 192.168.122.1: icmp_seq=1 ttl=64 time=0.757 ms
64 bytes from 192.168.122.1: icmp_seq=2 ttl=64 time=0.538 ms
--- 192.168.122.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 65ms
rtt min/avg/max/mdev = 0.538/0.647/0.757/0.112 ms
```

Сообщение о том, что конечный хост недоступен (`Destination Host Unreachable`), свидетельствует, что маршрутизатор либо выключен, либо физически не подключен к компьютеру (возможно, он не подключен к коммутатору, который вы совместно используете). Если команда была выполнена и вы можете связаться с маршрутизатором, далее необходимо найти адрес за пределами маршрутизатора.

Попробуйте пропинговать широкодоступный IP-адрес. Например, IP-адрес публичного DNS-сервера Google — 8.8.8.8 (`ping-c 2 8.8.8.8`). Если это удалось, ваша сеть в порядке и, скорее всего, должным образом не работает перенаправление имени хоста на адрес.

Если вы можете связаться с удаленной системой, но соединение очень медленное, примените команду `traceroute`, чтобы следовать по маршруту к удаленному хосту. Например, каждый шаг на пути к www.google.com показывает команда:

```
# traceroute www.google.com
```

Выходные данные демонстрируют время, затраченное на каждый шаг на пути к сайту Google. Вместо команды `traceroute` используйте команду `mtr` (`yum install mtr`), чтобы просмотреть маршрут к хосту. С помощью команды `mtr` маршрут запрашивается непрерывно, так что можно наблюдать за выполнением всех этапов маршрута с течением времени.

Проверка разрешения имени хоста

Если вы не можете связаться с удаленными хостами по имени, но можете сделать это, отправив IP-адреса, значит, есть проблемы с разрешением имен хостов. Системы, подключенные к Интернету, преобразуют имена в адреса, связываясь с сервером системы доменных имен (DNS), который может предоставить им IP-адреса запрашиваемых хостов.

DNS-сервер, используемый вашей системой, можно ввести вручную или автоматически получить с DHCP-сервера при запуске сетевых интерфейсов. В любом случае имена и IP-адреса одного или нескольких DNS-серверов попадают в ваш файл `/etc/resolv.conf`. Вот пример этого файла:

```
search example.com
nameserver 192.168.0.254
nameserver 192.168.0.253
```

При подключении к имени хоста в системах Fedora или Red Hat Enterprise Linux выполняется поиск файла `/etc/hosts`, затем запрашиваются записи сервера имен в файле `resolv.conf` в том порядке, в котором они появляются. Вот несколько способов решения проблем с переводом имен в адрес.

- **Проверьте, можно ли связаться с DNS-сервером.** Зная адреса серверов имен, вы можете применить команду `ping` для IP-адреса каждого сервера имен, чтобы узнать, доступен ли он. Например, `ping -c 2 192.168.0.254`. Если IP-адреса можно достичь, возможно, вам был назначен неправильный адрес для DNS-сервера или он в настоящее время не работает.
- **Проверьте, работает ли сервер DNS.** Каждый DNS-сервер необходимо использовать с командой `host` или `dig`. Например, любая из этих команд проверяет, может ли DNS-сервер на адресе 92.168.0.254 разрешить имя хоста `www.google.com` для IP-адреса. Повторите эту команду для каждого IP-адреса сервера имен, пока не определите, какие из них работают:

```
# host www.google.com 192.168.0.254
Using domain server:
Name: 192.168.0.254
Address: 192.168.0.254#53
Aliases:
www.google.com has address 172.217.13.228
www.google.com has IPv6 address 2607:f8b0:4004:809::2004
# dig @192.168.0.254 www.google.com
...
;; QUESTION SECTION:
;www.google.com.              IN  A

;; ANSWER SECTION:
www.google.com.       67  IN  A   172.217.13.228
...
```

- **Настройте свои DNS-серверы.** Если вы определили, что у вас есть неправильные IP-адреса, установленные для DNS-серверов, изменить эти адреса может быть довольно сложно. Поищите в файле `/var/log/messages` или в выводе команды `journalctl` IP-адреса своих DNS-серверов. Если для подключения к сети и DHCP-серверу задействуется программа NetworkManager, вы увидите строки сервера имен с назначенными IP-адресами. Если адреса неверны, можете их переопределить.

  При использовании NetworkManager нельзя просто добавить записи о сервере имен в файл `/etc/resolv.conf`, поскольку программа перезаписывает этот файл собственными записями о сервере. Вместо этого добавьте строку `PEERDNS=no` в файл `ifcfg` для сетевого интерфейса (например, `ifcfg-eth0` в каталоге `/etc/sysconfig/network-scripts`). Затем установите `DNS1=192.168.0.25` (или свой IP-адрес DNS-сервера). Новый адрес будет задействован при следующем перезапуске сети.

При использовании сетевой службы вместо программы NetworkManager все равно можно применить строку `PEERDNS=no`, чтобы предотвратить перезапись DHCP-сервером ваших DNS-адресов. Однако в этом случае вы можете отредактировать файл `resolv.conf` непосредственно для установки адресов DNS-серверов.

Описанная проверка исходящего сетевого подключения применима к любому типу системы, будь то ноутбук, настольный компьютер или сервер. Входящие соединения чаще всего не становятся серьезной проблемой для ноутбуков или настольных компьютеров, потому что большинство запросов просто отклоняются. А способы сделать сервер доступным, если у клиентов проблемы с доступом к службам, описываются в следующем подразделе.

Диагностика входящих соединений

Диагностика сетевых интерфейсов отличается от диагностики настольных систем. Поскольку большинство систем Linux настроены как серверы, необходимо знать, как устранять проблемы, возникающие при попытке связаться с серверами Linux.

Начнем с веб-сервера Apache (`httpd`), работающего в системе Linux, к которому не могут подключиться веб-клиенты. В следующих разделах описаны действия, помогающие обнаружить проблему.

Проверьте, может ли клиент вообще связаться с вашей системой

Чтобы сервер стал общедоступным, имя хоста вашей системы должно быть разрешимым, чтобы любой клиент в Интернете мог к нему подключиться. Это подразумевает блокировку вашей системы на определенном общедоступном IP-адресе и его регистрацию на общедоступном DNS-сервере. Используйте для этого регистратор доменов, например www.networksolutions.com.

Когда клиенты не могут получить доступ к вашему сайту по имени из своих браузеров и при этом клиент — это система Linux, примените `ping` `host` `racerout` и другие команды, описанные в предыдущем разделе, чтобы выяснить, в чем проблема с подключением. Системы Windows имеют собственную версию `ping`.

Если перенос «имя — адрес» работает и вы можете пинговать свой сервер извне, нужно рассмотреть доступность службы.

Проверьте, доступна ли служба клиенту

С помощью клиента Linux вы можете проверить, доступна ли искомая служба (в данном случае `httpd`) с сервера. Один из способов сделать это — использовать команду `nmap`.

Команда `nmap` — это популярный у системных администраторов инструмент для проверки различных видов информации в сетях. Но этот инструмент популярен и у хакеров, потому что он позволяет сканировать серверы в поисках потенциальных уязвимостей. Таким образом, можете использовать команду `nmap` для сканирования собственных систем на наличие проблем, но знайте, что ее применение в другой системе похоже на взлом.

Проверить собственную систему, чтобы увидеть, какие порты на сервере открыты для внешнего мира (по сути, это проверка того, какие службы работают), совершенно законно и просто. После установки пакета `nmap` (`yum install nmap`) используйте имя хоста своей системы или IP-адрес, чтобы просканировать систему с помощью команды `nmap`:

```
# nmap 192.168.0.119
Starting Nmap 6.40 ( http://nmap.org ) at 2019-12-08 13:28 EST
Nmap scan report for spike (192.168.0.119)
Host is up (0.0037s latency).
Not shown: 995 filtered ports
PORT    STATE  SERVICE
21/tcp  open   ftp
22/tcp  open   ssh
80/tcp  open   http
443/tcp open   https
631/tcp open   ipp
MAC Address: 00:1B:21:0A:E8:5E (Intel Corporate)
Nmap done: 1 IP address (1 host up) scanned in 4.77 seconds
```

Выходные данные показывают, что TCP-порты открыты для обычных (`http`) и безопасных (`https`) веб-служб. Когда вы видите, что TCP-порты открыты, это говорит о том, что служба прослушивает порт. Значит, сетевое соединение в порядке и нужно устранять неполадки в том, как настроена сама служба (например, вы можете заглянуть в файл `/etc/httpd/conf/httpd.conf`, чтобы узнать, разрешен или запрещен доступ к определенным хостам).

Если TCP-порты 80 и/или 443 не отображаются, это означает, что они фильтруются. Вам нужно проверить, блокирует ли ваш брандмауэр эти порты (не принимает пакеты). Если порт не фильтруется, но состояние закрыто, значит, служба `httpd` либо не работает, либо не прослушивает эти порты. Необходимо войти на сервер и проверить это.

Проверьте брандмауэр на сервере

Со своего сервера запустите команду `iptables` для перечисления существующих правил таблицы фильтров, например:

```
# iptables -vnL
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target prot opt in out source    destination
...
```

```
   0     0 ACCEPT tcp  --  *  *   0.0.0.0/0 0.0.0.0/0   state NEW tcp
dpt:80
   0     0 ACCEPT tcp  --  *  *   0.0.0.0/0 0.0.0.0/0   state NEW tcp
dpt:443
...
```

Для систем RHEL 8 и Fedora 30, в которых включена служба `firewalld`, можно использовать окно Firewall (Межсетевой экран) для открытия необходимых портов. Выбрав вкладку Zone and Services (Общедоступная зона и службы), установите флажки для `http` и `https`, чтобы открыть эти порты для всего входящего трафика. Если ваша система применяет базовую службу `iptables`, среди других правил должны быть правила брандмауэра, подобные двум показанным в предыдущем коде. Если их нет, добавьте эти правила в файл `/etc/sysconfig/iptables`. Вот примеры того, как они могут выглядеть:

```
-A INPUT -m state --state NEW -m tcp -p tcp --dport 80 -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 443 -j ACCEPT
```

Добавив правила в файл, очистите остальные правила брандмауэра (`systemctl stop iptables.service` или `service iptables stop`), а затем добавьте их снова (`systemctl start iptables.service` или `service iptables start`).

Если брандмауэр все еще блокирует доступ клиента к портам веб-сервера, вот что еще сделайте в брандмауэре.

- **Проверьте порядок правил.** Проверьте правила в файле `/etc/sysconfig/iptables` и посмотрите, не стоит ли правило `DROP` или `REJECT` перед правилами, открывающими порты 80 и/или 443. Переместите эти правила вперед, поставив перед конечными `DROP` или `REJECT`, — это может помочь решить проблему.
- **Поищите отклоненные хосты.** Проверьте, не запрещают или не отклоняют ли какие-то правила пакеты определенных хостов или сетей. Ищите правила, включающие в себя `-s` или `--source1`, за которыми стоят IP-адрес или диапазон адресов, а затем `-j DROP` или `ACCEPT`. Измените правило или сделайте его приоритетным, чтобы создать исключение для хоста, которому нужно разрешить доступ к вашей службе.

Если порт открыт, но сама служба закрыта, убедитесь, что она работает и прослушивает соответствующие интерфейсы.

Проверьте службу на сервере

Если кажется, что доступ клиента к серверу ничего не блокирует через фактические порты, предоставляющие службу, стоит проверить ее саму. Предположим, что служба запущена (для проверки введите команду `service httpd status` или `systemctl status httpd.service` в зависимости от своей системы), далее нужно проверить, прослушивает ли она соответствующие порты и сетевые интерфейсы.

Команда `netstat` — это отличный универсальный инструмент для проверки сетевых сервисов. Следующая за ней команда перечисляет имена и ID (`p`) всех

процессов, которые прослушивают (`l`) службы TCP (`t`) и UDP (`u`), а также номер порта (`n`) прослушивания. Командная строка отфильтровывает все строки, кроме связанных с процессом `httpd`:

```
# netstat -tupln | grep httpd
tcp    0  0 :::80        :::*         LISTEN     2567/httpd
tcp    0  0 :::443       :::*         LISTEN     2567/httpd
```

В примере показано, что процесс `httpd` прослушивает порты 80 и 443 для всех интерфейсов. Вполне возможно, что он может прослушивать данные интерфейсы. Например, если процесс `httpd` прослушивает только локальный интерфейс (127.0.0.1) для HTTP-запросов (порт 80), запись об этом будет выглядеть следующим образом:

```
tcp    0  0 127.0.0.1:80 :::*         LISTEN     2567/httpd
```

Для службы `httpd` и других сетевых служб, которые прослушивают запросы на сетевых интерфейсах, можно отредактировать основной файл конфигурации службы (в данном случае `/etc/httpd/conf/httpd.conf`), чтобы она прослушивала порт 80 на всех адресах (`Listen 80`) или на конкретном адресе (`Listen 192.168.0.100:80`).

Диагностика памяти

Диагностика и устранение неполадок с производительностью компьютера — одна из самых важных, хотя и неочевидных задач, которые необходимо решить. Возможно, у вас есть система, которая раньше работала нормально, но в конце концов замедлилась до такой степени, что стала практически непригодной. Возможно, приложения начинают зависать без видимой причины. Чтобы найти и устранить проблемы такого рода, необходимо подключить дедукцию.

В системы Linux входит множество инструментов для наблюдения за происходящими в системе событиями. Используя различные утилиты Linux, вы можете выяснить, какие процессы потребляют большие объемы памяти или предъявляют высокие требования к процессорам, дискам или сети. Перечислю, что можно с этим сделать.

- **Увеличить объем памяти.** Компьютер будет пытаться выполнить задачу в любом случае, но сбои могут произойти и из-за того, что у него не хватает памяти, вычислительной мощности, дискового пространства или недостаточно пропускной способности сети, чтобы обеспечить разумную производительность. Даже заканчивающиеся ресурсы могут вызвать проблемы с производительностью. Повышение производительности компьютера — самый простой способ решения этих проблем.
- **Настроить систему.** Системы Linux поставляются с настройками по умолчанию, которые определяют, как система сохраняет, перемещает и защищает данные.

Системные параметры можно изменить, если настройки по умолчанию не позволяют приложениям в вашей системе хорошо работать.

- **Обнаружить проблемные приложения или проблемных пользователей.** Иногда система работает плохо, потому что пользователь или приложение работают неправильно. Неверно настроенные или сбойные приложения могут зависнуть или поглотить все ресурсы компьютера. Неопытный пользователь может по ошибке запустить несколько программ, которые истощат системные ресурсы. Системному администратору необходимо знать, как найти и устранить эти проблемы.

Чтобы устранить проблемы с производительностью в Linux, используйте некоторые из основных инструментов наблюдения за процессами, запущенными в вашей системе, и управления ими. Обратитесь к главе 6 «Управление активными процессами», чтобы узнать о командах `ps`, `top`, `kill` и `killall`. В следующих разделах мы рассмотрим команду `memstat`, чтобы глубже разобраться в том, как работают процессы и где могут произойти сбои в работе.

Наиболее сложная область диагностики в Linux связана с управлением виртуальной памятью. В следующих подразделах обсудим, как просматривать виртуальную память и управлять ею.

Выявление проблем с памятью

Компьютеры могут хранить данные постоянно (жесткие диски) и временно (*оперативная память (RAM)* и *раздел подкачки*). Представим, что вы — процессор, работающий за столом, и хотите закончить свою работу. Данные, которые требуется постоянно хранить, вы бы поместили в шкаф в другом конце комнаты (похоже на хранение на жестком диске). А информацию, используемую в данный момент, поместили бы на свой стол (как оперативная память на компьютере).

Раздел подкачки позволяет расширить оперативную память. На самом деле это просто место, где можно расположить временные данные, которые не помещаются в оперативную память, но в какой-то момент могут понадобиться процессору. Хотя раздел подкачки находится на жестком диске, это не обычная файловая система Linux, в которой данные хранятся постоянно.

Оперативная память обладает иными, нежели жесткий диск, свойствами.

- **Она ближе к процессору.** Подобно тому как рабочий стол находится рядом с вами, когда вы работаете, память физически находится на материнской плате компьютера рядом с процессором. Таким образом, он сразу же может использовать любые необходимые ему данные, если они размещаются в оперативной памяти.
- **Она быстрее.** Оперативная память ближе к процессору и доступ к ней проще (твердотельные или механические жесткие диски), что позволяет процессору получать информацию из нее гораздо быстрее, чем с жесткого диска. Быстрее

взглянуть на лист бумаги на столе (маленьком, тесном пространстве), чем идти к шкафу в другом конце комнаты и долго искать то, что нужно.

- **Она меньше по объему.** Новый компьютер может иметь жесткий диск объемом 1 Тбайт или больше, но при этом 8 или 16 Гбайт оперативной памяти. Чтобы поместить каждый файл и каждый фрагмент данных, которые могут понадобиться процессору, в оперативную память (что заставило бы компьютер работать быстрее), она должна быть очень большой. Кроме того, как слоты физической памяти на компьютере, так и сама компьютерная система (64-разрядные компьютеры могут выделять больше оперативной памяти, чем 32-разрядные) могут ограничить объем оперативной памяти компьютера.
- **Она дороже.** Хотя оперативная память сейчас намного доступнее, чем было десять лет и даже пару лет назад, она все еще намного дороже (за гигабайт) жестких дисков.
- **Она временная.** Оперативная память содержит данные и метаданные, используемые процессором для работы (плюс данные, которые ядро Linux хранит поблизости, потому что считает, что процесс вскоре будет нуждаться в них). Однако при выключении компьютера все, что находится в оперативной памяти, теряется. Когда процессор заканчивает работу с данными, они отбрасываются, если не нужны, остаются в оперативной памяти, если вскоре понадобятся, или помечаются для переноса на диск постоянного хранения, если их нужно сохранить.

Конечно, важно понимать разницу между временным (оперативная память) и постоянным (жесткий диск) хранилищем, но и это еще не все. Если запрос на память превышает возможности ОЗУ, ядро может временно переместить данные из ОЗУ в область, называемую *разделом подкачки*.

Если вернуться к аналогии с письменным столом, это можно описать так: «На столе не осталось места, но мне нужно положить на него больше бумаг, необходимых для проектов, над которыми я сейчас работаю. Вместо того чтобы хранить бумаги, которые мне скоро понадобятся, в шкафу, я отведу специальное место (например, ящик стола) для хранения тех документов, с которыми все еще работаю и которые не готов хранить постоянно или выбросить».

Дополнительная информация о файлах, разделах подкачки и о том, как их создавать, есть в главе 12 «Управление дисками и файлами». А сейчас стоит изучить виды разделов подкачки и условия их использования.

- Когда данные переносятся из оперативной памяти в раздел подкачки (перекачиваются из памяти), производительность страдает. Помните, что запись данных на диск происходит намного медленнее, чем запись в оперативную память.
- Когда нужные данные возвращаются из раздела подкачки в оперативную память (перекачиваются в память), производительность снова уменьшается.
- Когда системе Linux не хватает места в оперативной памяти, процесс перехода в раздел подкачки похож на потерю высокой передачи в автомобиле.

Машина переключится на более низкую передачу, но не остановится полностью. Другими словами, все ваши процессы остаются активными, не теряют данных и не выходят из строя полностью, но производительность системы значительно снижается.

- Если и оперативная память, и раздел подкачки заполнены и никакие данные нельзя удалить или записать на диск, система перейдет в состояние *нехватки памяти* (out-of-memory, OOM). Когда это происходит, включается «убийца» ядра и начинает завершать все процессы один за другим, чтобы восстановить столько памяти, сколько нужно ядру, чтобы снова начать нормально функционировать.

Общим правилом всегда было то, что применение раздела подкачки — это плохо и его следует избегать. Тем не менее некоторые пользователи утверждают, что бывают случаи, когда более агрессивная передача данных в раздел подкачки может действительно улучшить производительность.

Представьте, вы открываете документ в текстовом редакторе, а затем сворачиваете его на рабочий стол, не используя в течение нескольких дней и работая над другими задачами. Если бы данные из этого документа были перенесены на диск, было бы доступно больше оперативной памяти для более активных приложений, которым это пространство пригодилось бы. Удар по производительности наступит в следующий раз, когда понадобится получить доступ к данным из отредактированного документа и данные будут перенесены с диска в оперативную память. Настройки системы, относящиеся к тому, насколько агрессивно она перекачивает данные, называются *swappiness*.

Насколько возможно, система Linux хочет сделать доступным все, что нужно активному приложению. И вновь проведем аналогию с рабочим столом: если я работаю над девятью активными проектами и на столе есть место для хранения информации, необходимой для всех них, почему бы не оставить их все в пределах досягаемости на столе? Точно так же и ядро иногда хранит в оперативной памяти библиотеки и другие данные, которые, по его мнению, могут понадобиться, даже если процесс не использует их в данный момент.

Тот факт, что ядро склонно хранить в оперативной памяти информацию, которая, как оно ожидает, может понадобиться в ближайшее время, даже если не нужна сейчас, может заставить неопытного системного администратора думать, что система практически исчерпала оперативную память и процессы вот-вот начнут зависать. Поэтому важно знать о различных видах информации, хранящейся в памяти, чтобы можно было определить, когда она действительно заканчивается. Проблема заключается не только в нехватке оперативной памяти, но и в том, что оперативная память заканчивается, когда остаются только данные, не подлежащие переносу в раздел подкачки.

Учитывайте эти сведения о *виртуальной памяти* (ОЗУ и раздел подкачки), поскольку в следующем разделе описываются способы устранения связанных с ней неполадок.

Проверка проблем с памятью

Допустим, на рабочем столе Linux запущено много приложений и все они начинают зависать. Чтобы выяснить, что это проблемы с производительностью, возникшие из-за нехватки памяти, выполните команды `top` и `ps`, которые ищут информацию о потреблении памяти в системе.

Чтобы отследить потребление памяти, введите команду `top`, а затем прописную букву `M`, например:

```
# top
top – 22:48:24 up 3:59, 2 users, load average: 1.51, 1.37, 1.15
Tasks: 281 total,  2 running, 279 sleeping,  0 stopped,  0 zombie
Cpu(s): 16.6%us, 3.0%sy, 0.0%ni, 80.3%id, 0.0%wa, 0.0%hi, 0.2%si, 0.0%st
Mem:  3716196k total, 2684924k used, 1031272k free,  146172k buffers
Swap: 4194296k total,       0k used, 4194296k free,  784176k cached
  PID USER     PR  NI  VIRT  RES  SHR S %CPU  %MEM TIME+  COMMAND
 6679 cnegus   20  0  1665m  937m 32m S   7.0 25.8 1:07.95 firefox
 6794 cnegus   20  0   743m  181m 30m R  64.8  5.0 1:22.82 npviewer.bin
 3327 cnegus   20  0  1145m  116m 66m S   0.0  3.2 0:39.25 soffice.bin
 6939 cnegus   20  0   145m   71m 23m S   0.0  2.0 0:00.97 acroread
 2440 root     20  0   183m   37m 26m S   1.3  1.0 1:04.81 Xorg
 2795 cnegus   20  0  1056m   22m 14m S   0.0  0.6 0:01.55 nautilus
```

В выводе есть две строки (`Mem` и `Swap`) и четыре столбца информации (`VIRT`, `RES`, `SHR` и `%MEM`), относящейся к памяти. В примере видно, что в строке `Mem` оперативная память не исчерпана (задействовано только `2684924 k` из `3716196 k`) и в строке `Swap` память не используется диском (`0 k`).

Но если суммировать только первые шесть строк вывода в столбце `VIR`, будет видно, что для приложений выделено 4937 Мбайт памяти, а это превышает 3629 Мбайт общей доступной оперативной памяти (`3716196 k`). А все потому, что столбец `VIR` отображает только объем памяти, выделенный приложению. Строка `RES` показывает объем фактически используемой памяти без подкачки, который составляет всего 1364 Мбайт.

Обратите внимание на то, что при сортировке по применению памяти с помощью прописной буквы `M` команда `top` знает, что нужно сортировать по столбцу `RES`. Столбец `SHR` отображает память, которая может быть совместно использована другими приложениями (например, библиотеками), а столбец `MEM` показывает процент общей памяти, потребляемой каждым приложением.

Если вы думаете, что система достигла состояния нехватки памяти, в данных нужно найти следующую информацию.

- Свободное пространство строки `Mem` равно нулю или близко к этому.
- Значение используемого пространства в строке `Swap` не равно нулю и продолжает расти. Это сопровождается замедлением производительности системы.
- Поскольку экран `top` обновляется каждые несколько секунд, то, если есть процесс с утечкой памяти (постоянно запрашивающий и использующий память,

но не возвращающий ее обратно), объем памяти VIRT растет, но, что более важно, расти будет и память RES.

- Если раздел подкачки действительно заканчивается, ядро начнет немедленно завершать процессы, чтобы справиться с состоянием нехватки памяти.

Если у вас установлен и включен интерфейс Cockpit, можете через него наблюдать за применением памяти в режиме реального времени. Откройте интерфейс Cockpit и выберите System ▸ Memory & Swap (Система ▸ Память и обмен). На рис. 21.4 показана система, в которой вся память потребляется несколькими видеопотоками, поэтому подключился раздел подкачки.

```
load_video
set gfx_payload=keep
insmod gzio
linux ($root)/vmlinuz-5.3.7-301.fc31.x86_64 root=/dev/mapper/fedora_localhost-\
-live-root ro resume=/dev/mapper/fedora_localhost--live-swap rd.lvm.lv=fedora_\
localhost-live/root rd.lvm.lv=fedora_localhost-live/swap rhgb quiet
initrd ($root)/initramfs-5.3.7-301.fc31.x86_64.img









      Press Ctrl-x to start, Ctrl-c for a command prompt or Escape to
      discard edits and return to the menu. Pressing Tab lists
      possible completions.
```

Рис. 21.4. Контролируйте использование оперативной памяти и подкачки в режиме реального времени с помощью интерфейса Cockpit

Решение проблем с памятью

Перечислю, что можно быстро сделать, чтобы справиться с состоянием нехватки памяти.

- **Немедленно завершить процесс.** Если проблема с памятью связана с одним ошибочным процессом, можете просто завершить его. Войдите в систему как суперпользователь или владелец активного процесса, введите в верхнем окне PID процесса, который хотите завершить, и выберите значение 15 или 9 в качестве сигнала.
- **Удалить кэш страниц.** Если вы просто хотите очистить часть памяти, чтобы решить проблему, сообщите системе, чтобы она удалила неактивный кэш страниц. После этого часть страниц памяти будет записана на диск, а другие удалены, потому что они хранятся постоянно и при необходимости их можно загрузить с диска.

  Это действие равносильно очистке рабочего стола и отправке всей информации, кроме самой важной, в корзину или в шкаф. Возможно, вскоре вам снова по-

надобится получить информацию из шкафа, но почти наверняка не все данные сразу. Продолжайте применять команду `top` в одном окне Terminal (Терминал), чтобы увидеть изменение строки `Mem` при вводе от имени суперпользователя в другое окно Terminal (Терминал) следующей команды:

```
# echo 3 > /proc/sys/vm/drop_caches
```

- **Завершить процесс за пределами памяти.** Иногда нехватка памяти делает систему настолько непригодной для использования, что нельзя получить ответ от оболочки или графического интерфейса. В этих случаях вы можете применить сочетание клавиш Alt+SysRq, чтобы немедленно завершить процесс, находящийся вне памяти. Это сочетание клавиш полезно и в случае зависания программ, так как ядро обрабатывает запросы Alt+SysRq раньше других.

  Чтобы активизировать сочетание клавиш Alt+SysRq, необходимо заранее установить строку `/proc/sys/kernel/sysrq` в значение `1`. Проще всего это сделать, добавив строку `kernel.sysrq =1` в файл `/etc/sysctl.conf`. Кроме того, необходимо использовать сочетание клавиш Alt+SysRq из текстового интерфейса (например, из виртуальной консоли, которая появляется при нажатии сочетания клавиш Ctrl+Alt+F2). Если строка `kernel.sysrq` установлена в значение `1`, вы можете завершить процесс в своей системе с самым высоким значением `OOM`, нажав сочетание клавиш Alt+SysRq+f из текстового интерфейса. На экране появится список всех процессов, запущенных в системе, а в конце будет указано имя завершенного процесса. Можно повторять эти действия до тех пор, пока не будет завершено достаточно процессов, чтобы снова появился нормальный доступ к системе из оболочки.

ПРИМЕЧАНИЕ

Сочетание клавиш Alt+SysRq можно использовать также для работы с невосприимчивой системой. Например, Alt+SysRq+e завершает все процессы, кроме init. Alt+SysRq+t выводит в консоль список всех текущих задач и информацию о них. Чтобы перезагрузить систему, нажмите Alt+SysRq+b. См. файл sysrq.txt в каталоге /usr/share/doc/kernel-doc*/Documentation, чтобы получить дополнительную информацию о сочетании клавиш Alt+SysRq.

Диагностика в режиме восстановления

Если ваша система Linux не загружается, лучше всего, вероятно, перейти в режим восстановления. Чтобы сделать это, нужно обойти систему Linux на жестком диске и загрузить какой-нибудь восстанавливающий носитель (например, загрузочный USB-ключ или CD). После загрузки восстанавливающий носитель попытается смонтировать любые файловые системы, которые найдет в системе Linux, чтобы можно было устранить любые проблемы.

Для многих дистрибутивов Linux установочный CD или DVD могут служить загрузочным носителем для перехода в режим восстановления. Рассмотрим, как можно использовать установочный DVD системы RHEL для перехода в режим

восстановления и исправления нерабочей системы Linux (запишите образ на USB-накопитель, если на компьютере нет DVD-накопителя).

1. Загрузите установочный образ CD или DVD и запишите его на соответствующий носитель (CD или DVD). Информацию о записи на CD и DVD см. в приложении А. (В качестве примера я использовал установочный DVD Red Hat Enterprise Linux 8.)
2. Вставьте CD или DVD в дисковод на компьютере, на котором установлена нерабочая система Linux, и перезагрузите компьютер.
3. Как только появится экран BIOS, нажмите функциональную клавишу, отмеченную на этом экране для выбора загрузочного носителя (возможно, это будет F12 или F2).
4. Выберите дисковод (CD или DVD) из списка загрузочных устройств и нажмите клавишу Enter.
5. Когда появится меню загрузки RHEL 8, с помощью клавиш ←, ↑, ↓ и → выделите слово Troubleshooting (Диагностика) и нажмите клавишу Enter. На других загрузочных носителях Linux может быть вариант Rescue Mode (Режим восстановления) или что-то подобное. На следующем экране выберите пункт Rescue a Red Hat Enterprise Linux system (Восстановить Red Hat Enterprise Linux) и нажмите клавишу Enter.
6. Через несколько мгновений система Linux загрузится на восстановительном носителе. При появлении запроса выберите язык и клавиатуру, а также ответьте, хотите ли запустить сетевые интерфейсы в системе.
7. Если вы считаете, что вам нужно загрузить что-то из другой системы в своей сети (например, пакеты RPM или средства отладки), выберите вариант Yes (Да) и настройте сетевые интерфейсы. Затем вас спросят, хотите ли вы попробовать смонтировать файловые системы из установленной системы Linux в файле `/mnt/sysimage`.
8. Выберите Continue (Продолжить), чтобы продолжить монтирование файловых систем (если это возможно) в каталоге `/mnt/sysimage`. Если все пройдет успешно, появится сообщение о том, что файловые системы были смонтированы в файл `/mnt/sysimage`.
9. Выберите OK, чтобы продолжить. Появится приглашение оболочки для суперпользователя (`#`). Теперь можно приступить к диагностике из режима восстановления. После того как вы перейдете в этот режим, часть неповрежденной файловой системы будет смонтирована в каталоге `/mnt/sysimage`. Введите команду `ls /mnt/sysimage`, чтобы проверить наличие файлов и каталогов с жесткого диска.

В данный момент корень файловой системы (`/`) находится в файловой системе на восстановительном носителе. Для устранения неполадок в установленной системе Linux можно ввести следующую команду:

```
# chroot /mnt/sysimage
```

Теперь каталог `/mnt/sysimage` становится корнем вашей файловой системы (`/`), а также он выглядит как файловая система, установленная на вашем жестком диске. Перечислю то, что можно сделать, чтобы восстановить систему, пока вы находитесь в режиме восстановления.

- **Исправить файл /etc/fstab.** Если файловые системы не удалось смонтировать из-за ошибки в файле `/etc/fstab`, необходимо исправить все проблемные строки (например, неправильные имена устройств или несуществующий каталог точки монтирования). Введите команду `mount -a`, чтобы убедиться, что все файловые системы монтируются.
- **Переустановить недостающие компоненты.** Возможно, файловые системы в порядке, но система не загрузилась из-за отсутствия какой-то необходимой команды или файла конфигурации. Эту проблему можно устранить, переустановив пакет с отсутствующими компонентами. Например, если кто-то по ошибке удалил `/bin/mount`, у системы не будет команды для монтирования файловых систем. Пакет `utillinux` заменит отсутствующую команду `mount`.
- **Проверить файловые системы.** Если проблемы с загрузкой связаны с поврежденными файловыми системами, запустите команду `fsck` (проверка файловой системы), чтобы проверить, есть ли какие-либо повреждения в разделе диска. Если они есть, команда `fsck` попытается их исправить.

После окончания настройки системы введите команду `exit`, чтобы выйти из среды `chroot`, и вернитесь к макету файловой системы, который видит «живой» носитель. Если вы все закончили, введите команду `reboot`, чтобы перезагрузить систему. Перед этим обязательно извлеките носитель.

Резюме

Диагностика в системах Linux может начаться с момента включения компьютера. Проблемы могут возникнуть с BIOS компьютера, загрузчиком или другими частями процесса загрузки, которые можно исправить, перехватив их на разных этапах загрузки.

После запуска системы можно устранить неполадки с программными пакетами, сетевыми интерфейсами или исчерпанием памяти. Система Linux включает в себя множество инструментов для поиска и исправления любой части системы Linux, которая может сломаться и нуждается в исправлении.

Следующая глава посвящена теме безопасности Linux. Задействуя описанные в ней инструменты, вы сможете предоставить доступ к службам, необходимым вам и вашим пользователям, одновременно блокируя доступ к системным ресурсам, которые необходимо защитить.

Упражнения

Упражнения, приводимые в этом разделе, позволяют опробовать методы диагностики и устранения неполадок в Linux. Поскольку некоторые из описанных здесь методов потенциально могут повредить систему, я рекомендую не использовать рабочую систему, которую нельзя повреждать. В приложении Б представлены варианты решения упражнений.

Данные упражнения относятся к теме диагностики проблем в Linux. Они предполагают, что у вас компьютер со стандартной BIOS. Чтобы выполнить эти упражнения, необходимо иметь возможность перезагрузить компьютер и прервать любые задачи, которые он может выполнять.

1. Загрузите компьютер и, как только появится экран BIOS, перейдите в режим настроек, как указано на экране BIOS.
2. На экране настроек BIOS определите, является ваш компьютер 32- или 64-разрядным, включает ли он поддержку виртуализации и способна ли ваша сетевая интерфейсная карта загружать PXE.
3. Перезагрузитесь. Сразу после того, как экран BIOS исчезнет, появится обратный отсчет до загрузки системы Linux. Нажмите любую клавишу, чтобы перейти к загрузчику GRUB.
4. В загрузчике GRUB добавьте параметр загрузки до уровня выполнения 1, чтобы выполнить обслуживание системы.
5. После загрузки системы посмотрите на сообщения, созданные в кольцевом буфере ядра, которые отображают активность ядра при загрузке.
6. В дистрибутивах Fedora и RHEL запустите пробную версию команды `yum update` и исключите все имеющиеся пакеты ядра.
7. Проверьте, какие процессы прослушивают входящие соединения в вашей системе.
8. Проверьте, какие порты открыты на внешнем сетевом интерфейсе.
9. Выполните команду `top` в окне Terminal (Терминал). Откройте второе окно терминала, очистите кэш страниц и проверьте на экране команды `top`, доступно ли теперь больше памяти `RES`.
10. Если в системе подключен интерфейс Cockpit, откройте его, чтобы просмотреть подробную информацию о текущем использовании памяти и подкачки системы.

Часть V

Методы обеспечения безопасности в Linux

В этой части:

- Глава 22. Базовые методы обеспечения безопасности.
- Глава 23. Продвинутые методы обеспечения безопасности.
- Глава 24. Повышенная безопасность с технологией SELinux.
- Глава 25. Защита Linux в сети.

22 Базовые методы обеспечения безопасности

В этой главе

- Использование базовой безопасности.
- Мониторинг безопасности.
- Аудит и проверка безопасности.

На базовом уровне защита системы Linux начинается с физической безопасности, защиты данных, защиты учетных записей пользователей и безопасности программного обеспечения. Периодически нужно проверять все это, чтобы убедиться, что система остается безопасной.

Вот что нужно узнать.

- Кто может физически добраться до системы?
- Создаются ли резервные копии данных на случай опасности для системы?
- Хорошо ли защищены учетные записи пользователей?
- Является ли программное обеспечение частью безопасного дистрибутива Linux и обновлены ли все патчи безопасности?
- Не была ли система взломана или повреждена?

В этой главе для начала мы изучим основные темы безопасности Linux. В последующих главах более подробно описываются современные механизмы обеспечения безопасности.

Физическая безопасность системы

Первая линия обороны — это замок на двери серверной. Хотя закрыть его очень просто, это часто игнорируют. Доступ к физическому серверу означает доступ ко всем содержащимся на нем данным. Никакое программное обеспечение не может

полностью защитить ваши системы, если кто-то посторонний имеет физический доступ к серверу Linux.

Базовая физическая безопасность серверной включает в себя следующие элементы.

- Замок или охранную сигнализацию на двери серверной.
- Средства контроля доступа, разрешающие только авторизованный доступ и определяющие, кто входил в комнату и когда это произошло, например систему ввода ключ-карт.
- Табличку на двери «Посторонним вход воспрещен».
- Правила, определяющие, кто и когда может получить доступ в комнату, для таких групп, например, как уборщики, администраторы сервера и др.

Физическая безопасность системы включает в себя контроль окружающей среды. Необходимо установить системы пожаротушения и обеспечить надлежащую вентиляцию серверной.

Аварийное восстановление

План аварийного восстановления должен содержать ответы на следующие вопросы.

- Какие данные должны быть включены в резервные копии?
- Где должны храниться резервные копии?
- Как долго сохраняются резервные копии?
- Как резервные носители заменяются в хранилище?

Резервные копии данных, носителей и программного обеспечения должны быть включены в контрольный список Access Control Matrix (матрица контроля доступа).

ВНИМАНИЕ!

Важно определить, сколько резервных копий каждого объекта будет поддерживаться. Вам могут понадобиться только три резервные копии одного конкретного объекта, а другого, более важного, — большее количество копий.

Утилиты резервного копирования в системе Linux:

- `amanda` (Advanced Maryland Automatic Network Disk Archiver, продвинутый автоматический сетевой дисковый архиватор из Университета Мэриленда);
- `cpio`;
- `dump/restore`;
- `tar`;
- `rsync`.

Утилиты `cpio`, `dump/restore` и `tar` обычно предустановлены в дистрибутивах Linux. Простой, но эффективный инструмент для резервного копирования данных по сетям — это утилита `rsync`. С ее помощью можно настроить задание для хранения копий всех данных в выбранных каталогах или зеркальных точных копий каталогов на удаленных компьютерах.

Из упомянутых инструментов только `amanda` не устанавливается по умолчанию. Тем не менее утилита `amanda` чрезвычайно популярна, потому что она очень гибкая и может создавать резервные копии даже системы Windows. Если вам нужна дополнительная информация о программе резервного копирования `amanda`, см. сайт `amanda.org`. В конечном счете выбранная вами утилита резервного копирования должна отвечать конкретным требованиям безопасности, принятым в вашей организации.

Защита учетных записей пользователей

Учетные записи пользователей являются частью процесса аутентификации, позволяющего им входить в систему Linux. Правильное управление учетными записями пользователей повышает безопасность системы. Настройка учетных записей рассматривалась в главе 11 «Управление учетными записями». Однако для повышения безопасности с помощью управления учетными записями необходимо ввести несколько дополнительных правил.

- Один пользователь имеет одну учетную запись пользователя.
- Доступ к учетной записи суперпользователя должен быть ограничен.
- Требуется определять даты истечения срока действия временных учетных записей.
- Неиспользуемые учетные записи следует удалять.

Один пользователь на одну учетную запись

Учетные записи следует контролировать. То есть несколько человек не должны входить через одну учетную запись. Когда несколько человек совместно применяют одну учетную запись, нет возможности доказать, кто из них выполнил определенное действие.

Ограниченный доступ к учетной записи суперпользователя

Если несколько человек могут войти в учетную запись суперпользователя, снова появляется проблема. Вы не можете отслеживать, кто конкретно задействует эту учетную запись. Чтобы отследить, кто это делает, необходимо добавить правила применения команды `sudo` (см. главу 8 «Системное администрирование») вместо входа в учетную запись напрямую.

Вместо того чтобы предоставлять нескольким пользователям root-права в системе Linux, вы можете предоставить их с помощью команды `sudo`. Команда `sudo` дает следующие преимущества безопасности.

- Не обязательно передавать пароль суперпользователя.
- Можно точно настроить доступ к командам.
- Все попытки применения (в том числе неудачные) команды `sudo` (кто, что, когда) записываются в файл `/var/log/secure`. Последние системы Linux хранят все такие попытки в журнале `systemd` (используйте команду `journalctl -f`, чтобы наблюдать за попытками доступа через `sudo` в режиме реального времени вместе с другими системными сообщениями).
- После предоставления кому-то прав `sudo` вы можете попытаться ограничить корневой доступ к определенным командам в файле `/etc/sudoers` с помощью команды `visudo`. Однако после предоставления пользователю root-прав, даже ограниченных, трудно быть уверенными в том, что он не сможет найти способы получить полный root-доступ к вашей системе и делать с ней то, что хочет.

Один из способов держать такого пользователя в узде — отправка сообщений безопасности, предназначенных для файла `/var/log/secure`, на удаленный сервер журналов, к которому ни один из локальных администраторов не имеет доступа. Таким образом, любое злоупотребление привилегиями суперпользователя привязывается к конкретному пользователю и записывается так, что он не сможет скрыть свои следы.

Срок действия временных учетных записей

Если у вас есть консультанты, стажеры или временные сотрудники, которым нужен доступ к системам Linux, важно настроить их учетные записи пользователей с указанием срока действия. Это гарантия на случай, если вы забудете удалить их учетные записи, когда им больше не понадобится доступ к системам организации.

Чтобы создать учетную запись пользователя с датой срока действия, примените команду `usermod` в формате *`usermod –e гггг-мм-дд имя_пользователя`*. В примере далее срок действия учетной записи `tim` истекает 1 января 2021 года:

```
# usermod -e 2021-01-01 tim
```

Чтобы убедиться, что срок действия учетной записи установлен правильно, дважды проверьте себя с помощью команды `chage`. Она используется в основном для просмотра и изменения устаревшей информации о пароле учетной записи пользователя, но может получить и доступ к информации об истечении срока действия учетной записи. Параметр `-L` позволяет просматривать различную информацию, к которой команда `chage` имеет доступ. Просто передайте выходные данные команды `chage` в `grep` и найдите слово `Account`. Это действие отобразит только дату срока действия учетной записи пользователя:

```
# chage -l tim | grep Account
Account expires                                 :   Jan   01,   2021
```

Как видно из примера, дата, когда истекает срок действия учетной записи пользователя `tim`, была успешно изменена на 1 января 2021 года.

СОВЕТ

Если вы не задействуете файл /etc/shadow для хранения паролей своей учетной записи, команда chage не сработает. Чаще всего файл /etc/shadow по умолчанию хранит информацию о пароле в большей части систем Linux.

Установите даты истечения срока действия учетной записи для всех временных сотрудников. Кроме того, просматривайте все даты истечения срока действия учетной записи в рамках мониторинга безопасности. Эти действия помогают устранить любые потенциальные лазейки в вашу систему Linux.

Удаление неиспользуемых учетных записей

Хранить просроченные учетные записи опасно для системы. После того как пользователь покинул организацию, лучше всего выполнить ряд шагов, чтобы удалить его учетную запись вместе с данными.

1. Найдите файлы в системе, принадлежащей учетной записи, с помощью команды `find / -user username`.
2. Установите срок действия учетной записи или отключите ее.
3. Сделайте резервную копию файлов.
4. Удалите файлы или передайте их новому владельцу.
5. Удалите учетную запись из системы.

Проблемы возникают, если не выполнить пункт 5 и истекшие или отключенные учетные записи сохранятся в системе. Злоумышленник, получивший доступ к вашей системе, может обновить учетную запись, а затем выдать себя за ее пользователя.

Чтобы найти эти учетные записи, выполните поиск в файле `/etc/shadow`. Дата, обозначающая срок действия учетной записи, находится в восьмом поле каждой записи. Удобно было бы применять формат даты, однако вместо этого в данном поле отображается дата истечения срока действия записи в виде количества дней, прошедших с 1 января 1970 года.

Используйте двухэтапный процесс автоматического поиска просроченных учетных записей в файле `/etc/shadow`. Во-первых, установите переменную оболочки (см. главу 7 «Простейшие скрипты оболочки») с сегодняшней датой в формате «дней с 1 января 1970 года». Затем с помощью команды `gawk` отформатируйте необходимую информацию из файла `/etc/shadow`.

Настройка переменной оболочки с текущей датой в виде числа дней, прошедших с 1 января 1970 года, не представляет особой сложности. Команда `date` может выдавать количество секунд с 1 января 1970 года. Чтобы получить необходимое значение, разделите результат команды `date` на количество секунд в сутках — 86 400.

Далее показано, как настроить переменную оболочки TODAY:

```
# TODAY=$(echo $(($(date --utc --date "$1" +%s)/86400)))
# echo $TODAY
16373
```

Затем учетные записи и даты их истечения извлекаются из файла /etc/shadow с помощью команды gawk. Команда gawk — это версия программы GNU, используемая в UNIX. Вывод команды показан в следующем примере. Как и следовало ожидать, многие учетные записи не имеют срока действия. Однако две записи, Consultant и Intern, показывают дату истечения срока действия в формате «дней с 1 января 1970 года». Обратите внимание на то, что этот шаг вы можете пропустить. Он служит лишь для демонстрации.

```
# gawk -F: '{print $1,$8}' /etc/shadow
...
chrony
tcpdump
johndoe
Consultant 13819
Intern 13911
```

Параметры $1 и $8 в команде gawk представляют поля имени пользователя и даты истечения срока действия в записях файла /etc/shadow. Чтобы проверить даты истечения срока действия этих учетных записей, необходима усовершенствованная версия команды gawk:

```
# gawk -F: '{if (($8 > 0) && ($TODAY > $8)) print $1}' /etc/shadow
Consultant
Intern
```

После команды gawk ($8 > 0) отображаются только записи с истекшим сроком действия. Чтобы убедиться, что эти даты миновали, переменная TODAY сравнивается с полем даты $8. Если TODAY больше, чем дата истечения срока действия учетной записи, она появится в списке. Как видно в предыдущем примере, в системе есть две просроченные учетные записи, которые необходимо удалить.

Это все, что нужно сделать. Настройте переменную TODAY и выполните команду gawk. Все просроченные учетные записи в файле /etc/shadow будут перечислены. Чтобы удалить их, примените команду userdel.

Учетные записи пользователей — это только часть процесса аутентификации, позволяющего им входить в систему Linux. Пароли учетных записей пользователей также играют важную роль.

Защита паролей

Пароли — это основной инструмент безопасности любой современной операционной системы, и, следовательно, их атакуют чаще всего. Естественно, пользователи хотят установить пароль, который легко запомнить, но его легко и угадать.

Для подбора пароля применяется метод грубой силы — перечисление популярных вариантов. Примеры наиболее распространенных паролей:

- 123456;
- Password;
- princess;
- rockyou;
- abc123.

Перейдите в любимую поисковую систему и найдите примеры «общих» паролей. Если вы можете их найти, то и злоумышленники на это способны. Очевидно, что надежные пароли имеют решающее значение для создания безопасной системы.

Создание надежного пароля

В целом пароль не должен быть легко угадываемым, распространенным или популярным и связанным с вами каким-либо образом. Вот несколько правил, которым нужно следовать при выборе пароля.

- Не используйте варианты названия своей учетной записи или полного имени.
- Не применяйте словарные слова.
- Не используйте никаких имен собственных.
- Не используйте свои номер телефона, адрес или имена членов семьи и клички домашних животных.
- Не применяйте названия сайтов.
- Не используйте непрерывную строку букв или цифр на клавиатуре (например, qwerty или asdfg).
- Не используйте ничего из перечисленного с добавлением цифр и знаков препинания в начале или в конце, а также в обратном порядке.

Теперь, когда вы знаете, какие пароли не следует создавать, рассмотрим два основных элемента, определяющих надежный пароль.

1. Длина пароля должна быть 15–25 символов.
2. Пароль должен содержать:
   - строчные буквы;
   - прописные буквы;
   - цифры;
   - специальные символы, например $ % * ( ) - + = , < > : : ” ’.

Пароль из 25 символов считается длинным. При этом чем длиннее пароль, тем он безопаснее. Выбор минимальной длины пароля зависит от потребностей в безопасности.

СОВЕТ

В исследовательском центре Гибсона есть множество материалов о надежных паролях, в том числе статья под названием How big is your haystack... and how well hidden is your needle? («Как спрятать иголку в стоге сена?») на странице grc.com/haystack.html.

Выбрать хороший пароль может быть трудно. Он должен быть достаточно надежным, чтобы нельзя было его угадать, и достаточно легким, чтобы можно было его запомнить. Хороший способ выбрать надежный пароль — взять первую букву каждого слова легко запоминающегося предложения. Обязательно добавляйте цифры, специальные символы и используйте различные регистры. Выбранное вами предложение должно иметь смысл только для вас и не быть общеизвестным. В табл. 22.1 приведены примеры надежных паролей и их запоминания.

Таблица 22.1. Идеи надежных паролей

Пароль	Как запомнить
Mrci7yo!	My rusty car is 7 years old! («Моей ржавой машине 7 лет!»)
2emBp1ib	2 elephants make BAD pets, 1 is better («2 слона ПЛОХИЕ домашние животные, 1 лучше»)
ItMc?Gib	Is that MY coat? Give it back («Это что, Мое пальто? Отдай обратно»)

Пароли выглядят бессмысленными, но на самом деле их довольно легко запомнить. Конечно, не нужно использовать пароли, перечисленные в примере. Они стали общеизвестными и точно были добавлены в словари злоумышленников.

Установка и смена пароля

Вы устанавливаете собственный пароль с помощью команды `passwd`. Введите ее, и она позволит вам изменить пароль. Во-первых, она предложит вам ввести свой старый пароль. Чтобы никто не увидел его, он не отображается при вводе.

Если вы правильно ввели старый пароль, команда `passwd` предложит ввести новый. После ввода новый пароль проверяется с помощью утилиты `cracklib`, чтобы определить, надежен он или нет. Пользователи, не являющиеся суперпользователями, должны ввести другой пароль, если введенный ненадежен.

Суперпользователь — единственный, кому разрешено назначать ненадежные пароли. После того как пароль был принят командой `cracklib`, команда `passwd` просит ввести новый пароль еще раз, чтобы убедиться, что опечаток нет (которые трудно обнаружить, когда вы не видите, что печатаете).

При запуске от имени суперпользователя изменить пароль можно, указав его логин в качестве параметра команды `passwd`, как в примере:

```
# passwd joe
Changing password for user joe.
New UNIX password: ********
Retype new UNIX password: ********
passwd: all authentication tokens updated successfully.
```

Команда `passwd` дважды предложит вам ввести новый пароль для пользователя `joe`. В этом случае она не запрашивает его старый пароль.

Лучшие методы работы с паролями

Теперь вы знаете, как выглядит надежный пароль и как его изменить, но как применить его в своей системе Linux? Можно начать с библиотеки подключаемых модулей PAM. С ее помощью вы точно определите требования, которым должны соответствовать пароли. Например, чтобы убедиться, что пароли должны быть длиной 12 символов, минимум с двумя цифрами, тремя прописными и двумя строчными буквами и отличаться от предыдущих паролей, можете добавить в файл `/etc/pam.d/common-password` или `/etc/pam.d/common-auth` такую строку:

```
password requisite pam_cracklib.so minlen=12, dcredit=2, ucredit=3, lcredit=2,
difok=4
```

Следующий вопрос: как заставить пользователей регулярно менять пароли? Может оказаться утомительно придумывать новые надежные пароли каждые 30 дней! Вот почему необходимы некоторые методы принуждения к этому.

СОВЕТ

Если пользователи испытывают трудности с созданием надежных и уникальных паролей, установите в своей системе Linux утилиту pwgen. Эта утилита для генерации паролей с открытым исходным кодом создает произносимые и запоминающиеся пароли. Можно сделать эти сгенерированные слова отправной точкой для создания паролей учетных записей.

Значения по умолчанию в файле `/etc/login.defs` для новых учетных записей рассматривались в главе 11. В файле `login.defs` есть настройки, влияющие на срок действия и длину пароля:

```
PASS_MAX_DAYS     30
PASS_MIN_DAYS     5PASS_MIN_LEN     16PASS_WARN_AGE     7
```

В примере максимальное количество дней `PASS_MAX_DAYS` до изменения пароля равно 30. Число, которое вы устанавливаете, зависит от конкретной настройки учетной записи. Для организаций, где одну учетную запись применяет один пользователь, значение можно увеличить. Если у вас есть общие учетные записи или несколько человек знают пароль суперпользователя, очень важно регулярно менять пароль. Так вы сможете эффективно обновить список тех, кто его знает.

Чтобы пользователи не меняли свой пароль на новый, а затем сразу же меняли его обратно, вам нужно установить `PASS_MIN_DAYS` на число больше 0. В предыдущем примере самое раннее, когда можно снова изменить пароль, — через 5 дней.

Параметр `PASS_WARN_AGE` — это количество дней, в течение которых пользователь получает предупреждение перед принудительной сменой пароля. Люди, как правило, нуждаются в большом количестве предупреждений и напоминаний, поэтому в предыдущем примере устанавливается время предупреждения 7 дней.

Ранее в этой главе я упоминал, что надежный пароль имеет длину от 15 до 25 символов. С помощью параметра PASS_MIN_LEN вы можете заставить пользователей применять определенное минимальное количество символов в своих паролях. Выбранный параметр должен основываться на жизненных циклах безопасности организации.

ПРИМЕЧАНИЕ

Дистрибутив Ubuntu не имеет параметра PASS_MIN_LEN в своем файле login.defs. Настройка осуществляется программой PAM, которая рассматривается в главе 23 «Продвинутые методы обеспечения безопасности».

Сроком действия пароля для уже созданных учетных записей необходимо управлять с помощью команды chage. Необходимые для этого параметры перечислены в табл. 22.2. Обратите внимание на то, что у команды chage нет параметра длины пароля.

Таблица 22.2. Параметры команды chage

Параметр	Описание
-M	Устанавливает максимальное количество дней до смены пароля. Эквивалентен параметру PASS_MAX_DAYS в файле /etc/login.defs
-m	Устанавливает минимальное количество дней до повторного изменения пароля. Эквивалентен параметру PASS_MIN_DAYS в файле /etc/login.defs
-W	Задает количество дней, в течение которых пользователь получает предупреждение перед принудительной сменой пароля учетной записи. Эквивалентен параметру PASS_WARN_AGE в файле /etc/login.defs

В следующем примере используется команда chage для установки параметров срока действия пароля учетной записи tim. Все три варианта параметров задействуются одновременно:

```
# chage -l tim | grep days
Minimum number of days between password change          : 0
Maximum number of days between password change          : 99999
Number of days of warning before password expires       : 7
# chage -M 30 -m 5 -W 7 tim
# chage -l tim | grep days
Minimum number of days between password change          : 5
Maximum number of days between password change          : 30
Number of days of warning before password expires       : 7
```

Команду chage можно применить и в качестве еще одного метода установки срока действия учетной записи, который основан на истечении срока действия пароля. Ранее для задания истечения срока действия учетной записи использовалась утилита usermod. Командой chage с параметрами -M и -I заблокируйте учетную запись. В следующем примере учетная запись tim просматривается

с помощью chage -l и отображается только информация о настройках пароля пользователя tim:

```
# chage -l tim | grep Password
Password expires                        : never
Password inactive                       : never
```

Здесь видно, что в настройках учетной записи не установлены срок действия пароля (Password expires) и время его бездействия (Password inactive). В следующем примере учетная запись будет заблокирована через 5 дней после истечения срока действия пароля пользователя tim с помощью только параметра -I:

```
# chage -I 5 tim
# chage -l tim | grep Password
Password expires                        : never
Password inactive                       : never
```

Обратите внимание на то, что настройки не изменились! Без установленного срока действия пароля параметр -I ничего не меняет. Таким образом, с помощью параметра -M устанавливается максимальное количество дней до истечения срока действия пароля и установка времени бездействия пароля сохраняется:

```
# chage -M 30 -I 5 tim
# chage -l tim | grep Password
Password expires                        : Mar 03, 2017
Password inactive                       : Mar 08, 2017
```

Теперь учетная запись пользователя tim будет заблокирована через 5 дней после истечения срока действия его пароля. Это полезно в ситуациях, когда сотрудник покинул компанию, но его учетная запись еще не удалена. В зависимости от требований безопасности вашей организации рассмотрите возможность установки блокировки всех учетных записей на определенное количество дней после истечения срока действия паролей.

Файлы и хеши паролей

Ранние системы Linux хранили свои пароли в файле /etc/passwd, они были хешированы. *Хешированный пароль* создается с помощью одностороннего математического процесса. После создания хеша вы не сможете воссоздать исходные символы из хеша. Вот как это работает.

Когда пользователь вводит пароль учетной записи, система Linux переписывает пароль, а затем сравнивает результат хеширования с исходным хешем в файле /etc/passwd. Если они совпадают, пользователь проходит аутентификацию и получает доступ в систему.

Проблема с хранением хешей паролей в файле /etc/passwd связана с настройками безопасности файловой системы (см. главу 4 «Файловая система»). Параметры безопасности файловой системы для файла /etc/passwd перечислены далее:

```
# ls -l /etc/passwd
-rw-r--r--. 1 root root 1644 Feb  2 02:30 /etc/passwd
```

Как видно в примере, каждый может прочитать файл пароля. Кажется, что это не проблема, потому что все пароли хешированы. Однако злоумышленники создали файлы, называемые *радужными таблицами* (rainbow tables). Радужная таблица — это словарь возможных хешированных паролей. Например, радужная таблица содержит хеш популярного пароля Password, который выглядит следующим образом:

```
$6$dhN5ZMUj$CNghjYIteau5xl8yX.f6PTOpendJwTOcXjlTDQUQZhhyV8hKzQ6Hxx6Egj8P3VsHJ8
Qrkv.VSR5dxcK3QhyMc.
```

Из-за простоты доступа к хешам паролей в файле `/etc/passwd` то, когда хешированный пароль будет сопоставлен с радужной таблицей, — лишь вопрос времени.

ПРИМЕЧАНИЕ

Эксперты по безопасности скажут вам, что пароли не просто хешируются, но и «солятся». Шифрование хеша солью означает, что случайно сгенерированное значение добавляется к исходному паролю перед его хешированием. Это еще больше затрудняет сопоставление хешированного пароля с исходным. Однако в Linux хеш-соль хранится вместе с хешированными паролями. Таким образом, доступ на чтение к файлу /etc/passwd означает, что у вас есть хеш-значение и его соль.

Хешированные пароли были перенесены в новый файл конфигурации `/etc/shadow` много лет назад. Этот файл имеет следующие параметры безопасности:

```
# ls -l /etc/shadow
----------. 1 root root 1049 Feb 2 09:45 /etc/shadow
```

Несмотря на отсутствие прав, просматривать этот файл может только суперпользователь. Таким образом, хешированные пароли защищены. В примере хвостовой части файла `/etc/shadow` видно, что в записи каждого пользователя есть длинные бессмысленные строки символов. Это и есть хешированные пароли:

```
# tail -2 /etc/shadow
johndoe:$6$jJjdRN9/qELmb8xWM1LgOYGhEIxc/:15364:0:99999:7:::
Tim:$6$z760AJ42$QXdhFyndpbVPVM5oVtNHs4B/:15372:5:30:7:16436::
```

ВНИМАНИЕ!

Вы можете начать пользоваться системой Linux, в которой все еще действует старый метод хранения хешированных паролей в файле /etc/passwd. Это легко исправить. Просто примените команду pwconv, и файл /etc/shadow будет создан, а хешированные пароли перемещены в него.

Кроме имени учетной записи и хешированного пароля в файле `/etc/shadow` хранятся следующие данные:

- количество дней (с 1 января 1970 года) с момента изменения пароля;
- количество дней до изменения пароля;
- количество дней до обязательного изменения пароля;
- количество дней перед предупреждением пользователя об обязательном изменении пароля;

- количество дней после истечения срока действия пароля, в течение которых учетная запись будет отключена;
- количество дней (с 1 января 1970 года), в течение которых учетная запись была отключена.

Звучит знакомо, не правда ли? Это и есть настройки срока действия пароля, рассмотренные ранее в этой главе. Помните, что команда `chage` не сработает, если у вас нет настроенного файла `/etc/shadow` или файл `etc/login.defs` не поддерживается.

Очевидно, что параметры безопасности файловой системы очень важны для обеспечения безопасности всей системы Linux. Это особенно верно для файлов конфигурации всех систем Linux.

Защита файловой системы

Еще одна важная часть защиты вашей системы Linux — обеспечение безопасности файловой системы. Основы настройки безопасности были рассмотрены в главе 4, а списки контроля доступа (ACL) — в главе 11. Однако есть несколько дополнительных моментов, которые необходимо изучить.

Управление опасными правами файловой системы

Представьте, какой хаос возникнет, если предоставить полный доступ `rwxrwxrwx` (`777`) к каждому файлу в системе Linux. Во многом это может произойти из-за отсутствия четкого управления правами set UID (SUID) и set GID (SGID) (см. главу 4 «Файловая система» и главу 11 «Управление учетными записями»).

Файлы с правами SUID в категории `Owner` и правами на выполнение `execute` в категории `Other` позволяют любому пользователю временно стать владельцем файла, пока он выполняется. Хуже всего, если файлом владеет суперпользователь.

Аналогично файлы с правами SGID в категории `Group` и правами `execute` в категории `Other` позволяют любому пользователю временно стать членом группы файла, пока он выполняется. SGID также может быть настроен для каталогов, что устанавливает идентификатор группы любых файлов, созданных в каталоге, в идентификатор группы каталога.

Исполняемые файлы с SUID или SGID чаще всего становятся мишенью злоумышленников. Таким образом, лучше всего задействовать их по минимуму. Однако некоторые файлы должны сохранять эти настройки. Например, команды `passwd` и `sudo`, которые следуют за ними. Каждый из этих файлов должен поддерживать свои права SUID:

```
$ ls -l /usr/bin/passwd
-rwsr-xr-x. 1 root root 28804 Aug 17 20:50 /usr/bin/passwd
$ ls -l /usr/bin/sudo
---s--x--x. 2 root root 77364 Nov 3 08:10 /usr/bin/sudo
```

Такие команды, как `passwd` и `sudo`, предназначены для применения в качестве SUID-программ. Несмотря на то что они выполняются от имени суперпользователя, как обычный пользователь вы можете изменить собственный пароль только с помощью команды `passwd` и перейти к правам суперпользователя только с помощью команды `sudo`, если вам были даны соответствующие права в файле `/etc/sudoers`. Наиболее опасно, если хакер применяет команду `SUID bash`: любой, кто запускает ее, может изменить все в системе, имеющей корневой доступ.

С помощью команды `find` вы можете выполнить поиск в своей системе, чтобы увидеть скрытые или иным образом выделяющиеся команды SUID и SGID, например:

```
# find / -perm /6000 -ls
4597316 52 -rwxr-sr-x 1 root games 51952 Dec 21 2013 /usr/bin/atc
4589119 20 -rwxr-sr-x 1 root tty   19552 Nov 18 2013 /usr/bin/write
4587931 60 -rwsr-xr-x 1 root root  57888 Aug  2 2013 /usr/bin/at
4588045 60 -rwsr-xr-x 1 root root  57536 Sep 25 2013 /usr/bin/crontab
4588961 32 -rwsr-xr-x 1 root roo t 32024 Nov 18 2013 /usr/bin/su
...
5767487 85 -rwsrwsr-x 1 root root  68928 Sep 13 11:52 /var/.bin/myvi
...
```

Обратите внимание на то, что команда `find` раскрывает команды SUID и SGID, которые обычные пользователи могут запускать для добавления дополнительных прав по определенным причинам. В этом примере также есть файл, который пользователь пытался скрыть (`myvi`). Это копия команды `vi`, которая из-за разрешений и права собственности может изменять файлы, принадлежащие суперпользователю. И это, очевидно, то, что пользователь не должен делать.

Защита файлов с паролями

Файл `/etc/passwd` — это файл, который система Linux применяет для проверки информации об учетной записи пользователя и который мы рассмотрели ранее в этой главе. Файл `/etc/passwd` должен иметь следующие настройки прав:

- Owner: `root`;
- Group: `root`;
- Permissions: (644) Owner: `rw-` Group: `r--` Other: `r--`.

В следующем примере показано, что у файла `/etc/passwd` имеются соответствующие настройки:

```
# ls -l /etc/passwd
-rw-r--r--. 1 root root 1644 Feb  2 02:30 /etc/passwd
```

Эти настройки необходимы для того, чтобы пользователи могли войти в систему и увидеть имена, связанные с ID пользователей и групп. Однако они не должны

иметь возможности напрямую изменять файл /etc/passwd. Например, злоумышленник может добавить новую учетную запись в файл, если доступ для записи был предоставлен пользователю Other.

Следующий файл — это /etc/shadow. Конечно, он тесно связан с файлом /etc/passwd, поскольку также задействуется в процессе аутентификации входа в систему. Файл /etc/shadow должен иметь следующие настройки прав:

- Owner: root;
- Group: root;
- Permissions: (000) Owner: --- Group: --- Other: ---.

Приведенный далее код показывает, что файл /etc/shadow имеет соответствующие настройки:

```
# ls -l /etc/shadow
----------. 1 root root 1049 Feb      2 09:45 /etc/shadow
```

Файл /etc/passwd имеет права на чтение для владельца, группы и других пользователей. Обратите внимание на то, как сильно ограничен в правах файл /etc/shadow по сравнению с файлом /etc/passwd. Для файла /etc/shadow нет прав на доступ, хотя суперпользователь все еще может получить доступ к нему. Итак, если только суперпользователь может просматривать этот файл, как пользователи могут менять свои пароли, которые хранятся в файле /etc/shadow? Команда passwd в файле /usr/bin/passwd задействует специальный SUID-параметр прав. Он показан далее:

```
# ls -l /usr/bin/passwd
-rwsr-xr-x. 1 root root 28804 Aug 17 20:50 /usr/bin/passwd
```

Таким образом, пользователь, выполняющий команду passwd, на время ее выполнения становится суперпользователем, а затем может записать ее в файл /etc/shadow, но только для того, чтобы изменить собственную информацию о пароле.

ПРИМЕЧАНИЕ

Суперпользователь не имеет доступа к изменению разрешения каталога /etc/shadow. Так как же он записывает в файл /etc/shadow? Суперпользователь имеет полный доступ ко всем файлам независимо от того, перечислены права для него или нет.

Файл /etc/group (см. главу 11 «Управление учетными записями») содержит все группы в системе Linux. Его права доступа к файлам должны быть установлены точно так же, как и в файле /etc/passwd:

- Owner: root;
- Group: root;
- Permissions: (644) Owner: rw- Group: r-- Other: r--.

Кроме того, файл пароля группы, `/etc/gshadow`, должен быть хорошо защищен. Как и следовало ожидать, права файла должны быть установлены точно такими же, как и для файла `/etc/shadow`:

- Owner: `root`;
- Group: `root`;
- Permissions: (000) Owner: `---` Group: `---` Other: `---`.

Блокировка файловой системы

Таблица файловых систем (см. главу 12 «Управление дисками и файлами») в файле `/etc/fstab` также требует особого внимания. Файл `/etc/fstab` используется во время загрузки для монтирования устройств хранения данных в файловых системах. Он также применяется командами `mount`, `dump` и `fsck`. Файл `/etc/fstab` должен иметь следующие настройки прав:

- Owner: `root`;
- Group: `root`;
- Permissions: (644) Owner: `rw-` Group: `r--` Other: `r--`.

В таблице файловой системы есть важные параметры безопасности, которые необходимо изучить. Помимо тех, что касаются корневых, загрузочных разделов и разделов подкачки, остальные параметры файловой системы по умолчанию достаточно безопасны. Однако вы можете рассмотреть следующее.

- Как правило, вы помещаете подкаталог `/home`, где находятся пользовательские каталоги, в свой собственный раздел. Добавляя параметры к команде `mount` для монтирования этого каталога в `/etc/fstab`, можете применять параметр `nosuid`, чтобы запретить запуск исполняемых программ с разрешениями SUID и SGID оттуда. Программы, которые нуждаются в правах SUID и SGID, не должны храниться в каталоге `/home` и, скорее всего, являются вредоносными. Можно установить параметр `nodev` так, чтобы ни один файл устройства, расположенный там, не был распознан. Файлы устройств должны храниться в файле `/dev`, а не в `/home`.

  Установите параметр `noexec` так, чтобы никакие исполняемые программы, хранящиеся в `/home`, нельзя было запустить.
- Вы можете поместить подкаталог `/tmp`, где находятся временные файлы, в свой собственный раздел и использовать те же параметры настройки, что и для `/home`:
  - `Nosuid`;
  - `Nodev`;
  - `Noexec`.
- Можете поместить подкаталог `/usr`, где находятся пользовательские программы и данные, в собственный раздел и установить параметр `nodev` так, чтобы ни один

файл устройства, расположенный там, не был распознан. После установки программного обеспечения каталог `/usr` практически не изменяется (иногда по соображениям безопасности он даже монтируется только для чтения).

- Если система настроена как сервер, вы, вероятно, захотите поместить каталог `/var` в собственный раздел. Каталог `/var` должен увеличиваться по мере добавления сообщений журнала и содержимого для веб-, FTP- и других серверов. Для раздела `/var` вы можете использовать те же параметры монтирования, что и для раздела `/home`:
  - `Nosuid;`
  - `Nodev;`
  - `Noexec;`

Если добавить эти параметры для команды `mount` в файл `/etc/fstab`, он будет выглядеть следующим образом:

```
/dev/sdb1     /home    ext4     defaults,nodev,noexec,nosuid   1 2
/dev/sdc1     /tmp     ext4     defaults,nodev,noexec,nosuid   1 1
/dev/sdb2     /usr     ext4     defaults,nodev                 1 2
/dev/sdb3     /var     ext4     defaults,nodev,noexec,nosuid   1 2
```

Эти параметры монтирования помогут еще лучше заблокировать вашу файловую систему и добавить еще один уровень защиты от злоумышленников. И вновь отмечу, что управление различными правами файлов и параметрами команды `fstab` должно быть частью вашей политики безопасности. Элементы безопасности должны определяться потребностями организации.

Управление программным обеспечением и службами

Администратор сосредоточивается на том, чтобы убедиться, что необходимые программное обеспечение и службы в системе Linux имеются. С точки зрения безопасности нужно убедиться, что, наоборот, ненужных программного обеспечения и служб в Linux нет.

Обновление пакетов программного обеспечения

Как и удаление ненужных служб и программного обеспечения, обновление программного обеспечения имеет решающее значение для безопасности. Различные ошибки, в том числе ошибки безопасности, исправляются именно с помощью обновлений программного обеспечения. Обновления пакетов программного обеспечения рассматривались в главе 9 «Установка Linux» и в главе 10 «Управление программами».

Обновлять программное обеспечение следует регулярно. Как часто и когда вы это делаете, зависит от потребностей вашей организации в безопасности.

Можно легко автоматизировать обновление программного обеспечения, но как и удаление служб и программного обеспечения, было бы разумно сначала протестировать обновления в отдельной системе. Если обновленное программное обеспечение исправно, можете обновить его и в своих системах Linux.

Советы о безопасности

Поскольку недостатки безопасности есть и в программном обеспечении Linux, проект Common Vulnerabilities and Exposures (CVE) отслеживает и помогает быстро исправить их, над чем работает все сообщество Linux.

Такие компании, как Red Hat, предоставляют обновленные пакеты для устранения недостатков безопасности — они называются *эрратами* (errata). Эрраты могут состоять из одного или нескольких обновленных пакетов. Если вы используете дистрибутив Red Hat Enterprise Linux, необходимо найти пакеты RPM (RPM Package Manager), связанные с конкретными базами CVE и эрратами.

По мере появления новых форм упаковки программного обеспечения необходимо убедиться, что программное обеспечение в этих пакетах проверяется на наличие уязвимостей. Например, каталог контейнеров Red Hat (access.redhat.com/containers) перечисляет поддерживаемые Red Hat контейнерные образы вместе с соответствующими эрратами и индексами работоспособности для каждого образа.

Дополнительные сведения о том, как обрабатываются обновления безопасности в Red Hat Enterprise Linux, см. на странице Security Updates портала Red Hat (access.redhat.com/security/updates/). Сайт содержит огромное количество информации, связанной с уязвимостями безопасности и тем, как они обрабатываются. Возможность получать своевременные обновления систем безопасности — одна из основных причин, по которым компании покупают подписку на дистрибутив Red Hat Enterprise Linux.

Расширенная настройка безопасности

Планируя расширение настроек безопасности, необходимо узнать о других важных темах, относящихся к безопасности, — криптографии, подключаемых модулях аутентификации (PAM) и SELinux. Эти темы будут рассмотрены в главах 23 и 24.

Мониторинг системы

Если спланировать и обеспечить безопасность своей системы, большинство вредоносных атак будут остановлены. Но если атака все-таки произошла, необходимо уметь распознать ее. Мониторинг системы — это то, что должно происходить непрерывно.

Мониторинг системы включает в себя наблюдение за файлами журналов, учетными записями пользователей и самой файловой системой. Кроме того, нужны инструменты, которые помогут обнаружить атаки и другие типы вредоносных программ.

Мониторинг файлов журналов

Понимание того, как ведется журнал сообщений, имеет решающее значение для поддержания системы Linux и устранения в ней неполадок. До того как функция `systemd` начала использоваться для сбора сообщений в так называемый журнал `systemd`, сообщения, генерируемые ядром и системными службами, направлялись в файл в каталоге `/var/log`. Хотя это в значительной степени происходит и для `systemd`, теперь вы можете просматривать сообщения журнала непосредственно из журнала `systemd` с помощью команды `journalctl`.

Файлы журналов системы Linux в основном находятся в каталоге `/var/log`. Большинство файлов в каталоге `/var/log` направляются туда из журнала `systemd` через службу `rsyslogd` (см. главу 13 «Администрирование серверов»). Список файлов `/var/log` и краткое описание каждого из них содержатся в табл. 22.3.

Таблица 22.3. Файлы журналов в каталоге /var/log

Имя журнала	Имя файла	Описание
Apache Access Log	/var/log/httpd/access_log	Регистрирует запросы на получение информации с вашего веб-сервера Apache
Apache Error Log	/var/log/httpd/error_log	Регистрирует ошибки, возникшие у клиентов, пытающихся получить доступ к данным на вашем веб-сервере Apache
Bad Logins Log	btmp	Регистрирует неудачные попытки входа в систему
Boot Log	boot.log	Содержит сообщения, указывающие, какие системные службы были успешно запущены и завершены, а какие (если таковые имеются) не запустились или были остановлены. Самые последние загрузочные сообщения перечислены в конце файла
Kernel Log	dmesg	Записывает сообщения ядра при загрузке системы
Cron Log	cron	Содержит сообщения о состоянии демона crond
dpkg Log	dpkg.log	Содержит информацию об установленных пакетах Debian
FTP Log	vsftpd.log	Содержит сообщения, относящиеся к передачам, выполняемым с помощью демона vsftpd (FTP-сервер)

Имя журнала	Имя файла	Описание
FTP Transfer Log	xferlog	Содержит информацию о файлах, передаваемых с помощью службы FTP
GNOME Display Manager Log	/var/log/gdm/:0.log	Содержит сообщения, связанные с экраном входа в систему (GNOME display manager). Да, в имени файла действительно есть двоеточие
LastLog	lastlog	Записывает последний вход учетной записи в систему
Login/out Log	wtmp	Содержит историю входов в систему и выходов из нее
Mail Log	maillog	Содержит информацию об адресах, на которые и с которых было отправлено электронное письмо. Полезно для обнаружения спама
MySQL Server Log	mysqld.log	Включает в себя информацию, связанную с деятельностью сервера баз данных MySQL (mysqld)
News Log	spooler	Предоставляет каталог, содержащий журналы сообщений с сервера новостей Usenet, если он запущен
Samba Log	/var/log/samba/smbd.log/ var/log/samba/nmbd.log	Показывает сообщения от демона файловой службы Samba SMB
Security Log	secure	Записывает дату, время и продолжительность попыток входа в систему и сеансов
Sendmail Log	sendmail	Показывает сообщения об ошибках, записанные демоном sendmail
Squid Log	/var/log/squid/access.log	Содержит сообщения, связанные с прокси/кэш-сервером squid
System Log	messages	Предоставляет файл журнала общего назначения, в который многие программы записывают сообщения
UUCP Log	uucp	Показывает сообщения о состоянии от демона UNIX Copy Protocol
YUM Log	yum.log	Показывает сообщения, связанные с программными пакетами RPM
X.Org X11 Log	Xorg.0.log	Показывает сообщения, выводимые сервером X.Org X

Файлы журналов, находящиеся в каталоге `/var/log` системы, зависят от того, какие службы вы задействуете. Кроме того, некоторые файлы журналов зависят от дистрибутива. Например, если используется дистрибутив Fedora, то не будет файла журнала `dpkg`.

Большинство файлов журнала отображаются с помощью команд `cat`, `head`, `tail`, `more` или `less`. Однако некоторые из них нуждаются в специальных командах для просмотра (табл. 22.4).

Таблица 22.4. Файлы журнала, требующие специальных команд

Имя файла	Команда просмотра
btmp	dump-utmp btmp
dmesg	dmesg
lastlog	lastlog
wtmp	dump-utmp wtmp

С переходом Fedora, RHEL, Ubuntu и других дистрибутивов Linux на `systemd`, который управляет процессом загрузки и службами, как отмечалось ранее, изменился механизм сбора и отображения сообщений журнала, связанных с ядром и системными службами. Эти сообщения направляются в журнал `systemd` и могут быть отображены с помощью команды `journalctl`.

Вы можете просматривать сообщения журнала непосредственно из журнала `systemd`, а не просто перечислять содержимое файлов `/var/log`. На самом деле в последней версии Fedora просто нет файла `/var/log/messages`, в который многие службы по умолчанию направляют сообщения журнала. Вместо этого вы можете использовать команду `journalctl` для отображения сообщений журнала различными способами. Чтобы просмотреть сообщения ядра, введите следующую команду:

```
# journalctl -k
Logs begin at Sun 2019-06-09 18:59:23 EDT, end at
    Sun 2019-10-20 18:11:06 EDT.
Oct 19 11:43:04 localhost.localdomain kernel:
    Linux version 5.0.9-301.fc30.x86_64
    (mockbuild@bkernel04.phx2.fedoraproject.org)
    (gcc version 9.0.1 20190312 (Red Hat 9.0.1-0.10) (GCC))
        #1 SMP Tue Apr 23 23:57:35 UTC 2019
Oct 19 11:43:04 localhost.localdomain kernel: Command line:
        BOOT_IMAGE=(hd0,msdos1)/vmlinuz-5.0.9-301.fc30.x86_64
    root=/dev/mapper/fedora_localhost--live-root ro
    resume=/dev/mapper/fedora_localhost--live-swap
    rd.lvm.lv=fedora_localhost-live/root
    rd.lvm.lv=fedora_localhost-live/swap rhgb quiet
...
```

Чтобы просмотреть сообщения, связанные с определенной службой, используйте параметр `-u`, после которого указано имя службы, как в этом примере:

```
# journalctl -u NetworkManager.service
# journalctl -u httpd.service
# journalctl -u avahi-daemon.service
```

Если вы считаете, что безопасность под угрозой, можете просмотреть все или определенные сообщения по мере их поступления в режиме реального времени. Например, чтобы следить за сообщениями ядра или службы `httpd` по мере их поступления, добавьте параметр `-f` (когда закончите, нажмите сочетание клавиш Ctrl+C):

```
# journalctl -k -f
# journalctl -f -u NetworkManager.service
```

Чтобы проверить только загрузочные сообщения, можете перечислить ID загрузки для всех системных загрузок, а затем загрузить интересующий вас экземпляр загрузки. В следующих примерах показаны ID загрузки и сообщения для выбранного ID загрузки:

```
# journalctl --list-boots
-3 6b968e820df345a781cb6935d483374c
   Sun 2019-08-25 12:42:08 EDT—Mon 2019-08-26 14:30:53 EDT
-2 f2c5a74fbe9b4cb1ae1c06ac1c24e89b
   Mon 2019-09-02 15:49:03 EDT—Thu 2019-09-12 13:08:26 EDT
-1 5d26bee1cfb7481a9e4da3dd7f8a80a0
   Sun 2019-10-13 12:30:27 EDT—Thu 2019-10-17 13:37:22 EDT
0 c848e7442932488d91a3a467e8d92fcf
   Sat 2019-10-19 11:43:04 EDT—Sun 2019-10-20 18:11:06 EDT
# journalctl -b c848e7442932488d91a3a467e8d92fcf
-- Logs begin at Sun 2019-06-09 18:59:23 EDT,
   end at Sun 2019-10-20 18:21:18 EDT. --
Oct 19 11:43:04 localhost.localdomain kernel: Linux version
5.0.9-301.fc30.x86_64 (mockbuild@bkernel04.phx2.fedoraproject.org)
...
Oct 19 11:43:04 localhost.localdomain kernel: Command line:
   BOOT_IMAGE=(hd0,msdos1)/vmlinuz-5.0.9-301.fc30.x86_64
   root=/dev/mapper/fedora_local>
...
Oct 19 11:43:04 localhost.localdomain kernel:
   DMI: Red Hat KVM, BIOS 1.9.1-5.el7_3.3 04/01/2014
Oct 19 11:43:04 localhost.localdomain kernel: Hypervisor detected: KVM
```

Мониторинг учетных записей пользователей

Учетные записи пользователей часто задействуются в атаках на систему путем получения несанкционированного доступа к текущей учетной записи, создания фиктивных учетных записей или лазеек в учетной записи для последующего доступа. Чтобы избежать таких проблем, необходимо наблюдать за учетными записями пользователей.

Поиск поддельных учетных записей и прав

Учетные записи, созданные без прохождения соответствующей авторизации, считаются поддельными. Кроме того, изменение учетной записи любым способом, дающим ей другой номер идентификации пользователя (UID) или иное членство

в группах, также является превышением прав. Наблюдая за файлами `/etc/passwd` и `/etc/group`, можно отслеживать эти потенциальные нарушения.

Для контроля файлов `/etc/passwd` и `/etc/group` вы можете применить демон аудита. Это чрезвычайно мощный инструмент, который позволяет выбирать системные события, чтобы отслеживать их и записывать, а также создает отчеты.

Чтобы начать аудит файлов `/etc/passwd` и `/etc/group`, нужно использовать команду `auditctl`. Для запуска этого процесса требуется как минимум два варианта команд:

- `-w filename`. Установите нужное имя файла на `filename`. Демон аудита отслеживает файл по его номеру индексных узлов. *Номер индексного узла* — это структура данных, содержащая информацию о файле, включая его местоположение;
- `-p trigger(s–)`. Если реализуется один из типов доступа (`=read`, `=write`, `=execute`, `=attribute`) к *имени файла*, то запись аудита начинается.

В следующем примере наблюдение было направлено в файл `/etc/passwd` с помощью команды `auditctl`. Демон аудита будет контролировать доступ любых действий: чтения, записи или изменения атрибутов файлов:

```
# auditctl -w /etc/passwd -p rwa
```

После запуска аудита файлов вы можете отключить его в любой момент. Чтобы отключить аудит, используйте команду

```
# auditctl -W имя_файла -p trigger(s)
```

Чтобы просмотреть список текущих проверяемых файлов и настройки их наблюдения, введите `auditctl -l` в командной строке.

Чтобы просмотреть журналы, примените команду `ausearch` демона аудита. Единственный параметр, необходимый здесь, — это `-f`, который указывает, какие записи нужно просмотреть из журнала. Далее приведен пример данных аудита `/etc/passwd`:

```
# ausearch -f /etc/passwd
time->Fri Feb  7 04:27:01 2020
type=PATH msg=audit(1328261221.365:572):
item=0 name="/etc/passwd" inode=170549
dev=fd:01 mode=0100644 ouid=0 ogid=0
rdev=00:00 obj=system_u:object_r:etc_t:s0
type=CWD msg=audit(1328261221.365:572):  cwd="/"
...
time->Fri Feb 7 04:27:14 2020
type=PATH msg=audit(1328261234.558:574):
item=0 name="/etc/passwd" inode=170549
dev=fd:01 mode=0100644 ouid=0 ogid=0
rdev=00:00 obj=system_u:object_r:etc_t:s0
type=CWD msg=audit(1328261234.558:574):
cwd="/home/johndoe"
```

