Глава 14

Веб-сервер Apache

Unix был разработан **AT&T** в конце 1960-х и начале 1970-х годов, и в те годы он был свободно распространен среди ряда крупных университетов. Когда **AT&T** начала взимать плату за Unix, ряд разработчиков университетов попытались создать клоны этой операционной системы. Один из этих клонов, **Linux**, был разработан и выпущен в начале 1990-х годов.

Многие из этих же университетов также развивали сеть, которая трансформировалась в Интернет. Благодаря последним усовершенствованиям, это делает Linux, пожалуй, самой дружелюбной сетевой операционной системой, доступной в Интернете. Обширные сетевые сервисы, доступные в Linux, не только являются лучшими в своей области, но и создают одну из самых мощных и полезных интернет-платформ, доступных сегодня по любой цене.

В настоящее время Apache является самым популярным веб-сервером в Интернете. Согласно опросу Netcraft (http://www.netcraft.com), в настоящее время Apache используется почти 50 процентами всех активных веб-сайтов в Интернете. Apache входит в комплект поставки RHEL 7.

В этой главе рассматриваются концепции, связанные с использованием веб-сервера Apache на базовом уровне конфигурации.

Внутри экзамена

Эта глава непосредственно посвящена пяти задачам в области **RHCE**. Хотя эти цели определяют протоколы **HTTP (Hypertext Transfer Protocol) и HTTPS (HTTP, secure)**, это неявная ссылка на веб-сервер **Apache**. Это единственный веб-сервер, поддерживаемый в настоящее время на **RHEL 7**. Целью является

- **Конфигурирование виртуальных хостов HostVirtual**

Вертуальные хосты являются хлебом и маслом для сервера **Apache**. Они поддерживают конфигурацию нескольких веб-сайтов на одном сервере.

- **Настройка частных (private) директорий**

Приватные директории на веб-сервере **Apache** ограничивают доступ для группы пользователей или хостов.

- **Настройка группового управления контентом (group-managed content).**

Порой группы пользователей должны совместно поддерживать содержимое веб-сайта. Поскольку частные директории могут быть настроены для отдельных пользователей в их домашних директориях, их можно настроить для групп пользователей в общей директории.

- **Развертывание базового приложения CGI**

Не беспокойтесь, если вы ни чего не знаете об интерфейсе общего шлюза **Common Gateway Interface** (**CGI**), но динамический контент на веб-страницах часто зависит от сценариев, таких как сценарии, связанные с **CGI**.

- **Настройка безопасности TLS.**

Мы уже встречались с **TLS** в предыдущих главах. **TLS** (и его предшественник, **SSL**) - это набор протоколов, используемых для шифрования сетевых соединений. Первоначально он был разработан в середине 1990-х годов для обеспечения проверки подлинности на основе сертификатов и безопасной связи для веб-браузера **Netscape Naviga-tor**. Поэтому неудивительно, что сегодня **TLS** по-прежнему играет важную роль в защите связи на **веб-сервере Apache**. Кроме того, существуют стандартные требования для всех сетевых служб, которые обсуждаются в **главах 10 и 11**. Подводя итог, необходимо установите службу, сделайте так, чтобы она работала с **SELinux**, убедитесь, что она запускается при загрузке, настройте службу для базовой работы и настройте безопасность на основе пользователя и хоста.

ЦЕЛЬ СЕРТИФИКАЦИИ 14.01

Веб-сервер Apache

Основанный на демоне **HTTP (httpd), Apache** обеспечивает простой и безопасный доступ ко всем типам контента с использованием обычного протокола **HTTP**, а также его безопасного кузена, **HTTPS**.

Apache был разработан на основе серверного кода, созданного Национальным центром суперкомпьютерных приложений (**NCSA**). Он включал в себя так много **патчей**, что стал известен как «**a patchy**» сервер. Веб-сервер **Apache** продолжает совершенствовать веб-технологии и предоставляет один из самых стабильных, безопасных, надежных и заслуживающего доверия веб-серверов. Этот сервер постоянно разрабатывается **Apache Software Foundation (http://www.apache.org).**

Чтобы получить полную копию документации **Apache**, обязательно включите RPM-пакет **httpd-manual** во время процесса установки. Он предоставит полную **HTML**-копию руководства **Apache** в каталоге **/usr/share/httpd/manual**, к которому можно перейти с локального сервера, указав в браузере **http://localhost/manual.**

Apache 2.4

Как и положено для надежности и стабильности, **RHEL** 7 включает в себя обновленную версию **Apache 2.4. В RHEL 6** включена более старая версия **Apache 2.2**. Однако **Apache 2.4**, включенный в **RHEL 7**, содержит все обновления, необходимые для поддержки новейших функций, с максимальной защитой от рисков, связанных с Интернетом.

Стек LAMP

Одна из возможностей **Apache** как веб-сервера заключается в том, что его легко интегрировать с другими программными компонентами. Наиболее распространенный набор известен как стек **LAMP**, который относится к его компонентам: **Linux, Apache, MySQL** и одному из трех языков сценариев (**Perl, Python или PHP**).

В целях **RHCE** для **RHEL 7** от вас ожидается установка **MariaDB**, разработанного сообществом форка системы управления базами данных **MySQL**. Мы обсудим установку и настройку **MariaDB** в **главе 17**.

Инсталяция

Пакеты **RPM**, требуемые **Apache**, включены в группу пакетов веб-сервера. Самый простой способ установить **Apache** с помощью следующей команды:

 # yum install httpd

Однако необходимы дополнительные пакеты услуг. Может быть проще установить обязательные пакеты и пакеты по умолчанию, связанные с группой пакетов веб-сервера, с помощью следующей команды:

 # yum group install "Web Server"

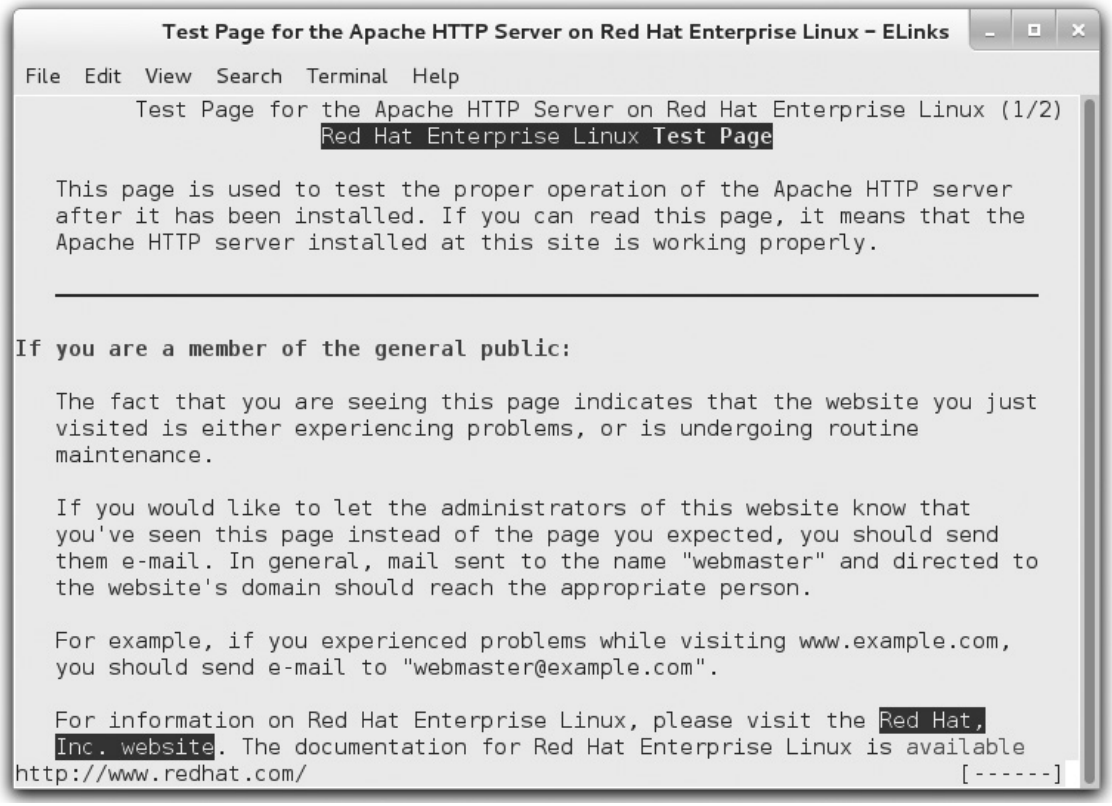
Существует также группа среды с именем **Basic Web Server**, которая устанавливает группу веб-серверов по умолчанию, но также включает в себя некоторые дополнительные группы, такие как клиент

MariaDB и **PostgreSQL**, расширения **Perl и PHP, Java** и так далее. Если вы не помните названия доступных групп, выполните команду **yum group list**. Стандартный метод запуска служб **Linux** - через утилиту **systemctl**. Однако вы можете остановить и запустить Apache, а также аккуратно перезагрузить файл конфигурации с помощью следующих команд:

 # apachectl stop
 # apachectl start
 # apachectl graceful

Стандартный пакет **Red Hat Apache** поддерживает базовые операции без дополнительной настройки. После запуска **Apache** запустите веб-браузер и введите URL-адрес **http://localhost**. Например, на **рисунке 14-1** в веб-браузере **elinks** отображается домашняя страница по умолчанию для **Apache**, основанная на конфигурации по умолчанию.

РИСУНОК 14-1 Установленная по умолчанию домашняя страница Apache



Эта веб-страница основана на содержимом файла **/etc/httpd/conf.d/welcome.conf**, в котором отображается файл **/usr/share/httpd/noindex/index.html**, если нет файла **index.html**. веб-сайта по умолчанию.

УПРАЖНЕНИЕ 14-1

Установите сервер Apache

В этом упражнении вы установите все пакеты, обычно связанные с сервером **Apache**. Затем вы настроите систему так, чтобы **Apache** был активен при следующей загрузке **Linux**. Суть в том, что вы будете делать все это из интерфейса командной строки. Это предполагает, что вы уже предприняли шаги, описанные в **главе 7,** чтобы либо зарегистрироваться на портале Red Hat, либо подключить систему к носителю **RHEL 7** (или перестроить DVD) в качестве хранилища.

1. Если вы находитесь в графическом интерфейсе, откройте консоль командной строки. Нажмите **alt-f2** и войдите в систему как пользователь **root.**
2. Выполните следующую команду, чтобы просмотреть доступные группы. Вы должны увидеть **«Basic Web Server»** в списке доступных групп среды.

 # yum group info

3. Проверьте, какие группы входят в группу среды «**Basic Web Server**». Вы должны увидеть веб-сервер в списке обязательных групп.

   # yum group info "Basic Web Server"

4. Отобразите пакеты, включенные в группу веб-серверов, с помощью следующей команды:

   # yum group info web-server

5. Вы можете установить все пакеты по умолчанию в группе пакетов веб-сервера с помощью следующей команды:

   # yum group install web-server

   Если вы просто установите пакет **RPM httpd**, другие важные пакеты могут не быть установлены, включая **mod_ssl**, для защищенных веб-сайтов, указанных в целях **RHCE**.
6. Выполните следующую команду, чтобы увидеь, настроен ли **Apache** для запуска при загрузке:

   # systemctl is-enabled httpd

7. Теперь используйте следующую команду, чтобы убедиться, что **Apache** запускается с целью по умолчанию при следующей загрузке **Linux**:

   # systemctl enable httpd

8. Запустите службу **Apache** с помощью следующей команды:

   # systemctl start httpd

9. Если вы еще этого не сделали в **главе 2**, установите текстовый веб-браузер. Стандарт RHEL 7 - это **elinks**, который можно установить с помощью следующей команды:

   # yum install elinks

10. Теперь запустите браузер **ELinks**, указывая на локальную систему, с помощью следующей команды:

#elinks http://localhost

11. Просмотрите результат. Вы видите тестовую страницу **Apache**?
12. Выход из браузера **ELinks**. Нажмите **q**, и когда появится текстовое меню **Exit ELinks**, нажмите **y**.
13. Создайте резервную копию файла конфигурации **httpd.conf** по умолчанию; логическое расположение - ваш домашний каталог.
14. Запустите команду **rpm -q httpd-manual**, чтобы подтвердить установку документации **Apache**. Поскольку этот пакет является частью группы пакетов веб-сервера по умолчанию, вы не должны получать сообщение «**не установлено**». Однако, если вы получите это сообщение, установите этот пакет с помощью команды **yum install httpd-manual**.
15. Просмотрите документацию, указав в браузере **ELinks** следующий **URL**:

    #elinks http://localhost/manual

Конфигурационные файлы Apache

Два ключевых файла конфигурации для веб-сервера **Apache: httpd.conf** в каталоге **/etc/httpd/conf** и **ssl.conf** в каталоге **/etc/httpd/conf.d**. Версии этих файлов по умолчанию создают общую

службу веб-сервера. Все файлы конфигурации находятся в трех каталогах: **/etc/httpd/conf**, **/etc/httpd/conf.d** и **/etc/httpd/conf.modules.d**. Они показаны на **рисунке 14-2**.

Apache может работать со многими другими программами, такими как **Python, PHP, прокси-сервер Squid и многое другое**. Если установлено, связанные файлы конфигурации обычно можно найти в директории **/etc/httpd/conf.d/**

РИСУНОК 14-2 Конфигурационные файлы Apache

```
[root@server1 ~]# ls /etc/httpd/conf
httpd.conf  magic
[root@server1 ~]# ls /etc/httpd/conf.d
autoindex.conf  manual.conf  ssl.conf      welcome.conf
fcgid.conf      README       userdir.conf
[root@server1 ~]# ls /etc/httpd/conf.modules.d
00-base.conf  00-lua.conf  00-proxy.conf  00-systemd.conf  10-fcgid.conf
00-dav.conf   00-mpm.conf  00-ssl.conf    01-cgi.conf
[root@server1 ~]#
```

Основной файл конфигурации Apache

В этом разделе рассматривается файл конфигурации **Apache** по умолчанию, **httpd.conf**. Мы рекомендуем вам использовать тестовую систему, такую как **server1.example.com**. Здесь обсуждаются только директивы по умолчанию в этом файле. Ознакомьтесь с комментариями; они содержат дополнительную информацию и варианты.

После того как **Apache** и **RPM-руководства по httpd** установлены **в упражнении 14-1,** обратитесь к **http://localhost/manual/mod/quickreference.html**. Он предоставляет подробную информацию по каждой директиве. Директивы по умолчанию приведены в следующих таблицах. **Таблица 14-1** определяет директивы, показанные в начале файла.

В **таблицах 14-1 и 14-2** директивы перечислены в порядке, указанном в версии **httpd.conf** по умолчанию. Если вы хотите поэкспериментировать с различными значениями для каждой директивы, сохраните изменения и затем используйте **systemctl restart httpd**, чтобы перезапустить демон **Apache**, или **systemctl reload httpd**, чтобы просто перечитать файлы конфигурации **Apache**.

В **таблице 14-2** указаны директивы, связанные с разделом «Конфигурация основного сервера».

Базовая конфигурация Apache для простого веб-сервера

Как описано в **Таблице 14-2, Apache** ищет веб-страницы в каталоге, указанном в директиве **DocumentRoot**. В файле **httpd.conf** по умолчанию эта директива указывает на **/var/www/html** каталог. Другими словами, все, что вам нужно для того, чтобы ваш веб-сервер установился и заработал, - это перенести веб-страницы в каталог **/var/www/html.**

Директива **DirectoryIndex** по умолчанию ищет файл веб-страницы **index.html** в этом каталоге. Стандартная страница **index.html RHEL 7** доступна в **/usr/share/doc/HTML/en-US** каталог. Скопируйте этот файл в каталог **/var/www/html**, а затем перейдите по адресу **http://localhost** с помощью браузера, такого как **ELinks**.

ТАБЛИЦА 14-1 Глобальные экологические директивы

Директива	Описание
ServerRoot	Устанавливает каталог по умолчанию для файлов конфигурации; любой относительный путь, указанный в конфигурации, является относительным путем к каталогу **ServerRoot.**
Listen	Указывает порт и, возможно, IP-адрес (для многосетевых систем) для прослушивания запросов.
Include	Добавляет содержимое других файлов конфигурации.
User	Указывает имя пользователя, которое **Apache** запускает как в локальной системе.
Group	Задает имя группы, в которой работает **Apache**, как в локальной системе.