```
type=SYSCALL msg=audit(1328261234.558:574):
arch=40000003 syscall=5 success=yes exit=3
a0=3b22d9 a1=80000 a2=1b6 a3=0 items=1 ppid=3891
pid=21696 auid=1000 uid=1000 gid=1000 euid=1000
suid=1000 fsuid=1000 egid=1000 sgid=1000 fsgid=1000
tty=pts1 ses=2 comm="vi" exe="/bin/vi"
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023"
----
```

Этой информации достаточно для полного обзора файла. Рассмотрим, что значат те или иные события из примера:

- `time` — отметка времени действия;
- `name` — имя файла, за которым ведется наблюдение, — `/etc/passwd`;
- `inode` — номер индексного узла `/etc/passwd` в этой файловой системе;
- `uid` — ID пользователя, выполняющего программу, — `1000`;
- `exe` — программа `/bin/vi`, применяемая в файле `/etc/passwd`.

Чтобы определить, какой учетной записи пользователя присвоен UID `1000`, посмотрите файл `/etc/passwrd`. В примере UID `1000` принадлежит пользователю `johndoe`. Таким образом, из приведенной записи аудита вы можете определить, что учетная запись `johndoe` пыталась использовать редактор `vi` в файле `/etc/passwd`. Сомнительно, чтобы это было сделано случайно, так что требуется более тщательное расследование.

ПРИМЕЧАНИЕ

Команда ausearch ничего не выводит, если в файле не было запущено никаких событий.

Демон аудита и связанные с ним инструменты чрезвычайно многогранны. Чтобы узнать о них больше, посмотрите справочные страницы для следующих утилит демона аудита и файлов конфигурации:

- `auditd` — демон аудита;
- `auditd.conf` — файл конфигурации демона;
- `autditctl` — контролирует систему аудита;
- `audit.rule` — правила конфигурации загружаются при загрузке;
- `ausearch` — поиск указанных элементов в журналах аудита;
- `aureport` — создает отчеты журналов аудита;
- `audispd` — отправляет данные аудита другим программам.

Демон аудита — это один из способов следить за важными файлами. Вы также должны регулярно просматривать свои учетные записи и групповые файлы вручную, чтобы отслеживать необычные события.

Важные файлы, такие как `/etc/passwd`, необходимо отслеживать на предмет создания неавторизованной учетной записи. Кроме того, плохо, если учетная запись авторизованная, но у нее неподходящий пароль.

Отслеживание учетных записей с неподходящими паролями

Какие бы усилия вы ни приложили, «плохие» пароли будут проскальзывать. Поэтому действительно нужно следить за паролями учетных записей пользователей, чтобы убедиться, что они достаточно надежны и могут противостоять атаке.

Один из инструментов мониторинга надежности паролей — это тот же инструмент, который злоумышленники применяют для взлома учетных записей, — John the Ripper. Это бесплатный инструмент с открытым исходным кодом, который вы можете использовать в командной строке Linux. Он не установлен по умолчанию. Для дистрибутива Fedora нужно выполнить команду `yum install john`, чтобы установить его.

СОВЕТ

Чтобы установить John the Ripper в дистрибутиве Ubuntu, используйте команду sudo apt-get install john.

Чтобы задействовать инструмент John the Ripper для проверки паролей пользователей, необходимо сначала извлечь имена учетных записей и пароли с помощью команды `unshadow`. Эта информация должна быть перенаправлена в файл для применения командой `john`, как показано далее:

```
# unshadow /etc/passwd /etc/shadow > password.file
```

Теперь отредактируйте `password.file` с помощью любого текстового редактора, чтобы удалить учетные записи без паролей. Поскольку разумно ограничить инструмент John the Ripper тестированием нескольких учетных записей одновременно, удалите все имена учетных записей, которые не нужно тестировать в данный момент.

ВНИМАНИЕ!

Утилита john чрезвычайно интенсивно использует процессор — она устанавливает свой приоритет на значение 19. Однако было бы разумно запустить ее в тестовой системе или в нерабочее время и только для нескольких учетных записей одновременно.

Теперь с помощью команды `john` попытайтесь взломать пароль. Чтобы запустить `john` против созданного файла паролей, выполните команду `john` *`имя_файла`*. В следующем фрагменте кода напротив файла `password.file` показаны выходные данные команды `john`. Для демонстрации в образце файла была оставлена только одна учетная запись. Кроме того, учетной записи `Samantha` был дан неверный па-

роль. Из примера видно, как мало времени потребовалось инструменту John the Ripper, чтобы взломать пароль:

```
# john password.file
Loaded 1 password hash (generic crypt(3) [?/32])
password         (Samantha)
guesses: 1  time: 0:00:00:44 100% (2)  c/s: 20.87
 trying: 12345 – missy
Use the "--show" option to display all of the
 cracked passwords reliably
```

Чтобы продемонстрировать, насколько важны надежные пароли, посмотрим, что происходит, когда пароль учетной записи `amantha` изменяется с `password` на `password1234`. Несмотря на то что `Password1234` все еще является слабым паролем, для его взлома требуется больше семи дней работы процессора. В следующем коде команда `john` наконец закончила попытку взлома:

```
# passwd Samantha
Changing password for user Samantha.
...
# john password.file
Loaded 1 password hash (generic crypt(3) [?/32])
...
time: 0:07:21:55 (3)  c/s: 119  trying: tth675 – tth787
Session aborted
```

Как только процесс взлома паролей будет завершен, `password.file` должен быть удален из системы. Чтобы узнать больше об инструменте John the Ripper, посетите сайт www.openwall.com/john.

Мониторинг файловой системы

Вредоносные программы часто изменяют файлы. К тому же они могут попытаться скрыть свои следы, выдавая себя за обычные файлы и программы. Однако есть способы раскрыть их с помощью различных методик мониторинга, описанных далее.

Проверка пакетов программного обеспечения

Как правило, если вы устанавливаете пакет программного обеспечения из стандартного репозитория или загружаете пакет с подтвержденного сайта, у вас не будет никаких проблем. Но всегда полезно дважды проверить установленные пакеты, чтобы убедиться, что они не были скомпрометированы. Команда для выполнения этой задачи — `rpm -V` *`имя_пакета`*.

При проверке программного обеспечения информация из установленных файлов пакетов сравнивается с метаданными пакета (см. главу 10 «Управление программами») в базе данных `rpm`. Если никаких проблем не обнаружено, команда

`rpm -V` ничего не выводит. Но если при проверке появляются расхождения, команда выводит кодированный список. В табл. 22.5 приведены используемые коды и описание расхождений.

Таблица 22.5. Расхождения в проверке пакетов

Код	Расхождение
S	Размер файла
M	Права доступа и тип файла
5	Контрольная сумма MD5
B	Основные и второстепенные номера файла устройства
L	Символические ссылки
U	Владелец
G	Группа
T	Время изменения файла (mtime)
P	Другие установленные пакеты, от которых зависит этот пакет (так называемые инструменты)

В приведенном далее неполном списке все установленные пакеты проходят проверку на верификацию. Вы видите, что во вводе появились коды `5`, `S` и `T`, что указывает на потенциальные проблемы:

```
# rpm -qaV
5S.T.....  c /etc/hba.conf
...
...T.....    /lib/modules/3.2.1-3.fc16.i686/modules.devname
...T.....    /lib/modules/3.2.1-3.fc16.i686/modules.softdep
```

Нет необходимости проверять все пакеты сразу. Можете проверить только один пакет за один раз. Например, если вы хотите проверить пакет `nmap`, просто введите команду `rpm-V nmap`.

ПРИМЕЧАНИЕ

Для проверки пакетов Ubuntu вам понадобится утилита debsums. Она не установлена по умолчанию. Для ее установки используйте команду sudo apt-get install debsums. Чтобы проверить все установленные пакеты, используйте команду debsums –a. Если хотите проверить один пакет, введите команду debsums *имя_пакета*.

Сканирование файловой системы

Если вы недавно не обновляли свою систему, двоичные файлы не должны быть изменены. Такие команды, как `find` и `rpm -V`, помогут определить, был ли изменен двоичный файл.

Чтобы проверить наличие изменений двоичного файла, команда `find` использует время его изменения, то есть `mtime`. Файл `mtime` отображает время, когда содержимое файла было изменено в последний раз. Кроме того, команда `find` может контролировать время создания/изменения файла, то есть `ctime`.

Если вы заметили подозрительную активность, быстро просканируйте свою файловую систему, чтобы увидеть, были ли какие-либо двоичные файлы изменены и когда (сегодня или вчера, в зависимости от того, когда, по вашему мнению, произошло вторжение). Чтобы выполнить сканирование, используйте команду `find`.

В следующем примере сканируется каталог `/sbin`. Чтобы узнать, были ли какие-либо двоичные файлы изменены менее 24 часов назад, применяется команда `find /sbin -mtime -1`. В этом примере отображаются несколько файлов, показывающих, что они недавно были изменены. Это говорит о том, что в системе протекает вредоносная деятельность. Чтобы продолжить исследование, просмотрите время каждого отдельного файла, используя команду `stat filename`, как показано далее:


```
# find /sbin -mtime -1
/sbin
/sbin/init
/sbin/reboot
/sbin/halt
#
# stat /sbin/init
  File: '/sbin/init' -> '../bin/systemd'
  Size: 14    Blocks: 0      IO Block: 4096   symbolic link
Device: fd01h/64769d    Inode: 9551        Links: 1
Access: (0777/lrwxrwxrwx)
Uid: (    0/    root)   Gid: (    0/    root)
Context: system_u:object_r:bin_t:s0
Access: 2016-02-03 03:34:57.276589176 -0500
Modify: 2016-02-02 23:40:39.139872288 -0500
Change: 2016-02-02 23:40:39.140872415 -0500
 Birth: -
```

Вы можете создать базу данных всех исходных `mtimes` и `ctimes` двоичного файла, а затем запустить скрипт, чтобы найти текущие `mtimes` и `ctimes`, сравнить их с базой данных и отметить любые расхождения. Однако этот процесс может выполнить специальная программа. Она называется системой обнаружения вторжений (Intrusion Detection System) и будет рассмотрена далее в этой главе.

Необходимо регулярно выполнять и другие варианты сканирования файловой системы. Любимые файлы и настройки файлов злоумышленников перечислены в табл. 22.6. Здесь также приведены команды для выполнения сканирования и названы причины, по которым файл или настройка файла потенциально проблематичны.

Таблица 22.6. Дополнительное сканирование файловой системы

Файл или настройка	Команда сканирования	Проблемы с файлом или настройкой
Права SUID	find / -perm -4000	Позволяет любому пользователю временно стать владельцем файла, пока тот выполняется в памяти
Права SGID	find / -perm -2000	Позволяет любому пользователю временно стать членом группы файла, пока тот выполняется в памяти
Файлы rhost	find /home -name.rhost	Позволяет системе полностью доверять другой системе. Не должен находиться в каталогах /home
Файлы без владельца	find / -nouser	Указывает на файлы, не связанные с каким-либо именем пользователя
Файлы без группы	find / -nogroup	Указывает на файлы, не связанные ни с одним именем группы

Команда `rpm -V` *`пакет`* может сообщить об изменениях, произошедших в файле после его установки из пакета RPM. Для каждого файла из выбранного пакета, который изменился с момента установки, отображается следующая информация:

- `S` — отличается размер файла;
- `M` — права или тип файла (режим) различаются;
- `5` — хеш отличается (ранее сумма MD5);
- `D` — главный/второстепенный номер устройства отличается;
- `L` — отличается путь `readLink(2)`;
- `U` — отличается владелец;
- `G` — отличается группа;
- `T` — отличается `mTime`;
- `P` — отличаются возможности.

По умолчанию отображаются только измененные файлы. Добавьте параметр `-v` (`verbose`), чтобы показать и файлы, которые не изменились, например:

```
# rpm -V samba
S.5....T. /usr/sbin/eventlogadm
```

В этом примере я повторил несколько символов в двоичном файле `eventlogadm`. Он показывает, что размер файла изменился, дайджест больше не соответствует исходному дайджесту и время модификации файла изменилось.

Такие проверки файловой системы помогают отслеживать происходящее в системе и обнаруживать вредоносные атаки. Но к файлам могут быть применены и другие типы атак, включая вирусы и руткиты.

Поиск вирусов и руткитов

Два популярных инструмента вредоносных атак — это вирусы и руткиты, потому что они действуют скрытно. Системы Linux необходимо проверять на наличие обоих вторжений.

Поиск вирусов. *Компьютерный вирус* — это вредоносное программное обеспечение, которое может присоединяться к уже установленному системному программному обеспечению и распространяться через средства массовой информации или сети. Заблуждением является мнение, что вирусов Linux не существует. Злонамеренные создатели вирусов часто фокусируются на более популярных настольных операционных системах, таких как Windows. Однако это не означает, что вирусы не создаются и для систем Linux.

Что еще более важно, системы Linux часто применяются для обработки служб, таких как почтовые серверы, для настольных систем Windows. Поэтому системы Linux, используемые для таких целей, также необходимо проверять на наличие вирусов Windows.

Антивирусное программное обеспечение сканирует файлы с помощью сигнатур вирусов. Сигнатура вируса — это хеш, созданный из двоичного кода вируса. Хеш положительно идентифицирует этот вирус. Антивирусные программы имеют базу данных сигнатур вирусов, которая используется для сравнения с файлами. В зависимости от количества новых угроз база данных сигнатур вирусов может часто обновляться, чтобы обеспечить защиту от новых угроз.

Хорошим антивирусом для открытой бесплатной системы Linux является ClamAV. Чтобы установить его в системе Fedora или RHEL, введите команду `dnf install clamav`. Больше узнать о ClamAV можно на странице clamav.net, где есть документация и инструкция для настройки и запуска антивирусного программного обеспечения.

СОВЕТ

Вы можете просмотреть пакеты, доступные для установки в дистрибутиве Ubuntu, введя команду apt-cache search clamav. Для Ubuntu доступны несколько различных пакетов, поэтому просмотрите информацию о сайте ClamAV, прежде чем выбрать пакет.

Поиск руткитов. Руткит коварнее, чем вирус. *Рутким* — это вредоносная программа, которая:

- скрывается, часто заменяя собой системные команды или программы;
- поддерживает высокоуровневый доступ к системе;
- способна обойти программное обеспечение, созданное для ее обнаружения.

Цель руткита — получить и поддерживать доступ к системе на корневом уровне. Термин был создан путем объединения слов *root*, что означает доступ администратора, и *kit*, что означает несколько программ, работающих совместно.

Поисковик руткитов, который можно использовать в системе Linux, — это `chkrootkit`. Чтобы установить `chkrootkit` в системе Fedora или RHEL, выполните команду `yum install chkrootkit`. Чтобы установить `chkrookit` в системе Ubuntu, примените команду `sudo apt-get install chkrootkit`.

СОВЕТ

Для запуска команды chkrootkit лучше всего использовать CD или флеш-накопитель, чтобы результаты не были скрыты руткитом. Программа Fedora Security Spin применяет chkrootkit на CD. Вы можете получить этот дистрибутив на странице labs.fedoraproject.org/en/security.

Найти руткит с помощью `chkrootkit` очень просто. После установки пакета или загрузки CD введите команду `chkrootkit` в командной строке. Она выполняет поиск по всей файловой структуре, отмечая любые зараженные файлы.

В примере далее показана работа команды `chkrootkit` в зараженной системе. Команда `grep` используется для поиска по ключевому слову `INFECTED`. Обратите внимание на то, что многие файлы, перечисленные как зараженные, являются командными файлами оболочки bash, а это типично для руткита:

```
# chkrootkit | grep INFECTED
Checking 'du'... INFECTED
Checking 'find'... INFECTED
Checking 'ls'... INFECTED
Checking 'lsof'... INFECTED
Checking 'pstree'... INFECTED
Searching for Suckit rootkit... Warning: /sbin/init INFECTED
```

В последней строке кода `chkrootkit` указывает, что система была заражена руткитом `Suckit`. На самом деле она заражена не этим руткитом. При запуске утилит, таких как антивирус и программное обеспечение для обнаружения руткитов, часто выводится ряд ошибок первого рода — ложных срабатываний (false positive). *Ложное срабатывание* — это признак вируса, руткита или другой вредоносной активности, которой на самом деле не существует. В данном конкретном случае оно вызвано известной ошибкой.

Утилиту `chkrootkit` нужно запускать регулярно, и, конечно, следует делать это всякий раз, когда есть подозрение, что была предпринята атака руткитом. Чтобы найти более подробную информацию о `chkrootkit`, перейдите на страницу chkrootkit.org.

СОВЕТ

Еще один поисковик руткитов, который может вас заинтересовать, — это Rootkit Hunter (rkhunter). Запустите скрипт rkhunter, чтобы проверить систему на наличие вредоносных программ и известных руткитов. Настройте rkhunter в файле /etc/rkhunter.conf. Запустите rkhunter -c, чтобы проверить файловую систему на наличие различных руткитов и уязвимостей.

Поиск вторжений. *Система обнаружения вторжений* (Intrusion Detection System, IDS) — это пакет программ, который отслеживает действия системы (или ее сети) на предмет потенциальных вредоносных действий и сообщает о них, помогая обнаружить вторжения. Тесно связан с этой системой программный пакет *Intrusion Prevention System*. Пакеты могут быть объединены, чтобы обеспечить обнаружение и предотвращение вторжений.

Для системы Linux доступны несколько пакетов программ обнаружения вторжений. Некоторые из наиболее популярных утилит перечислены в табл. 22.7. Вы должны знать, что `tripwire` больше не является частью открытого исходного кода. Тем не менее ее исходный код по-прежнему доступен. Более подробную информацию см. на сайте `tripwire`, приведенном в табл. 22.7.

Таблица 22.7. Популярные системы обнаружения вторжений Linux

Система	Инсталлятор	Сайт
aide	yum install aide apt-get install aide	aide.sourceforge.net
Snort	Пакеты с сайта rpm or tarbal	snort.org
tripwire	yum install tripwire apt-get install tripwire	tripwire.org

Advanced Intrusion Detection Environment (aide, усовершенствованная система обнаружения вторжений) IDS использует метод сравнения для поиска вторжений. Ребенком вы, возможно, играли в игру, где нужно сравнить две картинки и найти между ними различия. Утилита `aide` применяет тот же метод. Создается база данных (первая картинка), через некоторое время создается еще одна база данных (вторая картинка), и `aide` сравнивает их, сообщая о различиях.

Итак, нужно сделать начальную базу данных (первую картинку). Лучше всего, когда система только установлена. Команда `aide -i` используется для создания исходной базы данных, и это может занять много времени. Примеры из вывода команды приведены далее. Обратите внимание на то, что команда `aide` сообщает, где она создает свою первую картинку:

```
# aide -i
AIDE, version 0.16.11

### AIDE database at /var/lib/aide/aide.db.new.gz initialized.
```

Далее необходимо переместить исходную базу данных в новое место. Это защитит ее от перезаписи. Кроме того, сравнение не сработает, если база данных не была перемещена. Команда для перемещения базы данных и присвоения ей нового имени выглядит следующим образом:

```
# cp /var/lib/aide/aide.db.new.gz /var/lib/aide/aide.db.gz
```

Чтобы проверить, не подделаны ли ваши файлы, нужно создать новую базу данных (вторую картинку) и сравнить ее с исходной базой данных. Параметр

проверки в команде `aide`, `-c`, создает новую базу данных и запускает сравнение со старой базой данных. Выходные данные, показанные далее, отображают процесс сравнения и отчет команды `aide` о проблемах:

```
# aide -C
...
---------------------------------------------------
Detailed information about changes:
---------------------------------------------------
File: /bin/find
Size : 189736 , 4620
Ctime : 2020-02-10 13:00:44 , 2020-02-11 03:05:52
MD5 : <NONE> , rUJj8NtNa1v4nmV5zfoOjg==
RMD160 : <NONE> , 0CwkiYhqNnfwPUPM12HdKuUSFUE=
SHA256 : <NONE> , jg60Soawj4S/UZXm5h4aEGJ+xZgGwCmN
File: /bin/ls
Size : 112704 , 6122
Ctime : 2020-02-10 13:04:57 , 2020-02-11 03:05:52
MD5 : POeOop46MvRx9qfEoYTXOQ== , IShMBpbSOY8axhw1Kj8Wdw==
RMD160 : N3V3Joe5Vo+cOSSnedf9PCDXYkI= ,
 e0ZneB7CrWHV42hAEgT2lwrVfP4=
SHA256 : vuOFe6FUgoAyNgIxYghOo6+SxR/zxS1s ,
 Z6nEMMBQyYm8486yFSIbKBuMUi/+jrUi
...
File: /bin/ps
Size : 76684 , 4828
Ctime : 2020-02-10 13:05:45 , 2020-02-11 03:05:52
MD5 : 1pCVAWbpeXINiBQWSUEJfQ== , 4ElJhyWkyMtm24vNLya6CA==
RMD160 : xwICWNtQH242jHsH2E8rV5kgSkU= ,
 AZlI2QNlKrWH45i3/V54H+1QQZk=
SHA256 : ffUDesbfxx3YsLDhD0bLTW0c6nykc3m0 ,
 w1qXvGWPFzFir5yxN+n6t3eOWw1TtNC/
...

File: /usr/bin/du
Size : 104224 , 4619
Ctime : 2020-02-10 13:04:58 , 2020-02-11 03:05:53
MD5 : 5DUMKWj6LodWj4C0xfPBIw== , nzn7vrwfBawAeL8nkayICg==
RMD160 : Zlbm0f/bUWRLgi1B5nVjhanuX9Q= ,
 2e5S00lBWqLq4Tnac4b6QIXRCwY=
SHA256 : P/jVAKr/SO0epBBxvGP900nLXrRY9tnw ,
 HhTqWgDyIkUDxA1X232ijmQ/OMA/kRgl
File: /usr/bin/pstree
Size : 20296 , 7030
Ctime : 2020-02-10 13:02:18 , 2020-02-11 03:05:53
MD5 : <NONE> , ry/MUZ7XvU4L2QfWJ4GXxg==
RMD160 : <NONE> , tFZer6As9EoOi58K7/LgmeiExjU=
SHA256 : <NONE> , iAsMkqNShagD4qe7dL/EwcgKTRzvKRSe
...
```

Файлы, перечисленные проверкой `aide` в этом примере, заражены. Однако `aide` может отображать и ложные срабатывания.

Информация о том, где создаются базы данных `aide`, какие сравнения производятся, и другие параметры конфигурации обрабатываются в файле `/etc/aide.conf`. Далее приводится частичное отображение файла. В примере видны имена файлов базы данных и каталогов файлов журналов:

```
# cat /etc/aide.conf
# Example configuration file for AIDE.

@@define DBDIR /var/lib/aide
@@define LOGDIR /var/log/aide

# The location of the database to be read.
database=file:@@{DBDIR}/aide.db.gz

# The location of the database to be written.
#database_out=sql:host:port:database:login_name:passwd:table
#database_out=file:aide.db.new
database_out=file:@@{DBDIR}/aide.db.new.gz
```

Intrusion Detection System отлично помогает в ходе мониторинга системы. При обнаружении потенциальных вторжений сравнение выходных данных с информацией из других команд (например, `rpm -V`) и файлов журналов может помочь лучше понять и исправить любые атаки на систему.

Аудит и проверка Linux

При проведении аудита работоспособности системы Linux необходимо понимать две важные вещи. *Проверка соответствия требованиям* — это аудит общей среды компьютерной системы для обеспечения правильного выполнения политик и процедур, установленных для системы. *Проверка безопасности* — это аудит текущих политик и процедур для обеспечения их соответствия общепринятым лучшим практикам безопасности.

Проверки соблюдения требований

Подобно аудиту в других областях, таких как бухгалтерский учет, аудит может проводиться как работниками системы, так и сторонними людьми. Аудит может быть таким же простым, как сравнение реализованной безопасности с заявленными политиками вашей компании. Однако более популярен аудит с помощью тестирования на проникновение.

Тестирование на проникновение — это метод оценки, используемый для проверки безопасности компьютерной системы путем моделирования вредоносных атак. Его называют также пентестом и этическим хакингом. Не нужно собирать инструменты и нанимать местного хакера, чтобы провести эти тесты.

Kali Linux (www.kali.org) — это дистрибутив, созданный специально для тестирования на проникновение. Его можно задействовать с «живого» DVD или флешки. Компания Offensive Security обучает использованию Kali Linux (www.offensive-security.com/ information-security-training).

Хотя тестирование на проникновение — это очень весело, для тщательного анализа соответствия требуется больше инструментов. Необходимо также использовать контрольные списки с сайтов безопасности по отраслям.

Проверки безопасности

Для проверки безопасности необходимо изучить актуальные рекомендации по обеспечению безопасности. Далее приведен краткий список организаций, которые помогают оставаться в курсе текущих тенденций безопасности.

- Компания United States Cybersecurity and Infrastructure Security Agency (CISA).
  - Сайт www.us-cert.gov.
  - Предлагает National Cyber Alert System (Национальная система компьютерного оповещения).
  - Использует RSS-каналы на последних угрозах безопасности.
- Компания The SANS Institute.
  - Сайт www.sans.org/security-resources.
  - Предлагает последние новости исследований Computer Security Research.
  - Использует RSS-каналы на последних угрозах безопасности.
- Компания Gibson Research Corporation.
  - Сайт www.grc.com.
  - Предлагает сервис *Security Now!*.

Информация с этих сайтов поможет создать более эффективные политики и процедуры. Учитывая, как быстро меняются методы обеспечения безопасности, было бы разумно проверять безопасность чаще, в зависимости от потребностей вашей организации.

Теперь вы гораздо лучше разбираетесь в базовой безопасности Linux, а впереди самое трудное — применить все эти концепции на практике.

Резюме

Основные методы обеспечения безопасности Linux, такие как управление учетными записями пользователей, защита паролей и управление программным обеспечением и службами, формируют основу для безопасности всей вашей системы Linux. В соответствии с этим постоянный мониторинг системы включает в себя наблюдение за файлами системного журнала, проверку на наличие вредоносных вторжений и мониторинг файловой системы.

Кроме того, важно регулярно пересматривать свои политики безопасности. Аудит помогает убедиться в том, что ваша система Linux защищена, и в наличии надлежащих политик и практик безопасности.

Вы сделали первый шаг в получении базовых знаний о процедурах и принципах безопасности. Но недостаточно просто знать основы — необходимо изучить расширенные инструменты безопасности Linux. В следующей главе мы рассмотрим дополнительные вопросы безопасности модулей криптографии и аутентификации.

Упражнения

Обратитесь к материалам этой главы, чтобы выполнить упражнения. Если затрудняетесь с решением заданий, воспользуйтесь ответами к упражнениям, приведенными в приложении Б (хотя Linux позволяет решать задачи разными способами). Выполните все упражнения и только потом посмотрите в ответы. Задачи подходят для систем Fedora или Red Hat Enterprise Linux (некоторые сработают и в других системах Linux).

1. Проверьте сообщения из журнала `systemd` для следующих служб: `NetworkManager.service`, `sshd.service` и `auditd.service`.
2. Перечислите права файла, содержащего пароли пользователей вашей системы, и определите, надежны ли они.
3. Определите срок действия пароля вашей учетной записи с помощью одной команды.
4. Запустите аудит событий в файле `/etc/shadow` с помощью демона `auditd`, а затем проверьте настройки аудита.
5. Создайте отчет из демона `auditd` в файле `/etc/shadow`, а затем отключите в нем аудит.
6. Установите пакет `lemon`, повредите файл `/usr/bin/lemon` (например, скопируйте туда файл `/etcservices`), убедитесь, что файл поврежден, и удалите пакет `lemon`.
7. Вы подозреваете, что сегодня на вашу систему совершена вредоносная атака и были изменены важные двоичные файлы. Какую команду вы должны использовать, чтобы найти эти файлы?
8. Установите и запустите `chkrootkit`, чтобы проверить, не установила ли вредоносная атака из задания 7 руткит.
9. Найдите файлы с набором прав SUID или SGID.
10. Установите пакет `aide`, выполните команду `aide` для инициализации базы данных `aide`, скопируйте базу данных в нужное место и выполните команду `aide`, чтобы проверить, были ли изменены какие-либо важные файлы в вашей системе.

23 Продвинутые методы обеспечения безопасности

В этой главе

- Хеширование и шифрование.
- Проверка целостности файлов.
- Шифрование файлов, каталогов и файловых систем.
- Подключаемые модули аутентификации.
- Управление безопасностью Linux с помощью PAM.

Из-за постоянно меняющихся процессов и растущих угроз уже недостаточно базовой компьютерной безопасности. Как злоумышленники получают доступ к передовым инструментам, так и системный администратор Linux должен делать это. Необходимо разбираться в текущих тенденциях и использовать актуальные инструменты безопасности компьютера.

В этой главе поговорим об основах криптографии, таких как шифры и шифрование. Вы также узнаете, как утилита модуля аутентификации может упростить административные обязанности даже в продвинутой структуре безопасности.

Безопасность Linux с помощью криптографии

Криптография повышает безопасность вашей системы Linux и ее сетевых коммуникаций. *Криптография* — это наука о сокрытии информации. У нее долгая и богатая история, которая уходит корнями в далекое прошлое, еще до появления компьютеров. Из-за активного использования математических алгоритмов криптография легко прижилась в компьютерной сфере. Linux включает в себя много готовых криптографических инструментов.

Чтобы понять криптографию и различные инструменты Linux для ее применения, необходимо изучить несколько криптографических терминов.

- **Plain text (плоский текст)** — текст, который человек или машина могут прочитать и понять.
- **Ciphertext (зашифрованный текст)** — текст, который человек или машина не могут прочитать и понять.
- **Encryption (шифрование)** — процесс преобразования обычного текста в зашифрованный с помощью алгоритма.
- **Decryption (расшифровка)** — процесс преобразования шифрованного текста в обычный текст с помощью алгоритма.
- **Cipher (шифр)** — алгоритм для превращения обычного текста в зашифрованный, а зашифрованного — в обычный.
- **Block cipher (блочный шифр)** — шифр, который разбивает данные на блоки перед шифрованием.
- **Stream cipher (потоковый шифр)** — шифр, который шифрует данные, не разбивая их.
- **Key (ключ)** — фрагмент данных, необходимый шифру для успешного шифрования или дешифрования данных.

Родители маленьких детей часто используют такую форму криптографии. Они пишут слова, а не произносят их. Родители могут взять слово «конфеты» и превратить его в шифрованный текст, говоря друг другу: «К-О-Н-Ф-Е-Т-Ы». Они расшифровывают слово с помощью одинакового шифра и понимают, что это слово «конфеты». К сожалению, детям не требуется много времени, чтобы научиться расшифровывать зашифрованное с помощью орфографического шифра.

Возможно, вы заметили, что в приведенном списке нет хеширования. Его часто путают с шифрованием, поэтому оно требует особого внимания.

Хеширование

Хеширование — это не шифрование, а одна из форм криптографии. В главе 22 «Базовые методы обеспечения безопасности» сказано, что хеширование — это односторонний математический процесс, используемый для создания зашифрованного текста. Однако, в отличие от шифрования, вы не можете дехешировать созданный хеш, превратив его в исходный обычный текст.

Для того чтобы алгоритм хеширования применялся в сфере компьютерной безопасности, он должен быть избавлен от ошибок — это означает, что алгоритм не должен выводить один и тот же хеш для двух совершенно разных входных данных. Каждый вход должен иметь уникальный хешированный выход. Таким образом, *криптографическое хеширование* — это односторонний безошибочный математический процесс.

По умолчанию криптография уже используется в системе Linux. Например, файл `/etc/shadow` содержит хешированные пароли. Хеширование применяется в системах Linux для следующих целей:

- создания паролей (глава 22);
- проверки файлов;
- создания цифровых подписей;
- создания сигнатур вирусов (глава 22).

Хеш называется также *отпечатком контрольной суммы дайджеста сообщения* или *сигнатурой*. Одна из утилит Linux, которая производит дайджесты сообщений, — это `sha256sum`. В главе 10 «Управление программами» было описано программное обеспечение для вашей системы Linux. При загрузке файла программного обеспечения вы можете убедиться, что он не был поврежден.

На рис. 23.1 показан сайт для загрузки дистрибутивного программного обеспечения Fedora, хранящегося и в виде файла, и в виде ISO-образа. На этой странице описано, как загрузить и использовать утилиту `sha256sum`, чтобы убедиться, что загруженный вами ISO-образ не был поврежден во время загрузки.

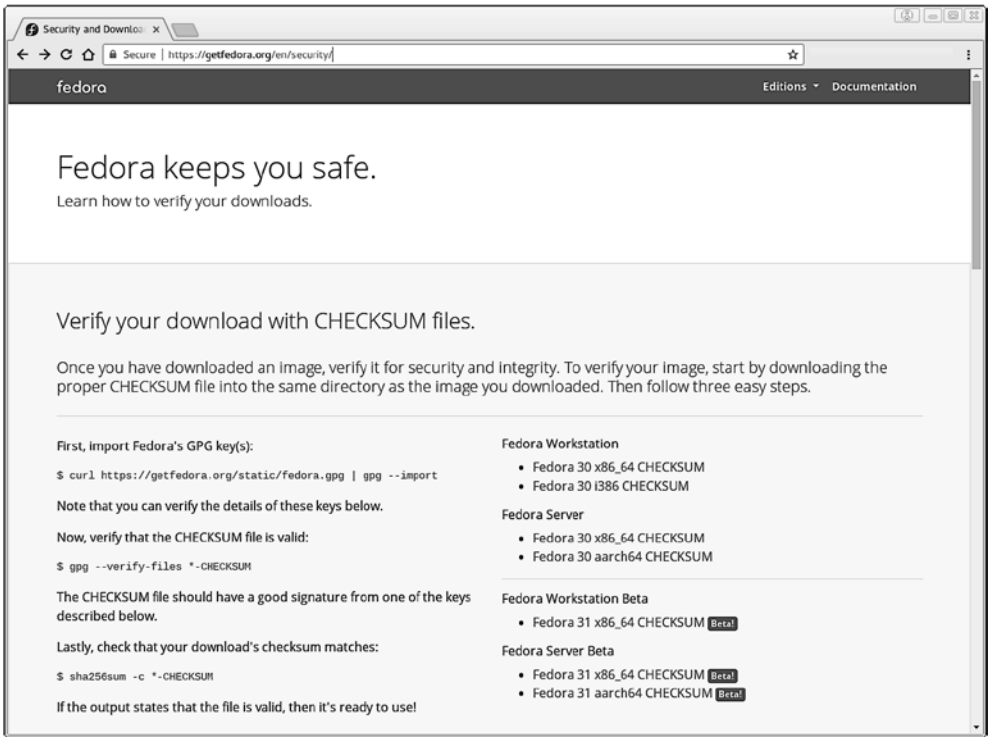


Рис. 23.1. Страница безопасности Fedora ISO security с описанием того, как загрузить и проверить утилиту sha256sum

Хеш создается из программного файла с исходным местоположением с использованием алгоритма хеширования SHA-256. Результаты хеширования могут быть опубликованы открыто, как сделано на рис. 23.1. Чтобы обеспечить целостность загруженного файла программного обеспечения, вы создаете хеш этого файла `sha256 sumhash` в нужном месте. Затем сравниваете результаты вашего хеша с опубликованными результатами хеша. Если они совпадают, значит, файл программного обеспечения не был поврежден.

Чтобы создать хеш, выполните команду `sha256sum` в ISO-образе после его загрузки. Результаты хеширования `sha256sum` для загруженного программного файла показаны далее:

```
$ sha256sum Fedora-Workstation-Live-x86_64-30-1.2.iso
a4e2c49368860887f1cc1166b0613232d4d5de6b46f29c9756bc7cfd5e13f39f
   Fedora-Workstation-Live-x86_64-30-1.2.iso
```

Полученный хеш *действительно* совпадает с тем, который доступен на сайте на рис. 23.1. Это означает, что загруженный ISO-файл не поврежден и готов к работе.

Вы можете реализовать в системе Linux не только хеширование, но и другие средства криптографии. Утилиты криптографии Linux очень просты в использовании. Но сначала вам нужно понять еще несколько основных принципов криптографии.

Шифрование и расшифровка

Основное применение криптографии в системе Linux заключается в кодировании данных (шифрование), чтобы скрыть их от посторонних глаз, а затем декодировании (дешифрование), чтобы их могли увидеть авторизованные пользователи. В системе Linux вы можете зашифровать:

- личные файлы;
- разделы и тома;
- связи веб-страниц;
- сеть;
- резервные копии;
- архивы.

Процессы шифрования/дешифрования применяют специальные математические алгоритмы для выполнения своей задачи. Алгоритмы называются *криптографическими шифрами*.

Криптографические шифры

Один из оригинальных шифров, названный *шифром Цезаря*, создан и использован самим Юлием Цезарем. Однако взломать его было очень легко. Сегодня доступно гораздо больше безопасных шифров. Важно понимать, как работает каждый шифр, потому что сила выбранного вами шифра должна напрямую соотноситься

с потребностями в безопасности данных. В табл. 23.1 перечислена часть современных шифров.

Таблица 23.1. Криптографические шифры

Метод	Описание
AES (Advanced Encryption Standard), также Rijndael	Симметричные криптосистемы. Блочный шифр, шифрующий данные в 128-, 192-, 256-, 512-битовые блоки, использующие 128-, 192-, 256- или 512-битный ключ для шифрования/расшифровки
Blowfish	Симметричные криптосистемы. Блочный шифр, шифрование данных в 64-битные блоки с применением одних и тех же 32- и 448-битных ключей для шифрования/расшифровки
CAST5	Симметричные криптосистемы. Блочный шифр, шифрование данных в 64-битные блоки с использованием одного и того же до 128-битного ключа для шифрования/расшифровки
DES (Data Encryption Standard)	Больше не считается безопасным. Симметричные криптосистемы. Блочный шифр, шифрование данных в 64-битные блоки с применением одного и того же 56-битного ключа для шифрования/расшифровки
3DES Improved DES cipher	Симметричные криптосистемы. Данные шифруются до 48 раз тремя различными 56-битными ключами до завершения процесса шифрования
El Gama	Асимметричное шифрование. Использует два ключа, полученных из логарифмического алгоритма
Elliptic Curve Cryptosystems	Асимметричное шифрование. Применяет два ключа, полученных из алгоритма, содержащего две случайно выбранные точки на эллиптической кривой
IDEA	Симметричные криптосистемы. Блочный шифр, шифрование данных в 64-битные блоки с использованием одного и того же 128-битного ключа для шифрования/расшифровки
RC4, или ArcFour, или ARC4	Потоковый шифр, шифрование данных в 64-битные блоки с применением переменного размера ключа для шифрования/расшифровки
RC5	Симметричные криптосистемы. Блочный шифр, шифрование данных в 32-, 64- или 128-битные блоки с использованием одних и тех же до 2048-битных ключей для шифрования/расшифровки
RC6	Симметричные криптосистемы. Такой же, как RC5, но быстрее
Rijndael или AES	Симметричные криптосистемы. Блочный шифр, шифрующий данные в 128-, 192-, 256-, 512-битные блоки, применяющие 128-, 192-, 256- или 512-битный ключ для шифрования/расшифровки
RSA	Самое популярное асимметричное шифрование. Использует два ключа, полученных из алгоритма, содержащего кратное двум случайно сгенерированным простым числам

Криптографические ключи шифрования

Криптографические шифры требуют фрагмента данных, называемого ключом, для завершения математического процесса шифрования/расшифровки. Ключ может быть либо одним ключом, либо парой ключей.

Обратите внимание на различные размеры ключей шифрования, перечисленные в табл. 23.1. От размера ключа напрямую зависит то, насколько легко взломать шифр. Чем больше ключ, тем меньше вероятность взлома шифра. Например, DES больше не считается безопасным из-за небольшого, 56-битного ключа. Однако шифр с ключом 256 или 512 бит считается безопасным, потому что потребуются триллионы лет, чтобы взломать его грубой силой.

Криптография с симметричным ключом. *Симметричная криптография*, также называемая *криптографией закрытого ключа*, шифрует обычный текст с помощью одного ключевого шифра. Тот же ключ необходим для расшифровки данных. Преимуществом криптографии с симметричным ключом является скорость. Недостаток — необходимость совместного использования одного ключа, если зашифрованные данные должны быть расшифрованы другим человеком.

Пример криптографии с симметричным ключом в системе Linux выполняется с помощью утилиты OpenPGP, GNU Privacy Guard, `gpg2`. Пакет `gnupg2` устанавливается по умолчанию в Fedora и RHEL. Для Ubuntu требуется установить пакет `gnupg2`, чтобы применять команду `gpg2`.

Шифрование и расшифровка архивного файла tar. В следующем примере показаны команда `tar`, используемая для создания сжатого архива `tar` (`backup.tar.gz`), и утилита `gpg2` для шифрования файла. С помощью параметра `-c` утилита `gpg2` шифрует файл симметричным ключом. Исходный файл сохраняется, и создается новый зашифрованный файл `backup.tar.gz.gpg`:

```
# tar -cvzf /tmp/backup.tar.gz /etc
# gpg2 -c --force-mdc \
    -o /tmp/backup.tar.gz.gpg /tmp/backup.tar.gz
Enter passphrase: ******
Repeat passphrase: ******
# cd /tmp ; file backup*
/tmp/enc/backup.tar.gz:     gzip compressed data, last modified: Thu
    Jan 30 02:36:48 2020, from Unix, original size modulo 2^32
49121280
/tmp/enc/backup.tar.gz.gpg: GPG symmetrically encrypted data
(CAST5 cipher)
```

Единственный ключ, применяемый для шифрования файла, защищен парольной фразой. Это просто пароль или фраза, выбранная пользователем во время шифрования.

Чтобы расшифровать файл, снова задействуйте утилиту `gpg2`. Например, если вы передадите файл другому пользователю, он может запустить `gpg2` с параметром `-d` и предоставить кодовую фразу секретного ключа:

```
$ gpg2 -d --force-mdc /tmp/backup.tar.gz.gpg > /tmp/backup.tar.gz
<Во всплывающем окне появится запрос на ввод пароля>
```

```
gpg: CAST5 encrypted data
gpg: encrypted with 1 passphrase
...
```

В результате исходный файл `tar` расшифровывается и копируется в файл `/tmp/backup.tar.gz`. Если в системе работает демон `gpg-agent`, то парольная фраза кэшируется, чтобы файл можно было расшифровать, не вводя ее снова.

Криптография с симметричным ключом довольно проста и понятна. Асимметричная криптография гораздо сложнее, и при ее реализации часто совершают ошибки.

Криптография с асимметричным ключом. *Асимметричная криптография*, также называемая криптографией с закрытым/открытым ключом, использует два ключа, называемых *парой ключей*. Пара ключей состоит из открытого и закрытого ключей. Открытый ключ именно открытый. Нет никакой необходимости держать его в секрете. А вот секретный ключ раскрывать никому нельзя.

Суть криптографии с асимметричным ключом показана на рис. 23.2. Обычный текстовый файл шифруется с помощью открытого ключа из пары ключей. Затем зашифрованный файл может быть передан другому человеку. Для его расшифровки используется закрытый ключ. Он тоже должен быть из пары открытого/закрытого ключей. Таким образом, данные, зашифрованные с помощью открытого ключа, могут быть расшифрованы только с помощью соответствующего закрытого ключа. Преимуществом асимметричной криптографии является повышенная безопасность, недостатком — скорость и управление ключами.

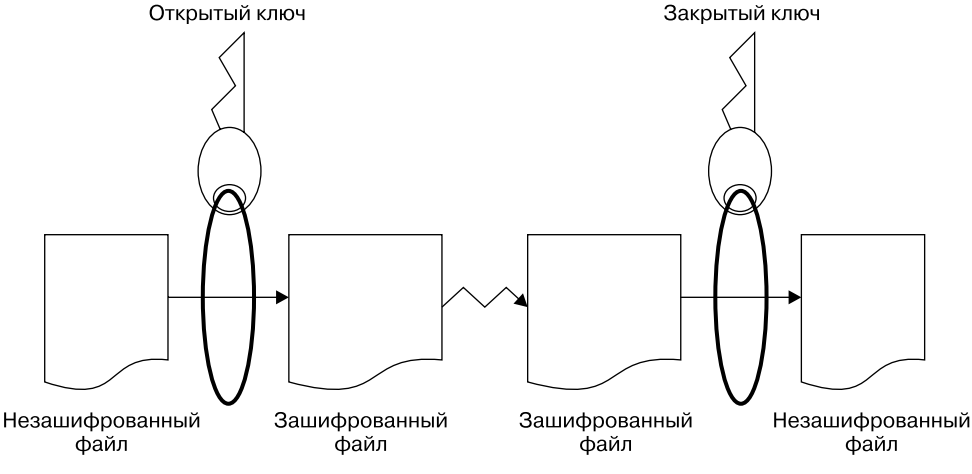


Рис. 23.2. Базовая криптография с асимметричным ключом

Создание пары ключей. Асимметричное шифрование в системе Linux осуществляется с помощью утилиты `gpg2`. Это универсальная утилита криптографии. Прежде чем зашифровать файл, вы должны создать пару ключей и связку ключей. В примере далее была применена команда `gpg2 --gen-key`. Она создает пару из от-

крытого/закрытого ключа для пользователя johndoe в соответствии с желаемыми настройками. Она также генерирует связку ключей для хранения этих ключей:

```
$ gpg2 --gen-key
gpg (GnuPG) 2.2.9; Copyright (C)
 2018 Free Software Foundation, Inc.
...
GnuPG needs to construct a user ID to identify your key.
Real name: John Doe
Email address: jdoe@example.com
You selected this USER-ID:
    "John Doe <jdoe@gmail.com>"
Change (N)ame, (E)mail or (O)kay/(Q)uit? O
You need a Passphrase to protect your secret key.
<Во всплывающем окне появится запрос на ввод пароля>
Enter passphrase: **********
Repeat passphrase: **********
...
gpg: /home/jdoe/.gnupg/trustdb.gpg: trustdb created
gpg: key 383D645D9798C173 marked as ultimately trusted
gpg: directory '/home/jdoe/.gnupg/openpgp-revocs.d' created
gpg: revocation certificate stored as '/home/jdoe/.gnupg/openpgprevocs.d/
7469BCD3D05A4
3130F1786E0383D645D9798C173.rev'
public and secret key created and signed.
pub   rsa2048 2019-10-27 [SC] [expires: 2021-10-26]
      7469BCD3D05A43130F1786E0383D645D9798C173
uid                      John Doe <jdoe@example.com>
sub   rsa2048 2019-10-27 [E] [expires: 2021-10-26]
```

В предыдущем примере утилита gpg2 запрашивает несколько спецификаций для генерирования желаемых открытых/закрытых ключей:

- **ID пользователя**, который определяет часть открытого ключа пары «открытый/закрытый ключ»;
- **адрес электронной почты**, связанный с ключом;
- **парольную фразу**, которая и применяется для идентификации и защиты части закрытого ключа пары «открытый/закрытый ключ».

Пользователь johndoe может проверить свой набор ключей с помощью команды gpg2 --list-keys, как показано в следующем примере. Обратите внимание на то, что идентификатор пользователя (UID) открытого ключа отображается таким же, как был создан, и содержит настоящее имя пользователя johndoe, комментарий и адрес электронной почты:

```
$ gpg2 --list-keys
/home/jdoe/.gnupg/pubring.kbx
------------------------------
pub   rsa2048 2019-10-27 [SC] [expires: 2021-10-26]
      7469BCD3D05A43130F1786E0383D645D9798C173
uid           [ultimate] John Doe <jdoe@example.com>
sub   rsa2048 2019-10-27 [E] [expires: 2021-10-26]
```

После создания пары ключей и их связки файлы могут быть зашифрованы и расшифрованы. В первую очередь открытый ключ должен быть извлечен из связки ключей, чтобы его можно было использовать совместно. В следующем примере утилита gpg2 применяется для извлечения открытого ключа из связки ключей пользователя johndoe. Извлеченный ключ помещается в файл совместного применения. Имя файла может быть любым. В данном случае пользователь johndoe выбрал имя файла JohnDoe.pub:

```
$ gpg2 --export John Doe > JohnDoe.pub
$ ls *.pub
JohnDoe.pub
$ file JohnDoe.pub
JohnDoe.pub: PGP/GPG key public ring (v4) created Sun Oct 27 16:24:27
2019 RSA (Encrypt or Sign) 2048 bits MPI=0xc57a29a6151b3e8d...
```

Передача открытого ключа. Файл, содержащий открытый ключ, может быть передан любым доступным способом. Его можно отправить в виде вложения по электронной почте или даже разместить на веб-странице. Открытый ключ — он и есть открытый, поэтому нет необходимости его скрывать. В следующем примере пользователь johndoe передал файл, содержащий его открытый ключ, пользователю jill. Далее последний добавляет открытый ключ пользователя johndoe в свою связку ключей с помощью команды gpg2 --import. Пользователь jill убеждается, что открытый ключ johndoe добавлен, с помощью команды gpg2 --list-keys, которая просматривает ключи в общей связке ключей:

```
$ ls *.pub
JohnDoe.pub
$ gpg2 --import JohnDoe.pub
gpg: directory '/home/jill/.gnupg' created
...
gpg: directory '/home/jill/.gnupg' created
gpg: keybox '/home/jill/.gnupg/pubring.kbx' created
gpg: /home/jill/.gnupg/trustdb.gpg: trustdb created
gpg: key 383D645D9798C173: public key "John Doe <jdoe@example.com>"
imported
gpg: Total number processed: 1
gpg:               imported: 1
$ gpg2 --list-keys
/home/jill/.gnupg/pubring.gpg
----------------------------
pub   rsa2048 2019-10-27 [SC] [expires: 2021-10-26]
      7469BCD3D05A43130F1786E0383D645D9798C173
uid           [ unknown] John Doe <jdoe@example.com>
sub   rsa2048 2019-10-27 [E] [expires: 2021-10-26]
```

Шифрование сообщения электронной почты. Добавленный в связку ключей открытый ключ можно применять для шифрования данных для первоначального владельца. В приведенном далее примере кода обратите внимание на то, что пользователь jill создал текстовый файл MessageForJohn.txt для пользователя johndoe.

- Пользователь `jill` шифрует файл с помощью открытого ключа пользователя `johndoe`.
- Зашифрованный файл `MessageForJohn` создается с помощью параметра `--out`.
- Параметр `--recipient11` идентифицирует открытый ключ пользователя `johndoe`, применяя только часть реального UID его открытого ключа в кавычках — `"John Doe"`:

```
$ gpg2 --out MessageForJohn --recipient "John Doe" \
--encrypt MessageForJohn.txt
...
$ ls
JohnDoe.pub MessageForJohn MessageForJohn.txt
```

Зашифрованный файл сообщения `MessageForJohn`, созданный из обычного текстового файла `MessageForJohn.txt`, может быть отправлен пользователю `johndoe`. Чтобы расшифровать это сообщение, он задействует *свой* закрытый ключ, идентифицированный и защищенный секретной парольной фразой, которая применялась при создании ключа. После того как пользователь `johndoe` укажет правильную парольную фразу, команда `gpg2` расшифрует файл сообщения и поместит его в файл `JillsMessage`, обозначенный параметром `-out`. После этого текст сообщения можно прочитать:

```
$ ls MessageForJohn
MessageForJohn
$ gpg2 --out JillsMessage --decrypt MessageForJohn
<A pop-up window prompts you for a passphrase>
gpg: encrypted with 2048-bit RSA key, ID D9EBC5F7317D3830, created
2019-10-27
      "John Doe <jdoe@example.com>"
$ cat JillsMessage
I know you are not the real John Doe.
```

Итак, вот что нужно сделать для того, чтобы зашифровать и расшифровать файлы с использованием асимметричных ключей.

1. Создать пару ключей и связку ключей.
2. Добавить копию открытого ключа в файл.
3. Передать файл с открытым ключом.
4. Пользователи, которые хотят отправить вам зашифрованные файлы, должны добавить этот открытый ключ в свою связку ключей.
5. Файл шифруется с помощью *вашего* открытого ключа.
6. Зашифрованный файл отправляется вам.
7. Вы расшифровываете файл с помощью *своего* закрытого ключа.

Именно поэтому асимметричные ключи могут вызвать путаницу! Помните, что в асимметричной криптографии каждый открытый и закрытый ключ — это пара, которая работает вместе.

Цифровые подписи

Цифровая подпись — это электронный механизм, используемый для аутентификации и проверки данных. Цифровая подпись — это не скан вашей физической подписи. Это криптографический токен, отправленный вместе с файлом, поэтому получатель файла может быть уверен, что файл пришел от вас и никоим образом не был изменен.

При создании цифровой подписи выполняются следующие действия.

1. Создается файл или сообщение.
2. С помощью утилиты `gpg2` можете создать хеш, или дайджест, сообщения файла.
3. Затем утилита `gpg2` шифрует хеш и файл, используя асимметричный шифр ключа. Для шифрования применяется закрытый ключ пары «открытый/закрытый ключ». Теперь это зашифрованный файл с цифровой подписью.
4. Вы отправляете зашифрованный хеш (он же цифровая подпись) и файл получателю.
5. Получатель повторно создает хеш, или дайджест, сообщения полученного зашифрованного файла.
6. Используя утилиту `gpg2`, получатель расшифровывает полученную цифровую подпись с помощью открытого ключа, чтобы получить исходный хеш, или дайджест, сообщения.
7. Утилита `gpg2` сравнивает исходный хеш с воссозданным, чтобы проверить, совпадают ли они. Если совпадают, получателю сообщается, что с цифровой подписью все в порядке.
8. Теперь получатель может прочитать расшифрованный файл.

Обратите внимание на то, что в пункте 3 сначала используется закрытый ключ. В описании криптографии с асимметричным ключом говорилось, что первым был применен открытый ключ. Криптография с асимметричным ключом довольно гибка и позволяет вам использовать свой закрытый ключ для шифрования, а получателю — задействовать ваш открытый ключ для расшифровки.

ПРИМЕЧАНИЕ

Цифровые подписи имеют собственные специальные шифры. Несколько шифров могут обрабатывать как шифрование, так и создание подписей, однако есть и такие, чья единственная работа заключается в создании цифровых подписей. Ранее самыми популярными криптографическими шифрами для создания подписей были RSA и алгоритм цифровой подписи (Digital Signature Algorithm, DSA). Алгоритм RSA может использоваться как для шифрования, так и для создания подписей, в то время как DSA — только для создания цифровых подписей. Сегодня алгоритм Ed25519 считается более безопасным и быстрым, чем RSA, а алгоритм ECDSA обеспечивает лучшую защиту, чем DSA.

Как видите, цифровая подпись содержит как криптографическое хеширование, так и асимметричную криптографию ключа. Этот сложный процесс часто обраба-

тывается специально настроенным для этого приложением, а не непосредственно пользователями системы Linux. Однако вы можете вручную добавить собственные цифровые подписи к документам.

Подпись файла цифровой подписью. Предположим, что пользователь `johndoe` собирается отправить пользователю `christineb` сообщение вместе со своей цифровой подписью. Он создал файл, содержащий текстовое сообщение. Он применяет утилиту `gpg2` для создания файла подписи и шифрования файла сообщения. Параметр `--sign` сообщает утилите `gpg2`, что `MessageForChristine.txt` — это файл для шифрования и использования цифровой подписи. В ответ утилита `gpg2`:

- создает дайджест сообщения (он же хеш) файла сообщения;
- шифрует дайджест сообщения, который создает цифровую подпись;
- шифрует файл сообщения;
- помещает зашифрованное содержимое в файл, указанный параметром `--output`, то есть в файл `JohnDoe.DS`.

Теперь файл пользователя `JohnDoe.DS` содержит зашифрованное и подписанное цифровой подписью сообщение. В примере продемонстрирован этот процесс:

```
$ gpg2 --output JohnDoe.DS --sign MessageForJill.txt
```

После того как пользователь `jill` получит подписанный и зашифрованный файл, он может применить утилиту `gpg2` для проверки цифровой подписи и расшифровки файла. В следующем примере используется параметр `--decrypt` вместе с именем файла с цифровой подписью `JohnDoe.DS`. Сообщение файла расшифровывается и отображается. Цифровая подпись файла проверяется и считается действительной:

```
$ gpg2 --decrypt JohnDoe.DS
I am the real John Doe!
gpg: Signature made Sun 27 Oct 2019 07:03:21 PM EDT
gpg:                using RSA key
7469BCD3D05A43130F1786E0383D645D9798C173
gpg: Good signature from "John Doe <jdoe@example.com>" [unknown]
...
```

Без открытого ключа пользователя `johndoe` на связке ключей пользователя `jill` последний не смог бы расшифровать это сообщение и проверить цифровую подпись.

СОВЕТ

В предыдущем примере цифровая подпись документа позволяет любому человеку, имеющему открытый ключ, расшифровать документ. Чтобы по-настоящему скрыть его, задействуйте открытый ключ получателя для шифрования с помощью параметров --sign и --encrypt команды gpg2. Получатель сможет расшифровать документ с помощью своего закрытого ключа.

Понимание основ криптографии поможет вам начать работу по защите системы Linux с помощью шифрования. Имейте в виду, что в этой главе мы рассмотрели только основы. Есть еще много относящихся к криптографии тем, таких как цифровые сертификаты и инфраструктура открытых ключей, которые стоит изучить дополнительно.

Криптография в системе Linux

В вашей системе Linux доступны многие инструменты криптографии. Какие из них вы решите использовать, зависит от требований безопасности в вашей организации. Далее приведен краткий обзор доступных инструментов криптографии Linux.

Обеспечение целостности файлов

Ранее в этой главе целостность файла ISO проверялась с помощью утилиты дайджеста сообщений `sha256sum`.

Связанные утилиты дайджеста сообщений — это:

- `sha224sum`;
- `sha256sum`;
- `sha384sum`;
- `sha512sum`.

Эти инструменты работают точно так же, как команда `sha1sum`, за исключением, конечно, использования стандарта криптографического хеша SHA-2. Единственное различие между инструментами SHA-2 — это длина ключа, который они применяют. Команда `sha224sum` использует длину ключа 224 бита, команда `sha256sum` — 256 бит и т. д. Помните, что чем больше длина ключа, тем меньше вероятность взлома.

Криптографический хеш-стандарт SHA-2 был создан Агентством национальной безопасности (National Security Agency, NSA). SHA-3 — еще один криптографический стандарт хеширования, который был выпущен Национальным институтом стандартов и технологий (The National Institute of Standards and Technology, NIST) в августе 2015 года.

Шифрование файловой системы Linux при установке

Возможно, вам потребуется зашифровать всю файловую систему на своем сервере Linux. Это можно сделать несколькими способами, в том числе с помощью бесплатного программного обеспечения с открытым исходным кодом (FOSS) сторонних инструментов, например Linux Unified Key Setup (LUKS) (gitlab.com/cryptsetup/cryptsetup).

Один из вариантов — это зашифровать корневой раздел при установке системы Linux (см. главу 9 «Установка Linux»). Возможность шифрования в процессе установки имеют многие дистрибутивы Linux, например Red Hat Enterprise Linux (рис. 23.3).

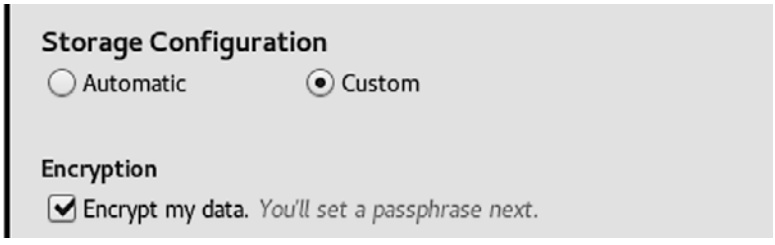


Рис. 23.3. Шифрование при установке дистрибутива Red Hat Enterprise Linux

После выбора этого варианта во время установки нужно будет ввести пароль. Это симметричная ключевая криптография с паролем, защищающим единственный ключ. На рис. 23.4 показано диалоговое окно, запрашивающее пароль ключа. Пароль должен содержать не менее восьми символов.

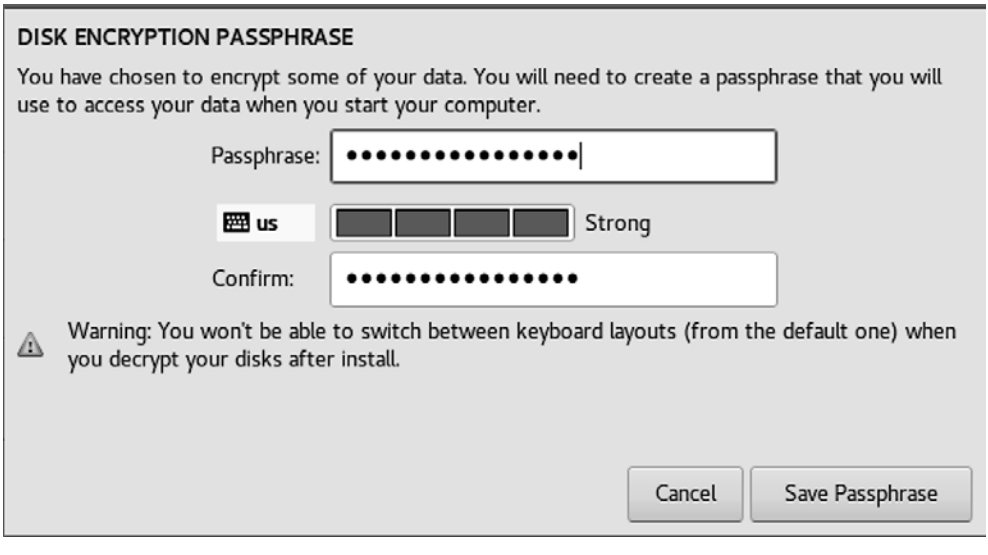


Рис. 23.4. Шифрование симметричным ключом в дистрибутиве Fedora

Если вы выберете этот параметр шифрования, то при каждой загрузке системы нужно будет вводить пароль симметричного ключа. На рис. 23.5 показано, как это выглядит. Это защитит корневой раздел, если диск, на котором он находится, будет украден.

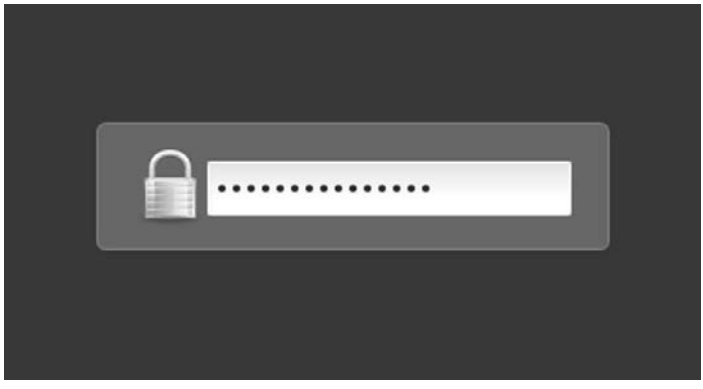


Рис. 23.5. Запрос пароля симметричного ключа шифрования при загрузке

Если вы получили систему с зашифрованным диском, применив привилегии суперпользователя, то можете задействовать команды `lvs` и `cryptsetup` и файл `/etc/crypttab`. В следующем примере команда `lvs` отображает все логические тома, находящиеся в данный момент в системе, и их базовые имена устройств (обзор различных команд LVM см. в главе 12 «Управление дисками и файлами»):

```
# lvs -o devices
  Devices
  /dev/mapper/luks-b099fbbe-0e56-425f-91a6-44f129db9f4b(56)
  /dev/mapper/luks-b099fbbe-0e56-425f-91a6-44f129db9f4b(0)
```

В этой системе обратите внимание на то, что имена базовых устройств начинаются с `luks`. Это говорит о том, что для шифрования жесткого диска был использован стандарт Linux Unified Key Setup (LUKS).

ПРИМЕЧАНИЕ

В Ubuntu по умолчанию не установлена команда lvs. Чтобы установить ее, введите команду sudo apt-get install lvm2 в командной строке.

Зашифрованные логические тома монтируются во время загрузки с помощью информации из файла `/etc/fstab`. Однако содержимое файла `/etc/crypttab`, которое используется для запроса пароля во время загрузки, будет расшифровывать записи файла `/etc/fstab` в порядке их монтирования. Это продемонстрировано в следующем коде. Обратите внимание на то, что имена `luks` совпадают с именами, перечисленными командой `lvs` в предыдущем примере:

```
# cat /etc/crypttab
luks-b099fbbe-0e56-425f-91a6-44f129db9f4b
     UUID=b099fbbe-0e56-425f-91a6-44f129db9f4b none
```

Можете применить команду `cryptsetup`, чтобы получить больше информации о зашифрованных томах своей системы Linux. В следующем примере параметр

status используется вместе с именем устройства luks для определения дополнительной информации:

```
# cryptsetup status luks-b099fbbe-0e56-425f-91a6-44f129db9f4b
/dev/mapper/luks-b099fbbe-0e56-425f-91a6-44f129db9f4b
 is active and is in use.
  type:    LUKS1
  cipher:  aes-xts-plain64
  keysize: 512 bits
  device:  /dev/sda3
  offset:  4096 sectors
  size:    493819904 sectors
  mode:    read/write
```

Шифрование каталога Linux

Для шифрования в системе Linux можете использовать утилиту ecryptfs. Она не является типом файловой системы, как можно подумать, судя по названию. Это совместимая с POSIX утилита, которая позволяет создавать слой шифрования поверх любой файловой системы.

Утилита ecryptfs по умолчанию не установлена в Fedora и недоступна в RHEL. Чтобы установить ее в дистрибутиве Fedora, используйте команду dnf install ecryptfs-utils. Если утилита не установлена в системе Debian, примените команду sudo apt-get install ecrypt-utils.

СОВЕТ

Поскольку утилита ecryptfs применяется для шифрования, распространенная ошибка — ставить букву n после буквы e в синтаксисе ecryptfs. Если вы получили ошибку при использовании утилиты ecryptfs, убедитесь, что случайно не написали encryptfs.

В следующем примере у пользователя johndoe будет подкаталог, зашифрованный с помощью утилиты ecryptfs. Прежде чем зашифровать каталог, убедитесь, что в нем нет никаких файлов. Если они там есть, переместите их в безопасное место до завершения шифрования. Если вы этого не сделаете, то не сможете получить к ним доступ в дальнейшем.

Теперь, чтобы зашифровать каталог /home/johndoe/Secret, примените команду mount. Вы должны обладать правами суперпользователя для монтирования и размонтирования зашифрованного каталога в этом методе. Посмотрите на команду mount в следующем примере. Она похожа на обычную команду mount, за исключением того, что используется тип раздела ecryptfs. Элемент для монтирования и его точка монтирования — это один и тот же каталог! Вы буквально шифруете каталог и монтируете его на себя. Другое необычное отличие этой команды mount заключается в том, что она запускает утилиту ecryptfs, которая задает несколько интерактивных вопросов:

```
# mount -t ecryptfs /home/johndoe/Secret /home/johndoe/Secret
Select key type to use for newly created files:
```

```
 1) tspi
 2) passphrase
 3) pkcs11-helper
 4) openssl
Selection: 2
Passphrase: **********
Select cipher:
 1) aes: blocksize = 16;
 min keysize = 16; max keysize = 32 (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]: 1
Select key bytes:
 1) 16
 2) 32
 3) 24
Selection [16]: 16
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=70993b8d49610e67
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 [70993b8d49610e67] 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
```

Утилита `ecryptfs` позволяет:

- выбрать тип ключа;
- задать кодовую фразу;
- выбрать шифр;
- указать размер ключа (в байтах);

- включить или отключить передачу простого текста;
- включить или отключить шифрование имени файла.

Она также предупреждает вас, когда вы впервые монтируете этот зашифрованный каталог, потому что ключ ранее не использовался. Утилита позволяет применить цифровую подпись к смонтированному каталогу, так что, если вы смонтируете его еще раз, она просто смонтирует каталог и не потребует парольной фразы.

СОВЕТ

Запишите параметры, которые устанавливаете при первом монтировании папки ecryptfs. При повторном монтировании папки они вам понадобятся.

Чтобы убедиться, что зашифрованный каталог теперь смонтирован, можете снова применить команду `mount`. В следующем примере используется команда `mount`, а затем передается в команду `grep` для поиска каталога `/home/johndoe/Secret`. Как видите, каталог монтируется с типом `ecryptfs`:

```
# mount | grep /home/johndoe/Secret

/home/johndoe/Secret on /home/johndoe/Secret type ecryptfs
(rw,relatime,ecryptfs_sig=70993b8d49610e67,ecryptfs_cipher=aes,
ecryptfs_key_bytes=16,ecryptfs_unlink_sigs)
```

До сих пор вы не видели, как выглядит смонтированный и зашифрованный каталог. В приведенном далее примере файл `my_secret_file` копируется в зашифрованном виде. Пользователь `johndoe` по-прежнему может применять команду `cat` для отображения файла в виде обычного текста. Файл автоматически расшифровывается типом `ecryptfs`:

```
$ cp my_secret_file Secret
$ cat /home/johndoe/Secret/my_secret_file
Shh... It's a secret.
```

Суперпользователь может применить команду `cat` для отображения файла в текстовом виде:

```
# cat /home/johndoe/Secret/my_secret_file
Shh... It's a secret.
```

Однако после размонтирования зашифрованного каталога с помощью команды `umount` файлы больше не расшифровываются автоматически. Содержимое файла `my_secret_file` теперь стало полной тарабарщиной, и его не может прочитать даже суперпользователь:

```
# umount /home/johndoe/Secret
```

Таким образом, утилита `ecryptfs` позволяет создать в файловой системе точку для быстрого шифрования и расшифровки файлов. Однако после того, как этот каталог больше не монтируется как тип `ecryptfs`, файлы оказываются защищенными и не могут быть расшифрованы.

СОВЕТ

Как обычный пользователь, вы можете применить команды ecryptfs-setup-private и ecryptfs-mount-private для настройки частной криптографической точки монтирования.

Шифрование файла Linux

Самый широко распространенный инструмент для шифрования файлов в системе Linux — утилита OpenPGP GNU Privacy Guard, `gpg`. Ее гибкость и разнообразие опций, а также то, что она установлена по умолчанию в большинстве дистрибутивов Linux, добавляют ей популярности.

ВНИМАНИЕ!

Если ваша организация работает со сторонними облачными хранилищами, вы должны знать, что некоторые из этих компаний, например Dropbox, не шифруют файлы до тех пор, пока они не будут получены. Это означает, что компания имеет ключи, необходимые для расшифровки файлов, и может оставить данные вашей организации незащищенными. Шифрование файлов в системе Linux перед их отправкой в облако усиливает их защиту.

В системе Linux вы можете использовать и другие инструменты криптографии для шифрования файлов. Как и утилита `gpg`, многие из них позволяют вам сделать гораздо больше, чем просто зашифровать файл. Далее приведены некоторые популярные инструменты криптографии Linux, с помощью которых можно шифровать файлы.

- `aescrypt`. Использует шифр симметричного ключа Rijndael, называемый также AES. Этот сторонний инструмент FOSS можно загрузить на www.aescriypt.com.
- `bcrypt`. Применяет симметричный ключ шифра *blowfish*. По умолчанию не установлен. После установки `bcrypt` будут доступны справочные страницы:
  - для Fedora (недоступно в RHEL) — `yum install bcrypt`;
  - для Ubuntu — `sudo apt-get install bcrypt`.
- `ccrypt`. Использует шифр симметричного ключа Rijndael, называемый также AES. Был создан для замены стандартной утилиты Unix `crypt` и не установлен по умолчанию. После установки `ccrypt` будут доступны справочные страницы:
  - для Fedora (недоступно в RHEL) — `yum install ccrypt`;
  - для Ubuntu — `sudo apt-get install ccrypt`.
- `gpg`. Может использовать либо асимметричные пары ключей, либо симметричный ключ. Установлен по умолчанию и является предпочтительным инструментом криптографии для серверов Linux. Применяемый по умолчанию шифр задается в файле `gpg.conf`. Есть справочные страницы, также можно использовать команду `info gnupg`.

Имейте в виду, что это лишь наиболее популярные инструменты. Кроме того, помните, что многие из них могут применяться не только для шифрования файлов.

Шифрование Linux с помощью различных инструментов

Вы можете применить *криптографию*, определяемую как акт написания или генерирования кодов, предназначенных для хранения секретов, практически ко всему содержимому в Linux. Помимо файловых систем, каталогов и файлов, можно шифровать резервные копии, архивные файлы, сетевые подключения и многое другое.

В табл. 23.2 перечислены некоторые из инструментов криптографии Linux и указано, что они делают. Если хотите увидеть полный список инструментов криптографии, установленных в вашем дистрибутиве Linux, введите `man -k crypt` в командной строке.

Таблица 23.2. Инструменты криптографии в Linux

Инструмент	Описание
Duplicity	Шифрует резервные копии. Чтобы установить инструмент в дистрибутиве Fedora, введите команду yum install duplicity. Чтобы установить в Ubuntu, введите команду sudo apt-get install duplicity
gpg-zip	Использует программу GNU Privacy Guard для шифрования или подписи файлов в архиве. Устанавливается по умолчанию
Openssl	Инструмент, реализующий протоколы Secure Sockets Layer (SSL) и Transport Layer Security (TLS). Эти протоколы требуют шифрования. Устанавливается по умолчанию
Seahorse	Программа шифрования ключей GNU Privacy Guard. Установлена по умолчанию в дистрибутиве Ubuntu. Чтобы установить в дистрибутивах Fedora и RHEL, введите команду yum install seahorse
Ssh	Шифрует удаленный доступ по сети. Устанавливается по умолчанию
Zipcloak	Шифрует записи в ZIP-файле. Устанавливается по умолчанию

В системе Linux инструментов криптографии много, как и прочих элементов. Это обеспечивает гибкость и разнообразие, необходимые для реализации стандартов криптографии в вашей организации.

Шифрование с помощью GUI-инструментов

Окно Passwords and Keys (Пароли и ключи) предоставляет средства просмотра ключей и паролей и управления ими с рабочего стола GNOME. Это окно можно запустить, выбрав соответствующий значок на экране Acivities (Приложения) и выполнив команду `seahorse`. Откроется окно, в котором можно изменить:

- **пароли.** Когда вы заходите на сайт из браузера Chromium или Chrome и вводите имя пользователя и пароль (и сохраняете последний), он хранится в вашей системе для следующего посещения этого сайта. Выберите под полем для пароля строку с логином для входа в систему, чтобы просмотреть каждые из сохраненных имен пользователей и паролей;

- **сертификаты.** Вы можете просмотреть сертификаты, связанные с хранилищами Gnome2 Key Storage, User Key Storage, System Trust и Default Trust;
- **ключи PGP.** Можете просмотреть ключи GPG, выбрав строку GnuPG keys;
- **протокол Secure Shell.** Вы можете создавать открытые и закрытые ключи OpenSSH, которые, действуя как пароли для аутентификации, позволяют входить в удаленные системы с помощью `ssh`, `scp`, `rsync`, `sftp` и связанных с ними команд. Выберите строку OpenSSH keys, чтобы просмотреть все созданные ключи. (См. раздел об аутентификации на основе ключей в главе 13, чтобы узнать, как создавать такие ключи.)

Еще один мощный инструмент безопасности, доступный в Linux, — это модули PAM. В следующих разделах этой главы рассматриваются основные концепции PAM и то, как можно использовать этот инструмент для повышения безопасности системы Linux.

Безопасность Linux при помощи PAM

Подключаемые модули аутентификации (Pluggable Authentication Modules, PAM) были созданы компанией Sun Microsystems и первоначально реализованы в операционной системе Solaris. Проект Linux-PAM начался в 1997 году. Сегодня большинство дистрибутивов Linux используют PAM.

PAM упрощают управление аутентификацией. Помните, что аутентификация (см. главу 22 «Базовые методы обеспечения безопасности») — это процесс определения того, что субъект (он же пользователь или процесс) является тем, за кого себя выдает. Его иногда называют также идентификацией и аутентификацией. PAM — это централизованный метод обеспечения аутентификации для системы Linux и ее приложений.

Существуют приложения, написанные под PAM. Не нужно переписывать и перекомпилировать приложение с поддержкой PAM, чтобы изменить его параметры аутентификации. Все необходимые изменения вносятся в файл конфигурации PAM для приложений, поддерживающих PAM. Таким образом, управление аутентификацией для этих приложений централизовано и упрощено.

Вы можете проверить, являются ли конкретное приложение или утилита Linux подходящими для PAM. Проверьте, скомпилированы ли они с библиотекой PAM `libpam.so`. В следующем примере приложение `crontab` проверяется на связь с PAM. Команда `ldd` проверяет зависимости общей библиотеки файла. Чтобы было проще, для поиска библиотеки PAM используется команда `grep`. Как видите, `crontab` в этой конкретной системе Linux знает о существовании PAM:

```
# ldd /usr/bin/crontab | grep pam
libpam.so.0 => /lib64/libpam.so.0 (0x00007fbee19ce000)
```

Перечислю преимущества применения PAM в системе Linux.

- Упрощенное и централизованное управление аутентификацией администратором.
- Упрощенная разработка приложений, поскольку разработчики могут писать приложения с использованием библиотеки PAM вместо создания собственных процессов аутентификации.
- Гибкость аутентификации:
  - разрешение или запрет на доступ к ресурсам на основе традиционных критериев, например идентификация;
  - разрешение или запрет на доступ на основе дополнительных критериев, например ограничения по времени суток;
  - установка предметных ограничений, например использование ресурсов.

Хотя преимущества PAM упрощают управление аутентификацией, на самом деле модули работают не так просто.

Процесс аутентификации с помощью PAM

Когда субъект (пользователь или процесс) запрашивает доступ к приложению или к утилите с поддержкой PAM, для завершения процесса аутентификации субъекта задействуются два основных компонента:

- файл конфигурации приложения с поддержкой PAM;
- PAM, используемые файлом конфигурации.

Каждый файл конфигурации приложения с поддержкой PAM находится в центре процесса. Файлы конфигурации PAM вызывают определенные модули для выполнения необходимой аутентификации. PAM аутентифицируют субъектов по данным авторизации системы, например по централизованной учетной записи пользователя с помощью LDAP (см. главу 11 «Управление учетными записями»).

Linux поставляется со многими приложениями, которые поддерживают PAM, с уже установленными нужными файлами конфигурации и модулями. Если у вас есть особые потребности в аутентификации, скорее всего, вы найдете модуль PAM, предназначенный для этого. Но прежде, чем начать настраивать PAM, нужно больше узнать о том, как они работают.

PAM предпринимают ряд шагов с использованием модулей и файлов конфигурации для обеспечения правильной аутентификации приложения.

1. Субъект (пользователь или процесс) запрашивает доступ к приложению.
2. Файл конфигурации PAM приложения, содержащий политику доступа, открывается и считывается. Политика доступа устанавливается с помощью списка

всех PAM, которые будут применяться в процессе аутентификации. Этот список называется *стеком*.

3. Модули PAM в стеке вызываются в том порядке, в котором они перечислены.
4. Каждый PAM либо успешно выполняется, либо нет.
5. Стек продолжает считываться по порядку, и он не обязательно будет останавливаться после одного сбоя.
6. Результаты состояния всех PAM объединяются в единый общий результат успеха или неудачи аутентификации.

Как правило, если один PAM возвращает состояние сбоя, доступ к приложению будет запрещен. Однако это зависит от параметров файла конфигурации. Большинство файлов конфигурации PAM находятся в файле `/etc/pam.d`. Общий формат файла конфигурации PAM выглядит так:

```
context   control flag   PAM module   [ module options ]
```

Контексты модулей PAM

Модули PAM имеют стандартные функции, предоставляемые различными службами аутентификации. Эти стандартные функции внутри PAM можно разделить на типы функций, называемые *контекстами*. Контексты можно назвать также *интерфейсами модулей* или их *типами*. В табл. 23.3 перечислены разные контексты PAM и указано, какой тип службы проверки подлинности они предоставляют.

Таблица 23.3. Контексты PAM

Контекст	Описание службы
auth	Предоставляет службы управления аутентификацией, например проверку паролей учетных записей
account	Предоставляет службы проверки учетных записей, например ограничение доступа по времени суток
password	Управляет настройками паролей учетных записей, например ограничивает длину пароля

Флаги управления PAM

В файле конфигурации PAM флаги управления используются для определения общего состояния, которое возвращается приложению. Флаг управления может быть одним из следующих.

- **Простое ключевое слово.** Единственная проблема здесь заключается в том, что соответствующий модуль PAM возвращает статус `failed` или `success`. Как обрабатываются эти статусы, смотрите в табл. 23.4.

Таблица 23.4. Флаги управления PAM и обработка ответов

Флаг	Описание
required	В случае сбоя возвращает приложению статус сбоя после того, как остальные контексты в стеке будут запущены. Например, необходимый элемент управления может привести к сбою входа в систему, если введен неверный пароль. Но пользователь может не знать о сбое до тех пор, пока не введет пароль, так как скрыт факт, что причиной сбоя было неправильное имя пользователя
requisite	В случае сбоя возвращает приложению статус сбоя немедленно, не запуская остальную часть стека. (Будьте осторожны при размещении этого элемента управления в стеке.) Например, необходимый элемент управления может потребовать проверки подлинности на основе ключа, которая немедленно завершится неудачей, если действительный ключ не будет предоставлен. В этом случае он может потерпеть неудачу, даже не запросив имя пользователя/пароль
sufficient	В случае сбоя состояние модуля игнорируется. В случае успеха статус успеха немедленно возвращается приложению без запуска остальной части стека. (Будьте осторожны при размещении этого элемента управления в стеке)
optional	Этот флаг управления важен только для окончательного общего состояния возврата успеха или неудачи. Это как тайм-брейк. Когда другие модули в стеке конфигурационных файлов возвращают статусы, которые не являются четкими статусами сбоя или успеха, статус этого модуля используется для определения окончательного состояния. В тех случаях, когда другие модули в стеке возвращают четкий отказ или успех, этот статус игнорируется
include	Получите все статусы из стека этого конкретного файла конфигурации PAM, чтобы включить их в общий статус возврата этого стека. Это как если бы весь стек из файла конфигурации теперь находился в этом файле конфигурации
substack	Аналогичен флагу управления include, за исключением того, как определенные ошибки и оценки влияют на основной стек. Он вынуждает стек файлов конфигурации действовать как подстек основного стека. Таким образом, определенные ошибки и оценки влияют только на него, а не на основной стек

- **Ряд действий.** Возвращаемый статус модуля обрабатывается с помощью ряда действий, перечисленных в файле.

В табл. 23.4 показаны различные флаги управления ключевыми словами и их ответы на возвращаемое состояние модуля. Обратите внимание на то, что некоторые флаги управления должны быть аккуратно добавлены в стек файла конфигурации. Некоторые флаги вызывают немедленную остановку процесса аутентификации. Флаги просто управляют тем, как результаты состояния PAM объединяются в единый общий результат. В таблице показано, как объединяются эти результаты состояний.

Вы должны знать, что PAM возвращают не только состояния успеха или неудачи, но и гораздо больше других результатов состояния. Например, модуль может возвращать код состояния `PAM_ACCT_EXPIRED`, что означает, что срок действия учетной записи пользователя истек. Это состояние относится к неудачному.

Модули PAM

PAM на самом деле представляет собой набор модулей общей библиотеки (DLL-файлов), хранящихся в файле `/usr/lib64/security` (64-разрядный). Вы можете просмотреть список установленных в вашей системе модулей PAM, введя команду `ls /usr/lib64/security/pam*.so`.

ПРИМЕЧАНИЕ

В дистрибутиве Ubuntu, чтобы найти имеющиеся в системе модули, введите команду sudo find / -name pam*.so.

Система Linux включает в себя много необходимых модулей PAM, которые устанавливаются по умолчанию. Если вам действительно нужен неустановленный модуль, скорее всего, его уже создал кто-то другой. Найти различные модули можно здесь:

- www.openwall.com/pam;
- puszcza.gnu.org.ua/software/pam-modules/download.html;
- ресурсы с общими сведениями о файлах конфигурации системных событий PAM.

До сих пор основное внимание уделялось приложениям с поддержкой PAM и их файлам конфигурации. Однако другие системные события, например вход в систему Linux, также используют модули PAM. То есть и эти события имеют свои файлы конфигурации.

Далее приведена часть списка файлов конфигурации PAM. Обратите внимание на то, что существуют файлы конфигурации приложений с поддержкой PAM, например `crond`, и файлы конфигурации системных событий, такие как `postlogin-ac`:

```
# ls -l /etc/pam.d
total 204
-rw-r--r--. 1 root root  272 Nov 15 10:06 atd
...
-rw-r--r--. 1 root root  232 Jan 31 12:35 config-util
-rw-r--r--. 1 root root  293 Oct 26 23:10 crond
...
-rw-r--r--. 1 root root  109 Feb 28 01:33 postlogin

...
-rw-r--r--. 1 root root  981 Feb 28 01:33 system-auth
...
```

Вы можете изменить эти файлы конфигурации системных событий для удовлетворения конкретных нужд вашей организации в том, что касается безопасности. Например, можно изменить файл `system-auth`, чтобы принудительно применить определенные ограничения для пароля.

ВНИМАНИЕ!

Неправильное изменение или удаление файлов конфигурации системных событий PAM может привести к блокировке системы. Убедитесь, что вы протестировали любые изменения в виртуальной или тестовой среде, прежде чем вносить их в рабочие серверы Linux.

Эти файлы конфигурации системных событий PAM работают точно так же, как файлы конфигурации приложений с поддержкой PAM. Они имеют один и тот же формат, используют один и тот же синтаксис и вызывают модули PAM. Однако многие из этих файлов символически связаны (см. главу 4 «Файловая система»), поэтому они требуют выполнения дополнительных действий при внесении в них изменений. Инструкции для этого будут рассмотрены далее в этой главе.

СОВЕТ

Многие файлы конфигурации PAM имеют связанные с ними страницы руководства. Например, чтобы узнать больше о модуле pam_unix, введите man pam_unix в командной строке вашего дистрибутива Fedora и RHEL. Документация о файлах модулей находится и в каталоге /usr/share/doc/pam-*/txts/.

Несмотря на то что Linux поставляется со множеством приложений, поддерживающих PAM, различными файлами конфигурации и уже установленными модулями, нельзя просто надеяться на то, что модули будут настроены сами по себе. Для управления PAM необходимы определенные действия администратора.

Управление модулями PAM в системе Linux

Задачи администратора PAM в системе Linux минимальны. Необходимо убедиться, что PAM правильно реализованы, и внести коррективы в соответствии с потребностями безопасности конкретной организации.

Кроме того, модули не только реализуют аутентификацию, как было описано ранее. Они могут также ограничивать ресурсы и время доступа, обеспечивать правильный выбор пароля и т. д.

Управление файлами конфигурации приложений с поддержкой PAM

Вы должны просмотреть файлы конфигурации PAM для приложений и утилит, поддерживающих PAM, чтобы убедиться, что их процесс аутентификации соответствует желаемому процессу аутентификации вашей организации. Матрица контроля доступа (см. главу 22 «Базовые методы обеспечения безопасности») и информация о PAM, представленная в этой главе, должны помочь вам выполнить проверку файлов конфигурации PAM.

Каждое приложение, поддерживающее PAM, должно иметь собственный файл конфигурации PAM. Каждый файл конфигурации определяет, какие именно модули применяются для данного приложения. Если файла конфигурации не существует, то в ходе работы этого приложения может появиться дыра в безопасности, которой может воспользоваться злоумышленник. В качестве меры предосторожности PAM поставляются вместе с файлом конфигурации `other`. Если приложение, поддерживающее PAM, не имеет файла конфигурации PAM, оно по умолчанию применяет файл конфигурации `other`.

Вы можете проверить, есть ли в вашей системе Linux файл конфигурации `/etc/pam.d/other`, с помощью команды `ls`. В следующем примере показано, что файл конфигурации `/etc/pam.d/other` PAM действительно существует в этой системе:

```
$ ls /etc/pam.d/other
/etc/pam.d/other
```

Файл конфигурации PAM `/etc/pam.d/other` должен запрещать любой доступ, что с точки зрения безопасности называется `Implicit Deny` (Неявный запрет). В сфере компьютерной безопасности `Implicit Deny` означает: если определенные критерии не соблюдены явно, то доступ должен быть запрещен. В этом случае, если для приложения с поддержкой PAM не существует файла конфигурации, доступ к нему будет запрещен. Далее показано содержимое файла `/etc/pam.d/other`:

```
$ cat /etc/pam.d/other
#%PAM-1.0
auth     required         pam_deny.so
account  required         pam_deny.so
password required         pam_deny.so
session  required         pam_deny.so
```

Обратите внимание на то, что в примере перечислены все четыре контекста PAM: `auth`, `account`, `password` и `session`. Каждый контекст использует нужный флаг управления и модуль `pam_deny.so`. Модуль `pam_deny.so` применяется для запрета доступа.

Если файла конфигурации PAM для приложения нет, то, даже при наличии файла конфигурации `other`, его необходимо создать. Этот пункт входит в контрольный список аудита модулей. Вы также должны просмотреть свой файл конфигурации `other` в системе Linux, чтобы убедиться, что он применяет `Implicit Deny`.

Управление файлами конфигурации PAM

Как и файлы конфигурации приложений и утилит с поддержкой PAM, файлы конфигурации системных событий PAM должны проверяться с помощью матрицы контроля доступа вашей организации. Однако для внесения в эти файлы любых изменений необходимо предпринять дополнительные действия.

Далее вы узнаете, как в вашей системе Linux настроить с помощью PAM специальные требования безопасности, например, ограничения времени входа в учетную запись. Многие специальные настройки требуют внесения изменений в файлы конфигурации системных событий PAM, например в файл `/etc/pam.d/system-auth`.

Проблема с внесением изменений в некоторые из файлов конфигурации системных событий PAM заключается в том, что утилита `authselect` может переписать эти файлы и удалить любые локально внесенные изменения. К счастью, каждый файл конфигурации PAM, который подвергается такому риску, записывается в строке комментария внутри. Используя команду `grep`, вы можете быстро найти файлы конфигурации PAM, имеющие ту же потенциальную проблему:

```
# grep "authselect" /etc/pam.d/*
fingerprint-auth:# Generated by authselect on Mon Oct 21 19:24:36
2019
password-auth:# Generated by authselect on Mon Oct 21 19:24:36 2019
postlogin:# Generated by authselect on Mon Oct 21 19:24:36 2019
smartcard-auth:# Generated by authselect on Mon Oct 21 19:24:36 2019
system-auth:# Generated by authselect on Mon Oct 21 19:24:36 2019
```

Эти файлы конфигурации системных событий PAM используют символические ссылки (см. главу 4 «Файловая система»). Например, в дистрибутиве Fedora вы можете видеть, что файл `system-auth` на самом деле является символической ссылкой, указывающей на файл `/etc/authselect/system-auth`. Первый символ в правах безопасности файла — это буква `l`. Это говорит о том, что файл связан с другим. Символ `->` показывает, что файл символически связан с другим файлом:

```
# ls -l system-auth
lrwxrwxrwx. 1 root root 27 Oct 1 15:24 system-auth -> /etc/
authselect/system-auth
```


ПРИМЕЧАНИЕ

В некоторых дистрибутивах Linux утилита pam-auth-config похожа на утилиту authselect, так как тоже умеет перезаписывать файлы конфигурации. Это может произойти, если в командной строке ввести команду pam-auth-config --force. Прочтите справочную страницу man pam-auth-config, чтобы больше узнать об этой утилите, если она установлена в вашей системе.

Ограничение ресурсов с помощью модулей PAM

Управление ресурсами — это задача не только системного администрирования, но и обеспечения безопасности. Установка ограничений ресурсов в системе Linux поможет вам избежать многих неблагоприятных ситуаций. Например, действие таких вредоносных программ, как fork-бомбы, можно предотвратить, ограничив количество процессов, которые может создать один пользователь. Fork-бомба возникает, когда процесс рекурсивно порождает один процесс за другим до тех пор, пока не будут израсходованы все системные ресурсы. Fork-бомбы могут быть злонамеренными или случайными, то есть возникшими из-за некачественной разработки программного кода.

Модуль PAM `pam-limits` для установки ограничений ресурсов использует специальный файл конфигурации `/etc/security/limits.conf`. По умолчанию он не имеет установленных ограничений ресурсов. Поэтому вам необходимо просмотреть его и установить такие ограничения в соответствии с потребностями безопасности вашей организации.

ПРИМЕЧАНИЕ

Файлы конфигурации PAM находятся в каталогах /etc/pam.d и /etc/security.

В следующем примере показан файл `/etc/security/limits.conf`. Он хорошо закомментирован. Необходимо прочитать содержимое этого файла, где подробно описан формат и приведены примеры ограничений, которые можно установить:

```
$ cat /etc/security/limits.conf
# /etc/security/limits.conf
#
#This file sets the resource limits for the users logged in via PAM.
#It does not affect resource limits of the system services.
#
#Also note that configuration files in /etc/security/limits.d
directory,
#which are read in alphabetical order, override the settings in this
#file in case the domain is the same or more specific.
...
#Each line describes a limit for a user in the form:
#
#<domain>        <type>  <item>  <value>
...
#*               soft    core            0
#*               hard    rss             10000
#@student        hard    nproc           20
#@faculty        soft    nproc           20
#@faculty        hard    nproc           50
#ftp             hard    nproc           0
#@student        -       maxlogins       4
# End of file
```

Элементы формата *domain* (домен) и *type* (тип) нуждаются в дополнительном объяснении вдобавок к задокументированному в файле конфигурации.

- `domain` (домен). Ограничение применяется к указанным пользователю или группе, а если в значении домена указан символ *, то ко *всем* пользователям.
- `type` (тип). Предел `hard` не может быть превышен. Предел `soft` может быть превышен, но только временно.

Посмотрите на настройки файла `limits.conf` в примере. В списке указана группа преподавателей `faculty`, однако нужно приглядеться к элементу `nproc`. Ограничение `nproc` устанавливает максимальное количество процессов, которые может запустить пользователь. Именно эта настройка предотвращает fork-бомбу. Обратите внимание на то, что тип `type` установлен в предел `hard`, таким образом, предел в 50 процессов не может быть превышен. Конечно, это ограничение не применяется, потому что строка закомментирована символом #:

```
#@faculty           hard       nproc            50
```

Настройки предела устанавливаются для каждого входа в систему и действуют лишь в течение всего сеанса входа. Злоумышленник может войти в систему несколько раз, чтобы создать fork-бомбу. Таким образом, стоит также установить максимальное количество входов для учетных записей пользователей.

Максимальное количество входов можно задать для каждого пользователя по отдельности. Например, пользователю `johndoe` можно войти в систему Linux только один раз. Чтобы другие не могли задействовать учетную запись `johndoe`, установите `maxlogins` его учетной записи в значение `1`:

```
johndoe           hard       maxlogins             1
```

Чтобы переопределить любые настройки в файле `limits.conf`, добавьте файлы с именем `.conf` в каталог `/etc/security/limits.d`.

Это удобный способ управлять файлом RPM или другим для добавления и удаления ограничений, не редактируя сам файл `limits.conf`.

Последним шагом в установке ограничений для этого ресурса является обеспечение того, чтобы модуль, использующий файл `limits.conf`, был включен в один из файлов конфигурации системных событий PAM. Модуль PAM, задействующий `limits.conf`, называется `pam_limits`. В приведенном далее примере команда `grep` проверяет, применяется ли PAM в файлах конфигурации системных событий:

```
# grep "pam_limits" /etc/pam.d/*
/etc/pam.d/fingerprint-auth:session required     pam_limits.so
/etc/pam.d/password-auth:session    required     pam_limits.so
/etc/pam.d/runuser:session          required     pam_limits.so
/etc/pam.d/system-auth:session      required     pam_limits.so
```

Ограничения по времени доступа к службам и учетным записям не обрабатываются файлом конфигурации PAM `/etc/security/limits.conf`. Вместо этого они обрабатываются файлом `time.conf`.

Временные ограничения с помощью модулей PAM

Модули PAM могут заставить всю вашу систему Linux работать в режиме PAM. Временны́е ограничения, такие как доступ к определенным приложениям в определенное время дня или разрешение входить в систему только в определенные дни недели, обрабатываются с помощью модулей.

Файл конфигурации PAM, который обрабатывает эти ограничения, находится в каталоге `/etc/security`. В следующем примере показан хорошо закомментированный файл конфигурации PAM `/etc/security/time.conf`:

```
$ cat /etc/security/time.conf
# this is an example configuration file for the pam_time module. Its
# syntax was initially based heavily on that of the shadow package
(shadow-960129).
```

```
#
# the syntax of the lines is as follows:
#
#       services;ttys;users;times
...
```

Я советую вам ознакомиться с содержимым файла `time.conf`. Обратите внимание на то, что формат каждой допустимой записи соответствует синтаксису: `services;ttys;users;times`. Поля разделяются точкой с запятой. Допустимые значения полей задокументированы в файле конфигурации `time.conf`.

Хотя в файле `time.conf` много комментариев, всегда полезно изучить все на примере. Например, вы решили, что обычным пользователям следует разрешать использовать терминалы только в будние дни (с понедельника по пятницу). Они могут войти в систему с 7 утра до 7 вечера по будням. В списке далее указано, какие элементы необходимо установить:

- `services` — учетная запись;
- `ttys–*` — включает все терминалы;
- `users` — включает всех, кроме суперпользователя (`!root`);
- `times` — позволяет вход по будням (`Wd`) с 7 утра (`0700`) до 7 вечера (`1900`).

Такая запись в файле `time.conf` будет выглядеть следующим образом:

```
login; * ; !root ; Wd0700-1900
```

В конце нужно убедиться, что модуль PAM, применяющий файл `time.conf`, включен в один из файлов конфигурации системных событий PAM. Модуль PAM, задействующий `time.conf`, называется `pam_time`. В примере далее команда `grep` отображает модуль PAM, `pam_time` не используется ни в одном из файлов конфигурации системных событий:

```
# grep "pam_time" /etc/pam.d/*
config-util:auth              sufficient   pam_timestamp.so
config-util:session           optional     pam_timestamp.so
```

Поскольку `pam_time` не указан в списке, вы должны изменить файл `/etc/pam.d/system-auth`, чтобы PAM мог применить временны́е ограничения. Файл конфигурации PAM `system-auth` используется модулями при входе в систему и при изменении пароля. Этот файл конфигурации проверяет множество элементов, включая временны́е ограничения.

Добавьте следующее в верхней части раздела `account` файла конфигурации:

```
account    required    pam_time.so
```

Теперь модуль `pam_time` проверяет ограничения входа, установленные вами в файле `/etc/security/time.conf`.

ПРИМЕЧАНИЕ

В дистрибутиве Ubuntu вам необходимо использовать файл /etc/pam.d/common-auth вместо файла system-auth.

Помните, что `system-auth` — это символически связанный файл. Если вы изменяете его, то должны предпринять дополнительные действия, чтобы сохранить изменения из утилиты `authselect`. Можете задействовать дополнительные модули и файлы конфигурации, чтобы установить еще больше ограничений. Одним из важных модулей безопасности является `pam_cracklib`.

Использование надежных паролей с помощью PAM

При изменении пароля в процесс включается модуль PAM `pam_cracklib`. Модуль запрашивает у пользователя пароль и проверяет его надежность по системному словарю и набору правил для выявления неудачных вариантов.

ПРИМЕЧАНИЕ

Модуль pam_cracklib установлен по умолчанию в дистрибутивах Fedora и RHEL. Для систем Ubuntu Linux он не устанавливается по умолчанию. Поэтому, чтобы получить доступ к модулю pam_cracklib в Ubuntu, выполните команду sudo apt-get install libpam-cracklib.

Используя модуль `pam_cracklib`, вы можете проверить новый пароль следующими вопросами.

- Это словарное слово?
- Это палиндром?
- Это старый пароль с измененным регистром?
- Он похож на старый пароль?
- Он слишком короткий?
- Это перевернутая версия старого пароля?
- Применяет ли он одни и те же последовательности символов?
- Содержит ли он имя пользователя в какой-то форме?

Чтобы изменить правила, применяемые `pam_cracklib` для проверки новых паролей, внесите изменения в файл `/etc/pam.d/system-auth`. Можно подумать, что изменения должны быть внесены в файл конфигурации PAM `passwd`. Однако файл `/etc/pam.d/passwd` включает файл `system-auth` в свой стек:

```
# cat /etc/pam.d/passwd
#%PAM-1.0
# This tool only uses the password stack.
password   substack   system-auth
-password   optional        pam_gnome_keyring.so use_authtok
password   substack   postlogin
```

ПРИМЕЧАНИЕ

В дистрибутиве Ubuntu вам нужно изменить файл /etc/pam.d/common-password вместо файла конфигурации system-auth.

В примере далее показаны текущие настройки файла `system-auth`. Сейчас одна запись вызывает модуль `pam_cracklib` PAM:

```
# cat /etc/pam.d/system-auth
#%PAM-1.0
# Generated by authselect on Mon Oct 21 19:24:36 2019
# Do not modify this file manually.
auth        required     pam_env.so
auth        required     pam_faildelay.so delay=2000000
auth        sufficient   pam_fprintd.so
...
auth        sufficient   pam_unix.so nullok try_first_pass
auth        requisite    pam_succeed_if.so uid >= 1000 quiet_success
auth        required     pam_deny.so
auth        sufficient   pam_sss.so forward_pass
auth        required     pam_deny.so
account     required     pam_unix.so
account     sufficient   pam_localuser.so
account     sufficient   pam_succeed_if.so uid < 1000 quiet
account     [default=bad success=ok user_unknown=ignore] pam_sss.so
account     required     pam_permit.so
password    requisite    pam_cracklib.so try_first_pass retry=3
```

Запись `pam_cracklib` здесь использует ключевое слово `retry`. Для `cracklib` доступны следующие ключевые слова.

- `debug`. Заставляет модуль записывать информацию в системный журнал.
- `authtok_type=XXX`:
  - по умолчанию используется `New UNIX password:` и `Retype UNIX password:` для запроса паролей;
  - замените `XXX` нужным словом.
- `retry=`*N*:
  - по умолчанию значение 1;
  - запрашивает пользователя не более *N* раз, прежде чем вывести ошибку.
- `difok=`*N*:
  - по умолчанию значение 5;
  - количество символов в новом пароле, которое не должно повторяться в старом пароле;
  - *исключение 1*: если половина символов в новом пароле отличаются от содержащихся в старом, то новый пароль принимается;
  - *исключение 2*: см. кодовое слово `difignore`.
- `difignore=`*N*:
  - по умолчанию значение 23;
  - количество символов в пароле перед настройкой `difok` игнорируется.

- `minlen=N`:
  - по умолчанию значение 9;
  - минимальный допустимый размер нового пароля;
  - посмотрите, как настройки ключевых слов `dcredit`, `ucredit`, `lcredit` и `ocredit` влияют на `minlen`.
- `dcredit=N`:
  - по умолчанию значение 1;
  - если $N \geq 0$ — максимальное количество цифр в новом пароле. Если установлено N цифр или меньше, к каждой из них прибавляется 1 для достижения текущего значения `minlen`;
  - если $N < 0$ — минимальное количество цифр, которое должно быть установлено для нового пароля.
- `ucredit=N`:
  - по умолчанию значение 1;
  - если $N \geq 0$ — максимальное количество прописных букв в новом пароле. Если установлено N прописных букв или меньше, к каждой из них прибавляется 1 для достижения текущего значения `minlen`;
  - если $N < 0$ — минимальное количество прописных букв, которое должно быть установлено для нового пароля.
- `lcredit=N`:
  - по умолчанию значение 1;
  - если $N \geq 0$ — максимальное количество букв в новом пароле. Если установлено N строчных букв или меньше, к каждой из них прибавляется 1 для достижения текущего значения `minlen`;
  - если $N < 0$ — минимальное количество строчных букв, которое должно быть установлено для нового пароля.
- `ocredit=N`:
  - по умолчанию значение 1;
  - если $N \geq 0$ — максимальное количество других символов в новом пароле. Если установлено N других символов или меньше, к каждой цифре прибавляется 1 для достижения текущего значения `minlen`;
  - если $N < 0$ — минимальное количество других символов, которое должно быть установлено для нового пароля.
- `minclass=N`:
  - по умолчанию значение 0;
  - для нового пароля требуется N символов четырех классов. Четыре класса — это цифры, прописные буквы, строчные буквы и другие символы.

- `maxrepeat=`*N*:
  - по умолчанию значение 0;
  - отклоняет пароли, которые содержат больше чем *N* последовательных символов.
- `reject_username`. Проверяет, содержится ли имя пользователя в каком-либо виде в новом пароле. Если имя найдено, пароль будет отклонен.
- `try_first_pass`. Пробует получить пароль от предыдущего модуля PAM. Если это не сработает, запросите пароль у пользователя.
- `use_authtok`. Этот аргумент применяется для того, чтобы *заставить* модуль не запрашивать у пользователя новый пароль. Вместо этого новый пароль предоставляется модулем *password*.
- `dictpath=/path`. Путь к словарям `cracklib`.
- `maxsequence=`*N*:
  - по умолчанию значение 0, то есть отключено;
  - значение *N* устанавливает любое число, отличное от 0, и отклоняет пароли с похожими символами длиннее этого числа.
- `maxclassrepeat=`*N*:
  - по умолчанию значение 0, то есть отключено;
  - значение *N* — это любое число, отличное от 0; отклоняются пароли, в которых подряд стоит большее количество символов одного класса.
- `gecoscheck=`*N*. Приводит к тому, что пароли с более чем тремя символами из поля GECOS учетной записи пользователя, содержащего, как правило, реальные имена, отклоняются.
- `enforce_for_root` *N*:
  - по умолчанию значение 0, то есть отключено;
  - значение *N* — это любое число, отличное от 0; отклоняются пароли, в которых подряд стоит большее количество символов одного класса.
- `enforce_for_root`. Проверяет пароль суперпользователя. Функция отключена по умолчанию.

Например, если в вашей организации требуется, чтобы пароли были длиной десять символов и содержали две цифры, вы должны добавить в файл `/etc/pam.d/system-auth` следующую строку:

```
password required pam_cracklib.so minlen=10 dcredit=-2
```

Ключевые слова, используемые в этом примере с `pam_cracklib`:

- `minlen=10` — новый пароль должен содержать как минимум десять символов;
- `dcredit=-2` — новый пароль должен содержать две цифры.

ПРИМЕЧАНИЕ

Ограничения pam_cracklib не применяются к суперпользователю, если вы не задействуете параметр enforce_for_root.

Использование команды sudo вместе с модулями PAM

Чтобы разрешить отслеживание применения учетной записи суперпользователя отдельными лицами (см. главу 22 «Базовые методы обеспечения безопасности»), вы должны ограничить использование команды `su` и поощрять использование команды `sudo`. Если в вашей организации такая политика, можете сделать это с помощью PAM всего за несколько шагов.

Команда `su` может использовать модули PAM, что значительно упрощает процесс. Она задействует модуль PAM `pam_wheel` для проверки пользователей в группе `wheel`. Здесь показан файл конфигурации `/etc/pam.d/su`:

```
# cat /etc/pam.d/su
#%PAM-1.0
auth        required      pam_rootok.so
auth        sufficient    pam_rootok.so
# Uncomment the following line to implicitly trust users
# in the "wheel" group.
#auth        sufficient  pam_wheel.so trust use_uid
# Uncomment the following line to require a user to be
# in the "wheel" group.
#auth       required      pam_wheel.so use_uid
auth        substack      system-auth
auth        include       postlogin
account     sufficient    pam_succeed_if.so uid = 0 use_uid quiet
account     include       system-auth
password    include       system-auth
session     include       system-auth
session     include       postlogin
session     optional      pam_xauth.so
```

Поначалу, чтобы ограничить использование команды `su`, если вы задействуете группу `wheel` в качестве административной, необходимо переназначить свою группу (см. главу 11 «Управление учетными записями»). Если вы не используете группу `wheel`, просто не назначайте никого в нее в будущем.

Далее нужно отредактировать файл конфигурации `/etc/pam.d/su`. Удалите знак комментария # из следующей строки:

```
#auth           required           pam_wheel.so use_uid
```

После таких изменений модуль PAM отключает использование команды. Администраторы теперь должны применять команду `sudo`, которую система отслеживает и для которой обеспечивает желаемую среду (см. главу 22).

Дополнительная информация о модулях PAM

Модули PAM — это универсальный инструмент безопасности, доступный в системе Linux. На справочных страницах Linux вы можете прочитать об управлении файлами конфигурации PAM и о модулях в каталоге `/usr/lib64/security` (64-bit).

- Чтобы получить дополнительную информацию о файлах конфигурации PAM, используйте команду `man pam.conf`.
- Вы можете просмотреть все модули, доступные в системе, введя команду `ls /usr/lib64/security/pam*.so`. Чтобы получить более подробную информацию о каждом модуле, введите `man pam_module_name`. Обязательно добавьте расширение файла `so` к названию `pam_module_name`. Например, введите `man pam_lastlog`, чтобы узнать больше о модуле `pam_lastlog.so`. Сайты, на которых можно получить дополнительную информацию о модулях PAM:
  - официальный сайт Official Linux-PAM: linux-pam.org;
  - сайт Linux-PAM System Administrator's Guide: linux-pam.org/ Linux-PAM-html/ Linux-PAM_SAG.html;
  - сайт PAM Module: linux-pam.org/Linux-PAM-html/sag- module-reference.html.

Резюме

Инструменты криптографии предлагают способы защиты и проверки достоверности данных, которые применяются в системе Linux. Модули PAM представляют собой средства создания политик для защиты инструментов, с помощью которых происходит аутентификация пользователей в системе.

Как с инструментами криптографии, так и с модулями PAM следует обращаться осторожно. Обязательно тестируйте любые изменения в тестовой или виртуализированной системе Linux, прежде чем внедрять их в основную систему.

В следующей главе мы разберем систему SELinux. Если криптография и модули PAM — это инструменты, которые вы можете использовать в своей системе Linux, то SELinux — это целая система повышения уровня безопасности.

Упражнения

С помощью этих упражнений проверьте свои знания об инструментах криптографии и модулях PAM. Задачи подходят для систем Fedora или Red Hat Enterprise Linux (некоторые будут работать и в других системах Linux). Если затрудняетесь с решением заданий, воспользуйтесь ответами к упражнениям, приведенными в приложении Б (хотя Linux позволяет решать задачи разными способами).

1. Зашифруйте файл с помощью утилиты `gpg2` и симметричного ключа.
2. Создайте связку открытых ключей с помощью утилиты `gpg2`.

3. Перечислите все ключи, относящиеся к только что созданной связке ключей.
4. Зашифруйте файл и добавьте свою цифровую подпись с помощью утилиты gpg2.
5. Перейдите на страницу загрузки дистрибутива Fedora getfedora.org. Выберите подходящий дистрибутив. После окончания загрузки подтвердите его образ.
6. Используя команду `which su`, определите полное имя файла команды. Затем определите, является ли команда в вашей системе Linux подходящей для модулей PAM.
7. Есть ли у команды файл конфигурации PAM? Если это так, выведите его на экран и перечислите контексты PAM, которые он использует.
8. Выведите на экран список модулей, имеющихся в вашей системе.
9. Найдите файл конфигурации PAM `other` в своей системе. Существует ли он? Применяет ли он механизм `Implicit Deny`?
10. Есть ли у него настройки, предотвращающие появление fork-бомбы в вашей системе?

24 Повышенная безопасность с технологией SELinux

В этой главе

- Преимущества SELinux.
- Работа SELinux.
- Настройка SELinux.
- Устранение неполадок с SELinux.
- Дополнительная информация о SELinux.

Технология *Security Enhanced Linux (SELinux)* была разработана Агентством национальной безопасности (National Security Agency, NSA) совместно с другими исследовательскими организациями в области безопасности, включая Secure Computing Corporation (SCC). SELinux была выпущена для сообщества с открытым исходным кодом в 2000 году и стала популярной, когда компания Red Hat включила ее в свои дистрибутивы Linux. Сейчас SELinux широко доступна и используется множеством организаций.

Преимущества SELinux

SELinux — это модуль повышения безопасности, развернутый над Linux. Он обеспечивает дополнительные меры безопасности, включен по умолчанию и настроен на принудительный режим в Red Hat Enterprise Linux (RHEL) и Fedora.

SELinux обеспечивает повышенную безопасность в системе Linux с помощью управления доступом на основе ролей (RBACs) к субъектам и объектам (они же процессы и ресурсы). Традиционная безопасность Linux использует избирательное управление доступом (Discretionary Access Controls, DAC).

С помощью управления DAC процесс может получить доступ к любому файлу, каталогу, устройству или другому ресурсу, который остается открытым для досту-

па. С помощью RBAC процесс имеет доступ только к тем ресурсам, к которым он явно разрешен в зависимости от назначенной роли. Здесь кстати то, что SELinux реализует управление доступом на основе ролей, чтобы назначить политику SELinux процессам. Эта политика ограничивает доступ следующим образом.

- Позволяет процессу получить доступ к ресурсам, у которых есть только явные метки.
- Предотвращает потенциально небезопасные функции (например, запись в каталог), доступные в виде логических значений, которые могут быть включены или выключены.

Служба, например веб-сервер, включающая политику SELinux, часто устанавливается с метками SELinux, настроенными для определенных каталогов и файлов. Это может привести к тому, что запущенный процесс сервера может читать файлы только из определенных каталогов и записывать в них же. Если вы хотите это изменить, нужно добавить правильные метки SELinux на файлы и каталоги, к которым вы хотите получить доступ.

Технология SELinux не является заменой управления доступом DAC, она представляет собой дополнительный уровень безопасности.

- Правила DAC применяются и при использовании SELinux.
- Сначала проверяются правила DAC, а если доступ разрешен, то проверяются политики SELinux.
- Если правила DAC запрещают доступ, политики SELinux не рассматриваются.

Если пользователь пытается выполнить файл, к которому у него нет доступа (`rw-`), традиционные элементы управления DAC Linux запрещают доступ. Таким образом, политики SELinux даже не проверяются.

ПРИМЕЧАНИЕ

Система SELinux — это повышенная безопасность по умолчанию для дистрибутивов Red Hat, а как система AppArmor — это повышенная безопасность по умолчанию для Ubuntu. Вы можете установить SELinux на Ubuntu с помощью команды sudo apt-get install selinux, а затем перезагрузить систему. Однако на момент написания этой книги вики-страница Ubuntu для SELinux предполагает, что вы не используете пакет SELinux Ubuntu (wiki.ubuntu.com/SELinux). Если вы хотите больше узнать об AppArmor, перейдите на страницу help.ubuntu.com/community/AppArmor.

Несмотря на то что традиционные средства управления безопасностью Linux все еще работают, использование SELinux имеет ряд преимуществ. Далее приведены некоторые преимущества SELinux.

- **Система реализует модель контроля доступа RBAC.** Она считается самой сильной моделью контроля доступа.

- **Система задействует доступ с наименьшими привилегиями для субъектов, например пользователей и процессов.** Принцип *минимальных привилегий* означает, что каждому субъекту предоставляется ограниченный набор привилегий, достаточных для того, чтобы он мог выполнять свои задачи. При реализации этого принципа пользователь или процесс рискуют случайно (или преднамеренно) повредить объекты.
- **Система позволяет обрабатывать «песочницу».** Термин *«песочница»* означает, что каждый процесс выполняется в собственной области. Он не может получить доступ к другим процессам или их файлам, если не будут предоставлены специальные права. Области, в которых выполняются процессы, называются доменами.
- **Система позволяет проверить всю функциональность перед началом работы.** У SELinux есть разрешительный режим, который позволяет увидеть эффект применения SELinux в системе. В разрешительном режиме SELinux по-прежнему регистрирует то, что посчитает нарушениями безопасности (отказы AVC), но не предотвращает их.

Еще один способ взглянуть на преимущества SELinux — это изучить последствия того, что в вашей системе Linux не работает SELinux. Так, в примере с веб-сервером демон веб-сервера (`httpd`) прослушивает порт, ожидая, что что-то произойдет. Из браузера приходит простой запрос на просмотр домашней страницы. Реализуя обычный порядок действий, демон `httpd` видит запрос, в это время применяется только традиционная безопасность Linux. Будучи неограниченной системой SELinux, служба `httpd` может:

- дать доступ к *любому* файлу или каталогу на основе прав на чтение/запись/выполнение для связанного владельца и группы;
- выполнять потенциально небезопасные действия, например давать права на загрузку файла или изменение системных ограничений;
- прослушивать любой порт в поиске входящих запросов.

В системе, ограниченной SELinux, демон `httpd` управляется гораздо жестче. Как и в предыдущем примере, служба `httpd` может прослушивать только тот порт, на котором SELinux позволяет ему прослушивать. SELinux запрещает `httpd` доступ к любому файлу, который не имеет соответствующего контекста безопасности, и запрещает потенциально небезопасные действия, которые явно не включены с помощью логических типов. По сути, система SELinux строго ограничивает то, к чему вредоносный код может получить доступ, и вообще ограничивает активность в системе Linux.

Как работает система SELinux

Систему SELinux можно сравнить с охранником у двери, когда субъект (пользователь) хочет получить доступ к объекту (файлу) внутри комнаты. Условия получения доступа к этому объекту следующие.

1. Субъект должен предъявить охраннику значок ID.
2. Охранник должен просмотреть значок ID и правила доступа, хранящиеся в большом руководстве.
   - Если правила разрешают этому значку ID войти в комнату, субъект может сделать это, чтобы получить доступ к объекту.
   - Если правила доступа не позволяют этому значку ID получить доступ к объекту, то охранник не дает субъекту пройти.

SELinux обеспечивает комбинацию управления доступом на основе ролей (RBAC), *многоуровневой безопасности* (multi-level security, MLS) и модели *принудительной типизации доступа* (type enforcement, TE). При управлении доступом на основе ролей доступ к объекту определяется назначенной субъекту ролью в организации. Поэтому доступ не основан на имени пользователя субъекта или идентификаторе процесса. Каждой роли предоставляются права доступа.

Принудительная типизация доступа

Принудительная типизация доступа необходима для реализации модели RBAC. Она обеспечивает безопасность системы с помощью следующих методов:

- маркировки объектов как определенных типов безопасности;
- назначения субъектов определенным доменам и ролям;
- предоставления правил, позволяющих определенным доменам и ролям получать доступ к определенным типам объектов.

В следующем примере используется команда `ls -l` для отображения элементов управления DAC в файле `my_stuff`. Выходные данные показывают владельца файла (`johndoe`) и группу (`johndoe`), а также его назначенные права. Если вам нужна информация о правах доступа к файлам, обратитесь к главе 4 «Файловая система».

```
$ ls -l my_stuff
-rw-rw-r--. 1 johndoe johndoe 0 Feb 12 06:57 my_stuff
```

Следующий пример включает в себя команду `ls -lZ` и тот же файл `my_stuff`, но вместо элементов управления DAC параметр `-Z` также отображает элементы управления безопасностью RBAC:

```
$ ls -lZ my_stuff
-rw-rw-r--. johndoe johndoe unconfined_u:object_r:user_home_t:s0
... my_stuff
```

В примере `ls -Z` отображаются четыре связанных с файлом элемента, которые относятся к SELinux:

- `user` (`unconfined_u`);
- `role` (`object_r`);
- `type` (`user_home_t`);
- `level` (`s0`).

Эти четыре элемента RBAC (пользователь, роль, тип и уровень) применяются в управлении доступом SELinux для определения соответствующих уровней доступа. Вместе эти элементы называются *контекстом безопасности* SELinux. Контекст безопасности (значок ID) иногда называют меткой безопасности.

Элементы контекста безопасности присваиваются субъектам (процессам и пользователям). Каждый контекст безопасности имеет имя. Оно зависит от того, какому объекту или субъекту присвоено. То есть файлы имеют контекст файла, пользователи имеют контекст пользователя, а процессы имеют контекст процесса, также называемый доменом.

Правила, разрешающие доступ, называются разрешающими правилами или правилами политики. *Правило политики* — это процесс, которого SELinux придерживается, чтобы предоставить или запретить доступ к определенному типу безопасности системы. Возвращаясь к сравнению, SELinux служит охранником, который должен увидеть контекст безопасности субъекта (значок ID) и просмотреть правила политики (руководство по правилам доступа), прежде чем разрешить или запретить доступ к объекту. Таким образом, принудительная типизация доступа гарантирует, что только определенные типы субъектов могут получить доступ к определенным типам объектов.

Многоуровневая модель безопасности

В SELinux тип политики по умолчанию называется *целевой политикой*, которая в основном управляет доступом к сетевым службам, таким как веб-серверы и файловые серверы, в системе Linux. Целевая политика накладывает меньше ограничений на то, что допустимые учетные записи пользователей могут делать в системе. Для более ограниченной политики можно выбрать модель многоуровневой безопасности (Multi-Level Security, MLS). MLS применяет принудительную типизацию доступа наряду с дополнительной функцией прав безопасности. Она также обеспечивает мультикатегорийную безопасность, которая назначает уровни классификации объектам.

СОВЕТ

Названия многоуровневой безопасности (MLS) могут вызвать путаницу. Безопасность Multi-Category Security (MCS) иногда называют безопасностью Multi-Clearance Security. Поскольку MLS предлагает MCS, его иногда называют MLS/MCS.

Многоуровневая безопасность обеспечивает соблюдение модели принудительного доступа Белла — Лападулы (Bell — LaPadula Mandatory Access). Модель Белла — Лападулы была разработана правительством США для обеспечения конфиденциальности информации. Модель применяется путем предоставления доступа к объекту на основе разрешения безопасности роли и уровня классификации объекта.

Разрешение безопасности — это атрибут, предоставляемый ролям, разрешающим доступ к классифицированным объектам. *Уровень классификации* — это атрибут, предоставляемый объекту, который обеспечивает защиту от субъектов со слишком низким атрибутом разрешения безопасности. Вы, скорее всего, слышали о совершенно секретном уровне классификации. Вымышленный персонаж книг и фильмов Джеймс Бонд имел сверхсекретный допуск, который давал ему доступ к сверхсекретной информации. Это классический пример модели Белла — Лападулы.

Сочетание RBAC вместе с принудительной типизацией доступа или многоуровневой безопасностью позволяет SELinux обеспечить максимальное повышение безопасности. В системе SELinux существуют также различные режимы работы.

Модели безопасности SELinux

Управление доступом на основе ролей, принудительная типизация доступа, многоуровневая безопасность и модели Белла — Лападулы — все это очень интересные темы. Система SELinux реализует эти модели с помощью комбинации четырех основных частей SELinux:

- режимов работы;
- контекстов безопасности;
- типов политик;
- пакетов правил политики.

Хотя мы уже коснулись некоторых из этих элементов, в дальнейшем рассмотрим их глубже. Необходимо разобраться в них, прежде чем вы сможете начать изменять настройки SELinux в своей системе.

Режимы работы SELinux

У системы SELinux есть три режима работы: отключенный, разрешительный и принудительный. У каждого из них есть различные преимущества для безопасности системы Linux.

Отключенный режим. В *отключенном* режиме SELinux выключен. Вместо этого по умолчанию используется метод избирательного контроля доступа (Discretionary Access Control, DAC). Этот режим полезен в тех случаях, когда повышенная безопасность не требуется.

Если это возможно, Red Hat рекомендует установить SELinux в разрешительный режим, а не отключать его. Однако бывают случаи, когда отключить SELinux стоит.

Если вы запускаете приложения, которые работают правильно (с вашей точки зрения), но генерируют огромное количество сообщений об отказе SELinux AVC (даже в разрешительном режиме), то в итоге файлы журналов заполнятся до такой степени, что системы станут непригодными для работы. Лучший подход — установить надлежащий контекст безопасности для файлов, к которым ваши приложения

должны получить доступ. Тем не менее отключение SELinux — это более быстрое решение.

Однако, прежде чем отключить SELinux, подумайте о том, захотите ли вы снова включить его в этой системе. Если вы решите установить его в принудительный или разрешительный режим позже, то в следующий раз при перезагрузке системы она пройдет автоматическую повторную маркировку файла SELinux.

СОВЕТ

Если все, что вас волнует, — это отключение SELinux, то можете это сделать. Просто отредактируйте файл конфигурации /etc/selinux/config и измените текст SELINUX= на SELINUX=disabled. SELinux будет отключен после перезагрузки системы. Остальную часть этой главы можете пропустить.

Разрешительный режим. В *разрешительном* режиме SELinux включен, но правила политики безопасности не применяются. Когда правило политики безопасности должно запретить доступ, он все равно будет разрешен. Однако в файл журнала отправляется сообщение о том, что доступ должен был быть запрещен.

Разрешительный режим SELinux используется:

- для аудита текущих правил политики SELinux;
- тестирования новых приложений, чтобы увидеть, как повлияют на них правила политики SELinux;
- тестирования новых правил политики SELinux, чтобы увидеть, как они повлияют на текущие службы и приложения;
- устранения неполадок, из-за которых конкретная служба или приложение больше не работают должным образом в SELinux.

В некоторых случаях можно использовать команду `audit2allow` для чтения журналов аудита SELinux и создания новых правил SELinux, позволяющих выборочно разрешать запрещенные действия. Это быстрый способ заставить приложения работать в системе Linux, не отключая SELinux.

Принудительный режим. Название говорит само за себя. В *принудительном* режиме система SELinux включена и все правила политики безопасности применяются.

Контексты безопасности SELinux

Как упоминалось ранее, контекст безопасности SELinux — это метод классификации объектов, таких как файлы, и субъектов, таких как пользователи и программы. Определенный контекст безопасности позволяет SELinux задействовать правила политики для субъектов, обращающихся к объектам. Контекст безопасности состоит из четырех атрибутов: `user`, `role`, `type` и `level`.

- `user`. Атрибут `user` — это сопоставление имени пользователя Linux с именем SELinux. Это не то же самое, что логин пользователя, он упоминается именно

как пользователь SELinux. Имя пользователя SELinux заканчивается на букву `u`, что облегчает его идентификацию в выходных данных. Обычные пользователи без ограничений имеют атрибут пользователя `unconfined_u` в целевой политике по умолчанию.

- `role`. Назначенная роль сопоставляется с именем роли SELinux. Затем атрибут `role` присваивается различным субъектам и объектам. Каждой роли предоставляется доступ к другим субъектам и объектам в зависимости от уровня ее безопасности и уровня классификации объекта. Если точнее, для SELinux пользователям назначается роль, а роли разрешаются для определенных типов или доменов. Применение ролей может привести к тому, что учетные записи, например, суперпользователя окажутся в менее привилегированном положении. В имени роли SELinux на конце стоит буква `r`. В целевой системе SELinux процессы, запущенные суперпользователем, имеют роль `system_r`, в то время как обычные пользователи применяют роль `unconfined_r`.
- `type`. Атрибут `type` определяет тип домена для процессов, тип пользователя для пользователей и тип файла для файлов. Этот атрибут называется также типом безопасности. Большинство правил политики касаются типа безопасности процесса и того, к каким файлам, портам, устройствам и другим элементам системы этот процесс имеет доступ (на основе их типов безопасности). Имя типа SELinux заканчивается на букву `t`.
- `level`. `level` является атрибутом многоуровневой безопасности (MLS) и обеспечивает соблюдение модели Белла — Лападулы. Он необязателен при использовании TE, но обязателен при задействовании MLS.

  Уровень MLS представляет собой комбинацию значений чувствительности и категории, которые вместе образуют уровень безопасности. Уровень записывается как `sensitivity : category`.

  Значение `sensitivity`:

  - представляет уровень безопасности или чувствительности объекта, например конфиденциальный или совершенно секретный;
  - иерархично, причем уровень `s0` (неклассифицированный) обычно самый низкий;
  - отображается как пара уровней чувствительности *(lowlevel-highlevel)*, если уровни различаются;
  - указывается как единый уровень чувствительности (`s0`), если нет низких и высоких уровней. Однако в некоторых случаях, даже если нет низких и высоких уровней, диапазон все равно отображается (`s0-s0`).

  Значение `category`:

  - представляет категорию объекта, например `No Clearance`, `Top Clearance` и т. д.;
  - традиционно находится между `c0` и `c255`;
  - перечисляется как пара уровней категории (*`lowlevel.highlevel`*), если уровни различаются;

- перечисляется как единая категория (уровень), если нет низких и высоких уровней.

Контексты безопасности пользователей. Чтобы увидеть пользовательский контекст SELinux, введите команду `id` в командной строке оболочки. Далее приведен пример контекста безопасности для пользователя `johndoe`:

```
$ id
uid=1000(johndoe) gid=1000(johndoe) groups=1000(johndoe)
context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

В списке контекста безопасности пользователя находится следующее.

- `User`. Пользователь Linux `johndoe` сопоставляется с пользователем SELinux `unconfined_u`.
- `role`. Пользователь SELinux `unconfined_u` сопоставляется с ролью `unconfined_r`.
- `type`. Пользователю присваивается тип `unconfined_t`.
- `level`:
  - `sensitivity`. У пользователя есть только один уровень чувствительности, и он самый низкий — `s0`;
  - `categories`. Пользователь имеет доступ к `c0.c1023`, то есть ко всем категориям от `c0` до `c1023`.

Контексты безопасности файлов. Файл имеет также контекст безопасности. Чтобы увидеть контекст отдельного файла, примените параметр `-Z` в команде `ls`. Далее приведен контекст безопасности для файла `my_stuff`:

```
$ ls -Z my_stuff
-rw-rw-r--. johndoe johndoe
unconfined_u:object_r:user_home_t:s0 my_stuff
```

В списке контекста безопасности файла находится следующее.

- `user`. Файл сопоставляется с пользователем SELinux `unconfined_u`.
- `role`. Файл сопоставляется с ролью `object_r`.
- `type`. Файл считается частью домена `user_home_t`.
- `level`:
  - `sensitivity`. У пользователя есть только один уровень чувствительности, и он самый низкий — `s0`;
  - `categories`. Модель MCS для этого файла не установлена.

Контексты безопасности процессов. Контекст безопасности процесса имеет те же четыре атрибута, что и контекст пользователя и файла. Чтобы просмотреть информацию о процессе в системе Linux, обычно применяется вариант команды `ps`. В следующем примере задействована команда `ps -el`:

```
# ps -el | grep bash
0 S  1000  1589  1583  0  80   0 -  1653 n_tty_ pts/0     00:00:00
```

```
bash
0 S  1000  5289  1583  0  80   0 -  1653 wait   pts/1     00:00:00
bash
4 S     0  5350  5342  0  80   0 -  1684 wait   pts/1
00:00:00 bash
```

Чтобы увидеть контекст безопасности процесса, нужно использовать параметр `-Z` в команде `ps`. В следующем примере команда `ps -eZ` была передана в `grep` для поиска процессов, выполняющих оболочку `bash`:

```
# ps -eZ | grep bash
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 1589 pts/0
00:00:00 bash
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 5289 pts/1
00:00:00 bash
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 5350 pts/1
00:00:00 bash
```

В списке контекста безопасности процесса находится следующее.

- `user`. Процесс сопоставляется с пользователем SELinux `unconfined_u`.
- `role`. Процесс выполняется как роль `unconfined_r`.
- `type`. Процесс выполняется в домене `unconfined_t`.
- `level`:
  - `sensitivity`. У процесса только один уровень чувствительности `s0`;
  - `categories`. Процесс имеет доступ к `c0.c1023`, то есть ко всем категориям от `c0` до `c1023`.

Все эти контексты безопасности могут быть изменены в соответствии с конкретными потребностями вашей организации в области безопасности. Однако, прежде чем узнать, как изменить настройки этих контекстов безопасности, вам нужно разобраться еще в одной части системы SELinux — типах политик SELinux.

Типы политик в системе SELinux

Выбранный *тип политики* напрямую определяет, какие наборы правил политики используются для определения того, к чему объект может получить доступ. Тип политики также определяет, какие конкретные атрибуты контекста безопасности необходимы. Именно здесь появляется основной уровень контроля доступа, который может быть реализован с помощью SELinux.

ПРИМЕЧАНИЕ

Типы политик, доступные в вашем дистрибутиве, могут не совпадать с перечисленными далее. Например, в старых дистрибутивах Linux применяется более строгая политика. В новых дистрибутивах строгая политика объединена с политикой Targeted, используемой по умолчанию.

SELinux имеет различные политики, среди которых вы можете выбрать:

- Targeted;
- MLS;
- Minimum.

Каждая политика реализует различные элементы управления доступа в соответствии с потребностями вашей организации. Очень важно изучить эти типы политик, чтобы выбрать подходящую для конкретных требований безопасности.

Политика Targeted. Основная цель политики Targeted — это ограничение целевых демонов. Однако она может ограничивать и другие процессы и пользователей. Целевые демоны находятся в «песочнице». «Песочница» — это среда, в которой программы могут работать, но их доступ к другим объектам строго контролируется.

Процесс, работающий в такой среде, также называется «песочницей». Таким образом, целевой демон ограничен, так что никакие вредоносные атаки, запущенные через него, не могут повлиять на другие службы или систему Linux в целом. Целевые демоны делают более безопасным совместное использование сервера печати, файлового сервера, веб-сервера или других служб, ограничивая при этом риски, связанные с доступом к этим службам, для других ресурсов вашей системы.

Все субъекты и объекты, не являющиеся целевыми, запускаются в домене `unconfined_t`. Домен `unconfined_t` не имеет ограничений политики SELinux и, таким образом, использует только традиционную безопасность Linux.

В системе SELinux политика Targeted установлена по умолчанию. Таким образом, по умолчанию SELinux нацелена лишь на несколько демонов.

Политика MLS (Multi-Level Security). Основная цель политики MLS — обеспечить соблюдение модели Белла — Лападулы. Она предоставляет доступ к другим субъектам и объектам на основе разрешения безопасности роли и *уровня классификации* объекта.

В политике MLS атрибут MLS контекста безопасности имеет решающее значение. В противном случае правила политики не будут знать, как применять ограничения доступа.

Политика Minimum. Как следует из названия, данная политика сама по себе минимальна. Первоначально она была создана для машин или устройств с небольшим количеством памяти, таких как смартфоны.

Политика Minimum, по существу, схожа с политикой Targeted, но использует только пакет правил базовой политики. Она, как скелет, может быть применена для проверки влияния SELinux на одного назначенного демона. Политика Minimum позволяет SELinux работать на устройствах с малым объемом памяти, не потребляя значительных ресурсов.

Пакеты правил политики в системе SELinux

Правила политики, называемые также *разрешающими правилами*, — это правила, задействуемые SELinux для определения того, имеет ли субъект доступ к объекту.

Правила политики устанавливаются вместе с SELinux и группируются в пакеты, называемые также *модулями*.

В системе Linux имеется пользовательская документация по различным модулям политики в виде HTML-файлов. Чтобы просмотреть эту документацию на Fedora или RHEL, откройте браузер и введите следующий URL-адрес: file:///usr/share/doc/selinux-policy/html/index.html. Для Ubuntu URL-адрес такой: file:///usr/share/doc/selinux-policy-doc/html/index.html. Если у вас нет документации по политике вашей системы, можете установить ее в Fedora или RHEL, набрав команду `yum install selinux-policy-doc`. В Ubuntu введите в командной строке `sudo apt-get install selinux-policy-doc`.

Просмотрите эту документацию, чтобы узнать, как создаются и упаковываются правила политики.

Пакеты правил политики, а также режимы работы SELinux, тип политики и различные контексты безопасности работают вместе, чтобы защитить вашу систему Linux с помощью механизма SELinux. В следующих разделах описано, как начать настройку SELinux для удовлетворения потребностей вашей организации в области безопасности.

Настройка системы SELinux

Изначально система SELinux настроена. Вы можете сразу начать использовать ее функции без каких-либо дополнительных настроек. Однако предварительно настроенные параметры редко удовлетворяют всем требованиям безопасности системы Linux.

Настройки SELinux могут быть установлены и изменены только суперпользователем. Файлы конфигурации и политики находятся в каталоге `/etc/selinux`. Основным файлом конфигурации является файл `/etc/selinux/config`, он выглядит следующим образом:

```
# cat /etc/selinux/config
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#      enforcing - SELinux security policy is enforced.
#      permissive - SELinux prints warnings instead of enforcing.
#      disabled - SELinux is fully disabled.
SELINUX=enforcing
# SELINUXTYPE= can take one of these three values:
#     targeted - Targeted processes are protected,
#     minimum - Modification of targeted policy.
#               Only selected processes are protected.
#     mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

Этот основной файл конфигурации SELinux позволяет установить режим и тип политики.

Установка режима SELinux

Чтобы определить режим SELinux в своей системе, используйте команду `getenforce`. Чтобы увидеть как текущий режим, так и режим, заданный в файле конфигурации, примените команду `sestatus`. Обе команды показаны в следующем примере:

```
# getenforce
Enforcing
# sestatus
SELinux status:                 enabled
SELinuxfs mount:                /sys/fs/selinux
SELinux root directory:         /etc/selinux
Loaded policy name:             targeted
Current mode:                   enforcing
Mode from config file:          enforcing
Policy MLS status:              enabled
Policy deny_unknown status:     allowed
Memory protection checking:     actual (secure)
Max kernel policy version:      31
```

Чтобы изменить настройку режима, используйте команду `setenforce newsetting`, где `newsetting` — это один из вариантов:

- `enforcing` или `1`;
- `permissive` или `0`.

Обратите внимание на то, что вы не можете применить команду `setenforce` для перевода SELinux в отключенный режим.

В следующем примере показано, что с помощью команды `setenforce` режим SELinux немедленно изменяется на разрешительный. Команда `sestatus` показывает текущий режим работы и режим в файле конфигурации, который не был изменен. Когда система перезагружается, она определяет режим работы SELinux из файла конфигурации. Таким образом, разрешительный режим, установленный в следующем примере, временный, поскольку в файле конфигурации установлен принудительный режим:

```
# setenforce 0
# getenforce
Permissive
# sestatus
SELinux status:                 enabled
SELinuxfs mount:                /sys/fs/selinux
SELinux root directory:         /etc/selinux
Loaded policy name:             targeted
Current mode:                   permissive
Mode from config file:          enforcing
...
```

ВНИМАНИЕ!

Лучше всего переключиться из отключенного режима в принудительный, изменив файл конфигурации и перезагрузив систему. Перевод из отключенного режима в принудительный с помощью команды setenforce может затормозить систему из-за неправильных меток файлов. Имейте в виду, что при перезагрузке после перехода из режима disabled можно долго ожидать повторной маркировки файловой системы после того, как она вернется в разрешительный или принудительный режим.

Чтобы отключить SELinux, необходимо отредактировать файл конфигурации SELinux. Перезагрузка системы всегда меняет режим на тот, который задан в этом файле конфигурации. Предпочтительным способом изменения режима SELinux является изменение файла конфигурации и перезагрузка системы.

При переключении из отключенного режима в принудительный или разрешительный SELinux автоматически переназначает файловую систему после перезагрузки. Это означает, что SELinux проверяет контексты безопасности любых файлов и изменяет неправильные (например, неверно помеченные файлы), которые могут создавать проблемы в новом режиме. Кроме того, любые не помеченные файлы помечаются контекстами. Процесс переназначения может занять много времени, поскольку проверяется контекст каждого файла. Далее показано сообщение, которое вы получите, когда система будет повторно маркироваться после перезагрузки:

```
*** Warning -- SELinux targeted policy relabel is required.
*** Relabeling could take a very long time, depending on file
*** system size and speed of hard drives.
```

Чтобы изменить режим в файле `/etc/selinux/config`, замените строку `SELINUX` одной из следующих:

- `SELINUX=disabled`;
- `SELINUX=enforcing`;
- `SELINUX=permissive`.

В примере далее файл конфигурации SELinux показывает, что был установлен разрешительный режим. Теперь при перезагрузке системы он меняется:

```
# cat /etc/selinux/config
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#     targeted - Targeted processes are protected,
#     minimum - Modification of targeted policy.
#               Only selected processes are protected.
#     mls - Multi Level Security protection
SELINUX=permissive
...
```

Основной файл конфигурации SELinux содержит не только настройку режима. Он также указывает тип применяемой политики.

Установка типа политики SELinux

Выбранный тип политики определяет, будет ли SELinux применять базовую политику или политики TE, MLS. Тип непосредственно определяет наборы правил политики, используемых для настройки того, к чему может получить доступ объект.

По умолчанию для типа политики задано значение `targeted`. Чтобы изменить тип политики по умолчанию, отредактируйте файл `/etc/selinux/config`. Замените строку `SELINUXTYPE=` одной из следующих:

- `SELINUX=targeted`;
- `SELINUX=mls`;
- `SELINUX=minimum`.

Если вы установили тип SELinux в `mls` или `minimum`, вначале убедитесь, что установлен подходящий пакет политики. Проверьте это, введя следующую команду:

```
yum list selinux-policy-mls or yum list selinux-policy-minimum
```

ПРИМЕЧАНИЕ

Чтобы проверить пакеты политики SELinux в дистрибутиве Ubuntu, задействуйте команду sudo apt-cache policy package_name.

Приведенный далее пример файла конфигурации SELinux показывает, что тип был установлен в `mls`. Теперь, когда происходит перезагрузка системы, тип политики изменяется:

```
# cat /etc/selinux/config
# This file controls the state of SELinux on the system.
...
# SELINUXTYPE= type of policy in use. Possible values are:

#     targeted - Targeted processes are protected,
#     minimum - Modification of targeted policy.
#               Only selected processes are protected.
#     mls - Multi Level Security protection.
SELINUXTYPE=mls
```

Управление контекстами безопасности SELinux

Контекст безопасности SELinux позволяет системе применять правила политики для доступа субъектов к объектам. Система Linux поставляется с уже назначенными контекстами безопасности.

Чтобы просмотреть текущие контексты безопасности файлов и процессов SELinux, используйте команду `secon`. В табл. 24.1 перечислены доступные параметры команды `secon`.

Таблица 24.1. Параметры команды secon

Параметр	Описание
-u	Отображает контекст безопасности
-r	Отображает роль контекста безопасности
-t	Отображает тип контекста безопасности
-s	Отображает уровень чувствительности контекста безопасности
-c	Отображает уровень разрешения контекста безопасности
-m	Отображает уровни чувствительности и разрешения контекста безопасности в виде диапазона MLS

Если вы применяете команду `secon` без параметров, она показывает контекст безопасности текущего процесса. Чтобы увидеть контекст безопасности другого процесса, используйте параметр `-p`. В следующем примере показано, как с помощью команды `secon` просмотреть текущий контекст безопасности и контекст безопасности процесса `systemd`:

```
# secon -urt
user: unconfined_u
role: unconfined_r
type: unconfined_t
# secon -urt -p 1
user: system_u
role: system_r
type: init_t
```

Чтобы просмотреть контекст безопасности файла, используйте параметр `-f`, как показано далее:

```
# secon -urt -f /etc/passwd
user: system_u
role: object_r
type: passwd_file_t
```

Команда `secon` не отображает ваш контекст безопасности. Чтобы увидеть его, используйте команду `id`.

Управление контекстом безопасности пользователя

Помните, что каждый идентификатор входа системного пользователя сопоставляется с определенным идентификатором пользователя SELinux. Чтобы просмотреть список сопоставлений в своей системе, введите команду `semanage login -l`. Команда `semanage` и ее выходные данные показаны в следующем коде. Если идентификатор входа пользователя не указан в списке, то он применяет сопоставление входа по умолчанию, то есть `Login Name` файла `_default_`. Обратите внимание на то, что

отображаются и соответствующие настройки MLS/MCS для каждого пользователя SELinux:

```
# semanage login -l
Login Name           SELinux User           MLS/MCS Range     Service
__default__          unconfined_u           s0-s0:c0.c1023    *
root                 unconfined_u           s0-s0:c0.c1023    *
```

Чтобы увидеть текущих пользователей SELinux и связанные с ними роли, возьмите команду `semanage user-l`. В следующем примере частично отображены роли, сопоставленные с именами пользователей SELinux:

```
# semanage user -l

             Labeling MLS/      MLS/
SELinux User Prefix   MCS Level MCS Range         SELinux Roles
guest_u      user     s0        s0                guest_r
...
user_u       user     s0        s0                user_r
xguest_u     user     s0        s0                xguest_r
```

Если вам нужно добавить новое имя пользователя SELinux, снова примените утилиту `semanage`. На этот раз задействуется команда `semanage user -a` *`selinux_имя_пользователя`*. Сопоставить идентификатор входа с недавно добавленным именем пользователя SELinux можно с помощью команды `semanage login -a -s` *`selinux_имя_пользователя LoginID`*. Команда `semanage` — это мощный инструмент управления конфигурацией SELinux. Чтобы получить более подробную информацию о ней, см. справочные страницы.

Управление контекстом безопасности файла

Метки файлов имеют решающее значение для поддержания надлежащего контроля доступа к данным из файлов. SELinux устанавливает метки безопасности файлов при установке и перезагрузке системы, когда SELinux переключается из режима `disabled` в любой другой. Чтобы увидеть текущую метку файла (она же контекст безопасности), используйте команду `ls -Z`:

```
# ls -Z /etc/passwd
-rw-r--r--. root root system_u:object_r:etc_t:s0 /etc/passwd
```

Управлять метками контекста безопасности файлов можно с помощью нескольких команд (табл. 24.2).

Таблица 24.2. Команды управления метками контекста безопасности

Утилита	Описание
chcat	Изменяет категории метки контекста безопасности файла
chcon	Изменяет метки контекста безопасности файла
fixfiles	Вызывает команду restorecon/setfiles

Утилита	Описание
restorecon	Делает то же самое, что и утилита setfiles, но у нее другой интерфейс
setfiles	Проверяет и/или исправляет метки контекста безопасности. Может быть запущена для проверки меток файлов и/или их смены при добавлении в систему нового модуля политики. Делает то же самое, что и команда restorecon, но у нее другой интерфейс

Команды `chcat` и `chcon`, показанные в табл. 24.2, позволяют изменить контекст безопасности файла. В следующем примере команда `chcon` применяется для изменения пользователя SELinux, связанного с `file.txt`, от `undefined_u` к `system_u`:

```
# ls -Z file.txt
-rw-rw-r--. johndoe johndoe
 unconfined_u:object_r:user_home_t:s0 file.txt
# chcon -u system_u file.txt
# ls -Z file.txt
-rw-rw-r--. johndoe johndoe
 system_u:object_r:user_home_t:s0 file.txt
```

Рассматривая табл. 24.2, обратите внимание на то, что команды `fixfiles`, `restorecon` и `setfiles` — это, по сути, одна и та же утилита. Однако `restorecon` чаще всего используется при фиксации меток файлов. Команда `restorecon filename` возвращает файл в контекст безопасности по умолчанию.

Управление контекстом безопасности процесса

Процесс — это запущенная программа. Когда вы запускаете программы или службы в системе Linux, каждой из них присваивается идентификатор процесса (см. главу 6 «Управление активными процессами»). В системе с SELinux процесс также имеет контекст безопасности.

То, как процесс получает свой контекст безопасности, зависит от того, какой процесс его запустил. Помните, что процесс `systemd` (ранее `init`) является «матерью» всех процессов (см. главу 15 «Запуск и остановка служб»). Таким образом, многие демоны и процессы запускаются `systemd`. Процессы, запускаемые `systemd`, получают новые контексты безопасности. Например, когда демон `apache` запускается `systemd`, ему присваивается тип (он же домен) `httpd_t`. Назначенный контекст обрабатывается политикой SELinux, написанной специально для этого демона. Если для процесса не существует политики, то ему присваивается тип по умолчанию `unconfined_t`

Новый процесс (дочерний) программы или приложения, запущенного пользователем (родительский процесс), наследует контекст безопасности пользователя. Конечно, это происходит только в том случае, если пользователю разрешено запускать программу. Процесс также может запускать программу. Дочерний процесс в этом случае тоже наследует контекст безопасности родительского процесса, то есть дочерний процесс выполняется в том же домене.

Таким образом, контекст безопасности процесса устанавливается перед запуском программы и зависит от того, кто ее запустил. Для изменения контекстов безопасности, в которых выполняется программа, можете использовать несколько команд:

- `runcon`. Запустите программу, применяя параметры для определения пользователя, роли и типа (он же домен);
- `sandbox`. Запустите программу в строго контролируемом домене (он же «песочница»).

При использовании команды `runcon` могут возникнуть проблемы, поэтому делайте это осторожно. Команда `sandbox` более стабильна. Она позволяет тестировать новые программы в вашей системе Linux.

Управление правилами политики SELinux

Правила политики — это правила, используемые SELinux для определения того, имеет ли субъект доступ к объекту. Они группируются в пакеты, называемые также модулями, и устанавливаются вместе с SELinux. Простой способ просмотреть модули в своей системе — применить команду `semodule -l`. В ней перечисляются все модули политики и их номера версий. Пример политики для `semodule -L` показан далее:

```
# semodule -l
abrt
accountsd
acct
...
xserver
zabbix
zarafa
zebra
zoneminder
zosremote
```

Существуют инструменты, которые могут помочь вам управлять собственными модулями политики и даже создавать их. Они перечислены в табл. 24.3 и доступны в операционной системе Fedora.

Таблица 24.3. Инструменты управления пакетами политик SELinux

Инструмент	Описание
audit2allow	Генерирует правила политики allow/dontaudit1 из журналов запрещенных операций
audit2why	Генерирует описание причины отказа в доступе из журналов запрещенных операций
checkmodule	Компилирует модули
checkpolicy	Компилирует политики SELinux

Инструмент	Описание
load_policy	Загружает новые политики в ядро
semodule_expand	Расширяет пакет модулей политики
semodule_link	Связывает пакеты модулей политики вместе
semodule_package	Создает пакет из модуля политик

Далее приведен пример политики, обычно используемой в качестве основы для создания локальных правил политики. Она довольно длинная, поэтому показана лишь часть:

```
# cat /usr/share/doc/selinux-policy/example.te

policy_module(myapp,1.0.0)

########################################
#
# Declarations
#

type myapp_t;
type myapp_exec_t;
domain_type(myapp_t)
domain_entry_file(myapp_t, myapp_exec_t)

type myapp_log_t;
logging_log_file(myapp_log_t)
type myapp_tmp_t;
files_tmp_file(myapp_tmp_t)
...
allow myapp_t myapp_tmp_t:file manage_file_perms;
files_tmp_filetrans(myapp_t,myapp_tmp_t,file)
#
```

Из примера видно, что в коде политики используется специальный синтаксис. Чтобы создавать и изменять правила политики, вам нужно изучить синтаксис этого языка правил политики, научиться применять компиляторы политик SELinux и связывать файлы правил политики для формирования модулей. Для этого вам, вероятно, потребуется пройти дополнительное обучение. В этот момент может возникнуть искушение отказаться от работы с системой SELinux. Однако использовать логические типы для изменения политик гораздо проще.

Управление SELinux через логические типы

Написание правил политики SELinux и создание модулей — довольно сложная и трудоемкая работа. Создание неверных правил политики может поставить под угрозу безопасность вашей системы Linux. К счастью, SELinux работает с логическими типами.

Логический тип — это своего рода тумблер, который включает или выключает настройки. Он позволяет изменять части правил политики SELinux, не изучая язык написания политики. Изменить политики можно и без перезагрузки системы!

Чтобы просмотреть список логических типов, используемых в SELinux, задействуйте команду `getsebool -a`. Далее приведен пример правила политики SELinux с логическими типами в операционной системе Fedora и Linux:

```
# getsebool -a
abrt_anon_write --> off
abrt_handle_event --> off
...
xserver_object_manager --> off
zabbix_can_network --> off
```

Чтобы увидеть политику, которую можно изменить с помощью логического типа, снова используйте команду `getsebool`. На этот раз ей передается имя политики, как показано в следующем примере:

```
# getsebool httpd_can_connect_ftp
httpd_can_connect_ftp --> off
```

Для переключения политики можно использовать команду `setsebool`. Она временно изменяет правило политики. Когда система перезагружается, логический тип возвращается в исходное значение. Если вам нужно, чтобы этот параметр был постоянным, можете применить команду `setsebool` с параметром `-P`.

Команда `setsebool` имеет шесть параметров: три параметра включения политики (`on`, `1` или `true`) и три параметра ее выключения (`off`, `0` или `false`).

Один из случаев использования команды `setsebool` связан с ограничением применения исполняемых файлов. В некоторых ситуациях небезопасно позволять пользователям выполнять программы из своего каталога `/home`. Чтобы этого не произошло, правило политики `allow_user_exec_content` должно быть отключено. В следующем примере команда `setsebool` используется именно для этого. Обратите внимание на то, что параметр `-P` делает эту настройку постоянной:

```
# setsebool -P allow_user_exec_content off
```

Команда `getsebool` проверяет правильность установки логического типа:

```
# getsebool allow_user_exec_content
allow_user_exec_content --> off
```

Логические типы значительно упрощают изменение текущих правил политики SELinux. В целом утилиты настройки командной строки SELinux, такие как `getsebool`, просты в использовании. Однако у системы SELinux есть также инструмент настройки графического интерфейса. Он устанавливается с помощью команды `yum install policycoreutils-gui`. В дистрибутиве Ubuntu примените команду `sudo apt-get install policycoreutils`. Чтобы задействовать этот инструмент, введите команду `system-config-selinux` — и появится графический интерфейс.

Мониторинг и устранение неполадок в системе SELinux

Система SELinux — это еще один инструмент для мониторинга вашей системы. Он регистрирует все отказы в доступе, что может помочь вам определить, была ли система атакована. Эти же файлы журнала SELinux полезны и при устранении неполадок SELinux.

Журнал SELinux

SELinux использует кэш, называемый *кэшем вектора доступа* (access vector cache, AVC), при просмотре правил политики для конкретных контекстов безопасности. При отказе в доступе, называемом отказом AVC, сообщение об этом помещается в файл журнала.

Сообщения об отказе могут помочь вам диагностировать и устранять обычные нарушения политики SELinux. Место сохранения сообщений об отказе зависит от состояния демонов `auditd` и `rsyslogd`.

- Если демон `auditd` запущен, сообщения об отказе регистрируются в файле `/var/log/audit/audit.log`.
- Если демон `auditdis` не запущен, но запущен демон `rsyslogd`, сообщения об отказе регистрируются в файле `/var/log/messages`.

ПРИМЕЧАНИЕ

Если и auditd, и rsyslogd запущены и в системе есть демон setroubleshootd, сообщения об отказе отправляются как в файлы журнала audit.log, так и в файлы журнала messages. Однако информация об отказе в файле журнала messages сохраняется в формате, который демон setroubleshootd понимает лучше.

Просмотр сообщений SELinux

Если запущен демон `auditd`, вы можете быстро увидеть, были ли зарегистрированы какие-либо отказы AVC с помощью команды `aureport`. В следующем примере показано использование команд `aureport` и `grep` для поиска отказов AVC. По крайней мере один отказ был зарегистрирован в файле `/var/log/audit/audit.log`:

```
# aureport | grep AVC
Number of AVC's: 1
```

После обнаружения отказа AVC в журнале `audit.log` вы можете использовать команду `ausearch` для просмотра сообщений об отказе. В следующем примере показана команда `ausearch`, применяемая для просмотра зарегистрированного сообщения об отказе AVC:

```
# ausearch -m avc
type=AVC msg=audit(1580397837.344:274): avc: denied { getattr } for
```

```
pid=1067
    comm="httpd" path="/var/myserver/services" dev="dm-0" ino=655836
    scontext=system_u:system_r:httpd_t:s0
    tcontext=unconfined_u:object_r:var_t:s0 tclass=file permissive=0
```

В примере отображается информация о том, кто пытался получить доступ, а также их контекст безопасности при попытке доступа. Ищите в сообщении об отказе AVC следующие ключевые слова:

- `type=AVC`;
- `avc: denied`;
- `com="httpd"`;
- `path="/var/myserver/services"`.

Вывод команды дает достаточно данных, чтобы либо начать устранять проблему, либо отследить вредоносную активность. В примере каталог `/var/myserver/services` имеет неправильный контекст файла SELinux для чтения службой `httpd`.

Просмотр сообщений SELinux в журнале сообщений

Если у вас запущена служба `auditd`, вы можете найти сообщения об отказе AVC, выполнив поиск в файле `/var/log/audit/audit.log` с помощью команды `grep`. Для последних систем RHEL и Fedora или любой системы Linux, использующей службу `systemd`, можете запустить команду `journalctl`, чтобы проверить наличие сообщений журнала отказа AVC. Внутри каждого сообщения журнала находится сообщение AVC, позволяющее изучить информацию об этом отказе AVC, как в следующем примере:

```
# journalctl | grep AVC
type=AVC msg=audit(1580397837.346:275): avc: denied { getattr }for
pid=1067
   comm="httpd" path="/var/myserver/services" dev="dm-0" ino=655836
   scontext=system_u:system_r:httpd_t:s0
   tcontext=unconfined_u:object_r:var_t:s0 tclass=file permissive=0
```

Поскольку вы знаете, что отказы AVC есть, можете передать весь файл `/var/log/audit/ audit.log` в команду `sealert`, чтобы решить следующие проблемы:

```
# sealert -a /var/log/audit/audit.log
SELinux is preventing httpd from getattr access on the file
/var/myserver/services.

*****  Plugin catchall (100. confidence) suggests  *************

If you believe that httpd should be allowed getattr access on the
services file by default.
Then you should report this as a bug.
You can generate a local policy module to allow this access.
Do
allow this access for now by executing:
```

```
# ausearch -c 'httpd' --raw | audit2allow -M my-httpd
# semodule -X 300 -i my-httpd.pp

Additional Information:
Source Context system_u:      system_r:httpd_t:s0
Target Context unconfined_u:  object_r:var_t:s0
Target Objects                /var/myserver/services [ file ]
...
Raw Audit Messages
type=AVC msg=audit(1580397837.346:275): avc:  denied  { getattr }
for  pid=1067 comm="httpd" path="/var/myserver/services" dev="dm-0"
ino=655836 scontext=system_u:system_r:httpd_t:s0
tcontext=unconfined_u:object_r:var_t:s0 tclass=file permissive=0
Hash: httpd,httpd_t,var_t,file,getattr
```

В этом случае, если вы хотите разрешить службе `httpd` доступ к содержимому запрещаемого каталога, можете запустить команды `ausearch` и `semodule`, как в примере. Это создает и применяет новую политику SELinux, которая разрешает доступ к содержимому. При условии, что нет никаких других проблем с разрешениями, `httpd` должен получить доступ к этому контенту.

Диагностика журналов SELinux

Очевидно, что файлы журналов чрезвычайно важны для диагностики и устранения нарушений политики SELinux. Ведение файлов журнала или прямого запроса журнала `systemd` (команда `journalctl`) — это первые шаги по устранению неполадок SELinux. Таким образом, важно убедиться, что ваша система Linux регистрирует сообщения в первую очередь.

Быстрый способ определить, ведется ли журнал, — проверить, работают ли соответствующие демоны: `auditd`, `rsyslogd` и/или `setroubleshootd`. Используйте соответствующую команду, например `systemctl status auditd.service`. Конечно, применяемая команда зависит от вашего дистрибутива Linux и его версии. Более подробную информацию см. в главе 15. Если демон не запущен, запустите его, чтобы начать ведение журнала.

ВНИМАНИЕ!

Иногда отказы AVC не регистрируются из-за правил политики dontaudit. Хотя правила dontaudit помогают уменьшить количество ложных срабатываний в журналах, они могут вызвать проблемы при устранении неполадок. Чтобы исправить ситуацию, временно отключите все правила политики dontaudit с помощью команды semodule -DB.

Устранение распространенных проблем SELinux

В начале работы с SELinux легко упустить из виду очевидное. Всякий раз, когда доступ оказывается запрещен, вы должны сначала проверить традиционные разрешения Linux DAC. Например, используйте команду `ls -l` и дважды проверьте правильность назначений владельца файла, группы, чтения, записи и выполнения.

Что обычно может вызвать проблемы в системе SELinux:

- применение нестандартного каталога для службы;
- использование нестандартного порта для службы;
- перемещение файлов, которые приводят к потере меток контекста безопасности;
- неверно заданные логические типы.

Любую из этих проблем можно довольно быстро решить.

Использование нестандартного каталога для службы

По разным причинам вы можете хранить файлы службы в нестандартном каталоге. Система SELinux должна знать об этом. В противном случае она запрещает доступ к законным запросам доступа к услугам.

Например, вы решили хранить свои HTML-файлы не в стандартном каталоге `/var/www/html`, а в другом месте. Вы помещаете файлы в `/abc/www/html`. Необходимо сообщить SELinux, что служба `http` должна иметь доступ к файлам в `/abc/www/html`. Команды для этого — `semanage` или `restorecon`. В следующем примере с их помощью добавляется соответствующий тип контекста безопасности в каталог `/abc/www/html`:

```
# semanage fcontext -a -t httpd_sys_content_t  "/abc/www/html(/.*)?"
```

Чтобы фактически установить новый тип контекста безопасности для файлов в каталоге, нужно использовать команду `restorecon -R`, например:

```
# restorecon -R -v /abc/www/html
# ls -Z /abc/www/html
unconfined_u:object_r:httpd_sys_content_t:s0 abc
```

Теперь демон `httpd` имеет разрешение на доступ к вашим HTML-файлам в нестандартном каталоге.

Использование нестандартного порта для службы

Вы можете установить, что служба прослушивает нестандартный порт, так же, как и при решении описанной ранее проблемы. Когда вы меняете этот порт, служба перестает запускаться.

Например, в целях безопасности вы решили перенести службу `sshd` с 22-го порта на нестандартный порт 47 347. SELinux не знает об этом порте, и служба не запускается. Чтобы устранить эту проблему, сначала необходимо найти тип контекста безопасности для `sshd`. Для этого с помощью следующего кода выдается команда `semanage port -l`, а результаты передаются в команду `grep` для поиска службы `ssh`:

```
# semanage port -l | grep ssh
ssh_port_t                 tcp                  22
```

Здесь вы можете видеть, что необходимый тип контекста — ssh_port_t. Теперь, снова используя команду semanage, вы добавляете этот тип в порт 47 347:

```
# semanage port -a -t ssh_port_t -p tcp 47347
# semanage port -l | grep ssh
ssh_port_t                tcp              47347, 22
```

На этом этапе отредактируйте файл /etc/ssh/sshd_config, чтобы добавить в него строку порта 47 347. Затем перезапустите службу sshd, чтобы она прослушивала нестандартный порт 47 347.

Перемещение файлов и потеря меток контекста безопасности

Вы использовали команду cp для временного перемещения файла из /etc в каталог /tmp. Затем — команду mv, чтобы вернуть его обратно. Теперь вместо исходного контекста безопасности файл имеет контекст безопасности временного каталога, и ваша система получает сообщения об отказе AVC, когда служба, применяющая этот файл, пытается запустить его.

Это легко исправить благодаря команде restorecon -R. Просто введите команду restorecon, и файл восстановит свой постоянный контекст безопасности.

Неверно заданные логические типы

Еще одна распространенная проблема — неправильная установка логического типа, которая выдает несколько отказов AVC.

Например, если скрипты вашей системы больше не могут подключаться к сети и вы получаете отказы AVC в журналах, нужно проверить логические типы httpd. Используйте команду getsebool -a и передайте ее в команду grep для поиска любых логических типов, влияющих на службу httpd. В примере показано, как используются эти команды:

```
# getsebool -a | grep http
...
httpd_can_network_connect --> off
...
```

Команда getsebool показывает, что логический тип httpd_can_network_connect установлен в значение off. Чтобы изменить логический тип, используйте команду setsebool -P httpd_can_network_connect on.

Обратите внимание: параметр -P использован для того, чтобы сделать настройку постоянной. Теперь ваши скрипты могут подключаться к сети.

Выводы о SELinux

Очевидно, что система SELinux — довольно сложный и масштабный инструмент. Теперь вы больше знаете о ее основных функциях. Дадим некоторые рекомендации, которые стоит учесть при внедрении SELinux в свою систему.

Режим `targeted` по умолчанию может использоваться для защиты большинства основных сетевых служб (`httpd`, `vsftpd`, Samba и т. д.), при этом не нужно назначать специальные роли пользователей или иным образом блокировать вашу систему. В этом случае главное — поместить файлы в стандартные места (или выполнить команды для назначения соответствующих контекстов файлов нестандартными), убедиться, что логические типы включены для менее безопасных функций, и просмотреть отказы AVC, чтобы узнать, вызывают ли они проблемы.

- Начните с разрешительного режима работы. Он позволяет успешно выполнять запросы, которые SELinux считает небезопасными.
- Запускайте текущую систему в течение значительного времени в разрешительном режиме. Просмотрите журналы на предмет того, какие проблемы могут возникнуть с настройками SELinux по умолчанию. Затем можете изменить логические типы или контексты файлов так, чтобы функции, запрещенные неправильно, могли быть разрешены. После того как проблемы будут решены, включите принудительный режим.
- В целом реализуйте изменения конфигурации SELinux по одному за раз в тестовой среде или в разрешительном режиме. Посмотрите, какой эффект дает каждое изменение конфигурации, прежде чем переходить к следующему. Затем можете использовать команду `audit2allow`, чтобы выборочно разрешать действия для службы, которые остановлены отказами AVC.

Больше информации о SELinux

Несколько дополнительных источников информации могут помочь в работе с SELinux в системе Linux.

- **Справочные страницы системы.** Выполните команду `man -k selinux`, чтобы найти все справочные страницы для утилит SELinux, установленных в настоящее время в вашей системе. Если вы отлаживаете проблемы SELinux для хорошо известной службы, например `httpd`, `vsftpd`, Samba и т. д., то, вероятно, существует справочная страница, связанная с тем, как исправить проблемы SELinux именно с ней.
- **Руководства The Red Hat Enterprise Linux Manuals.** Сайт docs.redhat.com содержит полное руководство по системе SELinux.
- **Руководство The Fedora Project SELinux Guide.** На сайте selinuxproject.org имеется полноценный гайд по SELinux. Однако руководство обновляется не для каждой версии Fedora, поэтому вам, возможно, придется поискать информацию в более старых версиях. Кроме того, руководства SELinux нет в руководстве безопасности Security manual, и его тоже стоит изучить.
- **Вики-страница SELinux Project Wiki.** Это официальная страница проекта SELinux. Часть ресурсов доступна на сайте selinuxproject.org.

Резюме

SELinux обеспечивает повышение безопасности Linux и устанавливается по умолчанию во многих ее дистрибутивах. В этой главе вы узнали о преимуществах системы SELinux, о том, как она работает, как ее настроить, как устранить различные проблемы с ней и как получить дополнительную информацию об этой важной части повышенной безопасности.

На первый взгляд SELinux кажется довольно сложным инструментом. Однако если разбить его работу на компоненты — режимы работы, контексты безопасности, типы политик и пакеты политик, — видно, как различные части работают вместе. Каждый компонент играет важную роль в обеспечении и тестировании выбранных требований безопасности для вашей организации.

Вы также узнали о различных действиях, доступных для настройки SELinux. Несмотря на то что система SELinux предварительно настроена, может потребоваться внести некоторые изменения для удовлетворения потребностей вашей организации в области безопасности. Каждый компонент имеет собственные настройки и конфигурации. Хотя создание правил политики мы не рассматривали, вы узнали, как изменять предоставленные политики с помощью логических типов.

SELinux имеет еще один инструмент для мониторинга безопасности вашей системы Linux. Поскольку SELinux регистрирует все отказы в доступе, она может помочь определить, была ли предпринята атака на вашу систему. Даже самые продуманные планы могут провалиться. Поэтому в этой главе вы узнали, как исправить распространенные проблемы конфигурации SELinux.

В следующей главе поговорим о том, как защитить вашу систему Linux в сети. Вы узнаете о контроле доступа, управлении брандмауэрами и защите удаленного доступа.

Упражнения

С помощью этих упражнений проверьте свои знания о системе SELinux. Задачи подходят для систем Fedora или Red Hat Enterprise Linux (некоторые сработают и в других системах Linux). Если затрудняетесь с решением заданий, воспользуйтесь ответами к упражнениям, приведенными в приложении Б (хотя Linux позволяет решать задачи разными способами).

1. Не внося никаких изменений в основной файл конфигурации SELinux, используйте нужную команду, чтобы перевести систему в разрешительный режим работы SELinux.
2. Не внося никаких изменений в основной файл конфигурации SELinux, используйте нужную команду, чтобы перевести систему в принудительный режим для SELinux.

3. Какие текущие и постоянные типы политик SELinux установлены в вашей системе и как вы их нашли?
4. Перечислите контекст безопасности для файла `/etc/hosts` и определите его атрибуты.
5. Создайте файл с именем `test.html` в своем домашнем каталоге и назначьте ему тип `httpd_sys_content_t.` (Это делается для того, чтобы контент был доступен для совместного использования вашим веб-сервером за пределами общего каталога `/var/www/html`.)
6. Перечислите контекст безопасности для запущенного процесса `crond` и определите его атрибуты.
7. Создайте файл с именем `/etc/test.txt`, измените его контекст файла на `user_tmp_t`, восстановите правильное содержимое (контекст по умолчанию для каталога `/etc`) и удалите файл. Используйте команду `ls -Z /etc/test.txt` для проверки файла в каждой точке процесса.
8. У вас есть tftp-сервер в частной сети, и вы хотите разрешить анонимную запись и доступ к домашнему каталогу службы `tftp` (SELinux находится в принудительном режиме). Определите, какие логические типы разрешают анонимную запись и доступ к домашнему каталогу службы `tftp`, и включите их.
9. Какая команда перечисляет все модули политики SELinux в вашей системе вместе с номером их версии?
10. Укажите SELinux разрешить доступ к службе `sshd` через TCP-порт 54 903.

25 Защита Linux в сети

В этой главе

- Управление сетевыми службами.
- Управление доступом к сетевым службам.
- Работа брандмауэров.

Настройка системы Linux в сети, особенно общедоступной, создает целый ряд новых проблем, когда речь заходит о безопасности. Лучший способ защитить систему Linux — это отключить ее от всех сетей. Обычно это неосуществимо. Множество книг написано о том, как защитить компьютерную систему в сети. Многие организации нанимают штатных администраторов компьютерной безопасности для наблюдения за своими системами Linux. Поэтому главу можно рассматривать как краткое введение в защиту Linux в сети.

Аудит сетевых служб

Большинство систем Linux, работающих на крупных предприятиях, настроены как серверы, которые по сети предлагают различные службы удаленным клиентам. *Сетевая служба* — это любая задача, выполняемая компьютером, для которой требуются отправка и получение информации по сети с использованием предопределенного набора правил. Маршрутизация электронной почты — это сетевая служба, как и обслуживание веб-страниц.

Сервер Linux имеет потенциал для предоставления тысяч услуг. Многие из них перечислены в файле `/etc/services`. Рассмотрим следующие разделы этого файла:

```
$ cat /etc/services
# /etc/services:
# $Id: services,v 1.55 2013/04/14 ovasik Exp $
#
# Network services, Internet style
# IANA services version: last updated 2013-04-10
```

```
#
# Note that it is presently the policy of IANA to assign ...
# Each line describes one service, and is of the form:
#
# service-name  port/protocol  [aliases ...]   [# comment]
...
echo            7/tcp
echo            7/udp
discard         9/tcp           sink null
discard         9/udp           sink null
systat          11/tcp          users
systat          11/udp          users
daytime         13/tcp
daytime         13/udp
qotd            17/tcp          quote
qotd            17/udp          quote
...
chargen         19/tcp          ttytst source
chargen         19/udp          ttytst source
ftp-data        20/tcp
ftp-data        20/udp
# 21 is registered to ftp, but also used by fsp
ftp             21/tcp
...
http            80/tcp      www www-http    # WorldWideWeb HTTP
http            80/udp      www www-http    # HyperText Transfer
Protocol
http            80/sctp                     # HyperText Transfer
Protocol
kerberos        88/tcp      kerberos5 krb5  # Kerberos v5
kerberos        88/udp      kerberos5 krb5  # Kerberos v5
...
blp5            48129/udp           # Bloomberg locator
com-bardac-dw   48556/tcp           # com-bardac-dw
com-bardac-dw   48556/udp           # com-bardac-dw
iqobject        48619/tcp               # iqobject
iqobject        48619/udp               # iqobject
```

Обратите внимание на три столбца информации, идущих после строк с комментариями. Левый столбец содержит названия всех служб. Средний определяет номер порта и тип протокола, используемого для этой службы. Правый содержит необязательный псевдоним или список псевдонимов для службы.

Многие дистрибутивы Linux поставляются с запущенными ненужными сетевыми службами. Такая служба угрожает безопасности вашей системы Linux. Например, если ваш сервер Linux является сервером печати, то он должен оказывать только услуги печати и не должен задействовать веб-сервисы Apache. Из-за этого ваш сервер печати может без необходимости подвергнуться любым вредоносным атакам, использующим уязвимости веб-служб.

Первоначально ограничение количества служб в системах Linux означало установку отдельных физических серверов Linux, на каждом из которых работали всего

несколько служб. Позже запуск нескольких виртуальных машин Linux на физическом хосте позволил заблокировать небольшие наборы служб на виртуальных машинах. В последнее время контейнерные приложения позволяют запускать на каждом физическом узле гораздо больше отдельных и защищенных служб.

Оценка доступа к сетевым службам с помощью команды nmap

Замечательный инструмент, который поможет вам просмотреть свои сетевые службы с точки зрения сети, — это сканер безопасности `nmap`. Утилита `nmap` доступна в большинстве дистрибутивных репозиториев Linux. Веб-страница инструмента — это nmap.org.

Чтобы установить инструмент `nmap` в дистрибутив Fedora или RHEL, задействуйте команды `yum` и `dnf` (применяя привилегии суперпользователя), как показано в следующем примере:

```
# yum install nmap -y
Updating Subscription Management repositories.
Last metadata expiration check: 0:03:41 ago on Sat 12 Oct 2019
11:24:07 PM EDT.
Dependencies resolved.
================================================================================
 Package      Arch    Version       Repository                    Size
================================================================================
Installing:
 nmap         x86_64  2:7.70-4.el8  rhel-8-for-x86_64-appstream-rpms
5.8 M

Transaction Summary
================================================================================
Install  1 Package

Total download size: 5.8 M
Installed size: 24 M
...
Installed:
  nmap-2:7.70-4.el8.x86_64

Complete!
```

Чтобы установить утилиту `nmap` в дистрибутив Ubuntu, введите команду `sudo apt-get install nmap` в командной строке.

Полное название утилиты `nmap` — Network Mapper. Она имеет множество функций для аудита безопасности и исследования сети. Сканирование различных портов с помощью `nmap` позволяет увидеть, какие службы работают на всех серверах вашей локальной сети и сообщают ли они о своей доступности.

ПРИМЕЧАНИЕ

Что такое порт? Порты, точнее, сетевые порты — это числовые значения, используемые сетевыми протоколами TCP и UDP в качестве точек доступа к службам в системе. Стандартные номера портов назначаются службам таким образом, чтобы служба знала, что нужно прослушивать определенный номер порта, а клиент знал, что нужно запрашивать службу по этому номеру порта. Например, порт 80 — это стандартный сетевой порт для незашифрованного (HTTP) трафика веб-службы Apache. Так что, если вы введете в браузере www.example.com, он предположит, что вы собираетесь использовать TCP-порт 80 на сервере. Представьте, что сетевой порт — это как дверь на сервер Linux. Все двери пронумерованы, и за каждой из них находится особая служба, готовая помочь тому, кто постучит в эту дверь.

Для проверки портов вашего сервера утилита `nmap` предлагает несколько типов сканирования. На сайте `nmap` http://nmap.org/book/man-port-scanning-techniques.html есть полное руководство по всем методам сканирования портов, которые вы можете использовать. Вот два основных способа сканирования, с которых можно начать аудит службы.

- **TCP Connect port scan.** Для реализации этого способа `nmap` пытается подключиться к портам, применяющим протокол управления передачей (TCP) на сервере. Если порт прослушивается, попытка подключения завершается успешно.

  TCP — это сетевой протокол, используемый в наборе сетевых протоколов TCP/IP. Он ориентирован на соединение. Его основная цель — договориться и инициировать соединение с помощью так называемого тройного рукопожатия. TCP отправляет пакет синхронизации (SYN) на удаленный сервер, указывая в нем конкретный номер порта. Удаленный сервер получает SYN и отвечает исходящему компьютеру пакетом подтверждения (SYN-ACK). Затем исходный сервер подтверждает (ACK) ответ, и TCP-соединение устанавливается. Это тройное рукопожатие часто называют SYN-SYN-ACK или SYN, SYN-ACK, ACK.

  Если вы выберете сканирование порта TCP Connect, утилита `nmap` использует тройное рукопожатие для выполнения небольших действий на удаленном сервере. Любые службы, применяющие протокол TCP, будут реагировать на сканирование.
- **UDP port scan.** Для сканирования этим способом `nmap` отправляет UDP-пакет на каждый порт сканируемой системы. *UDP* — еще один популярный протокол в наборе сетевых протоколов TCP/IP. Однако, в отличие от TCP, UDP — это *протокол без подключения*. Если порт прослушивает и имеет службу, использующую протокол UDP, он реагирует на сканирование.

СОВЕТ

Имейте в виду, что свободное программное обеспечение с открытым кодом (free and open source software, FOSS) доступно и злоумышленникам. Пока вы выполняете проверки с помощью утилиты nmap, помните, что результаты удаленного сканирования, которые вы видите для сервера Linux, увидят и другие пользователи. Это поможет вам оценить параметры безопасности своей системы с точки зрения того, сколько информации выдается для сканирования портов. Вы должны применять инструмент nmap только в собственных системах, потому что сканирование портов на компьютерах других людей может создать впечатление, что вы пытаетесь взломать систему.

При запуске утилита nmap представляет удобный небольшой отчет с информацией о системе, которую вы сканируете, и портах, которые она видит. Портам присваивается определенный статус. Утилита сообщает о шести возможных состояниях порта.

- open. Это самое опасное состояние порта, которое может выявить сканирование nmap. Открытый порт указывает на то, что у сервера есть служба, обрабатывающая запросы на этом порте. Представьте, что это табличка на двери: «Входите! Мы поможем вам». Конечно, при условии, что вы действительно предлагаете любому воспользоваться этой службой.
- closed. Доступ к порту closed возможен, но по ту сторону двери нет никакой службы. Однако состояние сканирования по-прежнему указывает на наличие сервера Linux по этому конкретному IP-адресу.
- filtered. Это наилучшее состояние для защиты порта, доступ к которому должен быть ограничен. Невозможно определить, действительно ли сервер Linux находится на сканируемом IP-адресе. Вполне вероятно, что служба может прослушивать определенный порт, но брандмауэр блокирует доступ к этому порту, эффективно предотвращая любой доступ к службе через определенный сетевой интерфейс.
- unfiltered. Сканирование с помощью утилиты nmap видит порт, но не может определить его состояние: open или closed.
- open|filtered. Сканирование при помощи nmap видит порт, но не может определить его состояние — open или filtered.
- closed|filtered. Сканирование посредством утилиты nmap видит порт, но не может определить его состояние — closed или filtered.

Чтобы лучше понять, как использовать утилиту nmap, рассмотрим следующий пример. Для построения списка сетевых служб здесь сканирование nmap выполняется в системе Fedora. Первое сканирование — это сканирование TCP Connect из командной строки с помощью адреса обратной связи 127.0.0.1:

```
# nmap -sT 127.0.0.1
Starting Nmap 7.70 ( https://nmap.org ) at 2020-1-10 11:47 EDT
Nmap scan report for localhost (127.0.0.1)

Host is up (0.016s latency).
Not shown: 998 closed ports

PORT    STATE SERVICE
25/tcp  open  smtp
631/tcp open  ipp

Nmap done: 1 IP address (1 host up) scanned in 1.34 seconds
```

Сканирование TCP Connect сообщает, что два TCP-порта открыты и службы прослушивают запросы к ним на локальном хосте (127.0.0.1).

- Простой протокол передачи почты (Simple Mail Transfer Protocol, SMTP) прослушивается через порт TCP 25.

- Протокол межсетевой печати (Internet Printing Protocol, IPP) прослушивается через порт TCP 631.

Следующее сканирование `nmap` — это UDP-сканирование по адресу интернет-протокола loopback системы Fedora.