ТАБЛИЦА 14-2 Основные директивы конфигурации сервера

Директива	Описание
ServerAdmin	Устанавливает административный адрес электронной почты; может быть показано (или связано) на страницах ошибок по умолчанию.
AllowOverride	Поддерживает переопределение предыдущих директив из **файлов .htaccess**.
Require	Предоставляет или запрещает доступ к каталогу для всех пользователей или определенных пользователей/групп.
DocumentRoot	Назначает корневой каталог для файлов сайта.
Options	Определяет функции, связанные с веб-каталогами, такими **как ExecCGI, FollowSymLinks, Includes, Indexes, MultiViews и SymLinksIfOwnerMatch**.
DirectoryIndex	Определяет файлы, которые нужно искать при переходе в каталог; по умолчанию устанавливается **index.html**.
ErrorLog	Находит файл журнала ошибок относительно **ServerRoot**.
LogLevel	Определяет уровень сообщений журнала.
LogFormat	Устанавливает информацию, включенную в файлы журнала.
CustomLog	Создает настроенный файл журнала, используя существующий формат журнала, с расположением относительно **ServerRoot**.
ScriptAlias	Подобно **Alias**, отображает веб-путь в расположение файловой системы вне **DocumentRoot**; в дополнение к псевдониму, он сообщает **Apache**, что указанный каталог содержит **CGI-скрипты**.
TypesConfig	Находит **mime.types**, который указывает типы файлов, связанные с расширениями.
AddType	Сопоставляет расширения файлов с указанным типом содержимого.
AddOutputFilter	Сопоставляет расширения файлов с указанным фильтром.
AddDefaultCharset	Устанавливает кодировку символов по умолчанию.
MIMEMagicFile	Обычно использует файл **/etc/httpd/conf/magic** для определения типа файла **MIME**.
EnableSendfile	Использует системный вызов **sendfile** для отправки статических файлов клиентам для улучшения производительности.

Базовое расположение файлов конфигурации и журналов определяется директивой **ServerRoot**. Значением по умолчанию из **httpd.conf** является

ServerRoot "**/etc/httpd**"

Рисунок 14-2 подтверждает, что основные файлы конфигурации **Apache** хранятся в подкаталогах **conf/, conf.d/** и **conf.d.modules/** в **ServerRoot**. Запустите команду **ls -l /etc/httpd**. Обратите внимание на символически связанные каталоги. Вы должны увидеть ссылку из каталога **/etc/httpd/logs** на каталог с реальными файлами журналов **/var/log/httpd**.

Логи Apache

Как предлагалось ранее, хотя файлы журнала **Apache** настроены для сохранения в каталоге **/etc/httpd/logs**, они фактически хранятся в каталоге **/var/log/httpd**. Фактически, **/etc/httpd/logs** является символической ссылкой на **/var/log/httpd**. Стандартная информация регистрации в **Apache** хранится в двух базовых файлах журнала. Пользовательские файлы журнала также могут быть настроены. Такие файлы журналов могут иметь разные имена, в зависимости от того, как настроены виртуальные хосты, как настроены безопасные веб-сайты и как чередуются журналы.

На основе стандартных файлов конфигурации **Apache** попытки доступа регистрируются в файле **access_log**, а ошибки записываются в файл **error_log**. Стандартные защищенные файлы журнала включают **ssl_access_log, ssl_error_log** и **ssl_request_log**.

В целом, полезно настроить разные наборы файлов журнала для разных веб-сайтов. Для этого вам также следует настроить различные файлы журналов для защищенных версий веб-сайта. Трафик на сайте важен при выборе частоты вращения лог файлов.

Существуют стандартные форматы файлов **журнала Apache**. Для получения дополнительной информации взгляните на директиву **LogFormat на рисунке 14-3**. Показаны три разных формата:

common, combined (аналогично общему, но также включает в себя веб-страницу, используемую для перехода на ваш сайт, тип и версию веб-браузера пользователя) и combinedio (такой же, как объединенный формат, плюс журнал байты, полученные и отправленные сервером и клиентом). Строки **LogFormat** включают несколько знаков процента, за которыми следуют строчные буквы. Эти директивы определяют, что входит в журнал.

РИСУНОК 14-3 Конкретные форматы журналов

```
# LogLevel: Control the number of messages logged to the error_log.
# Possible values include: debug, info, notice, warn, error, crit,
# alert, emerg.
#
LogLevel warn

<IfModule log_config_module>
    #
    # The following directives define some format nicknames for use with
    # a CustomLog directive (see below).
    #
    LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined
    LogFormat "%h %l %u %t \"%r\" %>s %b" common

    <IfModule logio_module>
      # You need to enable mod_logio.c to use %I and %O
      LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\" %I %O" com
binedio
    </IfModule>

    #
    # The location and format of the access logfile (Common Logfile Format).
    # If you do not define any access logfiles within a <VirtualHost>
    # container, they will be logged here.  Contrariwise, if you *do*
    # define per-<VirtualHost> access logfiles, transactions will be
    # logged therein and *not* in this file.
    #
    #CustomLog "logs/access_log" common
```

Затем вы можете использовать директиву **CustomLog**, чтобы выбрать местоположение для файла журнала, например, **logs/special_access_log**, и желаемый формат файла журнала, например, общий. Для получения дополнительной информации о файлах журналов и их форматах см. **http://localhost/manual/logs.html**.

!!!! On the job !!!!!
Некоторые анализаторы веб-журналов предъявляют особые требования к форматам файлов журналов. Например, популярный инструмент с открытым исходным кодом AWStats (Advanced Web Statistics) использует комбинированный формат журнала. AWStats - отличный инструмент для графического отображения активности сайта. Вы можете установить его из репозитория EPEL (Extra Packages for Enterprise Linux).
!!!!!!

ЦЕЛЬ СЕРТИФИКАЦИИ 14.02

Стандартная конфигурация безопасности Apache

Вы можете настроить несколько уровней безопасности для веб-сервера **Apache**. Межсетевые экраны, основанные на команде **firewall-cmd**, могут ограничивать доступ к определенным хостам. Параметры безопасности, основанные на правилах в файлах конфигурации **Apache**, также можно использовать для ограничения доступа для определенных пользователей, групп и хостов. Конечно, безопасные веб-сайты **Apache** могут шифровать связь. Если есть проблема, **SELinux** может ограничить риски.

Порты и брандмауэры

С помощью директив **Listen** и **VirtualHost** веб-сервер **Apache** определяет стандартные порты связи, связанные с протоколами **HTTP** и **HTTPS**, **80** и **443**. Чтобы разрешить внешнюю связь через отмеченные порты, вы можете настроить оба порта в качестве доверенных служб в межсетевом экране. **Инструмент настройки.**

Конечно, для систем, где **HTTP** и **HTTPS** настроены на нестандартных портах, вам необходимо соответствующим образом настроить соответствующие правила **firewall-cmd**.

Если вы просто открываете эти порты без разбора, брандмауэр пропускает трафик со всех систем. Может быть целесообразно установить расширенное правило для ограничения доступа к одной или нескольким системам или сетям. Например, следующее пользовательское правило расширенного доступа разрешает доступ ко всем системам, за исключением системы с **IP-адресом 192.168.122.150,** через **порт 80:**

```
firewall-cmd --permanent --add-rich-rule='rule family=ipv4 source address=192.168.122.150 service \
name=http reject'
firewall-cmd --reload
```

Аналогичные правила могут потребоваться для **порта 443**. Конечно, это зависит от требований работы и, возможно, экзамена **RHCE**.

Apache и SELinux

Посмотрите на настройки **SELinux**, связанные с **Apache**. Для обзора настройки **SELinux** в основном делятся на две категории: **логические настройки(boolean) и метки(labels) файлов**. Начните с **меток файла**.

Ярлыки(labels) файлов Apache и SELinux

Метки файлов по умолчанию для файлов конфигурации **Apache** являются согласованными, как показано в выходных данных для команд **ls -Z /etc/httpd и ls -Z /var/www**. Отдельные файлы используют тот же контекст, что и их каталог. Различия в контекстах файла показаны в **таблице 14-3.**

Первые пять являются просто контекстами **SELinux** по умолчанию для стандартных каталогов. Для веб-сайтов, где скрипты читают и/или добавляют данные в веб-формы, вы должны рассмотреть два последних контекста, которые поддерживают доступ на **чтение/запись (rw)** и **чтение/добавление (ra).**

Контексты, перечисленные в **таблице 14-3**, являются наиболее распространенными. Чтобы получить полный список всех файловых контекстов, связанных с веб-сервером Apache, и соответствующих им правил маркировки **SELinux**, выполните следующую команду:

```
# semanage fcontext -l | grep httpd_
```

Создать специальный веб-каталог

Во многих случаях вы создаете выделенные каталоги для каждого виртуального сайта. Лучше разделять файлы для каждого веб-сайта в их собственном дереве каталогов. Но с **SELinux** вы не можете просто создать специальный веб-каталог. Вы должны убедиться, что новый каталог, по крайней мере, соответствует контекстам **SELinux** каталога по умолчанию **/var/www**.

Запустите команду **ls -Z /var/www**. Обратите внимание на контексты **SELinux**. Для большинства подкаталогов **/var/www** тип по умолчанию - **http_sys_content_t**. Для вновь созданного каталога **/www** вы можете просто создать новое правило **SELinux** и изменить контексты файла следующей командой. **-R** применяет изменения рекурсивно, поэтому новые контексты применяются ко всем файлам и подкаталогам.

```
# semanage fcontext -a -t httpd_sys_content_t '/www(/.*)?'
# restorecon -R /www
```

Первая команда создает файл **file_contexts.local** в каталоге **/etc/selinux/target/contexts/files**. Если есть также подкаталог **cgi-bin/**, вы захотите установить соответствующие контексты для этого подкаталога также с помощью следующей команды:

 # semanage fcontext -a -t httpd_sys_script_exec_t '/www/cgi-bin(/.*)?'

ТАБЛИЦА 14-3 Контексты файлов SELinux для веб-сервера Apache

Каталог	Тип контекста SELinux
etc/httpd, /etc/httpd/conf, /etc/httpd/conf.d, /etc/httpd/conf.modules.d, /etc/httpd/run	**httpd_config_t**
/usr/lib64/httpd/modules	**httpd_modules_t**
/var/log/httpd	**httpd_log_t**
/var/www, /var/www/html	**httpd_sys_content_t**
/var/www/cgi-bin	**httpd_sys_script_exec_t**
n/a	**httpd_sys_rw_content_t**
n/a	**httpd_sys_ra_content_t**

Булевы настройки Apache и SELinux

Булевы настройки более обширны. В целях отображения мы изолировали их в инструменте администрирования **SELinux**, как показано на **рисунке 14-4**. По умолчанию включены только несколько логических настроек **SELinux**, и они описаны в **таблице 14-4**.

РИСУНОК 14-4 Логические настройки SELinux для Apache

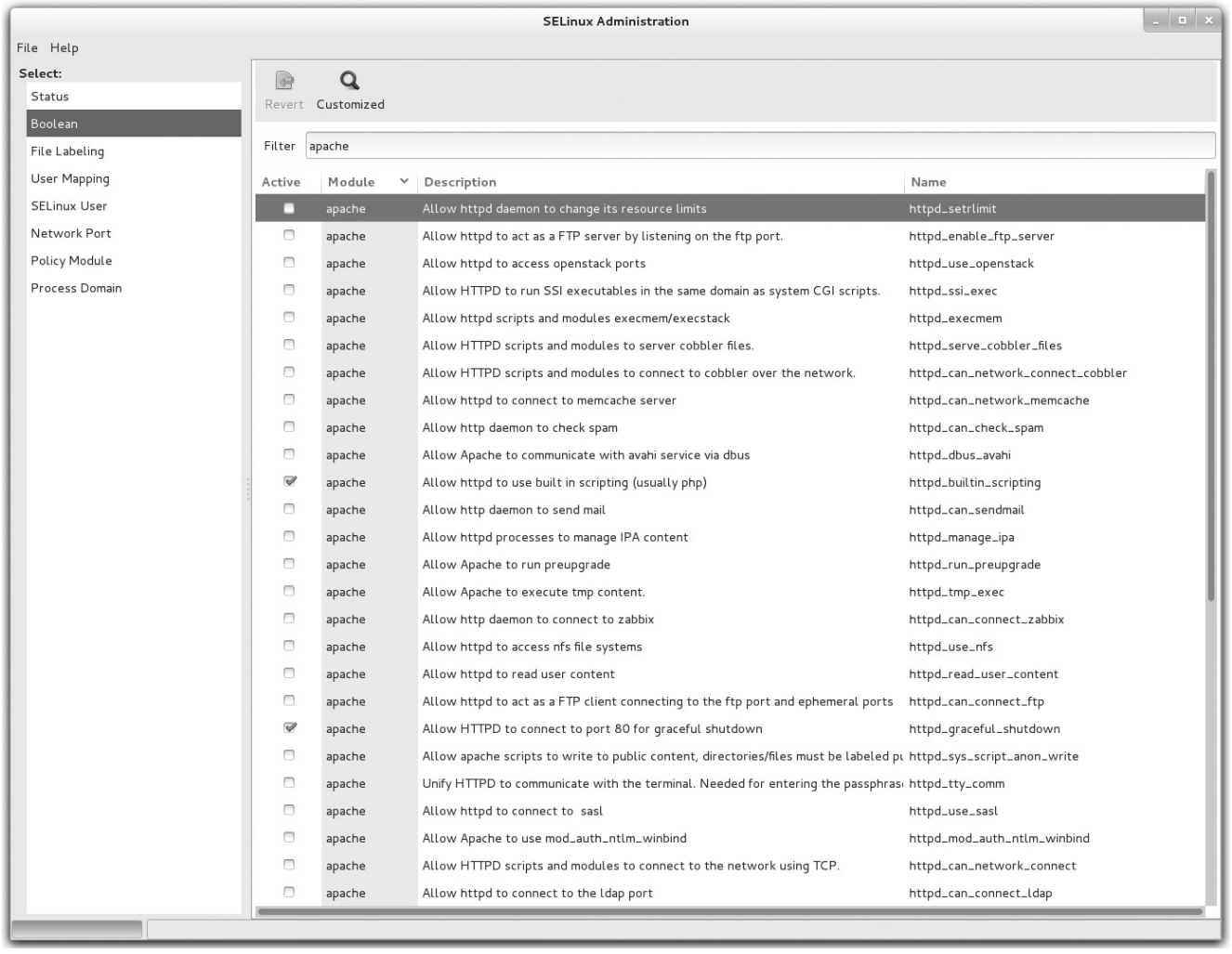


ТАБЛИЦА 14-4 Стандартные логические параметры SELinux, связанные с активным Apache

Активный логический	Описание

httpd_builtin_scripting	Поддерживает использование скриптов (таких как **PHP**)
httpd_enable_cgi	Позволяет сервисам **HTTP** выполнять сценарии **CGI**, помеченные **httpd_sys_script_exec_t** type
httpd_graceful_shutdown	Позволяет **Apache** подключаться к **порту 80** для корректного завершения работы

Из многих других опций **SELinux** обратите внимание на **httpd_enable_homedirs**, который поддерживает доступ к файлам в домашних каталогах пользователей. Другие сценарии, представляющие потенциальный интерес, относятся к взаимодействиям с другими службами, в частности, **httpd_enable_ftp_server, httpd_use_cifs** и **httpd_use_nfs**. Эти параметры позволяют **Apache** выступать в качестве **FTP-сервера**, а также читать общие каталоги **Samba/NFS**.

Использование этих и других отключенных параметров **SELinux Apache**, показанных на **рис. 14-4**, приведено в **таблице 14-5**. Все описания основаны на перспективе «Что бы произошло, если бы логическое значение было включено?». Для разнообразия термины **HTTP и Apache** используются взаимозаменяемо; Строго говоря, **Apache** - это один из вариантов сервисов **HTTP и HTTPS**.