```
# nmap -sU 127.0.0.1

Starting Nmap 7.70 ( https://nmap.org ) at 2020-1-10 11:48 EDT
Nmap scan report for localhost (127.0.0.1)
Host is up (0.00048s latency).
Not shown: 997 closed ports

PORT     STATE         SERVICE
68/udp   open|filtered dhcpc
631/udp  open|filtered ipp
Nmap done: 1 IP address (1 host up) scanned in 2.24 seconds
```

Сканирование UDP `nmap` сообщает, что два UDP-порта открыты и службы прослушивают их.

- Протокол динамической настройки узла (Dynamic Host Control Protocol, `dhcpc`) прослушивает порт 68.
- Протокол межсетевой печати прослушивает порт 631.

Обратите внимание на то, что порт IPP 631 сканируется по протоколам и TCP, и UDP, поскольку служба IPP может взаимодействовать через TCP- и UDP-протокол и, таким образом, отображается в обоих случаях.

Используя эти два простых способа сканирования `nmap`, TCP Connect и UDP на адресе интернет-протокола loopback, вы можете узнать список сетевых служб, предлагаемых вашим Linux-сервером. Имейте в виду, что номера портов связаны с определенным протоколом (TCP или UDP) и определенным сетевым интерфейсом. Например, если у вас есть одна карта сетевого интерфейса (NIC) на компьютере, выходящем в Интернет, и другая, на выходящем в частную сеть, вы можете оказывать частные услуги (например, услугу CUPS для печати) в частной сети. Однако, скорее всего, потребуется отфильтровать этот порт (631) на сетевой карте, выходящей в Интернет.

Применение утилиты nmap для аудита широковещательных сетевых служб

Вы, вероятно, хотите, чтобы ваш сайт посещали много людей (служба `httpd`). Вероятно, вы не хотите, чтобы все пользователи Интернета могли получить доступ к вашим файловым ресурсам SMB (служба `smb`). Чтобы убедиться, что вы правильно разделяете доступ к этим двум типам служб, необходимо проверить, какие службы, доступные на ваших общедоступных сетевых интерфейсах, видны вредоносному сканеру.

Суть этих действий состоит в том, чтобы сравнить, как выглядит ваш Linux-сервер изнутри и снаружи. Если вы определили, что сетевые службы, которые должны быть закрыты, доступны для других, необходимо заблокировать доступ к ним с внешних интерфейсов.

СОВЕТ

Может возникнуть соблазн не выполнять сканирование из внутренней сети вашей организации. Так поступать не стоит. Вредоносную деятельность часто ведут сами сотрудники компании или кто-то, кто уже проник через внешнюю защиту. И здесь утилита nmap очень помогает. Чтобы получить правильное представление о том, какие порты вашего Linux-сервера видны, нужно провести сканирование из нескольких мест. Например, простой аудит настроил бы сканирование:

- на самом сервере Linux;
- с другого сервера в той же сети организации;
- за пределами сети организации.

В следующих примерах выполняется часть простого аудита. Утилита `nmap` запускается в системе Fedora, обозначенной как Host-A. Host-A — это сервер Linux, сетевые службы которого должны быть защищены. Host-B — это Linux-сервер, использующий дистрибутив Linux Mint и находящийся в той же сети, что и Host-A.

СОВЕТ

При проведении аудиторских проверок следует учитывать параметры безопасности различных сетевых компонентов, таких как брандмауэр сервера и маршрутизаторы компании.

В этом примере аудита сканирование выполняется с хоста-A с использованием не адреса обратной связи, а фактического IP-адреса. Сначала IP-адрес для Host-A определяется с помощью команды `ip addr show`. IP-адрес — 10.140.67.23:

```
# ip addr show
fconfig
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
     group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel
     state UP group default qlen 1000
    link/ether 52:54:00:c4:27:4e brd ff:ff:ff:ff:ff:ff
    inet 10.140.67.23/24 brd 10.140.67.255 scope global dynamic
       noprefixroute ens3
       valid_lft 3277sec preferred_lft 3277sec
    inet6 fe80::5036:9ec3:2ae8:7623/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

Затем с помощью IP-адреса Host-A сканирование nmap TCP Connect выполняется с Host-A. Сканирование nmap переходит в сеть для продолжения работы. У всех портов статус closed:

```
# nmap -sT 10.140.67.23
Starting Nmap 7.80 ( https://nmap.org ) at 2020-1-31 11:53 EDT

Nmap scan report for rhel8 (10.140.67.23)

Host is up (0.010s latency).
All 1000 scanned ports on 10.140.67.23 are closed

Nmap done: 1 IP address (1 host up) scanned in 1.48 seconds
```

Сканирование nmap перемещается с Host-A на Host-B. Теперь сканирование TCP Connect выполняется на портах Host-A из командной строки Host-B:

```
$ nmap -sT 10.140.67.23
Starting Nmap 7.80 ( https://nmap.org ) at 2020-1-31 11:57 EDT

Note: Host seems down. If it is really up,
 but blocking our ping probes, try -PN

Nmap done: 1 IP address (0 hosts up) scanned in 0.11 seconds
```

В рассмотренном примере утилита nmap дает полезную подсказку. Host-A, похоже, не работает или просто блокирует попытки подключения. Поэтому предпринимается еще одна попытка сканирования nmap с хоста Host-B, причем nmap приказано отключить пинг с помощью параметра -PN:

```
$ nmap -sT -PN 10.140.67.23
Starting Nmap 7.80 ( https://nmap.org ) at 2020-1-31 11:58 EDT
Nmap scan report for rhel8 (10.140.67.23)

Host is up (0.0015s latency).
All 1000 scanned ports on 10.140.67.23 are filtered

Nmap done: 1 IP address (1 host up) scanned in 5.54 seconds
```

В примере видно, что Host-A (10.140.67.23) запущен и работает и все его порты имеют статус filtered. Это означает, что на Host-A установлен брандмауэр. Сканирование с Host-B позволяет лучше понять, что может увидеть вредоносный сканер при сканировании Linux-сервера. Здесь вредоносный сканер почти ничего не увидит.

ПРИМЕЧАНИЕ

Если вам известна утилита nmap, то вы знаете, что сканирование TCP SYN — это реализуемое ею сканирование по умолчанию. TCP SYN scan отлично справляется с исследованием удаленной системы в скрытом режиме. Поскольку вы изучаете собственную систему в целях аудита безопасности, имеет смысл использовать более тяжеловесные утилиты сканирования nmap. Для сканирования TCP SYN scan необходимо ввести команду nmap -SS ip_address.

Службы, работающие в настоящее время на Host-A, не так уж интересны. В следующем примере другая служба, `sshd`, запускается на Host-A с помощью команды `systemctl` (см. главу 15 «Запуск и остановка служб»). Эта служба интереснее для сканирования утилитой `nmap`:

```
# systemctl start sshd.service
# systemctl status sshd.service
• sshd.service – OpenSSH server daemon
   Loaded: loaded (/usr/lib/systemd/system/sshd.service; enabled;
vendor preset: enabled)
   Active: active (running) since Fri 2020-1-30 15:08:29 EDT; 1 day
20h ago
     Docs: man:sshd(8)
           man:sshd_config(5)
 Main PID: 807 (sshd)
    Tasks: 1 (limit: 12244)
   Memory: 10.9M
   CGroup: /system.slice/sshd.service
           └─ 807 /usr/sbin/sshd -D -oCiphers=...
```

Кроме того, поскольку брандмауэр Host-A блокирует сканирование `nmap` от Host-B, было бы интересно посмотреть, что может сообщить `nmap`, когда брандмауэр не работает. В следующем примере показано, что брандмауэр отключен на Host-A для системы Fedora 21 или RHEL 7 (для других систем вам, вероятно, потребуется отключить службу `iptables`):

```
# systemctl stop firewalld.service
# systemctl status firewalld.service
```

При запущенной новой службе и отключенном брандмауэре Host-A сканирование `nmap` должно что-то найти. В следующем примере сканирование `nmap` выполняется снова с Host-B. На этот раз утилита `nmap` показывает службу `ssh`, работающую на открытом порте 22. Обратите внимание на то, что при отключенном брандмауэре на Host-A оба сканирования `nmap` собирают гораздо больше информации. Это действительно демонстрирует важность брандмауэра вашего Linux-сервера:

```
# nmap -sT 10.140.67.23
Starting Nmap 7.80 ( http://nmap.org ) at 2020-1-31 11:58 EDT
Nmap scan report for 10.140.67.23
Host is up (0.016s latency).
Not shown: 999 closed ports

PORT   STATE SERVICE
22/tcp open  ssh
Nmap done: 1 IP address (1 host up) scanned in 0.40 seconds

# nmap -sU 10.140.67.23
[sudo] password for johndoe: ***************
Starting Nmap 5.21 ( http://nmap.org ) at 2020-1-31 11:59 EDT
Nmap scan report for 10.140.67.23
Host is up (0.00072s latency).
Not shown: 997 closed ports
```

```
PORT    STATE         SERVICE
68/udp  open|filtered dhcpc
631/udp open|filtered ipp
...
Nmap done: 1 IP address (1 host up) scanned in 1081.83 seconds
```

Для того чтобы провести тщательный аудит, обязательно включите UDP-сканирование. Кроме того, существуют дополнительные проверки `nmap`, которые могут оказаться полезными, их можно найти на сайте утилиты `nmap`.

ВНИМАНИЕ

Если вы отключили брандмауэр своего сервера для проверки nmap, обязательно включите его снова. Задействуйте команду systemctl start firewalld.service.

Нужно также реализовать элементы управления для тех служб, которые должен предлагать ваш Linux-сервер. Один из способов сделать это — использовать правила брандмауэра.

Ранние версии Linux применяют TCP-оболочки для разрешения или запрета доступа к службам Linux. В них есть файлы `/etc/hosts.allow` и `/etc/hosts.deny`, в которых можно конкретно указать, какие службы должны быть доступны, а какие заблокированы для определенных внешних системных имен и/или IP-адресов. Начиная с Fedora 28 и RHEL 8, функция TCP `wrappers` была исключена из этих дистрибутивов. Однако некоторые функции, такие как `vsftpd`, по-прежнему поддерживают эти файлы конфигурации, только другими способами.

Работа с брандмауэрами

Брандмауэр в здании — это огнеупорная стена, которая предотвращает распространение огня по всему строению. Брандмауэр компьютера блокирует передачу вредоносных или нежелательных данных в компьютерную систему или сеть и из нее. Например, брандмауэр может блокировать вредоносное сканирование с портов сервера Linux. Он может также изменять сетевые пакеты, проходящие через систему, и перенаправлять их различными способами.

В Linux служба *iptables* — это функция брандмауэра на уровне ядра. Чаще всего она используется для разрешения или блокирования доступа из внешних систем к службам, работающим в локальной системе. Служба `iptables` работает, позволяя создавать правила, которые могут быть применены к каждому пакету, который пытается войти в систему (`INPUT`), выйти из нее (`OUTPUT`) или пройти через нее (`FORWARD`).

Хотя пропуск или блокирование пакетов, пытающихся войти в систему, — это основная функция `iptables`, вы также можете создать правила для службы, которые позволят осуществлять следующее.

- Эффективно блокировать пакеты, выходящие из вашей системы, чтобы процесс в ней не достиг удаленного хоста, диапазона адресов или выбранных служб.

- Переадресовывать пакеты с одного сетевого интерфейса системы на другой, позволяя компьютеру эффективно действовать в качестве маршрутизатора между двумя сетями.
- Переадресовывать порт пакета, предназначенный для переадресации выбранного порта, на другой порт вашей локальной системы или на удаленную систему, чтобы и в других местах запрос из пакета мог быть обработан.
- Изменить информацию в заголовке пакета, чтобы перенаправить его или каким-то образом пометить как нуждающийся в дополнительной обработке.
- Разрешить нескольким компьютерам в частной сети (например, компьютерам, телевизорам или другим устройствам в домашней сети) взаимодействовать с Интернетом по одному общедоступному IP-адресу (это называется *IP-маскарадинг*).

В следующих разделах я опишу многие из этих функций, но в основном сосредоточусь на правилах блокировки или разрешения доступа к службам, работающим в вашей системе Linux.

Что такое брандмауэр

Можно представлять брандмауэр как полноценный барьер, но на самом деле брандмауэр Linux — это просто фильтр, который проверяет каждый сетевой пакет или запрос приложения, поступающий в компьютерную систему или сеть или выходящий из нее.

ПРИМЕЧАНИЕ

Что такое сетевой пакет? Сетевой пакет — это передаваемые фрагменты, на которые были разбиты данные. Фрагменты, или пакеты, имеют дополнительные данные, добавляемые к ним по мере прохождения вниз по модели OSI. Это как класть письмо в конверт на каждом этапе, когда оно движется вниз по стеку протоколов. Одна из целей присоединения дополнительных данных — обеспечение безопасного прибытия пакета в пункт назначения. Дополнительные данные удаляются из пакета, когда он проходит обратно по модели OSI в пункте назначения (например, как будто снимают внешний конверт и передают письмо на слой выше).

Брандмауэры могут быть разделены на категории в зависимости от их функций. Все категории играют важную роль в обеспечении безопасности ваших сервера и сети.

- **Брандмауэр может быть основан либо на сети, либо на хосте.** Сетевой брандмауэр защищает всю сеть или подсеть. Например, сетевой брандмауэр будет использоваться на рабочем месте, где сеть должна быть защищена брандмауэром экранирующего маршрутизатора.

  Брандмауэр на основе хоста — это брандмауэр, работающий на отдельном хосте или сервере и защищающий его. Скорее всего, на вашем домашнем компьютере есть брандмауэр — это и есть брандмауэр на базе хоста.

- **Аппаратный или программный брандмауэр.** Брандмауэры могут быть расположены на сетевых устройствах, например на маршрутизаторах. Их фильтры настроены в соответствующем программном обеспечении маршрутизатора. В доме интернет-провайдер (ISP) может предоставить маршрутизатор, чтобы вы могли получить доступ к Интернету. Маршрутизатор содержит встроенное программное обеспечение брандмауэра и считается аппаратным брандмауэром.

  Брандмауэры могут быть расположены в компьютерной системе как приложение. Оно позволяет установить правила фильтрации входящего трафика. Это пример программного брандмауэра. Он называется также брандмауэром на основе правил.
- **Брандмауэр как фильтр сетевого либо прикладного уровня.** Брандмауэр, который проверяет отдельные сетевые пакеты, называется *пакетным фильтром*. *Брандмауэр сетевого уровня* допускает только определенные пакеты в систему и из нее. Он работает на нижних уровнях эталонной модели OSI.

  Брандмауэр *прикладного уровня* фильтрует на более высоких уровнях эталонной модели OSI. Он позволяет только определенным приложениям получать доступ в систему и из нее.

Вы можете видеть, что эти категории брандмауэра пересекаются. Лучшая настройка брандмауэра — это сочетание всех категорий. Как и во многих других методах обеспечения безопасности, чем больше у нее слоев, тем труднее проникнуть в систему.

Добавление брандмауэров

В системе Linux брандмауэр представляет собой сетевой брандмауэр на уровне хоста, управляемый утилитой `iptables` и связанными с ней компонентами уровня ядра. С помощью службы `iptables` вы можете создать ряд правил для каждого сетевого пакета, проходящего через сервер Linux. И можете точно настроить правила, чтобы разрешить сетевой трафик из одного места, но не из другого. Эти правила, по сути, формируют список контроля доступа к сети для сервера Linux.

Fedora, RHEL и другие дистрибутивы Linux добавили службу `firewalld`, чтобы обеспечить более динамичный способ управления правилами брандмауэра, чем было прежде. В последних версиях систем RHEL и Fedora брандмауэр `iptables` был заменен на `nftables`. Окно конфигурации брандмауэра (команда `firewall-config`) позволяет очень просто открыть на нем порты и выполнить маскировку (маршрутизацию частных адресов в общедоступную сеть) или переадресацию портов. Служба `firewalld` может реагировать на изменения условий, что статическая служба `iptables` сама по себе сделать не может. Разрешая доступ к службе, `firewalld` может также загружать модули, необходимые для разрешения доступа к службе.

СОВЕТ

Утилита iptables управляет брандмауэром Linux, называемым netfilter. Таким образом, вы часто будете видеть брандмауэр netfilter/iptables. Синтаксис iptables по-прежнему поддерживается, но в последних версиях RHEL и Fedora nftables фактически предоставляет базу для iptables.

Служба firewalld

Служба `firewalld` уже может быть установлена в вашей системе Linux. Чтобы проверить это, введите следующее:

```
# systemctl status firewalld
• firewalld.service – firewalld – dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; ena>
   Active: active (running) since Sat 2019-10-19 11:43:13 EDT; 5m>
     Docs: man:firewalld(1)
 Main PID: 776 (firewalld)
    Tasks: 2 (limit: 2294)
   Memory: 39.6M
   CGroup: /system.slice/firewalld.service
           └── 776 /usr/bin/python3 /usr/sbin/firewalld --nofork -->
```

Если это не так, вы можете установить службу `firewalld` и связанный с ней графический интерфейс пользователя, а затем запустить ее следующим образом:

```
# yum install firewalld firewall-config
# systemctl start firewalld.service
# systemctl enable firewalld.service
```

Для управления службой `firewalld` запустите окно Firewall Configuration (Настройка межсетевого экрана). Для этого введите следующую команду:

```
# firewall-config &
```

На рис. 25.1 показан пример окна Firewall Configuration (Настройка межсетевого экрана).

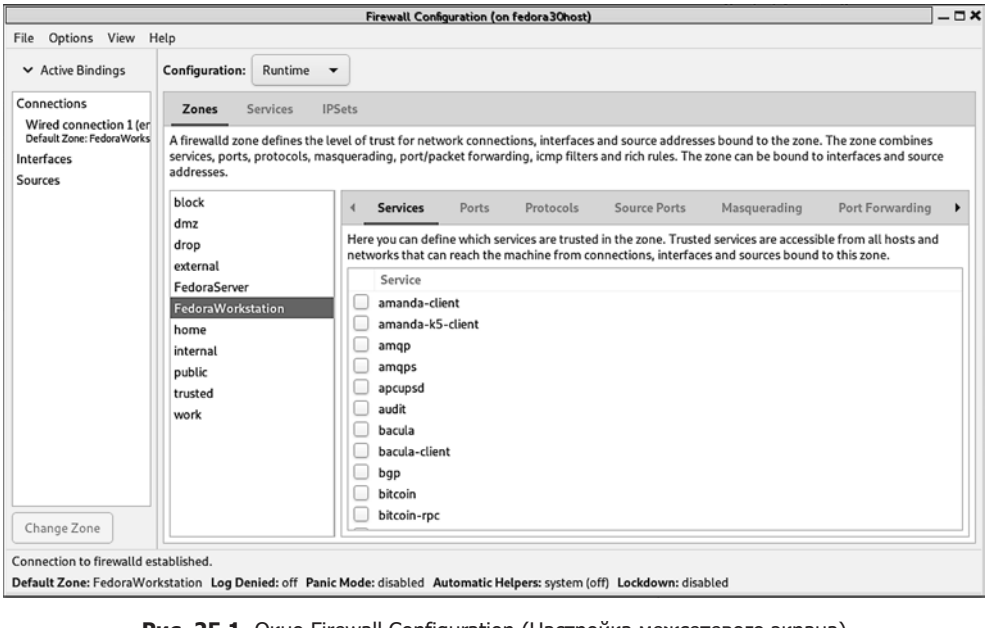


Рис. 25.1. Окно Firewall Configuration (Настройка межсетевого экрана)

С помощью `firewalld` вы можете выбрать одну из нескольких зон брандмауэра в зависимости от того, какие службы хотите использовать совместно и какой уровень защиты получить для системы. Выбранный в этом примере набор правил Fedora Workstation по умолчанию подходит для рабочей станции Linux, работающей в домашней сети. Он содержит:

- **клиент DHCPv6** — включает автоматическое назначение адресов в сетях IPv6;
- **групповой DNS (mDNS)** — разрешает интерфейсы системы доменных имен на небольших сетевых интерфейсах, не требуя регулярного DNS-сервера;
- **клиент и межсетевую печать (IPP)** — разрешает общий доступ к принтерам в локальной системе и сети;
- **клиент Samba** — разрешает общий доступ к файлам с системами Windows и другими в локальной сети;
- **службу SSH** — разрешает другим пользователям попытаться войти в вашу систему из сети;
- **интерфейс Cockpit** — разрешает доступ к веб-администрированию с помощью Cockpit из сети. Интерфейс установлен по умолчанию в дистрибутиве RHEL 8, но не установлен по умолчанию на рабочей станции Fedora 30. Таким образом, Cockpit не появится в окне Firewall Configuration (Настройка межсетевого экрана) до тех пор, пока вы не установите пакет `cockpit`.

Если вы подключаете компьютер к сетям с разным уровнем доверия (например, к беспроводной сети в аэропорту), можете настроить правила брандмауэра, выбрав другую зону. Например, чтобы перейти в общедоступную зону из окна Firewall Configuration (Настройка межсетевого экрана), выполните следующие действия.

1. В столбце Active Bindings выберите активное соединение (в данном примере `Wired connection 1`).
2. Выберите новую зону (например, `public`).
3. Нажмите кнопку Change Zone.

Зона `public`, по-прежнему разрешающая IPv6-соединения, удаленный вход в систему (SSH) и службу mDNS, не позволяет получить доступ к более уязвимым службам печати, обмена файлами Windows (Samba) и Cockpit.

Помимо смены зон, можно открыть некоторые порты брандмауэра, чтобы разрешить доступ к выбранным службам. В окне Firewall Configuration (Настройка межсетевого экрана) с зоной рабочей станции Fedora, установленной в качестве текущей зоны, просто щелкните на каждой службе, которую хотите открыть. Порт, разрешающий доступ ко всем службам, открывается немедленно (при выборе конфигурации среды выполнения) или открывается постоянно (при выборе постоянной конфигурации).

Одной из приятных особенностей окна Firewall Configuration (Настройка межсетевого экрана) является то, что, когда вы разрешаете доступ к службе, можно сделать больше, чем просто открыть порт. Например, включение службы FTP вызывает загрузку модулей отслеживания соединений, которые позволяют при необходимости получать доступ к нестандартным портам через брандмауэр.

Изменение правил брандмауэра из веб-интерфейса Cockpit

Cockpit предлагает еще один интуитивно понятный способ работы с брандмауэром системы. Чтобы просмотреть и изменить брандмауэр с помощью Cockpit, выполните следующие действия.

1. Откройте браузер для работы с интерфейсом Cockpit (https://yourhost:9090) и войдите в систему с привилегией суперпользователя.
2. Выберите пункт Networking (Сеть) ▸ Firewall (Межсетевой экран), как показано на рис. 25.2.

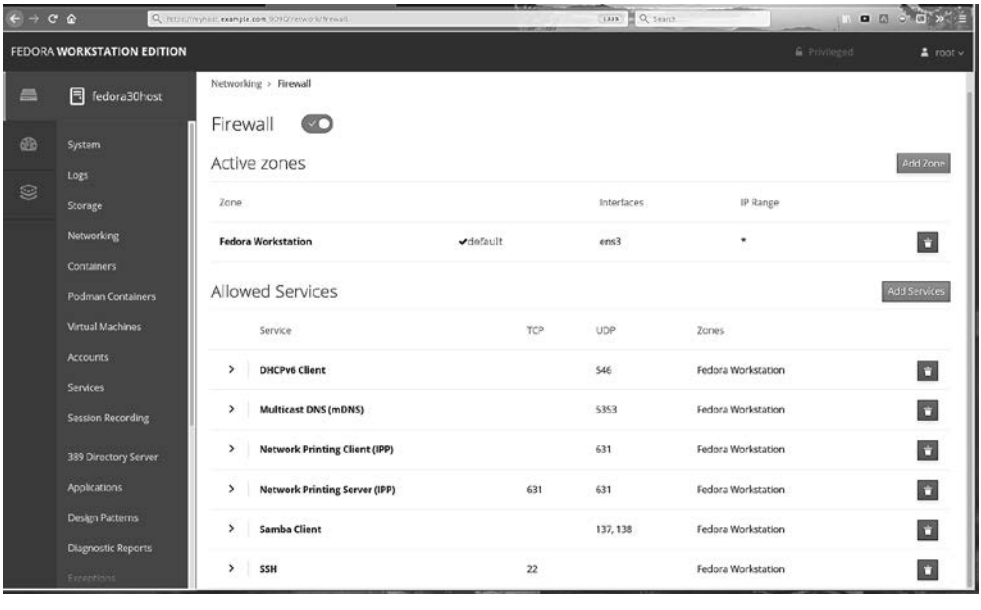


Рис. 25.2. Настройка брандмауэра

3. Выберите пункт Add Services (Добавить службы). Появится соответствующее диалоговое окно.
4. Установите флажок рядом со службой, которую хотите включить в текущей зоне, и нажмите кнопку Add Services (Добавить службы).

Доступ к выбранному порту задан. То есть если у вас есть служба, работающая на этом порте, доступ тому, кто ее запрашивает (например, доступ к вашему веб-серверу на портах 80 и 443), будет разрешен.

Как упоминалось ранее, в основе служб Cockpit и `firewalld` лежит служба `iptables`. Если у вас есть система Linux без служб Cockpit или `firewalld` (или с отключенной службой `firewalld`), вы все равно можете использовать службу `iptables`. В следующих разделах описывается, как установить правила брандмауэра `iptables` вручную и применять службу `iptables` напрямую, без службы `firewalld`.

Служба iptables

Прежде чем начать изменять правила брандмауэра с помощью утилиты `iptables`, необходимо понять основы `netfilter/iptables`, которые включают в себя:

- таблицы;
- цепочки;
- политики;
- правила.

Понимание основ поможет правильно настроить брандмауэр сервера Linux и управлять им.

Таблицы netfilter/iptables. Брандмауэр `iptables` способен на большее, чем просто фильтрация на низовом уровне — уровне пакетов. Он определяет, какой тип функции брандмауэра применяется. В утилите `iptables` есть четыре таблицы и дополнительная таблица SELinux. Рассмотрим их функции.

- `filter`. Таблица `filter` фильтрует пакеты брандмауэра. В ней решения по управлению доступом принимаются для пакетов, перемещающихся по системе Linux, из нее и через нее.
- `nat`. Задействуется для преобразования сетевых адресов (`NAT`). Правила таблицы `NAT` позволяют перенаправлять пакет туда, куда нужно.
- `mangle`. Как вы могли бы заподозрить, пакеты искажаются (модифицируются) в соответствии с правилами таблицы искажений. Использование таблицы `mangle` напрямую не особо распространено, обычно с ее помощью изменяют способ управления пакетом.
- `raw`. Таблица `raw` применяется для освобождения определенных сетевых пакетов от так называемого отслеживания соединений. Эта функция важна, когда вы используете преобразование сетевых адресов и виртуализацию на своем сервере Linux.
- `security`. Эта таблица доступна только в дистрибутивах Linux с функцией SELinux (см. главу 24 «Повышенная безопасность с технологией SELinux»). Хотя таблица безопасности обычно не используется напрямую, она позволяет SELinux разрешать или блокировать пакет на основе политик SELinux, добавляя еще один уровень фильтрации к стандартным правилам фильтрации пакетов.

Из перечисленных таблиц три сосредоточены на преобразовании сетевых адресов (network address translation). Таким образом, `filter` является основной таблицей, о которой идет речь в главе о базовой фильтрации пакетов брандмауэра.

Цепочки службы netfilter/iptables. Брандмауэр `netfilter/iptables` классифицирует сетевые пакеты по категориям, называемым цепочками. Существует пять цепочек (категорий), к которым может быть отнесен сетевой пакет:

- `INPUT` — сетевые пакеты, поступающие *на* сервер Linux;
- `FORWARD` — поступающие на сервер Linux сетевые пакеты, которые должны быть *перенаправлены* через другой сетевой интерфейс на сервере;

- `OUTPUT` — сетевые пакеты, поступающие *из* сервера Linux;
- `PREROUTING` — используется преобразование сетевых адресов `NAT` для изменения сетевых пакетов, когда они поступают на сервер Linux;
- `POSTROUTING` — применяется преобразование сетевых адресов `NAT` для изменения сетевых пакетов до того, как они выйдут из сервера Linux.

Таблица `netfilter/iptables`, с которой вы решите работать, определяет, какие цепочки доступны для отнесения сетевых пакетов к определенной категории. В табл. 25.1 показано, какие цепочки доступны для каждой из таблиц.

Таблица 25.1. Цепочки, доступные для таблиц netfilter/iptables

Таблица	Доступные цепочки
filter	INPUT, FORWARD, OUTPUT
nat	PREROUTING, OUTPUT, POSTROUTING
mangle	INPUT, FORWARD, PREROUTING, OUTPUT, POSTROUTING
raw	PREROUTING, OUTPUT
security	INPUT, FORWARD, OUTPUT

После того как сетевой пакет отнесен к определенной цепочке, служба `iptables` может определить, какие политики или правила применяются конкретно к нему.

Правила, политики и цели службы netfilter/iptables. Для каждого сетевого пакета можно установить правило, определяющее, что с ним делать. Сетевые пакеты могут быть идентифицированы брандмауэром `netfilter/iptables` многими способами. Вот несколько:

- IP-адрес источника;
- IP-адрес назначения;
- сетевой протокол;
- входящий порт;
- исходящий порт;
- состояние сети.

Если для конкретного пакета не существует правила, то используется общая политика. Каждая категория пакетов или цепочка имеет политику по умолчанию. Если сетевой пакет соответствует определенному правилу или политике по умолчанию, то над ним выполняется действие. Оно зависит от того, какая цель службы `iptables` задана. Вот несколько действий (целей), которые можно предпринять:

- `ACCEPT` — сетевой пакет принимается на сервер;
- `REJECT` — сетевой пакет отбрасывается и не допускается на сервер. Отправляется сообщение об отказе;
- `DROP` — сетевой пакет отбрасывается и не допускается на сервер. Сообщение об отказе не отправляется.

Действие REJECT передает сообщение об отказе, в то время как DROP этого не делает. Вы можете использовать действие REJECT для внутренних сотрудников, которым следует сообщить, что их исходящий сетевой трафик отклоняется и почему. Применяйте действие DROP для входящего трафика, чтобы он весь блокировался без уведомления об этом.

СОВЕТ

Есть несколько дополнительных, более сложных целей для службы iptables, таких как QUEUE. Вы можете узнать больше об этих действиях на справочной странице iptables.

Утилита iptables реализует программный брандмауэр, используя таблицу filter с помощью политик и правил. Теперь, когда у вас есть общее представление о реализации программного брандмауэра, можете начать углубляться в конкретные команды для реализации брандмауэра с помощью утилиты iptable.

Использование службы iptables

На Linux-сервере брандмауэр должен быть установлен по умолчанию. Тем не менее необходимо посмотреть и убедиться, что он действительно включен.

- **Брандмауэр netfilter/iptables в системах RHEL 7, RHEL 8 и последние версии Fedora.** Служба интерфейса брандмауэра, работающая в этих дистрибутивах, называется firewalld. Служба iptables по умолчанию не запускается непосредственно в этих системах. Чтобы узнать, работает ли эта служба брандмауэра, введите systemctl status firewalld.service в командной строке:
  - чтобы включить брандмауэр, введите systemctl start firewalld.service и systemctl enable firewalld.service в командной строке;
  - чтобы отключить брандмауэр, введите systemctl stop firewalld.service в командной строке.
- **Брандмауэр Ubuntu netfilter/iptables.** Служба интерфейса брандмауэра, работающая в этом дистрибутиве, называется ufw. Чтобы проверить, работает ли она, введите sudo ufw status в командной строке. Служба ufw — это интерфейс к утилите iptables, которая не работает как служба в Ubuntu. Вы можете использовать команды ufw для управления правилами брандмауэра. Однако все команды утилиты iptables по-прежнему действительны для Ubuntu:
  - чтобы включить брандмауэр, введите sudo ufw enable в командной строке;
  - чтобы отключить брандмауэр, введите sudo ufw disable в командной строке.

После того как вы проверили состояние и включили или отключили брандмауэр netfilter/iptables, различий между дистрибутивами не остается.

Чтобы узнать, какие политики и правила существуют в настоящее время для таблицы filter (по умолчанию), введите iptables-vnL в командной строке:

```
# iptables -vnL
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)...
```

Обратите внимание на то, что в системах с включенной службой `firewalld` существует гораздо больше цепочек `iptables` и правил, перечисленных по умолчанию, чем вы могли бы использовать в системе, задействующей службу `iptables` напрямую. Это делается для того, чтобы обеспечить большую гибкость при построении брандмауэров, позволяя разделить ваши правила на зоны различных уровней безопасности.

В предыдущем примере показана только первая строка вывода `iptables`. Она показывает, что политика по умолчанию цепочки `INPUT` применяется ко всем сетевым пакетам, которые не соответствуют другому правилу. В настоящее время все политики `INPUT`, `FORWARD` и `OUTPUT` по умолчанию настроены на действие `ACCEPT`. Все сетевые пакеты могут входить, проходить и выходить. Брандмауэр в этом состоянии, по существу, отключен до тех пор, пока не будут добавлены определенные правила `REJECT` или `DROP`.

СОВЕТ

Если ваш Linux-сервер имеет дело с сетевыми пакетами IPv6, можете использовать утилиту ip6tables для управления брандмауэром для IPv6-адресов. Утилита ip6tables почти идентична утилите iptables. Для получения дополнительной информации введите man ip6tables в командной строке.

Изменение политик и правил службы iptables. Прежде чем начать изменять брандмауэр `netfilter/iptables` непосредственно с помощью команды `iptables`, вам следует зайти в систему, которая применяется для тестирования, и отключить службу `firewalld`.

Чтобы начать работу, необходимо изучить несколько параметров команд.

Параметры изменения брандмауэра:

- `-t` *таблица*. Команда `iptables`, перечисленная вместе с этим параметром, применяется к *таблице*. По умолчанию используется таблица `filter`, например:

  ```
  # iptables -t filter -P OUTPUT DROP
  ```

- `-P` *цепочка действие*. Задает общую политику для конкретной *цепочки*. Правила в *цепочке* проверяются на совпадения. Если совпадений нет, выполняется указанное *действие*, например:

  ```
  # iptables -P INPUT ACCEPT
  ```

- `-A` *цепочка*. Устанавливает правило, называемое добавленным, которое является исключением из общей политики для назначенного параметра *цепочка*, например:

  ```
  # iptables -A OUTPUT -d 10.140.67.25 -j REJECT
  ```

- `-I` *правило# цепочка*. Вставляет добавленное правило *правило#* в определенное место в списке добавленных правил, назначенных параметру *цепочка*, например:

  ```
  # iptables -I 5 INPUT -s 10.140.67.23 -j DROP
  ```

- `-D` *цепочка правило#*. Удаляет конкретное правило *правило#* из назначенных параметру `chain`, например:

  ```
  # iptables -D INPUT 5
  ```

- `-j` *`действие`*. Выполняет `действие`, если правило подошло, например:

```
# iptables -A INPUT -s 10.140.67.25 -j DROP
```

- `-d` *`IP-адрес`*. Назначает указанное правило для применения к назначенному *`IP-адресу`*, например:

```
# iptables -A OUTPUT -d 10.140.67.25 -j REJECT
```

- `-s` *`IP-адрес`*. Назначает указанное правило для применения к назначенному *`IP-адресу`*, например:

```
# iptables -A INPUT -s 10.140.67.24 -j ACCEPT
```

- `-p` *`протокол`*. Назначает указанное правило для применения к указанному *`протоколу`*. Например, здесь отбрасываются входящие запросы `ping` (`icmp`):

```
# iptables -A INPUT -p icmp -j DROP
```

- `--dport` *`порт#`*. Назначает указанное правило для применения к определенным пакетам протокола, поступающим в назначенный параметр *`порт#`*, например:

```
# iptables -A INPUT -p tcp --dport 22 -j DROP
```

- `--sport` *`порт#`*. Назначает указанное правило для применения к определенным пакетам протокола, выходящим из назначенного параметра *`порт#`*, например:

```
# iptables -A OUTPUT -p tcp --sport 22 -j ACCEPT
```

- `-m state –state` *`состояние_соединения`*. Назначает указанное правило для применения к назначенному *`состоянию`* (состояниям) соединения, например:

```
# iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
```

Чтобы увидеть, как работают параметры `iptables`, рассмотрим следующий пример. У вас есть сервер Linux (Host-A) по IP-адресу 10.140.67.23. В вашей сети имеется еще два сервера Linux. Один из них — Host-B по IP-адресу 10.140.67.22, а другой — Host-C по IP-адресу 10.140.67.25. Вам необходимо выполнить следующее:

- разрешить полный доступ с сервера Host-C к серверу Host-A;
- блокировать удаленные подключения входа с помощью службы `ssh` от Host-B к Host-A.

Настройка политики DROP. В следующем коде показаны политики брандмауэра сервера Host-A по умолчанию. Здесь брандмауэр открыт полностью, без каких-либо ограничений. Никаких правил не установлено, и все политики настроены на режим `ACCEPT`:

```
# iptables -vnL

Chain INPUT (policy ACCEPT)
target     prot opt source                         destination

Chain FORWARD (policy ACCEPT)
target     prot opt source                         destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source                         destination
```

Во-первых, что произойдет, если политика INPUT будет изменена с ACCEPT на DROP? Достигнет ли она цели? Посмотрите, что будет, если попытаться это сделать. Помните, что если для входящего пакета не указано никаких правил, то соблюдается политика chain's policy. Это изменение внесено в брандмауэр Host-A в следующем примере:

```
# iptables -P INPUT DROP
# iptables -vnL

Chain INPUT (policy DROP)
target     prot opt source                         destination

Chain FORWARD (policy ACCEPT)
target     prot opt source                         destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source                         destination
```

СОВЕТ

Для политик нельзя установить цель REJECT. При такой попытке выдается сообщение iptables: Bad policy name. Вместо этого используйте в качестве политики DROP.

Host-B пытается пропинговать Host-A, а затем установить ssh-соединение, как показано в следующем примере. Здесь обе попытки провалились. Поскольку команда ping заблокирована, это не соответствует цели блокировать только удаленные подключения входа с помощью ssh от Host-B:

```
$ ping -c 2 10.140.67.23
PING 10.140.67.23 (10.140.67.23) 56(84) bytes of data.

--- 10.140.67.23 ping statistics ---
2 packets transmitted, 0 received, 100% packet loss, time 1007ms
$ ssh root@10.140.67.23

ssh: connect to host 10.140.67.23 port 22: Connection timed out
```

Попытки сервера Host-C пинговать Host-A и установить ssh-соединение терпят неудачу. Таким образом подтверждается, что настройка брандмауэра, где политика INPUT равна DROP, не поможет в достижении цели:

```
$ ping -c 2 10.140.67.23
PING 10.140.67.23 (10.140.67.23) 56(84) bytes of data.

--- 10.140.67.23 ping statistics ---
2 packets transmitted, 0 received, 100% packet loss, time 1008ms
$ ssh root@10.140.67.23

ssh: connect to host 10.140.67.23 port 22: Connection timed out
```

Блокировка источника IP-адресов. А что, если вместо этого будет заблокирован только IP-адрес сервера Host-B? Это позволило бы Host-C достичь сервера Host-A. Поможет ли эта настройка добиться желаемого?

В следующем примере политика DROP должна быть сначала изменена на ALLOW в iptables Host-A. После этого необходимо добавить специальное правило для блокировки сетевых пакетов только с IP-адреса Host-B 10.140.67.22:

```
# iptables -P INPUT ACCEPT
# iptables -A INPUT -s 10.140.67.22 -j DROP
# iptables -vnL
Chain INPUT (policy ACCEPT)
target     prot opt source                         destination
DROP all -- 10.140.67.22 anywhere

Chain FORWARD (policy ACCEPT)
Target     prot opt source                         destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source                         destination
```

Теперь сервер Host-C может успешно использовать ping и ssh на сервере Host-A, реализуя одну из поставленных целей:

```
$ ping -c 2 10.140.67.23
PING 10.140.67.23 (10.140.67.23) 56(84) bytes of data.
64 bytes from 10.140.67.23: icmp_req=1 ttl=64 time=11.7 ms
64 bytes from 10.140.67.23: icmp_req=2 ttl=64 time=0.000 ms

--- 10.140.67.23 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1008ms
rtt min/avg/max/mdev = 0.000/5.824/11.648/5.824 ms
$ ssh root@10.140.67.23
root@10.140.67.23's password:
```

Однако Host-B не может использовать ping и ssh на сервере Host-A. Таким образом, применяемое правило не помогает в достижении общей цели:

```
$ ping -c 2 10.140.67.23

PING 10.140.67.23 (10.140.67.23) 56(84) bytes of data.

--- 10.140.67.23 ping statistics ---
2 packets transmitted, 0 received, 100% packet loss, time 1007ms

$ ssh root@10.140.67.23

ssh: connect to host 10.140.67.23 port 22: Connection timed out
```

Блокировка протокола и порта. А если вместо полной блокировки IP-адреса сервера Host-B заблокировать только соединения с ssh-портом (порт 22) с IP-адреса Host-B? Даст ли серверу Host-C полный доступ к серверу Host-A то, что заблокированы только ssh-соединения от сервера Host-B?

В следующем примере правила iptables для сервера Host-A модифицируются, чтобы попытаться заблокировать IP-адрес Host-B из порта 22. Обратите внимание на то, что параметр --dport должен сопровождать определенный протокол, напри-

мер `-p tcp`. Перед добавлением нового правила следует удалить правило из предыдущего примера с помощью параметра `-D`. В противном случае оно будет использоваться брандмауэром `netfilter/iptables` для пакетов от 10.140.67.22 (Host-B):

```
# iptables -D INPUT 1
# iptables -A INPUT -s 10.140.67.22 -p tcp --dport 22 -j DROP
# iptables -vnL

Chain INPUT (policy ACCEPT)
Target     prot opt source      destination
DROP tcp -- 10.140.67.22 anywhere tcp dpt:ssh

Chain FORWARD (policy ACCEPT)
target     prot opt source      destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source      destination
```

Во-первых, новое правило `iptables` тестируется с сервера Host-C, чтобы гарантировать, что и попытки команды `ping`, и `ssh`-соединения не затрагиваются. Как видно в примере, это сработало:

```
$ ping -c 2 10.140.67.23
PING 10.140.67.23 (10.140.67.23) 56(84) bytes of data.
64 bytes from 10.140.67.23: icmp_req=1 ttl=64 time=1.04 ms
64 bytes from 10.140.67.23: icmp_req=2 ttl=64 time=0.740 ms

--- 10.140.67.23 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1000ms
rtt min/avg/max/mdev = 0.740/0.892/1.045/0.155 ms

$ ssh root@10.140.67.23
root@10.140.67.23's password:
```

Затем новое правило `iptables` тестируется с сервера Host-B, чтобы убедиться, что `ping` работает и `ssh`-соединения заблокированы. Как видно в примере, это тоже сработало:

```
$ ping -c 2 10.140.67.23

PING 10.140.67.23 (10.140.67.23) 56(84) bytes of data.
64 bytes from 10.140.67.23: icmp_req=1 ttl=64 time=1.10 ms
64 bytes from 10.140.67.23: icmp_req=2 ttl=64 time=0.781 ms

--- 10.140.67.23 ping statistics ---

2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 0.781/0.942/1.104/0.164 ms

$ ssh root@10.140.67.23

ssh: connect to host 10.140.67.23 port 22: Connection timed out
```

Матрица контроля доступа вашей организации (см. главу 22 «Базовые методы обеспечения безопасности») поможет создать необходимые правила для брандмауэра `netfilter/iptables` на сервере Linux. Затем нужно протестировать в тестовой или виртуальной среде каждую модификацию перед внедрением ее в брандмауэр рабочих систем Linux.

Сохранение настроек службы iptables. Поскольку `firewalld` — это рекомендуемая служба для создания брандмауэров в RHEL, Fedora и других системах Linux, ручная установка постоянных правил брандмауэра не очень популярна. Однако при желании вы можете вручную сохранять и восстанавливать правила брандмауэра, созданные непосредственно с помощью службы `iptables`.

В следующем примере внесенные ранее изменения зафиксированы в брандмауэре. Текущий набор правил фильтра брандмауэра можно сохранить с помощью команды `iptables-save`:

```
# iptables -vnL
Chain INPUT (policy ACCEPT 8 packets, 560 bytes)
 pkts bytes target prot opt in out source destination
    0     0 DROP   tcp -- *    * 10.140.67.22 0.0.0.0/0  tcp dpt:22
    0     0 DROP   tcp -- *    * 0.0.0.0/0    0.0.0.0/0  tcp dpt:33
    0     0 DROP   icmp -- *   * 0.0.0.0/0    0.0.0.0/0
...

# iptables-save > /tmp/myiptables
```

Чтобы восстановить эти правила позже, сначала сбросьте текущие правила (`iptables-F`), а затем восстановите их (`iptables-restore`):

```
# iptables -F
# iptables -vnL
Chain INPUT (policy ACCEPT 8 packets, 560 bytes)
 pkts bytes target prot opt in out source     destination
    0     0 DROP   tcp  --  *  *   0.0.0.0/0  0.0.0.0/0  tcp dpt:33
    0     0 DROP   icmp --  *  *   0.0.0.0/0  0.0.0.0/0
...
```

Сброс правил не влияет на файл конфигурации `iptables`. Чтобы восстановить исходное состояние брандмауэра, используйте команду `iptables-restore`. В следующем примере файл конфигурации `iptables` перенаправляется в команду `restore` и восстанавливается исходное правило `DROP` для 10.140.67.2:

```
# iptables-restore < /tmp/myiptables
# iptables -vnL
Chain INPUT (policy ACCEPT 16 packets, 1120 bytes)
 pkts bytes target prot opt in out source       destination
    0     0 DROP   tcp -- *    *   10.140.67.22 0.0.0.0/0  tcp dpt:22
    0     0 DROP   tcp -- *    *   0.0.0.0/0    0.0.0.0/0  tcp dpt:33
    0     0 DROP   icmp -- *   *   0.0.0.0/0    0.0.0.0/0
```

ПРИМЕЧАНИЕ

В системе Ubuntu сохранение и восстановление изменений службы netfilter/iptables похоже на то, как это происходит в Fedora. Можно использовать команду iptables-save, чтобы создать файл конфигурации iptables из текущей настройки iptables, а команду iptables-restore — чтобы его восстановить. Существует несколько вариантов загрузки файла конфигурации при загрузке системы. (См. сайт сообщества Ubuntu по адресу help.ubuntu.com/community/IptablesHowTo, чтобы изучить различные варианты.)

Вы также можете сохранить правила брандмауэра `netfilter/iptables` для создания отчета аудита. Просмотр этих правил должен быть частью этапа аудита/проверки жизненного цикла системы организации.

Резюме

Защита Linux-сервера в сети очень важна. Большинство вредоносных атак происходит именно из сети, особенно из Интернета. В этой главе были рассмотрены основы, необходимые для обеспечения безопасности сервера.

Защита сетевых служб может стать проще после определения и удаления всех ненужных служб. Здесь вам поможет утилита `nmap`. Кроме того, вы можете использовать утилиту `nmap` для аудита широковещательных сетевых служб вашего Linux-сервера. Подобный аудит помогает определить, какие модификации брандмауэра необходимы.

Последние версии систем Fedora и RHEL добавили службу `firewalld` в качестве внешнего интерфейса к брандмауэру `iptables`, встроенному в ядро Linux. С помощью инструмента `firewalld-config` и веб-интерфейса Cockpit вы можете легко открывать порты в брандмауэре, чтобы разрешить доступ к выбранным службам. Брандмауэр `netfilter/iptables` — это программный брандмауэр на основе хоста на сетевом уровне. Им управляют утилиты `iptables` и `ip6tables`. С их помощью можно создать ряд политик и правил для каждого сетевого пакета, проходящего через сервер Linux.

На данном этапе прочтения книги вы должны хорошо понимать, что относится к настройке и защите настольных и серверных систем Linux. В следующих двух главах мы распространим эти знания на облачные вычисления и виртуализацию.

Упражнения

Обратитесь к материалам этой главы, чтобы выполнить приведенные далее упражнения. Если затрудняетесь с решением заданий, воспользуйтесь ответами к упражнениям, приведенными в приложении Б (хотя Linux позволяет решать задачи разными способами). Выполните упражнение самостоятельно, прежде чем обратиться

к ответам. Задачи подходят для систем Fedora или Red Hat Enterprise Linux (некоторые сработают и в других системах Linux). Пожалуйста, не используйте рабочую систему, чтобы протестировать команды `iptables`, как предлагается в упражнениях. Хотя показанные здесь команды не изменят ваш брандмауэр навсегда (старые правила вернутся при перезапуске службы брандмауэра), его неправильные настройки могут открыть нежелательный доступ.

1. Установите утилиту Network Mapper в локальную систему Linux.
2. Запустите сканирование TCP-соединения на своем локальном петлевом (loopback) адресе. На каких портах работает служба?
3. Запустите проверку UDP-портов в своей системе Linux из удаленной системы.
4. Проверьте, работает ли в вашей системе служба `firewalld`. Если нет, установите `firewalld` и `firewall-config`, а затем запустите и включите эту службу.
5. Используйте окно Firewall Configuration (Настройка межсетевого экрана), чтобы открыть доступ к защищенным (TCP-порт 443) и небезопасным (TCP-порт 80) портам веб-службы.
6. Определите текущие политики и правила брандмауэра `netfilter/iptables` в своей системе Linux.
7. Сохраните текущие правила брандмауэра своей системы Linux, очистите их, а затем восстановите.
8. Для брандмауэра системы Linux установите политику таблицы `filter` для входной цепочки в значение `DROP`.
9. Измените политику таблицы `filter` брандмауэра системы Linux обратно на `accept` для входной цепочки, а затем добавьте правило для удаления всех сетевых пакетов с IP-адреса 10.140.67.23.
10. Не сбрасывая и не восстанавливая правила брандмауэра системы Linux, удалите только что добавленное правило.

Часть VI

Работа с облачными вычислениями

В этой части

- Глава 26. Работа с облаками и контейнерами.
- Глава 27. Облачные вычисления в системе Linux.
- Глава 28. Развертывание приложений Linux в облаке.
- Глава 29. Автоматизация приложений и инфраструктуры с помощью системы Ansible.
- Глава 30. Развертывание приложений в контейнеры с помощью кластера Kubernetes.

26 Работа с облаками и контейнерами

В этой главе

- Ключевые технологии облачных вычислений.
- Работа контейнеров Linux.
- Установка и запуск контейнерного программного обеспечения.
- Работа образов контейнеров.
- Перезапуск остановленного контейнера.
- Построение образа контейнера.
- Пометка и перемещение изображений контейнеров в реестр.

Хотя большая часть этой книги посвящена установке отдельных компьютеров, служб и приложений и управлению ими, в следующих главах рассматриваются технологии, необходимые для внедрения Linux в крупные центры обработки данных. Чтобы центр обработки данных работал эффективно, его компьютеры должны стать как можно более универсальными, а работающие компоненты — более автоматизированными. Главы данной части посвящены технологиям, благодаря которым это достигается.

Компьютеры становятся более универсальными, когда приложения отделяются от операционных систем. Это означает не только упаковку приложений в пакеты, которые устанавливаются в операционной системе (например, пакеты RPM или Deb), но и объединение наборов программного обеспечения в пакеты, которые могут работать сами по себе после отрыва их от операционной системы. *Виртуальные машины* (VM) и *контейнеры* — это два варианта упаковки наборов программного обеспечения и их зависимостей.

Если смотреть глубже, *виртуальная машина* — это полная операционная система, работающая в другой операционной системе, что позволяет одновременно иметь множество виртуальных машин на одном физическом компьютере. Все, что

необходимо для запуска приложения или службы, может храниться в этой виртуальной машине или в подключенном хранилище.

Виртуальная машина имеет собственные ядро, файловую систему, таблицу процессов, сетевые интерфейсы и другие функции операционной системы отдельно от хоста, при этом использует процессор и оперативную память совместно с хост-системой. Вы можете развернуть эту виртуальную машину в физической системе таким образом, чтобы легко было запустить приложение, а затем отключить ее, когда закончите работу с ней. Вы можете запустить несколько виртуальных машин на одном компьютере или клонировать VM и запустить на нескольких компьютерах. Термин *«виртуальная машина»* возник из-за того, что каждая VM видит эмуляцию компьютерного оборудования, а не непосредственно само оборудование.

Контейнер похож на виртуальную машину, с той лишь разницей, что у него нет собственного ядра. В большинстве других функций контейнер похож на VM в том смысле, что его пространства имен отделены от операционной системы хоста и вы можете перемещать их с хоста на хост, чтобы запускать там, где это удобно.

В главах этой части мы познакомимся с концепциями, инструментами и технологиями, необходимыми для работы с облачными вычислениями. Вы можете попробовать использовать виртуальные машины на одном хосте Linux с помощью KVM. Затем VM можно развернуть в облачных хранилищах, таких как OpenStack и Amazon Web Services (AWS).

Чтобы развернуть наборы хостов, будь то коммутатор без ОС или облако, необходимо применять технологию Ansible. С помощью Ansible playbooks вы можете также определять программное обеспечение, которое устанавливается и запускается на каждой хост-системе.

Что касается контейнеров, то проект Kubernetes привлек внимание в качестве ведущей технологии организации огромного количества контейнеров в крупных центрах обработки данных. Такие продукты, как Red Hat OpenShift, обеспечивают поддержку платформ Kubernetes на крупных предприятиях.

Технология, которая несколько лет назад положила начало потоку контейнеров, — это проект Docker. Команда `docker` и ее демон предлагали упрощенные способы создания и запуска контейнеров в системах Linux. Сегодня стандартизированные контейнерные форматы (такие как Open Container Initiative) и другие контейнерные инструменты, например `podman`, предлагают способы работы с контейнерами, лучше согласующиеся с экосистемой Kubernetes.

Дальнейший материал этой главы посвящен началу работы с контейнерами. Здесь рассматриваются команды `docker` и `podman`, а также другие популярные инструменты для работы с отдельными контейнерами.

Контейнеры в системе Linux

Контейнеры упрощают загрузку и запуск приложений, а затем отменяют их работу, когда вы закончите. Есть несколько вещей, которые необходимо знать о контейнерах, прежде чем приступать к работе.

Работая с контейнерами, пользователи называют объект, который вы перемещаете, *образом контейнера* (или просто *образом*). Когда вы запускаете этот образ или когда он приостановлен либо остановлен, он называется *контейнером*.

Контейнер остается отделенным от хост-системы и использует собственный набор пространств имен. Обычно пользователи создают свои образы контейнеров, получая защищенный базовый образ, а затем добавляют собственные слои программного обеспечения поверх этого образа, чтобы создать новый. Чтобы поделиться своими образами, необходимо переместить их в общие реестры контейнеров, доступ к которым есть у других пользователей.

Пространство имен

Поддержка в Linux *пространств имен* — это то, что позволяет контейнерам существовать. С помощью пространств имен ядро Linux может связать один или несколько процессов с набором ресурсов. Обычные процессы, а не те, которые выполняются в контейнере, используют одни и те же пространства имен хостов. По умолчанию процессы в контейнере видят только пространства имен контейнера, а не пространства имен хоста. Пространства имен включают в себя следующее.

- **Таблица процессов.** Контейнер имеет собственный набор идентификаторов процессов и по умолчанию может видеть только процессы, запущенные внутри контейнера. И если PID 1 на хосте — это процесс инициализации (`systemd`), то в контейнере PID 1 является первым процессом, запущенным внутри него.
- **Сетевой интерфейс.** По умолчанию контейнер имеет один сетевой интерфейс (`eth0`), и при запуске контейнера назначается IP-адрес. По умолчанию служба, запущенная внутри контейнера (например, веб-сервер, прослушивающий порты 80 и 443), недоступна за пределами хост-системы. Плюсом этого является то, что сотни веб-серверов могут работать на одном хосте и не будут между собой конфликтовать. Недостаток заключается в том, что вам нужно управлять тем, как эти порты отображаются за пределами хоста.
- **Таблица монтирования.** По умолчанию контейнер не может видеть корневую файловую систему хоста или любую другую смонтированную файловую систему, указанную в таблице монтирования хоста. Контейнер имеет собственную файловую систему, состоящую из приложения и любых зависимостей, которые ему нужно запустить. Файлы или каталоги, требующиеся от хоста, могут быть выборочно привязаны внутри контейнера.
- **ID пользователя.** Хотя контейнеризованные процессы выполняются как UID в пространстве имен хоста, другой набор UID вложен в контейнер. Это может, например, позволить процессу работать от имени суперпользователя в контейнере, но не иметь никаких особых привилегий для хост-системы.
- **Пространство UTS.** Пространство имен UTS позволяет контейнеризованному процессу иметь другой хост и доменное имя, отличное от имени хоста.

- **Контрольная группа (cgroup).** В некоторых системах Linux, таких как Fedora и RHEL, контейнеризованный процесс выполняется в пределах выбранной контрольной группы и не может видеть другие контрольные группы, доступные в хост-системе.
- **Межпроцессное взаимодействие (inter-process communication, IPC).** Контейнеризованный процесс не может видеть пространство имен IPC с хоста.

Хотя доступ к любому пространству имен хостов ограничен по умолчанию, привилегии для пространств имен можно открывать выборочно. Так что вы можете выполнять такие действия, как монтирование файлов конфигурации или данных внутри контейнера и сопоставление портов контейнера с портами хоста, чтобы выставить их за пределы хоста.

Реестры контейнеров

Контейнеры постоянно хранятся в так называемом *реестре контейнеров*. Когда вы хотите поделиться созданным образом контейнера, поместите его в публичный или частный реестр, который ведете сами (например, реестр Red Hat Quay). Если кто-то захочет использовать образ, он возьмет его из реестра.

Существуют большие общедоступные реестры образов контейнеров, например Docker Hub (docker.io) и Quay Registry (Quay.io). На этих сайтах можно зарегистрироваться бесплатно и начать работу с контейнерами. Если вы хотите получить доступ к дополнительным функциям, например сохранить свой реестр в тайне, необходимо приобрести премиум-аккаунт.

Базовые образы и слои

Вы можете создать контейнер с нуля, однако чаще всего он создается с помощью хорошо известного базового образа и нового программного обеспечения. Базовый образ обычно совпадает с операционной системой, из которой вы устанавливаете программное обеспечение в свой контейнер.

Можете скачать официальные базовые образы для Ubuntu (hub.docker.com/_/ubuntu), CentOS (hub.docker.com/_/centos), Fedora (hub.docker.com/_/fedora) и многих других дистрибутивов Linux. Эти дистрибутивы используют различные формы базовых образов, например стандартные и упрощенные версии. Существуют также базовые образы, которые можно построить на основе среды php, Perl, Java и других сред разработки.

Red Hat использует подписку для работы со своим программным обеспечением, но если вы хотите взять его в качестве основы для образов контейнеров, то у Red Hat в свободном доступе есть каталог Red Hat Container Catalog (UBIs) для стандартных, минимальных и многих других контейнеров времени выполнения. Эти образы можно найти, применяя поиск UBI в каталоге контейнеров Red Hat (catalog.redhat.com/software/containers/explore).

Вы можете добавить программное обеспечение в базовый образ с помощью таких команд, как `docker`, `build` или `podman`. Используя Dockerfile для определения сборки, добавьте команду `yum apt-get` для установки программного обеспечения из репозиториев в свой новый контейнер.

Программное обеспечение, добавленное к образу, создает новый слой, чтобы стать частью нового образа. Повторное применение одних и тех же базовых образов для создаваемых контейнеров дает ряд преимуществ. Одним из них является то, что при запуске образа контейнера на хосте требуется только одна копия базового образа. Таким образом, если вы запускаете десять различных контейнеров на основе одного и того же базового образа, вам нужно только один раз вытащить и сохранить базовый образ, а затем добавить лишь несколько мегабайт дополнительных данных для каждого нового образа.

Если вы посмотрите на содержимое базового образа, то увидите, что он выглядит, как небольшая файловая система Linux. Файлы конфигурации находятся в каталоге `/etc`, исполняемые файлы — в каталогах `/bin` и `/sbin`, а библиотеки — в каталоге `/lib`. Другими словами, он будет получать основные компоненты, необходимые приложению, из хост-системы Linux.

Имейте в виду, что образы контейнеров, которые вы запускаете, не обязательно должны соответствовать хост-системе Linux. Так, например, вы можете запустить базовый образ Fedora в системе Ubuntu, если в образ контейнера не встроены конкретные требования к ядру или библиотеке.

Начало работы с контейнерами Linux

Для запуска контейнеров в системе Linux требуется небольшая подготовка. Следующие действия описывают, как подготовить систему Linux к началу работы с контейнерами.

У Docker Inc. теперь есть бесплатная версия программного обеспечения, доступная через проект Moby, а сам Docker стал коммерческим продуктом. Чтобы опробовать более старую версию пакета `docker`, можете, установив пакет `docker`, а затем запустив и включив службу `docker`, выполнить в системе RHEL 7 следующие действия:

```
# yum install docker -y
# systemctl start docker
# systemctl enable docker
```

Команда `podman` поддерживает большинство параметров командной строки `docker` для работы с контейнерами, поэтому ее можно использовать вместо команды `docker`. Имейте в виду, что у команды `podman` другая кодовая база, в отличие от `docker`, хотя и поддерживает аналогичные параметры команд управления. С командой `podman` вам не нужно запускать службу, как происходит во время работы с командой `docker`. Чтобы установить пакет `podman` на Fedora или RHEL, выполните следующую команду:

```
# yum install podman -y
```

Теперь можно применять команды podman или docker для работы с контейнерами и образами контейнеров для проработки примеров этой главы.

Загрузка и запуск контейнеров

Установив и подготовив к использованию пакеты docker или podman, вы можете попробовать запустить контейнер. Для начала перетащите контейнер в свою локальную систему и запустите его. При желании можете пропустить команду pull, так как запуск контейнера приведет к ошибке, если запрошенного образа еще нет в вашей системе.

Загрузка контейнера

Выберите надежный образ контейнера для тестирования из официального источника, актуального и желательно проверенного на наличие уязвимостей. Вот пример загрузки базового образа RHEL 8 UBI с помощью команды podman (в этих примерах можно заменить команду podman на docker):

```
# podman pull registry.access.redhat.com/ubi8/ubi
Trying to pull .../ubi8/ubi...Getting image source signatures
Copying blob fd8daf2668d1 done
Copying blob cb3c77f9bdd8 done
Copying config 096cae65a2 done
Writing manifest to image destination
Storing signatures
096cae65a2078ff26b3a2f82b28685b6091e4e2823809d45aef68aa2316300c7
```

Чтобы убедиться, что образ находится в вашей системе, выполните следующие действия:

```
# podman images
REPOSITORY   TAG      IMAGE ID       CREATED       SIZE
/ubi8/ubi    latest   096cae65a207   2 weeks ago   239 M
```

Запуск командной оболочки из контейнера

Используйте команды podman или docker для запуска оболочки внутри контейнера. Можно идентифицировать образ либо по его идентификатору (096cae65a207), либо по имени (registry.access.redhat.com/ubi8/ubi). Задействуйте параметры -i (interactive) и -t (terminal), чтобы подключить интерактивный сеанс внутри контейнера из оболочки bash:

```
# podman run -it 096cae65a207 bash
[root@e9086da6ed70 /]#
```

При запущенной оболочке команды, которые вы вводите, будут работать в контейнере. Например, вы перечисляете файловую систему контейнера или

проверяете файл `os-release`, чтобы увидеть операционную систему, на которой основан контейнер:

```
[root@e9086da6ed70 /]# ls /
bin   dev  home   lib64       media  opt   root  sbin  sys   usr
boot  etc  lib    lost+found  mnt    proc  run   srv   tmp   var
[root@e9086da6ed70 /]# cat /etc/os-release | grep ^NAME
NAME="Red Hat Enterprise Linux"
```

Поскольку контейнеры предназначены для хранения минимального объема содержимого, необходимого для запуска предполагаемого приложения, многих стандартных инструментов в контейнере может не быть. Вы можете установить программное обеспечение в работающий контейнер. Однако имейте в виду, что контейнеры — вещь непостоянная. Поэтому, если хотите добавить программное обеспечение на постоянной основе, необходимо создать новый образ, чтобы включить в него нужное ПО.

Вот пример добавления программного обеспечения в работающий контейнер:

```
[root@e9086da6ed70 /]# yum install procps iproute -y
```

Теперь можно запускать команды как `ps` и `ip` внутри контейнера.

```
[root@e9086da6ed70 /]# ps -ef
UID        PID  PPID  C STIME TTY          TIME CMD
root         1     0  0 17:44 pts/0    00:00:00 bash
root        40     1  0 17:45 pts/0    00:00:00 ps -ef
[root@e9086da6ed70 /]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 ...
    inet 127.0.0.1/8 scope host lo
    ...
3: eth0@if11: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 ...
    inet 10.88.0.6/16 brd 10.88.255.255 scope global eth0
    ...
```

Обратите внимание на то, что внутри контейнера запущены только два процесса — оболочка и команда `ps`. PID 1 — это оболочка bash. Часть вывода из команды `ip a` показывает, что существует только один внешний сетевой интерфейс из контейнера (`eth0@if11`) и ему присвоен IP-адрес 10.88.0.6/16.

Закончив, введите команду `exit`, чтобы выйти из оболочки и остановить контейнер:

```
[root@e9086da6ed70 /]# exit
```

Хотя оболочка и контейнер больше не работают, контейнер по-прежнему доступен в вашей системе в остановленном состоянии. Обратите внимание, что просто команда `podman ps` не отображает контейнер — нужно добавить параметр `--all`:

```
[root@e9086da6ed70 /]# podman ps
CONTAINER ID  IMAGE  COMMAND  CREATED  STATUS  PORTS  NAMES
[root@e9086da6ed70 /]# podman ps --all
```

```
CONTAINER ID IMAGE         COMMAND CREATED    STATUS        PORTS
NAMES
437ec53386ca ...ubi:latest bash    1 hour ago Up 1 minute ago
go_ein
```

Позже вы узнаете, как удалить и перезапустить остановленный контейнер.

Запуск FTP-сервера из контейнера

Всегда должна быть возможность удалить контейнер, закончив работу с ним, но при этом сохранять любые изменяемые данные вне контейнера. Далее приведен простой пример запуска FTP-сервера (`vsftpd`) из контейнера. Если хотите решить этот пример самостоятельно, рекомендую перейти к подразделу «Создание образа контейнера» далее в этом разделе, чтобы узнать, как самостоятельно создать образ контейнера `vsftpd`.

Для этого вам понадобятся файл конфигурации (`vsftpd.conf`) и каталог FTP, содержащий один или два файла для совместного использования (`/var/ftp/pub`) в хост-системе. Когда контейнер `vsftpd` запускается, он связывает эти элементы в контейнер в виде томов.

1. **Создайте файл vsftpd.conf.** Создайте файл `vsftpd.conf` в папке по умолчанию `/etc/vsftpd/vsftpd.conf` (дополнительные сведения см. на справочной странице `vsftpd.conf`), например:

   ```
   anonymous_enable=YES
   local_enable=YES
   write_enable=YES
   local_umask=022
   dirmessage_enable=YES
   xferlog_enable=NO
   connect_from_port_20=YES
   listen=NO
   listen_ipv6=YES
   pam_service_name=vsftpd
   userlist_enable=YES
   tcp_wrappers=NO
   vsftpd_log_file=/dev/stdout
   syslog_enable=NO
   background=NO
   pasv_enable=Yes
   pasv_max_port=21100
   pasv_min_port=21110
   ```

2. **Создайте каталог ftp.** Создайте анонимный FTP-каталог службы `vsftpd` для совместного использования в месте стандартного расположения на хосте (`/var/ftp/pub`) и скопируйте в него несколько файлов:

   ```
   # mkdir -p /var/ftp/pub
   # cp /etc/services /etc/login.defs /var/ftp/pub/
   ```

3. **Загрузите образ контейнера vsftpd.** Создайте образ `vsftpd`, как описано в разделе «Создание образа контейнера» далее в этой главе. Чтобы проверить, можно ли его запустить, введите следующее:

```
# podman images
REPOSITORY  TAG    IMAGE ID     CREATED        SIZE
vsftpd      latest 487d0db26098 5 seconds ago  208 MB
```

4. **Запустить образ контейнера vsftpd.** При запуске контейнера `vsftpd` необходимо предоставить доступ к портам и монтировать файлы с хоста в контейнер. Вот пример (вы можете использовать команду `docker` вместо `podman`):

```
# podman run -d -p 20:20 -p 21:21 \
   -p 21100-21110:21100-21110 \
-v /etc/vsftpd/:/etc/vsftpd/ \
-v /var/ftp/pub:/var/ftp/pub \
--name vsftpd vsftpd
# podman ps
CONTAINER ID  IMAGE         COMMAND          CREATED
    STATUS             PORTS                      NAMES
3a5d094dd4b5  vsftpd:latest /usr/local/s2... 9 seconds ago
    Up 10 seconds ago  0.0.0.0:20-21->20-21/tcp  vsftpd
```

В примере используются следующие параметры:

- `-d` — запускает контейнер отдельно, поэтому служба `vsftpd` работает в фоновом режиме;
- `-p` — стандартные FTP-порты (TCP 20 и TCP 21) сопоставляются с одинаковыми номерами портов на сетевых интерфейсах хоста, поэтому доступ к службе можно получить за пределами локального хоста. Ряд портов, необходимых для пассивного сервера FTP, также открыт для их аналогов на хосте (`21100-21110`);
- `--rm` — хотя этот параметр не входит в пример, при добавлении в командную строку он будет удалять контейнер при выходе из него;
- `-v` — чтобы использовать файлы конфигурации (каталог `/etc/vsftpd`) и содержимое, которым вы хотите поделиться (каталог `/var/ftp`) из хост-системы, эти каталоги привязываются к тем же местоположениям в контейнере с параметром `-v`;
- `--name` — установите имя контейнера `vsftpd` (или любое другое, которое вам нравится).

Имейте в виду, что можно привязать содержимое контейнера и порты к другим местам расположения на хосте. Таким образом вы получите несколько версий одного и того же программного обеспечения, работающих на одном хосте и не конфликтующих друг с другом.

Чтобы сделать свою службу `vsftpd` доступной за пределами локальной системы, обязательно откройте FTP-порты и пассивные FTP-порты, которые вы только что назначили, следующим образом:

```
# firewall-cmd --zone=public --add-service=ftp
# firewall-cmd --zone=public \
     --permanent --add-service=ftp
```

```
# firewall-cmd --zone=public \
     --add-port=21100-21110/tcp
# firewall-cmd --zone=public \
     --permanent --add-port=21100-21110/tcp
# firewall-cmd reload
```

Теперь можно применить любой FTP-клиент для доступа к FTP-службе через анонимного пользователя. Для дальнейшей настройки службы `vsftpd` и проверки ее работоспособности обратитесь к главе 18 «Настройка FTP-сервера».

Запуск и остановка контейнеров

Если вы специально не настраивали контейнер для удаления при его остановке (параметр `--rm`), то если контейнер остановлен, приостановлен или просто выходит из строя, он все еще будет находиться в вашей системе. Вы можете увидеть состояние всех контейнеров в системе (запущенных в данный момент или нет) с помощью параметра `ps`:

```
# podman ps
CONTAINER ID  IMAGE                       COMMAND           CREATED
    STATUS                PORTS                      NAMES
4d6be3e63fe3  localhost/vsftpd:latest  /usr/local/s2...  About an
hour ago
    Up About an hour ago  0.0.0.0:20-21->20-21/tcp  vsftpd
# podman ps -a
CONTAINER ID  IMAGE                       COMMAND           CREATED
    STATUS                PORTS                      NAMES
7da88bd62667  ubi8/ubi:latest          bash              2 minutes
ago
    Exited 7 seconds ago silly_wozniak
4d6be3e63fe3  localhost/vsftpd:latest  /usr/local/s2i/ru... About an
hour ago
    Up About an hour ago  0.0.0.0:20-21->20-21/tcp  vsftpd
```

Только запущенные контейнеры отображаются с помощью `podman ps`. Добавив параметр `-a`, вы можете увидеть все контейнеры, включая те, которые больше не работают, но еще не удалены. Вы можете перезапустить существующий неработающий контейнер, используя параметр `start`.

```
# podman start -a 7da88bd62667
[root@7da88bd62667 /]#
```

Перезапущенный контейнер запускает оболочку bash. Поскольку сеанс терминала контейнера уже существовал, не было необходимости начинать новый (`-it`). Нужно было прикрепить его (`-a`) к существующему сеансу. Контейнер, который только что работал в отключенном режиме, можно просто запустить и остановить по мере необходимости:

```
# podman stop 4d6be3e63fe3
4d6be3e63fe3...
# podman start 4d6be3e63fe3
4d6be3e63fe3
```

Обратите внимание на то, что, если контейнер запущен с параметром `--rm`, он будет удален, как только вы его остановите. Таким образом, вам придется запустить новый контейнер вместо того, чтобы просто перезапустить старый. Поскольку файлы конфигурации и данные хранятся вне контейнера, запустить новый можно легко и безболезненно. Обновить приложение в будущем так же просто, как удалить старый контейнер и запустить его из обновленного образа контейнера.

Создание образа контейнера

Чтобы создать образ контейнера, вам нужен лишь файл Dockerfile, описывающий, как создать образ и любое содержимое, которое вы хотите включить в него. Следующие действия описывают, как создать простой контейнер из вашего собственного файла Dockerfile и как загрузить программное обеспечение, необходимое для создания службы `vsftpd`, в контейнер из программного обеспечения, доступного на веб-сервисе GitHub.

Создание простого образа контейнера

Сделав приведенные далее шаги, вы создадите простой контейнер из файла Dockerfile. Сначала формируются файл Dockerfile и простой скрипт, затем они добавляются в новый образ контейнера.

1. Создайте каталог, в котором будет храниться контейнер, и введите следующее:

```
# mkdir myproject
# cd myproject
```

2. Создайте скрипт под названием `cworks.sh` в этом каталоге, содержащем следующий текст:

```
#!/bin/bash
set -o errexit
set -o nounset
set -o pipefail
echo "This Container Works!"
```

3. Создайте в этом каталоге файл с именем `Dockerfile` и следующим содержимым:

```
FROM registry.access.redhat.com/ubi7/ubi-minimal
COPY ./cworks.sh /usr/local/bin/
CMD ["/usr/local/bin/cworks.sh"]
```

4. Создайте образ контейнера с именем `myproject` из файла `Dockerfile`:

```
# podman build -t myproject .
STEP 1: FROM registry.access.redhat.com/ubi7/ubi-minimal
STEP 2: COPY ./cworks.sh /usr/local/bin/
6382dfd00f7bedf1a64c033515a09eff37cbc6d1244cbeb4f4533ad9f00aa970
STEP 3: CMD ["/usr/local/bin/cworks.sh"]
STEP 4: COMMIT myproject
6837ec3a37a241...
```

5. Запустите контейнер, чтобы проверить, работает ли он. Для этого можно использовать либо имя контейнера (`myproject`), либо идентификатор его образа (`6837ec3a37a241`):

```
# podman run 6837ec3a37a241
The Container Works!
```

Создание FTP-контейнера из веб-сервиса GitHub

Следующие шаги позволяют загрузить программное обеспечение, необходимое для создания службы `vsftpd`, в контейнер из веб-сервиса GitHub. Затем мы рассмотрим, как настроить и запустить этот контейнер.

1. Если у вас еще нет пакета `git`, установите его в своей локальной системе:

```
# yum install git -y
```

2. Начнем с проекта `vsftpd container-images` на GitHub. Создайте копию этого программного обеспечения в локальном каталоге следующим образом:

```
# git clone
https://github.com/container-images/vsftpd.git
# cd vsftpd
# ls
default-conf/  Dockerfile  LICENSE  Makefile  README.md
root/          s2i/        tests/
```

3. Изменяйте файлы по мере необходимости. В частности, перейдите в Dockerfile и используйте последний доступный образ Fedora. Опуская `:tag` в конце имени изображения, следует искать версию этого изображения, которая включает в себя тег `:latest` — специальный тег, идентифицирующий последнюю доступную версию этого образа. Например, измените строку `FROM` в начале так, чтобы она выглядела следующим образом:

```
FROM registry.fedoraproject.org/fedora
```

4. Из каталога `vsftpd` используйте команду `docker` или `podman` для создания образа контейнера, например:

```
# podman build -t vsftpd .
STEP 1: FROM registry.fedoraproject.org/fedora:31
Getting image source signatures
Copying blob c0a89efa8873 done
Copying config aaaa3e1d6a done
Writing manifest to image destination
Storing signatures
STEP 2: ENV SUMMARY="Very Secure Ftp Daemon" ...
STEP 3: LABEL maintainer="Dominika Hodovska <dhodovsk@redhat.com>"
...
STEP 4: RUN dnf install -y vsftpd && dnf clean all
          && mkdir /home/vsftpd
...
```

```
Installing:
vsftpd    x86_64   3.0.3-32.fc31   fedora   164 k
...
Complete!
99931652dceacc2e9...
STEP 5: VOLUME /var/log/vsftpd
b79b229d09f726356...
STEP 6: EXPOSE 20 21
b0af5428800140104...
STEP 7: RUN mkdir -p ${APP_DATA}/src
b3652e0d07e35af79...
STEP 8: WORKDIR ${APP_DATA}/src
f9d96dee640c5cedc...
STEP 9: COPY ./s2i/bin/ /usr/local/s2i
ded9b512693ccabaa...
STEP 10: COPY default-conf/vsftpd.conf /etc/vsftpd/vsftpd.conf
0c48af8d4f72b76c7...
STEP 11: CMD ["/usr/local/s2i/run"]
STEP 12: COMMIT vsftpd
aa0274872f23ae94dfee...
```

5. Убедитесь, что новый образ создан:

```
# podman images
REPOSITORY       TAG    IMAGE ID     CREATED       SIZE
localhost/vsftpd latest aa0274872f23 4 minutes ago 607 MB
```

Целиком процесс состоит из 12 шагов. Строка `FROM` на первом шаге извлекает образ `fedora` из реестра контейнеров `registry.fedoraproject.org`. На каждом последующем шаге выполняется команда. Если содержимое добавляется во время выполнения команды, для образа создается новый слой. Шаги 2 и 3 устанавливают переменные среды и метки, которые используются во время сборки, а также для идентификации атрибутов образа контейнера при его дальнейшем применении.