Модуль управления

Веб-сервер **Apache** включает в себя множество модульных функций. Например, невозможно настроить веб-сайты, защищенные **SSL**, без пакета **mod_ssl**, который включает модуль **mod_ssl.so** вместе с файлом конфигурации **ssl.conf.**

Ряд других подобных систем организован в виде модулей. Загруженные модули включены в стандартные файлы конфигурации **Apache** с директивой **LoadModule**. Полный список доступных модулей находится в каталоге **/usr/lib64/httpd/modules**, но доступные модули не используются, если они не загружены с помощью директивы **LoadModule** в соответствующих файлах конфигурации **Apache** в каталог **/etc/httpd/conf.modules.d**

Безопасность внутри Apache

Вы читали (и, надеюсь, протестировали) параметры безопасности **Apache**, связанные с межсетевым экраном на основе зон, а также с **SELinux**. Теперь вы изучите параметры безопасности, доступные в основном файле конфигурации **Apache, httpd.conf**. Этот файл может быть изменен для защиты всего сервера или для настройки безопасности на основе каталога за каталогом. Каталог управляет безопасным доступом со стороны сервера, а также пользователей, которые подключаются к веб-сайтам на сервере.

Чтобы изучить основы безопасности **Apache**, давайте начнем с директивы **ServerTokens:**

ServerTokens OS

Эта линия выглядит обманчиво простой; он ограничивает информацию, которую **Apache** отправляет в своем заголовке ответа «**Server**». Эта информация иногда отображается, если вы переходите на несуществующую страницу, но вы также можете получить **HTTP**-заголовки, которые **Apache** отправляет клиентам, используя следующую команду:

curl --head http://localhost

Отредактируйте файл **httpd.conf** и добавьте строку ОС **ServerTokens** вверху. Затем перезагрузите конфигурацию сервера, запустив **systemctl reload httpd** и откройте веб-страницу по умолчанию в браузере. Вы должны увидеть следующий заголовок сервера:

Сервер: Apache / 2.4.6 (Red Hat Enterprise Linux)

Сравните этот результат с тем, что произойдет, если вы измените эту строку **на ServerTokens Full**:

Server: Apache/2.4.6 (Red Hat) OpenSSL/1.0.1e-fips mod_auth_kerb/5.4 mod_fcgid/2.3.9 mod_wsgi/3.4 Python/2.7.5

TABLE 14-5 Default Inactive Apache-Related SELinux Boolean Settings

Неактивный логический	Описание
httpd_anon_write	Позволяет веб-серверу записывать в файлы, помеченные Тип файла **public_content_rw_t.**
httpd_can_check_spam	Работает с веб-приложениями электронной почты для проверки на спам.
httpd_can_network_connect	Позволяет скриптам/модулям **Apache** устанавливать сетевые соединения **TCP**.
httpd_can_network_connect_cobbler	Включает скрипты/модули **Apache** для подключения к **Cobbler** по сети.
httpd_can_network_connect_db	Позволяет скриптам/модулям **Apache** подключаться к серверу базы данных по сети.
httpd_can_network_memcache	Позволяет Apache подключаться к серверу **memcache**.
httpd_can_network_relay	Поддерживает использование службы **HTTP** в качестве прямого или обратного прокси.
httpd_can_sendmail	Позволяет **Apache** отправлять почту.
httpd_enable_homedirs	Предоставляет **Apache** разрешение на доступ к файлам в домашних каталогах пользователей; файлы должны быть помечены как **httpd_sys_content_t SELinux**.
httpd_execmem	Поддерживает доступ из модулей **HTTP** к областям исполняемой памяти; некоторым приложениям **Java** может потребоваться это разрешение.
httpd_mod_auth_ntlm_winbind	Поддерживает аутентификацию в **Microsoft Active Directory**, если загружен модуль **mod_auth_ntlm_winbind**.
httpd_mod_auth_pam	Разрешает доступ к модулям аутентификации **PAM**, если загружен модуль **mod_auth_pam**.
httpd_setrlimit	Позволяет **Apache** изменять ограничения ресурсов, например максимальное количество файловых дескрипторов.
httpd_ssi_exec	Позволяет **Apache** выполнять сценарии включения на стороне сервера (**SSI**) на странице.
httpd_tmp_exec	Поддерживает выполнение скриптов в каталоге **/tmp**.
httpd_tty_comm	Поддерживает доступ к терминалу; необходим **Apache** для запроса пароля, если закрытый ключ сертификата **TLS** защищен паролем.
httpd_use_cifs	Включает доступ **Apache** к общим каталогам Samba, если они помечены типом файла **cifs_t**.
httpd_use_fuse	Позволяет **Apache** получать доступ к файловым системам **FUSE**, таким как тома **GlusterFS**.
httpd_use_gpg	Предоставляет разрешения **Apache** для запуска **gpg**.
httpd_use_nfs	Включает доступ **Apache** к общим каталогам **NFS**, если они помечены типом файла **nfs_t**.
httpd_use_openstack	Разрешает **Apache** доступ к портам **OpenStack**.
httpd_sys_script_anon_write	Настраивает доступ записи с помощью сценариев к файлам, помеченным **Тип файла public_content_rw_t.**

Другими словами, с помощью одной опции посторонние могут видеть, были ли загружены такие модули, как **FastCGI**, вместе с их номерами версий. Поскольку не все обновляют свое программное обеспечение совершенно своевременно, представьте себе, что происходит, когда хакер видит версию, которая была взломана. По этой причине мы рекомендуем вам установить **ServerTokens Prod**, чтобы ограничить объем информации о сервере, который отправляется клиентам.

Далее, посмотрите настройки доступа по умолчанию для всех файлов и каталогов в корневой файловой системе:

<Directory />
AllowOverride None
Require all denied

</Directory>

Это настраивает очень ограниченный набор разрешений. Строка **Require all denied** запрещает доступ ко всему контенту в корневой файловой системе для всех пользователей. Строка **AllowOverride None** отключает любые файлы **.htaccess**. Файл **.htaccess** находится внутри веб-каталога и содержит директивы, которые могут переопределять настройки веб-сервера по умолчанию.

Тем не менее, есть подходящее использование для файлов **.htaccess**. Например, в среде общего хостинга при размещении в подкаталоге, таком как **/www/html/customer023**, файл **.htaccess** может переопределить настройки сервера по умолчанию и разрешить доступ для аутентифицированных пользователей, и такие изменения будут применяться только к этому каталогу и его подкаталоги.

Вы также можете ограничить доступ ко всем доменам или **IP-адресам**, за исключением явно разрешенных, добавив следующие команды в нужный контейнер **<Directory>**:

```
Order Allow,Deny
Allow from example.com
Deny from all
```

Следующий контейнер <**Directory**> ограничивает доступ к **/var/www**, местоположению по умолчанию для файлов веб-сайта и **CGI-скриптов**:

```
<Directory "/var/www">
AllowOverride None
# Allow open access:
Require all granted
</Directory>
```

Директива " **Require all granted** " безоговорочно предоставляет доступ к содержимому **/var/www**. Следующий блок <**Directory**> регулирует доступ к каталогу **/var/www/html**, на который ссылается директива **DocumentRoo**t (хотя следующие директивы разделены многочисленными комментариями, все они находятся в одном контейнере):

```
<Directory "/var/www/html">
Options Indexes FollowSymLinks
AllowOverride None
Require all granted
</Directory>
```

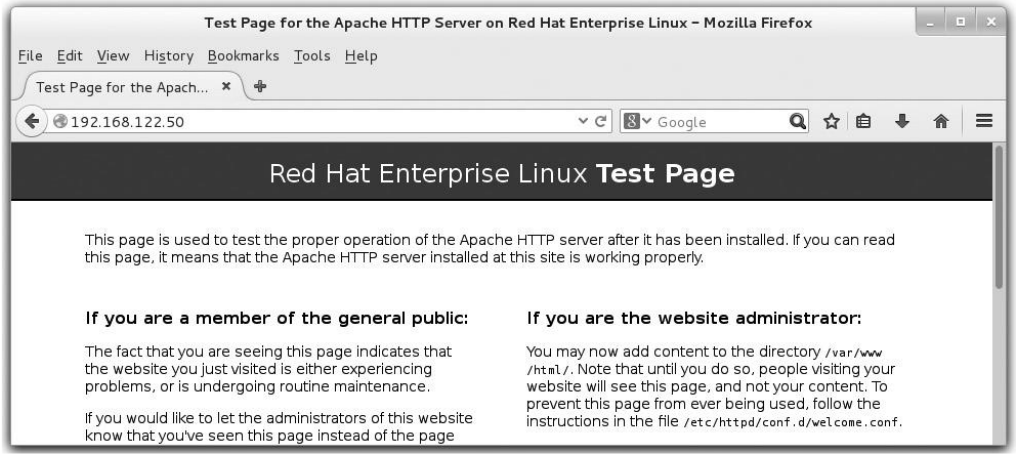
Директива **Options** включает две функции: параметр **Indexes** позволяет читателям просматривать список файлов на веб-сервере, если в указанном каталоге отсутствует **файл index.html**, а опция **FollowSymLinks** поддерживает использование символических ссылок.

Но подождите секунду! По умолчанию в каталоге **/var/www/html** нет файлов. Основываясь на описании, вы должны перейти к рассматриваемой системе и увидеть экран, показанный на **рисунке 14-5**. Поскольку в каталоге **/var/www/html** нет файлов, файлы не отображаются в выходных данных.

Однако при переходе на веб-сайт по умолчанию, связанный с сервером **Apache**, появляется страница, показанная на **рисунке 14-6**. Для получения дополнительной информации о том, как это работает, см. **Упражнение 14-2**.

РИСУНОК 14-5. Просмотр индекса файлов

РИСУНОК 14-6. Перейдите на тестовую страницу Apache по умолчанию.



Наконец, директива **Listen** определяет **IP-адрес и порт TCP/IP** для этого сервера. Например, значение по умолчанию, показанное ниже, означает, что этот сервер будет работать с каждым клиентом, который запрашивает веб-страницу с любого из **IP-адресов** вашего сервера через стандартный порт **TCP/IP, 80**:

Listen 80

Если в локальной системе доступно более одного **IP-адреса**, директива **Listen** может использоваться для ограничения доступа к одному конкретному **IP-адресу**. Например, если система имеет две сетевые карты **с IP-адресами 192.168.0.200/24 и 192.168.122.1/24**, следующая директива может помочь ограничить доступ к системам в сети **192.168.122.0/24**:

Listen 192.168.122.1:80

Для защищенных веб-сайтов есть вторая директива **Listen** в файле **ssl.conf** в каталоге **/etc/httpd/conf.d**. Данные из этого файла автоматически включаются в общую конфигурацию **Apache**, благодаря директиве, описанной в **упражнении 14-2**. Он включает в себя следующую директиву, которая указывает на порт безопасного **HTTP (HTTPS)** по умолчанию для **TCP/IP, 443**:

Listen 443 https

!!!!! Exam watch !!!!!!
Цели RHCE предполагают, что вы должны быть готовы к настройке обычных веб-сайтов HTTP и безопасных HTTPS.
!!!!!!!

УПРАЖНЕНИЕ 14-2

Приветствие Apache и история noindex.html

В этом упражнении вы проследите историю за стандартной тестовой страницей, связанной с веб-сервером **Apache**, как показано на **рисунке 14-6**. В этом упражнении предполагается, что пакет **httpd** уже установлен и служба **Apache** запущена. Вы также увидите, что произойдет, если путь к этой веб-странице будет поврежден, с указателем набора тестовых файлов в каталоге **/var/www/html**.

1. Откройте файл **httpd.conf** в каталоге **/etc/httpd/conf**. Найдите следующую строку:

   IncludeOptional conf.d/*.conf

   Директива **IncludeOptional conf.d/*.conf** включает содержимое файлов ***.conf** из каталога **/etc/httpd/conf.d** в конфигурации **Apache**. Выход из файла **httpd.conf**.
2. Перейдите в каталог **/etc/httpd/conf.d**. Откройте файл **welcome.conf**.

3. Определите и запишите параметры директивы **Alias**.
4. Обратите внимание на страницу **ErrorDocument**. Хотя он указывает на файл **/.noindex.html**, он основан на вышеупомянутой директиве **Alias**. Другими словами, вы сможете найти файл **index.html** в каталоге **/usr/share/httpd/noindex**.
5. Посмотрите на файл **/usr/share/httpd/noindex/index.html**. Чтобы открыть его в браузере **ELinks**, выполните команду **elinks /usr/share/httpd/noindex/index.html**. Веб-страница, которая появляется, должна теперь быть знакомой.
6. Выйдите из браузера нажав клавишу **q**. Переместите файл **welcome.conf** из каталога **/etc/httpd/conf.d** в папку для резервного копирования.
7. Перезагрузите конфигурацию **Apache** с помощью команды **systemctl reload httpd**.
8. Перейдите в систему **localhost** с помощью команды **elinks http://127.0.0.1**. Что вы видите?
9. Откройте второй терминал, перейдите в каталог **/var/www/html** и выполните команду **touch test {1,2,3,4}.**
10. Перезагрузите браузер в исходном терминале. В **ELinks ctrl-R** перезагружает браузер. Что вы видите?
11. Выйдите из браузера. Восстановите файл **welcome.conf** в каталоге **/etc/httpd/conf.d**.

УПРАЖНЕНИЕ 14-3

Создать список файлов

В этом упражнении вы будете настраивать список файлов, которыми вы будете делиться с другими пользователями вашего веб-сервера. Процесс довольно прост; вы настроите соответствующее правило брандмауэра, создадите подкаталог **DocumentRoot**, заполните его несколькими файлами, настроите соответствующие контексты безопасности и активируете **Apache**.

1. Убедитесь, что брандмауэр не блокирует доступ к **портам 80 и 443**. Один из способов сделать это - с помощью команды **firewall-cmd --list-all**, которая отображает все службы, включенные в зоне по умолчанию. В качестве альтернативы, вы можете использовать графический интерфейс **firewall-config**.
2. Создайте подкаталог **DocumentRoot**, который по умолчанию является **/var/www/html**. Для этого упражнения мы создали каталог **/var/www/html/help**.
3. Скопируйте файлы из каталога **/var/www/manual**:

   # cp -a /usr/share/httpd/manual/* /var/www/html/help/

4. Убедитесь, что служба **Apache** работает с помощью следующей команды:

   # systemctl status httpd

5. Убедитесь, что **Apache** запускается при следующей загрузке:

 # systemctl enable httpd

6. Используйте команды **ls -Zd /var/www/html и ls -Z /var/www/html/help** для просмотра контекста безопасности для каталога общего доступа и скопированных файлов. Если контекст безопасности еще не соответствует показанным здесь контекстам, настройте их с помощью следующей команды:

   # restorecon -R /var/www/html/help

7. Запустите браузер **ELinks** на локальном сервере, который находится в подкаталоге **help/:**

   # elinks http://127.0.0.1/help

8. Перейдите в удаленную систему и попробуйте получить доступ к тому же веб-каталогу. Например, если IP-адрес локальной системы - **192.168.122.50**, перейдите по адресу **http://192.168.122.50/help**. Если возможно, попробуйте это второй раз из обычного браузера с графическим интерфейсом.

Хост-безопасность

Вы можете добавить директивы **Order, allow и deny** для регулирования доступа на основе имен **хостов или IP-адресов**. Следующая стандартная последовательность команд разрешает доступ по умолчанию. Сначала он читает **директиву deny**:

Order deny,allow

Вы можете запретить или разрешить использование различных форм имен хостов или IP-адресов. Например, следующая директива запрещает доступ со всех компьютеров в домене **mheducation.com**:

Deny from mheducation.com

Если вы не хотите полагаться на службу **DNS**, вы можете использовать **IP-адреса**. В первой из следующих примерных директив используется один IP-адрес; в качестве альтернативы, вы можете настроить подсеть **192.168.122.0** в частичной нотации (**subnet in partial**), маске сети или CIDR (бесклассовой маршрутизации между доменами), как показано здесь:

Deny from 192.168.122.66
Allow from 192.168.122
Deny from 192.168.122.0/255.255.255.0
Allow from 192.168.122.0/24

!!!! Exam watch !!!!
Если вы установили Order allow,deny, доступ запрещен по умолчанию. Только те имена хостов или IP-адреса, которые связаны с директивой allow, имеют доступ.
!!!!!!