Шаг 4 запускает команду `dnf`, которая устанавливает пакет `vsftpd` из репозитория Fedora `yum` в контейнер. Обратите внимание на то, что инструкция `RUN` содержит как `dnf install`, так и `dnf clean`. Это предотвращает включение дополнительного слоя кэшированных данных `dnf` в образ.

Шаг 5 определяет том, используемый для хранения файлов журнала `vsftpd`. Шаг 6 добавляет TCP-порты 20 и 21 для службы FTP. Обратите внимание: даже если порты открыты (то есть их можно увидеть снаружи контейнера), они все равно должны быть сопоставлены с портами хоста при последующем запуске контейнера, если вы хотите, чтобы эти порты были доступны за пределами локальной системы.

Шаги 7 и 8 создают каталог и устанавливают его в качестве рабочего каталога для приложения. Шаг 9 копирует скрипты `source-to-image` (`s2i`) в контейнер для запуска службы `vsftpd`. Шаг 10 копирует файл конфигурации по умолчанию `vsftpd.conf` в контейнер.

На шаге 11 инструкция CMD устанавливает каталог `/usr/local/s2i/run` в качестве команды по умолчанию, если контейнер запускается без ее переопределения. Шаг 12 фиксирует окончательный образ `vsftpd` в локальном хранилище (которое можно увидеть, набрав команду `podman images`).

Дополнительные сведения о создании и использовании Dockerfile для создания образов контейнеров см. в справочнике Dockerfile (docs.docker.com/engine/reference/builder/). Чтобы узнать больше о вариантах построения образов контейнеров, обратитесь к справочным страницам команды `podman` (`man podman build`).

Добавление тегов и загрузка образа в реестр

До сих пор я показывал, как создавать образ контейнера и запускать его в вашей локальной системе. Чтобы сделать образ доступным для других пользователей в других системах, необходимо добавить его в реестр контейнеров.

Следуйте дальнейшим инструкциям, чтобы пометить образ в локальной системе и отправить его в удаленный реестр контейнеров.

Чтобы опробовать простой реестр в локальной системе, установите дистрибутив `docker` в систему Fedora или RHEL 7. Можете также завести учетные записи в публичных реестрах контейнеров, например в Quay.io и Docker Hub. Как бесплатные пробные версии, так и подписки доступны на странице Quay.io (quay.io/plans/). Вдобавок можете настроить и запустить собственный поддерживаемый реестр контейнеров, например Red Hat Quay (www.openshift.com/products/quay).

Чтобы начать работу, установите пакет docker-distribution в локальную систему, а затем пометьте и вставьте в него образ.

1. **Установите дистрибутив docker.** В системах RHEL 7 или Fedora последней версии установите и запустите дистрибутив `docker`:

```
# yum install docker-distribution -y
# systemctl start docker-distribution
# systemctl enable docker-distribution
# systemctl status docker-distribution
• docker-distribution.service-v2 Registry server for Docker
  Loaded: loaded
   (/usr/lib/systemd/system/docker-distribution.service;
   enabled; vendor pres>
   Active: active (running) since Wed 2020-01-01...
```

2. **Откройте порт реестра.** Чтобы иметь возможность загружать и устанавливать образы контейнеров из других хост-систем, необходимо открыть TCP-порт 5000 на брандмауэре:

```
# firewall-cmd --zone=public
--add-port=5000/tcp --permanent
```

3. **Установите тег к образу.** Пометив локальный образ, определите местоположение реестра, в котором образ будет храниться. Замените идентификатор образа и `host.example.com` идентификатором образа и именем хоста или IP-адресом, чтобы пометить изображение:

```
# podman images | grep vsftpd
localhost/vsftpd  latest  aa0274872f23  2 hours ago  607 MB
# podman tag aa0274872f23
host.example.com:5000/myvsftpd:v1.0
```

4. **Добавьте образ в реестр.** Вставьте образ в локальный реестр (замените имя хоста или IP-адрес). Отключите `tls-verify`, так как `docker-registry` использует протокол `http`:

```
# podman push --tls-verify=false
host.example.com:5000/myvsftpd:v1.0
```

5. **Загрузите образ из реестра.** Чтобы убедиться, что образ можно загрузить из реестра, попробуйте сделать это. Для этого либо удалите образ из локальной системы, либо перейдите на другой хост:

```
# podman pull --tls-verify=false \
              host.example.com:5000/myvsftpd:v1.0
```

Теперь у вас есть возможность делиться своими образами с другими пользователями из собственного реестра.

Для загрузки из публичного реестра и добавления в него образа необходимо выполнить следующее:

```
# podman login quay.io
Username: myownusername
Password: ***************
# podman tag aa0274872f23 \
    quay.io/myownusername/myvsftpd:v1.0
# podman push quay.io/myownusername/myvsftpd:v1.0
```

Контейнеры на предприятии

Проект Docker успешно упростил использование отдельных контейнеров, однако именно проект Kubernetes помог продвинуть применение контейнеров Linux на предприятиях. В то время как инструменты командной строки, такие как `docker` и `podman`, задействуются для управления отдельными контейнерами, Kubernetes предлагает целую платформу для развертывания больших и сложных приложений в огромных центрах обработки данных. Сведения о том, как использовать Kubernetes для развертывания контейнерных приложений на предприятии и управления ими, см. в главе 30 «Развертывание приложений в контейнеры с помощью кластера Kubernetes».

Резюме

В последние несколько лет контейнеры получили широкое распространение. Проект Docker внес огромный вклад в упрощение контейнеризации отдельных приложений и запуска их на некоторых системах. Такие инструменты, как `podman`, также стали доступны для развертывания и управления отдельными контейнерами в системах Linux.

В этой главе описано, как извлекать, запускать, создавать контейнеры и управлять ими с помощью инструментов командной строки, `docker` и `podman`. Эти знания являются основой понимания того, как работает контейнеризация и как это используется в Kubernetes для управления контейнерными приложениями на предприятии в целом, что будет описано в главе 30.

Упражнения

Упражнения в этом разделе связаны с работой с контейнерами. Если затрудняетесь с решением заданий, воспользуйтесь ответами к упражнениям, приведенными в приложении Б (хотя Linux позволяет решать задачи разными способами).

1. Выберите инструмент `podman` (для любой системы RHEL или Fedora) или `docker` (RHEL 7), установите пакет программного обеспечения, содержащий этот инструмент, и запустите все необходимые службы для использования данных команд.
2. С помощью команды `docker` или `podman` добавьте этот образ на свой хост:

   ```
   registry.access.redhat.com/ubi7/ubi
   ```

3. Запустите образ `ubi7/ubi`, чтобы открыть оболочку bash.
4. Когда оболочка bash открыта внутри контейнера, выполните несколько команд, чтобы увидеть операционную систему, на которой основан контейнер, установите пакет `proc-ps`, выполните команду, чтобы увидеть процессы, запущенные внутри контейнера, а затем выйдите.
5. Снова перезагрузите контейнер и подключитесь к нему с помощью интерактивной оболочки. Выйдите из оболочки, когда закончите.
6. Создайте простой файл Dockerfile из базового образа `ubi7/ubi`, включите скрипт `cworks.sh`, который выводит строку `The Container Works!` и добавляет этот скрипт к образу, чтобы он выполнялся как команда по умолчанию.
7. Используйте команду `docker` или `podman` для создания образа `containerworks` из только что созданного файла Dockerfile.
8. Получите доступ к реестру контейнеров, установив дистрибутив `docker` или создав учетную запись на Quay.io или Docker Hub.
9. Пометьте и вставьте новый образ в выбранный реестр контейнеров.

27 Облачные вычисления в системе Linux

В этой главе

- Использование системы Linux в облаках.
- Базовые облачные технологии.
- Настройка гипервизора.
- Создание виртуальных машин.

Операционные системы изначально предназначались для установки непосредственно на компьютерное оборудование. Если требовались память, хранилище, вычислительная мощность или сетевые интерфейсы, компьютерная операционная система искала физическую оперативную память, жесткие диски, процессоры и карты сетевых интерфейсов. Если данных ресурсов требовалось больше, чем установлено, приходилось добавлять их вручную и физически. В настоящее время виртуализация таких элементов — это то, что делает облачные вычисления возможными.

Виртуализация в том, что касается компьютеров, — это акт создания вычислительных ресурсов, которые изначально были спроектированы как физические объекты, а далее стали виртуальными. Например, виртуальная операционная система, называемая *виртуальной машиной* (VM), не взаимодействует напрямую с физическим обеспечением. Она взаимодействует со специально сконфигурированным хост-компьютером, называемым гипервизором, поэтому вместо того, чтобы запускать одну операционную систему на физическом компьютере, вы можете запустить десятки или даже сотни виртуальных машин на одном физическом компьютере.

Преимущества запуска виртуальных машин огромны. На одном компьютере могут работать не только несколько операционных систем, но и разные системы — Linux, BSD, Windows или любая другая, созданная для работы на аппаратном обеспечении компьютера. Если вам нужно выключить главный компьютер для

обслуживания, можете перенести запущенные виртуальные машины на другой гипервизор и тем самым допустить лишь незначительные простои.

Чтобы поддерживать виртуальные машины на нескольких гипервизорах, вы можете виртуализировать функции их поддержки. Например, виртуальные сети и виртуальное хранилище могут работать с несколькими гипервизорами, поэтому, если VM должна перейти на другой гипервизор, на ней будут доступны те же виртуальные сети и хранилище.

Вам не нужно строить целый центр обработки данных, чтобы начать понимать виртуализацию и использовать некоторые из базовых технологий, делающих облачные вычисления возможными. Эта глава поможет настроить главный компьютер для работы в качестве гипервизора, запустить на нем виртуальные машины, а затем научиться переносить их на другие гипервизоры, чтобы предотвратить простои или просто увеличить емкость.

Облачные вычисления в Linux

Облачные вычисления привели нас к тому, что все, чему вы научились ранее в этой книге, отходит на второй план и автоматизируется. В облачной среде при установке системы не нужно загружаться с физического DVD, стирать локальный жесткий диск и устанавливать Linux непосредственно на физический компьютер. Нет необходимости входить в установленную систему и настраивать вручную программное обеспечение и функции, которые хотите запустить в этой системе.

Вместо этого вы устанавливаете систему на виртуальную машину или запускаете контейнер, который находится в хост-системе в облаке. Сетевые интерфейсы могут быть представлены не физическим коммутатором, а виртуальными сетями, которые существуют на одном компьютере или охватывают несколько гипервизоров.

Сегодня каждый программный аспект облачных вычислений может быть реализован с помощью технологии с открытым исходным кодом, работающей на системах Linux. Чтобы получить представление о том, как действуют некоторые из основных технологий облачных вычислений, поговорим о них в этой главе, а затем будет описано, как настроить гипервизор и начать применять на нем виртуальные машины.

Гипервизоры (вычислительные узлы)

В облачных вычислениях операционные системы, обслуживающие облачных пользователей, не работают непосредственно на компьютерном оборудовании. Вместо этого гипервизоры настраиваются для запуска многих операционных систем в качестве так называемых виртуальных машин.

В зависимости от вашей облачной среды гипервизор может называться вычислительным узлом, рабочим узлом или просто хостом. Поскольку гипервизоры, как правило, являются отдельными элементами (десятки или сотни гипервизоров

могут быть настроены для определенного местоположения), система Linux — это отличный выбор операционной системы, работающей так же, как гипервизоры, непосредственно на оборудовании.

Виртуальная машина *Kernel-based Virtual Machine (KVM)* — это базовая технология виртуализации, реализованная в большинстве дистрибутивов Linux для преобразования системы Linux в гипервизор. KVM поддерживается в Ubuntu, Red Hat Enterprise Linux, Fedora, CentOS и многих других системах.

Другая важная технология, которая может быть использована вместо KVM для превращения системы Linux в гипервизор, — это проект Xen (www.xenproject.org). Xen существует дольше, чем KVM, и поддерживается в системах Citrix Systems и Oracle.

Далее в этой главе я расскажу, как проверить, есть ли у компьютера необходимые аппаратные возможности для применения его в качестве гипервизора, и как настроить его для использования с машиной KVM.

Облачные контроллеры

Поскольку настройка облака может включать в себя несколько гипервизоров, пулов хранилищ, виртуальных сетей и множество виртуальных машин, необходимы централизованные инструменты для управления этими функциями и их мониторинга. Для управления облачными средами можно использовать как графические инструменты, так и инструменты командной строки.

Графический интерфейс Virtual Machine Manager (`virt manager`), хотя он и не считается полноценным облачным контроллером, и команда `virsh` могут использоваться для управления небольшой облачной средой. Применяя `virt-manager`, вы можете получить представление об управлении несколькими виртуальными машинами через несколько гипервизоров, а также научиться работать с виртуальными сетями и общими пулами хранения.

Полномасштабные облачные платформы имеют собственные контроллеры для обеспечения гораздо более сложных взаимодействий между облачными компонентами. Например, платформа Red Hat OpenStack (access.redhat.com/products/red-hat-openstack-platform) и ее проект RDO (www.rdoproject.org) предоставляют гибкие расширяемые облачные среды для управления виртуальными машинами и всеми связанными с ними вспомогательными функциями. Для Red Hat Virtualization (RHV) менеджер RHV предоставляет многие из тех же функций. Однако, если вы хотите начать с более простого варианта управления своей первой мини-облачной средой, начните с использования команды `virt-manager` и инструмента VM Desktop.

Облачное хранилище

Новые требования к хранению данных возникают при перемещении операционных систем и приложений в облачную среду. Чтобы виртуальная машина могла работать на другом гипервизоре, ее хранилище должно быть доступно с нового

гипервизора. Облачное хранилище включает в себя внутреннее хранилище для виртуальных машин, образы для запуска VM и базы данных для хранения информации о самом облаке.

Совместное хранилище для нескольких гипервизоров сделать так же просто, как и создать общий ресурс NFS (см. главу 20 «Настройка NFS-сервера») и установить его в одной точке монтирования между несколькими гипервизорами. NFS — это один из самых простых способов реализации общего хранилища.

Более надежное общее хранилище, способное обрабатывать сбои дисков и обеспечивать более высокую производительность, лучше работает для облаков, предоставляющих критически важные службы. Совместное блочное хранилище, где вы монтируете целый диск или раздел диска, может быть выполнено с помощью таких технологий, как iSCSI или Fibre Channel.

Ceph (ceph.com) — это проект с открытым исходным кодом для управления как блочным, так и объектным хранилищем, который часто используется для управления хранилищем в облачных средах. GlusterFS (www.gluster.org) — это масштабируемая файловая система, часто применяемая в облачных средах.

В простом примере мини-облака в этой главе с целью предоставления общего хранилища для гипервизоров я взял NFS. Ceph и GlusterFS лучше подходят для установки на предприятиях.

Аутентификация в облаке

Чтобы иметь возможность ограничить объем облачных ресурсов, потребляемых пользователем, и, возможно, отслеживать и взимать плату за них, вам нужны механизмы аутентификации. Аутентификация необходима как для тех, кто применяет облачные функции, так и для тех, кому разрешено их администрировать.

Проекты облачных платформ иногда позволяют подключать централизованные механизмы аутентификации для проверки и авторизации облачных пользователей. К ним можно отнести Kerberos, Microsoft Active Directory и др. В Linux программное обеспечение Identity, Policy и Audit (IPA) (см. www.freeipa.org) предлагает полный набор функций аутентификации, которые могут работать на корпоративной облачной платформе.

Развертывание и настройка облака

Если вы управляете большой облачной инфраструктурой, вам не нужно подходить к каждой машине и использовать установку через графический интерфейс каждый раз, когда нужно добавить в свою сеть гипервизор или другой узел. Сегодня многие инструменты могут развертывать и настраивать системы Linux так же просто, как перезагружать компьютер и загружать его до предварительно настроенного инсталлятора.

В главе 9 «Установка Linux» я говорил о том, как применять PXE-сервер (для автоматической загрузки инсталлятора Linux по сети со своей карты сетевого

интерфейса) и файлы Kickstart (для определения всех ответов, необходимых для завершения установки). С этой настройкой вы можете просто загрузить компьютер с сетевого интерфейса и автоматически установить систему Linux.

После развертывания компьютера системы могут быть сконфигурированы и, возможно, отслежены и обновлены с помощью таких инструментов, как Puppet (puppetlabs.com) и Chef (www.chef.io). Целые рабочие среды могут быть развернуты в виртуальных машинах с помощью инструмента Vagrant (www.vag rantup.com). Ansible (www.ansible.com) — это еще один инструмент для автоматизации ИТ-инфраструктур и приложений, которые на них работают.

Облачные платформы

Если вы хотите реализовать в организации собственное частное облако, платформа OpenStack с открытым исходным кодом поможет вам в этом. Ее настройки гибкие и мощные.

Red Hat Virtualization (RHV) — еще одна популярная облачная платформа. Она позволяет начать с простого менеджера RHV и одного или двух гипервизоров для запуска виртуальных машин, которыми она управляет, а затем развить облачную платформу, добавив больше гипервизоров, пулов хранения и других функций.

Если вы хотите использовать общедоступные облака, основанные на технологии с открытым исходным кодом, для запуска необходимых операционных систем, задействуйте любой из облачных провайдеров. Поставщики общедоступных облаков, которые можно применять для запуска виртуальных машин Linux, — это Amazon Web Services (www.amazon.com/aws), Google Cloud Platform (cloud.google.com) и Rackspace (www.rackspace.com). Глава 28 «Развертывание приложений Linux в облаке» описывает, как использовать систему Linux в некоторых из них.

Базовые облачные технологии

Чтобы вы могли разобраться в основах облачных технологий, в этом разделе показаны некоторые из основных строительных блоков современной облачной инфраструктуры. Взяв три компьютера, я помогу вам создать установку, которая включает в себя следующее.

- **Гипервизоры.** *Гипервизор* — это программный компонент, который позволяет запускать на нем несколько других компьютерных систем. Эти системы называются виртуальными машинами (VM). В облачной инфраструктуре могут работать десятки или сотни гипервизоров и тысячи виртуальных машин.
- **Виртуальные машины.** Виртуальные машины, которые вы запускаете на гипервизоре Linux, могут быть одним и тем же типом системы Linux, иной системой Linux, системой Windows или любым другим типом системы, совместимым с оборудованием, на котором работает гипервизор. Таким образом, виртуаль-

ные машины, работающие на гипервизорах, которые мы здесь построим, могут включать системы Fedora, Ubuntu, RHEL, CentOS, Microsoft Windows и др.

- **Общее хранилище.** Чтобы обеспечить максимальную гибкость, хранилище, которое гипервизоры предоставляют виртуальным машинам, часто используется пулом гипервизоров совместно. Это позволяет им применять общий набор образов для установки или запуска виртуальных машин, а также дает возможность одним и тем же виртуальным машинам работать на любом гипервизоре из группы и даже перемещаться на другой гипервизор, не выключаясь. Перемещение запущенных виртуальных машин полезно, когда гипервизор перегружается или если его необходимо выключить для обслуживания.

Вот что включает в себя конфигурация, которая позволяет работать с виртуальными машинами:

- установка новой виртуальной машины на гипервизор;
- настройка функций на виртуальных машинах;
- вход в виртуальную машину, работающую на гипервизоре, и ее использование;
- перемещение запущенных виртуальных машин на другой гипервизор.

Мы рассмотрим следующие технологии.

- **Kernel-based Virtual Machine (KVM).** KVM — это базовая технология ядра, которая позволяет виртуальным машинам взаимодействовать с ядром Linux.
- **QEMU Processor Emulator.** Для каждой активной виртуальной машины в системе выполняется один процесс `qemu`. QEMU предоставляет функции, которые заставляют каждую VM вести себя так, как будто она работает на физическом оборудовании.
- **Libvirt Service Daemon (libvirtd).** На каждом гипервизоре работает одна служба `libvirtd`. Демон `libvirtd` прослушивает запросы на запуск, остановку, паузу и иные действия по управлению виртуальными машинами на гипервизоре. Эти запросы могут поступать из приложения, предназначенного для управления VM (например, `virt-manager` или OpenStack Dashboard), или из приложения, созданного для непосредственного взаимодействия с интерфейсом прикладного программирования `libvirt`.
- **Virtual Machine Manager.** Диспетчер виртуальных машин (команда `virt-manager`) — это графический инструмент для управления VM. Помимо того что `virt-manager` позволяет запрашивать запуск и остановку виртуальных машин, он обеспечивает установку и настройку виртуальных машин, а также управление ими различными способами. Вы можете задействовать команду `virsh` для передачи параметров в командную строку для работы с виртуальными машинами вместо использования окна графического интерфейса.
- **Virtualization Viewer.** Команда `virt-viewer` запускает на рабочем столе окно консоли виртуальной машины. Это окно позволяет работать из окна консоли, с рабочего стола или из интерфейса командной строки, с выбранной

виртуальной машины (в зависимости от того, что может предложить VM). Это означает, что кто-то потребляющий ваши вычисления PaaS («платформа как услуга») может объединить собственную операционную систему, приложение, файлы конфигурации и данные и развернуть их. Они будут полагаться на ваши PaaS для обеспечения вычислительной мощности, хранения, памяти, сетевых интерфейсов и функций управления, необходимых для запуска виртуальных машин, содержащих их приложения.

В следующем разделе вы познакомитесь с некоторыми основополагающими технологиями облаков Linux. В нем описывается, как создать небольшое облако, настроив собственные гипервизоры, виртуальные машины и виртуальное хранилище.

Создание небольшого облака

С помощью трех физических машин, объединенных в сеть, можно проиллюстрировать некоторые основные понятия, необходимые для понимания того, как построить собственное облако. Три компьютера под управлением системы Fedora 30 и соединяющая их сеть настроены следующим образом.

- **Сеть.** Для соединения трех компьютеров создается высокоскоростная проводная сеть. Быстрые сетевые подключения имеют решающее значение для успешной миграции виртуальных машин. В этом примере каждый гипервизор имеет также сетевой мост, настроенный таким образом, чтобы каждая виртуальная машина могла получать IP-адрес непосредственно из службы DHCP в сети.
- **Гипервизоры.** Два компьютера настроены как гипервизоры. *Гипервизор* (иногда называемый *хост-вычислительным узлом*) позволяет запускать виртуальные машины. В системе Fedora 30 базовая технология гипервизора называется *Kernel-based Virtual Machine (KVM)*, в то время как фактическими виртуальными машинами управляет служба `ibvirtd`.
- **Хранилище.** Один компьютер настроен как общее хранилище для двух гипервизоров. Для создания общего хранилища используется сервер NFS, так проще, хотя в производственной среде лучше выбирать технологии iSCSI или Fibre Channel.

ПРИМЕЧАНИЕ

В тестовых целях вы можете сделать так, чтобы один из двух гипервизоров выполнял функции общего хранилища. Однако одна из основных целей настройки двух гипервизоров и отдельного общего хранилища заключается в том, что у вас есть возможность отключить любой гипервизор и при этом все виртуальные машины будут работать. Если же общее хранилище реализовано на одном из гипервизоров, нельзя будет его отключать, не выключив все использующие его виртуальные машины.

Настройка гипервизоров

В следующем примере я установил Fedora 30 на два физических компьютера и настроил их как KVM-хосты, на которых работает служба `libvirtd`. Выполните следующие действия, чтобы сделать то же самое у себя.

Шаг 1. Загрузить программное обеспечение Linux

Перейдите на страницу Get Fedora (getfedora.org) и скачайте Fedora 30. Я решил загрузить дистрибутив Fedora 30, 64-битную версию Workstation через образ на DVD-носителе. Если доступна более поздняя версия Fedora, можете взять ее.

Используйте любое доступное приложение для записи DVD, чтобы записать образ на DVD или иным образом сделать его доступным для установки (например, при загрузке PXE).

Шаг 2. Проверить компьютеры

Компьютеры, которые вы используете в качестве гипервизоров в системе Fedora 30, должны соответствовать нескольким требованиям. Перед началом установки проверьте на своем компьютере:

- *поддержку виртуализации* — это можно сделать, посмотрев на флаги, установленные в процессоре;
- *память* — компьютер должен иметь достаточно оперативной памяти не только для запуска основной операционной системы, но и для каждой виртуальной машины, которая запускается в системе;
- *мощность процессора* — имейте в виду, что каждая виртуальная машина потребляет вычислительную мощность для себя и любого приложения, работающего внутри нее.

Необходимо принять во внимание также хранилище и его объем. Но, поскольку мы намерены настроить хранилище с отдельного узла в сети, рассмотрим эту тему позднее.

Чтобы убедиться, что доступные функции ваших компьютеров соответствуют требованиям, загрузите Linux Live с носителя — CD или DVD, откройте окно терминала и введите следующую команду:

```
# cat /proc/cpuinfo | grep --color -E "vmx|svm|lm"

Flags  : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca
cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe
syscall nx pdpe1gb rdtscp lm constant_tsc arch_perfmon pebs bts
rep_good xtopology nonstop_tsc aperfmperf pni pclmulqdq dtes64
monitor ds_cpl vmx smx es...
...
```

Выполнение команды показывает, что этот компьютер 64-разрядный (`lm`) и чип Intel поддерживает функции виртуализации (`vmx`). Если бы процессор был чипом AMD, поддерживающим виртуализацию, а не `vmx`, вы бы увидели пункт `svm`. Эти настройки говорят о том, что компьютер можно использовать в качестве гипервизора.

В начале запуска виртуальных машин на хосте памяти часто может не хватать. Нужно добавить необходимое количество памяти для хоста и для каждой виртуальной машины.

Вы можете снизить требования к памяти, не устанавливая настольное программное обеспечение, как делает большинство гипервизоров. Однако я в примере установил рабочую станцию Fedora, которая поставляется вместе с рабочим столом. Чтобы проверить память и раздел подкачки на компьютере, я ввел следующее:

```
# free -m
              total        used        free      shared buff/cache   available
Mem:          15318        4182        6331        1047        4805        9678
Swap:          7743           0        7743
```

Эта система имеет около 16 Гбайт оперативной памяти и 8 Гбайт подкачки. Я считаю, что 4 Гбайт хватит для настольной системы.

Если я задам 1 или 2 Гбайт для каждой виртуальной машины, эта система сможет запускать 6–12 виртуальных машин и рабочий стол. Проверьте требования к памяти для операционных систем и приложений, которые планируете запускать, чтобы определить конкретные потребности в памяти для вашей системы.

Чтобы проверить количество и типы процессоров на своем компьютере, введите следующие данные:

```
# grep processor /proc/cpuinfo

processor : 0
...
processor : 6
processor : 7
# head /proc/cpuinfo
processor  : 0
vendor_id  : GenuineIntel
cpu family : 6
model      : 60
model name : Intel(R) Core(TM) i7-4800MQ CPU @ 2.70GHz
stepping   : 3
cpu MHz    : 2701.000
cache size : 6144 KB
...
```

Первая команда в коде показывает, что на компьютере имеется восемь (от 0 до 7) процессоров. Вторая команда для первого процессора показывает, что это GenuineIntel, и дает его номер модели, ее название, скорость процессора и другую информацию.

Чтобы переносить виртуальные машины в реальном времени между двумя гипервизорами, процессоры должны находиться в одном семействе. Если у них нет

совместимых процессоров, можете перенести виртуальную машину, выключив ее на одном гипервизоре и запустив из общего хранилища на другом.

После того как вы определили размеры двух гипервизоров, начните устанавливать на них систему Fedora.

Шаг 3. Загрузить Linux на гипервизоры

Используя установочный носитель Fedora 30 Workstation, начните установку двух гипервизоров. Придерживайтесь процедуры установки системы Fedora, приведенной в главе 9 «Установка Linux». Необходимо знать о настройках, специфичных для нее.

- **Имена гипервизоров.** Я установил имена хостов на гипервизорах в host1.example.com и host2.example.com.
- **Разбиение дисков.** При разбиении на разделы я стираю весь жесткий диск целиком. Затем создаю раздел `/boot` размером 500 Мбайт и раздел подкачки 12 Гбайт, а остальную часть дискового пространства назначаю корневому разделу (`/`). Каталог `/var/lib/libvirt/images` содержит большую часть данных этой системы, но это общий каталог, доступный из другой системы в сети и совместно используемый двумя гипервизорами. (Об этом позже.)
- **Сеть.** Если есть такая возможность, включите проводные сетевые интерфейсы для каждого гипервизора. Гипервизоры и хранилище должны находиться в одной локальной сети, поскольку скорость сетевого соединения между этими машинами имеет решающее значение для достижения хорошей производительности.
- **Пакеты программного обеспечения.** Я устанавливаю только стандартные пакеты Fedora Workstation. После завершения установки и перезагрузки системы устанавливаю дополнительное программное обеспечение, необходимое для каждого гипервизора.

После завершения установки перезагрузите компьютер (извлеките DVD и запустите на жестком диске). После перезагрузки системы обновите программное обеспечение Fedora, добавьте новые пакеты и снова перезагрузите систему следующим образом:

```
# yum update -y
# yum install virt-manager libvirt-daemon-config-network
# reboot
```

Пакет `virt-manager` содержит графический инструмент для управления виртуальными машинами. Пакет `libvirt-daemon-config-network` создает сетевой интерфейс по умолчанию, который позволяет виртуальным машинам получать доступ к внешним сетям (через хост) с помощью преобразования сетевых адресов (NAT). Диапазон адресов по умолчанию, назначенный виртуальным машинам, — от 192.168.122.2 до 192.168.122.254.

Другие пакеты, которые вам понадобятся, уже должны быть включены в установку Fedora Workstation. Если вы выполнили другой тип установки, убедитесь, что добавлены следующие пакеты:

- `libvirt-client` (для команды `virsh`);
- `libvirt-daemon` (для службы `libvirtd`).

Шаг 4. Запустить службы на гипервизорах

Вам нужно убедиться, что служба `libvirtd` работает на обоих гипервизорах. Запустите также службу `sshd`. Возможно, они уже запущены, но, чтобы убедиться в этом, выполните от имени суперпользователя на обоих гипервизорах следующие действия:

```
# systemctl start sshd.service
# systemctl enable sshd.service
# systemctl start libvirtd.service
# systemctl enable libvirtd.service
```

Служба `sshd` позволяет при необходимости входить в гипервизоры по сети. Возможно, вы еще не знакомы со службой `libvirtd`. Она прослушивает запросы на управление вашими виртуальными машинами на каждом хосте.

Шаг 5. Настроить каталог /etc/hosts или сервер DNS

Чтобы было удобно взаимодействовать между гипервизорами и системой хранения данных, вы должны назначить каждой системе имена хостов и сопоставить их с IP-адресами. Настройка DNS-сервера, на который указывают все системы, — вероятно, лучший вариант для этого. Однако в нашем простом примере можете просто отредактировать файл `/etc/hosts` во всех системах и добавить записи для каждого хоста.

Вот пример того, как могут выглядеть дополнительные записи в файле `/etc/hosts` для трех систем:

```
192.168.0.138  host1.example.com host1
192.168.0.139  host2.example.com host2
192.168.0.1    storage.example.com storage
```

Далее необходимо настроить хранилище.

Настройка хранилища

Вы можете предоставить сетевое хранилище гипервизорам для этой процедуры многими способами. Я решил создать отдельную систему Fedora в той же локальной сети, что и гипервизоры, и применить NFS для подключения общего хранилища к обоим гипервизорам.

Сервер NFS — не самый эффективный метод совместного использования хранилища гипервизорами, но один из самых простых и распространенных. Далее я покажу, как применять графический инструмент Virtualization Manager GUI (`virt-manager`) для настройки пула хранения NFS.

Для обеспечения согласованности задействуется общий ресурс NFS, настроенный из системы хранения, — каталог `/var/lib/ libvirt/images`. Он монтируется в одном и том же месте на каждом из гипервизоров. (Если у вас только две машины, можете настроить хранилище с одного из гипервизоров и протестировать его. Однако это означает, что вы не сможете отключить данный гипервизор, не выключив все виртуальные машины.)

Шаг 1. Загрузить программное обеспечение Linux

Чтобы настроить хранилище на сервере NFS, можно использовать практически любую систему Linux, имеющую доступ к службе NFS. О чем нужно подумать перед установкой Linux.

- **Пространство на диске.** Убедитесь, что в разделе, содержащем общий каталог, достаточно места для хранения. В этом примере общий каталог — это `/var/lib/libvirt/images`.
- **Производительность.** Для достижения большей производительности необходим диск с быстрым доступом и высокой скоростью передачи данных.

Для дистрибутивов Fedora и RHEL программное обеспечение сервера NFS доступно из пакета `nfs-utils`. Для дистрибутива Ubuntu необходим пакет `nfs-kernel-server`. После завершения первоначальной установки убедитесь, что программное обеспечение сервера NFS установлено. Если это не так, можете установить его на Fedora или RHEL с помощью команды:

```
# yum install nfs-utils
```

Для установки на Ubuntu и похожих системах, введите:

```
# apt-get install nfs-kernel-server
```

Шаг 2. Настроить общий ресурс NFS

Чтобы создать общий ресурс NFS, необходимо определить каталог для общего доступа и добавить информацию о нем в файл `/etc/exports`. Выполните следующие действия.

1. **Создайте каталог.** Вы можете поделиться любым каталогом, содержащим свободное пространство. Подумайте о создании нового каталога и установке на него целого диска или раздела. Для этого примера я создаю каталог `/var/storage` следующим образом:

   ```
   # mkdir -p /var/storage
   ```

2. **Разрешите экспорт.** В своей системе хранения создайте запись в файле `/etc/exports`, чтобы поделиться каталогом с выбранными системами (по имени или IP-адресу). В этом примере я разрешил доступ на чтение и запись (`rw`) ко всем системам подсети 192.168.0:

```
/var/storage 192.168.0.*(no_root_squash,rw,sync)
```

Шаг 3. Запустить службу NFS

Запустите службу NFS и откройте брандмауэр в системе хранения, чтобы разрешить доступ к этой службе, следующим образом.

1. **Запустите и включите службу NFS.** В последних версиях систем Fedora и RHEL для запуска сервера NFS введите следующие команды:

```
# systemctl start nfs-server.service
# systemctl enable nfs-server.service
```

   В RHEL 6, старой версии Fedora и некоторых системах Ubuntu для запуска и включения службы NFS используйте команды:

```
# service nfs start
# chkconfig nfs on
```

2. **Откройте брандмауэр.** Чтобы открыть порты брандмауэра, позволив находящимся за пределами локальной системы задействовать ваш общий ресурс NFS, в системе Fedora 30 выполните следующие действия:

```
# firewall-cmd --permanent --add-service=rpc-bind
# firewall-cmd --permanent --add-service=nfs
# systemctl restart firewalld
```

Чтобы получить информацию о том, как открыть брандмауэр для службы NFS для систем, использующих службу `iptables` напрямую, см. главу 20.

Шаг 4. Смонтировать общий ресурс NFS на гипервизоры

Войдите в каждый гипервизор и выполните следующие действия, чтобы сделать общий ресурс доступным локально. Обратите внимание на то, что расположение каталога точек монтирования на всех гипервизорах должно быть одинаковым.

1. **Проверьте доступность общего ресурса NFS.** На каждом из двух гипервизоров убедитесь, что можете видеть доступную общую папку, введя следующее:

```
# showmount -e storage.example.com
Export list for storage.example.com:

/var/storage 192.168.0.*
```

2. **Смонтируйте общий ресурс NFS.** Добавьте информацию об общем ресурсе в файл `/etc/fstab`. В нашем примере, чтобы позволить каталогу из систе-

мы 192.168.0.1 монтироваться локально в один и тот же каталог при каждой загрузке системы, запись в файле `/etc/fstab` может выглядеть следующим образом:

```
storage.example.com:/storage /var/lib/libvirt/images nfs defaults 0 0
```

3. **Установите логические типы SELInux.** Если SELinux находится в принудительном режиме, установите следующий логический тип, чтобы разрешить команде `qemu-kvm` использовать общий ресурс NFS:

```
# setsebool -P virt_use_nfs 1
```

4. **Протестируйте монтирование службы NFS.** Чтобы проверить правильность монтирования, выполните следующую команду для монтирования в файле `/etc/fstab` всех еще не смонтированных записей и убедитесь, что общий ресурс NFS смонтирован:

```
# mount -a
# mount | grep libvirt
storage.example.com:/var/storage on /var/lib/libvirt/images type nfs4
(rw,relatime,vers=4.0,rsize=1048576,wsize=1048576,namlen=255,hard,proto=tcp,
port=0,timeo=600,retrans=2,sec=sys,clientaddr=192.168.0.1,local_lock=none,
addr=192.168.0.138)
```

Теперь, когда гипервизоры и хранилище установлены, можно приступать к созданию виртуальных машин.

Создание виртуальных машин

Программа Virtual Machine Manager (`virt-manager`) — это хороший инструмент для создания простейших виртуальных машин. Она поможет выполнить установку и настройку виртуальных машин, а также позволит просматривать и изменять состояние существующих VM.

Позже, когда вы поймете, какие функции используются при создании виртуальных машин, сможете применять команду `virt-install`. Ее преимущество заключается в том, что вы можете написать скрипт или легко скопировать и вставить командную строку для создания виртуальной машины вместо того, чтобы использовать окно графического интерфейса.

Ранее в этой главе мы загрузили ISO-образ Fedora 30 Workstation, поэтому я задействую его в примере создания виртуальной машины. Однако при желании вы можете установить в качестве своей VM множество версий Linux или Windows.

Шаг 1. Загрузить образ для создания виртуальных машин

Виртуальную машину можно создать разными способами. Как правило, вы либо начинаете с готового образа (в основном копии рабочей виртуальной машины), либо устанавливаете ее из установочного ISO-образа в новое хранилище. Мы возьмем

за основу второй вариант и создадим виртуальную машину из установочного ISO-образа Fedora 30 Workstation.

Предположим, что вы вошли в один из гипервизоров как суперпользователь и нашли ISO в текущем каталоге. Скопируйте образ ISO в каталог по умолчанию, применяемый командой `virt-manager` для хранения (`/var/lib/libvirt/images`):

```
# cp Fedora-Workstation-Live-x86_64-30-1.2.iso /var/lib/libvirt/images/
```

Поскольку этот каталог общий для обоих гипервизоров, можете перейти к любому из них, чтобы использовать этот образ.

Шаг 2. Проверить сетевой мост

На каждом гипервизоре должен быть сетевой мост по умолчанию с именем `virbr0`. Все виртуальные машины будут добавлены в этот сетевой интерфейс и автоматически присвоены IP-адресу. Данный мост по умолчанию существует благодаря виртуальной сети `libvirtd`. Гипервизор использует диапазон адресов от 192.168.122.2 до 192.168.122.254 для назначения виртуальным машинам. Преобразуя сетевые адреса (NAT), хост может маршрутизировать пакеты от виртуальных машин, применяющих эти частные адреса, к внешним сетевым интерфейсам.

Выполните на каждом гипервизоре следующие действия, чтобы проверить мосты:

```
# brctl show virbr0

bridge name  bridge id            STP enabled  interfaces
virbr0       8000.001aa0d7483e    yes          vnet0
# ip addr show virbr0

5: virbr0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue
        state UP group default
    link/ether fe:54:00:57:71:67 brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.1 brd 192.168.122.255 scope global
dynamic virbr0
```

Шаг 3. Запустить программу Virtual Machine Manager (virt-manager)

Чтобы открыть программу Virtual Machine Manager и подключить ее к гипервизору, на рабочем столе любого из гипервизоров выполните следующие действия.

1. **Запустите команду virt-manager.** Перейдите на экран Acivities (Приложения), введите `Virtual Machine Manager` в поле поиска и нажмите клавишу Enter или введите команду `virt-manager` из командной консоли. При появлении запроса введите пароль суперпользователя. Появится программа Virtual Machine Manager.

2. **Проверьте подключение к гипервизору.** Во всплывающем окне Add Connection (Добавить соединение) уже должны быть установлены гипервизор (QEMU/KVM) и флажок Autoconnect (Автосоединение). Нажмите кнопку Connect (Подключиться), чтобы подключиться к локальному гипервизору, если это не произошло автоматически.

Шаг 4. Проверить соединение

После подключения к гипервизору настройте детали подключения. Для этого в окне Virtual Machine Manager выполните следующие действия.

1. **Просмотрите детали подключения.** Выберите вкладку Edit ▸ Connection Details (Изменить ▸ Сведения о соединении), чтобы увидеть окно Connection Details (Сведения о соединении). Выберите вкладки Overview (Обзор), Virtual Networks (Виртуальные сети), Storage (Хранилище) и Network Interfaces (Сетевые интерфейсы), чтобы ознакомиться с информацией о соединении для вашего гипервизора. Например, на вкладке Storage (Хранилище) (рис. 27.1) видно, что в распоряжении гипервизора имеется 438,40 Гбайт свободного места (каталог `/var/lib/libvirt/images`).

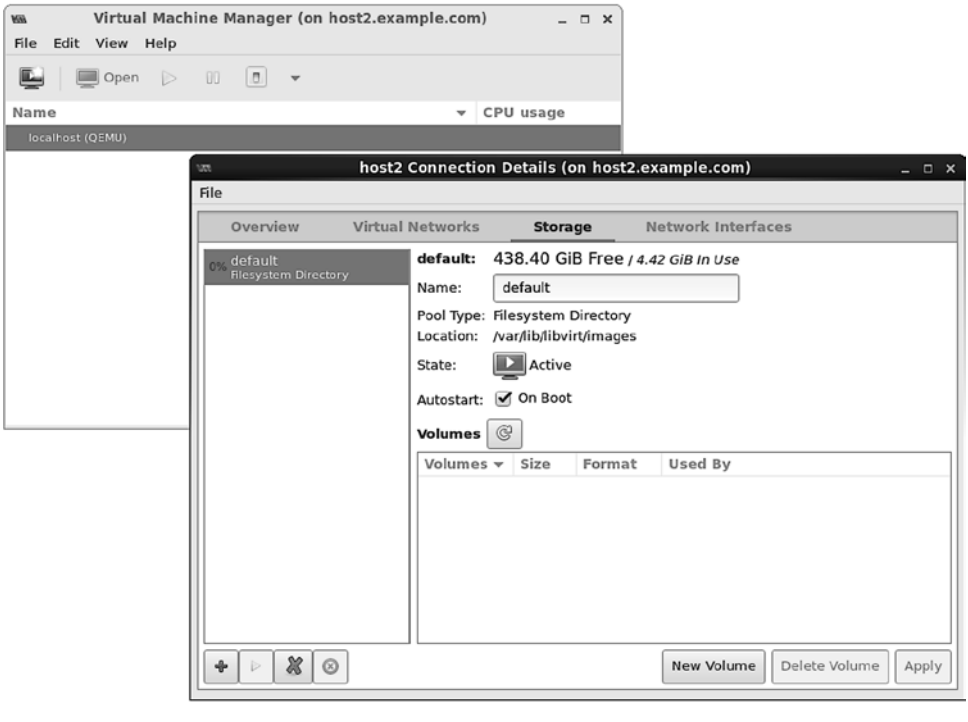


Рис. 27.1. Запустите программу Virtual Machine Manager и проверьте сведения о подключении

2. **Убедитесь, что сетевой мост доступен.** Выберите вкладку Virtual Networks (Виртуальные сети) и убедитесь, что мост (`virbr0`) есть в списке доступных сетевых интерфейсов.

Шаг 5. Создать новую виртуальную машину

Чтобы создать новую виртуальную машину в окне Virtual Machine Manager, выполните следующие действия.

1. **Запустите мастер установки.** Чтобы открыть окно Create a New Virtual Machine Wizard, выберите вкладку File ▸ New Virtual Machine (Файл ▸ Новая виртуальная машина). Появится окно Create a New Virtual Machine (Создать новую виртуальную машину).
2. **Выберите способ установки.** Есть четыре способа создания виртуальной машины. Первые три — это способы определения местоположения установочного носителя. Четвертый позволяет импортировать существующий образ диска. Здесь выберите первый вариант, Local install media (Локальный установочный носитель), и нажмите кнопку Forward (Далее).
3. **Выберите образ ISO.** Нажмите кнопку Use ISO Image (Использовать ISO-образ) и выберите пункт Browse (Обзор). В появившемся окне перейдите к Fedora 3021 Workstation ISO, выберите пункт Choose Volume (Выбрать том) и нажмите кнопку Forward (Далее), чтобы продолжить.
4. **Выберите объем памяти и процессор.** Выберите объем оперативной памяти и количество процессоров, доступных для виртуальной машины, и нажмите кнопку Forward (Далее). Я советую взять по крайней мере 1024 Мбайт оперативной памяти и по крайней мере один процессор. Если доступно 2048 Мбайт оперативной памяти, задействуйте этот объем.
5. **Подключите хранилище.** Выберите объем дискового пространства, который должна использовать виртуальная машина. Я советую выбрать 10 Гбайт для рабочей станции Fedora, но вам, вероятно, хватит и меньшего объема. Созданный образ `qcow2` вырастет до того размера, который вы фактически применяете (вплоть до выделенной суммы), поэтому чрезмерное пространство не вызывает никаких проблем, пока вы не попытаетесь использовать его. Установите режим кэширования `none` или `directsync`, чтобы иметь возможность перенести виртуальную машину позже. Нажмите кнопку Forward (Далее).
6. **Проверьте все настройки перед началом установки.** Выберите имя виртуальной машины и просмотрите другие параметры установки. Выберите пункт Customize Configuration Before Install (Настроить конфигурацию перед установкой), чтобы еще раз просмотреть настройки. Оставьте настройки по умолчанию и нажмите кнопку Finish (Готово).
7. **Проверьте настройки компьютера.** Если на предыдущем экране выбрали пункт Customize (Настроить), можете подробно изучить настройки. Убедитесь, что

установлен режим кэша none или directsync. Если вас все устраивает, нажмите кнопку Begin Installation (Начать установку).

8. **Установите виртуальную машину.** Вы можете установить систему точно так же, как если бы делали это непосредственно на компьютере. Завершите установку и перезагрузите виртуальную машину. Если окно VM не открыто, дважды щелкните на ее имени (в данном случае fedora1) в окне virt-manager и войдите в систему. На рис. 27.2 показан пример окна virt-manager с виртуальной машиной Fedora Workstation.

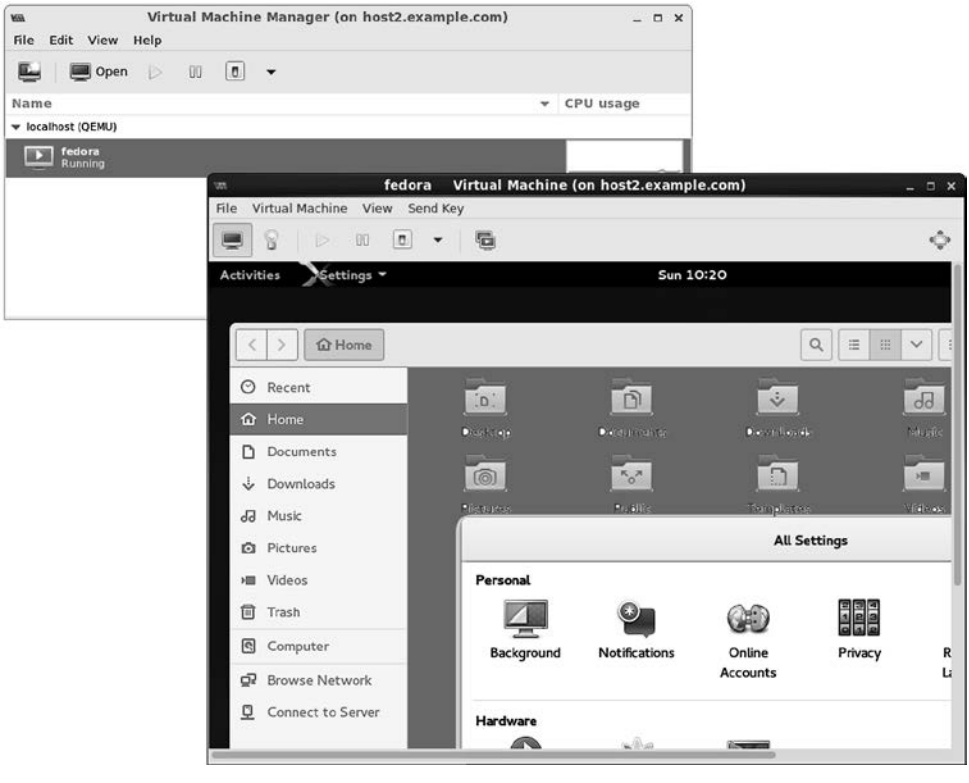


Рис. 27.2. Откройте виртуальную машину и начните работать с ней

Управление виртуальными машинами

После установки одной или нескольких виртуальных машин на гипервизор можно управлять ими почти так же, как системой, установленной на физическом компьютере. Можно делать следующее.

- **Просматривать систему из консоли.** Дважды щелкните на запущенной виртуальной машине в окне virt-manager. Откроется окно консоли VM, позволяющее

использовать ее так же, как и физическую консоль, для доступа к операционной системе компьютера. Вместо `virt-manager` для отображения консоли виртуальной машины можно задействовать команду `virt-viewer`. Например, для виртуальной машины с именем `rhel8-01` введите `virt-viewer rhel8-01`.

- **Выключить виртуальную машину.** Щелкните правой кнопкой мыши на имени виртуальной машины и выберите пункт Shut Down (Завершить работу). Затем выберите либо Shut Down (Завершить), чтобы корректно выключить ее, либо Force Off (Принудительно завершить), что по эффективности похоже на выдергивание вилки из розетки. Есть и вариант Reboot (Перезагрузка).
- **Запустить виртуальную машину.** Если виртуальная машина в данный момент не работает, щелкните правой кнопкой мыши на записи о ней и выберите Run (Запустить), чтобы запустить ее.
- **Удалить виртуальную машину.** Если вы закончили использовать виртуальную машину, выберите пункт Delete (Удалить). Программа спросит, хотите ли вы также удалить хранилище. Снимите этот флажок, если нужно сохранить хранилище, связанное с виртуальной машиной.

Теперь можете попробовать перенести виртуальную машину на другой гипервизор.

Перенос виртуальных машин

Возможность переноса виртуальных машин между различными гипервизорами позволяет очень гибко управлять рабочими нагрузками компьютера. Вот некоторые из преимуществ.

- *Улучшается производительность* за счет переноса виртуальных машин с перегруженных гипервизоров на те, у которых бóльшие объем памяти и емкость процессора.
- Можно выполнять *плановое техническое обслуживание* гипервизора, а виртуальные машины продолжат работать.
- Есть возможность *переместить виртуальные машины с недостаточно активно используемых гипервизоров*, чтобы их можно было отключить для экономии энергии до тех пор, пока они не понадобятся снова.
- Можно *переместить виртуальные машины* с сайта, если планируется закрытие центра обработки данных или ожидается, что по нему ударит ураган или случится другая катастрофа.

Реальный перенос, в частности, полезен, если вам нужно продолжать работу на виртуальных машинах, не отключая их. Ключом к тому, чтобы перенос виртуальных машин в реальном времени сработал, является правильная настройка среды. Убедитесь, что на месте следующие детали соединения (имейте в виду, что эти функции похожи на соответствующие функции Red Hat Virtualization):

- совместное сетевое хранилище между гипервизорами;
- одни и те же сетевые интерфейсы, настроенные на каждом гипервизоре;
- совместимые процессоры между гипервизорами (набор гипервизоров имеет точно такое же оборудование);
- быстрое сетевое соединение между гипервизорами и хранилищем;
- те же или аналогичные версии программного обеспечения виртуализации на гипервизорах (мы использовали на обоих Fedora 30).

Если вся эта подготовка выполнена, сам процесс переноса потребует всего нескольких действий и можно будет начать работать.

Шаг 1. Идентифицировать другие гипервизоры

Предположим, что окно программы Virtual Machine Manager все еще работает на одном из ваших гипервизоров. Перейдите в него и выполните следующие действия, чтобы подключиться к другому гипервизору.

1. **Подключитесь к гипервизору.** Выберите File ▸ Add Connection (Файл ▸ Добавить соединение). Должно появиться окно Add Connection (Добавить соединение).
2. **Добавьте соединение.** Установите флажок Connect to Remote Host (Подключиться к удаленному хосту), выберите SSH в качестве метода подключения, примените имя суперпользователя и введите имя хоста другого гипервизора (например, `host1.example.com`). Когда вы нажмете кнопку Connect (Подключиться), нужно будет ввести пароль суперпользователя удаленного гипервизора и другую информацию. Обратите внимание на то, что может потребоваться установить пакет `openssh-askpass`, чтобы получить запрос на ввод пароля.

Запись о новом гипервизоре должна появиться в окне программы Virtual Machine Manager.

Шаг 2. Перенести запущенную виртуальную машину на другой гипервизор

Прежде чем переносить виртуальную машину на другой гипервизор, может потребоваться изменить правила брандмауэра. При наличии правил брандмауэра по умолчанию прямая миграция `libvirt` завершится неудачно. Случайный TCP-порт должен быть открыт, чтобы разрешить перенос. По умолчанию используется 49 152, но можно выбрать любой доступный незанятый порт. Туннельный перенос требует аутентификации по SSH-ключу.

Открыв программу Virtual Machine Manager, щелкните правой кнопкой мыши на любой запущенной в данный момент виртуальной машине и выберите пункт Migrate (Перенести). Откроется окно Migrate (Перенос) виртуальной машины, как показано на рис. 27.3.

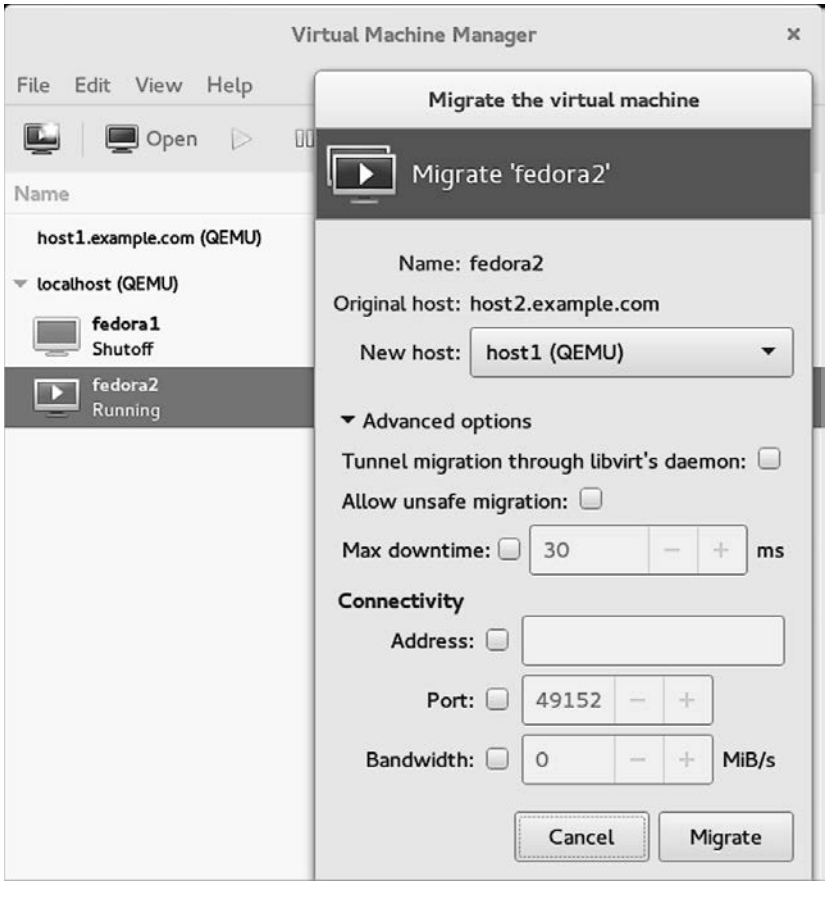


Рис. 27.3. Выберите, на какой гипервизор перенести виртуальную машину

Выберите новый хост. В моем примере виртуальная машина сейчас работает на хосте, поэтому я хочу выбрать `host1` в качестве нового хоста. Через некоторое время виртуальная машина появится на нем и можно будет скопировать образ памяти на другой хост.

Если по какой-то причине перенос завершится неудачно (несовместимые процессоры или другие проблемы), вы всегда можете выключить виртуальную машину на одном хосте и запустить ее снова на другом. Для этого требуется только общее хранилище. На втором хосте просто запустите Create a New Virtual Machine Wizard, но выберите запуск существующего образа вместо установочного ISO.

Настройка гипервизора, которую я продемонстрировал, хорошо подходит для домашней рабочей станции или даже для малого бизнеса. Хотя в эту книгу не входит разработка целой платформы облачных вычислений, полезная информация о том, как использовать различные облачные платформы для запуска систем Linux, представлена в следующей главе.

Резюме

Система Linux — это фундамент, на котором строится большинство современных облачных технологий. В этой главе описываются многие из основных компонентов построения облака на основе Linux и других технологий с открытым исходным кодом. Затем рассказывается о некоторых из этих базовых технологий — об установке пары гипервизоров и запуске виртуальной машины.

Упражнения

Упражнения в этом разделе связаны с настройкой гипервизора (хост-компьютера KVM) и применением его для запуска виртуальных машин. Если затрудняетесь с решением, воспользуйтесь ответами к упражнениям, приведенными в приложении Б (хотя Linux позволяет решать задачи разными способами).

Хотя в рассмотренном в этой главе примере настройки гипервизоров задействуются три физические машины, эти упражнения можно выполнять и на одной машине.

1. Проверьте, может ли ваш компьютер поддерживать виртуализацию KVM.
2. Установите систему Linux вместе с пакетами, необходимыми для использования ее в качестве хоста KVM и запуска приложения Virtual Machine Manager.
3. Убедитесь, что в системе запущены службы `shd` и `libvirtd`.
4. Загрузите установочный ISO-образ Linux, совместимый с вашим гипервизором, и скопируйте его в каталог по умолчанию, используемый программой Virtual Machine Manager для хранения образов.
5. Убедитесь, что сетевой мост по умолчанию (`virbr0`) в данный момент активен.
6. Установите виртуальную машину, применяя ISO-образ, скопированный ранее.
7. Убедитесь, что вы можете войти в виртуальную машину и работать с ней.
8. Убедитесь, что ваша виртуальная машина может подключаться к Интернету или другой сети за пределами гипервизора.
9. Остановите работу виртуальной машины.
10. Запустите виртуальную машину вновь.

28 Развертывание приложений Linux в облаке

В этой главе

- Создание облачных образов Linux.
- Развертывание облачного образа с помощью `virt-manager` (`libvirtd`).
- Развертывание облачного образа в OpenStack.
- Развертывание облачного образа в Amazon EC2.

Для загрузки новой системы Linux вместо того, чтобы просто запускать стандартную программу установки с физического DVD, вы можете получить образ Linux и развернуть его в облаке. Один из способов сделать это — взять общий образ Linux (загрузочный, но ненастроенный) и для его настройки задать параметры в соответствии со своими требованиями. Другой способ — перейти к облачному провайдеру, выбрать образ, щелкнуть на выбранных элементах, чтобы настроить его и запустить.

Дело в том, что облачные вычисления предлагают новые способы запуска и использования систем Linux. В главе 27 мы выполнили стандартную установку Linux для создания виртуальной машины, работающей на гипервизоре Linux. В этой главе я покажу, как задействовать облачные образы для запуска новой системы Linux.

Во-первых, я опишу, как применять службу `cloud-init` для объединения вручную облачного образа Linux с информацией о конфигурации, чтобы позволить ему работать в различных средах. Далее расскажу, как аналогичный процесс выполняется в облаке OpenStack или Amazon Elastic Compute Cloud (EC2) с помощью простых в использовании облачных контроллеров, которые позволяют выбрать образ и настроить его для запуска Linux Cloud.

Запуск Linux в облаке

Облачные платформы отлично подходят для быстрого и эффективного запуска новых виртуальных машин. Это возможно, потому что каждый раз, когда вам нужна операционная система, новая установка не требуется.

Публичные облака, такие как Amazon EC2 (aws.amazon.com/ec2), предлагают варианты различных дистрибутивов Linux. Вы выбираете подходящий вариант Linux, например Ubuntu, Red Hat Enterprise Linux (RHEL) или SUSE Linux Enterprise Server (SLES), и настраиваете его для конкретных целей. Например, существуют варианты систем, оптимизированные для обработки высокопроизводительных приложений или приложений с интенсивным использованием памяти.

Содержимое облачной системы, как правило, носит общий характер. Ожидается, что более детальную информацию для образа предоставит облачный пользователь или поставщик облачных служб, применяющий такую службу, как `cloud-init`. Эта информация делится на две основные категории, `meta-data` и `user-data`.

- `meta-data`. К метаданным `meta-data` прилагается информация, необходимая для загрузки образа. Это данные, которые находятся вне содержимого образа и обычно управляются облачным провайдером. Некоторые из них появляются из-за того, что хранилище, память и вычислительная мощность берутся из пула ресурсов, а не из физической машины, на которой устанавливается система. Таким образом, `meta data` сообщает облачному провайдеру, сколько этих и, возможно, других ресурсов необходимо выделить на ранней стадии запуска системы.
- `user-data`. Информация о пользовательских данных `user-data` добавляется в операционную систему, существующую на образе. Это данные, которые предоставляет пользователь, работающий с виртуальной машиной. Они могут включать учетную запись пользователя и пароль, файлы конфигурации, команды для запуска при первой загрузке, идентификаторы репозиториев программного обеспечения и что-либо еще, что он хочет запустить или изменить в самой операционной системе.

При запуске системы Linux в облачной среде необходимо ввести данные `meta-data` и `user-data`, установив флажки и заполнив формы из веб-облачного контроллера, например панели мониторинга OpenStack или Red Hat Virtualization Manager. Эта информация не может быть идентифицирована как `meta-data` и `user-data` при настройке системы через облачный контроллер.

Облако, которое вы используете для запуска виртуальных машин Linux, может быть публичным, частным или гибридным. Выбор типа облака зависит от ваших потребностей и бюджета.

- **Публичное облако.** Amazon EC2 и Google Compute Engine — это примеры облачных платформ, которые позволяют запускать и использовать виртуальные машины Linux из веб-интерфейса. Вы платите за время работы системы в облаке. Объем памяти, хранилища и виртуальных процессоров, задействуемых для запуска службы, также учитывается в расходах. Преимущество публичных облаков заключается в том, что вам не нужно покупать и поддерживать собственную облачную инфраструктуру.
- **Частное облако.** С помощью частного облака вы создаете собственную вычислительную инфраструктуру (гипервизоры, контроллеры, хранилище, сетевую конфигурацию и т. д.). Создание частного облака означает увеличение первоначальных затрат на владение инфраструктурой и ее обслуживание. Но это

обеспечивает дополнительную безопасность и контроль над вашими вычислительными ресурсами. Поскольку вы управляете инфраструктурой, то можете создавать образы, к которым клиенты имеют доступ в инфраструктуре OpenStack, и самостоятельно учитывать использование ими этой инфраструктуры.

- **Гибридное облако.** Многие компании ищут гибридные облачные решения. Гибридное облако позволяет централизованно управлять несколькими облачными платформами. Например, Red Hat CloudForms может развертывать виртуальные машины и управлять ими на платформах виртуализации OpenStack, VMware vSphere и Red Hat Enterprise, предоставляя различные типы рабочих нагрузок соответствующим средам. В периоды спроса CloudForms также может направлять виртуальные машины для работы в облаках Amazon EC2. Эти облачные среды реализуют различные способы подготовки и настройки виртуальных машин. Однако функции, которые облака должны предоставлять для управления виртуальными машинами, аналогичны. Понимание этих функций может помочь при настройке системы Linux для работы в облаке.

Чтобы вы лучше поняли, как настраивать облачные системы Linux, в следующих разделах я опишу, как работает служба `cloud-init` для настройки облачных систем. Затем помогу вам создать собственные файлы `meta-data` и `user-data` и применить их к облачной системе, чтобы информация могла использоваться при загрузке облачного образа.

Создание образов Linux для облаков

Вспомните, что вы делали, когда устанавливали систему Linux в главе 9 «Установка Linux». В процессе ручной установки вы задаете пароль суперпользователя, создаете обычную учетную запись пользователя и пароль, возможно, настраиваете сетевые интерфейсы и выполняете другие задачи. Эта информация стала постоянной частью операционной системы, оставаясь неизменной при каждой загрузке системы. Когда вы начинаете с предварительно построенного облачного образа в качестве системы Linux, можете задействовать службу `cloud-init`, чтобы подготовить систему Linux к запуску. Служба `cloud-init` (launchpad.net/cloud-init) настраивает общий инстанс виртуальной машины для работы так, как нужно, не требуя установки. В следующем разделе описаны способы применения службы `cloud-init`.

Настройка и запуск облачной службы init cloud

Я покажу на примере, как создавать данные вручную (данные могут быть объединены с загрузочным облачным образом Linux, чтобы при загрузке этот образ был настроен на основе ваших данных). Объединение данных с образом во время выполнения позволяет изменять данные каждый раз перед запуском образа вместо того, чтобы постоянно устанавливать их в образ.

Предлагаю выполнить действия, описанные далее, на одном из гипервизоров, настроенных в главе 27 «Облачные вычисления с помощью системы Linux». Это позволяет не только создавать настраиваемые данные для облачного образа Linux, но и запускать образ как виртуальную машину на этом гипервизоре.

Чтобы добавить данные и запустить существующий облачный образ, требуется иметь сам облачный образ и создать файлы данных и новый образ, объединяющий эти элементы. Процедура должна быть очень простой для загрузки облачного изображения. Позже я расскажу, как добавить дополнительные функции в эти файлы данных. Чтобы настроить и запустить облачный образ, выполните следующие действия.

1. **Создайте файл cloud-init meta-data.** Создайте файл с именем `meta-data` для хранения данных, идентифицирующих информацию об инстансе облака извне. Например, вы можете добавить имя для идентификации инстанса (`instance-id`), имя хоста (`local-hostname`) и другую информацию. Чтобы первая попытка была попроще, я назначаю только два поля (можете дать им любые имена):

   ```
   instance-id: FedoraWS01
   local-hostname: fedora01
   ```

2. **Создайте файл cloud-init user-data.** Создайте файл с именем `user-data` для хранения данных, которые настраиваются внутри операционной системы на самом образе. В качестве простого примера я установил пароль для пользователя по умолчанию (`fedora`) в значение `cloudpass` и убедился, что срок действия пароля `cloud-init` не истечет:

   ```
   #cloud-config
   password: cloudpass
   chpasswd: {expire: False}
   ```

3. **Объедините данные в отдельный образ.** Используя файлы `meta-data` и `user-data` в текущем каталоге, создайте ISO-образ, содержащий эти данные. Позже мы представим этот образ в виде CD с образом Linux, чтобы служба `cloud-init` знала, как настроить образ Linux. (Сначала установите пакеты `genisoimage` и `cloud-init`, если еще этого не сделали. Пакет `cloud-init` не нужен на гипервизоре.)

   ```
   # yum install genisoimage cloud-init
   # genisoimage -output fedora31-data.iso -volid cidata \
         -joliet-long -rock user-data meta-data
   ```

4. **Загрузите базовый образ облака.** Облачные образы для систем Ubuntu, Fedora и RHEL настроены для работы с `cloud-init`. Загрузите официальный облачный образ Fedora (образы для других дистрибутивов описаны далее) и выполните следующие действия:
   - откройте браузер и перейдите на страницу getfedora.org/en/cloud/download;
   - нажмите кнопку Download у образа OpenStack, чтобы загрузить образ `qcow2`, который можно использовать в среде OpenStack. Имя образа — это что-то вроде `Fedora-Cloud-Base-31-1.9.x86_64.qcow2`.

5. **Сделайте снимок образа.** Вероятно, вам придется выполнить это действие несколько раз, прежде чем вы загрузите нужный образ. Поэтому вместо использования загруженного образа напрямую сделайте его снимок. Чтобы отслеживать свои версии, я добавляю `01` к новому имени снимка:

```
# qemu-img create -f qcow2 \
   -o backing_file=Fedora-Cloud-Base-31-1.9.x86_64.qcow2 \
Fedora-Cloud-Base-01.qcow2
```

6. **Скопируйте файлы в каталог образов.** Рекомендую копировать образы в каталог `/var/lib/libvirt/images/` при использовании их на гипервизоре (служба `libvirtd`). Например, чтобы скопировать облачный образ и образ данных в этот каталог, введите следующее:

```
# cp Fedora-Cloud-Base-31-1.9.x86_64.qcow2 \
Fedora-Cloud-Base-01.qcow2 \
fedora31-data.iso \
/var/lib/libvirt/images/
```

7. **Запустите облачную систему.** Переместив файлы на место, выполните следующие команды, чтобы запустить экземпляр облачного образа:

```
# cd /var/lib/libvirt/images
# virt-install --import --name fedora31-01 --ram 4096 --vcpus 2 \
--disk path=Fedora-Cloud-Base-01.qcow2,format=qcow2,bus=virtio \
--disk path=fedora21-data.iso,device=cdrom \
--network network=default &
```

Приведенный пример `virt-install` показывает, что виртуальной машине назначены 4 Гбайт оперативной памяти (`--ram 4096`) и два виртуальных процессора (`--vcpus 2`). Значения RAM и VCPU в вашей системе могут отличаться в зависимости от ресурсов вашего компьютера.

Сейчас на вашем гипервизоре работает виртуальная машина `fedora31-01`. Когда она загрузится, должно открыться окно консоли, позволяющее войти в новую облачную виртуальную машину.

Исследование облачной системы

Чтобы исследовать созданный облачный образ, можете открыть запущенный экземпляр и посмотреть, что находится внутри него. Один из способов сделать это, если он еще не открыт, — открыть виртуальную машину командой `virt-viewer`:

```
# virt-viewer fedora31-01
```

В появившемся окне консоли примените данные, которые мы добавили в образ, чтобы войти в систему. Задействуйте `fedora` в качестве пользователя и `cloudpass` в качестве пароля для входа в систему. Пользователь `fedora` имеет привилегии `sudo`, поэтому вы можете применять эту учетную запись для исследования системы, вводя команды. Так вы увидите, откуда пользовательские данные были скопированы в этот экземпляр системы:

```
$ sudo cat /var/lib/cloud/instances/FedoraWS01/user-data.txt
#cloud-config
password: cloudpass
chpasswd: {expire: False}
```

Базовая настройка облака выполняется в файле `/etc/cloud/cloud.cfg`. В примере видно, что учетная запись суперпользователя по умолчанию отключена. В нижней части файла вы видите, что пользователь с именем `fedora` — это пользователь по умолчанию, он имеет права `sudo`, и не требует вводить пароль:

```
$ sudo cat /etc/cloud/cloud.cfg
users:
 - default
disable_root: 1
...
system_info:
  default_user:
    name: fedora0
    lock_paswd: true
    gecos: Fedora Cloud User
    groups: [wheel, adm, systemd-journal]
    sudo: ["ALL=(ALL) NOPASSWD:ALL"]
    shell: /bin/bash
  distro: fedora
  paths:
    cloud_dir: /var/lib/cloud
    templates_dir: /etc/cloud/templates
  ssh_svcname: sshd

# vim:syntax=yaml
```

В файле `cloud.cfg` вы можете увидеть и другую информацию. Например, какие модули `cloud_init_module` запускаются во время инициализации (те, которые устанавливают имя хоста или запускают ведение журнала `rsyslog`). Вы увидите модули `cloud_config_modules`, которые устанавливают локальный и часовой пояс и запускают дополнительные инструменты настройки, такие как Chef и Puppet.

Поскольку репозитории `yum` подключены, вы можете установить любые пакеты, доступные из репозиториев Fedora, при условии, что у вас есть доступное сетевое подключение (DHCP должен был назначить адреса виртуальной машине по умолчанию).

Копирование облачного образа

Если вас устраивает созданный облачный образ, сохраните его копию для последующего запуска, сделав клон из двух образов — облака и образа данных. Чтобы создать клон запущенного облачного экземпляра с помощью утилиты `virt-manager`, выполните следующие действия.

1. **Запустите утилиту virt-manager.** В хост-системе, на которой запущена виртуальная машина, выполните команду `virt-manager` или запустите Virtual Machine Manager с экрана Acivities (Приложения) на рабочем столе.
2. **Приостановите виртуальную машину.** Щелкните правой кнопкой мыши на обозначении образа виртуальной машины в окне `virt-manager` и выберите пункт Pause (Приостановить). Виртуальная машина станет неактивной.
3. **Скопируйте виртуальную машину.** Снова щелкните правой кнопкой мыши на имени образа виртуальной машины и выберите пункт Clone (Клонировать). Появится окно Clone Virtual Machine (Клонировать виртуальную машину) (рис. 28.1).
4. **Выберите настройки для копии.** Для облачного образа и образа данных можно выбрать либо создание новых копий, либо использование их совместно с существующей виртуальной машиной. Далее выберите пункт Clone (Клонировать).

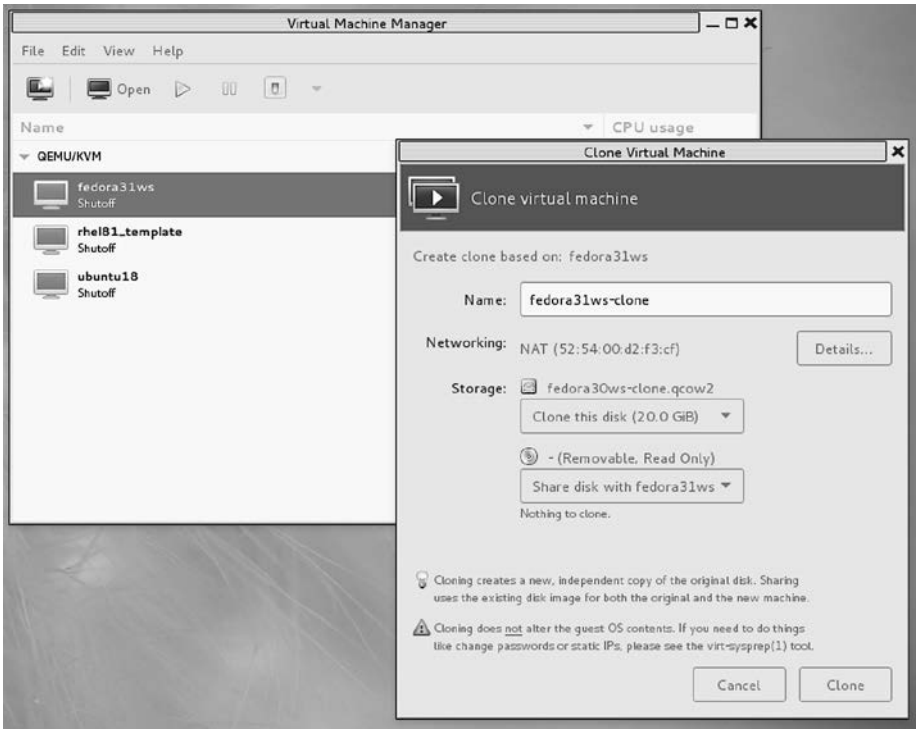


Рис. 28.1. Клонирование позволяет сохранить постоянную копию облачного образа

Теперь можно запускать и останавливать клонированный облачный образ, а также управлять им по своему усмотрению из окна Virtual Machine Manager или с помощью команды `virsh`. Большое преимущество создания клонов заключается в том, что вы можете вносить в них любые изменения, не меняя оригинал. Просто удалите клон, когда закончите, или быстро создайте новый, если он вам понадобится.

Расширение настроек службы cloud-init

Вы можете добавить в свои файлы `meta-data` и `user-data` гораздо больше информации для настройки облачных образов. Примеры настроек `cloud-init` можно найти на странице Cloud-Init (cloudinit.readthedocs.org/en/latest/topics/examples.html). В следующих разделах приведены примеры настроек, которые можно добавить в файлы `user-data`.

ПРИМЕЧАНИЕ

Файлы user-data и meta-data имеют формат yaml. Этот формат использует отступы и хорошо известные разделители. Элементам в списке предшествуют дефис и пробел. Ключи и значения разделяются двоеточием и пробелом. Если вы не знакомы с форматом yaml, рекомендую изучить его на сайте проекта Yaml (github.com/yaml).

Добавление ssh-ключей с помощью службы cloud-init

Вместо того чтобы использовать пароли для входа в облачные образы, вы можете задействовать аутентификацию на основе ключей вместе с командой `ssh` для входа в систему по сети. Такой способ обычно применяют облачные провайдеры для предоставления пользователю доступа к облачным образам.

Если вы уже сгенерировали открытый и закрытый `ssh`-ключи для учетной записи пользователя, с помощью которой планируете создать облачный образ, можете задействовать их для аутентификации. Если сгенерировали пару ключей RSA, то по умолчанию открытый ключ находится в файле `id_rsa.pub`:

```
# cat $HOME/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDMzdq6hqDUhueWzl7rIUwjxB/rrJY4
oZpoWINzeGVf6m8wXlHmmqd9C7LtnZg2P24/ZBb3S1j7vK2WymOcwEoWekhbZHBAyYeqX
KYQQjUB2E2Mr6qMkmrjQBx6ypxbz+VwADNC
wegY5RCUoNjrN43GVu6nSOxhFf7hv6dtCjvosOvtt0979YS3UcEyrobpNzreGSJ8FMPM
RFMWWg68Jz5hOMCIE1IldhpODvQVbTNsn/STxO7ZwSYV6kfDj0szvdoDDCyh8mPNC1kI
Dhf/qu/Zn1kxQ9xfecQ+SUi+2IwN69o1fNpexJPFr+Bwjkwcrk58C6uowG5eNSgnuu7G
MUkT root@host2.example.com
```

Открытый ключ из этого файла обычно копируется в файл `$HOME/.ssh/authorized_keys` для пользователя удаленной системы, в которую нужно войти. Можно добавить ключ к этому файлу на облачном образе, применяя записи в файле `user-data`, который выглядит следующим образом:

```
users:
  – default
  – name: wsmith
    gecos: William B. Smith
    primary-group: wsmith
    sudo: ALL=(ALL) NOPASSWD:ALL
    lock-passwd: true
    ssh-authorized-keys:
      – ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDMzdq6hqDUhueWzl7rIUwj
xB/rrJY4oZpoWINzeGVf6m8wXlHmmqd9C7LtnZg2P24/
```

```
ZBb3S1j7vK2WymOcwEoWekhbZHBAyYeqXKYQQjUB2E2Mr6qMkmrjQBx6ypxbz+V
wADNCwegY5RCUoNjrN43GVu6nSOxhFf7hv6dtCjvosOvtt0979YS3UcEyrobpNz
reGSJ8FMPMRFMWWg68Jz5hOMCIE1IldhpODvQVbTNsn/
STxO7ZwSYV6kfDj0szvdoDDCyh8mPNC1kIDhf/qu/
Zn1kxQ9xfecQ+SUi+2IwN69o1fNpexJPFr+Bwjkwcrk58C6uowG5eNS
gnuu7GMUkT root@host2.example.com
```

Из примера видно, что `wsmith` — это пользователь по умолчанию. Запись `gecos` обычно является полным именем пользователя, применяемым в пятом поле файла `/etc/passwd`. Пароль для этого пользователя заблокирован. Но, поскольку запись `ssh-rsa` из моей учетной записи суперпользователя на `host2.example.com` предоставляет `ssh-authorized-keys` для пользователя, я могу войти в облачный образ как пользователь `wsmith`, применив ключи `ssh` без ввода пароля (при условии, что мой закрытый ключ связан с этим открытым ключом).

Добавление программного обеспечения с помощью службы cloud-init

Вы не ограничены программным обеспечением, которое имеется в вашем облачном образе. В файле пользовательских данных вы можете определить репозитории `YUM` (в Fedora и RHEL) или `apt` (в Ubuntu или Debian), а затем выделить любые пакеты, которые хотите установить при запуске облачного образа.

В следующем примере показано, как могут выглядеть записи в файле `user-data` при добавлении репозитория `YUM` (для систем Fedora или RHEL) в облачный образ и последующей установке пакетов из этого или любого другого подключенного репозитория:

```
myownrepo:
    baseurl: http://myrepo.example.com/pub/myrepo/
    enabled: true
    gpgcheck: true
    gpgkey: file:///etc/pki/rpm-gpg/RPM-GPG-KEY-MYREPO
    name: My personal software repository
packages:
 - nmap
 - mycoolcmd
 - [libmystuff, 3.10.1-2.fc21.noarch]
```

Здесь новый репозиторий `yum` создается в файле `/etc/yum.repos.d/myownrepo.repo`. Для проверки правильности установленных пакетов используется служба `gpgkey` и включается проверка GPG. После этого устанавливаются пакет `nmap` (который находится в стандартном `yum`-репозитории Fedora), затем пакет `mycoolcmd` (из моего частного репозитория) и конкретная версия пакета `libmystuff`.

Настройка репозиториев программного обеспечения `apt` для Ubuntu выполняется немного иначе. Отказоустойчивые первичные и защитные зеркала пакетов `apt` настраиваются по умолчанию (в файле `cloud.cfg` на образе) вместе с параметрами, позволяющими образу при запуске в облаке Amazon EC2 искать пакеты в ближай-

шем окружении. Чтобы добавить дополнительные репозитории, записи в файле `user-data` можно сделать такими:

```
apt_mirror: http://us.archive.ubuntu.com/ubuntu/
apt_mirror_search:
 - http://myownmirror.example.com
 - http://archive.ubuntu.com
packages:
 - nmap
 - mycoolcmd
 - [libmystuff, 3.16.0-25]
```

Строка `myownmirror.example.com` указывает утилите `apt` использовать собственный частный репозиторий `apt` для поиска пакетов. Обратите внимание на то, что пакеты, которые вы хотите установить, вводятся в основном в том же формате, что и в Fedora, хотя конкретная информация о версии (если она есть) в некоторых случаях может выглядеть иначе.

Вы можете добавить множество других настроек в свои файлы `user-data` и `meta-data`. Обратитесь к странице Cloud-Init Cloud Config Examples (cloudinit.readthedocs.org/en/ latest/topics/examples.html) за подробностями.

Использование службы cloud-init в корпоративных вычислениях

До сих пор описание службы `cloud-init` в этой главе было сосредоточено на создании облачного образа, ручном добавлении данных конфигурации и временном запуске его в качестве виртуальной машины на локальном гипервизоре. Этот подход полезен, если вы хотите понять, как работает `cloud-init` и какие есть возможности для настройки облачных образов в соответствии с вашими нуждами. Однако этот подход плохо масштабируется, если вы хотите управлять крупными системами виртуальных машин.

Служба `cloud-init` поддерживает концепцию *источников данных*. Поместив данные файлов `user-data` и `meta-data` в источник данных, не нужно вводить эту информацию вручную в облачный образ, как мы делали ранее в этой главе. Вместо этого, когда служба `cloud-init` начинает работать в образе, она знает, что нужно искать источники данных не только в локальной системе, но и за ее пределами.

Для облаков Amazon EC2 `cloud-init` запрашивает определенный IP-адрес (`http://169.254.169.254/`) данных. Например, он может проверить `http://169.254.169.254/2009-04-04/meta-data/` для поиска файлов `meta-data` и `http://169.254.169.254/2009-04-04/user-data/` для поиска файлов `user-data`. Это позволяет хранить данные конфигурации и получать к ним доступ из основного расположения.

Что касается того, что может содержаться в файлах `meta-data` и `user-data`, то для развертывания ваших облачных образов можно разработать гораздо более сложные схемы конфигурации. Служба `cloud-init` поддерживает инструмен-

ты настройки, такие как Puppet (puppetlabs.com/puppet/puppet-open-source) и Chef (www.chef.io/chef/). Они позволяют применять сценарии конфигураций к вашим облачным образам, даже заменяя компоненты или перезапуская службы по мере необходимости, чтобы вернуть систему в желаемое состояние.

Однако на данный момент моя задача не в том, чтобы сделать из вас полноценных облачных администраторов (несколько сотен страниц назад вы могли бы быть и новичком в Linux). Я лишь хочу, чтобы вы поняли, с чем будете иметь дело, если в итоге окажетесь в облачном центре обработки данных. Потому что многие считают, что в не слишком отдаленном будущем большинством центров обработки данных станут управлять как облачными инфраструктурами.

В этой главе мы рассматривали внутреннюю часть настройки Linux для облачных вычислений. Вернемся назад и посмотрим, как можно использовать для запуска собственных виртуальных машин две самые популярные облачные платформы на базе Linux, OpenStack и Amazon EC2.

Развертывание облачных образов с помощью OpenStack

Технология *OpenStack* обеспечивает постоянно развивающуюся платформу для управления физической инфраструктурой облачных вычислений, а также виртуальными системами, которые работают на ней. OpenStack позволяет развернуть собственное частное облако или сделать его публичным.

Вместо того чтобы создавать собственное облако OpenStack, я покажу, как можно использовать OpenStack для развертывания виртуальных машин с панели мониторинга OpenStack. Если хотите сделать это самостоятельно, задействуйте OpenStack одним из доступных способов.

- **Дистрибутивы Linux.** Дистрибутивы Fedora, Ubuntu и CentOS имеют бесплатные версии OpenStack, которые можно развернуть самостоятельно. Red Hat Enterprise Linux предлагает версию OpenStack, доступную по подписке, ее довольно сложно настроить. Часть универсальных настроек OpenStack может работать на одной машине, но я думаю, что будет лучше, если вы начнете с трех физических машин — одного узла контроллера и двух гипервизоров.
- **Публичные облака OpenStack.** Можно использовать общедоступные облака OpenStack, которые различаются по цене. Список общедоступных облаков OpenStack имеется на сайте проекта OpenStack www.openstack.org/marketplace/public-clouds.

Для начала я хочу помочь вам запустить систему Linux в облаке, когда нет возможности выполнять определенные действия на собственных компьютерах. И покажу, как веб-интерфейс облачного провайдера (например, панель мониторинга OpenStack) может значительно упростить конфигурацию облака, которую мы выполняли вручную с помощью службы `cloud-init` ранее в этой главе.

Панель мониторинга OpenStack

Необходимо начать с установки OpenStack. Администратор среды OpenStack создал для меня проект `cnegus-test-project` и учетную запись пользователя (`cnegus`), которая позволяет получить доступ к этому проекту.

Вот что я планирую сделать:

- **настроить сеть.** Точно так же, как настроил бы маршрутизатор и физически подключил свои компьютеры к нему, я создам виртуальную сеть. Она будет включать в себя набор адресов, которые распространяются на мои виртуальные машины через службу DHCP;
- **настроить виртуальные машины.** Я пройду через процесс настройки и развертывания нескольких виртуальных машин.

Версия OpenStack, используемая для этого примера, — это Red Hat OpenStack Platform (RHEL-OSP). Она похожа на любую другую среду OpenStack. В следующем разделе показано, как начать настройку сети.

Настройка виртуальной сети OpenStack

Чтобы настроить виртуальную сеть OpenStack, выполните следующие действия.

1. **Зайдите в OpenStack.** Применив имя пользователя и пароль, назначенные вам администратором OpenStack, войдите на панель мониторинга OpenStack из своего браузера. Вы должны увидеть экран Overview (Обзор), похожий на показанный на рис. 28.2.
2. **Создайте сеть.** Чтобы создать сеть, в левом столбце экрана Overview (Обзор) выберите пункт Networks (Сети). На появившемся экране Networks (Сети) создайте новую сеть следующим образом (примеры, которые я использовал, приведены в скобках).
   - Нажмите кнопку Create Network (Создать сеть).
   - На вкладке Network (Сеть) введите сетевое имя (`mynet`).
   - На вкладке Subnet (Подсеть) введите имя подсети (`mysub01`), сетевой адрес (192.168.100.0/24), версию IP (IPv4) и IP-адрес шлюза (192.168.100.). Оставьте поле Disable Gateway (Запретить шлюз) пустым.
   - На вкладке Subnet Detail (Детали подсети) введите диапазон IP-адресов, разделенных запятыми, в поле Allocation Pool (Выделение пулов). В своем примере я ввел адрес `192.168.100.10, 192.168.100.50`, чтобы раздать клиентам диапазон IP-адресов от 192.168.100.10 до 192.168.100.50. Примите предложение сервера имен от администратора вашего облака OpenStack или используйте общедоступный DNS-сервер (например, Google 8.8.8.8 или 8.8.4.4).
   - Нажмите кнопку Create (Создать), чтобы создать новую сеть. Она появится на экране Networks (Сети).

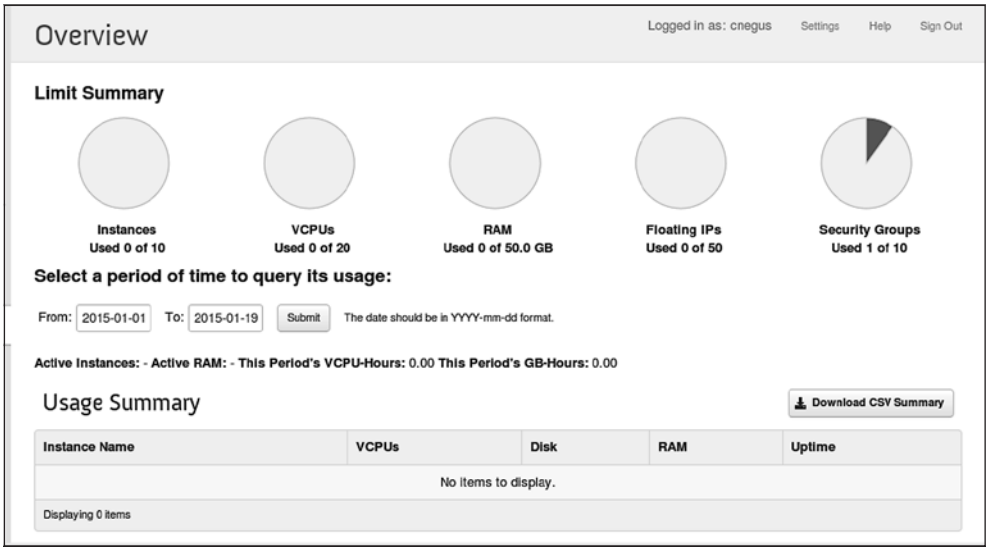


Рис. 28.2. Войдите на панель мониторинга OpenStack

3. **Создайте маршрутизатор.** Чтобы виртуальные машины могли получить доступ к Интернету, необходимо определить маршрутизатор, подключенный к вашей частной сети на одном интерфейсе, и сеть, которая может выйти в общедоступный Интернет на другом. Вот как это сделать.
   - В левой колонке выберите раздел Routers (Маршрутизаторы).
   - Нажмите кнопку Create Router (Создать маршрутизатор).
   - Введите имя маршрутизатора (`myrouter01`) и нажмите кнопку Create Router (Создать маршрутизатор).
   - Нажмите кнопку Set Gateway (Установить шлюз).
   - На экране Set Gateway (Установить шлюз) щелкните на поле External Network (Внешняя сеть) и выберите одну из доступных внешних сетей. Оставьте поля Router Name (Название маршрутизатора) и Router ID (ID маршрутизатора) такими, какие они есть. Нажмите кнопку Set Gateway (Установить шлюз). Новый маршрутизатор появится на экране Routers (Маршрутизаторы).
4. **Подключите свою сеть к внешнему маршрутизатору.** На экране Routers (Маршрутизаторы) выберите имя маршрутизатора, который вы только что создали (`myrouter1`).
   - На экране Router Details (Маршрутизатор) нажмите кнопку Add Interface (Добавить интерфейс).
   - На экране Add Interface (Добавить интерфейс) щелкните на поле Subnet (Подсеть) и выберите созданную ранее подсеть (`mynet:192.168.100.0/24 mysub01`). Не нужно менять имя маршрутизатора или его ID.
   - Нажмите кнопку Add Interface (Добавить интерфейс).

5. **Просмотрите сетевую топологию.** Щелкните кнопкой мыши на разделе Network Topology (Сетевая топология) в левом столбце. Затем наведите указатель мыши на имя маршрутизатора (`myroute01`). На рис. 28.3 показан пример того, как может выглядеть ваша сетевая конфигурация.

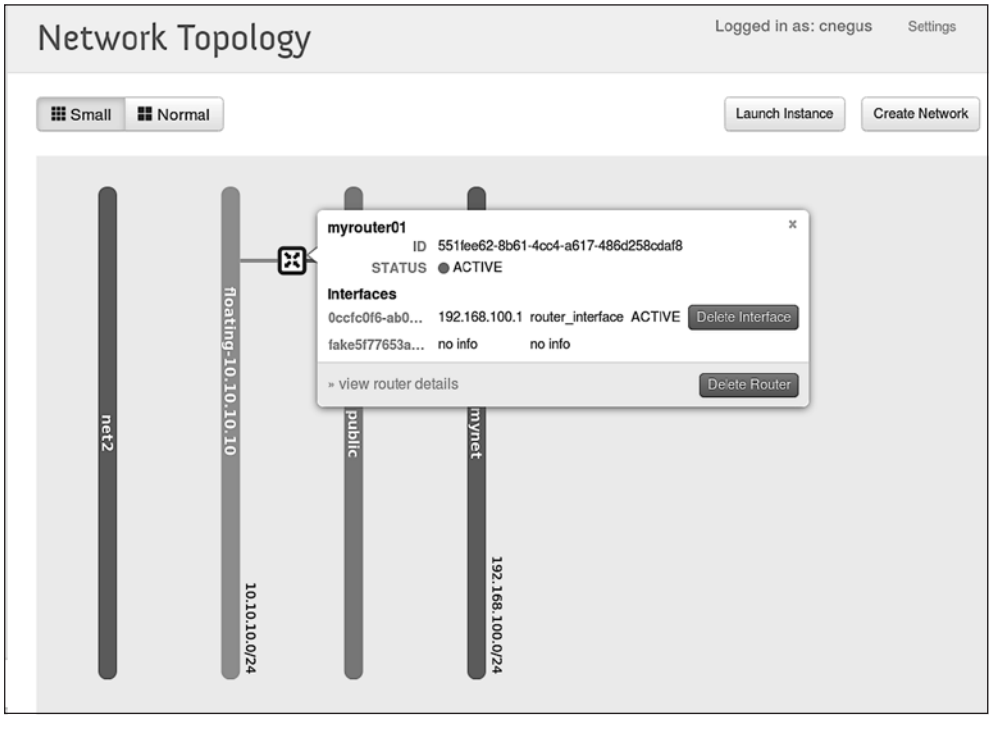


Рис. 28.3. Сетевая топология на панели мониторинга OpenStack

При наличии сети вы можете создавать ключи для доступа к виртуальным машинам в OpenStack.

Настройка ключей для удаленного доступа

Обычный способ настройки доступа к виртуальным машинам в облачной среде — создание пары «открытый/закрытый ключ», которая обеспечивает безопасный доступ к виртуальным машинам с помощью службы `ssh` и связанных с ней инструментов из настольной системы. Закрытый ключ хранится в домашнем каталоге пользователя рабочего стола, а открытый ключ вводится в виртуальную машину, чтобы вы могли удаленно войти в нее (через службу `ssh`) без ввода пароля. Вот как можно настроить ключи.

1. **Перейдите на вкладку Access & Security (Доступ и безопасность).** В левой колонке выберите Access & Security (Доступ и безопасность).

2. **Создайте пару ключей.** Если у вас уже есть пара ключей, можете перейти к следующему шагу. Если нет, выберите вкладку Keypairs (Ключевая пара) и нажмите кнопку Create Keypair (Создать ключевую пару). Когда появится окно Create Keypair (Создать ключевую пару), выполните следующие действия.
   - Введите название ключевой пары (`mycloudkey`) и нажмите кнопку Create Keypair (Создать ключевую пару). Появится всплывающее окно с вопросом, хотите вы открыть или сохранить файл `*.pem`.
   - Выберите вариант Save File (Сохранить файл) и нажмите кнопку OK. Сохраните файл в каталоге `ssh` своего домашнего каталога.

Теперь можете развертывать образ OpenStack (облачную виртуальную машину).

Запуск виртуальной машины в OpenStack

Чтобы начать запуск нового инстанса облачной виртуальной машины, перейдите в левый столбец и выберите раздел Instances (Инстансы). Затем нажмите кнопку Launch instance (Запустить инстанс). Появится окно Launch instance (Запустить инстанс). Чтобы ввести информацию, необходимую для запуска инстанса, выполните следующие действия.

1. **Выберите вкладку Details (Подробности).** На ней измените следующие параметры.
   - Availability Zone (Зона доступности). *Зона доступности* состоит из группы вычислительных узлов. Отдельные зоны иногда создаются для идентификации группы компьютеров, которые физически находятся вместе (например, на одной стойке) или выполняют одинаковые аппаратные функции (чтобы их можно было использовать для одних и тех же типов приложений). Выберите из списка одну из зон.
   - Instance Name (Имя инстанса). Дайте инстансу любое понятное имя.
   - Flavor (Тип инстанса). Выбирая тип, вы выделяете набор ресурсов инстансу виртуальной машины. Ресурсы включают в себя количество виртуальных ядер процессора, объем доступной памяти, выделенное дисковое пространство и доступное эфемерное дисковое пространство. (*Эфемерное пространство* — это пространство, доступное с локального диска во время работы инстанса, но не сохраняемое при его завершении.) Типы по умолчанию — это `m1.tiny`, `m1.small`, `m1.medium`, `m1.large` и `m1.xlarge`. Другие типы добавляет администратор облака.
   - Instance Count (Количество). По умолчанию это значение равно 1, что означает запуск одного инстанса. Если нужно, измените его, чтобы запустить больше инстансов.
   - Instance Boot Source (Выберите источник загрузки). Инстанс может быть загружен из образа, снимка, с тома, из образа, включающего новый том, или снимка тома, включающего новый том.

- Image Name (Имя образа). Выберите образ, который хотите запустить. Названия обычно включают имена операционных систем, которые вы загружаете.
- Device size (Размер устройства) и Device Name (Имя устройства) (не обязательно). Если выбрано включение нового тома при выборе источника загрузки инстанса, то необходимо задать размер (в гигабайтах) и имя устройства для тома в этих полях. Если выбрать `vda` в качестве имени устройства (для первого диска на виртуальной машине), то само это устройство будет иметь значение `/dev/vda`.

2. **Выберите вкладку Access & Security (Доступ и безопасность).** Перейдите на вкладку Access & Security (Доступ и безопасность) и выберите пару ключей, созданную ранее.
3. **Выберите вкладку Networking (Сеть).** Откройте вкладку Networking (Сеть). Из списка доступных сетей выберите нужную и с помощью мыши перетащите ее в поле Selected Networks (Выбранные сети).
4. **Задайте дополнительные настройки.** Вы можете добавить команды и скрипты, которые настраивают систему после загрузки. Здесь можно указать информацию, которая добавлена в файлы `user-data`, описанные в разделах, посвященных службе `cloud-init`, ранее в этой главе.

Нажмите кнопку Launch (Запустить), чтобы включить виртуальную машину. Когда она запущена, можете войти в эту систему, для чего выберите инстанс и перейдите на вкладку Console (Консоль). В окне Console (Консоль) виртуальной машины должно появиться приглашение войти в систему. Если вы хотите получить доступ к виртуальной машине с помощью службы `ssh` по сети, прочтите следующий пункт.

Доступ к виртуальной машине через службу ssh

Введите открытый ключ в работающей виртуальной машине, и она будет готова войти в систему с помощью службы `ssh`. Но прежде чем войти в систему, сделайте следующие шаги.

1. **Добавьте назначаемый IP-адрес.** На панели мониторинга OpenStack перейдите в раздел инстансов, нажмите кнопку More (Подробнее) на записи, содержащей инстанс, и нажмите кнопку Associate Floating IP (Связать назначаемый IP).

   Нажмите на знак плюс (+) рядом с полем IP Address (IP-адрес), выберите пул с плавающими IP-адресами и нажмите кнопку Allocate IP (Назначить). Назначаемый адрес должен появиться в поле IP Address (IP-адрес). Выберите порт, который необходимо связать, и нажмите кнопку Associate (Связать).
2. **Используйте службу ssh для доступа к инстансу.** В системе Linux, имеющей доступ к сети, в которой есть назначаемый адрес, выполните команду `ssh` для входа в систему. Предположим, что файл `.pem` вашего ключа называется `mycloud.pem`, тогда пользователь по умолчанию на инстансе — это `cloud-user`, а IP-адрес — 10.10.10.100, и вы можете ввести следующую команду для входа в систему:

```
# ssh -i mycloud.pem cloud-user@10.10.10.100
```

Теперь можно войти в систему без пароля. Для администрирования системы примените команду sudo от имени пользователя по умолчанию.

Развертывание облачных образов с помощью Amazon EC2

Amazon Elastic Computer Cloud (Amazon EC2) — это облачная платформа, которая подходит для платных облачных вычислений. Как и OpenStack, она позволяет выбирать из предварительно настроенных образов виртуальных машин нужные и настраивать их по мере необходимости.

Чтобы начать использовать Amazon EC2 для запуска виртуальных машин, откройте страницу Getting Started (Начало работы с AWS) на сайте Amazon Web Services (aws.amazon.com/getting-started) и перейдите по ссылкам для создания новой учетной записи. После входа в систему отображается полный спектр служб AWS. Выберите пункт Sign In to the Console (Войти в консоль), и появится консоль управления AWS (рис. 28.4).

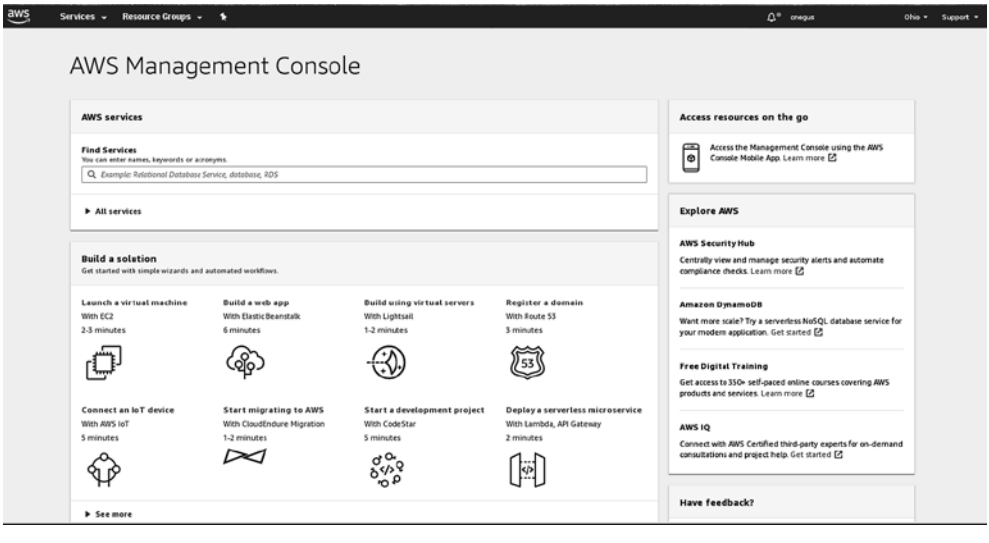


Рис. 28.4. Запуск облачных инстансов с помощью консоли управления Amazon EC2

Чтобы запустить первый инстанс виртуальной машины Linux, выполните следующие действия.

1. Выберите пункт Launch a Virtual Machine (Запустить виртуальную машину). Выберите систему Linux (Red Hat Enterprise Linux, SUSE Linux, Ubuntu и т. д.) или Windows AMIs (Amazon Machine Images) для запуска.

2. Найдите нужный образ и нажмите кнопку Select (Выбрать).
3. На странице выбора типа инстанса выберите нужный, основываясь на количестве процессоров, объеме памяти, типе хранилища и сетевых функциях.
4. Выбрав тип инстанса, нажмите кнопку Next (Далее) и перейдите в окно Configure Instance Details (Детальная информация).
5. На экране Configure Instance Details (Детальная информация) выберите существующий VPC или создайте новый. Затем изменяйте любые настройки. Например, выберите пункт Enable (Включить) в разделе Auto-assign Public IP (Назначить публичный IP), чтобы иметь возможность войти в свой инстанс через Интернет.
6. Выберите пункт Review and Launch (Просмотреть и запустить). Появится окно Review Instance Launch (Просмотр запуска инстанса).
7. Просмотрите настройки инстанса и нажмите кнопку Launch (Запустить), чтобы запустить его. На рис. 28.5 показан пример инстанса системы RHEL 8, готового к запуску.

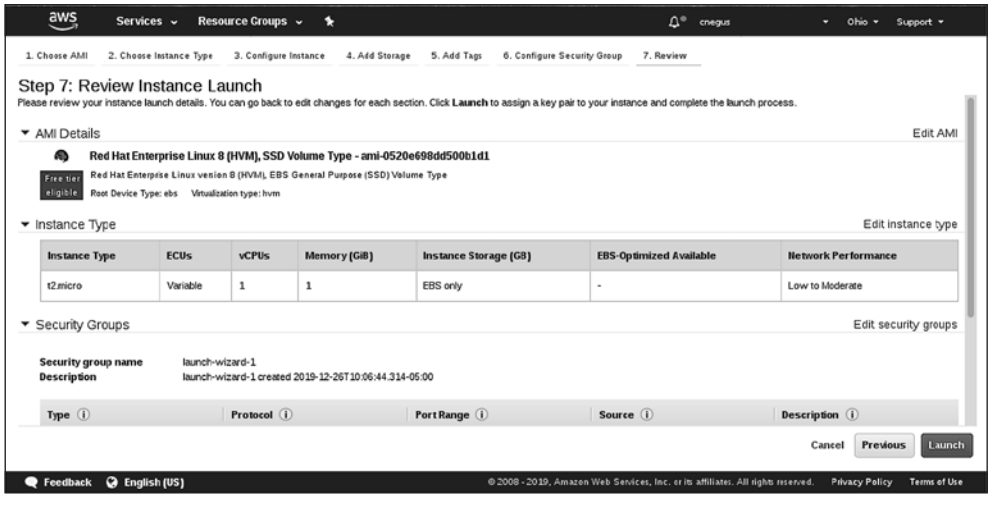


Рис. 28.5. Настройте и запустите инстанс RHEL 8 на AWS

8. Выберите существующую пару ключей или нажмите кнопку Create a New Key Pair (Создать новую пару ключей), чтобы создать закрытый и открытый ключи для инстанса.
9. Выберите пункт Launch Instances (Запустить инстансы), чтобы запустить инстанс.
10. Выберите пункт View Instances (Просмотр инстансов), чтобы просмотреть список запущенных инстансов. Для их поиска по имени используйте соответствующее поле.

11. Выберите нужный инстанс, а затем нажмите кнопку Connect (Подключиться). Следуйте инструкциям, применяя службу `ssh` для входа в созданный вами публичный IP-адрес. Например, команда для входа в инстанс AWS будет выглядеть следующим образом:

```
# ssh -i "youraws.pem" ec2-user@ec2-w-xx-yyy-zz.us-east-
2.compute.amazonaws.com
```

12. Закончив работу с инстансом, отключите его, выбрав нужный на странице Instances (Инстансы), а затем выбрав Actions ▸ Instance State ▸ Terminate (Действия ▸ Состояние инстанса ▸ Завершить). При появлении запроса согласитесь на завершение, чтобы удалить инстанс и связанное с ним хранилище.

Важно помнить, что нужно удалять инстансы после окончания работы с ними, иначе с вас будут продолжать взимать плату.

Резюме

Понимание того, чем облачные вычисления отличаются от простой установки операционной системы непосредственно на компьютер, поможет вам адаптироваться, так как все больше и больше центров обработки данных переходят к облачным вычислениям. В начале этой главы я советовал взять несколько облачных образов, объединить их с данными и запустить на локальном гипервизоре Linux, чтобы понять, как работают облачные образы.

После этого я показал, как можно запускать собственные виртуальные образы на облачной платформе OpenStack, включая настройку сетевых интерфейсов, выбор способа запуска виртуального инстанса и запуск виртуального образа. Я также описал облачный сервис Amazon Elastic Compute Cloud, в котором вы можете заплатить за использование облачного хранилища и время обработки, если у вас недостаточно собственных вычислительных ресурсов.

В следующей главе рассказывается, как задействовать Ansible для автоматизации развертывания хост-систем и приложений в вашем центре обработки данных.

Упражнения

Чтобы выполнить упражнения этой главы, настройте хост-систему в качестве гипервизора (хост-компьютер KVM). Он нужен для запуска виртуальной машины. Если затрудняетесь с решением заданий, воспользуйтесь ответами к упражнениям, приведенными в приложении Б (хотя Linux позволяет решать задачи разными способами).

1. Чтобы создать образ виртуальной машины, установите пакеты `genisoimage`, `cloud-init`, `qemu-img` и `virt-viewer`.
2. Загрузите облачный образ из проекта Fedora.

3. С помощью команды `qemu-img` создайте снимок этого образа в формате `qcow2` с названием `myvm.qcow2`, который вы сможете использовать позже для объединения с вашими собственными данными.
4. Создайте файл метаданных `cloud-init` с именем `meta-data`, который устанавливает идентификатору `id` инстанса значение `myvm`, а имени локального хоста — `myvm.example.com`.
5. Создайте файл пользовательских данных `cloud-init` с именем `user-data`, который устанавливает паролю пользователя по умолчанию значение `test` и `chpasswd` и не даст истечь сроку действия пароля с помощью параметра `{expire:False}`.
6. Запустите команду `genisoimage`, чтобы соединить файлы `meta-data` и `user-data` и создать файл `mydata.iso`, который в дальнейшем будет применяться виртуальной машиной.
7. Используйте команду `virt-install`, чтобы соединить образ виртуальной машины `myvm.qcow2` с образом `mydata.iso` и создать новый образ виртуальной машины `newvm` на вашем гипервизоре.
8. Задействуйте команду `virt-viewer`, чтобы открыть консоль для виртуальной машины `newvm`.
9. Войдите в виртуальную машину `newvm` с помощью учетной записи пользователя `fedora` и пароля `test`, которые вы установили ранее.

29 Автоматизация приложений и инфраструктуры с помощью системы Ansible

В этой главе

- Система Ansible.
- Установка системы Ansible.
- Развертывание.
- Выполнение ad-hoc-команд.

Ранее в книге в основном рассматривалась ручная настройка отдельных систем Linux. Мы научились устанавливать программное обеспечение, редактировать файлы конфигурации и запускать службы непосредственно на машинах. Работа на отдельных хостах Linux является основополагающим умением для управления системами Linux, но сама по себе она плохо масштабируется. Вот тут и вступает в дело система Ansible.

Система Ansible сосредоточивает администрирование Linux не на отдельных системах, а на группах систем. Она перемещает конфигурацию этих узлов с каждой отдельной машины на узел управления. Она заменяет пользовательский интерфейс оболочки на каждой машине каталогами Ansible, которые запускают задачи на других машинах сети.

Хотя наше внимание сосредоточено на управлении системами Linux, Ansible может управлять также многими элементами вокруг систем Linux. Существуют модули Ansible для обеспечения того, чтобы машины были включены, сетевые устройства правильно настроены и удаленное хранилище было доступно.

Во всех центрах обработки данных, кроме самых маленьких, умение автоматически развертывать системы Linux и управлять ими и окружающей инфраструктурой становится обязательным требованием для многих ИТ-специальностей. Для полностью контейнеризованных центров обработки данных платформы приложений на базе Kubernetes, такие как OpenShift, становятся отраслевым стандартом для оркестрации и автоматизации контейнеров (см. главу 30 «Развертывание приложений в контейнеры с помощью кластера Kubernetes»). Для инфраструктуры и более традиционных развертываний приложений чаще всего используется система Ansible.

В этой главе мы рассмотрим, как начать работу с Ansible. Затем перейдем к развертыванию приложения в наборе систем Linux с помощью Ansible и поговорим о том, как работать с этими системами, повторно развертывая каталоги и выполняя специальные команды.

Система Ansible

Система Ansible расширяет все полученные до этого знания о Linux. На базовом уровне Ansible включает в себя:

- язык автоматизации, описывающий задачи, которые нужно выполнить для достижения определенного состояния. Они собраны в файлы *playbooks*;
- механизм автоматизации, который используется для запуска файлов playbooks;
- интерфейсы, которые применяются для управления каталогами и другими компонентами автоматизации и их защиты и реализуются с помощью команд и архитектуры RESTful API.

Задействуя файлы inventory (которые определяют наборы хостов) и playbooks (они определяют наборы действий, выполняемых на этих хостах), система Ansible настраивает хост-системы следующим образом.

- **Простая настройка функций.** Вы создаете файлы inventory и каталоги в виде простых текстовых файлов, где идентифицируете компоненты Linux, на которые действуют модули. Кодирование не требуется.
- **Настройка нужных результатов.** В этом случае описываются ресурсы, которые определяют состояние, необходимое для того, чтобы объект находился на узле. Это состояние может быть запущенной службой `systemd`, сетевым интерфейсом с определенным набором адресов или созданным разделом диска определенного размера. Если по какой-то причине состояние объекта изменяется, вы можете снова запустить каталог, чтобы Ansible вернула узел в заданное состояние.
- **Соединения SSH.** По умолчанию на каждом узле хоста должна быть запущена служба SSH, настроенная таким образом, чтобы система Ansible могла

связываться с ней из узла управления. Аутентификация на основе ключей для обычных учетных записей пользователей позволяет это сделать, а для получения привилегий суперпользователя применяется команда `sudo`. Поскольку вы задействуете службу SSH, которая, вероятно, уже запущена на хосте, не нужно запускать дополнительные агенты или настраивать специальные правила брандмауэра.

Изучив основы работы Ansible, вы сможете выполнять широкий спектр сложных действий, таких как перечисленные далее.

- **Создание инфраструктуры.** С помощью системы Ansible можно создать инфраструктуру, необходимую приложениям, будь то установка операционных систем на пустом компьютере или в качестве гипервизоров (вместе с их виртуальными машинами), настройка устройств хранения данных или сетевых устройств. В каждом из этих случаев Ansible может использовать существующие инструменты настройки, чтобы ими можно было управлять в одном месте.
- **Развертывание приложений.** Описывая желаемое состояние приложений, система Ansible может не только задействовать задачи для развертывания наборов приложений на нескольких узлах и устройствах, но и воспроизводить каталоги, чтобы вернуть приложение в желаемое состояние, когда функция нарушена или изменена непреднамеренно.
- **Управление контейнерами и операторами.** Недавние обновления системы Ansible позволяют ей развертывать контейнерные приложения в инфраструктуре Kubernetes, такой как OpenShift. Операторы в OpenShift, которыми может управлять оператор Ansible, способны не только определять состояние контейнерных приложений, но и реагировать на изменения в режиме реального времени и облегчать обновление. (Подробнее см. на странице Ansible Operator www.ansible.com/blog/ansible-operator.)
- **Управление сетью и хранилищем.** Задачи настройки, тестирования, проверки и улучшения сетевой инфраструктуры, которые чаще всего выполняются вручную, можно автоматизировать с помощью Ansible. Сейчас доступны тонны коммерческих и бесплатных каталогов, которые предлагают те же интуитивно понятные инструменты Ansible, которые вы используете для развертывания систем Linux и которые к тому же предназначены для конкретных сети (docs.ansible.com/ansible/latest/network/index.html), устройств и сред хранения (docs.ansible.com/ansible/latest/modules/list_of_storage_modules.html).
- **Управление облачными средами.** Конечно, вы можете развернуть инфраструктуру на пустом компьютере, но система Ansible позволяет использовать инструменты для подготовки инфраструктуры и приложений к облачным средам. Только Amazon Web Services (AWS) доступны около 200 модулей Ansible для управления инфраструктурой и приложениями. Имеются также модули для Alibaba, Azure, Google и нескольких десятков других облачных сред.

Элементы системы Ansible

Действие запущенных файлов playbooks распространяется на одну или несколько целевых хост-систем (представлены файлами inventories), они выполняют элементы, называемые *plays*. Каждый такой элемент содержит одну или несколько задач. Для выполнения задачи элемент вызывает *модули*, которые выполняются в том порядке, в котором установлены. Перед использованием Ansible необходимо изучить компоненты системы.

Файлы inventories

Собирая хост-системы (узлы), которыми вы хотите управлять в так называемых файлах *inventories*, вы можете управлять машинами, отчасти похожими на группы. Сходство заключается в следующем:

- они расположены в аналогичных местах;
- предоставляют такие же услуги;
- назначаются конкретному этапу процесса, например разработке, тестированию, постановке и производству.

Вхождение хостов более чем в одну группу позволяет воздействовать на них с помощью различных типов атрибутов. Например, `host01` может находиться как в группе `newyork` (ее местоположение), так и в группе `ftp` (приложение, которое она предоставляет). Задачи, выполняемые в этих группах inventory, могут позволить каждому хосту получать сетевые настройки в зависимости от его местоположения и приложений, которые он запускает, то есть в зависимости от его назначения.

Существует множество способов создания файлов inventories. Вы можете установить линию статических серверов или создать ряд систем. А можете использовать динамические списки серверов от облачных провайдеров, таких как Azure, AWS и GCP.

Задействуя переменные, вы можете назначить атрибуты набору узлов в файле inventory. Эти переменные могут настраивать такие функции, как порт, с которого служба доступна с хоста, значение тайм-аута для службы или местоположение службы, используемой хостом (например, база данных для сервера протокола сетевого времени).

Подобно файлам playbooks, файлы inventory могут быть простыми текстовыми файлами, а также реализоваться из скрипта inventories.

Файлы playbooks

Файлы *playbooks* создаются в формате YAML и описывают конечное состояние чего-либо. Такое состояние может привести к установке программного обеспечения, настройке приложений или запуску служб. Этот формат может фокусироваться

только на приложении или включать всю окружающую его среду (сеть, хранилище, аутентификацию или другие функции).

Файлы playbooks предназначены для последующего развертывания тех же компонентов, адаптации к другим компонентам или восстановления первоначального замысла конкретного инстанса файла playbook. Поскольку playbooks предназначены для повторного применения, многие пользователи контролируют свои файлы, задействуя исходный код. Таким образом, вы можете отслеживать изменения с течением времени и сделать файлы playbooks доступными.

Скрипты plays

Внутри файла playbooks находится один или несколько скриптов *plays*. У каждого такого скрипта есть цель, например идентификатор `host`, который сообщает файлам playbooks, какие хост-системы должны работать. За ним может последовать `remote_user`, который сообщает playbooks, какому пользователю нужно пройти аутентификацию на хосте. Роль также может указывать на то, что ей необходимо повысить привилегии с помощью команды `sudo`, прежде чем она начнет решать задачи. После этого на хостах выполняется одна или несколько задач для определения фактической активности.

Задачи

На базовом уровне каждая задача запускает один или несколько модулей. Задача — это способ связать запускаемый модуль с параметрами и возвращаемыми значениями, которые к нему относятся.

Модули

Сейчас доступны сотни модулей Ansible, и они продолжают создаваться. При запуске модуль проверяет, что необходимое состояние, определяемое предоставленными ему параметрами, достигнуто, и если цель не находится в нужном состоянии, предпринимает действия, необходимые для перехода в это состояние. Индекс модулей сортирует эти модули по категориям (docs.ansible.com/ansible/latest/modules/modules_by_category.html).

Примеры модулей — это `yum`, `mysql_db` и `ipmi_power`. Модуль `yum` может устанавливать и удалять программные пакеты и репозитории `YUM` или иным образом управлять ими. Модуль `mysql_db` позволяет добавлять базу данных MySQL на хост или удалять ее оттуда. Модуль `ipmi_power` позволяет проверить состояние компьютеров с интерфейсами IPMI и убедиться, что они находятся в требуемом состоянии (включено или выключено).

Переменные применяются ко всем задачам. Например, с помощью модуля `yum` вы можете определить, следует ли устанавливать пакет, по его состоянию: если состояние пакета равно `installed`, то его устанавливать не нужно, даже если доступна

более свежая версия. Однако если вы задействуете переменную `latest`, то более новая версия пакета будет установлена, если он не обновлен.

Параметры позволяют добавлять информацию для изменения задачи. Например, с помощью модуля `user` при добавлении пользователя в систему можно определить его имя, пароль, идентификатор и оболочку.

Помимо настройки модулей выполнения, из файлов playbooks вы также можете запускать модули непосредственно из командной строки. Это позволяет действовать на хосте немедленно, без запуска всего файла playbook. Например, можно пропинговать набор хостов, чтобы убедиться, что они запущены, или проверить состояние службы. (См. подраздел «Запуск ad-hoc-команд в системе Ansible» далее в этой главе для получения дополнительной информации.)

Чтобы узнать больше о конкретном модуле, перейдите на сайт Ansible (выберите раздел Modules на странице docs.ansible.compage) или используйте команду `ansible-doc`. Например, чтобы прочитать о том, как применить модуль `copy` для копирования файлов в удаленное место, введите следующее:

```
# ansible-doc copy
> COPY    (/usr/lib/python3.7/site-packages/ansible/modules/files/
copy.py)
        The 'copy' module copies a file from the local or
        remote machine to a location on the remote machine...
```

Большинство модулей имеют возвращаемые значения и предоставляют информацию о результатах своих действий. Общие возвращаемые значения включают логические типы, указывающие, была ли задача не выполнена (`failed`), пропущена (`skipped`) или изменена (`changed`).

Роли, импорты и вложения

По мере роста вашей коллекции файлов playbooks вы можете захотеть разбить их на более мелкие части, которые можно включить в несколько файлов playbooks. Можете разделить большой файл playbook на отдельные многоразовые файлы, а затем добавить их в основной файл файлов playbooks, используя *вложения* и *импорты*, охватывающие больший функционал, чем задачи: модули, переменные и обработчики.

Дополнительные сведения о применении вложений, импортов и ролей есть в разделе Creating Reusable Playbooks на странице docs.ansible.com/ansible/latest/user_guide/playbooks_reuse.html.

Развертывание с помощью системы Ansible

Чтобы начать использовать Ansible, необходимо выполнить развертывание веб-службы на множестве хостов. В следующем примере показано, как после установки Ansible создать файлы inventories и playbooks, необходимые для развертывания этой службы. Затем мы рассмотрим, как применять `ansible-playbook` для фактического развертывания файла playbook.

Подготовка к развертыванию

Для начала я создал четыре виртуальные машины со следующими именами:

```
ansible     Использовался как контрольный узел Ansible
host01      Первый целевой узел
host02      Второй целевой узел
host03      Третий целевой узел
```

Затем выполнил следующие действия, чтобы подготовиться к использованию этих хостов с системой Ansible.

1. Установил Fedora на каждой из виртуальных машин (система RHEL тоже подойдет).
2. Для каждого из трех целевых узлов (`host01`, `host02` и `host03`) сделал следующее:
   - запустил службу SSH (при необходимости она открывает TCP-порт 22) в узле управления Ansible;
   - создал учетную запись пользователя, не являющегося суперпользователем. Позднее, когда вы будете применять playbooks, добавьте параметр `--ask-become-pass`, чтобы получать запрос на ввод пароля для расширения привилегий;
   - установил пароль для этого пользователя.

При запуске Ansible я задействую обычную учетную запись пользователя для подключения к каждой системе, а затем включаю привилегии суперпользователя с помощью команды `sudo`.

Настройка SSH-ключей для каждого узла

Войдите в управляющий узел (`ansible`) и убедитесь, что он работает с тремя вашими узлами. Либо убедитесь, что можете связаться с хостами через DNS-сервер, либо добавьте их в файл `/etc/hosts` на контрольном узле. Затем настройте ключи для доступа к этим узлам, например, так.

1. Как суперпользователь добавьте в файл `/etc/hosts` IP-адрес и имя для каждого узла, на котором хотите развернуть свои Ansible playbooks:

   ```
   192.168.122.154  host01
   192.168.122.94   host02
   192.168.122.189  host03
   ```

2. Все еще находясь в системе `ansible`, сгенерируйте `ssh`-ключи, чтобы создать связь без пароля с каждым хостом. Вы можете запустить эту и более поздние команды Ansible как обычный пользователь в хост-системе `ansible`:

```
$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/joe/.ssh/id_rsa): <ENTER>
Created directory '/home/joe/.ssh'.
Enter passphrase (empty for no passphrase): <ENTER>
Enter same passphrase again:
Your identification has been saved in /home/joe/.ssh/id_rsa.
Your public key has been saved in /home/joe/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:Wz63Ax1UdZnX+qKDmefSAZc3zoKS791hfaHy+usRP7g joe@ansible
The key's randomart image is:
+---[RSA 3072]----+
|              ...*|
|             .  o+|
|           . . ..|
|          . + +  |
|        S..= * + |
|        o+o + O.o|
|        .ooB.Bo+o|
|          *+O+o.o|
|         ..=BEo  |
+----[SHA256]-----+
```

3. С помощью команды `ssh-copy-id` скопируйте свой открытый ключ в корневую учетную запись на каждом хосте. В следующем цикле `for` выполняется копирование пароля пользователя на все три хоста:

```
$ for i in 1 2 3; do ssh-copy-id joe@host0$i; done
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed:
 "/home/joe/.ssh/id_rsa.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the
new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed
-- if you are prompted now it is to install the new keys
joe@host01's password: <пароль>

Number of key(s) added: 1
Now try logging into the machine, with: "ssh 'joe@host01'"
and check to make sure that only the key(s) you wanted were added.

/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed:
 "/home/joe/.ssh/id_rsa.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the
new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed
-- if you are prompted now it is to install the new keys

joe@host02's password: <пароль> .. .
```

Далее необходимо установить пакет `ansible` на узел управления (`ansible`). С этого момента вся работа выполняется из узла управления.

Установка системы Ansible

Пакеты программного обеспечения Ansible доступны для RHEL, Fedora, Ubuntu и других дистрибутивов Linux. Поскольку файлы playbooks Ansible запускаются с управляющего узла, нет необходимости устанавливать программное обеспечение Ansible на других узлах, с которыми будет работать система.

Итак, начнем с установки пакета `ansible` на RHEL, Fedora, Ubuntu или другой системе Linux, которую вы хотите использовать в качестве контрольного узла. Этот узел должен иметь возможность подключаться к службе SSH, работающей на узлах хоста, на которых вы хотите осуществить развертывание.

Установите пакет `ansible` одним из следующих способов.

- В дистрибутиве RHEL 8:

```
# subscription-manager repos \
    --enable ansible-2.9-for-rhel-8-x86_64-rpms
# dnf install ansible -y
```

- В дистрибутиве Fedora:

```
# dnf install ansible -y
```

- В дистрибутиве Ubuntu:

```
$ sudo apt update
$ sudo apt install software-properties-common
$ sudo apt-add-repository --yes --update ppa:ansible/ansible
$ sudo apt install ansible
```

Установив Ansible, можно создавать файлы inventories, которые предоставляют целевые объекты для файлов playbooks.

Создание файлов inventories

Простой файл inventory может состоять из имени, представляющего цель для файла playbook, и хост-систем, связанных с этим именем. Для начала приведу пример inventory, который содержит три группы статических хостов:

```
[ws]
host01
host02
host03

[newyork]
host01

[houston]
host02
host03
```

Добавьте эти строки в файл `/etc/ansible/hosts`, чтобы сделать их доступными при выполнении команд и файлов playbooks в Ansible.

Хотя процедура просто развертывается на множестве хостов в группе `ws`, две другие группы показывают, как вы можете настроить playbooks для отдельных задач в зависимости от расположения узлов (`newyork` и `houston`).

Аутентификация на хостах

Чтобы убедиться, что вы можете получить доступ к каждому хосту из системы Ansible, настройте службу `ssh` для каждого хоста. Пароль вводить не нужно:

```
$ ssh joe@host01
Last login: Wed Feb  5 19:28:39 2020 from 192.168.122.208
$ exit
```

Повторите команду для каждого хоста.

Создание файла playbook

Файл playbook устанавливает и запускает программное обеспечение веб-сервера на хостах, определенных ранее в группе `ws`. Аналогично, playbook удостоверяется, что программное обеспечение брандмауэра установлено и работает и что порт 80 (http-порт) открыт в брандмауэре, обеспечивая доступ к веб-серверу. Я добавил в файл `simple_web.yaml` следующее содержимое:

```
---
- name: Create web server
  hosts: ws
  remote_user: joe
  become_method: sudo
  become: yes
  tasks:
  - name: Install httpd
    yum:
      name: httpd
      state: present
  - name: Check that httpd has started
    service:
      name: httpd
      state: started
  - name: Install firewalld
    yum:
      name: firewalld
      state: present
  - name: Firewall access to https
    firewalld:
      service: http
```

```
      permanent: yes
      state: enabled
  - name: Restart the firewalld service to load in the firewall
changes
    service:
      name: firewalld
      state: restarted
```

Три дефиса в начале файла playbook — `simple_web.yaml` — указывают на начало содержимого YAML в нем. Рассмотрим остальные части файла.

- `name`. Скрипт `play` называется `Create web server`.
- `hosts`. Применяет этот файл inventory к хостам в группе `ws`.
- `remote_user`. Аккаунт обычного пользователя, который используется для аутентификации во всех удаленных системах. Это делается для того, чтобы обезопасить систему и не позволить корневой вход в удаленную систему.
- `become`. Включение этой функции (`yes`) дает команду Ansible стать другим пользователем, не `remote_user`, для запуска модулей в задаче.
- `become_method`. Выбирает функцию, которую нужно применить, чтобы расширить привилегии (`sudo`).
- `become_user`. Устанавливает, какой пользователь должен пройти аутентификацию (`root`).
- `tasks`. Запускает раздел, содержащий задачи.
- `name`. Название, данное задаче. В первом случае это `Install httpd`, затем `Check that httpd has started` и т. д. Следующая строка начинается с имени модуля (`yum service firewalld` и т. д.).

Модуль `yum` говорит системе, что нужно проверить, есть ли пакет `httpd` (`present`), а если нет, то установить его.

`service` проверяет, работает ли демон `httpd` (`started`). Если служба `httpd` не запущена, Ansible запускает ее.

Модуль `yum` говорит системе, что нужно проверить, есть ли пакет `firewalld` (`present`), а если нет, то установить его.

`firewalld` делает порт для службы `http` (TCP 80) доступным (`enabled`) и постоянным (`permanent: yes`) через брандмауэр.

Модуль `service` перезапускает службу `firewalld` (`restarted`), чтобы включить доступ к новому порту брандмауэра службы `http`.

Запуск файла playbook

Используйте команду `ansible-playbook` для запуска playbook. Чтобы протестировать playbook перед запуском в реальном времени, примените параметр `-C`. Чтобы увидеть больше деталей в выводе (по крайней мере пока тестируете файл), добавьте параметр `-v`.

Имейте в виду: если вы запускаете playbook с параметром `-C`, нельзя полностью протестировать файл, чтобы убедиться в его корректности. Причина в следующем: более позднее действие может потребовать, чтобы более раннее действие завершилось, прежде чем он может быть выполнен. В этом примере пакет `httpd` должен быть установлен до запуска службы `httpd`.

Вот пример запуска Ansible playbook с подробным выводом:

```
$ ansible-playbook -v simple_web.yaml
Using /etc/ansible/ansible.cfg as config file

PLAY [Create web server] ************************************

TASK [Gathering Facts] **************************************
ok: [host03]
ok: [host02]
ok: [host01]

TASK [Install httpd]
*************************************************************
changed: [host01] => {"changed": true, "msg": "", "rc": 0,
    "results": ["Installed: httpd", ...
changed: [host02] => {"changed": true, "msg": "", "rc": 0,
    "results": ["Installed: httpd", ...
changed: [host03] => {"changed": true, "msg": "", "rc": 0,
    "results": ["Installed: httpd", ...

TASK [Check that httpd has started]
*************************************************************
changed: [host03] => {"changed": true, "name": "httpd",
    "state": "started", "status":
changed: [host02] => {"changed": true, "name": "httpd",
    "state": "started", "status": ...
changed: [host01] => {"changed": true, "name": "httpd",
    "state": "started", "status": ...
...
TASK [Install firewalld]*************************************
changed: [host03] => {"changed": true, "msg": "", "rc": 0, "results":
    ["Installed: firewalld", "Installed: python3-decorator...
changed: [host02] => {"changed": true, "msg": "", "rc": 0, "results":
    ["Installed: firewalld", "Installed: python3-decorator...
changed: [host01] => {"changed": true, "msg": "", "rc": 0, "results":
  ["Installed: firewalld"...

TASK [Firewall access to https]***********************************
****
ok: [host03] => {"changed": false, "msg": "Permanent operation,
    (offline operation: only on-disk configs were altered)"}
ok: [host02] => {"changed": false, "msg": "Permanent operation,
    (offline operation: only on-disk configs were altered)"}
ok: [host01] => {"changed": false, "msg": "Permanent operation,
    (offline operation: only on-disk configs were altered)"}
```

```
PLAY RECAP *********************************************************
host01: ok=6 changed=4 unreachable=0 failed=0 skipped=0 rescued=0
ignored=0
host02: ok=6 changed=4 unreachable=0 failed=0 skipped=0 rescued=0
ignored=0
host03: ok=6 changed=4 unreachable=0 failed=0 skipped=0 rescued=0
ignored=0
```

Выходные данные `ansible-playbook` пошагово проходят через каждую задачу. Первая задача (`Gathering Facts`) показывает, что все три хост-системы в inventory группы `ws` доступны. В выводе не показано только то, что команда использует учетные данные для подключения ко всем системам, а затем повышает привилегии этого пользователя до привилегий суперпользователя перед выполнением каждой последующей задачи.

Задача `Install httpd` проверяет, установлен ли пакет `httpd` на каждом хосте. Если это не так, Ansible устанавливает пакет вместе с любыми зависимыми пакетами. Затем Ansible проверяет состояние службы `httpd` на каждом хосте и, если она не запущена, запускает ее.

После этого на каждом хосте проверяется, установлен ли пакет `firewalld`, и если его нет, то система устанавливает его. Затем Ansible добавляет правило брандмауэра к каждому хосту, чтобы разрешить доступ к службе `http` (TCP-порт 80), и делает этот параметр постоянным.

Затем `PLAY RECAP` выводит результаты выполнения всех задач. В выводе видно, что все шесть задач на всех хостах выполнены. Если задача не выполняется, то система ее указывает в выводе.

Можете повторно запустить этот файл playbook, если кажется, что что-то вышло из строя или вы внесли в него изменения. Можете также воспользоваться им позже для развертывания playbook на разных системах.

Система Ansible хорошо развертывает несколько задач в playbook, а также может использоваться для одноразовых действий. В следующем разделе я покажу, как выполнить некоторые специальные ad-hoc-команды Ansible для запроса и дальнейшего изменения хостов, которые мы только что развернули.

Запуск ad-hoc-команд в системе Ansible

Ansible позволяет выполнить на своих узлах разовые задачи с помощью специальных *ad-hoc-команд*. Таким образом можно напрямую вызвать модуль из командной строки Ansible и заставить его действовать как файл inventory. Задачи могут включать:

- установку программных пакетов RPM;
- управление учетными записями пользователей;
- копирование файлов на узлы и с них;
- изменение прав доступа к файлу или каталогу;
- перезагрузку узла.

Как и при запуске playbook, выполнение специальных ad-hoc-команд фокусируется на достижении желаемого состояния. Команда ad-hoc принимает задачу, выясняет, что запрашивается, и делает то, что нужно, чтобы достичь запрошенного состояния. Чтобы опробовать работу таких команд в Ansible, задействуйте файл inventory `ws`, созданный ранее.

Примеры использования команд ad-hoc

Запуская команду ad-hoc, вы выполняете действия с помощью модуля Ansible. *Командный* модуль применяется по умолчанию, если не указан другой модуль. Используя модуль, вы указываете, какие команды и параметры нужно запустить на группе узлов в качестве одноразового действия.

Проверьте, что файл inventory запущен и работает. В примере видно, что в файле inventory `ws` все хосты работают:

```
$ ansible ws -u joe -m ping
host03 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false,
    "ping": "pong"
}
host02 | SUCCESS => { ...
host01 | SUCCESS => { ...
```

Вы можете узнать, работает ли служба `httpd` на хостах в inventory, проверив состояние этой службы с помощью команды `ansible` следующим образом:

```
$ ansible ws -u joe -m service \
     -a "name=httpd state=started" --check
host02 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false,
    "name": "httpd",
    "state": "started",
    "status": { ...
host 01 | SUCCESS => { ...
```

В данный момент на веб-серверах нет содержимого. Чтобы добавить файл `index.html`, содержащий текст `Hello from your web server!`, для всех хостов в inventory, вы можете запустить следующую команду (введите пароль суперпользователя при появлении запроса):

```
$ echo "Hello from your web server!" > index.html
$ ansible ws -m copy -a \
    "src=./index.html dest=/var/www/html/ \
```

```
    owner=apache group=apache mode=0644" \
    -b --user joe --become --ask-become-pass
BECOME password: *********
host01 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": true,
    "checksum": "213ae4bb07e9b1e96fbc7fe94de372945a202bee",
    "dest": "/var/www/html/index.html",
    "gid": 48,
    "group": "apache",
    "md5sum": "495feb8ad508648cfafcf69681d94f97",
    "mode": "0644",
    "owner": "apache",
    "secontext": "system_u:object_r:httpd_sys_content_t:s0",
    "size": 52,
    "src": "/home/joe/.ansible/tmp/ansible-tmp-1581027374.649223-
    29961128730253/source",
    "state": "file",
    "uid": 48
host02 | CHANGED => { ...
host03 | CHANGED => { ...
```

Из примера видно, что файл `index.html` создается владельцем `apache` (UID 48) и группой `apache` (GID 48) в каталоге `/var/www/html` на `host01`. Затем его копия сохраняется на `host02` и `host03`. Проверьте, что все работает, получив доступ к этому файлу с хоста `ansible` через веб-сервер с помощью команды `curl`:

```
$ curl host01
Hello from your web server!
```

Автоматизация задач с помощью Ansible Tower Automation Framework

Запуск Ansible playbook и команд отлично подходит для автоматизации и последующего изменения наборов хостов, но для создания полностью управляемой структуры можно пойти и дальше. Используя программу Ansible Tower, можно расширить систему развертывания Ansible.

Утилита *Ansible Tower* предоставляет веб-интерфейс для управления всей вашей ИТ-инфраструктурой с помощью Ansible playbook и других компонентов. Она позволяет сохранить все ресурсы Ansible в одном месте и получать уведомления. Утилита также дает возможность управлять различными административными ролями на предприятии. Интерфейс Ansible Tower позволяет легко и непрерывно обновлять подготовленные ресурсы. Вместо того чтобы запоминать параметры командной строки, вы можете просто нажать кнопку, чтобы настроить и запустить

в работу задачи Ansible. Управление ресурсами реализуется через графический интерфейс, а планирование задач осуществляется интуитивно понятными визуальными способами.

Интерфейс REST API можно установить вместе с утилитой Ansible Tower, он помогает встроить существующие инструменты инфраструктуры в Ansible. Таким образом, можно просто продолжить работу с уже начатыми процессами, но управлять ими с помощью Ansible. Чтобы узнать больше об утилите Ansible Tower, перейдите на сайт Ansible Tower (www.ansible.com/products/tower).

Резюме

Система Ansible предоставляет уникальный язык форматирования и набор инструментов для автоматизации многих задач, которые вы изучили в других частях этой книги.

Зная, как создать Ansible playbook, вы можете настроить нужную конфигурацию, а затем легко развернуть ее в одной или нескольких хост-системах.

С помощью Ansible playbook вы определяете точное состояние приложения и окружающих компонентов, а затем применяете его к хост-системам Linux, сетевым устройствам или другим целевым объектам. Можете сохранить эти файлы playbooks и повторно использовать их для получения аналогичных результатов в других системах или адаптировать для новых результатов.

Ansible может применять специальные команды ad-hoc для обновления систем. Из командной строки `ansible` вы можете добавлять пользователей, копировать файлы, устанавливать программное обеспечение или делать почти все, что можно делать с файлом playbook. С помощью этих команд можно быстро применить набор изменений на нескольких хостах или отреагировать на проблему, требующую быстрого исправления на множестве хостов.

В этой главе мы узнали о различных компонентах системы Ansible, а также создали файл playbook для развертывания простого веб-сервера. Затем запустили несколько специальных команд ad-hoc для изменения систем, в которых был развернут созданный файл playbook.

Упражнения

Эти упражнения позволяют попробовать установить Ansible в системе, где вам нужно будет создать свой первый файл Ansible playbook и выполнить несколько ad-hoc-команд Ansible. Выполнять их стоит в системе Fedora или Red Hat Enterprise Linux (хотя некоторые упражнения сработают и в других системах Linux).

Несмотря на то что Ansible предназначен для развертывания задач в удаленных системах, упражнения здесь просто позволят поработать с файлом playbook и несколькими командами в одной системе. Если затрудняетесь с решением заданий,

воспользуйтесь ответами, приведенными в приложении Б (хотя Linux позволяет решать задачи разными способами).

1. Установите Ansible в систему Fedora или RHEL.
2. Добавьте привилегию команды `sudo` для того аккаунта, который будете использовать для выполнения этих упражнений.
3. Создайте файл Ansible playbook (назовите его `my_playbook.yaml`), который включает в себя следующее:

```
---
- name: Create web server
  hosts: localhost
  tasks:
  - name: Install httpd
    yum:
      name: httpd
      state: present
```

4. Запустите команду `ansible-playbook` в файле `my_playbook.yaml` в режиме проверки, чтобы выяснить, нет ли проблем с заполнением playbook (*подсказка*: есть).
5. Измените файл `my_playbook.yaml` и повысьте привилегии, чтобы задачи запускались от имени суперпользователя.
6. Запускайте `ansible-playbook` до тех пор, пока пакет `httpd` не будет успешно установлен в системе.
7. Измените файл `my_playbook.yaml` снова, чтобы запустить службу `httpd` и установить ее так, чтобы она запускалась каждый раз при загрузке системы.
8. Запустите команду `ansible`, которая проверяет, работает ли служба `httpd` на хосте `localhost`.
9. Создайте файл `index.html`, содержащий текст `@Web server is up@`, и примените команду `ansible`, чтобы скопировать этот файл в каталог `/var/www/html` на хосте `localhost`.
10. Используйте команду `curl` для просмотра содержимого файла, который вы только что скопировали на веб-сервер.

30 Развертывание приложений в контейнеры с помощью кластера Kubernetes

В этой главе

- Что такое Kubernetes.
- Работа с Kubernetes.
- Запуск Kubernetes Basics.
- Kubernetes и OpenShift.

Контейнеры Linux отделяют свои приложения от операционных систем, в которых работают. Правильно построенный контейнер будет содержать дискретный набор программного обеспечения, который можно транспортировать и эффективно запускать. Но на этом история не заканчивается. Настроив несколько контейнеров, необходимо задействовать их на такой платформе, как Kubernetes, которая позволяет осуществлять следующее.

- Группировать наборы контейнеров, чтобы сформировать более крупное приложение. Например, можно развернуть совместно веб-сервер, базу данных и средства мониторинга.
- Масштабировать контейнеры по мере необходимости. На самом деле должна быть возможность масштабировать каждый компонент более крупного приложения индивидуально, не затрагивая все сразу.
- Устанавливать состояние приложения, а не просто запускать его. Это означает, что вместо того, чтобы просто запустить контейнер, вы можете сообщить системе: «Запустите три копии контейнера *X* и, если одна из них выйдет из строя, обязательно запустите другую на замену».

- Восстановить систему после того, как хост-компьютеры вышли из строя или перегрузились. Если хост, на котором запущен контейнер, выходит из строя, необходимо, чтобы контейнер быстро восстановился и запустился на другом хост-компьютере.
- Не волноваться по поводу инфраструктуры. Приложение подключается к нужным ему службам, и ему нет необходимости знать имена хостов, IP-адреса или номера портов, связанные с этими службами.
- Обновлять свои приложения контейнеров без задержки работы.

Платформа Kubernetes выполняет все эти и многие другие функции. Ранее на рынке имелись и другие платформы для оркестрации контейнеров, например Mesos и Docker Swarm, однако Kubernetes стал бесспорным лидером в оркестрации и развертывании контейнерных приложений, а также управлении ими.

В этой главе мы познакомимся с платформой Kubernetes и платформой корпоративного качества Kubernetes под названием OpenShift. Лучший способ изучить Kubernetes — запустить кластер Kubernetes и выполнить команды развертывания контейнерного приложения. Перед этим необходимо понять, что такое кластер Kubernetes и какие компоненты нужны для развертывания приложения в нем.

Что такое Kubernetes

Кластер Kubernetes состоит из главного и рабочего узлов. Вы можете запускать все основные и рабочие службы в одной системе для личного применения. Например, с помощью Minikube, как описано далее в этой главе, можно запустить кластер Kubernetes с виртуальной машины на своем ноутбуке (kubernetes.io/docs/tasks/tools/install-minikube).

В производственной среде Kubernetes распространяется на несколько физических или виртуальных систем. Перечислю различные компоненты, о которых необходимо помнить при создании производственной инфраструктуры Kubernetes.

- **Мастер Kubernetes.** Главный узел, *мастер* (master node), управляет компонентами, работающими в кластере Kubernetes. Он управляет связью между компонентами, планирует запуск приложений на рабочих станциях, масштабирует приложения по мере необходимости и следит за тем, чтобы работало нужное количество контейнеров, распределенных в *подах* (pod). У вас должен быть по крайней мере один главный узел, но обычно их три или более. Это нужно для того, чтобы всегда был доступен по крайней мере один мастер.
- **Рабочие узлы.** *Рабочий узел* (worker node) — это место, где фактически выполняются развернутые контейнеры. Количество рабочих узлов зависит от необходимой нагрузки. Для производственной среды, скорее всего, понадобится более одного рабочего узла на случай, если один из них выйдет из строя или ему потребуется техническое обслуживание.

- **Хранилище.** Сетевое хранилище, которое позволяет контейнерам получать доступ к одному и тому же хранилищу независимо от узла, который их запускает.
- **Другие службы.** Чтобы интегрировать среду Kubernetes в существующий центр обработки данных, вам может потребоваться подключиться к существующим службам. Например, вы, вероятно, будете задействовать DNS-сервер компании для разрешения адресов на хосте, службу LDAP или Active Directory для аутентификации пользователей и сервер Network Time Protocol (NTP) для синхронизации времени.

В Kubernetes самый маленький модуль, с помощью которого можно развернуть контейнер, называется *подом* (pod). Модуль может содержать один или несколько контейнеров, а также описывающие их метаданные. Под может содержать только один контейнер (иногда — и более одного). Например, в модуле может содержаться *sidecar-контейнер*, предназначенный для мониторинга службы, работающей в основном контейнере пода.

Мастер-узлы

Мастер-узел Kubernetes управляет деятельностью кластера Kubernetes. Мастер-узлы контролируют всю его деятельность с помощью набора служб. Центральным элементом мастера Kubernetes является API-сервер (`kube-apiserver`), который принимает объектные запросы. Связь между всеми узлами кластера поддерживается через сервер API.

Когда мастер Kubernetes получает *объект*, например запрос на запуск определенного количества модулей, планировщик Kubernetes (`kube-scheduler`) находит доступные узлы для запуска каждого модуля и планирует их запуск на этих узлах. Чтобы убедиться, что все объекты остаются в заданном состоянии, контроллеры Kubernetes (`kube-controller-manager`) работают непрерывно. Они проверяют, что пространства имен существуют, определенные учетные записи служб доступны, нужное количество реплик запущено и определенные конечные точки активны.

Рабочие узлы

В основе каждого *рабочего узла* Kubernetes лежит служба kubelet. Она регистрирует свой рабочий узел на сервере API. Затем сервер API направляет kubelet выполнять запуск контейнера, который запрашивается с сервера API через PodSpec, и убедиться, что он исправен и продолжает работать. Другая служба, которая работает на каждом узле, — это *среда контейнеров* (container engine, *runtime*). Первоначально служба `docker` была самой популярной средой контейнеров, используемой для запуска контейнеров, управления ими и их удаления в соответствии с требованиями PodSpec. Однако теперь доступны и другие среды, например CRI-O (`cri-o.io`), которая применяется с коммерческими платформами Kubernetes, такими как OpenShift.

Рабочие узлы должны быть как можно более универсальными, чтобы их можно было просто развернуть на новом узле, когда потребуется дополнительная емкость, и он был бы настроен для обработки большинства запросов на запуск контейнеров. Однако есть способы, при которых контейнер может не подойти для запуска на определенном узле. Допустим, модуль может запросить запуск на узле, имеющем минимальный объем доступной памяти и процессора, или он может запросить запуск на узле, на котором работает связанный контейнер. Точно так же, если для запуска модуля требуется что-то особенное, например определенная компьютерная архитектура, аппаратное обеспечение или операционная система, существуют способы планирования модулей для рабочих узлов, которые отвечают этим потребностям.

Приложения Kubernetes

В Kubernetes приложения управляются путем определения объектов API, которые задают состояние ресурсов в кластере. Например, вы можете создать в файле YAML объект `Deployment`, определяющий модули. Каждый из таких модулей запускает один или несколько контейнеров, а также пространство имен и количество `replicas` каждого модуля. Этот объект также может определять открытые порты `ports` и любые тома `volumes`, смонтированные для каждого контейнера. *Мастер-узлы* Kubernetes отвечают на такие запросы и следят за тем, чтобы они выполнялись на *рабочих* узлах Kubernetes.

Kubernetes использует концепцию *служб*, чтобы отделить местоположение приложения от его фактического IP-адреса и номера порта. Имя службы для набора модулей, предоставляющих эту службу, не должно раскрывать точное местоположение каждого модуля и быть известным за пределами кластера. Вместо этого Kubernetes направляет запрос на нужную службу в доступный модуль.

IP-адреса, связанные с активными модулями, по умолчанию не могут быть напрямую направлены извне кластера. Вы сами должны определить, как хотите предоставить доступ к службе, связанной с набором модулей за пределами кластера. Используя объект Service, можно предоставлять службы различными способами.

По умолчанию предоставление службы через `ClusterIP` и службу `type` делает ее доступной только для компонентов кластера. Чтобы предоставить доступ к службе за пределами кластера, вы можете использовать сервис NodePort, который создает модуль службы и делает его доступным через один и тот же назначенный Kubernetes порт на внешнем IP-адресе от каждого узла, на котором работает.

Третий метод — применять `LoadBalancer` для назначения внешнего фиксированного IP-адреса, который действует как балансировщик нагрузки для модулей, предоставляющих услугу. С помощью `LoadBalancer` внешний балансировщик нагрузки облака направляет трафик на серверные модули. Наконец, вы можете предоставить доступ к службе с помощью `ExternalName`, который связывает службу с определенной записью DNS CNAME.

Независимо от того, как вы предоставляете доступ к службе Kubernetes, когда есть запрос на нее, Kubernetes действует как маршрутизатор сообщений к набору модулей, которые предоставляют эту службу. Таким образом, поды (модули) могут подниматься и опускаться, не нарушая работу клиентов, использующих службу.

Интерфейсы Kubernetes

Kubernetes имеет интерфейсы командной строки и веб-консоли для доступа к кластеру Kubernetes. В этой главе представлены примеры инструментов командной строки. Команды — это `minikube`, которая используется для управления виртуальной машиной Kubernetes и перевода кластера вверх и вниз, и `kubectl` — универсальный инструмент управления кластером Kubernetes.

Начало работы с Kubernetes

Поскольку создание собственного производственного кластера Kubernetes требует определенной предусмотрительности, в этой главе мы сосредоточимся на нескольких простых способах быстрого запуска персонального кластера Kubernetes и доступа к нему. В частности, вот способы получить доступ к кластеру Kubernetes.

- **Сайт Kubernetes Tutorials.** Официальный сайт Kubernetes предлагает интерактивные учебные материалы с веб-интерфейсом, где вы можете запустить собственный кластер и попробовать работу Kubernetes. В Kubernetes Tutorials (kubernetes.io/docs/tutorials) рассматриваются основные темы, а также конфигурация, stateless-приложения и др.
- **Minikube.** С помощью инструмента Minikube можете запускать Kubernetes на своем компьютере. Система Linux, macOS или Windows, которая способна запускать виртуальные машины, может запустить виртуальную машину Minikube и кластер Kubernetes на ноутбуке или настольной системе в течение нескольких минут.
- **Docker Desktop.** Другой вариант (в книге не рассматривается) — это приложение Docker Desktop, которое позволяет включить предварительно настроенный кластер Kubernetes, который запускает главный и рабочий узлы на рабочей станции.

Перед тем как начать, я расскажу о некоторых руководствах Kubernetes и объясню их основные концепции. Вы можете следовать инструкциям, приводимым в руководстве, или выполнять те же команды на собственной системе Minikube. Далее я опишу, как получить доступ к кластеру Kubernetes одним из двух способов.

ПРИМЕЧАНИЕ

Если у вас подключена среда OpenShift, вы также можете выполнить большинство этих действий в OpenShift. В большинстве случаев можно применить команду kubectl, но обычно ее параметры и аргументы могут использоваться командой oc в OpenShift.

Доступ к кластеру Kubernetes

Далее я расскажу, как получить доступ к кластеру Kubernetes либо через руководство Kubernetes, либо установив и запустив Minikube.

Доступ с помощью руководства

Чтобы запустить интерактивное руководство Kubernetes, перейдите в браузере на страницу kubernetes.io/docs/tutorials/kubernetes-basics/create-cluster/cluster-interactive.

На рис. 30.1 показано начало обучения основам Kubernetes.

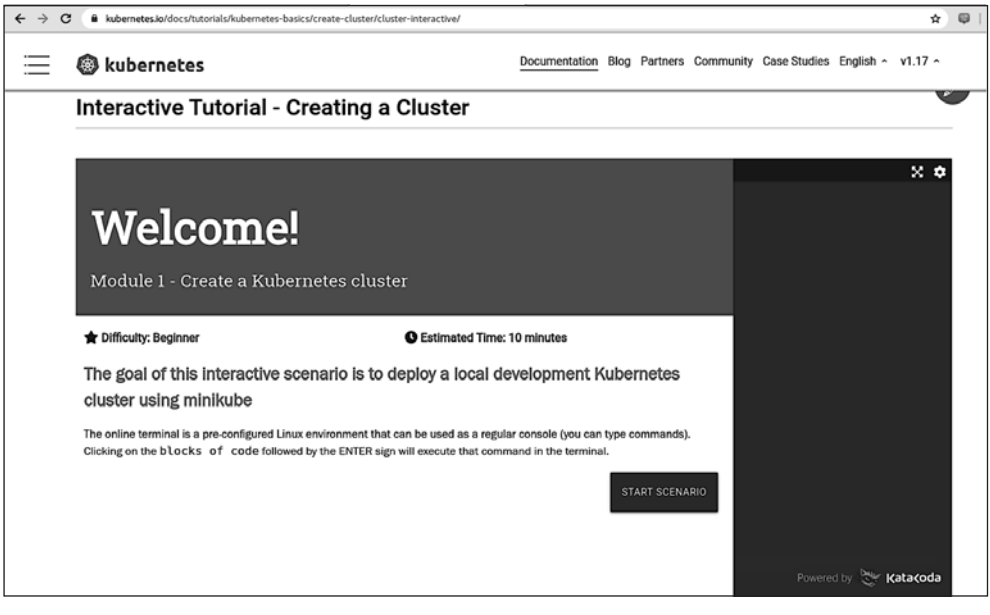


Рис. 30.1. Пошаговое руководство Kubernetes

На этом этапе вы можете следовать инструкциям, данным в руководстве. Поскольку в руководстве запускается живой кластер, можно использовать этот интерфейс и для других команд.

Запуск Minikube

Для запуска Minikube на персональном компьютере требуется выполнить несколько действий.

- Компьютер нужно настроить как гипервизор, чтобы он мог запускать виртуальные машины Minikube.

- Необходимо установить команду `kubectl` (используется для доступа к кластеру и работы с ним) и саму виртуальную машину Minikube.

Чтобы получить инструкцию по установке для систем Linux, macOS и Windows, перейдите на страницу kubernetes.io/docs/tasks/tools/install-minikube.

Вы можете установить Minikube от имени суперпользователя, но позже его нужно будет запустить из обычной учетной записи пользователя. Установка Minikube в Fedora, RHEL, Ubuntu или другую систему Linux (обратитесь к справочной странице `install-minikube`, если что-либо в установке изменилось) выполняется так.

1. **Установите пакет команды kubectl.** Установите ту версию команды `kubectl`, которая соотносится с версией Kubernetes в вашей системе Minikube. Последние версии команд `kubectl` и `minikube` отслеживают обновления. Введите следующее (все в одной строке):

```
# curl -LO \
https://storage.googleapis.com/kubernetes-release/release/ `curl \ -s
https://storage.googleapis.com/kubernetes-release/release/ stable.txt \ `/
bin/linux/amd64/kubectl
```

2. **Скопируйте kubectl в каталог bin.** Скопируйте команду `kubectl` в доступный каталог `bin` и сделайте ее исполняемой, например:

```
# mkdir /usr/local/bin
# cp kubectl /usr/local/bin
# chmod 755 /usr/local/bin/kubectl
```

3. **Настройте гипервизор.** Настройте свою систему Linux как гипервизор. Для KVM выполните действия, описанные в подразделе «Настройка гипервизоров» в главе 27.
4. **Загрузите команду minikube.** Загрузите исполняемый файл `minikube` и введите следующее (в одной строке):

```
# curl -Lo minikube \
https://storage.googleapis.com/minikube/releases/latest/minikubelinux-amd64
\ && chmod +x minikube
```

5. **Установите Minikube.** Введите следующее:

```
# install minikube /usr/local/bin/
```

6. **Запустите Minikube.** От имени обычного пользователя введите следующие команды для идентификации драйвера, если задействуете гипервизор KVM (если у вас другой гипервизор, см. страницу minikube.sigs.k8s.io/docs/ reference/drivers):

```
$ minikube config set vm-driver kvm2
$ minikube start --vm-driver=kvm2
```

7. **Начните использовать Minikube.** Теперь можно начать применять Minikube, выполнив команды `minikube` и `kubectl`. Далее рассмотрим, как это сделать.

Запуск руководства Kubernetes Basics

Руководство Kubernetes Basics описывает набор команд, с помощью которых начинается знакомство с Kubernetes: kubernetes.io/docs/tutorials/kubernetes-basics/create-cluster/cluster-interactive.

Далее мы познакомимся с первыми пятью модулями руководства Kubernetes Basics.

Если используете интерактивную систему непосредственно со страниц руководства Kubernetes, продолжайте там же и запустите Minikube (`minikube start`). Если вы задействуете Minikube с виртуальной машины, уже работающей на ноутбуке, также можете выполнить следующие шаги, так как оба варианта применяют Minikube.

Информация о кластере

Запустите следующие команды, чтобы получить основную информацию о своем кластере.

1. **Версия Minikube.** Чтобы узнать версию `minikube`, введите следующее:

```
$ minikube version
minikube version: v1.7.2
commit: 50d543b5fcb0e1c0d7c27b1398a9a9790df09dfb
```

2. **Информация о кластере.** Чтобы увидеть URL-адрес, с которого доступны службы Kubernetes master и DNS, введите следующее:

```
$ kubectl cluster-info
Kubernetes master is running at https://192.168.39.150:8443
KubeDNS is running at
https://192.168.39.150:8443/api/v1/namespaces/kube-system/
services/kube-dns:dns/proxy
To further debug and diagnose cluster problems, use
'kubectl cluster-info dump'.
```

3. **Информация об узлах.** Чтобы увидеть количество запущенных узлов (в Minikube только один мастер-узел) и их состояние, введите следующее:

```
$ kubectl get nodes
NAME     STATUS   ROLES  AGE   VERSION
minikube Ready    master 23m   v1.17.2
```

4. **Версии кластера и клиента.** Чтобы перечислить версии клиента `kubectl` и кластера Kubernetes (и убедиться, что они находятся в пределах одной версии), введите следующее:

```
$ kubectl version
Client Version: version.Info{Major:"1", Minor:"17",
GitVersion:"v1.17.2",
GitCommit:"59603c6e503c87169aea6106f57b9f242f64df89",
GitTreeState:"clean", BuildDate:"2020-01-18T23:30:10Z",
GoVersion:"go1.13.5", Compiler:"gc", Platform:"linux/amd64"}
```

```
Server Version: version.Info{Major:"1", Minor:"17",
GitVersion:"v1.17.2",
GitCommit:"59603c6e503c87169aea6106f57b9f242f64df89",
GitTreeState:"clean", BuildDate:"2020-01-18T23:22:30Z",
GoVersion:"go1.13.5", Compiler:"gc", Platform:"linux/amd64"}
```

Развертывание приложения Kubernetes

Запросы на запуск контейнерных приложений (в виде подов) и управление ими в кластере Kubernetes называются *развертыванием*. После создания развертывания кластер Kubernetes должен убедиться, что запрошенные поды всегда работают. Делает это он так:

- готовится к созданию развертывания через сервер API;
- запрашивает у планировщика запуск необходимых контейнеров из каждого пода на доступных рабочих узлах;
- наблюдает за подами, чтобы убедиться, что они продолжают работать в соответствии с запросом;
- запускает новый экземпляр пода (на том же или другом узле) в случае сбоя модуля (например, если контейнер перестает работать).

Далее показан пример создания простого развертывания из образа контейнера. В этом примере вы просто даете ему имя и определяете образ контейнера. Остальные параметры развертывания заполняются из значений по умолчанию.

1. **Создайте развертывание.** Чтобы запустить развертывание, которое извлекает контейнер `kubernetesbootcamp` с именем развертывания `kubernetes bootcamp`, введите следующее:

```
$ kubectl create deployment kubernetes-bootcamp \
      --image=gcr.io/google-samples/kubernetes-bootcamp:v1
deployment.apps/kubernetes-bootcamp created
```

2. **Перечислите развертывания.** Чтобы убедиться, что развертывание существует (а также имеет один запрошенный инстанс и один запущенный), введите следующее:

```
$ kubectl get deployments
NAME                  READY   UP-TO-DATE   AVAILABLE   AGE
kubernetes-bootcamp   1/1     1            1           4m38s
```

3. **Просмотрите сведения о развертывании.** Чтобы просмотреть подробные сведения о развертывании, введите следующие данные:

```
$ kubectl describe deployments kubernetes-bootcamp
Name:                   kubernetes-bootcamp
Namespace:              default
...
Replicas:   1 desired | 1 updated | 1 total | 1 available | 0
unavailable
...
```

```
Pod Template:
  Labels:  app=kubernetes-bootcamp
  Containers:
   kubernetes-bootcamp:
    Image:        gcr.io/google-samples/kubernetes-bootcamp:v1
    Port:         <none>
    Host Port:    <none>
    Environment:  <none>
    Mounts:       <none>
  Volumes:        <none>
...
```

В развертывании `kubernetes-bootcamp` обратите внимание на то, что устанавливается один инстанс (`replica`) модуля, связанного с развертыванием. Развертывание выполняется в текущем пространстве имен, которое имеет значение `default`. Обратите внимание также на то, что для модулей по умолчанию не открыты порты и не смонтированы тома.

Информация о подах (модулях) развертывания

После выполнения развертывания вы можете запросить информацию о созданном из него поде и предоставить API Kubernetes виртуальной машины своей локальной системе через прокси-службу.

1. **Предоставьте доступ к API Kubernetes локальной системе.** Чтобы открыть прокси-сервер из вашей системы к API Kubernetes, работающему в Minikube (`kubectl proxy`), введите следующее:

```
$ kubectl proxy
Starting to serve on 127.0.0.1:8001
```

2. **Запросите API Kubernetes.** Откройте второй терминал и запросите API Kubernetes, работающий на Minikube, введя следующее:

```
$ curl http://localhost:8001/version
{
 "major": "1",
 "minor": "17",
 "gitVersion": "v1.17.2",
 "gitCommit": "59603c6e503c87169aea6106f57b9f242f64df89",
 "gitTreeState": "clean",
 "buildDate": "2020-01-18T23:22:30Z",
 "goVersion": "go1.13.5",
 "compiler": "gc",
 "platform": "linux/amd64"
```

3. **Запросите информацию о поде.** Имя модуля, используемого в этом развертывании, — это `kubernetes-bootcamp`, за которым следует уникальная строка символов. Введите команды, приведенные далее, чтобы вывести имя пода, а затем перечислить описание этого модуля:

```
$ kubectl get pods
NAME                                  READY   STATUS   RESTARTS   AGE
kubernetes-bootcamp-69fbc6f4cf-njc4b  1/1     Running  0          12m
$ kubectl describe pod kubernetes-bootcamp-69fbc6f4cf-njc4b
Name:         kubernetes-bootcamp-69fbc6f4cf-njc4b
Namespace:    default
Priority:     0
Node:         minikube/192.168.39.150
...
Containers:
  kubernetes-bootcamp:
    Container ID:
docker://dd24fd43ff19d6cf12f5c759036cee74adcf2d0e2c55a42e...
    Image:          gcr.io/google-samples/kubernetes-bootcamp:v1
    Image ID:       docker-pullable://gcr.io/google-samples...
...
Events:
  Type    Reason     Age   From               Message
  ----    ------     ----  ----               -------
  Normal  Scheduled  14m   default-scheduler  Successfully assigned
default/kubernetes-bootcamp-69fbc6f4cf-njc4b to minikube
  Normal  Pulled     14m   kubelet, minikube  Container image
"gcr.io/google-samples/kubernetes-bootcamp:v1"
already present on machine
  Normal  Created    14m   kubelet, minikube  Created container
kubernetes-bootcamp
  Normal  Started    14m   kubelet, minikube  Started container
kubernetes-bootcamp
```

В выходных данных в примере видны имя пода, пространство имен, в котором он находится (`default`), и узел, на котором работает (`minikube/192.168.39.150`). В разделе `Containers` содержатся имя запущенного контейнера (`docker://dd24fd43ff1...`), образ, из которого он получен (`...kubernetes-bootcamp:v1`), и идентификатор образа для него. В разделе `Events`, начиная снизу, вы можете увидеть `kubelet` на узле `minikube`, который запускает и создает контейнер. Он проверяет образ и обнаруживает, что тот уже находится на узле. Затем он назначает модуль для запуска на этом узле.

4. **Подключитесь к поду.** Используйте команду `curl`, чтобы связаться с модулем и заставить его ответить на ваш запрос:

```
$ export POD_NAME=$(kubectl get pods -o go-template --template \
'{{range .items}}{{. metadata.name }}{{"\n"}}{{end}}') ; \
echo Name of the Pod: $POD_NAME
Name of the Pod: kubernetes-bootcamp-69fbc6f4cf-njc4b

$ curl \
http://localhost:8001/api/v1/namespaces/default/pods/$POD_NAME/
proxy/
Hello Kubernetes bootcamp!|Running on:kubernetes-bootcamp-
5b48cfdcbd-lf9t2|v=1
```

5. **Просмотрите журналы.** Чтобы просмотреть журналы любого контейнера, работающего внутри выбранного модуля, выполните следующую команду:

```
$ kubectl logs $POD_NAME
Kubernetes Bootcamp App Started At: 2020-02-13T21:29:21.836Z
| Running On:  kubernetes-bootcamp-5b48cfdcbd-lf9t2

Running On: kubernetes-bootcamp-5b48cfdcbd-lf9t2 | Total Requests:
1 | App Uptime: 34.086 seconds | Log Time: 2020-02-13T21:29:55.923Z
```

6. **Выполните команды в поде.** Используйте команду `kubectl exec` для запуска команд внутри модуля. Первая команда запускает `env` для просмотра переменных среды оболочки изнутри модуля, а вторая открывает оболочку внутри модуля, чтобы вы могли выполнить следующие команды:

```
$ kubectl exec $POD_NAME env
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
HOSTNAME=kubernetes-bootcamp-5b48cfdcbd-lf9t2
KUBERNETES_SERVICE_HOST=10.96.0.1
KUBERNETES_SERVICE_PORT=443
...
$ kubectl exec -ti $POD_NAME bash
root@kubernetes-bootcamp-5b48cfdcbd-lf9t2:/# date
Thu Feb 13 21:57:18 UTC 2020

kubernetes-bootcamp-5b48cfdcbd-lf9t2:/# ps -ef
UID        PID  PPID  C STIME TTY          TIME CMD
root         1     0  0 21:29 ?        00:00:00 /bin/sh -c node
server.js
root         6     1  0 21:29 ?        00:00:00 node server.js
root       115     0  0 21:55 pts/0    00:00:00 bash
root       123   115  0 22:01 pts/0    00:00:00 ps -ef

root@kubernetes-bootcamp-5b48cfdcbd-lf9t2:/# curl localhost:8080
Hello Kubernetes bootcamp!|Running on:kubernetes-bootcamp-5b48cfdcbd-lf9t2|v=1

root@kubernetes-bootcamp-5b48cfdcbd-lf9t2:/# exit
```

После запуска оболочки появляются выходные данные команд `date` и `ps`. Из вывода команды `ps` видно, что первый процесс, запущенный в контейнере (PID 1), — это скрипт `server.js`. После этого команда `curl` может успешно взаимодействовать с контейнером на локальном порте 0808.

Доступ к приложениям с помощью служб

Чтобы предоставить доступ к модулю `kubernetes-bootcamp`, описанному в примерах, с внешнего IP-адреса рабочего узла, можно создать объект NodePort. Вот один из способов сделать это.

1. **Проверьте, работает ли под.** Введите команду, приведенную далее, чтобы проверить, что модуль `kubernetes bootcamp` запущен:

```
$ kubectl get pods
NAME                                  READY   STATUS    RESTARTS   AGE
kubernetes-bootcamp-765bf4c7b4-fdl96  1/1     Running   0          26m
```

2. **Проверьте службы.** Введите приведенные далее команды, чтобы увидеть службы, работающие в пространстве имен `default`. Обратите внимание на то, что доступна только служба `kubernetes` и нет никакой службы, предоставляющей доступ к модулю `kubernetes-bootcamp` за пределами кластера:

```
$ kubectl get services
NAME         TYPE       CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP  10.96.0.1    <none>        443/TCP   31m
```

3. **Создайте службу.** Создайте службу, которая использует NodePort, чтобы сделать под доступным с IP-адреса на хосте по определенному номеру порта (8080). Например, введите команду

```
$ kubectl expose deployment/kubernetes-bootcamp \
    --type="NodePort" --port 8080
service/kubernetes-bootcamp exposed
```

4. **Просмотрите новую службу.** Введите приведенную далее команду, чтобы увидеть IP-адрес (10.96.66.230) и номер порта (8080), с которого служба становится доступной на хосте:

```
$ kubectl get services
NAME             TYPE         CLUSTER-IP   EXTERNAL-IP   PORT(S)     AGE
kubernetes       ClusterIP    10.96.0.1    <none>        443/TCP     33m
kubernetes-bootcamp NodePort  10.96.66.230 <none>        8080:32374/
TCP 5s
$ kubectl describe services/kubernetes-bootcamp
Name:                     kubernetes-bootcamp
Namespace:                default
Labels:                   app=kubernetes-bootcamp
Annotations:              <none>
Selector:                 app=kubernetes-bootcamp
Type:                     NodePort
IP:                        10.96.66.230
Port:                     <unset>  8080/TCP
TargetPort:               8080/TCP
NodePort:                 <unset>  30000/TCP
Endpoints:                172.17.0.6:8080
Session Affinity:         None
External Traffic Policy:  Cluster
```

5. **Назначьте порт для узла.** Чтобы получить порт, назначенный службе, и установить переменную `$NODE_PORT` в это значение, введите следующее:

```
$ export NODE_PORT=$(kubectl get services/kubernetes-bootcamp \
-o go-template='{{(index .spec.ports 0).nodePort}}')
$ echo NODE_PORT=$NODE_PORT
NODE_PORT=30000
```

6. **Проверьте доступ к службе.** Чтобы убедиться, что служба доступна из NodePort, используйте команду `curl` (применив IP-адрес для своего инстанса Minikube):

```
$ curl $(minikube ip):$NODE_PORT
Hello Kubernetes bootcamp!|Running on:kubernetes-bootcamp-
765bf4c7b4-fdl96|v=1
```

Метки для службы

Описанный далее процесс используется для добавления метки к существующей службе.