Безопасность на основе пользователя

Вы можете ограничить доступ к веб-сайтам, настроенным на сервере **Apache**, авторизованным пользователям с паролями. Как будет описано ниже, эти пароли могут отличаться от базы данных системной аутентификации.

Например, чтобы настроить защиту на уровне пользователя для веб-сайта, описанного в **упражнении 14-3**, вам необходимо настроить контейнер **<Directory>** в каталоге **/var/www/html/help**. Вам потребуется несколько команд в контейнере **<Directory>**:

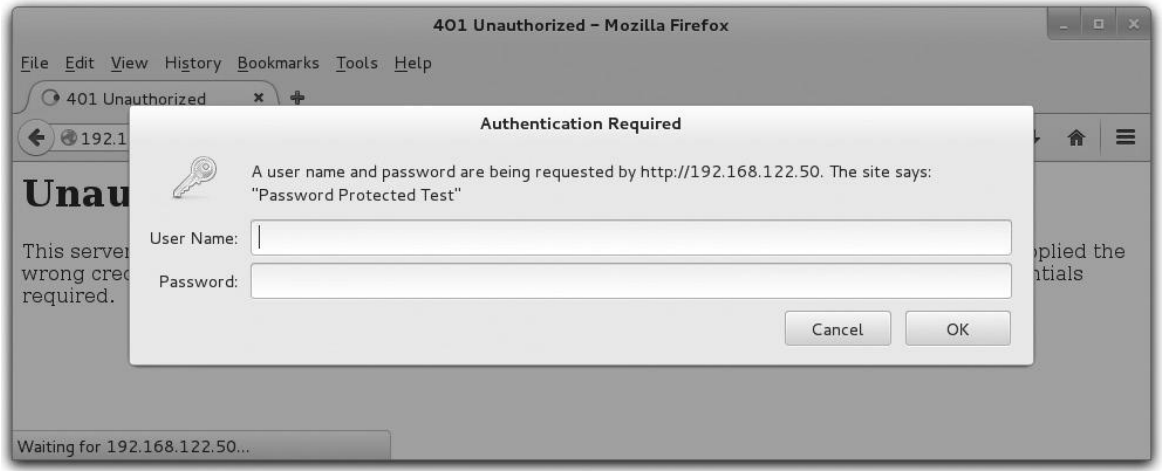
- Для настройки базовой аутентификации вам потребуется директива **AuthType Basic**.
- Чтобы описать сайт запрашивающим пользователям, вы можете включить в директиву **AuthName «*некоторые комментарии*»**.
- Чтобы обратиться к базе данных паролей веб-сервера с именем **/etc/httpd/webpass**, вам потребуется директива **AuthUserFile /etc/httpd/webpass**.
- Чтобы ограничить сайт одним пользователем с именем **engineer1**, вы можете добавить директиву **Require user engineer1**.
- В качестве альтернативы, чтобы ограничить сайт группой, как определено в **/etc/httpd/webgroups**, вы бы добавили директиву **AuthGroupFile /etc/httpd/webgroups**. Вам также понадобится директива, такая как **Require group design**, где **design** - это имя группы, указанной в веб-группах.

Вот пример кода, который мы добавили после контейнера **<Virtual Host>:**

```
<Directory "/var/www/html/help">
        AuthType Basic
        AuthName "Password Protected Test"
        AuthUserFile /etc/httpd/webpass
        Require user engineer1
</Directory>
```

С этой конфигурацией на **рисунке 14-7** показана подсказка имени **пользователя/пароля**, которая появляется при доступе к веб-сайту **http://server1.example.com/help** в обычном веб-браузере. Для аутентификации вам также необходимо создать **локальную базу паролей для Apache**. Мы рассмотрим эту тему в следующем разделе и в упражнении 14-4.

РИСУНОК 14-7. Защита паролем веб-сайта.



ЦЕЛЬ СЕРТИФИКАЦИИ 14.03

Специализированные каталоги Apache

В этом разделе вы изучите несколько вариантов специализированных каталогов **Apache**. Может быть целесообразно настроить специализированную защиту для некоторых из этих каталогов с помощью файла .**htaccess**. Как предлагалось ранее, вы можете настроить защиту паролем на основе пользователей и групп, что соответствует «частным каталогам», указанным в целях RHCE. Один пример, предварительно настроенный для частного домашнего каталога, показан в файле **conf.d/userdir.conf**. При правильных настройках такими каталогами также могут управлять члены группы.

После внесения каких-либо изменений в файлы конфигурации **Apache** вы можете проверить результат. Для этого вы можете запустить команду **systemctl restart httpd**. В качестве альтернативы, чтобы **Apache** перезагрузил файл конфигурации, не отключая пользователей, подключенных в данный момент, выполните команду **systemctl reload httpd**, которая функционально эквивалентна **apachectl graceful**.

Управление через файл .htaccess

При всей сложности, связанной с файлом **httpd.conf**, вы можете взглянуть на файл .**htaccess** и подумать: «Отлично, еще одно осложнение». Но при правильном использовании файл .**htaccess** может упростить список директив, применяемых к каталогу. или виртуальный хосту, поскольку его можно использовать для переопределения унаследованных разрешений. Для этого вам необходимо включить следующую команду в целевые контейнеры <**Directory**>:

AllowOverride Options

Затем вы можете настроить файлы .**htaccess** для переопределения ранее установленных опций каталога. Файл **.htaccess** может храниться в любом веб-каталоге, помеченном как тип **httpd_sys_content_t SELinux**.

Защищенный паролем доступ

Чтобы настроить пароли для веб-сайта, вам необходимо создать отдельную базу данных имен пользователей и паролей. Так же, как команды **useradd** и **passwd** используются для обычных

пользователей, так и команда **htpasswd** используется для настройки имен пользователей и паролей для **Apache**.

Например, чтобы создать файл базы данных с именем **webpass** в каталоге **/etc/httpd**, запустите следующую команду:

 # htpasswd -c /etc/httpd/webpass engineer1

Ключ -c создает указанный файл, и первый пользователь - **Engineer1**. Вам предлагается ввести пароль для **engineer1**. Пользователям в базе данных **webpass** не нужно иметь обычную учетную запись **Linux**. Обратите внимание на использование **каталога ServerRoot (/etc/httpd)**. Это также полезно при настройке виртуальных хостов.

Если вы хотите добавить больше пользователей в эту базу данных аутентификации, не указывайте **ключ -c**. Например, следующая команда настраивает вторую учетную запись для пользователя **drafter1**:

 # htpasswd /etc/httpd/webpass drafter1

Чтобы настроить доступ для более чем одного пользователя, вы также можете создать групповой файл. Например, чтобы настроить пользователей engineer1 и drafter1 в качестве **группы** с именем **design**, можно добавить следующую строку в файл **/etc/httpd/grouppass**:

design: engineer1 drafter1

В этом случае директива **AuthUserFile** будет связана с базой данных аутентификации **/etc/httpd/webpass**, а директива **AuthGroupFile** будет связана с базой данных группы.

Доступ К Домашнему Каталогу

Файл /etc/httpd/conf.d/userdir.conf по умолчанию содержит закомментированные директивы, которые могут обеспечить доступ к домашним каталогам пользователей. Одним из полезных вариантов является доступ к домашнему каталогу пользователя. Вы можете начать настраивать доступ к домашним каталогам пользователей, изменив следующие директивы

UserDir disabled
#UserDir public_html

на

#UserDir disabled
UserDir public_html

Тогда любой пользователь получит доступ к веб-страницам, которые пользователь помещает в свой каталог **~/public_html**. Например, пользователь с именем **michael** может создать каталог **/home/michael/public_html** и добавить веб-страницы по своему выбору.

Тем не менее, это требует некоторого компромисса безопасности; вам нужно сделать домашний каталог **michael исполняемым(executable)** для всех пользователей. Это также называется разрешением **701**, которое можно настроить с помощью следующей команды:

#chmod 701 /home/michael

Вам также необходимо сделать так, чтобы подкаталог **public_html** выполнялся всеми пользователями одинаково с помощью следующей команды:

#chmod 701 /home/michael/public_html

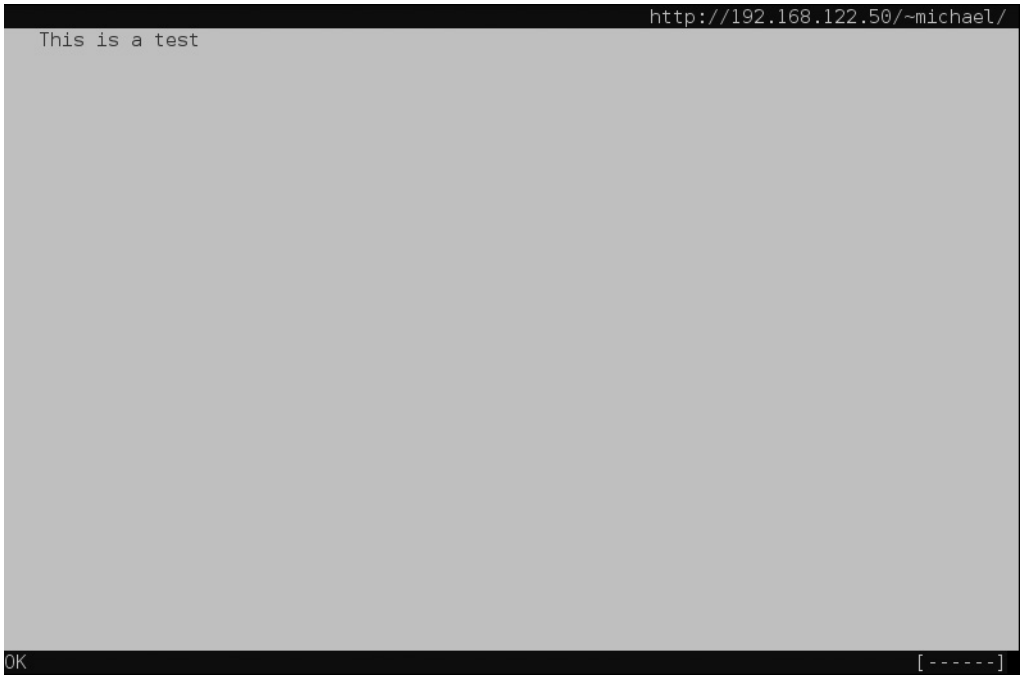
Но это влечет за собой некоторые риски безопасности. Даже если злонамеренный хакер не сможет непосредственно прочитать содержимое отмеченных каталогов, если он увидит скрипт на получающемся веб-сайте, он сможет выполнить этот скрипт как любой зарегистрированный пользователь.

Существует одна альтернатива для файловых систем с поддержкой списка контроля доступа **(ACL) (см. Главу 4).** Вы можете создавать **списки ACL** в указанных каталогах специально для пользователя с именем **apache**. Для пользователя **michael** и его домашнего каталога вы можете выполнить следующие команды:

 # setfacl -m u:apache:x /home/michael
 # setfacl -m u:apache:x /home/michael/public_html

Независимо от того, установлены ли разрешения напрямую или через **ACL**, следующим логическим шагом в качестве веб-сервера является добавление файла **index.html** в этот каталог. Для наших целей это может быть текстовый файл.

РИСУНОК 14-8 Посмотреть index.html файл для пользователя **Michael.**



Приведенный ниже прокомментированный контейнер является отличным способом помочь сохранить домашние каталоги, таким образом, более защищёнными для общего доступа.

Кроме того, **SELinux** должен быть настроен на «**Allow HTTPD To Read Home Directories**», связанный с логическим значением **httpd_enable_homedirs**. Вы можете активировать эту опцию либо с помощью **инструмента администрирования SELinux**, либо с помощью **команды setsebool -P httpd_enable_homedirs 1**.

В этот момент веб-сервер, может прочитать **index.html**-файл в подкаталоге **public_html** в каталоге пользователя **michael. Рисунок 14-8** иллюстрирует результат, где отображённый текст является единственным содержимым **index.html**. Обратите внимание, что пользовательские каталоги **public_html** доступны по **URL-адресу http://servername/~user**, где **user** - соответствующее имя пользователя.

Конечно, дополнительные настройки включены в файл **userdir.conf**. Контейнер, который начинается со следующей строки, поддерживает дополнительные уровни доступа к подкаталогу **public_html** всех домашних каталогов пользователей:

<Directory "/home/*/public_html">

Директива **AllowOverride** позволяет пользователям устанавливать файл .**htaccess** для переопределения настроек сервера по умолчанию, связанных с типами документов (**FileInfo**); доступ, связанный с директивами авторизации (**AuthConfig**); доступ защищен такими директивами, как **Allow**, **Deny** и **Order**; и переопределить настройки индексации каталога по умолчанию.

AllowOverride FileInfo AuthConfig Limit Indexes

Директива **Options** настраивает то, что можно увидеть в определенном каталоге, на основе согласования содержимого (**MultiViews**), списка файлов в текущем каталоге (**Indexes**), опции, которая допускает символические ссылки только в том случае, если они связаны с тем же владельцем (**SymLinksIfOwnerMatch**), а также активирует опцию, которая не разрешает скрипты (**IncludesNoExec**). Хотя использование сценария в пользовательском каталоге может быть плохой практикой безопасности, оно может подойти пользователям, являющимся разработчиками в тестовых системах, и, возможно, во время **экзамена Red Hat**. В этом случае вы удалили бы опцию **IncludeNoExec**:

Options MultiViews Indexes SymLinksIfOwnerMatch IncludesNoExec

Директива **Require** ограничивает доступ только к перечисленным **HTTP**-методам:

Require method GET POST OPTIONS

Вы можете объединить эти директивы с защитой паролем. Одной простой возможностью является **запрос имени пользователя и пароля пользователя**, чей домашний каталог используется совместно. Но, как отмечалось ранее, база данных аутентификации, сгенерированная **htpasswd**, не связана с набором теневых паролей. Вы можете использовать модуль **Apache mod_authnz_ldap**, если вы хотите реализовать аутентификацию и авторизацию для каталога **LDAP**. Однако это выходит за рамки экзамена RHCE.

Управление Групповыми каталогами

Вы можете объединить функции групповых каталогов, которые обсуждались **в Главе 8**, с только что описанным подкаталогом **public_html/.** Однако шаги, необходимые для настройки группы для **управления общим веб-контентом**, несколько отличаются. В частности, чтобы настроить управляемый группой каталог, лучше всего запустить эту группу как пользователя. Стандартные директивы конфигурации **Apache** для частного пользователя могут применяться к частным группам. Концептуально, вы должны предпринять следующие шаги:

1. Создать обычного пользователя.
2. Настройте этого пользователя с более высоким **UID** и номером **GID**, помимо тех, которые связаны с существующими локальными и сетевыми пользователями.
3. Настройте домашний каталог этого пользователя, указав в качестве владельца **nobody.** Установите **оболочку(shell)** входа в систему этого пользователя как **/sbin/nologin**.
4. Создайте подкаталог **public_html.**
5. Измените разрешения для домашнего каталога группы со связанными подкаталогами, чтобы соответствовать требованиям группы, описанным в **главе 8**, а также требованиям веб-сервера **Apache**. Например, если новый каталог группы **/home/design**, вы выполните следующую команду:

   #chmod -R 2771 /home/design

   Конечно, как обсуждалось в **главе 8**, вы можете заменить исполняемый **ACL**, ограниченный пользователем **apache**, битом выполнения для всех пользователей. В этом случае вы выполняете следующие команды:

   #chmod -R 2770 /home/design
   # setfacl -m u:apache:x/home/design
   # setfacl -m u:apache:x /home/design/public_html

6. Войдите как пользователь новой группы. Создайте новый файл в подкаталоге **public_html**. Проверьте владение этим файлом; с включенным в команду **chmod** битом **Super Group ID (SGID)** владелец группы должен быть владельцем всех файлов, созданных в подкаталоге **public_html**.
7. Внесите изменения, описанные ранее в этой главе, в файл **httpd.conf**, связанный с директивой **UserDir**.
8. Заставьте **веб-сервер Apache** перечитать файл.

У вас будет возможность настроить это в одной из глав лабораторной работы.

УПРАЖНЕНИЕ 14-4

Защита паролем для веб-каталога

В этом упражнении вы настроите защиту паролем для своей учетной записи обычного пользователя в подкаталоге **DocumentRoot**. Это включает использование директив **AuthType Basic**, **AuthName** и **AuthUserFile**. Это будет сделано с помощью стандартного веб-сайта **Apache**; Виртуальные хосты рассматриваются в следующем основном разделе.

1. Создайте резервную копию основного файла конфигурации **httpd.conf** из каталога **/etc/httpd/conf**. Затем откройте этот файл в текстовом редакторе.
2. Перейдите ниже строки **<Directory "/var/www/html"**>. Создайте новый контейнер для подкаталога **DocumentRoot**. Одним из вариантов является каталог **/var/www/html/chapter**. Первая и последняя директивы в строфе будут выглядеть так:

   <Directory "/var/www/html/chapter">
   </ Directory>

3. Добавьте следующие директивы: **AuthType Basic**, чтобы настроить базовую аутентификацию, директиву **AuthName** «**Test Protected Test**» для настройки комментария, который вы вскоре увидите, и директиву **AuthUserFile /etc/httpd/testpass**, указывающую на файл пароля. Замените свое обычное имя пользователя для **testuser** в поле «**Require user *testuser***».

   <Directory "/var/www/html/chapter">
   AuthType Basic
   AuthName "Password Protected Test"
   AuthUserFile /etc/httpd/testpass
   Require user testuser
   </Directory>

4. Проверьте синтаксис ваших изменений с помощью одной из следующих команд:

   # httpd -t
   # httpd –S

5. Предполагая синтаксис проверен, заставьте **Apache** перечитать файлы конфигурации:

 # systemctl reload httpd

6. Добавьте соответствующий файл **index.html** в каталог **/var/www/html/chapter**. Можно использовать текстовый редактор для ввода простой строки, такой как «**test was successful**». HTML-кодирование не требуется.
7. Создайте файл **/etc/httpd/testpass** с соответствующим паролем. В наших системах мы создали веб-пароль для пользователей **michael** и **alex** в указанном файле с помощью следующих команд:

   #htpasswd -c /etc/httpd/testpass michael
   #htpasswd /etc/httpd/testpass alex

   Если вы добавляете другого пользователя, **пропустите ключ -c.**
8. Проверьте результат, желательно из другой системы. (Другими словами, убедитесь, что брандмауэр разрешает доступ как минимум из одной удаленной системы.)
9. Теперь вы должны увидеть запрос имени пользователя и пароля с комментарием, связанным с директивой **AuthName**. Введите имя пользователя и пароль, только что добавленные в **/etc/httpd/testpass**, и просмотрите результат.
10. Закройте браузер и восстановите все предыдущие настройки.

ЦЕЛЬ СЕРТИФИКАЦИИ 14.04

Обычные и безопасные виртуальные хосты

Пожалуй, наиболее полезной функцией **Apache** является его способность обрабатывать несколько **веб-сайтов на одном IP-адресе**. В мире, где практически нет новых доступных адресов IPv4, это может быть полезно. Для этого вы можете **настроить виртуальные хосты для обычных веб-сайтов** в виде отдельных файлов конфигурации в каталоге **/etc/httpd/conf.d**. Таким образом, вы можете настроить несколько доменных имен, таких как **www.example.com и www.mheducation.com на тот же IP-адрес на том же сервере Apache**. Это называется **виртуальный хостинг на «основе имени (name-based)**».

И наоборот, вы можете настроить **разные IP-адреса для каждого виртуального хоста**. Это известно как виртуальный хостинг на основе IP (**«IP-based»**). Оба подхода верны, хотя виртуальный хостинг на основе имен обычно предпочтительнее, поскольку он может значительно снизить ваши общедоступные требования к IP.

!!!! On the job !!!!
Доменные имена example.com, example.org и example.net не могут быть зарегистрированы и официально зарезервированы Инженерной группой по Интернету (IETF) для документации. Многие другие примеры доменов. * Также зарезервированы соответствующими органами. !!!!!

Таким же образом вы можете создать несколько защищенных веб-сайтов, доступных по протоколу **HTTPS**. Несмотря на то, что детали различаются, основные директивы, связанные как с обычными, так и с защищенными виртуальными хостами, одинаковы.
Если вы используете текстовый браузер ELinks для проверки подключения к обычным и безопасным виртуальным веб-сайтам, созданным в этой главе, следует помнить о нескольких вещах:

Убедитесь, что файл **/etc/hosts** клиентских систем содержит IP-адрес с указанными полными доменными именами (FQDN). IP-адреса с разными FQDN являются нормальными. (Если есть DNS-сервер для локальной сети, вы можете пропустить этот шаг.)
Откройте файл конфигурации **/etc/elinks.conf** и установите для первой директивы в этом файле значение **0**, чтобы отключить проверку сертификата.
Чтобы получить доступ к обычному веб-сайту, обязательно укажите протокол перед полным доменным именем, например **http://vhost1.example.com или https://vhost2.example.com.**

Прелесть **VirtualHost** в том, что вы можете скопировать практически один и тот же контейнер, чтобы создать как можно больше веб-сайтов на сервере **Apache**, ограниченное только возможностями оборудования. Все, что требуется, это один **IP-адрес**. Следующий виртуальный хост может быть настроен с копией исходного контейнера **VirtualHost**. Все, что вам абсолютно необходимо изменить для виртуальных хостов на основе имен, - это **ServerName**. Большинство администраторов также изменят **DocumentRoot**, но даже это не является абсолютно необходимым. Вы увидите, как это работает для обычных и безопасных виртуальных хостов в следующих разделах.

!!!!EXAM wath !!!!!!
Будьте готовы создать несколько веб-сайтов на веб-сервере Apache, используя виртуальные хосты. Для этого лучше всего создать отдельные контейнеры VirtualHost в разных файлах конфигурации.
!!!!!!

Стандартный виртуальный хост

В **RHEL 6** стандартный **httpd.conf** включал примеры директив, которые можно использовать для создания одного или нескольких виртуальных хостов. Это уже не так, поэтому, если вы забудете синтаксис для создания нового виртуального хоста, вы можете посмотреть документацию по Apache по адресу **http://localhost/manual/vhosts**.

Как отмечалось ранее, директива **IncludeOptional conf.d/*.conf** автоматически включает информацию из файлов ***.conf** в этом каталоге. Имея это в виду, создайте и отредактируйте файл **vhost-dummy.conf** в каталоге **/etc/httpd/conf.d**.

Затем добавьте контейнер **<Directory>,** чтобы предоставить доступ к файлам содержимого веб-сайта. В следующем примере предполагается, что новый хост называется **dummy-host.example.com** и что содержимое сайта находится в каталоге **/srv/dummy-host/www**:

```
<Directory "/srv/dummy-host/www">
        Require all granted
</Directory>
```

Затем добавьте контейнер для конфигурации виртуального хоста:

```
<VirtualHost *:80>
        ServerAdmin webmaster@dummy-host.example.com
        DocumentRoot /srv/dummy-host/www
        ServerName dummy-host.example.com
        ServerAlias www.dummy-host.example.com
        ErrorLog logs/dummy-host.example.com-error_log
        CustomLog logs/dummy-host.example.com-access_log common
</VirtualHost>
```

Порт 80 по умолчанию для обслуживания веб-страниц. Вы также можете заменить **<VirtualHost 192.168.122.50:80**>, но, как правило, вы можете оставить эту директиву в том, что она поддерживает использование одного и того же IP-адреса для разных веб-сайтов.

Если вы читали описания первых двух разделов основной части файл **httpd.conf**, вы должны распознать большинство этих директив. Однако каждая директива указывает на нестандартные файлы и каталоги. Для просмотра:

- Адрес электронной почты, определенный **ServerAdmin**, включен во все сообщения об ошибках, которые возвращаются клиентам.
- Веб-страницы могут храниться в каталоге **DocumentRoot**. Убедитесь, что **контексты безопасности SELinux** любого создаваемого вами каталога **DocumentRoot** согласуются с контекстами каталога **/var/www** по умолчанию (и его подкаталогов). При необходимости примените команды **restorecon** и **semanage fcontext -a**, чтобы обеспечить соответствие контекстов безопасности. Обратите внимание, что по умолчанию политика **SELinux** уже помечает файлы в **/srv/*/www** типом **httpd_sys_content_t**.
- Основываясь на директиве **ServerName**, **Apache** знает, что запросы к **http://dummy-host.example.com** должны использовать конфигурацию, объявленную в блоке **<VirtualHost>**.
- **ServerAlias** указывает дополнительные имена, к которым может быть достигнут виртуальный хост.
- Директивы **ErrorLog** и **CustomLog** указывают относительный(***relative***) каталог журнала относительно **ServerRoot**. Эти файлы можно найти в каталоге **/etc/httpd/logs**. Обычно этот каталог связан сылкой с **/var/logs/httpd**.

Вы можете добавить дополнительные директивы для каждого контейнера виртуального хоста, чтобы настроить параметры виртуального хоста относительно основного файла конфигурации. Позже в этой главе вы настроите **CGI-скрипт** на виртуальном хосте с некоторыми пользовательскими директивами.

Легко настроить виртуальный хост-сайт. Подставьте **IP доменные имена**, **каталоги**, **файлы** и **адреса электронной почты** на ваш выбор. Создайте каталог **DocumentRoot**, если он еще не существует. Для этого мы настроили два виртуальных хоста со следующими контейнерами:

```
<Directory "/srv/vhost1.example.com/www">
        Require all granted
</Directory>
<VirtualHost *:80>
        ServerAdmin webmaster@vhost1.example.com
        DocumentRoot /srv/vhost1.example.com/www
        ServerName vhost1.example.com
        ErrorLog logs/vhost1.example.com-error_log
        CustomLog logs/vhost1.example.com-access_log common
```

</VirtualHost>

<Directory "/srv/vhost2.example.com/www">
Require all granted
</Directory>
<VirtualHost *:80>
ServerAdmin webmaster@vhost2.example.com
DocumentRoot /srv/vhost2.example.com/www
ServerName vhost2.example.com
ErrorLog logs/vhost2.example.com-error_log
CustomLog logs/vhost2.example.com-access_log common
</VirtualHost>

Не забудьте настроить файл **/etc/hosts** или **DNS-сервер для локальной сети с IP-адресами** для доменных имен виртуальных хостов, описанными ранее (**dummy-host.example.com, vhost1.example.com и vhost2.example.com**).

Вы также должны убедиться, что контексты **SELinux** соответствуют. Вы можете проверить синтаксис любых изменений конфигурации с помощью следующей команды:

 # httpd –t

Apache проверит вашу конфигурацию или определит конкретные проблемы. Когда вы запустите эту команду в конфигурации по умолчанию, вы получите следующее сообщение:

Syntax OK

Если вы создали несколько виртуальных хостов, вы также можете проверить их с помощью одной из следующих команд:

 # httpd -S
 # httpd -D DUMP_VHOSTS

Выходные данные должны содержать список по умолчанию и отдельных виртуальных хостов. Например, мы видим следующий вывод из нашей системы **server1.example.com RHEL 7**:

VirtualHost configuration:
***:443 is a NameVirtualHost**
default server server1.example.com (/etc/httpd/conf.d/ssl.conf:56)
port 443 namevhost server1.example.com (/etc/httpd/conf.d/ssl.conf:56)
wildcard NameVirtualHosts and _default_ servers:
***:80 is a NameVirtualHost**
default server vhost1.example.com (/etc/httpd/conf.d/vhost1.conf:1)
port 80 namevhost vhost1.example.com (/etc/httpd/conf.d/vhost1.conf:1)
port 80 namevhost vhost2.example.com (/etc/httpd/conf.d/vhost2.conf:1)

Безопасные виртуальные хосты

Если вы настраиваете защищенный веб-сервер, соответствующий протоколу **HTTPS**, **Red Hat** предоставляет для этой цели другой файл конфигурации**: ssl.conf** в каталоге **/etc/httpd/conf.d**. Если этот файл недоступен, вам необходимо установить пакет **mod_ssl**. Перед редактированием этого файла сделайте резервную копию.

Первая директива в **ssl.conf** гарантирует, что сервер прослушивает **TCP-порт 443**:

Listen 443 https

Как следует из заголовка, этот файл конфигурации содержит ряд других директив **SSL/TLS**. Как правило, никаких изменений не требуется для следующих строк:

SSLPassPhraseDialog exec:/usr/libexec/httpd-ssl-pass-dialog

SSLSessionCache shmcb:/run/httpd/sslcache(512000)
SSLSessionCacheTimeout 300
SSLRandomSeed startup file:/dev/urandom 256
SSLRandomSeed connect builtin
SSLCryptoDevice builtin

Теперь вы можете настроить виртуальные хосты с помощью следующих директив. В файле **ssl.conf** по умолчанию также есть контейнер виртуального хоста по умолчанию, его сложно прочитать со всеми комментариями. Поэтому образец пересмотренного файла конфигурации, ориентированный на контейнер виртуального хоста для системы vhost1.example.com, показан на рисунке **14-9**. Вы можете редактировать файл **ssl.conf** напрямую, хотя рекомендуется использовать отдельный файл конфигурации в **/etc/httpd/conf.d** для каждого виртуального хоста.

В версии файла **ssl.conf** по умолчанию проверьте контейнер **<VirtualHost _default_: 443>**. Сравните его с контейнером **<VirtualHost *: 80>** в предыдущей стандартной конфигурации виртуальных хостов. Некоторые изменения необходимы. Во-первых, вы должны заменить **_default_** в контейнере **VirtualHost** звездочкой **(*):**

<VirtualHost *:443>

```
<VirtualHost *:443>
    ServerAdmin webmaster@vhost1.example.com
    DocumentRoot /srv/vhost1.example.com/www
    ServerName vhost1.example.com

    ErrorLog logs/vhost1_ssl_error_log
    TransferLog logs/vhost1_ssl_access_log
    LogLevel warn

    SSLEngine on
    SSLProtocol all -SSLv2
    SSLCipherSuite HIGH:MEDIUM:!aNULL:!MD5
    SSLCertificateFile /etc/pki/tls/certs/localhost.crt
    SSLCertificateKeyFile /etc/pki/tls/private/localhost.key

    <Files ~ "\.(cgi|shtml|phtml|php3?)$">
        SSLOptions +StdEnvVars
    </Files>
    <Directory "/var/www/cgi-bin">
        SSLOptions +StdEnvVars
    </Directory>

    BrowserMatch "MSIE [2-5]" \
             nokeepalive ssl-unclean-shutdown \
             downgrade-1.0 force-response-1.0

    CustomLog logs/ssl_request_log \
              "%t %h %{SSL_PROTOCOL}x %{SSL_CIPHER}x \"%r\" %b"

</VirtualHost>
```

Не забудьте добавить службу **https** в зону по умолчанию на брандмауэре:

 # firewall-cmd --permanent --add-service=https
 # firewall-cmd –reload

В файле **ssl.conf** вы также должны включить директивы **ServerAdmin, DocumentRoot** и **ServerName**. Примеры директив, которые будут соответствовать виртуальным хостам, созданным в предыдущем разделе, включают следующее:

ServerAdmin webmaster@vhost1.example.com
DocumentRoot /srv/vhost1.example.com/www

ServerName vhost1.example.com

Хотя директива **DocumentRoot** может быть установлена для любого каталога, в организационных целях целесообразно хранить файлы, связанные с каждым виртуальным хостом, в выделенном каталоге.

Стандартные директивы журнала ошибок могут быть изменены. Фактически, если вы хотите, чтобы информация журнала для каждого защищенного веб-сайта настраивалась в разных файлах, их следует изменить, как показано ниже.

На основании директивы **ServerRoot** из файла **httpd.conf** эти файлы журнала можно найти в каталоге **/var/log/httpd**.