1. **Проверьте метку пода.** Пока что `kubernetes-bootcamp` — это единственная метка, присвоенная модулю. Чтобы убедиться в этом, введите следующее:

```
$ kubectl describe deployment
Name:                   kubernetes-bootcamp
Namespace:              default
CreationTimestamp:      Fri, 14 Feb 2020 05:43:49 +0000
Labels:                 run=kubernetes-bootcamp
Annotations:            deployment.kubernetes.io/revision: 1
```

2. **Добавьте другую метку.** Чтобы у модуля появилась еще одна метка (`v1`), получите имя пода и добавьте ее следующим образом:

```
$ export POD_NAME=$(kubectl get pods -o go-template --template \
'{{range .items}}{{. metadata.name }}{{"\n"}}{{end}}') ; \
echo Name of the Pod: $POD_NAME
Name of the Pod: kubernetes-bootcamp-765bf4c7b4-fdl96

$ kubectl label pod $POD_NAME app=v1
pod/kubernetes-bootcamp-765bf4c7b4-fdl96 labeled
```

3. **Проверьте метку и используйте ее.** Убедитесь, что метка `v1` назначена поду, а затем примените ее для отображения информации о нем:

```
$ kubectl describe pods $POD_NAME
Name:         kubernetes-bootcamp-765bf4c7b4-fdl96
Namespace:    default
Priority:     0
Node:         minikube/172.17.0.62
Start Time:   Fri, 14 Feb 2020 05:44:08 +0000
Labels:       app=v1
              pod-template-hash=765bf4c7b4
              run=kubernetes-bootcamp
$ kubectl get pods -l app=v1
NAME                                   READY   STATUS    RESTARTS
AGE
kubernetes-bootcamp-765bf4c7b4-fdl96   1/1     Running   0
60m
```

Удаление службы

Если вы закончили пользоваться службой, ее можно удалить. Следующий процесс удаляет доступ к службе из порта узла, но не удаляет само развертывание.

1. **Проверьте службу.** Убедитесь, что служба `kubernetes-bootcamp` все еще существует:

```
$ kubectl get services
NAME                 TYPE      CLUSTER-IP   EXTERNAL-IP PORT(S)
AGE
kubernetes           ClusterIP 10.96.0.1    <none>      443/TCP
63m
kubernetes-bootcamp NodePort  10.96.66.230 <none>  8   080:32374/
TCP 30m
```

2. **Удалите службу.** Используя имя метки, удалите службу:

```
$ kubectl delete service -l run=kubernetes-bootcamp
service "kubernetes-bootcamp" deleted
```

3. **Проверьте службу и развертывание.** Убедитесь, что служба удалена, но развертывание на месте:

```
$ kubectl get services
NAME                 TYPE      CLUSTER-IP   EXTERNAL-IP   PORT(S)
AGE
kubernetes           ClusterIP 10.96.0.1    <none>        443/TCP
64m
$ kubectl get deployment
NAME                  READY   UP-TO-DATE   AVAILABLE   AGE
kubernetes-bootcamp  1/1     1            1           65m
```

Масштабирование приложения

Одна из самых мощных функций Kubernetes — его способность масштабировать приложение по мере необходимости. Этот процесс начинается с развертывания `kubernetes-bootcamp`, которое запускает один модуль и масштабирует его, чтобы добавить дополнительные модули, работающие с использованием функции *ReplicaSet* и других средств предоставления приложению внешнего доступа.

1. **Найдите развертывание.** Перечислите информацию о развертывании *kubernetes-bootcamp* и обратите внимание на то, что оно настроено на активацию только одного набора реплик (`rs`):

```
$ kubectl get deployments
NAME                  READY   UP-TO-DATE   AVAILABLE   AGE
kubernetes-bootcamp   1/1     1            1           107s
$ kubectl get rs
NAME                              DESIRED   CURRENT   READY   AGE
kubernetes-bootcamp-5b48cfdcbd   1         1         1       3m4s
```

2. **Масштабируйте реплики.** Чтобы масштабировать развертывание до четырех наборов реплик, введите следующее:

```
$ kubectl scale deployments/kubernetes-bootcamp --replicas=4
deployment.extensions/kubernetes-bootcamp scaled
```

3. **Проверьте новые реплики.** Перечислите развертывания, чтобы убедиться, что теперь все четыре реплики готовы и доступны:

```
$ kubectl get deployments
NAME                  READY   UP-TO-DATE   AVAILABLE   AGE
kubernetes-bootcamp   4/4     4            4           8m44s
```

4. **Проверьте поды.** Теперь должны быть запущены четыре модуля `kubernetes-bootcamp`, каждый с собственным IP-адресом внутри кластера. Чтобы убедиться в этом, введите следующую команду:

```
$ kubectl get pods -o wide
NAME                          READY   STATUS    RESTARTS AGE    IP
   NODE      NOMINATED NODE   READINESS GATES
kubernetes-bootcamp-5b4... 1/1     Running   0        8m43s  172.18.0.4
   minikube  <none>           <none>
kubernetes-bootcamp-5b4... 1/1     Running   0        12s    172.18.0.8
   minikube  <none>           <none>
kubernetes-bootcamp-5b4... 1/1     Running   0        12s    172.18.0.6
   minikube  <none>           <none>
kubernetes-bootcamp-5b4..  1/1     Running   0        12s    172.18.0.7
   minikube  <none>           <none>
```

5. **Просмотрите сведения о развертывании.** Чтобы просмотреть подробные сведения о репликах в развертывании, введите следующие данные:

```
$ kubectl describe deployments/kubernetes-bootcamp
Name:                   kubernetes-bootcamp
Namespace:              default
...
Replicas:    4 desired | 4 updated | 4 total | 4 available | 0 unavailable
...
NewReplicaSet:   kubernetes-bootcamp-5b48cfdcbd (4/4 replicas created)
Events:
  Type    Reason             Age    From                   Message
  ----    ------             ----   ----                   -------
  Normal  ScalingReplicaSet  17m    deployment-controller  Scaled up
    replica set kubernetes-bootcamp-5b48cfdcbd to 1
  Normal  ScalingReplicaSet  9m25s  deployment-controller  Scaled up
    replica set kubernetes-bootcamp-5b48cfdcbd to 4
```

Проверка балансировщика нагрузки

Чтобы удостовериться, что трафик распределяется по всем четырем реплицированным модулям, используйте NodePort, а затем команду `curl`, убедившись, что несколько подключений к NodePort дают доступ к разным модулям.

1. **Перечислите сведения о службе.** Чтобы получить подробную информацию о службе `kubernetes-bootcamp`, введите следующую команду:

```
$ kubectl describe services/kubernetes-bootcamp
Name:            kubernetes-bootcamp
Namespace:       default
Labels:          run=kubernetes-bootcamp
Annotations:     <none>
Selector:        run=kubernetes-bootcamp
Type:            NodePort
IP:              10.99.183.8
Port:            <unset> 8080/TCP
TargetPort:      8080/TCP
NodePort:        <unset> 31915/TCP
Endpoints:       172.18.0.4:8080,172.18.0.6:8080,172.18.0.7:8080 +
1 more...
```

   Обратите внимание на IP-адрес и порт (172.18.0.4:8080, 172.18.0.4:8080 и т. д.), присвоенные каждому поду.

2. **Подключите NodePort.** Введите приведенную далее команду, чтобы установить `$NODE_PORT` в значение номера порта, назначенного службе:

```
$ export NODE_PORT=$(kubectl get services/kubernetes-bootcamp \
-o go-template='{{(index .spec.ports 0).nodePort}}')

$ echo NODE_PORT=$NODE_PORT
NODE_PORT=31915
```

3. **Запустите команду curl.** Выполните команду `curl` несколько раз, чтобы запросить службу. Если вы запустите ее несколько раз, то увидите, что она обращается к разным подам. Именно так можно понять, что балансировщик нагрузки работает:

```
$ curl $(minikube ip):$NODE_PORT
Hello Kubernetes bootcamp!|Running on:kubernetes-bootcamp-
5b48cfdcbd-9j4xp|v=1
```

Уменьшение масштаба приложения

Чтобы масштабировать количество наборов реплик ReplicaSets, определенных в развертывании, просто уменьшите количество реплик.

1. **Уменьшите количество реплик.** Введите приведенную далее команду, чтобы изменить количество реплик для развертывания на 2:

```
$ kubectl scale deployments/kubernetes-bootcamp –replicas=2
deployment.extensions/kubernetes-bootcamp scaled
```

2. **Проверьте развертывание.** Чтобы убедиться, что развертывание установлено на 2 и запущены только два модуля, введите следующее:

```
$ kubectl get deployments
NAME                  READY   UP-TO-DATE   AVAILABLE   AGE
kubernetes-bootcamp   2/2     2            2           52m
$ kubectl get pods -o wide
NAME                          READY   STATUS    RESTARTS AGE    IP
   NODE      NOMINATED NODE   READINESS GATES
kubernetes-bootcamp-5b4... 1/1     Running   0        8m43s
172.18.0.4
   minikube  <none>           <none>
kubernetes-bootcamp-5b4... 1/1     Running   0        12s
172.18.0.8
```

На этом этапе уже понятно, как вручную запрашивать свой кластер Kubernetes различными способами, а также запускать развертывания, модули и реплики и работать с ними. Чтобы продолжить работу с более продвинутыми руководствами Kubernetes, вернитесь на главную страницу Kubernetes Tutorials (kubernetes.io/docs/tutorials/). Я также рекомендую сайт Kubernetes By Example, где содержится дополнительная информация об использовании Kubernetes (kubernetesbyexample.com).

Корпоративная платформа Kubernetes с технологией OpenShift

Платформа *Red Hat OpenShift Container Platform* (www.openshift.com) — это продукт, предназначенный для предоставления корпоративной платформы Kubernetes, которая может использоваться для критически важных приложений. Как гибридная облачная платформа, OpenShift создана для развертывания и на пустом компьютере, и в облачных средах.

Kubernetes — это проект с открытым исходным кодом, который может быть построен и запущен огромным количеством способов, а продукты на базе Kubernetes, такие как OpenShift, предназначены для применения надежной поддерживаемой платформы, на которую может положиться бизнес. OpenShift также поставляется в различных вариантах, которые могут быть установлены в центре обработки данных и в облачных средах, таких как AWS и Azure, или просто использоваться из выделенного кластера OpenShift, поддерживаемого Red Hat.

Если заблокировать функции Kubernetes, которые Red Hat встраивает в OpenShift, их тщательно протестируют и проверят. Обучение и документация основаны на этих функциях. Кроме того, в систему можно встроить более сложные функции, которые соответствуют требованиям правительства или тесно интегрируются с различными облачными средами.

Благодаря интуитивно понятной веб-консоли Red Hat OpenShift становится проще в применении для пользователей, начинающих работать с Kubernetes. Пример консоли OpenShift показан на рис. 30.2.

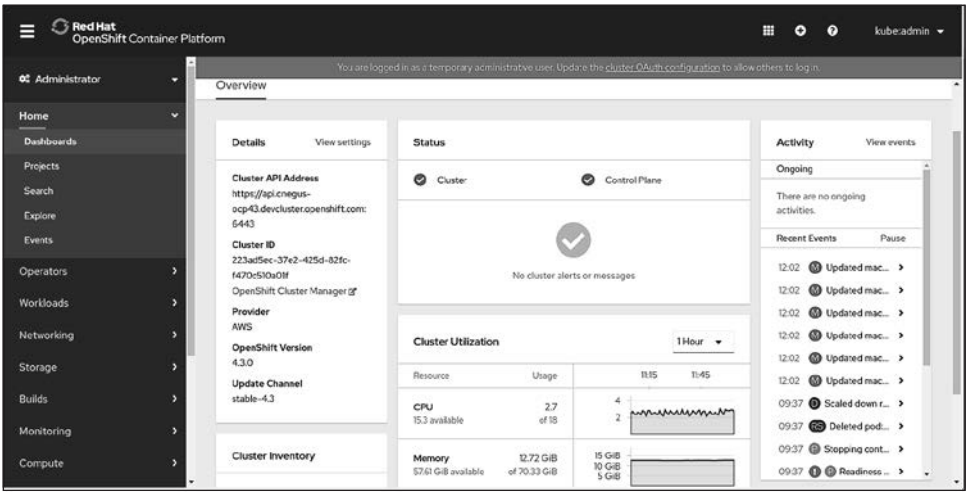


Рис. 30.2. Интуитивно понятный веб-интерфейс OpenShift применяется для развертывания объектов Kubernetes и управления ими

Существуют бесплатные пробные версии OpenShift, доступные на сайте try.openshift.com. Есть также сторонний проект с открытым исходным кодом для OpenShift, называемый OKD, который вы также можете получить бесплатно (www.okd.io).

Резюме

За последние несколько лет Kubernetes стал популярной платформой для развертывания контейнерных приложений в крупных центрах обработки данных. Кластер Kubernetes состоит из мастер-узлов (которые управляют деятельностью кластера) и рабочих узлов (которые фактически запускают контейнеризированные модули).

Как пользователь, применяющий Kubernetes для запуска контейнерных приложений, вы можете создавать развертывания, определяющие состояние запущенного приложения. Например, можно развернуть приложение, настроенное для запуска нескольких реплик модулей, представляющих это приложение. Вы можете идентифицировать приложение как службу и настроить его так, чтобы оно было доступно из определенных портов на узлах.

Используйте проекты на основе Kubernetes, если хотите запускать критически важные приложения в стабильных и поддерживаемых средах. Один из таких продуктов — контейнерная платформа Red Hat OpenShift. С ее помощью вы можете запускать поддерживаемые конфигурации кластеров на базе Kubernetes, которые работают в различных средах, включая компьютерный сервер и различные облачные среды.

Упражнения

Упражнения этого раздела позволят протестировать работу Kubernetes либо онлайн, либо путем настройки Minikube на компьютере. Если затрудняетесь с решением заданий, воспользуйтесь ответами к упражнениям, приведенными в приложении Б (хотя Linux позволяет решать задачи разными способами).

1. Установите Minikube в локальной системе или получите доступ к инстансу Minikube онлайн (например, через руководство Kubernetes.io).
2. Просмотрите свою версию Minikube, а также версии клиента `kubectl` и службы Kubernetes.
3. Создайте развертывание, которое управляет модулем, выполняющим образ контейнера `hello-node` (gcr.io/hello-minikube-zero-install/hello-node).
4. Используйте подходящие команды `kubectl`, чтобы просмотреть развертывание `hello-node` и подробно описать его.
5. Просмотрите текущий набор реплик, связанный с развертыванием `hello-node`.
6. Масштабируйте развертывание `hello-node` до трех реплик.
7. Откройте развертывание `hello-node` за пределами кластера Kubernetes с помощью LoadBalancer.
8. Получите IP-адрес своего инстанса Minikube и номер порта открытой службы `hello-node`.
9. Введите команду `curl` для запроса службы `hello-node`, используя IP-адрес и номер порта из предыдущего упражнения.
10. Задействуйте команды `kubectl` для удаления службы `hello-node` и развертывания, а затем примените команду `minikube` для остановки виртуальной машины Minikube.

Приложения

В этой части

- Приложение А. Устройства.
- Приложение Б. Ответы к упражнениям.

A Устройства

В этом приложении

- Загрузка разных дистрибутивов Linux.
- Создание CD- или DVD-накопителя с Linux.

Если на компьютере не предустановлена Linux или кто-то не установил ее за вас, необходимо найти способ загрузить дистрибутив Linux, а затем либо установить, либо запустить его в реальном времени на компьютере. К счастью, для Linux существует множество самых разных дистрибутивов.

В данном приложении вы узнаете:

- как установить разные дистрибутивы Linux;
- создать накопитель и установить дистрибутив через него;
- загрузить Linux через USB-накопитель.

Для эффективной работы с книгой важно сразу применять знания к дистрибутиву Linux. Поэтому пробуйте все представленные в ней примеры и выполняйте упражнения.

Дистрибутивы Linux чаще всего можно найти на сайтах организаций, которые их создали. В следующих разделах представлены сайты дистрибутивов Linux, на которых можно скачать соответствующие ISO-образы.

ПРИМЕЧАНИЕ

ISO-образ — это образ диска в формате файловой системы ISO 9660, который обычно используется с образами CD и DVD. Это хорошо известный формат, и он читается системами Windows, macOS и Linux.

ISO-образ в зависимости от его размера можно загрузить на USB-накопитель, компакт-диск или DVD-носитель. ISO-образ в файловой системе может быть смонтирован в Linux в петлевом режиме, чтобы можно было просматривать или копировать его содержимое.

Если ISO-образ содержит Linux Live CD или установочный образ, то он является загрузочным. Это означает, что вместо запуска уже установленной на жестком диске операционной системы (например, Windows или Linux) новая система загрузится с CD или DVD. Это позволит запускать другую операционную систему, не изменяя и не повреждая данные на жестком диске.

Установка Fedora

ПРИМЕЧАНИЕ

Я рекомендую загрузить «живой» образ Fedora Workstation Live Image и использовать его при чтении этой книги, так как в большей ее части рассматриваются примеры с этим дистрибутивом. Дистрибутив можно запустить в реальном времени без перезаписи жесткого диска компьютера, чтобы попробовать работу в системе перед полноценной установкой.

Для примеров в этой книге я применял 64-разрядные образы Fedora Workstation 30 и 31, которые можно найти на сайте GetFedora.org (getfedora.org/en/workstation/download). Если у вас 64-разрядный компьютер, необходимо использовать 64-разрядный ISO.

Более поздние версии Fedora с рабочим столом GNOME также работают на 64-разрядном компьютере. Прямая ссылка на образ Fedora 31 Workstation — download.fedoraproject.org/pub/fedora/linux/releases/31/Workstation/x86_64/iso/Fedora-Workstation-Live-x86_64-31-1.9.iso

Имейте в виду, что текущий ISO-образ Fedora Workstation не поместится на CD, поэтому его нужно записать либо на DVD-носитель, либо на USB-накопитель. См. описание способов записи CD/DVD, доступных для Windows, macOS и Linux, далее в этом приложении.

На рис. А.1 показан пример страницы сайта Get Fedora.

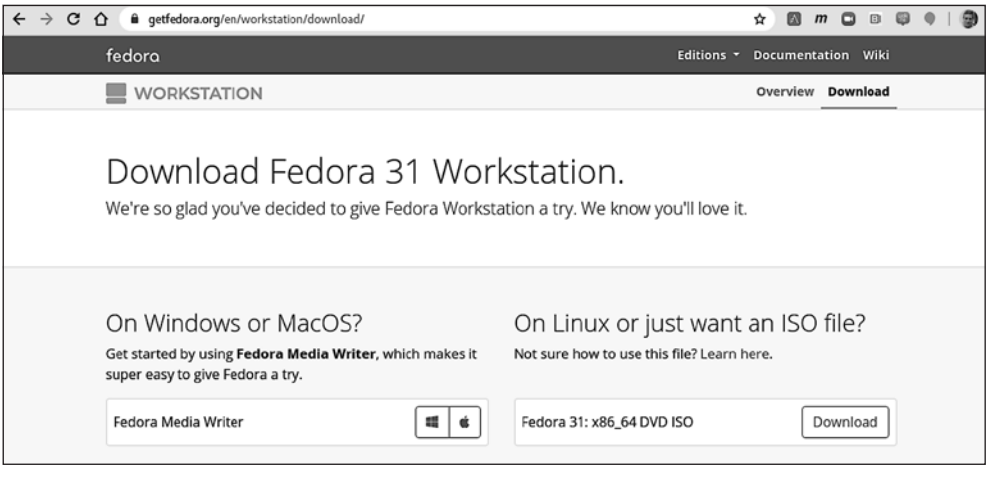


Рис. А.1. Страница загрузки образа Fedora ISO

Здесь загружается ISO-образ Fedora Workstation (GNOME) Live DVD на 64-разрядный компьютер. Этот образ можно загрузить на компьютер и при желании сразу установить на жесткий диск. Для того чтобы загрузить образ, сделайте следующее.

1. Выберите настольную систему Workstation или сервер Server на сайте GetFedora.org. Я советую для работы с книгой выбрать Workstation.
2. Нажмите кнопку Download Now (Загрузить сейчас), а затем кнопку Download (Загрузить). Откроется диалоговое окно для выбора места сохранения файла.
3. Сохраните ISO-образ. В зависимости от настроек менеджер загрузки либо скачает файл в папку по умолчанию, либо предложит выбрать другую (в системе Linux это папка Downloads (Загрузки)).
4. Выберите папку с достаточным количеством места для образа. Запомните ее местоположение, чтобы потом легко найти файл.

Если вам нужна дополнительная информация о том, что делать с загруженным образом, на странице Fedora есть страницы руководства. Актуальная на момент написания этого приложения страница Learn Here описывает, как создавать живые установочные носители для образов. Точные инструкции могут изменяться по мере обновления сайта.

Есть и другие варианты загрузки ISO с сайта. Внизу страницы GetFedora.org можно скачать специально настроенные ISO-образы Fedora — так называемые сборки (spins.fedoraproject.org). Вот список интересных сборок Fedora.

- **Сборка KDE desktop.** Те, кто предпочитает рабочий стол KDE рабочему столу GNOME, могут скачать сборку Plasma KDE.
- **Сборка Lightweight desktop.** Если на компьютере небольшой объем памяти или малая вычислительная мощность, обратите внимание на сборки Xfce и LXQt (легкие хорошо интегрированные среды рабочего стола).
- **Сборка с дополнительными эффектами.** Сборка MATE-Compiz — это классический рабочий стол Fedora Desktop с дополнительным 3D-менеджером окон и визуальными эффектами.
- **Сборка, подходящая для детей.** Сборка SOAS desktop — это операционная система с обучающей платформой Sugar с удобной для детей графической средой. Сборка помещается на обычном USB-накопителе и работает на любом компьютере.

Установка Red Hat Enterprise Linux

Многие крупные корпорации, правительственные учреждения и университеты используют Red Hat Enterprise Linux для работы своих критически важных приложений. Хотя большинство примеров в этой книге хорошо действуют на Fedora, существует много способов выполнить ту же задачу, но по-другому в Red Hat

Enterprise Linux. Для системного администратора Linux в большинстве случаев необходим опыт работы именно в дистрибутиве Red Hat Enterprise Linux.

Хотя исходный код Red Hat Enterprise Linux находится в свободном доступе, ISO с пакетами установки (называемые *двоичными*) доступны только подписчикам на портале Red Hat (access.redhat.com) или в демоверсии (пробный период 30 дней). Если у вас или вашей компании есть учетная запись в Red Hat, можете загрузить необходимые ISO-образы. Перейдите на сайт и следуйте инструкциям, чтобы загрузить ISO-файл Red Hat Enterprise Linux или зарегистрироваться для получения демоверсии: access.redhat.com/downloads.

Red Hat не предоставляет «живые» версии Red Hat Enterprise Linux. Вместо них можно загрузить установочные DVD, работа с которыми описана в главе 9 «Установка Linux».

ПРИМЕЧАНИЕ

Если у вас нет установочного DVD с Red Hat Enterprise Linux, можете воспользоваться аналогом CentOS. Дистрибутив CentOS — не совсем то же самое, что RHEL. Установочный DVD с CentOS для CentOS 8.x доступен по ссылкам на сайте CentOS (centos.org/download), и процедура установки аналогична описанной для Red Hat Enterprise Linux в главе 9.

Установка Ubuntu

Многие новички в Linux начинают с установки Ubuntu. У этого дистрибутива множество поклонников и активных участников. Существуют крупные активные форумы, где в случае проблем с Ubuntu вам могут помочь.

Если у вас уже установлена система Ubuntu, для большей части примеров из книги можете использовать ее. У Ubuntu с рабочим столом GNOME панель мониторинга dash по умолчанию схожа с командной оболочкой bash (также можно переключиться на bash в Ubuntu, чтобы оболочка соответствовала оболочке из примеров в книге). Хотя большинство примеров этой книги относятся к Fedora и RHEL, в новом издании книги я добавил информацию о дистрибутиве Ubuntu.

Чтобы установить Ubuntu, скачайте ISO-образ со страницы Download Ubuntu (ubuntu.com/download/ubuntu).

На рис. А.2 показана страница сайта Download Ubuntu Desktop.

Как и в случае с Fedora, самый простой способ загрузить Ubuntu — это выбрать 64-битный образ, загрузить его и записать на носитель. Вот как это сделать со страницы загрузки Ubuntu.

1. Нажмите кнопку Download. По умолчанию будет загружен самый последний 64-разрядный рабочий стол Ubuntu Live ISO.
2. Загрузка начнется либо автоматически, либо с определением папки для скачивания.
3. В случае с выбором папки выбирайте ту, в которой достаточно места. Запомните местоположение файла, чтобы воспользоваться им позднее.

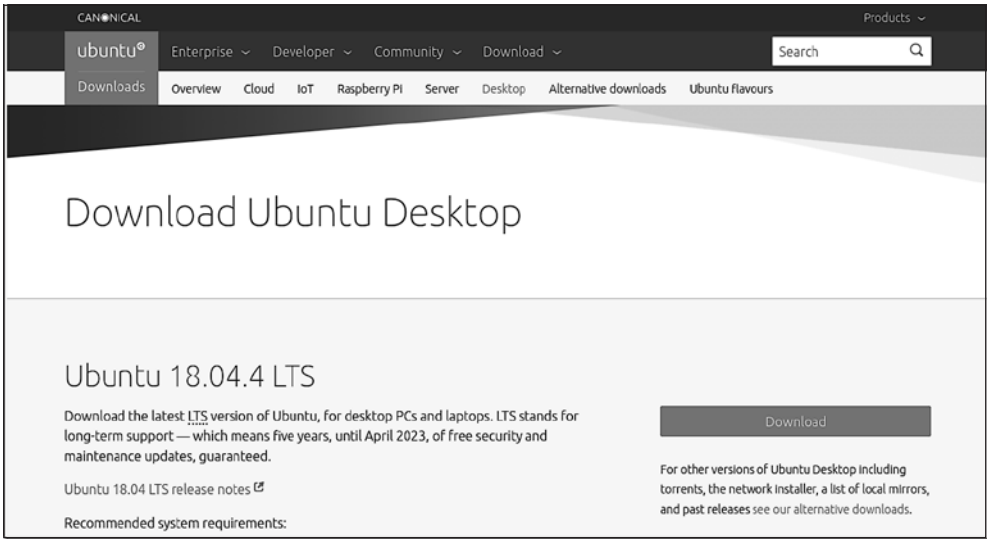


Рис. А.2. Скачайте ISO-образ Ubuntu или выберите другую сборку

После завершения загрузки запишите ISO-образ на DVD, применяя инструкцию из раздела «Создание CD и DVD Linux».

Для дистрибутива Ubuntu доступны и другие типы установочных носителей. Чтобы воспользоваться другими носителями, перейдите на страницу Alternative Downloads (ubuntu.com/download/alternative-downloads). На этом сайте вы получите носитель с множеством настольных и серверных установок.

Установки Linux с USB-накопителя

Кроме CD- или DVD-носителя, ISO-образы Linux можно записать на USB-накопитель. Преимущество USB-накопителей заключается в том, что они доступны как для записи, так и для чтения, что позволяет сохранять данные между сеансами. Большинство современных компьютеров могут загружаться с USB-накопителя, но, возможно, придется сообщить BIOS о загрузке с USB-накопителя, а не с жесткого диска или CD/DVD-привода.

Инструкции по загрузке Fedora и Ubuntu на USB-накопитель таковы.

- **Fedora на USB-накопитель.** С помощью инструмента Live USB Creator можно загрузить ISO-образ Fedora на USB-накопитель в Windows или Linux. Чтобы запустить Fedora, вставьте накопитель в USB-порт, перезагрузите компьютер, прервите загрузку BIOS (возможно, нажимая клавишу F12) и выберите загрузку с USB-накопителя. Инструкция по загрузке с помощью Live USB доступна по ссылке docs.fedoraproject.org/en-US/quick-docs/creating-and-using-a-live- installation-image/index.html.

- **Ubuntu на USB-накопитель.** Для создания загрузочного USB-накопителя с Ubuntu на нем, который работает с Windows, macOS или Linux, перейдите на страницу Ubuntu Download. В разделе Easy ways to switch to Ubuntu (Руководство по Ubuntu для новичков) найдите нужный вариант How to create a bootable USB stick (Запуск компьютера с CD или флешки): ubuntu.com/tutorials/tutorial-create-a-usb-stick-on-ubuntu#1-overview.

Создание CD и DVD Linux

После загрузки образа Linux для CD или DVD используйте один из вариантов загрузки образа на диск и запуска Linux с него. Перечислю необходимые составляющие.

- **ISO-образ для DVD или CD.** Загрузите на компьютер ISO-образы для физических DVD или CD. Большинство ISO-образов Linux слишком велики, чтобы поместиться на CD (в том числе RHEL, Fedora и Ubuntu).
- **Пустой DVD или CD.** Для записи образов вам понадобятся чистые DVD или CD. CD вмещают около 700 Мбайт, DVD (однослойные) — около 4,7 Гбайт.
- **Программа для записи CD/DVD.** Вам понадобятся специальная программа для записи на CD и DVD и привод. Не все CD/DVD-приводы могут записывать данные на DVD (особенно старые), а это значит, что нужен подходящий дисковод.

В следующих подразделах описывается, как записывать информацию на CD и DVD из систем Windows, macOS и Linux.

Запись CD/DVD в Windows

Если ISO-образ Linux загружен в систему Windows, записать его на CD или DVD можно различными способами, например такими.

- **Windows.** В последних версиях Windows функция записи ISO-образов на CD или DVD встроена в систему. После загрузки ISO-образа просто вставьте соответствующий CD или DVD в дисковод компьютера (при этом диск доступен для записи), щелкните правой кнопкой мыши на значке образа ISO из папки и выберите команду Burn Disc Image (Записать образ). Когда появится окно Windows Disc Image Burner, выберите Burn (Записать), чтобы записать образ.
- **Программа Roxio Creator.** Это сторонняя программа для системы Windows с множеством функций для копирования и записи CD и DVD. Больше информации о программе — на сайте roxio.com/en/products/creator/.
- **Программа Nero CD/DVD Burning ROM.** Nero — одна из популярных программ для записи CD и DVD для систем Windows. Больше информации о программе — на сайте nero.com.

Запись CD/DVD в macOS

Как и в системе Windows, в macOS встроено программное обеспечение для записи CD/DVD. Чтобы записать ISO-образ на диск в системе macOS, выполните следующие действия.

1. Скачайте нужный ISO-образ на компьютер. Значок образа появится на рабочем столе.
2. Вставьте пустой CD или DVD (в зависимости от размера образа) в дисковод.
3. Щелкните правой кнопкой мыши на значке образа и выберите команду Burn "Linux" to Disk (Записать на диск). В появившемся диалоговом окне будет задан вопрос, действительно ли вы хотите записать образ на диск.
4. Введите имя для ISO-образа и скорость записи, а затем выберите Burn (Записать). Начнется запись образа на диск.
5. После окончания записи извлеките диск. Теперь его можно использовать для установки системы.

Запись CD/DVD в Linux

Linux имеет как графические, так и командные инструменты для записи образов на физические носители. В этом разделе мы рассмотрим, как использовать программы K3b или `cdrecord` (или `wodim`) для записи ISO-образов на CD или DVD. Если они не установлены, установите их с помощью:

- **для Fedora или RHEL:**

```
# yum install k3b
# yum install wodim
```

- **для Debian или Ubuntu:**

```
# apt-get install k3b
# apt-get install wodim
```

Запись CD/DVD с рабочего стола Linux

Вот как можно создать загрузочные диски из работающей системы Linux (например, Fedora) с помощью программы K3b (она включена в рабочий стол KDE, но действует и в GNOME).

1. Скачайте ISO-образ на жесткий диск (CD вмещают около 700 Мбайт, DVD (однослойные) — около 4,7 Гбайт).
2. Откройте приложение для записи дисков. Я рекомендую использовать K3b CD and DVD Kreator (k3b.org). В Fedora выберите Activities (Обзор) и введите `K3b` в окне поиска (или в окне терминала). Появится окно K3b — The CD and DVD Kreator.

3. Перейдите на вкладку Tools ▸ Burn Image (Инструмент ▸ Записать образ), чтобы записать образ на диск. Вам будет предложено выбрать файл образа.
4. Перейдите к изображению, которое вы только что загрузили или скопировали на жесткий диск, и выберите его. После этого откроется окно записи образа и значения образа. На рис. А.3 показано окно K3b с выбранным образом Fedora.

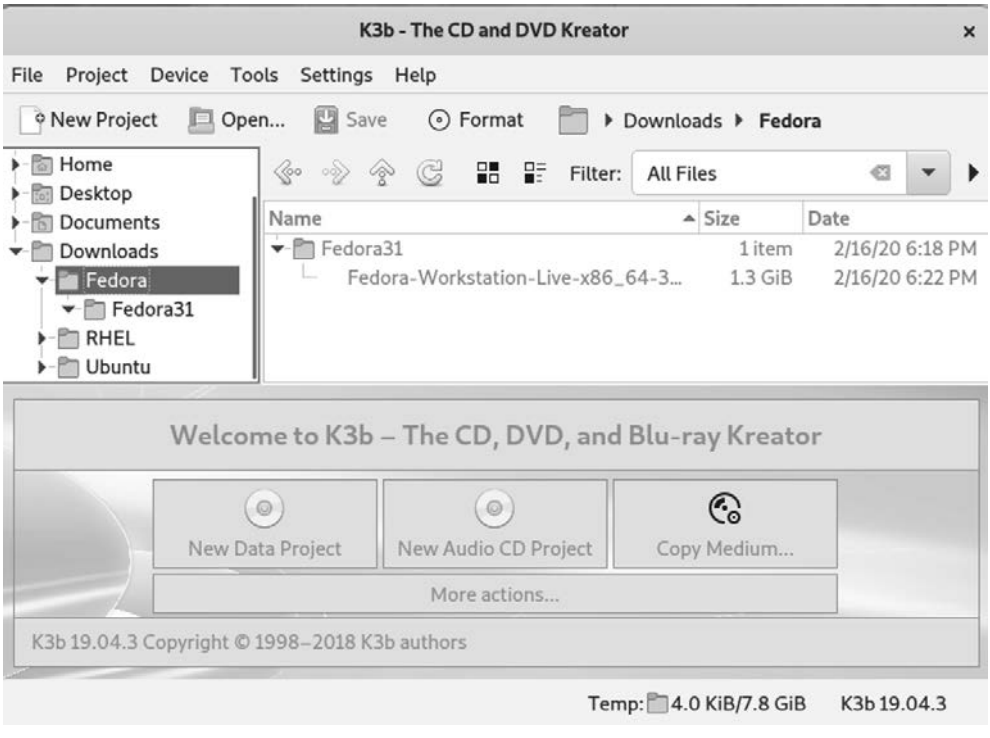


Рис. А.3. Запись CD/DVD с помощью K3b

5. Вставьте пустой диск в дисковод (если появится окно CD/DVD Creator, закройте его)
6. Проверьте настройки в окне Burn Image (Записать образ) (часто значения по умолчанию подходят, но можно замедлить скорость, если запись не удастся). Вы также можете установить флажок Simulate, чтобы проверить качество записи перед фактической записью. Нажмите Start, чтобы продолжить.
7. После завершения записи CD/DVD извлеките диск (или это будет выполнено автоматически) и сделайте на нем пометки (например, имя дистрибутива, номер версии, дату и имя ISO-образа).

Теперь можно воспользоваться этим диском для загрузки Linux на другой компьютер.

Запись CD/DVD с помощью командной строки

Если у вас нет графического интерфейса или вы предпочитаете работать из оболочки, примените для записи ISO команду `cdrecord`. Скачайте ISO-образ, вставьте пустой диск и используйте простую командную строку для записи образа:

```
# cdrecord -v whatever.iso
```

См. руководство к команде `cdrecord` для того, чтобы узнать, что еще можно сделать с ее помощью.

Б Ответы к упражнениям

В этом приложении представлены ответы к упражнениям из книги. Linux позволяет подходить к решению задач с разных сторон, и в данном приложении предлагаются некоторые из вариантов.

Часть упражнений требуют изменения системных файлов, которые могут поменять основной функционал системы или даже сделать так, чтобы система перестала загружаться. Поэтому я рекомендую выполнять упражнения в системе Linux, которую можно изменять и удалять, если что-то пойдет не так. Отличный вариант — использовать специальные виртуальные машины.

Глава 1. Начало работы в Linux

В первой главе не было упражнений.

Глава 2. Идеальный рабочий стол в Linux

Здесь подробно описаны способы выполнения данных задач как на рабочих столах GNOME 2, так и на рабочих столах GNOME 3.

1. Чтобы начать практиковать знания из книги, установите систему Linux, предпочтительно на жестком диске. Это позволит не потерять все изменения при перезагрузке. Для начала можете использовать установленные системы Fedora Live CD, Ubuntu или Red Hat Enterprise Linux, например:
   - **Fedora Live CD (GNOME 3).** Создайте Fedora Live CD, как описано в приложении А. Запустите его в реальном времени, как описано в разделе «Запуск Fedora со средой GNOME из образа» в главе 2, или установите его и запустите с жесткого диска, как говорилось в главе 9 «Установка Linux»;
   - **Ubuntu (GNOME 3).** Установите Ubuntu и программное обеспечение GNOME Shell, как описано в начале главы 2;

- **Red Hat Enterprise Linux 8 (GNOME 3).** Установите Red Hat Enterprise Linux 7, как описано в главе 9;
- **Red Hat Enterprise Linux 6 и ранее (GNOME 2).** Установите дистрибутив Red Hat Enterprise Linux 6.

2. Чтобы запустить браузер Firefox и перейти на домашнюю страницу GNOME (gnome.org), необходимо выполнить несколько простых шагов. Если подключения к сети нет, обратитесь к главе 14 «Администрирование сети» за информацией о подключении к проводным и беспроводным сетям.
   - **GNOME 3.** Нажмите клавишу Windows, чтобы перейти на обзорный экран. Введите `Firefox`, чтобы выделить значок браузера, и запустите его с помощью клавиши Enter. Введите в строке поиска `gnome.org` и нажмите клавишу Enter.
   - **GNOME 2.** Нажмите на значок браузера на верхней панели. Введите в строке поиска `gnome.org` и нажмите клавишу Enter.
3. Чтобы установить фоновое изображение с сайта `gnome-look.org`, скачайте понравившееся в папку Изображения и выберите его в качестве текущего фона. Порядок действий таков (актуален и для GNOME 2, и для GNOME 3).
   - Введите `gnome-look.org` в поисковой строке в Firefox и нажмите клавишу Enter.
   - Найдите подходящее изображение. Нажмите кнопку Download и скачайте изображение в свою папку Изображения.
   - Откройте папку с изображением, щелкните на нем правой кнопкой мыши и выберите вариант Set as Wallpaper (Установить как фон). Изображение будет установлено в качестве фона рабочего стола.
4. Чтобы запустить файловый менеджер, перейдите на второе рабочее место и следуйте инструкции.
   - **Для GNOME 3:**
     а) нажмите клавишу Windows;
     б) выберите значок Файлы на панели приложений (слева). На текущем рабочем месте откроется новое окно приложения;
     в) правой кнопкой мыши нажмите на имя окна (Файлы) и из контекстного меню выберите вариант Move to Monitor Down (Переместить на рабочее место вниз). Окно приложения Файлы переместится на второе рабочее место.
   - **Для GNOME 2:**
     а) откройте домашнюю папку на рабочем столе GNOME 2 (двойным щелчком на значке Home);
     б) правой кнопкой мыши щелкните на верхней панели окна и выберите или Move to Workspace Right (Переместить на рабочее место справа), или Move to Another Workspace (Переместить на другое рабочее место). Выберите нужное рабочее место из списка.

5. Чтобы найти загруженное ранее изображение и открыть его, сначала перейдите в домашнюю папку и откройте папку Изображения. Дважды щелкните на изображении, и оно откроется в менеджере изображений, установленном по умолчанию.
6. Перемещение между рабочим местом с браузером Firefox и местом с файловым менеджером осуществить довольно просто.

   Если предыдущие упражнения были выполнены правильно, окна приложений Файлы и Firefox должны находиться на разных рабочих местах. Вот как можно перемещаться между этими рабочими местами в GNOME 3 и GNOME 2.

   - **GNOME 3.** Нажмите клавишу Windows и выберите в правой колонке нужное рабочее место. Можно также нажать сочетание клавиш Alt+Tab и, удерживая клавишу Tab, с помощью клавиш ←, ↑, ↓ и → подсветить нужное приложение в списке.
   - **GNOME 2.** Выберите рабочее место с помощью нижнего правого значка, обозначающего список рабочих мест. Если у вас включены эффекты рабочего стола (System ▸ Preferences ▸ Desktop Effects ▸ Compiz (Система ▸ Параметры ▸ Эффекты рабочего стола)), нажмите сочетание клавиш Ctrl+Alt+→ (или ←), чтобы перейти к следующему рабочему месту.
7. Чтобы открыть список приложений, установленных в системе, и выбрать средство просмотра изображений, используя как можно меньше щелчков кнопкой мыши или нажатий клавиш, выполните следующее.
   - **GNOME 3.** Переведите указатель мыши в левый верхний угол, чтобы перейти в Overview (Обзор). Нажмите на значок Applications (Приложения), выберите папку Utilities (Утилиты) и в ней приложение Image Viewer (Просмотр изображений).
   - **GNOME 2.** Чтобы открыть изображение на рабочем столе, перейдите в Applications ▸ Graphics ▸ Image Viewer (Приложения ▸ Графика ▸ Просмотр изображений) и выберите приложение просмотра.
8. Чтобы отобразить весь список рабочих мест в уменьшенном виде, выполните следующие действия.
   - **GNOME 3.** Нажмите сочетание клавиш Alt+Tab. Удерживая клавишу Alt, нажимайте клавишу Tab, чтобы выбрать нужное приложение. Отпустите клавишу Alt, чтобы подтвердить выбор.
   - **GNOME 2.** Нажмите сочетание клавиш Ctrl+Alt+Tab. Удерживая клавиши Ctrl+Alt, нажимайте клавишу Tab, чтобы выбрать нужное приложение. Отпустите клавиши Ctrl и Alt, чтобы подтвердить выбор.
9. Как запустить музыкальный проигрыватель с помощью клавиатуры.
   - **GNOME 3:**

     а) нажмите клавишу Windows и перейдите в Обзор;

     б) вводите в поисковой строке слово Rhyth, пока значок приложения не подсветится, и нажмите Enter. Если в дистрибутиве Ubuntu не установлена

программа Rhythmbox, введите `Bansh`, чтобы открыть проигрыватель Banshee Media Player.

- **GNOME 2.** Нажмите сочетание клавиш Alt+F2. Появится командная строка (Run Application), введите в ней `rhythmbox` и нажмите Enter.

10. Чтобы сделать снимок рабочего стола с помощью клавиатуры, нажмите на клавишу Print Screen (актуально для GNOME 3 и GNOME 2). Нажмите сочетание клавиш Alt+Print Screen, чтобы сделать снимок текущего окна. В обоих случаях снимок сохранится в папке Изображения в домашней папке.

Глава 3. Использование оболочки

1. Чтобы отключить виртуальные консоли и вернуться на рабочий стол в системах Fedora или Ubuntu (эта функция отключена в некоторых системах RHEL), выполните следующее:
   а) нажмите и удерживайте сочетание клавиш Ctrl+Alt и нажмите F2 (Ctrl+Alt+F2). Появится текстовая консоль;
   б) введите имя пользователя (нажмите клавишу Enter) и пароль (нажмите клавишу Enter);
   в) введите несколько команд, например `id`, `pwd` и `ls`;
   г) введите команду `exit`, чтобы выйти из оболочки и вернуться к входу в систему;
   д) нажмите сочетание клавиш Ctrl+Alt+F1, чтобы вернуться в виртуальную консоль вашего рабочего стола. (В разных системах Linux рабочий стол может находиться в разных виртуальных консолях. Чаще всего они открываются с помощью сочетаний клавиш Ctrl+Alt+F7 и Ctrl+Alt+F2).
2. Как сделать шрифт текста красным, а фон — желтым в окне Terminal (Терминал):
   а) на рабочем столе GNOME выберите Applications ▸ System Tools ▸ Terminal (Приложения ▸ Системные утилиты ▸ Терминал);
   б) в окне Terminal (Терминал) выберите пункт Edit ▸ Profile Preferences (Параметры ▸ Профиль);
   в) выберите вкладку Colors (Цвета) и уберите выделение с пункта Use colors from system theme (Использовать цвета из системной темы);
   г) установите флажок Text Color (Цвет текста), выберите красный цвет и нажмите кнопку Select (Выбрать);
   д) установите флажок Background Color (Цвет фона), выберите желтый цвет и нажмите кнопку Select (Выбрать);
   е) закройте окно Profile Preferences (Профиль). Цвет окна Terminal (Терминал) обновится до выбранного;

ж) снова установите флажок Use colors from system theme (Использовать цвета из системной темы), чтобы вернуться к настройкам по умолчанию.

3. Как найти справочные страницы команд `mount` и `tracepath`:
   а) запустите команду `mount`, чтобы найти местоположение команды в каталоге `/usr/bin/mount` или в `/bin/mount`;
   б) запустите команду `locate tracepath`, чтобы найти справочную страницу в каталоге `/usr/share/man/man8/tracepath.8.gz`.
4. Чтобы запустить, вызвать и изменить эти команды, введите следующее:

```
$ cat /etc/passwd
$ ls $HOME
$ date
```

   а) нажимайте на клавишу ↑, пока не увидите команду `cat /etc/passwd`. Если курсор находится не в конце строки, переведите его в конец, нажав сочетание клавиш Ctrl+E. Нажмите клавишу Пробел над словом `passwd`, введите слово `group` и нажмите клавишу Enter;
   б) введите команду `man ls` и найдите параметр сортировки по времени (`-t`). Нажимайте клавишу ↑, пока не увидите команду `ls $HOME`. Используйте клавишу → или сочетание клавиш Alt+B, чтобы установить курсор слева от `$HOME`. Введите `-t`, чтобы создать команду `ls -t $HOME`. Нажмите клавишу Enter, чтобы запустить команду;
   в) введите `man date`, чтобы отобразить справочную страницу команды `date`. С помощью клавиши ↑ вызовите команду `date` и добавьте индикатор нужного формата. Формат `%D` выводит результат в нужном виде:

```
$ date +%D
04/27/20
```

5. Используя автозаполнение команд, введите `basename/usr/share/doc/`. Для этого наберите `basen<Tab> / u<Tab>sh<Tab>do<Tab>`.
6. Перенаправьте каталог `/etc/services` в команду `less` с помощью контейнера: `$ cat /etc/services | less`.
7. Сделайте вывод команды `date` в формате `Today is Thursday, April 23, 2020`:

```
$ echo "Today is $(date +'%A, %B %d, %Y')"
```

8. Просмотрите переменные, чтобы найти текущие имена хоста, пользователя, оболочки и домашних каталогов:

```
$ echo $HOSTNAME
$ echo $USERNAME
$ echo $SHELL
$ echo $HOME
```

9. Как добавить постоянный псевдоним `mypass`, который отображает содержимое файла `/etc/passwd`:
   а) введите команду `nano $HOME/.bashrc`;
   б) переместите курсор на открытую строку в нижней части страницы. (При необходимости нажмите клавишу Enter, чтобы открыть новую строку);
   в) в отдельной строке введите `alias m="cat /etc/passwd"`;
   г) нажмите сочетание клавиш Ctrl+O, чтобы сохранить изменения, и Ctrl+X, чтобы закрыть файл;
   д) введите команду `source $HOME/.bashrc`;
   е) введите команду `alias m`, чтобы убедиться в том, что псевдоним установлен правильно: `alias m='cat/etc/passwd'`;
   ж) введите команду `m` (файл `/etc/passwd` появится на экране).
10. Чтобы отобразить справочную страницу системного вызова `mount`, используйте команду `man -k`. Затем примените команду `mount` с правильным номером раздела (`8`), чтобы открыть нужную справочную страницу `mount`:

```
$ man -k mount | grep ^mount
mount       (2)  - mount filesystem
mount       (8)  - mount a filesystem
...
mountpoint  (1)  - see if a directory is a mountpoint
mountstats  (8)  - Displays various NFS client per-mount statistics
$ man 2 mount
MOUNT(2)      Linux Programmer's Manual MOUNT(2)
NAME
      mount - mount file system
SYNOPSIS
      #include <sys/mount.h>
```

Глава 4. Файловая система

1. Создайте каталог `projects`, девять пустых файлов (от `house1` до `house9`) и перечислите только эти файлы:

```
$ mkdir $HOME/projects/
$ touch $HOME/projects/house{1..9}
$ ls $HOME/projects/house{1..9}
```

2. Создайте путь к каталогу `$HOME/projects/houses/doors/` и несколько пустых файлов в нем:

```
$ cd
$ mkdir $HOME/projects/houses
$ touch $HOME/projects/houses/bungalow.txt
$ mkdir $HOME/projects/houses/doors/
$ touch $HOME/projects/houses/doors/bifold.txt
$ mkdir -p $HOME/projects/outdoors/vegetation/
$ touch $HOME/projects/outdoors/vegetation/landscape.txt
```

3. Скопируйте файлы house1 и house5 в каталог $HOME/projects/houses/:

```
$ cp $HOME/projects/house[15] $HOME/projects/houses
```

4. Рекурсивно скопируйте каталог /usr/share/doc/initscripts* в домашний каталог $HOME/projects/:

```
$ cp -ra /usr/share/doc/initscripts*/ $HOME/projects/
```

5. Рекурсивно перечислите содержимое каталога $HOME/projects/. Передайте выходные данные в команду less, чтобы просмотреть их на странице:

```
$ ls -lR $HOME/projects/ | less
```

6. Удалите файлы house6, house7 и house8 без запроса о подтверждении:

```
$ rm -f $HOME/projects/house[678]
```

7. Переместите файлы house3 и house4 в каталог $HOME/projects/houses/doors:

```
$ mv $HOME/projects/house{3,4} $HOME/projects/houses/doors/
```

8. Удалите каталог $HOME/projects/houses/doors и его содержимое:

```
$ rm -rf $HOME/projects/houses/doors/
```

9. Измените права доступа к файлу $HOME/projects/house2 таким образом, чтобы он мог быть прочитан и записан только владельцем, мог быть прочитан только группой и не имел прав для других:

```
$ chmod 640 $HOME/projects/house2
```

10. Рекурсивно измените права каталога $HOME/projects/ так, чтобы никто не мог записывать ни в какие файлы или каталоги, расположенные ниже этой точки файловой системы:

```
$ chmod -R a-w $HOME/projects/
$ ls -lR $HOME/projects/
/home/joe/projects/:

total 12

-r--r--r--. 1 joe joe    0 Jan 16 06:49 house1

-r--r-----. 1 joe joe    0 Jan 16 06:49 house2

-r--r--r--. 1 joe joe    0 Jan 16 06:49 house5

-r--r--r--. 1 joe joe    0 Jan 16 06:49 house9

dr-xr-xr-x. 2 joe joe 4096 Jan 16 06:57 houses

dr-xr-xr-x. 2 joe joe 4096 Jul  1  2014 initscripts-9.03.40

dr-xr-xr-x. 3 joe joe 4096 Jan 16 06:53 outdoors
...
```

Глава 5. Работа с текстовыми файлами

1. Выполните следующие команды, чтобы создать файл `/tmp/services`, а затем отредактировать его так, чтобы название `WorldWideWeb` отображалось как `World Wide Web`:

```
$ cp /etc/services /tmp
$ vi /tmp/services
/WorldWideWeb <Enter>
cwWorld Wide Web <Esc>
```

   Следующие две строки показывают исходный текст и результат:

```
http                  80/tcp       www www-http     # WorldWideWeb HTTP
http                  80/tcp       www www-http     # World Wide Web HTTP
```

2. Один из способов переместить абзац в файл `/tmp/services` — это найти первую строку абзаца, удалить пять строк (`5dd`), перейти к концу файла (`G`) и вставить текст (`p`):

```
$ vi /tmp/services
/Note that it is<Enter>
5dd
G
p
```

3. Чтобы использовать режим `ex` для поиска термина `tcp` (чувствительного к регистру) в файле `/tmp/services` и изменить его на `WHATEVER`, введите следующее:

```
$ vi /tmp/services
:g/tcp/s//WHATEVER/g<Enter>
```

4. Чтобы найти в каталоге `/etc` каждый файл с именем `passwd` и перенаправить ошибки из поиска в каталог `/dev/null`, введите следующее:

```
$ find /etc -name passwd 2> /dev/null
```

5. Создайте в своем домашнем каталоге каталог `TEST`. Создайте в нем файлы с именем `one`, `two` и `three`, которые имеют полные права на чтение/запись/выполнение для всех (пользователя, группы и др.). Введите команду `find`, которая найдет эти файлы и любые другие файлы, имеющие права на запись, открытые для «других» из вашего домашнего каталога и расположенные ниже:

```
$ mkdir $HOME/TEST
$ touch $HOME/TEST/{one,two,three}
$ chmod 777 $HOME/TEST/{one,two,three}
$ find $HOME -perm -002 -type f -ls
148120  0 -rwxrwxrwx   1 chris chris 0 Jan  1 08:56 /home/chris/TEST/two
148918  0 -rwxrwxrwx   1 chris chris 0 Jan  1 08:56 home/chris/TEST/three
147306  0 -rwxrwxrwx   1 chris chris 0 Jan  1 08:56 /home/chris/TEST/one
```

6. Найдите в каталоге `/usr/share/doc` файлы, которые не изменялись более чем 300 дней:

```
$ find /usr/share/doc -mtime +300
```

7. Создайте каталог `/tmp/FILES`. Найдите в каталоге `/usr/share` все файлы размером более 5 и менее 10 Мбайт и скопируйте их в каталог `/tmp/FILES`:

```
$ mkdir /tmp/FILES
$ find /usr/share -size +5M -size -10M -exec cp {} /tmp/FILES \;
$ du -sh /tmp/FILES/*
6.6M    /tmp/FILES/BidiCharacterTest.txt
7.6M    /tmp/FILES/BidiTest.txt
5.2M    /tmp/FILES/day.jpg
```

8. Найдите в каталоге `/tmp/FILES` все файлы и сделайте резервную копию каждого здесь же. Используйте существующее имя каждого файла и добавьте файл `.mybackup` для создания резервных копий:

```
find /tmp/FILES/ -type f -exec cp {} {}.mybackup \;
```

9. Установите пакет `kernel-doc` в Fedora или Red Hat Enterprise Linux. Используя команду `grep`, найдите в файлах, содержащихся в каталоге `/usr/share/doc/kernel-doc*`, термин `e1000` (без учета регистра) и перечислите имена файлов, содержащих его:

```
# yum install kernel-doc
$ cd /usr/share/doc/kernel-doc*
$ grep -rli e1000 .
./Documentation/powerpc/booting-without-of.txt
./Documentation/networking/e100.txt
...
```

10. Снова найдите термин `e1000` в том же месте. Однако на этот раз перечислите каждую строку, содержащую термин, и выделите его цветом:

```
$ cd /usr/share/doc/kernel-doc-*
$ grep -ri --color e1000 .
```

Глава 6. Управление активными процессами

1. Чтобы перечислить все процессы, запущенные в вашей системе, с полным набором столбцов и передать выходные данные в команду `less`, введите следующее:

```
$ ps -ef | less
```

2. Чтобы перечислить все процессы, запущенные в системе, и отсортировать их по имени пользователя, выполняющего каждый процесс, введите следующее:

```
$ ps -ef --sort=user | less
```

3. Чтобы вывести список всех процессов, запущенных в системе, со столбцами имени, идентификатора процесса, имени пользователя, названия группы, приоритета, размера виртуальной памяти, размера оперативной памяти и команды, введите следующее:

```
$ ps -eo 'pid,user,group,nice,vsz,rss,comm' | less
  PID USER     GROUP     NI    VSZ   RSS COMMAND
```

```
1 root     root       0  19324  1236 init
2 root     root       0      0     0 kthreadd
3 root     root       -      0     0 migration/0
4 root     root       0      0     0 ksoftirqd/0
```

4. Чтобы запустить команду `top`, а затем отсортировать данные по использованию процессора и памяти, введите следующее:

```
$ top
P
M
P
M
```

5. Чтобы запустить процесс `gedit` с рабочего стола и использовать окно System Monitor (Системный монитор) для его завершения, выполните следующие действия:

```
$ gedit &
```

Далее в GNOME 2 выберите Applications ▸ System Tools ▸ System Monitor (Приложения ▸ Системные утилиты ▸ Системный монитор). А в GNOME 3 с экрана Applications (Приложения) введите System Monitor и нажмите клавишу Enter. Найдите процесс `gedit` на вкладке Processes (Процессы). Вы можете отсортировать процессы по алфавиту, щелкнув на заголовке. Щелкните правой кнопкой мыши на команде `gedit` и выберите вариант либо End Process (Завершить), либо Kill Process (Убить). Окно gedit на экране должно исчезнуть.

6. Чтобы запустить процесс `gedit` и использовать команду `kill` для отправки сигнала на паузу (остановку) этого процесса, введите следующее:

```
$ gedit &
[1] 21532

$ kill -SIGSTOP 21532
```

7. Чтобы использовать команду `killall` для указания команде `gedit` (приостановленной в предыдущем упражнении) продолжить работу, выполните следующее:

```
$ killall -SIGCONT gedit
```

Убедитесь, что текст, который вы набрали после того, как приостановили команду `gedit`, теперь появляется в окне.

8. Чтобы установить команду `xeyes`, запустите ее около 20 раз в фоновом режиме и запустите `killall`, чтобы убить все 20 процессов `xeyes` одновременно:

```
# yum install xorg-x11-apps
$ xeyes &
$ xeyes &
...
$ killall xeyes &
```

Помните, что вы должны быть суперпользователем, чтобы установить пакет. После этого не забудьте повторить команду `xeyes` 20 раз. Распределите окна по всему экрану и двигайте мышь, чтобы наблюдать за движением глаз. Все окна `xeyes` должны исчезнуть сразу же, как только вы наберете команду `killall xeyes`.

9. От имени обычного пользователя запустите команду `gedit` с приоритетом `5`:

```
# nice -n 5 gedit &
[1] 21578
```

10. Чтобы изменить значение приоритета запущенной команды `gedit` на `7`, введите команду `renice`:

```
# renice -n 7 21578
21578: old priority 0, new priority 7
```

Используйте любую команду, чтобы убедиться, что текущее значение приоритета для команды `gedit` теперь равно `7`. Например, можете ввести следующее:

```
# ps -eo 'pid,user,nice,comm' | grep gedit
21578 chris     7 gedit
```


Глава 7. Простые скрипты оболочки

1. Пример создания в своем каталоге `$HOME/bin` скрипта под названием `myownscript`. При запуске скрипт должен выводить следующую информацию:

```
Сегодня Sat Jun 10 15:45:04 EDT 2019.
Вы находитесь в каталоге /home/joe, и ваш хост-адрес — abc.example.com.
```

Далее приведен один из способов создания скрипта с именем `myownscript`:

а) создайте каталог `bin`, если его нет:

```
$ mkdir $HOME/bin
```

б) используя любой текстовый редактор, создайте скрипт с именем `$HOME/bin/myownscript`, содержащий следующее:

```
#!/bin/bash
# myownscript
# Перечислите сведения о вашей текущей системе.
echo "Сегодня $(дата)."
echo "Вы находитесь $(pwd) и ваш хост-адрес - $(hostname)."
```

в) сделайте скрипт исполняемым:

```
$ chmod 755 $HOME/bin/myownscript
```

2. Создайте скрипт, который считывает три позиционных параметра из командной строки и присваивает эти параметры переменным с именами `ONE`, `TWO` и `THREE` соответственно. Кроме того, замените `X` на количество параметров, а `Y` — на все

введенные параметры. Затем замените A содержимым переменной ONE, B — переменной TWO и C — переменной THREE, как показано далее:

а) чтобы создать скрипт, откройте файл с именем $HOME/bin/myposition и добавьте следующее содержимое:

```
#!/bin/bash
# myposition
ONE=$1
TWO=$2
THREE=$3
echo "Существует $# параметров, которые включают в себя параметр $@"
echo "Первый – это $ONE, второй – это $TWO, третий – это $THREE."
```

б) чтобы сделать скрипт с именем $HOME/bin/myposition исполняемым, введите следующее:

```
$ chmod 755 $HOME/bin/myposition
```

в) чтобы проверить скрипт, запустите его с несколькими аргументами командной строки, как показано далее:

```
$ myposition Where Is My Hat Buddy?
Существует 5 параметров, которые включают в себя параметр: Where Is My
Hat Buddy?
Первый – это Where, второй – это Is, третий – это My.
```

3. Чтобы создать описанный скрипт, выполните следующие действия:

а) создайте файл с именем $HOME/bin/myhome и сделайте его исполняемым:

```
$ touch $HOME/bin/myhome
$ chmod 755 $HOME/bin/myhome
```

б) пример того, как может выглядеть скрипт myhome:

```
#!/bin/bash
# myhome
read -p "На какой улице вы выросли?" mystreet
read -p "В каком городе вы выросли?" mytown
echo "Я вырос на улице под названием $mystreet в городе $mytown."
```

в) запустите скрипт, чтобы убедиться, что он работает. В следующем примере показано, как могут выглядеть входные и выходные данные для него:

```
$ myhome
На какой улице вы выросли? Harrison
В каком городе вы выросли? Princeton
Я вырос на улице под названием Harrison в городе Princeton.
```

4. Чтобы создать требуемый скрипт, выполните следующие действия:

а) используя любой текстовый редактор, создайте скрипт с именем $HOME/bin/myos и сделайте его исполняемым:

```
$ touch $HOME/bin/myos
$ chmod 755 $HOME/bin/myos
```

б) скрипт может содержать следующее:

```
#!/bin/bash
# myos
read -p "Ваша любимая операционная система – macOS, Windows или Linux? "
opsys
if [ $opsys = macOS ] ; then
  echo "macOS – хороший выбор, но он недостаточно хорош для меня."
elif [ $opsys = Windows ] ; then
  echo "Как-то я использовал Windows. Для чего этот синий экран?"
elif [ $opsys = Linux ] ; then
  echo "Отличный выбор!"
else
  echo "Разве $opsys – это операционная система?"
fi
```

5. Чтобы создать скрипт с именем `$HOME/bin/animals`, который проходит через слова «мышь», «корова», «гусь» и «свинья» и через цикл `for` и добавляет каждое из них в конец строки «У меня есть...», выполните следующие действия:

   а) сделайте скрипт исполняемым:

```
$ touch $HOME/bin/animals
$ chmod 755 $HOME/bin/animals
```

   б) скрипт может содержать следующее:

```
#!/bin/bash
# animals
for ANIMALS in moose cow goose sow ; do
  echo "У меня есть $ANIMALS"
done
```

   в) при запуске скрипта выходные данные должны выглядеть так:

```
$ animals
У меня есть мышь
У меня есть корова
У меня есть гусь
У меня есть свинья
```

Глава 8. Системное администрирование

1. Чтобы запустить интерфейс Cockpit в своей системе, введите следующее:

```
# systemctl enable --now cockpit.socket
Created symlink /etc/systemd/system/sockets.target.wants/cockpit.socket
   → /usr/lib/systemd/system/cockpit.socket
```

2. Чтобы открыть интерфейс Cockpit в браузере, введите имя хоста или IP-адрес системы, поддерживающей службу, а затем номер порта 9090. Например, наберите в адресной строке браузера следующее:

```
https://host1.example.com:9090/
```

3. Чтобы найти в каталоге `/var/spool` все файлы, принадлежащие пользователям, отличным от суперпользователя, и вывести их длинный список, введите следующее (я рекомендую войти от имени суперпользователя, чтобы найти недоступные для других пользователей файлы):

```
$ su -
Password: *********
# find /var/spool -not -user root -ls | less
```

4. Чтобы стать суперпользователем и создать пустой или обычный текстовый файл с именем `/mnt/test.txt`, введите следующее:

```
$ su -
Password: *********
# touch /mnt/test.txt
# ls -l /mnt/test.txt
-rw-r--r--. 1 root root 0 Jan  9 21:51 /mnt/test.txt
```

5. Чтобы стать суперпользователем и отредактировать файл `/etc/sudoers` так, чтобы ваша обычная учетная запись пользователя (например, `bill`) получала полные привилегии суперпользователя с помощью команды `sudo`, введите:

```
$ su -
Password: *********
# visudo
o
bill     ALL=(ALL)       ALL
Esc ZZ
```

Поскольку команда `visudo` открывает файл `/etc/sudoers` в редакторе `vi`, в примере вводится параметр `o`, который открывает строку, а затем пишет в ней, чтобы дать пользователю `bill` полные привилегии суперпользователя.

После ввода строки нажмите клавишу Esc, чтобы вернуться в командный режим, и введите `ZZ`, чтобы записать и выйти.

6. Чтобы с помощью команды `sudo` создать файл `/mnt/test2.txt` и убедиться, что он находится там и принадлежит суперпользователю, введите следующее:

```
[bill]$ sudo touch /mnt/test2.txt
We trust you have received the usual lecture from the local System
Administrator. It usually boils down to these three things:
    #1) Respect the privacy of others.
    #2) Think before you type.
    #3) With great power comes great responsibility.
[sudo] password for bill:  *********
[bill]$ ls -l /mnt/text2.txt
-rw-r--r--. 1 root root 0 Jan  9 23:37 /mnt/text2.txt
```

7. Чтобы смонтировать и размонтировать USB-накопитель и просмотреть системный журнал:

а) выполните команду `journalctl -f` от имени суперпользователя в окне Terminal (Терминал) и проследите за выводом в течение следующих нескольких шагов:

```
# journalctl -f
Jan 25 16:07:59 host2 kernel: usb 1-1.1: new high-speed USB device
    number 16 using ehci-pci
Jan 25 16:07:59 host2 kernel: usb 1-1.1: New USB device found,
    idVendor=0ea0, idProduct=2168
Jan 25 16:07:59 host2 kernel: usb 1-1.1: New USB device strings:
    Mfr=1, Product=2, SerialNumber=3
Jan 25 16:07:59 host2 kernel: usb 1-1.1: Product: Flash Disk
Jan 25 16:07:59 host2 kernel: usb 1-1.1: Manufacturer: USB
...
Jan 25 16:08:01 host2 kernel: sd 18:0:0:0: [sdb] Write Protect is off
Jan 25 16:08:01 host2 kernel: sd 18:0:0:0: [sdb]
    Assuming drive cache: write through
Jan 25 16:08:01 host2 kernel: sdb: sdb1
Jan 25 16:08:01 host2 kernel: sd 18:0:0:0: [sdb]
    Attached SCSI removable disk
```

б) подключите USB-накопитель, который автоматически монтирует файловую систему с этого диска. Если этого не происходит, выполните на втором терминале следующие команды (от имени суперпользователя), чтобы создать каталог точки монтирования и смонтировать устройство:

```
$ mkdir /mnt/test
$ mount /dev/sdb1 /mnt/test
$ umount /dev/sdb1
```

8. Чтобы узнать, какие USB-устройства подключены к вашему компьютеру, введите команду:

```
$ lsusb
```

9. Чтобы загрузить модуль `bttv`, укажите модули, которые уже были загружены, и загрузите его, введя следующее:

```
# modprobe -a bttv
# lsmod | grep bttv
ttv                   167936  0
tea575x                16384  1 bttv
tveeprom               28672  1 bttv
videobuf_dma_sg        24576  1 bttv
videobuf_core          32768  2 videobuf_dma_sg,bttv
v4l2_common            16384  1 bttv
videodev              233472  3 tea575x,v4l2_common,bttv
i2c_algo_bit           16384  1 bttv
```

Обратите внимание на то, что при загрузке `bttv` другие модули (`v4l2_common`, `videodev` и др.) также были загружены с помощью команды `modprobe -a`.

10. Введите следующие команды, чтобы удалить модуль `bttv` с модулями, которые были загружены вместе с ним. Убедитесь, что все они исчезли после запуска команды `modprobe -r`:

```
# modprobe -r bttv
# lsmod | grep bttv
```

Глава 9. Установка Linux

1. Чтобы установить систему Fedora с помощью «живого» носителя Fedora Live media, следуйте инструкциям, приведенным в разделе «Установка Fedora с “живого” носителя» главы 9. Инструкция включает в себя следующие шаги:
   а) загрузку «живого» носителя;
   б) выбор загрузки на жесткий диск при запуске системы;
   в) добавление информации с общей страницы, необходимой для первоначальной настройки системы;
   г) перезагрузку компьютера и удаление «живого» носителя, чтобы вновь установленная система загрузилась с жесткого диска.
2. Чтобы обновить пакеты после завершения установки Fedora Live, выполните следующие действия:
   а) перезагрузите компьютер и ответьте на первые вопросы загрузки;
   б) используя проводное или беспроводное соединение, убедитесь, что вы подключены к Интернету. Обратитесь к главе 14 «Администрирование сети», если возникли проблемы с подключением к Интернету. Откройте оболочку от имени суперпользователя и введите команду `sudo dnf update`;
   в) при появлении приглашения введите `y`, чтобы принять появившийся список пакетов. Система начнет загружать и устанавливать их.
3. Чтобы запустить установку системы RHEL в текстовом режиме, выполните следующие действия:
   а) загрузите DVD-носитель RHEL;
   б) при появлении меню загрузки выделите один из вариантов загрузки установки и нажмите клавишу Tab. Переместите курсор вправо, в конец строки ядра, и введите там параметр `ex`. Нажмите клавишу Enter, чтобы запустить программу установки;
   в) попробуйте выполнить остальную часть установки в текстовом режиме.
4. Чтобы настроить разделы диска для установки DVD Red Hat Enterprise Linux, выполните следующие действия:
   а) в компьютер, на котором можно стереть не менее 10 Гбайт дискового пространства, вставьте установочный DVD RHEL, перезагрузите систему и используйте установочные экраны;

б) перейдя к экрану Installation Summary (Обзор установки), выберите пункт Installation Destination (Местоположение установки);

в) в окне Installation Destination (Местоположение установки) выберите устройство для установки (устройство `sda`, если у вас один жесткий диск, который можно полностью стереть, или `vda` для виртуальной установки);

г) нажмите кнопку Custom (Вручную);

д) нажмите кнопку Done (Готово), чтобы перейти к экрану создания разделов;

е) если существующее дисковое пространство уже занято, необходимо удалить разделы;

ж) нажмите кнопку с плюсом (+) в нижней части экрана. Затем добавьте каждую из следующих точек монтирования:

```
/boot - 400M
/ - 3G
/var - 2G
/home -2G
```

з) нажмите кнопку Done (Готово). Появится обзор изменений;

и) если изменения подходят, нажмите кнопку Accept Changes (Принять изменения). Если вы практикуетесь и на самом деле не хотите менять свои разделы, нажмите кнопку Cancel & Return (Отменить и вернуться) к пользовательскому разделению. Затем выйдите из программы установки.

ПРИМЕЧАНИЕ

Этот процесс в конечном итоге удалит все содержимое жесткого диска. Если вы хотите использовать упражнение, чтобы попрактиковаться в разбиении на разделы, можете перезагрузить компьютер перед началом фактического процесса установки, не повредив жесткий диск. После того как вы перейдете дальше и сохраните разделение, все данные будут удалены.

Глава 10. Управление программами

1. Чтобы найти в репозитории YUM пакет, предоставляющий команду `mogrify`, введите следующее:

```
# yum provides mogrify
```

2. Чтобы отобразить информацию о пакете, предоставляющем команду `mogrify`, и определить, какая у него домашняя страница (URL), введите следующее:

```
# yum info ImageMagick
```

Вы увидите, что URL-адрес домашней страницы ImageMagick — это www.imagemagick.org.

3. Чтобы установить пакет, содержащий команду `mogrify`, введите следующее:

```
# yum install ImageMagick
```

4. Чтобы перечислить все файлы документации, содержащиеся в пакете команды `mogrify`, введите следующее:

```
# rpm -qd ImageMagick
...
/usr/share/doc/ImageMagick/README.txt
...
/usr/share/man/man1/identify.1.gz
/usr/share/man/man1/import.1.gz
/usr/share/man/man1/mogrify.1.gz
```

5. Чтобы просмотреть журнал изменений пакета команды `mogrify`, введите следующее:

```
# rpm -q --changelog ImageMagick | less
```

6. Чтобы удалить команду `mogrify` из системы и проверить ее пакет по базе данных RPM, убедившись, что эта команда действительно удалена, введите следующее:

```
# type mogrify
mogrify is /usr/bin/mogrify
# rm /usr/bin/mogrify
rm remove regular file '/usr/bin/mogrify'? y
# rpm -V ImageMagick
missing   /usr/bin/mogrify
```

7. Чтобы переустановить пакет, предоставляющий команду `mogrify`, и убедиться, что весь пакет снова установлен, введите следующее:

```
# yum reinstall ImageMagick
# rpm -V ImageMagick
```

8. Чтобы загрузить пакет, предоставляющий команду `mogrify`, в текущий каталог, введите следующее:

```
# yum download ImageMagick
ImageMagick-6.9.10.28-1.fc30.x86_64.rpm
```

9. Чтобы отобразить общую информацию о пакете, который вы только что загрузили, запросив файл RPM пакета в текущем каталоге, введите следующее:

```
# rpm -qip ImageMagick-6.9.10.28-1.fc30.x86_64.rpm
Name        : ImageMagick
Epoch       : 1
Version     : 6.9.10.28
Release     : 1.fc30



...
```

10. Чтобы удалить пакет, содержащий команду `mogrify`, из своей системы, введите следующее:

```
# yum remove ImageMagick
```

Глава 11. Управление учетными записями

Для выполнения упражнений, связанных с добавлением и удалением учетных записей пользователей, можно применять окно Users (Пользователи), программу User Manager (Менеджер пользователей) или инструменты командной строки, например команды `useradd` и `usermod`. Чтобы получить те же результаты, которые даны в ответах, не обязательно выполнять упражнения только описанным здесь способом.

Существует множество путей достижения одних и тех же результатов. Приведенные далее ответы показывают, как выполнить упражнения из командной строки. (Станьте суперпользователем, когда увидите приглашение.)

1. Чтобы добавить локальную учетную запись пользователя в систему Linux с именем `jbaxter` и полным именем `John Baxter`, которая применяет `/bin/sh` в качестве оболочки по умолчанию и имеет следующий доступный UID (ваш может отличаться от показанного), введите следующее. Можете применить команду `grep` для проверки новой учетной записи пользователя. Затем установите пароль `My1N1te0ut` для пользователя `jbaxter`:

```
# useradd -c "John Baxter" -s /bin/sh jbaxter
# grep jbaxter /etc/passwd
jbaxter:x:1001:1001:John Baxter:/home/jbaxter:/bin/sh
# passwd jbaxter
Changing password for user jbaxter
New password: My1N1te0ut!
Retype new password: My1N1te0ut!
passwd: all authentication tokens updated successfully
```

2. Чтобы создать учетную запись группы с именем `testing`, использующую идентификатор группы `315`, введите следующее:

```
# groupadd -g 315 testing
# grep testing /etc/group
testing:x:315:
```

3. Чтобы добавить `jbaxter` в группу `testing` и группу `bin`, введите следующее:

```
# usermod -aG testing,bin jbaxter
# grep jbaxter /etc/group
bin:x:1:bin,daemon,jbaxter
jbaxter:x:1001:
testing:x:315:jbaxter
```

4. Чтобы стать пользователем jbaxter и временно сделать группу testing группой jbaxter по умолчанию, запустите файл touch /home/jbaxter/file.txt так, чтобы группа testing была назначена в качестве его группы:

```
$ su - jbaxter
Password: My1N1te0ut!
sh-4.2$ newgrp testing
sh-4.2$ touch /home/jbaxter/file.txt
sh-4.2$ ls -l /home/baxter/file.txt
-rw-rw-r--. 1 jbaxter testing 0 Jan 25 06:42 /home/jbaxter/file.txt
sh-4.2$ exit ; exit
```

5. Обратите внимание на то, какой идентификатор пользователя был назначен пользователю jbaxter, а затем удалите учетную запись пользователя, не удаляя домашний каталог, назначенный для jbaxter:

```
$ userdel jbaxter
```

6. Примените следующую команду, чтобы найти в каталоге /home (и любых подкаталогах) все файлы, назначенные идентификатору пользователя, который недавно принадлежал пользователю с именем jbaxter. (У меня оба идентификатора, UID и GID, были 1001, ваши могут отличаться.) Обратите внимание на то, что имя пользователя jbaxter больше не назначается в системе, поэтому любые файлы, созданные пользователем, перечислены как принадлежащие к UID 1001 и GID 1001, за исключением нескольких файлов, которые были назначены группе testing из-за команды newgrp, выполненной ранее:

```
# find /home -uid 1001 -ls
262184  4 drwx------ 4 1001  1001  4096 Jan 25 08:00 /home/jbaxter
262193  4 -rw-r--r-- 1 1001  1001   176 Jan 27  2011 /home/jbaxter/.bash_
profile
262196  4 -rw------- 1 13602 testing 93 Jan 25 08:00 /home/jbaxter/.bash_
history
262194  0 -rw-rw-r-- 1 13602 testing  0 Jan 25 07:59 /home/jbaxter/file.txt
...
```

7. Выполните приведенные далее команды, чтобы скопировать файл /etc/services в каталог /etc/skel/, затем добавьте в систему нового пользователя с именем mjones, полным именем Mary Jones и домашним каталогом /home/maryjones. Укажите ее домашний каталог, чтобы убедиться, что файл служб находится там:

```
# cp /etc/services /etc/skel/
# useradd -d /home/maryjones -c "Mary Jones" mjones
# ls -l /home/maryjones
total 628
-rw-r--r--. 1 mjones mjones 640999 Jan 25 06:27 services
```

8. Выполните следующую команду, чтобы найти в каталоге /home все файлы, принадлежащие mjones. Если вы выполняли упражнения по порядку, обратите внимание на то, что после удаления пользователя с самыми высокими

ID пользователя и группы его номера были присвоены пользователю mjones. В результате все файлы, оставленные в системе jbaxter, теперь принадлежат mjones (по этой причине вы должны удалить или изменить владельца файлов, оставшихся после удаления пользователя):

```
# find /home -user mjones -ls
262184 4 drwx------ 4 mjones mjones 4096 Jan 25 08:00 /home/jbaxter
262193 4 -rw-r--r-- 1 mjones mjones 176 Jan 27 2011 /home/jbaxter/.bash_
profile
262189 4 -rw-r--r-- 1 mjones mjones 18 Jan 27 2011 /home/jbaxter/.bash_
logout
262194 0 -rw-rw-r-- 1 mjones testing 0 Jan 25 07:59 /home/jbaxter/file.txt
262188 4 -rw-r--r-- 1 mjones mjones 124 Jan 27 2011 /home/jbaxter/.bashrc
262197 4 drwx------ 4 mjones  mjones 4096 Jan 25 08:27 /home/maryjones
262207 4 -rw-r--r-- 1 mjones mjones 176 Jan 27 2011 /home/maryjones/.bash_
profile
262202 4 -rw-r--r-- 1 mjones mjones 18 Jan 27 2011 /home/maryjones/.bash_
logout
262206 628 -rw-r--r-- 1 mjones mjones 640999 Jan 25 08:27 /home/maryjones/
services
262201 4 -rw-r--r-- 1 mjones mjones 124 Jan 27 2011 /home/maryjones/.bashrc
```

9. От имени пользователя mjones вы можете применить код, приведенный далее, чтобы создать файл с именем /tmp/mary-file.txt и с помощью ACL назначить этому файлу права пользователя bin на чтение/запись и права группы lp на чтение/запись:

```
[mjones]$ touch /tmp/maryfile.txt
[mjones]$ setfacl -m u:bin:rw /tmp/maryfile.txt
[mjones]$ setfacl -m g:lp:rw /tmp/maryfile.txt
[mjones]$ getfacl /tmp/maryfile.txt
# file: tmp/maryfile.txt
# owner: mjones
# group: mjones
user::rw-
user:bin:rw-
group::rw-
group:lp:rw-
mask::rw-
other::r& –
```

10. Выполните от имени пользователя mjones набор команд, чтобы создать каталог с именем /tmp/mydir и применить ACL для назначения ему прав по умолчанию, чтобы пользователь adm имел права на чтение/запись/выполнение этого каталога и любых файлов или каталогов, созданных в нем. Проверьте, сработал ли набор команд, создав каталог /tmp/mydir/testing/ и файл /tmp/mydir/newfile.txt:

```
[mary]$ mkdir /tmp/mydir
[mary]$ setfacl -m d:u:adm:rwx /tmp/mydir
[mjones]$ getfacl /tmp/mydir
```

```
# file: tmp/mydir
# owner: mjones
# group: mjones
user::rwx
group::rwx
other::r-x
default:user::rwx
default:user:adm:rwx
default:group::rwx
default:mask::rwx
default:other::r-x
[mjones]$ mkdir /tmp/mydir/testing
[mjones]$ touch /tmp/mydir/newfile.txt
[mjones]$ getfacl /tmp/mydir/testing/
# file: tmp/mydir/testing/
# owner: mjones
# group: mjones
user::rwx
user:adm:rwx
group::rwx
mask::rwx
other::r-x
default:user::rwx
default:user:adm:rwx
default:group::rwx
default:mask::rwx
default:other::r-x
[mjones]$ getfacl /tmp/mydir/newfile.txt
# file: tmp/mydir/newfile.txt
# owner: mjones
# group: mjones
user::rw-
user:adm:rwx     #effective:rwg-
roup::rwx        #effective:rw-
mask::rw-
other::r--
```

Обратите внимание на то, что пользователь `adm` фактически имеет только права на запись. Чтобы исправить это, нужно расширить права доступа маски. Один из способов сделать это — применить команду `chmod` следующим образом:

```
[mjones]$ chmod 775 /tmp/mydir/newfile.txt
[mjones]$ getfacl /tmp/mydir/newfile.txt
# file: tmp/mydir/newfile.txt
# owner: mjones
# group: mjones
user::rwx
user:adm:rwx
group::rwx
mask::rwx
other::r-x
```

Глава 12. Управление дисками и файлами

1. Чтобы определить имя устройства USB-накопителя, введите следующее и вставьте USB-накопитель (нажмите сочетание клавиш Ctrl+C после того, как увидите соответствующие сообщения):

```
# journalctl -f
kernel: [sdb] 15667200 512-byte logical blocks:
     (8.02 GB/7.47 GiB)
Feb 11 21:55:59 cnegus kernel: sd 7:0:0:0:
     [sdb] Write Protect is off
Feb 11 21:55:59 cnegus kernel: [sdb] Assuming
     drive cache: write through
Feb 11 21:55:59 cnegus kernel: [sdb] Assuming
     drive cache: write through
```

2. Чтобы перечислить разделы на флеш-накопителе USB в системе RHEL 6, введите следующую команду:

```
# fdisk -c -u -l /dev/sdb
```

   Чтобы перечислить разделы в системе RHEL 7, RHEL 8 или Fedora, введите:

```
# fdisk -l /dev/sdb
```

3. Чтобы удалить разделы на накопителе USB, предположив, что это `device/dev/sdb`, выполните следующие действия:

```
# fdisk /dev/sdb
Command (m for help): d
Partition number (1-6): 6
Command (m for help): d
Partition number (1-5): 5
Command (m for help): d
Partition number (1-5): 4
Command (m for help): d
Partition number (1-4): 3
Command (m for help): d
Partition number (1-4): 2
Command (m for help): d
Selected partition 1
Command (m for help): w
# partprobe /dev/sdb
```

4. Чтобы добавить на USB-накопитель Linux-раздел объемом 100 Мбайт, раздел подкачки объемом 200 Мбайт и LVM-раздел объемом 500 Мбайт, введите следующее:

```
# fdisk /dev/sdb

Command (m for help): n
Command action
```

Б

```
   e    extended
   p    primary partition (1-4)
p
Partition number (1-4): 1
First sector (2048-15667199, default 2048):  <ENTER>
Last sector, +sectors or +size{K,M,G} (default 15667199): +100M
Command (m for help): n
Command action
   e    extended
   p    primary partition (1-4)
p
Partition number (1-4): 2
First sector (616448-8342527, default 616448):  <ENTER>
Last sector, +sectors or +size{K,M,G} (default 15667199): +200M
Command (m for help): n
Command action
   e    extended
   p    primary partition (1-4)
p
Partition number (1-4): 3
First sector (616448-15667199, default 616448):  <ENTER>
Using default value 616448
Last sector, +sectors or +size{K,M,G} (default 15667199): +500M
Command (m for help): t
Partition number (1-4): 2
Hex code (type L to list codes): 82
Changed system type of partition 2 to 82 (Linux swap / Solaris)
Command (m for help): t
Partition number (1-4): 3
Hex code (type L to list codes): 8e
Changed system type of partition 3 to 8e (Linux LVM)
Command (m for help): w
# partprobe /dev/sdb
# grep sdb /proc/partitions
   8       16    7833600 sdb
   8       17     102400 sdb1
   8       18     204800 sdb2
   8       19     512000 sdb3
```

5. Чтобы поместить файловую систему `ext4` в раздел Linux, введите следующее:

```
# mkfs -t ext4 /dev/sdb1
```

6. Чтобы создать точку монтирования с именем `/mnt/mypart` и смонтировать на ней раздел Linux, выполните следующее:

```
# mkdir /mnt/mypart
# mount -t ext4 /dev/sdb1 /mnt/mypart
```

7. Чтобы включить раздел подкачки таким образом, чтобы дополнительное пространство подкачки сразу же стало доступно, введите следующее:

```
# mkswap /dev/sdb2
# swapon /dev/sdb2
```

8. Чтобы создать группу томов `abc` из раздела LVM, создайте логический том объемом 200 Мбайт из группы `data`, на нем — файловую систему VFAT, временно смонтируйте логический том в новом каталоге `/mnt/test`, а затем убедитесь, что он был успешно смонтирован:

```
# pvcreate /dev/sdb3
# vgcreate abc /dev/sdb3
# lvcreate -n data -L 200M abc
# mkfs -t vfat /dev/mapper/abc-data
# mkdir /mnt/test
# mount /dev/mapper/abc-data /mnt/test
```

9. Чтобы увеличить объем логического тома с 200 до 300 Мбайт, введите следующее:

```
# lvextend -L +100M /dev/mapper/abc-data
# resize2fs -p /dev/mapper/abc-data
```

10. Чтобы безопасно извлечь USB-накопитель из компьютера, выполните следующее:

```
# umount /dev/sdb1
# swapoff /dev/sdb2
# umount /mnt/test
# lvremove /dev/mapper/abc-data
# vgremove abc
# pvremove /dev/sdb3
```

Теперь можете безопасно извлечь USB-накопитель из компьютера.