ErrorLog logs/vhost1_ssl_error_log
TransferLog logs/vhost1_ssl_access_log
LogLevel warn
CustomLog logs/vhost1_ssl_request_log "%t %h %{SSL_PROTOCOL}x %{SSL_CIPHER}x \"%r\" \ %b"

Директивы **TLS** в файле основаны на сертификатах по умолчанию для системы **localhost**. Вскоре вы увидите, как настроить новый сертификат **TLS**. Следующие пять директив по порядку активируют **SSL/TLS**, отключают небезопасный **SSL** версии **2**, поддерживают различные шифровальные шифры и указывают на сертификат **TLS** по умолчанию, а также файл ключа **TLS**:

SSLEngine on
SSLProtocol all -SSLv2
SSLCipherSuite HIGH:MEDIUM:!aNULL:!MD5
SSLCertificateFile /etc/pki/tls/certs/localhost.crt
SSLCertificateKeyFile /etc/pki/tls/private/localhost.key

Контейнер, который следует, относится к файлам с расширениями, связанными с динамическим контентом. Для таких файлов наряду с любыми файлами в стандартном каталоге **CGI** используются стандартные переменные среды SSL:

<Files ~ "\.(cgi|shtml|phtml|php3?)$">
SSLOptions +StdEnvVars
</Files>
<Directory "/var/www/cgi-bin">
SSLOptions +StdEnvVars
</Directory>

В следующем контейнере рассматриваются ситуации, связанные с клиентами, использующими устаревшие версии браузера **Microsoft Internet Explorer**:

BrowserMatch "MSIE [2-5]" nokeepalive ssl-unclean-shutdown downgrade-1.0 force-response-1.0

Конечно, контейнер виртуального хоста заканчивается следующей директивой:

</ VirtualHost>

Вам не нужно применять все перечисленные директивы к новому виртуальному хосту на основе TLS. Минимальная конфигурация показана далее. Это включает DocumentRoot, ServerName и директивы, которые включают TLS и настраивают путь сертификата:

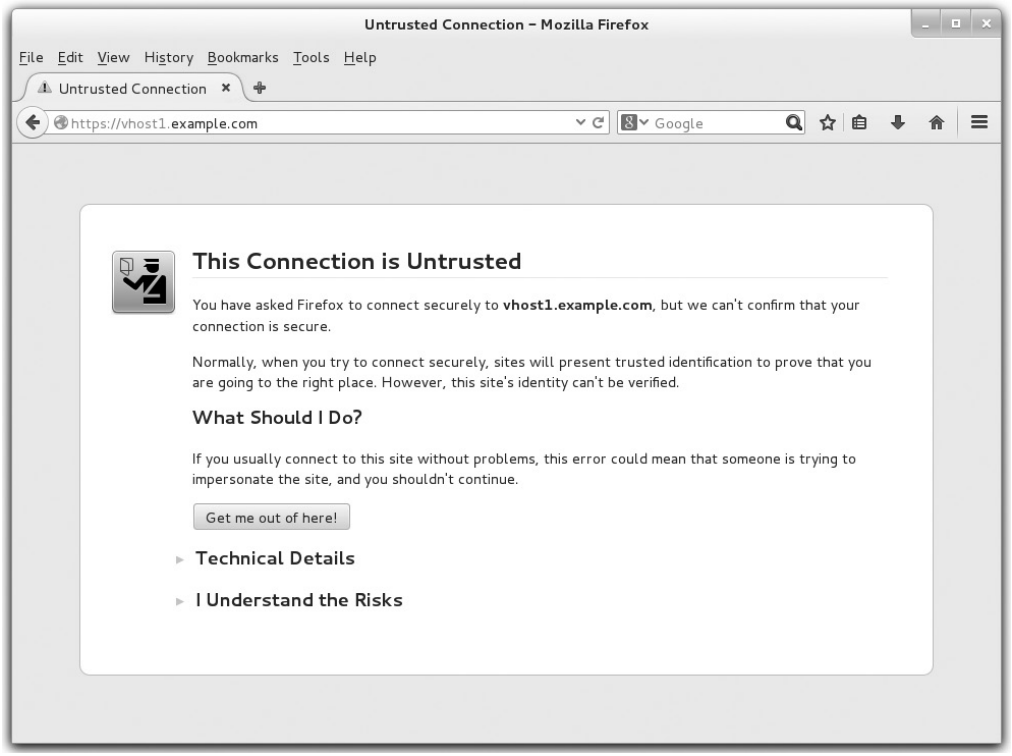
<VirtualHost *:443>
DocumentRoot /srv/vhost1.example.com/www
ServerName vhost1.example.com
SSLEngine on
SSLCertificateFile /etc/pki/tls/certs/vhost1.example.com.crt
SSLCertificateKeyFile /etc/pki/tls/private/vhost1.example.com.key
</VirtualHost>

Когда **Apache** настроен с ненадежным сертификатом, обычные клиенты, которые получают доступ к этому сайту, получают предупреждение о не безопасном веб-хосте, как показано на **рисунке 14-10.** В следующем сеансе вы увидите, как сгенерировать запрос сертификата, который будет подписан центром сертификации.

Создать новый сертификат TLS

Хотя сертификат **TLS** по умолчанию, указанный в файле конфигурации **ssl.conf**, может работать для базовой конфигурации, вы можете либо создать настраиваемый **самозаверяющий** сертификат, либо иным образом использовать действительный сертификат, подписанный авторитетным центром сертификации (**CA**), таким как **VeriSign** или **Thawte**. Перейдите в каталог **/etc/pki/tls/certs**. Обратите внимание на файл с именем **Makefile** в этом каталоге. Код в этом файле может использоваться командой **make** для создания нового сертификата для каждого виртуального хоста. В качестве альтернативы вы можете использовать команду **genkey** для автоматической генерации закрытого ключа и «**самозаверяющего сертификата**» для указанного полного доменного имени, как показано на **рисунке 14-11. Makefile** в этом каталоге. Код в этом файле может использоваться командой **make** для создания нового сертификата для каждого виртуального хоста. В качестве альтернативы вы можете использовать команду **genkey** для автоматической генерации закрытого ключа и «**самозаверяющего сертификата**» для **указанного полного доменного имени**, как показано на **рисунке 14-11**.

РИСУНОК 14-10. Предупреждение о безопасных хостах



 # genkey vhost2.example.com

Команда **genkey** удобна тем, что по завершении процесса она автоматически записывает ключ в каталог **/etc/pki/tls/private** и сертификат в каталог **/etc/pki/tls/certs.**

Для целей этого раздела выберите **Далее**, чтобы продолжить. На шаге, показанном на **рисунке 14-12,** вы должны выбрать размер ключа. В производственной среде обычно подходит размер по умолчанию **2048 бит**. Но в контексте экзамена вы можете выбрать меньший размер, чтобы сэкономить время. Генератор случайных чисел в **Linux** может потребовать дополнительной активности; это может быть отличным временем, чтобы отложить процесс и заняться чем-то другим.

Если вам больше нечего делать и вам нужно ускорить процесс, запустите некоторые из сценариев в каталоге **/etc/cron.daily**. Выполните некоторые из команд поиска, описанных в **главе 3**. Нажмите несколько раз в открытом терминале.

После того, как ключ сгенерирован, вам предлагается вопрос: генерировать ли запрос сертификата (CSR) для отправки в ЦС. Если у вас нет собственного внутреннего ЦС или вы фактически

не собираетесь приобретать подписанный сертификат в общедоступном ЦС, выберите «Нет», чтобы продолжить. Вам будет предложено зашифровать ключ с помощью ключевой фразы, как показано на **рисунке 14-13**.

РИСУНОК 14-11 Создать самозаверяющий сертификат

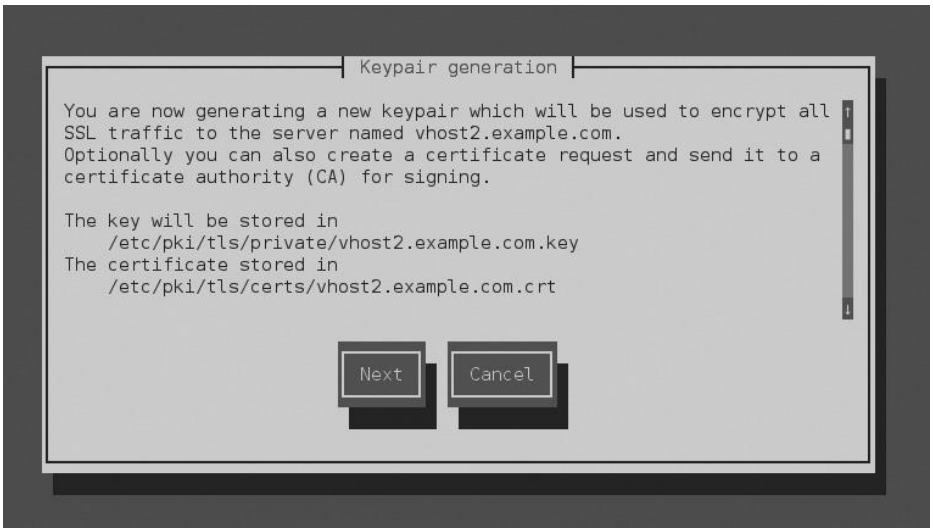


РИСУНОК 14-12 Выберите размер ключа для сертификата SSL.

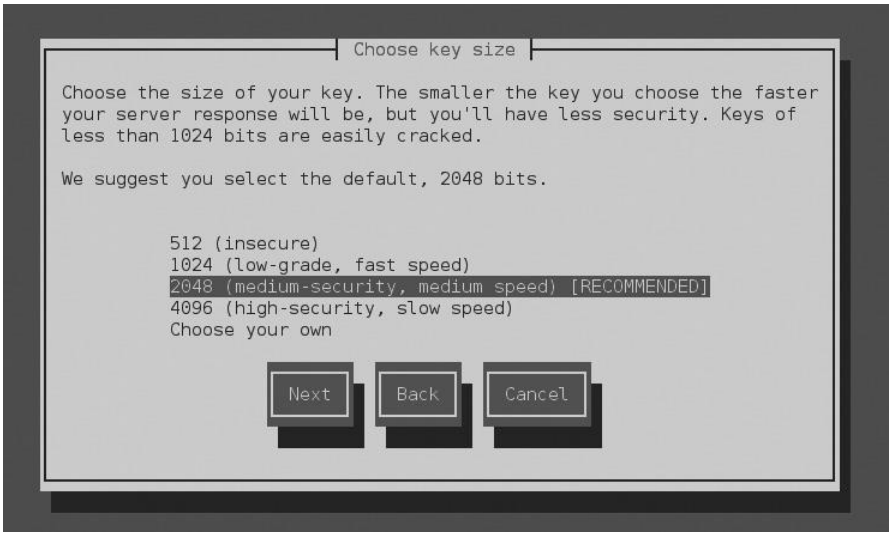


РИСУНОК 14-13 Возможность защитить паролем

Если безопасность наиболее важна, вы должны выбрать опцию **Зашифровать закрытый ключ(Encrypt the Private Key).** Если важна скорость, избегайте выбора. Сделайте выбор и нажмите **Далее**, чтобы продолжить. Если вы не выбрали опцию «**Зашифровать закрытый ключ**», вы сразу перейдете к сведениям о сертификате, показанным на **рис. 14-14**. Внесите соответствующие изменения и нажмите «**Далее**», чтобы продолжить.

В случае успеха вы увидите вывод, аналогичный показанному на **рисунке 14-15.**

РИСУНОК 14-13 Возможность защитить паролем

РИСУНОК 14-14 Данные SSL сертификата

РИСУНОК 14-15 Вывод команды SSL-сертификата

```
[root@server1 conf.d]# genkey vhost2.example.com
/usr/bin/keyutil -c makecert -g 2048 -s "CN=vhost2.example.com, O=My Company Ltd, L=Newb
ury, ST=Berkshire, C=GB" -v 1 -a -z /etc/pki/tls/.rand.23390 -o /etc/pki/tls/certs/vhost
2.example.com.crt -k /etc/pki/tls/private/vhost2.example.com.key
cmdstr: makecert

cmd_CreateNewCert
command:  makecert
keysize = 2048 bits
subject = CN=vhost2.example.com, O=My Company Ltd, L=Newbury, ST=Berkshire, C=GB
valid for 1 months
random seed from /etc/pki/tls/.rand.23390
output will be written to /etc/pki/tls/certs/vhost2.example.com.crt
output key written to /etc/pki/tls/private/vhost2.example.com.key


Generating key. This may take a few moments...

Made a key
Opened tmprequest for writing
/usr/bin/keyutil Copying the cert pointer
Created a certificate
Wrote 1682 bytes of encoded data to /etc/pki/tls/private/vhost2.example.com.key
Wrote the key to:
/etc/pki/tls/private/vhost2.example.com.key
[root@server1 conf.d]#
```

Тестовые страницы

Вам может понадобиться создать несколько файлов **index.html** для тестирования виртуальных хостов в различных ситуациях, в различных предварительных конфигурациях или даже во время экзамена. К счастью, экзамены Red Hat не проверяют знание HTML. Вы можете использовать веб-страницу **Apache** по умолчанию. Вы можете изменить эту или любую другую веб-страницу с помощью текстового или **HTML-редактора**.

Вы даже можете сохранить простой текстовый файл как **index.html**. Для целей этой главы все, что мы помещаем в файл **index.html** для обычного веб-сайта **vhost1.example.com**, представляет собой следующий текст:

Test web page for Virtual Host 1

После внесения соответствующих изменений в конфигурационные файлы **Apache** мы перезапустили сервис. Когда мы затем запустили **команду elinks http://vhost1.example.com**, появился экран, показанный на **рисунке 14-16**.

Проверка синтаксиса

Во многих случаях перезапуск **apachectl** и команды **httpd systemctl restart** выявляют синтаксические проблемы. Но это только во многих случаях. В некоторых случаях вы можете попытаться перезапустить **Apache**, приступить к тестированию результата с помощью клиентского браузера и получить разочарование, только чтобы обнаружить, что Apache не запустился из-за синтаксической ошибки. Чтобы минимизировать риск возникновения этой проблемы, следующая команда проверяет работу, которую вы проделали для редактирования файлов конфигурации Apache:

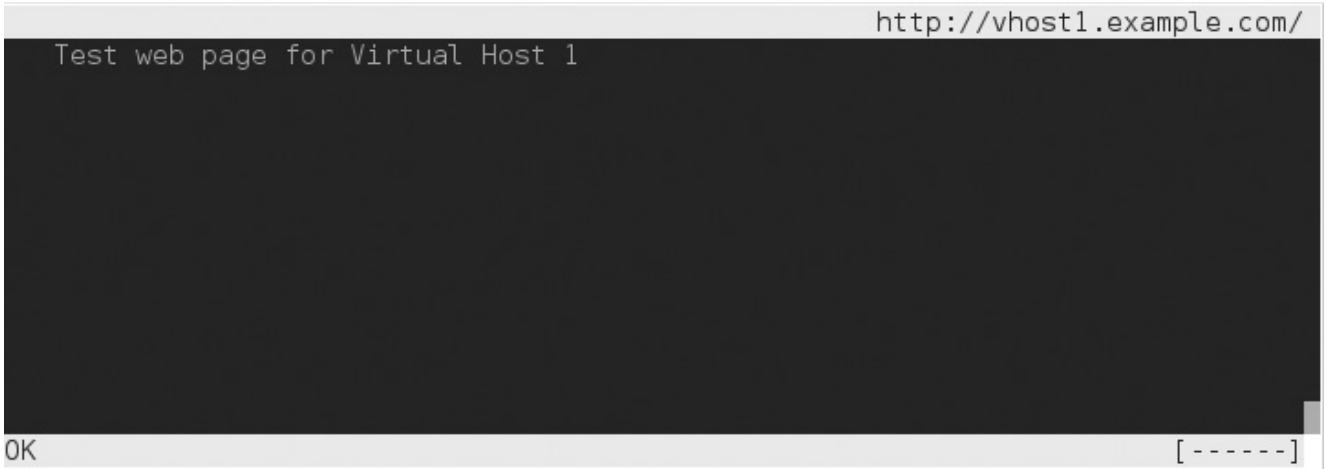
 # httpd –S

Вы также можете проверить сообщения журнала, захваченные журналом **systemd**:

 # journalctl -u httpd

Если проблем не обнаружено, вы сможете запустить локальный веб-сервер и подключиться с клиента с помощью запроса браузера.

РИСУНОК 14-16 Тестовая веб-страница



Устранение неполадок Apache

Когда установлены правильные пакеты **Apache**, конфигурация по умолчанию обычно создает работающую систему. Вы можете проверить основной синтаксис с помощью команды **httpd -t**. Но если вы настраиваете настоящий веб-сайт, вам, вероятно, нужно больше, чем просто тестовая страница.

Перед внесением изменений сделайте резервную копию файлов конфигурации Apache. Если что-то пойдет не так, вы всегда можете начать все сначала.

Некоторые ошибки Apache попадают в следующие категории:

- Сообщение об ошибке о невозможности привязки к адресу. Другой сетевой процесс уже может использовать **http-порт по умолчанию (80)**. Кроме того, **Apache** настроен на прослушивание **неправильного IP-адреса**.
- Ошибки сетевой адресации или маршрутизации. Дважды проверьте настройки сети. Для получения дополнительной информации о конфигурации сети см. Раздел 3 главы, посвященный настройке сети и устранению неполадок.
- **Apache** не работает Запустите **systemctl status httpd**. Проверьте **error_log** в каталоге **/var/log/httpd.**
- **Apache** не запускается после перезагрузки Запустите **systemctl is-enabled httpd**. Убедитесь, что **Apache (httpd)** настроен на запуск с соответствующей цели во время процесса загрузки с помощью команды

 # systemctl enable httpd

!!!!!! On the job !!!!!
Аадминистрирование Apache - необходимый навык для любого системного инженера Linux. Вы должны разработать возможность быстрой установки, настройки и устранения неполадок Apache. Вы также должны иметь возможность создавать и настраивать виртуальные веб-сайты.
!!!!!!!!!!!!!!!

УПРАЖНЕНИЕ 14-5

Настройте виртуальный веб-сервер

В этом упражнении вы настроите **веб-сервер** с виртуальным веб-сайтом. Вы можете использовать эту технику с разными каталогами для настройки дополнительных виртуальных веб-сайтов на одном сервере **Apache**.

1. Добавьте виртуальный веб-сайт для вымышленной компании LuvLinex с URL-адресом **www.example.com**. При необходимости используйте примеры конфигураций из **http://localhost/manual/vhosts**. Сохраните конфигурацию в файле **vhost-luvlinex.conf** в каталоге **/etc/httpd/conf.d**.
2. Назначьте директиву **DocumentRoot** директории **/luvlinex**. (Не забудьте также создать этот каталог в вашей системе.)
3. Предоставьте доступ к файлам для обслуживания с помощью директивы **Require all granted** внутри блока **<Directory>.**
4. Откройте файл **/luvlinex/index.html** в текстовом редакторе. Добавьте простую строку в текстовом формате, например

   This is the placeholder for the LuvLinex Website.

5. Сохраните этот файл.
6. Если вы включили **SELinux** в этой системе, вам придется изменить тип контекста и применить команду **restorecon** к каталогу **DocumentRoot:**

   # semanage fcontext -a -t httpd_sys_content_t "/luvlinex(/.*)?"
   # restorecon -R /luvlinex

7. Если вы используете службу **DNS**, обновите связанную базу данных. В противном случае обновите **/etc/hosts с www.example.com и соответствующим IP-адресом.**
8. Если вы хотите проверить синтаксис, выполните команды **httpd -t и httpd -D DUMP_VHOSTS.**
9. Не забудьте перезапустить службу **Apache**; правильный путь - с помощью команды **systemctl restart httpd.**

10. Убедитесь, что локальный брандмауэр настроен на предоставление доступа к соединениям со **службой HTTP:**

    # firewall-cmd --list-all
    # firewall-cmd --permanent --add-service = http
    # firewall-cmd --reload

11. Перейдите к удаленной системе. Обновите удаленный **/etc/hosts**, если необходимо. Откройте браузер по вашему выбору. Проверьте доступ к настроенному веб-сайту (**www.example.com**).
12. Закройте браузер в удаленной системе. Восстановите исходный файл конфигурации **httpd.conf**.

ЦЕЛЬ СЕРТИФИКАЦИИ 14.05

Развертывание базового приложения CGI

Когда вы видите цель **RHCE** «развернуть базовое приложение **CGI (deploy a basic CGI application**)», требование становится проще, чем кажется. Фактически, необходимые шаги можно прочитать из документации **Apache**, доступной из **пакета httpd-manual**. Когда приложение установлено, перейдите на страницу **http://localhost/manual**. Документация **Apache** должна появиться. Выберите CGI: динамический контент для подробных указаний, как описано в следующих разделах.

Изменения конфигурации Apache для файлов CGI

Чтобы позволить **Apache** читать файлы **CGI**, файл **conf.modules.d/00-cgi.conf** содержит директиву **LoadModule cgi_module**. Чтобы контролировать, какие каталоги содержат скрипты, **Apache** включает директиву **ScriptAlias**. Например, следующая директива **ScriptAlias** связывает путь **URL /cgi-bin/** с каталогом по умолчанию **/var/www/cgi-bin**:

ScriptAlias /cgi-bin/"/var/www/cgi-bin"

С помощью этой директивы **ScriptAlias**, если веб-сайт является **server1.example.com**, сценарии можно найти по адресу **http://server1.example.com/cgi-bin/.**

Кроме того, вы можете установить **CGI-скрипты** в каталоге, отличном от **/var/www/cgi-bin**, и соответственно изменить ссылку. Конфигурация блока **<Directory>** по умолчанию для **/var/www/cgi-bin** показывается следующим образом:

```
<Directory "/var/www/cgi-bin">
        AllowOverride None
        Options None
        Require all granted
</Directory>
```

Как указано в документации по веб-серверу **Apache**, доступной в пакете **httpd-manual**, вам необходимо внести изменения, чтобы позволить сценариям CGI вне каталога ScriptAlias фактически выполняться исполняемым сервером **Apache**:

```
<Directory "/home/*/public_html">
        AllowOverride None
        Options ExecCGI
        AddHandler cgi-script .pl
        Require all granted
</Directory>
```

В качестве меры безопасности команда **AllowOverride None** запрещает обычным пользователям изменять параметры конфигурации в этом каталоге с помощью файла **.htaccess**. Строка **Options ExecCGI** поддерживает исполняемые скрипты в указанном каталоге. Директива **AddHandler** связывает **CGI-сценарии** с файлами с расширением **.pl**. Строка «**Require all granted**» предоставляет доступ к каталогу всем пользователям.

Если сценарию **CGI** требуются для одного из ранее настроенных виртуальных хостов, вы можете настроить другую **ScriptAlias** и соответствующий контейнер **<Directory>.** Для **vhost1.example.com**, описанном ранее, мы добавляем следующие директивы:

```
ScriptAlias /cgi-bin/ /srv/vhost1.example.com/cgi-bin/
<Directory "/srv/vhost1.example.com/cgi-bin">
Options none
Require all granted
</Directory>
```

Настройка простого CGI-скрипта на Perl

Документация **Apache** содержит инструкции по настройке простого **CGI-скрипта** на языке программирования **Perl**. Убедитесь, что пакет **httpd-manual** установлен и локальная служба **httpd** активна. В браузере перейдите по адресу **http://localhost/manual**. В разделе «**How/To/Tutorials**» выберите **CGI: Динамический контент**. Прокрутите вниз до раздела «**Написание программы CGI**».

В этом разделе документация **Apache** предлагает простой **Perl-скрипт** с именем **first.pl**, основанный на следующем коде:

```
#!/usr/bin/perl
print "Content-type: text/html\n\n";
print "Hello, World!";
```

Создайте этот файл в каталоге **/srv/vhost1.example.com/cgi-bin**. Первая строка похожа на **#! / Bin / bash** shebang; в этом случае **perl** является интерпретатором команд. Объявлен тип содержимого, за которым следуют две новые строки (обозначенные **\n**). Последняя строка печатает выражение, обычно используемое для сценариев вводной программы. Сценарии **CGI** должны быть исполняемыми пользователем **apache**, и им обычно назначается разрешение **755**. Другими словами, после сохранения файла **first.pl** вы примените указанные разрешения с помощью следующей команды:

```
# chmod 755 first.pl
```

Запустите команду **ls -Z** для скрипта. В каталоге **/var/www/cgi-bin** он должен наследовать тип файла **SELinux httpd_sys_script_exec_t**, связанный с каталогом. При необходимости вы можете применить тип контекста к файлу и каталогу с помощью команды **restorecon**. Если каталог сценария настроен в пользовательском каталоге, отличном от **/var/www/cgi-bin**, убедитесь, что тип файла остается примененным после перемаркировки **SELinux** с помощью команды **semanage fcontext -a**:

```
# semanage fcontext -a -t httpd_sys_script_exec_t '/Srv/vhost1.example.com/cgi-bin(/.*)?
```

Соединения с сайтом

После настройки сценария **CGI** вы сможете получить доступ к этому сценарию из клиентского браузера. В этом упражнении предположим, что **Perl-скрипт first.pl** был настроен в системе **server1.example.com**. После этого вы сможете просмотреть результат с удаленной системы с помощью **команды elinks http://vhost1.example.com/cgi-bin/first.pl**. В случае успеха в теле браузера должны появиться следующие слова:

```
Hello, world!
```

Иногда может появляться сообщение об ошибке, например «Внутренняя ошибка сервера (**Internal Server Error**)». Наиболее вероятной причиной является сценарий **Perl**, который не имеет разрешений на выполнение для пользователя с именем **apache**. Повторим, это обычно решается путем предоставления этому **Perl-скрипту разрешения 755.**

СЦЕНАРИЙ И РЕШЕНИЕ.

Вам нужно настроить один веб-сайт.	Установите **Apache**; Настройте соответствующие файлы в каталоге **/var/www/html**.
Вам необходимо настроить несколько веб-сайтов.	С **Apache** используйте контейнеры **<VirtualHost>,** настроенные в отдельных файлах в каталоге **/etc/httpd/conf.d**.
Вам необходимо настроить безопасный **website**.	Натроить виртуальный хост в файле **ssl.conf** в каталоге **/etc/httpd/conf.d**.
Вам нужен специальный сертификат SSL для сайта www.example.org.	Запустите команду **genkey** www.example.org.
Служба **Apache** не запускается после перезагрузки.	Убедитесь, что служба **httpd** запускается в цели по умолчанию с помощью команды **systemctl enable httpd**. Если это нормально, проверьте содержимое файла **error_log** в каталоге **/var/log/httpd**.
Сценарии **CGI в Apache** не работают.	В файле конфигурации **Apache** убедитесь, что **ScriptAlias** указывает на соответствующий каталог; если **ScriptAlias** не настроен, убедитесь, что опция **ExecCGI** активна для каталога скриптов, а директива **AddHandler** указывает правильное расширение скрипта; убедитесь, что скрипт выполняется **Apache** и соответствует контекстам **SELinux** по умолчанию в каталоге **/var/www/cgi-bin**.

РЕЗЮМЕ СЕРТИФИКАЦИИ

Apache - самый популярный веб-сервер, используемый сегодня. Ключевые пакеты можно установить из группы пакетов «**Веб-сервер**». Пакет **httpd-manual** включает в себя локально просматриваемое руководство, которое может помочь с другими задачами настройки **Apache** даже во время экзамена. Основные конфигурационные файлы включают **httpd.conf** в каталоге **/etc/httpd/conf** и **ssl.conf** в каталоге **/etc/httpd/conf.d**. С помощью примеров контейнеров в обоих отмеченных файлах конфигурации вы можете создавать обычные и безопасные виртуальные хосты для нескольких веб-сайтов в одной системе, даже если доступен только один IP-адрес. Связанные файлы журналов хранятся в каталоге **/var/log/httpd**.

Вы можете разрешить доступ к **Apache** через порты **80** и **443** для некоторых или всех систем в **firewall-cmd**. Файлы и каталоги **Apache** связаны с несколькими разными контекстами **SELinux**. Различные функции **Apache** могут регулироваться множеством разных булевых настроек **SELinux.**

Директивы **Listen VirtualHost** направляют трафик на веб-сервер **Apache** через такие порты как **80** и **443**, вместе с указанными виртуальными хостами. Безопасность на уровне хоста и пользователя также может быть настроить в файлах конфигурации **Apache** с помощью таких команд, как **htpasswd** и таких директив как R**equire, Allow и Deny.**

При правильных параметрах безопасности возможны каталоги, управляемые пользователями и группами. На самом деле, есть закомментированный контейнер, который может включать содержимое в домашних каталогах пользователей.

Каталоги, управляемые группами, являются несколько более сложными, объединяя аспекты Apache-based пользовательские каталоги и каталоги общих групп, обсуждаемые в главе 8. Также для безопасности, новые сертификаты могут быть созданы для конкретного хоста, такого как www.example.org такой командой как **genkey www.example.org.**

Конфигурирование содержимого **CGI** на **веб-сайте Apache** проще, чем кажется. По факту, подробная информация о процессе предоставляется вместе с документацией **Apache**, включая сценарий **Perl**, который можно использовать для подтверждения правильности полученной конфигурации.

Пару минут проверки

Вот некоторые из ключевых моментов целей сертификации в главе 14.

Веб-сервер Apache

- **Red Hat Enterprise Linux** включает веб-сервер **Apache**, который в настоящее время используется больше интернет-сайтов, чем все остальные веб-серверы вместе взятые.
- Вы можете установить **Apache** и связанные пакеты как часть пакета группы **«Веб-сервер».**
- Файлы конфигурации **Apache** включают **httpd.conf** в каталоге **/etc/httpd/conf** и **ssl.conf** в каталоге **/etc/httpd/conf.d**.
- Информация логирования для **Apache** доступна в каталоге **/var/log/httpd**.

Стандартная конфигурация безопасности Apache

- **Apache** можно защитить с помощью правил **firewall-cmd** и различных логических выражений **SELinux** и **контекстов**.
- **Apache** поддерживает безопасность, указывая активные порты через директивы **Listen** и **VirtualHost.**
- **Apache** поддерживает безопасность на основе хоста по **IP-адресу** или **доменному имени.**
- **Apache** поддерживает пользовательскую безопасность по паролю с помощью команды **htpasswd.**

Специализированные каталоги Apache

- **Apache** упрощает настройку доступа к домашним каталогам пользователей в их подкаталогах **public_html/.**
- Групповые управляемые каталоги могут быть настроены так же, как домашние каталоги.

Обычные и безопасные виртуальные хосты

- Вы можете настроить несколько веб-сайтов на вашем сервере, даже с одним IP-адресом. Это возможно благодаря использованию виртуальных хостов.
- Конфигурация **RHEL** поддерживает виртуальные хосты для обычных и безопасных веб-сайтов, обычно устанавливаются в свои собственные файлы конфигурации в каталоге **/etc/httpd/conf.d**.
- Конфигурация **RHEL** включает в себя безопасный виртуальный хост по умолчанию в **/etc/httpd/conf.d/ssl.conf** файле.
- **SSL-сертификаты** могут быть созданы с помощью команды **genkey**.

Развертывание базового приложения CGI

- Использование содержимого **CGI** зависит от параметров конфигурации, таких как **ScriptAlias, ExecCGI** и **AddHandler cgi-скрипт**.
- Стандартные **CGI-скрипты** должны выполняться пользователем **Apache**. При необходимости образец инструкции приведены в руководстве **Apache**, доступном из пакета **httpd-manua**.

САМОПРОВЕРКА

Следующие вопросы помогут оценить ваше понимание материалов, представленных в этой главе. Поскольку на экзаменах **Red Hat** не появляется вопросов с несколькими вариантами ответов, вопросы с несколькими вариантами ответов не появляются и в этой книге. Эти вопросы исключительно проверяют ваше понимание главы. Это нормально, если у вас есть другой способ выполнения задачи. Получение результатов, а не запоминание пустяков - вот на что рассчитывает **Red Hat** экзамены. На многие из этих вопросов может быть более одного ответа.