Глава 13. Администрирование серверов

1. Чтобы войти в любую учетную запись на другом компьютере с помощью команды `ssh`, введите приведенную далее команду и пароль при появлении запроса:

```
$ ssh joe@localhost
joe@localhost's password:
*********
[joe]$
```

2. Чтобы отобразить содержимое удаленного файла `/etc/system-release` в локальной системе с помощью команды `ssh`, выполните следующее:

```
$ ssh joe@localhost "cat /etc/system-release"
joe@localhost's password: *******
Fedora release 30 (Thirty)
```

3. Чтобы использовать переадресацию X11 для отображения окна `gedit` в локальной системе, а затем сохранить файл в удаленном домашнем каталоге, выполните следующие действия:

```
$ ssh -X joe@localhost "gedit newfile"
joe@localhost's password: ********
```

```
$ ssh joe@localhost "cat newfile"
joe@localhost's password: ********
This is text from the file I saved in joe's remote home directory
```

4. Чтобы рекурсивно скопировать все файлы из каталога `/usr/share/selinux` удаленной системы в каталог `/tmp` локальной системы таким образом, чтобы все время изменения файлов обновлялось до времени их копирования в локальной системе, выполните следующее:

```
$ scp -r joe@localhost:/usr/share/selinux /tmp
joe@localhost's password:
 ********
irc.pp.bz2                            100% 9673     9.5KB/s   00:00
dcc.pp.bz2                            100%   15KB  15.2KB/s   00:01
$ ls -l /tmp/selinux | head
total 20
drwxr-xr-x. 3 root root  4096 Apr 18 05:52 devel
drwxr-xr-x. 2 root root  4096 Apr 18 05:52 packages
drwxr-xr-x. 2 root root 12288 Apr 18 05:52 targeted
```

5. Чтобы рекурсивно скопировать все файлы из каталога `/usr/share/logwatch` удаленной системы в каталог `/tmp` локальной системы таким образом, чтобы все время изменения файлов из удаленной системы сохранялось в локальной, выполните следующие действия:

```
$ rsync -av joe@localhost:/usr/share/logwatch /tmp
joe@localhost's password: ********
receiving incremental file list
logwatch/
logwatch/default.conf/
logwatch/default.conf/logwatch.conf
$ ls -l /tmp/logwatch | head
total 16
drwxr-xr-x. 5 root root 4096 Apr 19  2011 default.conf
drwxr-xr-x. 4 root root 4096 Feb 28  2011 dist.conf
drwxr-xr-x. 2 root root 4096 Apr 19  2011 lib
```

6. Чтобы создать пару «открытый/закрытый ключ» для применения для SSH-связи (без ключевой фразы), скопируйте файл открытого ключа в учетную запись удаленного пользователя с помощью команды `ssh-copy-id` и примените аутентификацию на основе ключа для входа в эту учетную запись пользователя без ввода пароля:

```
$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/joe/.ssh/id_rsa): ENTER
/home/joe/.ssh/id_rsa already exists.
Enter passphrase (empty for no passphrase):  ENTER
Enter same passphrase again:  ENTER
Your identification has been saved in /home/joe/.ssh/id_rsa.
Your public key has been saved in /home/joe/.ssh/id_rsa.pub.
```

```
The key fingerprint is:
58:ab:c1:95:b6:10:7a:aa:7c:c5:ab:bd:f3:4f:89:1e joe@cnegus.csb
The key's randomart image is:
...
$ ssh-copy-id -i ~/.ssh/id_rsa.pub joe@localhost
joe@localhost's password: ********
Now try logging into the machine, with "ssh 'joe@localhost'",
and check in:
.ssh/authorized_keys
to make sure we haven't added extra keys that you weren't expecting.
$ ssh joe@localhost
$ cat .ssh/authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEAyN2Psp5/LRUC9E8BDCx53yPUa0qoOPd
v6H4sF3vmn04V6E7D1iXpzwPzdo4rpvmR1ZiinHR2xGAEr2uZag7feKgLnww2KPcQ6S
iR7lzrOhQjV+SGb/a1dxrIeZqKMq1Tk07G4EvboIrq//9J47vI4l7iNu0xRmjI3TTxa
DdCTbpG6J3uSJm1BKzdUtwb413x35W2bRgMI75aIdeBsDgQBBiOdu+zuTMrXJj2viCA
XeJ7gIwRvBaMQdOSvSdlkX353tmIjmJheWdgCccM/1jKdoELpaevg9anCe/yUP3so31
tTo4I+qTfzAQD5+66oqW0LgMkWVvfZI7dUz3WUPmcMw== chris@abc.example.com
```

7. Чтобы создать в файле `/etc/rsyslog.conf` запись, которая хранит все сообщения аутентификации на уровне информации и выше в файле с именем `/var/log/myauth`, выполните показанные далее действия. Наблюдайте с терминала, как поступают данные:

```
# vim /etc/rsyslog.conf
authpriv.info                                /var/log/myauth
# service rsyslog restart
or
# systemctl restart rsyslog.service
<Terminal 1> <Terminal 2>
# tail -f /var/log/myauth                  $ ssh joe@localhost
Apr 18 06:19:34 abc unix_chkpwd[30631]     joe@localhost's password:
Apr 18 06:19:34 abc sshd[30631]            Permission denied,try again
 :pam_unix(sshd:auth):
 authentication failure;logname= uid=501
 euid=501 tty=ssh ruser= rhost=localhost
 user=joe
Apr 18 06:19:34 abc sshd[30631]:
 Failed password for joe from
 127.0.0.1 port 5564 ssh2
```

8. Чтобы определить самые большие структуры каталогов в каталоге `/usr/share`, отсортировать их от самых больших к самым маленьким и перечислить десять самых больших с помощью команды `du`, введите следующее:

```
$ du -s /usr/share/* | sort -rn | head
527800 /usr/share/locale
277108 /usr/share/fonts
196232 /usr/share/help

134984 /usr/share/backgrounds
...
```

9. Чтобы отобразить пространство, которое используется и доступно для всех файловых систем, подключенных в данный момент к локальной системе, но исключить любые файловые системы `tmpfs` или `devtmpfs` с помощью команды `df`, введите следующее:

```
$ df -h -x tmpfs -x devtmpfs
Filesystem      Size  Used Avail Use% Mounted on
/deev/sda4       20G  4.2G 16G    22% /
```

10. Чтобы найти в каталоге `/usr` любые файлы размером более 10 Мбайт, выполните следующие действия:

```
$ find /usr -size +10M
/usr/lib/locale/locale-archive
/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.212.b04-0.fc30.x86_64/jre/lib/rt.jar
/usr/libexec/cni/dhcp
/usr/libexec/gdb
/usr/libexec/gcc/x86_64-redhat-linux/9/lto1
/usr/libexec/gcc/x86_64-redhat-linux/9/cc1
```

Глава 14. Администрирование сети

1. Чтобы с помощью рабочего стола убедиться, что программа NetworkManager успешно запустила сетевой интерфейс (проводной или беспроводной), выполните следующие действия:
   а) щелкните левой кнопкой мыши в правом верхнем углу рабочего стола GNOME, чтобы открыть выпадающий список. В этом меню должны отображаться все активные проводные или беспроводные сетевые подключения;
   б) если система не подключена к сети, выберите нужную сеть из списка доступных, а затем, если появится запрос, введите имя пользователя и пароль, чтобы подключиться.
2. Чтобы выполнить команду для проверки активных сетевых интерфейсов, доступных на вашем компьютере, введите:

```
$ ifconfig
```

   или

```
$ ip addr show
```

3. Попробуйте связаться с google.com из командной строки таким образом, чтобы служба DNS работала правильно:

```
$ ping google.com
Ctrl-C
```

4. Чтобы выполнить команду для проверки маршрутов, используемых для связи за пределами локальной сети, введите следующее:

```
$ route
```

5. Чтобы увидеть маршрут, по которому идет соединение с google.com, примените команду `traceroute`:

```
$ traceroute google.com
```

6. Чтобы просмотреть сетевые интерфейсы и связанные с ними сетевые действия системы Linux через интерфейс Cockpit, откройте браузер на порте 9090, используя IP-адрес или имя хоста, например: localhost:9090/network.

7. Создайте запись о хосте, которая позволит общаться с локальной хост-системой, с помощью имени `myownhost`, отредактируйте файл `/etc/hosts` (`vi /etc/hosts`) и добавьте `myownhost` в конец строки `localhost`, чтобы он выглядел следующим образом (затем пропингуйте файл `myownhost`, чтобы увидеть, заработал ли он):

```
127.0.0.1         localhost.localdomain localhost myownhost
# ping myownhost
Ctrl+C
```

8. Чтобы увидеть DNS-серверы имен, используемые для добавления имен хостов и IP-адресов в вашей системе (ваши будут отличаться от показанных здесь), введите следующее:

```
# cat /etc/resolv.conf
    nameserver 10.83.14.9
    nameserver 10.18.2.10
    nameserver 192.168.1.254
    # dig google.com
    ...
    google.com.     91941   IN     NS     ns3.google.com.
    ;; Query time: 0 msec
    ;; SERVER: 10.18.2.9#53(10.18.2.9)
    ;; WHEN: Sat Nov 23 20:18:56 EST 2019
    ;; MSG SIZE  rcvd: 276
```

9. Чтобы создать пользовательский маршрут, который направляет трафик, предназначенный для сети 192.168.99.0/255.255.255.0, на некоторый IP-адрес в вашей локальной сети, например 192.168.0.5 (сначала убедитесь, что сеть 192.168.99 не применяется в вашей сети), выполните следующие действия:

   а) определите имя вашего сетевого интерфейса, например `enp4s0`. В этом случае от имени суперпользователя выполните следующие команды:

```
# cd /etc/sysconfig/network-scripts
# vi route-enp4s0
```

   б) добавьте к файлу следующие строки:

```
ADDRESS0=192.168.99.0
NETMASK0=255.255.255.0
GATEWAY0=192.168.0.5
```

   в) перезапустите сеть и введите команду `route`, чтобы проверить активность маршрута:

```
# systemctl restart NetworkManager
# route -n
```

```
Kernel IP routing table
Destination   Gateway       Genmask         Flags Metric Ref Use Iface
192.168.0.1   0.0.0.0       255.255.255.0   U     600    0     0 enp4s0
192.168.99.0  192.168.0.5   255.255.255.0   UG    600    0     0 enp4s0
```

10. Чтобы проверить, настроена ли ваша система на маршрутизацию пакетов IPv4 между сетевыми интерфейсами, введите следующее:

```
# cat /proc/sys/net/ipv4/ip_forward
0
```

Значение `0` показывает, что переадресация пакетов IPv4 отключена, значение `1` — что включена.

Глава 15. Запуск и остановка служб

1. Чтобы определить, какой демон инициализации используется вашим сервером в данный момент, рассмотрим следующее:

   а) в большинстве случаев PID 1 задействуется как демон `systemd`:

```
# ps -ef | head
UID        PID  PPID  C STIME TTY       TIME CMD
root         1     0  0 17:01 ?     00:00:04 /usr/lib/systemd/systemd --
     switched-root --system --deserialize 18
```

   Если ввести команду `ps -ef` и PID 1 — это `init`, то это все равно может быть демон `systemd`. Задействуйте команду `strings`, чтобы узнать, используется ли демон `systemd` в системе:

```
# strings /sbin/init | grep -i systemd
systemd.unit=
systemd.log_target=
systemd.log_level=
...
```

   б) скорее всего, у вас есть демон инициализации Upstart, SysVinit или BSD, если ваш демон `init` не является `systemd`. Но лучше это дважды проверить на wikipedia.org/wiki/Init.

2. Инструменты, применяемые для управления службами, в первую очередь зависят от используемой системы инициализации. Запустите команды `systemctl` и `service`, чтобы определить тип сценария инициализации для службы `ssh` в вашей системе:

   а) для демона `systemd` положительный результат, показанный здесь, означает, что служба `sshd` была преобразована в `systemd`:

```
# systemctl status sshd.service
sshd.service - OpenSSH server daemon
  Loaded: loaded (/lib/systemd/system/sshd.service; enabled)
  Active: active (running) since Mon, 20 Apr 2020 12:35:20...
```

б) если после выполнения предыдущего пункта положительного результата не получено, выполните следующую команду для демона SysVinit `init`. Здесь положительный результат, как и отрицательные результаты предыдущих тестов, означает, что служба `sshd` все еще использует демон SysVinit:

```
# service ssh status
sshd (pid 2390) is running...
```

3. Чтобы определить предыдущий и текущий уровни выполнения вашего сервера, введите команду `runlevel`. Она по-прежнему работает на всех демонах `init`:

```
$ runlevel
N 3
```

4. Чтобы изменить уровень выполнения по умолчанию или целевой юнит на своем сервере Linux, реализуйте одно из следующих действий (в зависимости от демона `init`):
   а) для SysVinit отредактируйте файл `/etc/inittab` и замените `#` в строке `id:#:initdefault:` на `2`, `3`, `4` или `5`;
   б) для `systemd` измените значение `default.target` на необходимый уровень выполнения `runlevel1#.target`, где значение `#` равно `2`, `3`, `4` или `5`. Далее показано, как изменить целевой юнит на `runlevel3.target`:

```
# systemctl set-default runlevel3.target
Removed /etc/systemd/system/default.target.
Created symlink /etc/systemd/system/default.target →
  /usr/lib/systemd/system/multi-user.target.
```

5. Чтобы перечислить службы, запущенные (или активные) на вашем сервере, нужно использовать различные команды в зависимости от демона инициализации:
   а) для SysVinit — команду `service`, как показано в этом примере:

```
# service --status-all | grep running | sort
anacron (pid 2162) is running...
atd (pid 2172) is running...
```

   б) для `systemd` — команду `systemctl` следующим образом:

```
# systemctl list-unit-files --type=service | grep -v disabled
UNIT FILE                                   STATE
abrt-ccpp.service                           enabled
abrt-oops.service                           enabled
...
```

6. Чтобы перечислить запущенные (или активные) службы на вашем сервере Linux, используйте соответствующую команду (команды) из ответа 5 для своего демона инициализации.
7. Для каждого демона инициализации текущее состояние конкретной службы показывают следующие команды:
   а) для SysVinit — команда `service` *service_name* `status`;
   б) для `systemd` — команды `systemctl status` *service_name*.

8. Чтобы отобразить состояние демона cups на своем сервере Linux, используйте следующую команду:

   а) для SysVinit:

   ```
   # service cups status
   cupsd (pid 8236) is running...
   ```

   б) для systemd:

   ```
   # systemctl status cups.service
   cups.service - CUPS Printing Service
   Loaded: loaded (/lib/systemd/system/cups.service; enabled)
   Active: active (running) since Tue, 05 May 2020 04:43:5...
   Main PID: 17003 (cupsd)
   CGroup: name=systemd:/system/cups.service
   17003 /usr/sbin/cupsd -f
   ```

9. Чтобы перезапустить демон cups на своем Linux-сервере, выполните следующие действия:

   а) для SysVinit:

   ```
   # service cups restart
   Stopping cups:          [  OK  ]
   ```

   б) для systemd:

   ```
   # systemctl restart cups.service
   ```

10. Чтобы перезагрузить демон cups на своем Linux-сервере, выполните следующие действия:

    а) для SysVinit:

    ```
    # service cups reload
    Reloading cups: [ OK ]
    ```

    б) а вот для systemd это не так просто. Нельзя перезагрузить демон cups на сервере systemd:

    ```
    # systemctl reload cups.service
    Failed to issue method call: Job type reload is
      not applicable for unit cups.service.
    ```

Глава 16. Настройка сервера печати

Для вопросов, связанных с принтерами, в большинстве случаев можно использовать графические средства или инструменты командной строки. В упражнениях важно убедиться, что вы получили те же результаты, какие показаны в ответах далее. Ответы включают в себя сочетание графических и командных способов решения упражнений. (Станьте суперпользователем, когда увидите приглашение #.)

1. Используйте окно Print Settings (Настройки принтера), чтобы добавить в систему новый принтер `myprinter` (универсальный принтер PostScript, подключенный к порту) в дистрибутиве Fedora 30:
   а) установите пакет `system-config-printer`:

      ```
      # dnf install system-config-printer
      ```

   б) с рабочего стола GNOME 2 выберите программу Print Settings (Настройки принтера) в окне Acivities (Приложения);
   в) разблокируйте интерфейс и введите пароль суперпользователя;
   г) нажмите кнопку Add (Добавить);
   д) выберите USB-носитель или другой порт в качестве устройства и нажмите кнопку Forward (Далее);
   е) для драйвера выберите вариант Generic (Общие) и нажмите кнопку Forward (Далее). Выберите PostScript и нажмите Forward (Далее);
   ж) нажмите кнопку Forward (Далее), чтобы пропустить настройки установки;
   з) назовите принтер `myprinter`, дайте ему любое подходящее описание, укажите место расположения и нажмите кнопку Apply (Применить);
   и) нажмите кнопку Cancel (Отмена), чтобы принтер не распечатывал пробную страницу. Новый принтер должен появиться в окне Print Settings (Настройки принтера).
2. Используйте команду `lpstat -t` для просмотра состояния всех ваших принтеров:

   ```
   # lpstat -t
   deskjet-5550 accepting requests since Mon 02 Mar 2020 07:30:03 PM EST
   ```

3. Примените команду `lpr`, чтобы распечатать файл `/etc/hosts`:

   ```
   $ lp /etc/hosts -P myprinter
   ```

4. Чтобы проверить очередь на печать для этого принтера, введите следующее:

   ```
   # lpq -P myprinter
   myprinter is not ready
   Rank    Owner   Job     File(s)                 Total Size
   1st     root    655     hosts                   1024 bytes
   ```

5. Чтобы удалить задачу из очереди на печать (отменить ее), введите следующее:

   ```
   # lprm -P myprinter
   ```

6. Чтобы использовать окно печати для установки основных параметров сервера печати так, чтобы другие системы в вашей локальной сети могли печатать на них, выполните следующие действия:
   а) на рабочем столе GNOME 3 на экране Acivities (Приложения) введите Настройки принтера и нажмите клавишу Enter;
   б) выберите Server ▸ Server Settings (Сервер ▸ Настройки сервера) и введите пароль суперпользователя, если появится запрос;

в) установите флажок Publish shared printers connected to this system (Показывать общие принтеры, подключенные к этой системе) и нажмите кнопку OK.

7. Чтобы разрешить удаленное администрирование системы из браузера, выполните следующие действия:
   а) на рабочем столе GNOME 3 на экране Acivities (Приложения) введите Настройки принтера и нажмите клавишу Enter;
   б) выберите Server ▸ Server Settings (Сервер ▸ Настройки сервера) и введите пароль суперпользователя, если появится запрос;
   в) установите флажок Allow remote administration (Разрешить удаленное администрирование) и нажмите кнопку OK.
8. Чтобы проверить, можете ли вы выполнять удаленное администрирование своей системы из браузера в другой системе, выполните следующие действия:
   а) в адресной строке браузера с другого компьютера в сети введите *hostname*`:631`, заменив *hostname* именем или IP-адресом системы, в которой запущена служба печати;
   б) введите `root` как пользователь и пароль суперпользователя при появлении запроса. Появится домашняя страница CUPS.
9. Чтобы с помощью команды `netstat` узнать, по каким адресам прослушивается демон `cupsd`, введите следующее:

```
# netstat -tupln | grep 631
Tcp    0    0 0.0.0.0:631     0.0.0.0:*     LISTEN   6492/cupsd
tcp6   0    0 :::631          :::*          LISTEN   6492/cupsd
```

10. Чтобы удалить принтер `myprinter` из своей системы, выполните следующие действия:
    а) нажмите кнопку Unlock (Разблокировать) и введите пароль суперпользователя при появлении запроса;
    б) в окне Print Settings (Настройки принтера) дважды нажмите на значке принтера `myprinter` и выберите вариант Delete (Удалить);
    в) подтвердите удаление.

Глава 17. Настройка веб-сервера

1. Чтобы установить все пакеты, связанные с группой Web Server в системе Fedora, выполните следующие действия:

```
# yum groupinstall "Web Server"
```

2. Чтобы создать файл `index.html` в каталоге, назначенном `DocumentRoot` в главном файле конфигурации Apache (с фразой `My Own Web Server` внутри), выполните следующие действия:

а) определите местоположение каталога `DocumentRoot`:

```
# grep ^DocumentRoot /etc/httpd/conf/httpd.conf
DocumentRoot "/var/www/html"
```

б) добавьте фразу `My Own Web Server` в файл `index.html`, расположенный в каталоге `DocumentRoot`:

```
# echo "My Own Web Server" > /var/www/html/index.html
```

3. Чтобы запустить веб-сервер Apache и настроить его на автоматический запуск во время загрузки, а затем проверить, доступен ли он из браузера на локальном хосте, выполните некоторые действия (если он работает правильно, вы должны увидеть фразу `My Own Web Server`).

   В разных системах Linux служба `httpd` запускается и включается по-разному. В системах Fedora 30 и выше и RHEL 7 или 8 введите следующее:

```
# systemctl start httpd.service
# systemctl enable httpd.service
```

   В системах RHEL 6 и ранее введите:

```
# service httpd start
# chkconfig httpd on
```

4. Чтобы с помощью команды `netstat` узнать, на каких портах прослушивается httpd-сервер, введите следующее:

```
# netstat -tupln | grep httpd
tcp6       0   0 :::80       :::*      LISTEN      2496/httpd
tcp6       0   0 :::443      :::*      LISTEN      2496/httpd
```

5. Подключитесь к веб-серверу Apache из браузера за пределами локальной системы. Если это не удается, исправьте ошибки, связанные с брандмауэром, SELinux и другими функциями безопасности.

   Если вы еще не настроили сервер DNS, используйте IP-адрес сервера для просмотра сервера Apache из удаленного браузера, например http://192.168.0.1. Если не можете подключиться, пытайтесь подключиться к серверу из браузера после выполнения каждого из следующих шагов в системе, на которой запущен сервер Apache:

```
# iptables -F
# setenforce 0
# chmod 644 /var/www/html/index.html
```

   Команда `iptables -F` временно сбрасывает правила брандмауэра. Если после этого подключение к веб-серверу завершится успешно, необходимо добавить новые правила брандмауэра, чтобы открыть TCP-порты 80 и 443 на сервере. В системе, использующей службу `firewalld`, для этого установите флажок рядом с портами в окне брандмауэра. Для систем, работающих под управлением

службы iptables, добавьте правила, как показано далее, перед последним правилом DROP или REJECT:

```
-A INPUT -m state --state NEW -m tcp -p tcp --dport 80 -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 443 -j ACCEPT
```

Команда setenforce 0 временно переводит SELinux в разрешительный режим. Если подключение к веб-серверу после этого завершится успешно, необходимо исправить контекст файла SELinux и/или логический тип (возможно, в данном случае только контекст файла). Должно сработать следующее:

```
# chcon --reference=/var/www/html /var/www/html/index.html
```

Если команда chmod работает, это означает, что пользователь и группа Apache не имели прав на чтение файла. Необходимо оставить эти права без изменений.

6. Чтобы применить openssl или аналогичную команду для создания собственного закрытого ключа RSA и самоподписанного SSL-сертификата, выполните следующие действия:

```
# yum install openssl
# cd /etc/pki/tls/private
# openssl genrsa -out server.key 1024
# chmod 600 server.key
# cd /etc/pki/tls/certs
# openssl req -new -x509 -nodes -sha1 -days 365 \
   -key /etc/pki/tls/private/server.key \
   -out server.crt
Country Name (2 letter code) [AU]: US
State or Province Name (full name) [Some-State]: NJ
Locality Name (eg, city) []: Princeton
Organization Name (eg, company) [Internet Widgits Pty
Ltd]:TEST USE ONLY
Organizational Unit Name (eg, section) []:TEST USE ONLY
Common Name (eg, YOUR name) []:secure.example.org
Email Address []:dom@example.org
```

Теперь у вас должны быть файл ключа /etc/pki/tls/private/server.key и файл сертификата /etc/pki/tls/certs/server.crt.

7. Чтобы настроить веб-сервер Apache на использование ключа и самоподписанного сертификата для обслуживания защищенного контента (HTTPS), выполните следующие действия:

   а) отредактируйте файл /etc/httpd/conf.d/ssl.conf, изменив ключ и расположение сертификата, чтобы использовать только что созданные:

   ```
   SSLCertificateFile /etc/pki/tls/certs/server.crt
   SSLCertificateKeyFile /etc/pki/tls/private/server.key
   ```

   б) перезапустите службу httpd:

   ```
   # systemctl restart httpd.service
   ```

8. Чтобы задействовать браузер для создания HTTPS-соединения с веб-сервером и просмотра содержимого созданного сертификата, в системе, работающей на сервере Apache, введите `https://localhost` в адресную строку браузера. Появится сообщение о ненадежности соединения. Чтобы завершить подключение:
   а) нажмите кнопку I Understand the Risks (Принять риск и продолжить);
   б) нажмите кнопку Add Exception (Добавить исключение);
   в) нажмите кнопку Get Certificate (Получить сертификат);
   г) нажмите кнопку Confirm Security Exception (Добавить исключение безопасности).
9. Создайте файл `/etc/httpd/conf.d/example.org.conf`, который включает виртуальный хостинг на основе имени и создает виртуальный хост, который: 1) прослушивает 80-й порт во всех интерфейсах; 2) имеет администратора `joe@example.org` на сервере; 3) имеет сервер `joe.example.org`; 4) имеет каталог `DocumentRoot` с файлом `/var/www/html/joe.example.org`; 5) имеет директиву `DirectoryIndex`. Она включает в себя по крайней мере файл `index.html` и создает в каталоге `DocumentRoot` файл `index.html` с фразой `Welcome to the House of Joe`.

   Создайте файл `example.org.conf`, который выглядит следующим образом:

```
NameVirtualHost *:80
<VirtualHost *:80>
    ServerAdmin     joe@
example.org
    ServerName      joe.
example.org
    ServerAlias     web.example.org
    DocumentRoot    /var/www/html/joe.example.org/
    DirectoryIndex  index.html
</VirtualHost>
```

   Чтобы создать нужную фразу в файле `index.html`, введите:

```
# echo "Welcome to the House of Joe" > \
     /var/www/html/joe.example.org/index.html
```

10. Добавьте `joe.example.org` в конец записи `localhost` в файле `/etc/hosts` на компьютере, где запущен веб-сервер, и проверьте его, набрав `http://joe.example.org` в адресной строке браузера, чтобы увидеть фразу `Welcome to the House of Joe` при отображении страницы. Для этого:
    а) перезагрузите файл `httpd.conf`, измененный в предыдущем упражнении, одним из двух способов:

```
# apachectl graceful
# systemctl restart httpd
```

    б) отредактируйте файл `/etc/hosts` с помощью любого текстового редактора, чтобы строка локального хоста выглядела следующим образом:

```
127.0.0.1     localhost.localdomain localhost joe.example.org
```

в) из браузера в локальной системе, где работает служба `httpd`, введите `joe.example.org` в адресной строке, чтобы получить доступ к веб-серверу Apache с помощью аутентификации на основе имени.

Глава 18. Настройка FTP-сервера

ВНИМАНИЕ!

Не выполняйте описанные здесь действия на активном общедоступном FTP-сервере, так как они помешают его работе. (Однако можно с их помощью настроить новый FTP-сервер.)

1. Чтобы определить, какой пакет предоставляет службу Very Secure FTP Daemon, от имени суперпользователя введите следующую строку:

```
# yum search "Very Secure FTP"
...
================ N/S Matched: Very Secure FTP ============
vsftpd.i686 : Very Secure Ftp Daemon
```

   Поиск отобразит пакет `vsftpd`.

2. Чтобы установить службу Very Secure FTP Daemon в системе и выполнить поиск файлов конфигурации в пакете `vsftpd`, введите следующее:

```
# yum install vsftpd
# rpm -qc vsftpd | less
```

3. Чтобы включить анонимный FTP и отключить локальный вход пользователя для службы Very Secure FTP Daemon, установите в файле `/etc/vsftpd/vsftpd.conf` следующее:

```
anonymous_enable=YES
write_enable=YES
anon_upload_enable=YES
local_enable=NO
```

4. Чтобы запустить службу Very Secure FTP Daemon и настроить ее запуск при загрузке системы, введите в актуальной версии системы Fedora или Red Hat Enterprise Linux следующее:

```
# systemctl start vsftpd.service
# systemctl enable vsftpd.service
```

   В системе Red Hat Enterprise Linux 6 введите:

```
# service vsftpd start
# chkconfig vsftpd on
```

5. В системе, на которой работает FTP-сервер, введите следующую команду, чтобы создать файл с именем `test` в анонимном каталоге FTP, содержащем слова `Welcome to your vsftpd server`:

```
# echo "Welcome to your vsftpd server" > /var/ftp/test
```

6. Чтобы открыть файл `test` из анонимного домашнего каталога FTP с помощью браузера в системе сервера, откройте браузер, введите в адресную строку следующие данные:

   ```
   ftp://localhost/test
   ```

   и нажмите клавишу Enter. В окне браузера появится фраза `Welcome to your vsftpd server`.
7. Чтобы получить доступ к файлу `test` в домашнем каталоге анонимного сервера FTP, выполните следующие действия (если не можете получить доступ к файлу, убедитесь, что ваш брандмауэр, SELinux и TCP-оболочки разрешают доступ к нему):
   а) в адресную строку браузера в системе, связанной с FTP-сервером (замените `host` полным именем хоста или IP-адресом вашей системы), введите:

      ```
      ftp://host/test
      ```

      Если вы не видите приветственной фразы в окне браузера, выясните, что помешало этому. Чтобы временно отключить брандмауэр (сбросить правила `iptables`), в качестве суперпользователя из оболочки на FTP-сервере введите следующую команду, а затем попытайтесь снова получить доступ к сайту:

      ```
      # iptables -F
      ```

   б) чтобы временно отключить SELinux, введите следующую команду и снова подключитесь к сайту:

      ```
      # setenforce 0
      ```

      Определив, что ошибка заключалась в недоступности файла на FTP-сервере, вернитесь к разделу «Защита FTP-сервера» главы 18 и выполните следующие шаги, чтобы определить, что может блокировать доступ к вашему файлу;
   в) для службы `iptables` убедитесь, что на сервере есть правило, открывающее TCP-порт 21;
   г) для SELinux убедитесь, что контексту файла установлено значение `public_content_t`.
8. Для настройки службы `vsftpd` так, чтобы разрешить анонимным пользователям загрузку файлов в каталог с именем `in`, выполните следующие действия от имени суперпользователя:
   а) создайте каталог `in`:

      ```
      # mkdir /var/ftp/in
      # chown ftp:ftp /var/ftp/in
      # chmod 777 /var/ftp/in
      ```

   б) для последней версии системы Fedora или RHEL откройте окно Firewall (Межсетевой экран) и установите флажок FTP в разделе службы, чтобы открыть доступ к службе FTP. Для более ранних систем RHEL и Fedora настройте брандмауэр `iptables` так, чтобы он разрешал новые запросы на TCP-порт 21,

добавив в файл /etc/sysconfig/iptables перед окончательным правилом DROP или REJECT следующее правило:

```
-A INPUT -m state --state NEW -m tcp -p tcp --dport 21 -j ACCEPT
```

в) настройте брандмауэр iptables для отслеживания подключений, загрузив соответствующий модуль в файл /etc/sysconfig/iptables-config:

```
IPTABLES_MODULES="nf_conntrack_ftp"
```

г) чтобы SELinux разрешил загрузку в каталог, сначала правильно установите контексты файлов:

```
# semanage fcontext -a -t public_content_rw_t "/var/ftp/in(/.*)?"
# restorecon -F -R -v /var/ftp/in
```

д) затем установите логический тип SELinux, чтобы разрешить загрузку:

```
# setsebool -P allow_ftpd_anon_write on
```

е) перезапустите службу vsftpd (service vsftpd restart systemctl restart vsftpd.service).

9. Установите FTP-клиент lftp (если у вас нет второй системы Linux, установите lftp на тот же хост, на котором работает FTP-сервер). При необходимости загрузите файл /etc/hosts в каталог in на сервере, чтобы убедиться, что он доступен. Выполните от имени суперпользователя следующие команды:

```
# yum install lftp
# lftp localhost
lftp localhost:/> cd in
lftp localhost:/in> put /etc/hosts
89 bytes transferred
lftp localhost:/in> quit
```

Вы не сможете увидеть, что скопировали файл hosts во входящий каталог. Однако введите следующую команду из оболочки на хосте, на котором запущен FTP-сервер, чтобы убедиться, что файл hosts находится в этом каталоге:

```
# ls /var/ftp/in/hosts
```

Если не можете загрузить файл, устраните проблему, как описано в упражнении 7, перепроверьте настройки файла vsftpd.conf и проверьте все права в каталоге /var/ftp/in.

10. Используя любой FTP-клиент, перейдите в каталог /pub/debian-meetings на сайте ftp://ftp.gnome.org и перечислите содержимое этого каталога. Вот как это можно сделать с помощью клиента lftp:

```
# lftp ftp://ftp.gnome.org/pub/debian-meetings/
cd ok, cwd=/pub/debian-meetings
lftp ftp.gnome.org:/pub/debian-meetings>> ls
drwxr-xr-x    3          ftp ftp         3 Jan 13 2 014 2004
drwxr-xr-x    6          ftp ftp         6 Jan 13  2014 2005
drwxr-xr-x    8          ftp ftp         8 Dec 20  2006 2006
...
```

Глава 19. Настройка Samba-сервера

1. Чтобы установить пакеты `samba` и `samba-client`, введите из локальной системы от имени суперпользователя следующую команду:

```
# yum install samba samba-client
```

2. Чтобы запустить и включить службы `smb` и `nmb`, введите от имени суперпользователя из локальной системы следующие команды:

```
# systemctl enable smb.service
# systemctl start smb.service
# systemctl enable nmb.service
# systemctl start nmb.service
```

или

```
# chkconfig smb on
# service smb start
# chkconfig nmb on
# service nmb start
```

3. Чтобы установить рабочую группу сервера Samba в `TESTGROUP`, имя NetBIOS — в `MYTEST` и строку сервера — в Samba Test System, от имени суперпользователя откройте файл `/etc/samba/smb.conf` в текстовом редакторе и измените три строки так, чтобы они выглядели следующим образом:

```
workgroup = TESTGROUP
netbios name = MYTEST
server string = Samba Test System
```

4. Чтобы добавить в свою систему пользователя Linux с именем `phil` и присвоить ему пароль Linux и пароль Samba, от имени суперпользователя из оболочки введите следующее:

```
# useradd phil
# passwd phil
New password: *******
Retype new password: *******
# smbpasswd -a phil
New SMB password: *******
Retype new SMB password: *******
Added user phil.
```

Обязательно запомните установленные пароли.

5. Настройте раздел `[homes]` таким образом, чтобы можно было просматривать домашние каталоги (`yes`) и записывать в них (`yes`) и пользователь `phil` являлся единственным допустимым пользователем. Откройте файл `/etc/samba/smb.conf` как суперпользователь и измените раздел `[homes]`:

```
[homes]
        comment = Home Directories
        browseable = Yes
        read only = No
        valid users = phil
```

6. Для установки логических типов SELinux, необходимых для того, чтобы пользователь `phil` мог получить доступ к своему домашнему каталогу через клиент Samba, как суперпользователь из оболочки введите следующее:

```
# setsebool -P samba_enable_home_dirs on
# systemctl restart smb
# systemctl restart nmb
```

   и перезапустите службы `smb` и `nmb`.

7. В локальной системе используйте команду `smbclient`, чтобы указать, что общий ресурс `[homes]` доступен:

```
# smbclient -L localhost
Enter TESTGROUP\root's password: <ENTER>
Anonymous login successful

     Sharename       Type      Comment
     ---------       ----      -------
     homes           Disk      Home Directories
  ...
```

8. Чтобы подключиться к разделу `[homes]` из окна Nautilus (Файлы) в локальной системе сервера Samba для пользователя `phil1` таким образом, чтобы можно было перетаскивать файлы в эту папку, выполните следующие действия:
   а) откройте Nautilus (Файлы), нажав на соответствующий значок;
   б) на панели слева выберите Other Locations (Другие места) и нажмите кнопку Connect to Server (Подключиться к серверу);
   в) введите адрес сервера, например `smb://localhost/phil/`;
   г) при появлении запроса выберите пункт Registered User (Зарегистрированный пользователь), введите имя пользователя `phil`, домен (`TESTGROUP`) и пароль этого пользователя;
   д) откройте еще одно окно Nautilus (Файлы) и перенесите файл в домашнюю папку пользователя `phil`.

9. Чтобы открыть брандмауэр так, чтобы любой, кто имеет доступ к серверу, мог получить доступ к службе Samba (демоны `smbd` и `nmbd`), откройте окно Firewall (Межсетевой экран) и установите флажки Samba и samba-client (как для Runtime, так и для Permanent). Если в вашей системе работает служба `iptables` (не служба `firewalld`), измените файл `/etc/sysconfig/iptables` так, чтобы брандмауэр выглядел следующим образом (правила, которые нужно добавить, выделены жирным шрифтом):

```
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
-A INPUT -p icmp -j ACCEPT
```

```
-A INPUT -i lo -j ACCEPT
-I INPUT -m state --state NEW -m udp -p udp --dport 137 -j ACCEPT
-I INPUT -m state --state NEW -m udp -p udp --dport 138 -j ACCEPT
-I INPUT -m state --state NEW -m tcp -p tcp --dport 139 -j ACCEPT
-I INPUT -m state --state NEW -m tcp -p tcp --dport 445 -j ACCEPT
-A INPUT -j REJECT --reject-with icmp-host-prohibited
-A FORWARD -j REJECT --reject-with icmp-host-prohibited
COMMIT
```

Затем введите следующую команду для правил брандмауэра, чтобы перезагрузить его:

```
# service iptables restart
```

10. Чтобы снова открыть общий ресурс `[homes]` от имени пользователя `phil` из другой системы в вашей сети (Windows или Linux) и убедиться, что можете перетащить в него файлы, выполните следующие действия:
    а) сначала повторите описанный ранее пример окна Nautilus (Файлы) или обратитесь к окну Explorer (Проводник) в Windows и откройте общий ресурс, выбрав сеть, а затем сервер Samba. Сложность заключается в том, что служба должна быть доступна в функциях безопасности сервера Linux;
    б) если не можете получить доступ к общему ресурсу Samba, отключите брандмауэр, а затем отключите SELinux. Если общий ресурс доступен при отключении любой из этих служб, вернитесь назад и ликвидируйте проблему с неработающей службой:

    ```
    # setenforce 0
    # service iptables stop
    ```

    в) когда исправите ее, вновь установите SELinux в принудительный режим и перезапустите службу `iptables`:

    ```
    # setenforce 1
    # service iptables start
    ```

Глава 20. Настройка NFS-сервера

1. Чтобы установить пакеты, необходимые для настройки службы NFS в выбранной вами системе Linux, введите от имени суперпользователя (в системе Fedora или RHEL) следующую команду:

   ```
   # yum install nfs-utils
   ```

2. Чтобы перечислить файлы документации, входящие в пакет программного обеспечения сервера NFS, введите следующее:

   ```
   # rpm -qd nfs-utils
   /usr/share/doc/nfs-utils-1.2.5/ChangeLog
   ...
   ```

```
/usr/share/man/man5/exports.5.gz
/usr/share/man/man5/nfs.5.gz
/usr/share/man/man5/nfsmount.conf.5.gz
/usr/share/man/man7/nfsd.7.gz
/usr/share/man/man8/blkmapd.8.gz
/usr/share/man/man8/exportfs.8.gz
...
```

3. Чтобы запустить и включить службу NFS, введите от имени суперпользователя на сервере NFS следующие данные:

```
# systemctl start nfs-server.service
# systemctl enable nfs-server.service
```

4. Чтобы проверить состояние только что запущенной службы NFS, от имени суперпользователя введите следующее:

```
# systemctl status nfs-server.service
```

5. Чтобы сделать каталог `/var/mystuff` доступным для всех, но только для чтения суперпользователем на клиенте, который имеет основной доступ к общему ресурсу, сначала создайте каталог монтирования следующим образом:

```
# mkdir /var/mystuff
```

Затем создайте запись в файле `/etc/exports`, как в примере далее:

```
/var/mystuff  *(ro,no_root_squash,insecure)
```

Чтобы сделать общий ресурс доступным, введите следующее:

```
# exportfs -v -a
exporting *:/var/mystuff
```

6. Чтобы убедиться, что созданный вами общий ресурс доступен всем хостам, сначала убедитесь, что демон `rpcbind` не заблокирован TCP-оболочками, добавив следующую запись в начало файла `/etc/hosts.allow`:

```
rpcbind: ALL
```

а) чтобы открыть брандмауэр в системах, использующих службу `firewalld` (RHEL 8 и недавние системы Fedora), установите пакет `firewall-config`. Затем запустите `firewall-config` из появившегося окна Firewall (Межсетевой экран), убедитесь, что NFS и `rpc-bind` включены в постоянные настройки брандмауэра;

б) чтобы открыть порты, позволяющие клиентам войти на сервер NFS через брандмауэр `iptables` (на RHEL 6 и более ранних системах Fedora, где нет службы `firewalld`), нужно открыть порты TCP и UDP 111 (демон `rpcbind`), 20048 (`mountd`) и 2049 (NFS), добавив следующее правило в файл `/etc/sysconfig/iptables` и при запуске службы `iptables`:

```
-A INPUT -m state --state NEW -m tcp -p tcp --dport 111 -j ACCEPT
-A INPUT -m state --state NEW -m udp -p udp --dport 111 -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 2049 -j ACCEPT
```

```
-A INPUT -m state --state NEW -m udp -p udp --dport 2049 -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 20048 -j ACCEPT
-A INPUT -m state --state NEW -m udp -p udp --dport 20048 -j ACCEPT
```

Система SELinux должна иметь возможность совместно задействовать файловые системы NFS в принудительном режиме без каких-либо изменений в контекстах файлов или логических типах. Чтобы убедиться в том, что созданный общий ресурс доступен только для чтения, выполните от имени суперпользователя на сервере NFS следующую команду:

```
# setsebool -P nfs_export_all_ro on
```

7. Чтобы просмотреть общие ресурсы, доступные на сервере NFS (имя сервера NFS — `nfsserver`), введите в клиенте NFS следующее:

```
# showmount -e nfsserver
Export list for nfsserver:
/var/mystuff  *
```

8. Чтобы создать каталог `/var/remote` и временно смонтировать каталог `/var/mystuff` с сервера NFS (в данном примере `nfsserver`) в этой точке монтирования, введите в качестве суперпользователя следующее:

```
# mkdir /var/remote
# mount -t nfs nfsserver:/var/mystuff /var/remote
```

9. Чтобы добавить запись так, чтобы то же самое монтирование выполнялось автоматически при перезагрузке, сначала размонтируйте каталог `/var/remote` следующим образом:

```
# umount /var/remote
```

Затем добавьте в файл `/etc/fstab` следующую запись:

```
/var/remote   nfsserver:/var/mystuff   nfs bg,ro 0 0
```

Чтобы проверить правильность настроек общего ресурса, введите в клиенте NFS в качестве суперпользователя следующие данные:

```
# mount -a
# mount -t nfs4
nfsserver:/var/mystuff on /var/remote type nfs4
 (ro,vers=4,rsize=524288...
```

10. Чтобы скопировать некоторые файлы в каталог `/var/mystuff`, введите на сервере NFS следующую команду:

```
# cp /etc/hosts /etc/services /var/mystuff
```

Чтобы убедиться, что вы можете видеть только что добавленные в этот каталог файлы и не можете записывать файлы в него из клиента, введите:

```
# ls /var/remote
hosts    services
# touch /var/remote/file1
touch: cannot touch '/var/remote/file1': Read-only file system
```

Глава 21. Диагностика Linux

1. Чтобы перейти в режим настройки BIOS вашего компьютера:
   а) перезагрузите компьютер;
   б) через несколько секунд появится экран BIOS с указанием того, какую функциональную клавишу нажать, чтобы перейти в режим Setup (на моем рабочем столе Dell это клавиша F2);
   в) должен появиться экран BIOS. (Если система начала загрузку Linux, вы недостаточно быстро нажимали функциональную клавишу.)
2. На экране настроек BIOS определите, является ли ваш компьютер 32- или 64-разрядным, включает ли он поддержку виртуализации и способна ли ваша сетевая интерфейсная карта загружать PXE.

   Эти данные могут отличаться от моих, они зависят от компьютера и системы Linux. Экран настройки BIOS также может быть иным. Однако в целом вы можете использовать клавиши ←, ↑, ↓ и → и клавишу Tab для перемещения между различными столбцами и нажимать клавишу Enter для выбора.

   На моем рабочем столе Dell под заголовком System (Система) находится информация о процессоре. Мой компьютер 64-разрядный. Загляните в раздел Processor Info (Информация о процессоре) или аналогичный на своем компьютере, чтобы узнать тип процессора.

   На рабочем столе Dell под заголовком Onboard Devices (Встроенные устройства) я выделил пункт Integrated NIC (Встроенная карта NIC) и нажал клавишу Enter. Встроенная карта NIC появится справа. Она позволяет включать или отключать саму карту сетевого интерфейса (On или Off) или включить ее с помощью PXE или RPL (если нужно загрузить компьютер по сети).
3. Чтобы прервать процесс загрузки и добраться до загрузчика GRUB:
   а) перезагрузите компьютер;
   б) сразу после того как экран BIOS исчезнет, вы увидите обратный отсчет до загрузки системы Linux. Нажмите любую клавишу, допустим Пробел;
   в) должно появиться меню загрузчика GRUB, оно позволит выбрать ядро операционной системы, которое следует загрузить.
4. Чтобы загрузить компьютер до уровня выполнения 1 и произвести обслуживание системы, перейдите на экран загрузки GRUB (как описано в предыдущем упражнении), а затем выполните следующие действия:
   а) используйте клавиши ←, ↑, ↓ и →, чтобы выделить операционную систему и ядро, которые вы хотите загрузить;
   б) введите e, чтобы увидеть строки, необходимые для загрузки операционной системы;
   в) переместите курсор на строку с ядром (включает в себя слово vmlinuz);
   г) переместите курсор в конец этой строки, добавьте пробел и введите init=bash;

д) следуйте инструкциям, чтобы загрузить новую строку. В данном случае придется нажать либо сочетание клавиш Ctrl+X, либо клавишу Enter. В случае другого экрана введите b.

Если это сработало, ваша система обойдет приглашение входа в систему и загрузится непосредственно в оболочку суперпользователя, где можно выполнять административные задачи, не вводя каждый раз пароль.

5. Чтобы просмотреть сообщения, созданные в кольцевом буфере ядра (который показывает активность ядра при загрузке), из оболочки после завершения загрузки системы введите следующее:

```
# dmesg | less
```

Или в системе, использующей службу systemd, введите:

```
# journalctl -k
```

6. Чтобы запустить пробную версию yum update из систем Fedora или RHEL и исключить любой доступный пакет ядра, введите следующее (при появлении запроса введите N, чтобы не выполнять обновление, если они доступны):

```
# yum update --exclude='kernel*'
```

7. Чтобы проверить, какие процессы прослушивают входящие соединения в вашей системе, введите следующую команду:

```
# netstat -tupln | less
```

8. Чтобы проверить, какие порты открыты в вашем внешнем сетевом интерфейсе, если это возможно, запустите команду nmap из другой системы Linux в своей сети, заменив *yourhost* именем хоста или IP-адресом вашей системы:

```
# nmap yourhost
```

9. Чтобы очистить системный кэш страниц и посмотреть, как он влияет на использование памяти, выполните следующие действия:

а) откройте окно Terminal (Терминал) из приложений на рабочем столе (в разных системах оно находится в разных меню);

б) выполните команду top, чтобы просмотреть процессы, запущенные в данный момент в вашей системе, а затем введите прописную букву M, чтобы отсортировать процессы по потреблению памяти;

в) в окне Terminal (Терминал) откройте меню и выберите создание второго окна;

г) во втором окне Terminal (Терминал) войдите в оболочку от имени суперпользователя (su -);

д) наблюдая за строкой Mem (используемый ранее столбец) в первом окне Terminal (Терминал), введите из второго окна следующее:

```
# echo 3 > /proc/sys/vm/drop_caches
```

е) в строке Mem используемая память RES должна значительно уменьшиться. Цифры в столбце RES для каждого процесса также должны уменьшаться.

10. Чтобы просмотреть задействование памяти и подкачки из веб-интерфейса Cockpit, откройте в браузере Cockpit (https://hostname:9090). Выберите System ▸ Memory & Swap (Система ▸ Память и обмен).

Глава 22. Базовые методы обеспечения безопасности

1. Чтобы проверить сообщения из журнала systemd для служб NetworkManager.service, sshd.service и auditd.service, введите следующее:

```
# journalctl -u NetworkManager.service
...
# journalctl -u sshd.service
...
# journalctl -u auditd.service
...
```

2. Пароли пользователей хранятся в файле /etc/shadow. Чтобы увидеть его права, введите команду ls-l /etc/shadow в командной строке. (Если файла не существует, необходимо запустить команду pwconv.)

   Далее приведены соответствующие настройки:

```
# ls -l /etc/shadow
----------. 1 root root 1049 Feb  10 09:45 /etc/shadow
```

3. Чтобы определить срок действия пароля вашей учетной записи с помощью одной команды, введите chage -l user_name, например:

```
# chage -l chris
```

4. Чтобы начать аудит событий в файле /etc/shadow с помощью демона auditd, введите в командной строке следующее:

```
# auditctl -w /etc/shadow -p w
```

5. Чтобы проверить настройки аудита, введите auditctl -l в командной строке. Чтобы создать отчет из демона auditd в файле /etc/shadow, введите ausearch -f /etc/shadow в командной строке. Чтобы отключить аудит этого файла, введите auditctl -W /etc/shadow -p w.

6. Чтобы установить пакет lemon, повредите файл /usr/bin/lemon и удалите пакет lemon, введя следующую команду:

```
# yum install -y lemon
# cp /etc/services /usr/bin/lemon
# rpm -V lemon
S.5....T.    /usr/bin/lemon
# yum erase lemon
```

Файл `lemon` отличается от оригинала по размеру файла (`S`), по `md4sum` (`5`) и времени его модификации (`T`). В системе Ubuntu установите пакет с `apt-get install lemon` и введите команду `debsums lemon`, чтобы проверить его.

7. Если вы подозреваете, что ваша система атакована и были изменены важные двоичные файлы, их можно найти, введя в командной строке `find directory-mtime -1` для каталогов `/bin`, `/sbin`, `/usr/bin` и `/usr/sbin`.
8. Чтобы установить и запустить `chkrootkit` и увидеть, был ли установлен руткит после вредоносной атаки, выберите свой дистрибутив и выполните следующие действия:
   а) на Fedora или RHEL введите команду `yum install chkrootkit`;
   б) на Ubuntu и в дистрибутивах на базе Debian введите команду `sudo apt-get nstall chkrootkit`;
   в) чтобы запустить проверку, введите `chkrootkit` в командной строке и просмотрите результаты.
9. Чтобы найти файлы в любой точке системы с установленным правом UID или SGID, введите команду `find / -perm /6000 -ls`.
10. Чтобы установить пакет `aide`, выполните команду `aide` для инициализации базы данных `aide`, скопируйте базу данных в нужное место и выполните команду `aide` для проверки того, изменены ли какие-либо важные файлы в вашей системе:

```
# yum install aide
# aide -i
# cp /var/lib/aide/aide.db.new.gz /var/lib/aide/aide.db.gz
# aide -C
```

Чтобы сделать вывод более интересным, можете установить пакет `lemon` (описан в упражнении 6) и изменить его перед запуском команды `aide -C`, чтобы увидеть, как выглядит модифицированный двоичный файл.

Глава 23. Продвинутые методы обеспечения безопасности

Для выполнения первых нескольких упражнений у вас должен быть установлен пакет `gnupg2`. Он не установлен по умолчанию в дистрибутиве Ubuntu, но имеется в последних версиях дистрибутивов Fedora и RHEL.

1. Чтобы зашифровать файл с помощью утилиты `gpg2` и симметричного шифра, введите следующую команду (утилита `gpg2` запрашивает ключевую фразу для защиты симметричного шифра):

```
$ gpg2 -c filename
```

2. Чтобы сгенерировать пару ключей с помощью утилиты gpg2, введите следующее:

```
$ gpg2 --gen-key
```

   Необходимо добавить:

   а) настоящее имя и адрес электронной почты;

   б) ключевую фразу для личного пароля.

3. Чтобы перечислить сгенерированные ключи, введите:

```
$ gpg2 --list-keys
```

4. Чтобы зашифровать файл и добавить свою цифровую подпись с помощью утилиты gpg2, выполните следующие действия:

   а) сгенерируйте связку ключей (см. упражнение 2);

   б) после этого введите:

```
$ gpg2 --output EncryptedSignedFile --sign FiletoEncryptSign
```

5. На странице getfedora.org выберите один из дистрибутивов Fedora для загрузки. Когда она будет завершена, выберите Verify your Download (Проверить загруженный образ), чтобы просмотреть инструкции по проверке образа. Например, загрузите файл CHECKSUM для загруженного образа, а затем введите следующее:

```
$ curl https://getfedora.org/static/fedora.gpg | gpg --import
$ gpg --verify-files *-CHECKSUM
$ sha256sum -c *-CHECKSUM
```

6. Чтобы определить, подключена ли команда su в вашей системе Linux к PAM, введите следующее:

```
$ ldd $(which su) | grep pam
libpam.so.0 => /lib64/libpam.so.0 (0x00007fca14370000)
ibpam_misc.so.0 => /lib64/libpam_misc.so.0 (0x00007fca1416c000
```

   Если команда su в системе Linux поддерживает PAM, то после выполнения команды ldd вы увидите в списке имя библиотеки PAM.

7. Чтобы определить, есть ли у команды файл конфигурации PAM, введите следующее:

```
$ ls /etc/pam.d/su
/etc/pam.d/su
```

   Если файл существует, то, чтобы отобразить его содержимое, введите в командной строке следующее (контексты модулей PAM, которые модули используют, включают любое из следующих значений — auth, account, password или session):

```
$ cat /etc/pam.d/su
```

8. Чтобы перечислить различные модули PAM в системе Fedora или RHEL, введите следующее:

```
$ ls /usr/lib64/security/pam*.so
```

Чтобы перечислить различные модули PAM в системе Ubuntu, введите:

```
# find / -name pam*.so
```

9. Чтобы найти в своей системе файл конфигурации PAM `other`, введите `ls /etc/pam.d/other` в командной строке. Файл конфигурации `other` применяет запрет `Implicit Deny` и должен выглядеть примерно так:

```
$ cat /etc/pam.d/other
#%PAM-1.0
auth     required       pam_deny.so
account  required       pam_deny.so
password required       pam_deny.so
session  required       pam_deny.so
```

10. Чтобы найти файл конфигурации ограничений PAM, введите следующее:

```
$ ls /etc/security/limits.conf
```

Чтобы вывести содержимое файла на экран, введите:

```
$ cat /etc/security/limits.conf
```

В этом файле настройки для предотвращения работы вредоносной программы (fork-бомбы) выглядят следующим образом:

```
@student     hard    nproc           50
@student     -       maxlogins        4
```


Глава 24. Повышенная безопасность с технологией SELinux

1. Чтобы перевести систему в разрешительный режим для SELinux, введите `setenforce permissive` в командной строке. Также можно ввести команду `setenforce 0`.
2. Соблюдайте осторожность при переводе системы SELinux в принудительный режим без изменения основного файла конфигурации SELinux. Лучше всего не запускать эту команду в системе для выполнения упражнения, пока не начнете использовать SELinux. Введите команду `setenforce enforcing` в командной строке. Можно также ввести команду `setenforce 1`.
3. Чтобы найти и просмотреть постоянную политику SELinux (задается во время загрузки), перейдите в основной файл конфигурации SELinux — `/etc/selinux/config`. Чтобы просмотреть его, введите `cat/etc/selinux | config / grep SELINUX=` в командной строке. Чтобы узнать, как он установлен в данный момент, введите команду `getenforce`.
4. Чтобы перечислить контекст безопасности файла `/etc/hosts` и определить различные атрибуты контекста безопасности, введите `ls -Z /etc/hosts` в командной строке:

```
$ ls -Z /etc/hosts
-rw-r--r--. root root system_u:object_r:net_conf_t:s0 /etc/hosts
```

а) пользовательский контекст файла — это `system_u`, он указывает на системный файл;

б) роль файла — это `object_r`, указывающая на объект в файловой системе (в данном случае текстовый файл);

в) тип файла — это `net_conf_t`, поскольку он является файлом конфигурации сети;

г) уровень чувствительности файла равен `s0`, что указывает на самый низкий уровень безопасности (это может быть номер от `s0` до `s3`);

д) уровень категории файла начинается с `c` и заканчивается числом. Он может относиться к диапазону чисел `c0-c102`. Это действие требуется только в высокозащищенных средах и в данном случае не используется.

5. Чтобы создать файл `test.html` и назначить ему тип `httpd_sys_content_t`, введите следующее:

```
$ touch test.html
$ chcon -t httpd_sys_content_t test.html
$ ls -Z test.html
-rw-rw-r--. chris chris unconfined_u:object_r:httpd_sys_content_t:s0 test.html
```

6. Чтобы перечислить контекст безопасности процесса `crond` и определить его атрибуты, введите в командной строке:

```
$ ps -efZ | grep crond
system_u:system_r:crond_t:s0-s0:c0.c1023 root 665  1  0
    Sep18 ?   00:00:00 /usr/sbin/crond -n
```

а) пользовательский контекст процесса — это `system_u`, указывающий на системный процесс;

б) роль процесса — `system_r`, что указывает на системную роль;

в) тип или домен процесса — это `crond_t`;

г) уровень чувствительности процесса начинается с `s0-s0`, это указывает на то, что процесс нечувствителен (однако он безопасен по обычным стандартам Linux, поскольку запускается от имени суперпользователя);

д) уровень категории процесса — это `c0.c1023`, где `c0` указывает на то, что категория также не очень безопасна с точки зрения системы SELinux.

7. Чтобы создать файл `/etc/test.txt`, измените его контекст на `user_tmp_t`, восстановите правильное содержимое (контекст по умолчанию для каталога `/etc`) и удалите файл:

```
# touch /etc/test.txt
# ls -Z /etc/test.txt
-rw-r--r--. root root unconfined_u:object_r:etc_t:s0   /etc/test.txt
# chcon -t user_tmp_t /etc/test.txt
# ls -Z /etc/test.txt
-rw-r--r--. root root unconfined_u:object_r:user_tmp_t:s0 /etc/test.txt
```

```
# restorecon /etc/test.txt
# ls -Z /etc/test.txt
-rw-r--r--. root root unconfined_u:object_r:etc_t:s0   /etc/test.txt
# rm /etc/test.txt
rm: remove regular empty file `/etc/test.txt'? y
```

8. Чтобы определить, какие логические типы разрешают анонимную запись и доступ к домашнему каталогу службы `tftp`, а затем включить эти логические типы навсегда, введите следующие команды:

```
# getsebool -a | grep tftp
tftp_home_dir --> off
tftpd_anon_write --> off
...
# setsebool -P tftp_home_dir=on
# setsebool -P tftp_anon_write=on
# getsebool tftp_home_dir tftp_anon_write
tftp_home_dir --> on
tftp_anon_write --> on
```

9. Чтобы перечислить все модули политики SELinux в своей системе вместе с их номерами версий, введите команду `semodule –l`.

ПРИМЕЧАНИЕ

Команда ls /etc/selinux/targeted/modules/active/modules/*.pp подходит в качестве ответа, однако она не дает вам номера версий модулей политики. Только команда semodule -l выводит номера версий.

10. Чтобы разрешить доступ к службе `sshd` через TCP-порт 54 903 в SELinux, введите следующее:

```
# semanage port -a -t ssh_port_t -p tcp 54903
# semanage port -l | grep ssh
ssh_port_t                tcp             54903, 22
```

Глава 25. Защита Linux в сети

1. Чтобы установить утилиту Network Mapper (она же `nmap`) в локальную систему Linux, выполните следующие действия:
   а) в системе Fedora или RHEL введите команду `yum install nmap` в командной строке;
   б) в системе Ubuntu пакет `nmap` может быть предустановлен заранее. Если его нет, введите команду `sudo apt-get install nmap` в командной строке.
2. Чтобы запустить сканирование TCP Connect на локальном адресе интернет-протокола loopback, введите в командной строке `nmap-sT 27.0.0`. Порты, которые

вы используете на своем сервере Linux, будут иными. Они могут выглядеть примерно так:

```
# nmap -sT 127.0.0.1
       ...
       PORT    STATE SERVICE
       25/tcp  open  smtp
       631/tcp open  ipp
```

3. Чтобы запустить сканирование UDP Connect в своей системе Linux из удаленной системы, выполните следующие действия:
   а) определите IP-адрес Linux-сервера, введя команду `ifconfigat`. Выходные данные будут выглядеть как в примере далее, и IP-адрес системы следует за `inet addr:` в выходных данных команды `ifconfig`:

   ```
   # ifconfig
   ...
   p2p1  Link encap:Ethernet   HWaddr 08:00:27:E5:89:5A
         inet addr: 10.140.67.23
   ```

   б) из удаленной системы Linux введите команду `nmap-sU IP address`, используя полученный ранее IP-адрес, например:

   ```
   # nmap -sU 10.140.67.23
   ```

4. Чтобы проверить, работает ли в вашей системе служба `firewalld`, а затем установить и запустить ее, выполните следующие действия:
   а) введите команду `systemctl status firewalld.service`;
   б) если служба `firewalld` не запущена в системе Fedora или RHEL, введите следующее:

   ```
   # yum install firewalld firewall-config -y
   # systemctl start firewalld
   # systemctl enable firewalld
   ```

5. Чтобы открыть порты в брандмауэре и разрешить удаленный доступ к локальной веб-службе, выполните следующие действия:
   а) откройте окно Firewall (Межсетевой экран) с помощью команды `firewalld-config`;
   б) выберите настройку Runtime (Рабочая среда);
   в) выберите текущую зону (например, FedoraWorkstation);
   г) в разделе Services (Службы) выберите варианты `http` и `https`;
   д) выберите настройку Permanent (Постоянная);
   е) в разделе Services (Службы) выберите варианты `http` и `https`.
6. Чтобы определить текущие политики и правила брандмауэра `netfilter/iptables` в своей системе Linux, введите `iptables-vnL` в командной строке.

7. Чтобы сохранить, очистить и восстановить текущие правила брандмауэра системы Linux, выполните следующее:
   а) чтобы сохранить текущие правила, введите:
   ```
   # iptables-save >/tmp/myiptables
   ```
   б) чтобы очистить текущие правила, введите:
   ```
   # iptables -F
   ```
   в) чтобы восстановить текущие правила, введите:
   ```
   # iptables-restore < /tmp/myiptables
   ```
8. Чтобы настроить таблицу фильтров брандмауэра системы Linux для цепочки ввода на политику `DROP`, введите `iptables -P INPUT DROP` в командной строке.
9. Чтобы изменить политику таблицы фильтров брандмауэра системы Linux обратно на `accept` для цепочки ввода, введите следующее:
   ```
   # iptables -P INPUT ACCEPT
   ```
   Чтобы добавить правило `DROP` для всех сетевых пакетов с IP-адреса 10.140.67.23, введите следующее:
   ```
   # iptables -A INPUT -s 10.140.67.23 -j DROP
   ```
10. Чтобы удалить только что добавленное правило, не сбрасывая и не восстанавливая правила брандмауэра системы Linux, введите `iptables -D INPUT 1` в командной строке. Предполагается, что правило, которое вы добавили ранее, является правилом 1. Если это не так, измените значение на соответствующий номер правила в команде `iptables`.

Глава 26. Работа с облаками и контейнерами

1. Чтобы установить и запустить контейнер, используйте либо команду `podman` (для любой системы RHEL или Fedora), либо команду `docker` (RHEL 7):
   ```
   # yum install podman -y
   ```
   или
   ```
   # yum install docker -y
   # systemctl start docker
   # systemctl enable docker
   ```
2. Используйте команду либо `podman`, либо `docker`, чтобы вытащить образ вашего хоста `registry.access.redhat.com/ubi7/ubi`:
   ```
   # podman pull registry.access.redhat.com/ubi7/ubi
   ```
   или
   ```
   # docker pull registry.access.redhat.com/ubi7/ubi
   ```

3. Чтобы запустить образ `ubi7/ubi`, откройте оболочку `bash`:

```
# podman run -it ubi7/ubi bash
```

или

```
# docker run -it ubi7/ubi bash
```

4. Чтобы выполнить команды для просмотра операционной системы, на которой основан контейнер, установите пакет `proc-ps` и выполните команду для просмотра процессов, запущенных внутри контейнера:

```
bash-4.4# cat /etc/os-release | grep ^NAME
NAME="Red Hat Enterprise Linux"
bash-4.4# yum install procps -y
bash-4.4# ps -ef
UID        PID  PPID  C STIME TTY          TIME CMD
root         1     0  0 03:37 pts/0    00:00:00 bash
root        20     1  0 03:43 pts/0    00:00:00 ps -ef
bash-4.4# exit
```

5. Чтобы перезапустить контейнер, который вы только что закрыли с помощью интерактивной оболочки, и подключиться к нему, введите следующее:

```
# podman ps -a
CONTAINER ID  IMAGE               COMMAND  CREATED
STATUS                     PORTS  NAMES
eabf1fb57a3a  ...ubi8/ubi:latest bash     7 minutes ago  Exited (0) 4
seconds ago
       compassionate_hawking
# podman start -a eabf1fb57a3a
bash-4.4# exit
```

6. Чтобы создать простой файл `Dockerfile` из базового образа `ubi7/ubi`, включите скрипт `cworks.sh`, который повторяет фразу `The Container Works!`, и добавьте его к образу:

   а) создайте и измените новый каталог:

```
# mkdir project
# cd project
```

   б) создайте файл `Dockerfile` со следующим содержимым:

```
FROM registry.access.redhat.com/ubi7/ubi-minimal
COPY ./cworks.sh /usr/local/bin/
CMD ["/usr/local/bin/cworks.sh"]
```

   в) создайте файл `cworks.sh` со следующим содержимым:

```
#!/bin/bash
set -o errexit
set -o nounset
set -o pipefail
echo "The Container Works!"
```

7. Используйте команду `docker` или `podman`, чтобы построить образ `containerworks` из только что созданного файла `Dockerfile`:

```
# podman build -t myproject .
```

или

```
# docker build -t myproject .
```

8. Получить доступ к реестру контейнеров можно, либо установив пакет `docker` для дистрибутива:

```
# yum install docker-distribution -y
# systemctl start docker-distribution
# systemctl enable docker-distribution
```

либо получив учетную запись в реестре Quay.IO (quay.io/plans/) или хабе Docker (Docker Hub):

```
# podman login quay.io
Username: <username>
Password: *********
```

9. Чтобы пометить новое изображение и поместить его в выбранный реестр контейнеров, выполните следующие команды:

```
# podman tag aa0274872f23 \
quay.io/<user>/<imagename>:v1.0
# podman push \
quay.io/<user>/<imagename>:v1.0
```

Глава 27. Облачные вычисления в системе Linux

1. Чтобы проверить, поддерживает ли ваш компьютер виртуализацию KVM, введите следующую команду:

```
# cat /proc/cpuinfo | grep --color -E "vmx|svm|lm"
flags  : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat
pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb
rdtscp lm constant_tsc arch_perfmon pebs bts rep_good xtopology nonstop_tsc
aperfmperf pni pclmulqdq dtes64 monitor ds_cpl vmx smx es...
```

Процессор должен поддерживать VMX или SVM. `Lm` указывает на то, что это 64-разрядный компьютер.

2. Чтобы установить систему Linux вместе с пакетами, необходимыми для использования ее в качестве хоста KVM, и запустить приложение Virtual Machine Manager, выполните следующие действия:
   а) загрузите «живой» или установочный образ с сайта Linux (например, getfedora.org) и запишите его на DVD (или иным образом установите его);

б) загрузите установочный образ и установите его на жесткий диск;

в) для рабочего стола Fedora Workstation после завершения установки и перезагрузки установите следующий пакет (для других дистрибутивов Linux может потребоваться установить пакет, который предоставляет службу `ibvirtd`):

```
# yum install virt-manager libvirt-daemon-config-network
```

3. Чтобы убедиться, что службы `sshd` и `libvirtd` запущены в системе, введите следующее:

```
# systemctl start sshd.service
# systemctl enable sshd.service
# systemctl start libvirtd.service
# systemctl enable libvirtd.service
```

4. Загрузите установочный ISO-образ Linux, совместимый с вашим гипервизором, и скопируйте его в каталог по умолчанию, используемый программой Virtual Machine Manager. Например, если DVD с Fedora Workstation находится в текущем каталоге, можете ввести следующее:

```
# cp Fedora-Workstation-Live-x86_64-30-1.2.iso /var/lib/libvirt/images/
```

5. Чтобы проверить настройки сетевого моста по умолчанию (`virbr0`), введите следующее:

```
# ip addr show virbr0
4: virbr0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc
     noqueue state UP group default
    link/ether de:21:23:0e:2b:c1 brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.1/24 brd 192.168.122.255 scope global virbr0
       valid_lft forever preferred_lft forever6.
```

6. Чтобы установить виртуальную машину с помощью ISO-образа, скопированного ранее, выполните следующие действия:

а) введите команду:

```
# virt-manager &
```

б) перейдите в меню File (Файл) и выберите New Virtual Machine (Новая виртуальная машина);

в) выберите Local Install Media (Жесткий диск) и нажмите Forward (Далее);

г) выберите Browse (Загрузить), выберите «живой» или установочный образ, нажмите Choose Volume (Выбрать том) и нажмите кнопку Forward (Далее);

д) выберите подходящие память и процессоры и нажмите кнопку Forward (Далее);

е) выберите нужный размер диска и нажмите кнопку Forward (Далее);

ж) выберите Virtual network default: NAT (Виртуальная сеть по умолчанию: NAT), возможно, она уже выбрана;

з) по окончании нажмите кнопку Finish (Завершить);

и) далее следуйте процессу установки, указанному в инсталляторе ISO-образа.

7. Чтобы убедиться, что вы можете войти в виртуальную машину и использовать ее, выполните следующие действия:
   а) дважды щелкните на строке новой виртуальной машины;
   б) когда появится соответствующее окно, войдите в систему как обычно.
8. Чтобы убедиться, что виртуальная машина может подключаться к Интернету или другой сети за пределами гипервизора, выполните одно из следующих действий:
   а) откройте браузер и подключитесь к сайту в Интернете;
   б) откройте окно Terminal (Терминал), введите команду `ping redhat.com`, а затем нажмите сочетание клавиш Ctrl+C, чтобы выйти.
9. Чтобы остановить виртуальную машину, выполните следующее:
   а) щелкните правой кнопкой мыши на строке виртуальной машины в окне virt-manager;
   б) выберите вариант Shut Down (Отключить) и выберите этот вариант снова;
   в) если виртуальная машина не отключилась сразу же, вы можете выбрать вариант Force Off (Принудительно отключить), но результат будет таким же, как если выдернуть вилку из розетки, — потеря всех данных.
10. Запустите виртуальную машину снова:
    а) щелкните правой кнопкой мыши на строке виртуальной машины в окне virt-manager;
    б) нажмите кнопку Run (Запустить).

Глава 28. Развертывание приложений Linux в облаке

1. Чтобы установить пакеты `genisoimage`, `cloud-init`, `qemu-img`, и `virt-viewer`, введите:

```
# dnf install genisoimage cloud-init qemu-img virt-viewer
```

2. Чтобы загрузить облачный образ Fedora, перейдите по ссылке getfedora.org/en/cloud/download/ и загрузите образ `qcow2`. В списке OpenStack есть подходящий с именем `Fedora-Cloud-Base-31-1.9.x86_64.qcow2`.
3. Чтобы создать снимок этого образа в формате `qcow2` с именем `myvm.qcow2`, введите следующее:

```
 # qemu-img create -f qcow2 \
-o backing_file=Fedora-Cloud-Base-31-1.9.x86_64.qcow2 \
myvm.qcow2
```

4. Создайте файл метаданных `cloud-init` под названием `meta-data`, который включает в себя следующее:

```
instance-id: myvm
local-hostname: myvm.example.com
```

5. Создайте файл данных пользователя `cloud-init` под названием `user-data`, который включает в себя следующее:

```
#cloud-config
password: test
chpasswd: {expire: False}
```

6. Выполните команду `genisoimage`, чтобы объединить файлы `meta-data` и `user-data` и создать файл `mydata.iso`:

```
# genisoimage -output mydata.iso -volid cidata \
    -joliet-long -rock user-data meta-data
```

7. Используйте команду `virt-install`, чтобы объединить образ виртуальной машины `myvm.qcow2` с файлом `mydata.iso` для создания нового образа виртуальной машины с именем `newvm`.

```
# virt-install --import --name newvm \
   --ram 4096 --vcpus 2 \
   --disk path=myvm.qcow2,format=qcow2,bus=virtio \
   --disk path=mydata.iso,device=cdrom \
   --network network=default &
```

8. Чтобы открыть виртуальную машину `newvm` с помощью команды `virt-viewer`, введите следующее:

```
# virt-viewer newvm
```

9. Войдите в виртуальную машину `newvm` с помощью пользователя `fedora` и пароля `test`:

```
Login: fedora
Password: test
```

Глава 29. Автоматизация приложений и инфраструктуры с помощью системы Ansible

1. Чтобы установить пакет `ansible`, выполните следующие действия:
   - **RHEL 8:**

```
# subscription-manager repos \
    --enable ansible-2.9-for-rhel-8-x86_64-rpms
# dnf install ansible -y
```

- **Fedora:**

```
# dnf install ansible -y
```

- **Ubuntu:**

```
$ sudo apt update
$ sudo apt install software-properties-common
$ sudo apt-add-repository --yes --update ppa:ansible/ansible
$ sudo apt install ansible
```

2. Чтобы добавить привилегии `sudo` для пользователя, выполняющего команды Ansible, запустите команду `visudo` и создайте запись, аналогичную следующей (измените `joe` на нужное имя пользователя):

```
joe   ALL=(ALL)          NOPASSWD: ALL
```

3. Откройте файл `my_playbook.yaml` и добавьте к нему следующее:

```
---
- name: Create web server
  hosts: localhost
  tasks:
  - name: Install httpd
    yum:
      name: httpd
      state: present
```

4. Чтобы запустить файл `my_playbook.yaml` в режиме проверки, выполните следующую команду (она не будет выполнена, потому что у пользователя нет прав на установку пакета):

```
$ ansible-playbook -C my_playbook.yaml
...

        TASK [Install httpd]
************************************************************
        fatal: [localhost]: FAILED! => {"changed": false, "msg": "This
             command has to be run under the root user.", "results": []}
        ...
```

5. Внесите в файл `my_playbook.yaml` следующие изменения:

```
---
- name: Create web server
  hosts: localhost
  become: yes
  become_method: sudo
  become_user: root
  tasks:
  - name: Install httpd
    yum:
      name: httpd
      state: present
```

Б

6. Чтобы запустить файл `my_playbook` снова и установить пакет `httpd`, введите следующее:

```
$ ansible-playbook my_playbook.yaml
...
TASK [Install httpd] ************************************
changed: [localhost]
PLAY RECAP **********************************************
localhost: ok=2 changed=1 unreachable=0 failed=0 skipped=0 rescued=0
ignored=0
```

7. Чтобы запустить службу `httpd` и установить ее так, чтобы она запускалась каждый раз при загрузке системы, измените файл `my_playbook.yaml` следующим образом:

```
---
- name: Create web server
  hosts: localhost
  become: yes
  become_method: sudo
  become_user: root
  tasks:
  - name: Install httpd
    yum:
      name: httpd
      state: present
- name: start httpd
  service:
    name: httpd
    state: started
```

8. Чтобы выполнить команду `ansible` так, чтобы она проверяла, находится ли служба `httpd` на `localhost`, введите:

```
$ ansible localhost -m service \
      -a "name=httpd state=started" --check
localhost | SUCCESS => {
    "changed": false,
    "name": "httpd",
    "state": "started",
    "status": { ...
```

9. Чтобы создать в текущем каталоге файл `index.html` с текстом `Web server is up`, который запускает команду `ansible` для копирования этого файла в каталог `/var/www/html` на `localhost`, выполните следующие действия (замените `joe` нужным именем пользователя):

```
$ echo "Web server is up" > index.html
$ ansible localhost
 -m copy -a \
    "src=./index.html dest=/var/www/html/ \
    owner=apache group=apache mode=0644" \
    -b --user joe --become-user root --become-method sudo
host01 | CHANGED => { ...
```

10. Чтобы использовать команду `curl` для просмотра содержимого файла, который вы только что скопировали на веб-сервер, выполните следующую команду:

```
$ curl localhost
Web server is up
```

Глава 30. Развертывание приложений в контейнеры с помощью кластера Kubernetes

1. Чтобы получить доступ к инструменту Minikube, выполните один из вариантов:
   а) установите Minikube, как описано на странице kubernetes.io/docs/tasks/ tools/ install-minikube;
   б) получите доступ к удаленному инструменту Minikube, например, через руководство Kubernetes.io на странице kubernetes.io/docs/tutorials/.
2. Чтобы просмотреть версии инструмента Minikube, клиента `kubect` и службы `Kubernetes`, введите следующее:

```
$ minikube version
$ kubectl version
```

3. Чтобы развернуть модуль под управлением образа контейнера `hello-node`, введите следующее:

```
$ kubectl create deployment hello-node \
    --image=gcr.io/hello-minikube-zero-install/hello-node
```

4. Чтобы просмотреть развертывание `hello-node` и подробно описать его, введите следующее:

```
$ kubectl get deployment
$ kubectl describe deployment hello-node
```

5. Чтобы просмотреть текущий набор реплик, связанный с развертыванием `hello-node`, введите следующее:

```
$ kubectl get rs
```

6. Чтобы увеличить развертывание `hello-node` до трех копий, введите следующую команду:

```
$ kubectl scale deployments/hello-node --replicas=3
```

7. Чтобы открыть развертывание `hello-node` за пределами кластера `Kubernetes` с помощью `LoadBalancer`, введите:

```
$ kubectl expose deployment hello-node \
    --type=LoadBalancer --port=8080
```

8. Чтобы получить IP-адрес вашего инструмента Minikube и номер порта открытой службы `hello-node`, введите следующее:

```
$ minikube ip
192.168.39.150
$ kubectl describe service hello-node | grep NodePort
NodePort:                 <unset>  31302/TCP
```

9. Примените команду `curl` для запроса службы `hello-node`, используя IP-адрес и номер порта из предыдущего пункта, например:

```
$ curl 192.168.39.105:31302
Hello World!
```

10. Чтобы удалить службу `hello-node` и ее развертывание, а затем остановить виртуальную машину Minikube, введите следующие данные:

```
$ kubectl delete service hello-node
$ kubectl delete deployment hello-node
$ minikube stop
```

Кристофер Негус

Библия Linux

10-е издание

Перевел с английского *А. Павлов*

Заведующая редакцией	*Ю. Сергиенко*
Руководитель проекта	*А. Питиримов*
Ведущий редактор	*Н. Гринчик*
Литературный редактор	*Н. Рощина*
Художественный редактор	*В. Мостипан*
Корректоры	*О. Андриевич, Е. Павлович*
Верстка	*Г. Блинов*

Изготовлено в России. Изготовитель: ООО «Прогресс книга».
Место нахождения и фактический адрес: 194044, Россия, г. Санкт-Петербург,
Б. Сампсониевский пр., д. 29А, пом. 52. Тел.: +78127037373.

Дата изготовления: 08.2021. Наименование: книжная продукция. Срок годности: не ограничен.

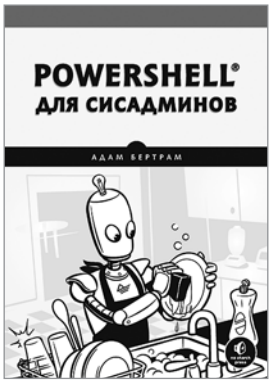
Налоговая льгота — общероссийский классификатор продукции ОК 034-2014, 58.11.12 — Книги печатные профессиональные, технические и научные.

Импортер в Беларусь: ООО «ПИТЕР М», 220020, РБ, г. Минск, ул. Тимирязева, д. 121/3, к. 214, тел./факс: 208 80 01.

Подписано в печать 21.07.21. Формат 70×100/16. Бумага офсетная. Усл. п. л. 74,820. Тираж 1000. Заказ 0000.

Адам Бертрам

POWERSHELL ДЛЯ СИСАДМИНОВ



PowerShell® — это одновременно язык сценариев и командная оболочка, которая позволяет управлять системой и автоматизировать практически любую задачу. В книге «PowerShell для сисадминов» обладатель Microsoft MVP Адам Бертрам aka «the Automator» покажет, как использовать PowerShell так, чтобы у читателя наконец-то появилось время на игрушки, йогу и котиков.

Вы научитесь

- Комбинировать команды, управлять потоком выполнения, обрабатывать ошибки, писать сценарии, запускать их удаленно и тестировать их с помощью фреймворка тестирования Pester.
- Анализировать структурированные данные, такие как XML и JSON, работать с популярными сервисами (например, Active Directory, Azure и Amazon Web Services), создавать системы мониторинга серверов.
- Создавать и проектировать модули PowerShell.
- Использовать PowerShell для удобной, полностью автоматизированной установки Windows.
- Создавать лес Active Directory, имея лишь узел Hyper-V и несколько ISO-файлов.
- Создавать бесчисленные веб- и SQL-серверы с помощью всего нескольких строк кода!

Реальные примеры помогают преодолеть разрыв между теорией и работой в настоящей системе, а легкий авторский юмор упрощает чтение.

Перестаньте полагаться на дорогое ПО и невнятные советы из сети!

Джейсон Андресс

ЗАЩИТА ДАННЫХ. ОТ АВТОРИЗАЦИИ ДО АУДИТА

Чем авторизация отличается от аутентификации? Как сохранить конфиденциальность и провести тестирование на проникновение? Автор отвечает на все базовые вопросы и на примерах реальных инцидентов рассматривает операционную безопасность, защиту ОС и мобильных устройств, а также проблемы проектирования сетей. Книга подойдет для новичков в области информационной безопасности, сетевых администраторов и всех интересующихся. Она станет отправной точкой для карьеры в области защиты данных.

Наиболее актуальные темы

- Принципы современной криптографии, включая симметричные и асимметричные алгоритмы, хеши и сертификаты.
- Многофакторная аутентификация и способы использования биометрических систем и аппаратных токенов для ее улучшения.
- Урегулирование вопросов защиты компьютерных систем и данных.
- Средства защиты от вредоносных программ, брандмауэры и системы обнаружения вторжений.
- Переполнение буфера, состояние гонки и другие уязвимости.

Лиз Райс

БЕЗОПАСНОСТЬ КОНТЕЙНЕРОВ. ФУНДАМЕНТАЛЬНЫЙ ПОДХОД К ЗАЩИТЕ КОНТЕЙНЕРИЗИРОВАННЫХ ПРИЛОЖЕНИЙ

Во многих организациях приложения работают в облачных средах, обеспечивая масштабируемость и отказоустойчивость с помощью контейнеров и средств координации. Но достаточно ли защищена развернутая система? В этой книге, предназначенной для специалистов-практиков, изучаются ключевые технологии, с помощью которых разработчики и специалисты по защите данных могут оценить риски для безопасности и выбрать подходящие решения.

Лиз Райс исследует вопросы построения контейнерных систем в Linux. Узнайте, что происходит при развертывании контейнеров, и научитесь оценивать возможные риски для безопасности развертываемой системы. Приступайте, если используете Kubernetes или Docker и знаете базовые команды Linux.