Веб-сервер Apache

1. Какая директива **Apache**, определяющая базовый каталог для файлов конфигурации и журналов?
   _____

2. Как только вы измените **httpd.conf**, какая команда заставит **Apache** перечитать этот файл без выкинуть подключенных пользователей?
   _____

3. Какая директива указывает порт **TCP/IP**, связанный с **Apache**?
   _____

Стандартная конфигурация безопасности Apache

4. Какая команда создает файл **/etc/httpd/passwords** и настраивает пароль для пользователя Элизабет?

   _____

5. Если вы видите следующие директивы, ограничивающие доступ в контейнере для виртуального хоста, каким компьютерам разрешен доступ?

   Order Allow,Deny
   Allow from 192.168.0.0/24

   _____

6. Какие стандартные сервисы вам нужно открыть в **FirewallD**, чтобы разрешить доступ к обычному веб-сайту и безопасный доступ к веб-сайту?

   _____

Специализированные каталоги Apache

7. Какие обычные разрешения будут работать с домашним каталогом, который используется **Apache**?

   _____

8. Какие обычные разрешения будут работать с общим каталогом группы, который также является общим через **Apache**?

   _____

Обычные и безопасные виртуальные хосты

9. Какой файл **RHEL** предоставляет для настройки виртуального хоста в качестве защищенного сервера?

   _____

10. Если вы создаете виртуальный хост на основе имени, сколько IP-адресов потребуется для трех виртуальные серверы?

    _____

11. Чтобы проверить конфигурацию одного или нескольких виртуальных хостов, какую опцию вы можете использовать с командой **httpd**?

    _____

Развертывание базового приложения CGI

12. Какая опция с директивой **Options** поддерживает динамический контент **CGI в Apache** в конфигурационном файле?

    _____

Лабораторная работа

Во время экзаменов **Red Hat** задания будут представлены в электронном виде. Таким образом, эта книга также представляет большинство лабораторий в электронном виде. Для получения дополнительной информации см. Раздел «Лабораторные вопросы» в конце главы 14. Большинство лабораторных работ для этой главы просты и требуют очень небольшого количества команд или изменений в одном или двух файлах конфигурации.

Лабораторная работа 1

В этой первой лабораторной работе вы установите и настроите **Apache** для автоматического запуска и запуска при следующей загрузке системы. Вы также настроите домашнюю страницу по умолчанию для локального распространения в качестве домашней страницы по умолчанию для локального компьютера.

Домашняя страница по умолчанию для **RHEL 7** находится в каталоге **/usr/share/doc/HTML**. Для пользователей из **США** на английском языке соответствующий файл **index.html** находится в каталоге **en-US/home**. Другие подкаталоги применяются для пользователей с другими языками и национальными местоположениями. Отсутствующие значки приемлемы для целей этой лабораторной работы.

Лабораторная работа 2

В этой лабораторной работе вы изучите руководство по **Apache**, которое поставляется с пакетом **httpd-manual**. Когда вы ознакомитесь с этим руководством, оно может пригодиться на работе или даже в ситуации, когда доступ в Интернет недоступен, например, во время экзамена **Red Hat**.

Если пакет **httpd-manual** еще не установлен, сделайте это через группу пакетов «**Веб-сервер**». Активируйте **веб-сервер Apache** и используйте браузер для перехода по URL-адресу **http://localhost/manual**. Для этой лабораторной работы приемлем браузер с графическим интерфейсом. Нажмите на ссылку Виртуальные хосты и просмотрите информацию о виртуальных хостах на основе имен. Он включает больше примеров того, как виртуальные хосты могут быть настроены в файлах конфигурации **Apache**.

Вернуться к исходному **URL-адресу** руководства. Нажмите **CGI: динамический контент.** Просмотрите информацию о сценариях **CGI**, включая информацию о конфигурации каталога, чтобы разместить такие сценарии.

Лабораторная работа 3

В этой лабораторной работе вы настроите два веб-сайта на локальном сервере **Apache**. Назовите их **big.example.com** и **small.example.com**. Не забудьте создать необходимые каталоги для настройки этих сайтов. Убедитесь, что эти сайты доступны для пользователей с удаленных компьютеров в сети **example.com**. Добавьте соответствующий файл **index.html** в **DocumentRoot** для каждого веб-сайта. Допустимы простые веб-страницы, такие как одна строка текста (HTML-кодирование не требуется). Сконфигурируйте **SELinux** и настройте брандмауэр по мере необходимости, а также добавьте эти два хоста в **DNS** или файл **/etc/hosts** на тестовом клиенте.

Лабораторная работа 4

Настройте самозаверяющий сертификат для URL-адреса **big.example.com**, описанного в лабораторной работе 3. Повторите процесс для **URL-адреса small.example.com.**

Лабораторная работа 5

Продолжая работу с **Apache**, настройте безопасные версии для каждого из двух веб-сайтов, созданных в **Лабораторной работе 3**, используя сертификаты, показанные в **Лабораторная работа 4**. Убедитесь, что для каждого защищенного веб-сайта доступны соответствующие каталоги.

Лабораторная работа 6

Настройте **Apache** таким образом, чтобы запросы к **http://servername/~ user** обслуживались каталогом **public_html** в домашнем каталоге соответствующего пользователя. Доступ через браузер должен быть ограничен владельцем этого каталога посредством обычной аутентификации.

Лабораторная работа 7

Настройте каталог для группы пользователей, доступ к которой через браузер ограничен членами этой группы. Для целей этой лабораторной работы группа называется **techsupport**, а пользователи - **john, ivan** и **rajiv**.

Лабораторная работа 8

Сконфигурируйте обычный веб-сайт **big.example.com** для обработки сценариев **CGI** в подкаталоге **cgi-bin/ DocumentRoot**. В случае успеха вы сможете перейти по адресу **http://big.example.com/cgi-bin/hello.pl** и увидеть слова **«Hello World».**

ОТВЕТЫ НА САМОПРОВЕРКУ

Веб-сервер Apache

1. Директива **ServerRoot** устанавливает каталог по умолчанию для сервера **Apache**. Любые файлы и каталоги, не настроенные иным образом или не настроенные как относительный каталог, задаются относительно **ServerRoot.**
2. Существует два основных способа заставить **Apache** перечитать файл конфигурации без перезапуска оказание услуг. Вы можете сохранить работу **Apache** и перечитать файл с помощью такой команды, как **apachectl graceful** или **systemctl reload httpd**. **Команда kill -HUP $(cat /run/httpd/httpd.pid)** также является приемлемым ответом.
3. Директива **Listen** указывает порт **TCP**, связанный с **Apache**.

Стандартная конфигурация безопасности Apache

4. Команда, которая создает файл **/etc/httpd/passwords** и настраивает пароль для пользователя **elizabeth** - это **htpasswd -c /etc/httpd/passwords**(или другой файл для паролей) **elizabeth**. Если **/etc/httpd/passwords** уже существует, все, что требуется, это **htpasswd /etc/httpd/passwords** elizabeth.
5. Как описано в главе, директива Order Allow, Deny запрещает доступ ко всем системам по умолчанию, кроме тех, которые явно разрешены доступ. Поэтому доступ ограничен компьютерами из сети 192.168.0.0/24(доступ разрешён только из указанной сети).
6. Стандартные сервисы, которые вам нужно открыть в FirewallD, чтобы разрешить доступ к обычным и безопасным **веб-сайты http** и **https.**

Специализированные каталоги Apache

7. Соответствующие разрешения - **701**, права на выполнение для других пользователей. Как «обычный разрешения », списки ACL не являются опцией.
8. Соответствующие разрешения - **2771**, которые объединяют разрешения **SGID**, разрешения **rwx** для каталога общей группы и права на выполнение для других пользователей. Как «обычные разрешения» указано, **списки ACL не вариант.**

Обычные и безопасные виртуальные хосты

9. Файл, связанный с защищенными серверами для виртуальных хостов, - это **ssl.conf** в каталоге **/etc/httpd/conf.d.**
10. Для виртуального сервера на основе имени требуется один IP-адрес, независимо от количества сконфигурированных виртуальных сайтов.
11. Чтобы проверить конфигурацию виртуальных хостов, вы можете использовать один из двух опций с команды **httpd. httpd -S** проверяет файл конфигурации, включая настройки виртуального хоста. С другой стороны, **httpd -D DUMP_VHOSTS** фокусируется на конфигурации виртуального хоста и поэтому также является приемлемый ответ.

Развертывание базового приложения CGI

12. Директива **Options ExecCGI** обычно используется в настроенных **Apache** каталогах, которые содержат C**GI-скрипты**, такие как **Perl-программы**.

ОТВЕТЫ Лабораторной работы

Лабораторная работа 1

Сначала убедитесь, что веб-сервер **Apache** установлен. Если команда **rpm -q httpd** сообщает вам, что **httpd** отсутствует, группа пакетов веб-сервера еще не установлена. Самый эффективный способ

сделать это командой **yum group install “WebServer”**. (Чтобы найти подходящие имена групп пакетов, запустите команду **yum group list hidden**.) Эти действия предполагают наличие надлежащего соединения с хранилищем (**repository**), как описано в разделе Глава 7.

Чтобы запустить **Apache**, запустите команду **systemctl start httpd**. Чтобы убедиться, что сервис запустится в следующий раз после перезагрузки системы, выполните команду **systemctl enable httpd.**

После установки **Apache** вы сможете получить к нему доступ из браузера через **http://localhost**. В файле конфигурации **Apache** по умолчанию **/etc/httpd/conf/httpd.conf**, вы можете убедиться, что **DocumentRoot** указывает на **/var/www/html** каталог. Затем вы можете скопировать файл **index.html** из каталога **/usr/share/doc/HTML/en-US** в каталог **/var/www/html**. Затем вы можете проверить результат, еще раз перейдя на **http://localhost**.

Если вы не скопировали другие файлы, связанные с домашней страницей по умолчанию, дисплей будет отсутствовать некоторые значки, но это не проблема для этой лабораторной работы.

Лабораторная работа 2

Это информационная **Лабораторная работа**. Когда она будет завершена, вы сможете ссылаться на эти настройки **Apache,** что бы найти советы в ситуациях, когда эта книга и Интернет недоступны, например, во время экзамена Red Hat. Конечно, вы должны изучить эти советы заранее. Если вы забыли синтаксис одной или двух команд, эти файлы могут быть спасителями.

Лабораторная работа 3

Эта лабораторная работа требует, чтобы вы создали два виртуальных хоста. Хотя, конечно, есть и другие способы настройки разные виртуальные хосты, описание в этом лабораторном ответе - один из методов, и важно, чтобы вы знали хотя бы один метод для создания виртуального хоста. Один из способов сделать это состоит в следующем:

1. Директива **ServerRoot** для системы устанавливает каталог по умолчанию для сервера **Apache**. Любые файлы и каталоги, не настроенные иным образом или не настроенные как относительные пути, задаются относительно к **ServerRoot**. Не меняйте эту настройку.
2. В каталоге **/etc/httpd/conf.d** создайте два файла с именами **vhost-big.conf** и **vhost-small.conf** для конфигурации двух виртуальных хостов.
3. Добавьте отдельные контейнеры **VirtualHost** с настройками, подходящими для виртуального хоста **big.example.com.**
4. Назначьте **ServerAdmin** для адреса электронной почты администратора этого веб-сайта.
5. Настройте уникальный каталог **DocumentRoot** для **big.example.com**.
6. Установите первое имя сервера как **big.example.com.**
7. Добавьте директивы **ErrorLog** и **CustomLog** и установите для них уникальные имена файлов в каталог **/etc/httpd/logs** (который связан с каталогом **/var/logs/httpd**). С помощью **ServerRoot** по умолчанию вы можте использовать относительный путь, такой как следующий:

   ErrorLog logs/big.example.com-error.log
   CustomLog logs/big.example.com-access.log combined

8. Обязательно закройте контейнер **VirtualHost** (директивой **</ VirtualHost>** в конце контейнера).
9. Добавьте контейнер **<Directory>,** чтобы предоставить доступ к каталогу, который вы настроили как **DocumentRoot**:

   <Directory "/srv/vhost-big/www">
   Require all granted
   </Directory>

10. Повторите процесс для второго веб-сайта, указав для второго **ServerName** значение **small.example.com.**
11. Закройте и сохраните файлы конфигурации с вашими изменениями.
12. Создайте все новые каталоги, которые вы настроили с помощью директив **DocumentRoot**.
13. Создайте текстовые файлы **index.html** в каждом каталоге, определенном связанным новым директивы **DocumentRoot**. Не беспокойтесь о **HTML-коде**; текстовый файл отлично подходит для этой лабораторной работы.

14. Убедитесь, что эти доменные имена настроены на локальном **DNS-сервере** или в файле **/etc/hosts**. Например, если сервер **Apache** находится в системе с **IP-адресом 192.168.122.150** (например, **tester1.example.com**), вы можете добавить следующие строки в **/etc/hosts**:

    192.168.122.150 big.example.com
    192.168.122.150 small.example.com

    Эти же данные должны быть включены в файл **/etc/hosts** удаленной клиентской системы.

15. Используйте утилиту **firewall-cmd**, чтобы разрешить передачу данных **HTTP** через брандмауэр:

    # firewall-cmd --permanent --add-service=http
    # firewall-cmd --reload

16. Вам может потребоваться настроить соответствующий контекст файла **SELinux** в каталоге, связанном с **DocumentRoot**. Например, если этот каталог **/vhost-big**, один из способов сделать это следующие команды:

    # semanage fcontext -a -t httpd_sys_content_t '/vhost-big(/.*)?'
    # restorecon -R /vhost-big

    Обратите внимание, что первая команда не требуется, если вы установили **DocumentRoot** в каталог что политика **SELinux** по умолчанию помечена типом **httpd_sys_content_t**, например **/srv/vhost-big/www**. Фактически, каталоги **/srv/*/www** имеют тип контекста по умолчанию, уже установленный в **httpd_sys_content_t**. Если сомневаетесь, проверьте настройки политики **SELinux**, запустив **semanage fcontext -l | grep httpd_sys_content_t**.
17. Обязательно запустите команду **systemctl reload httpd**, чтобы **Apache** перечитал файлы конфигурации, с внесенными вами изменениями.
18. Теперь вы можете проверить результаты. Перейдите к удаленной системе и попробуйте получить доступ к вновь созданным сайтам в браузере по вашему выбору. Если всё работает, доменные имена **big.example.com и small.example.com** должны отображать файлы **index.html**, созданные для каждого веб-сайта.
19. Если есть проблемы, проверьте синтаксис с помощью команд **httpd -t** и **httpd -S**. Проверьте журналы в каталоге **/var/log/httpd**.

Лабораторная работа 4

Эта лаборатория должна быть простой; когда он будет завершен, вы должны найти следующие два файла, которые может использоваться для поддержки виртуального хоста для безопасной версии веб-сайта **big.example.com**:

/etc/pki/tls/certs/big.example.com.crt
/etc/pki/tls/private/big.example.com.key

Соответствующие файлы для системы **small.example.com** теперь также должны существовать в этих каталогах. Процесс основан на стандартных ответах на вопросы, сгенерированные **genkey big.example.com** и **genkey small.example.com** команды.

Лабораторная работа 5

Основы этой лаборатории просты. Вам нужно будет повторить те же шаги, которые вы выполнили в **Лабораторная работа 3** и использовать файлы сертификатов и ключей, созданные в **Лабораторная работа 4**. Вы можете использовать содержимое файла **/etc/httpd/conf.d/ssl.conf** в качестве шаблона. Кроме того, вы должны быть озабоченны следующим:

1. Хотя это и не обязательно, вы можете настроить **DocumentRoot** в каталоге отличается от обычного веб-сервера. В противном случае одна и та же веб-страница будет отображаться как для обычные и безопасные версии сайта.
2. Рекомендуется настроить **ErrorLog** и **CustomLog** с соответствующими именами файлов для

возможности определить, что информация взята из защищенной версии данного веб-сайта.

3. Полезно скопировать **директивы SSL** из шаблона виртуального хоста **SSL** из **файла ssl.conf**. Все директивы могут применяться к защищенным версиям **big.example.com** и **small.example.com** веб-сайты. Вам нужно как минимум установить директиву **ServerName**, включить **SSLEngine** и установить правильные пути к **SSLCertificateFile** и **SSLCertificateKeyFile**:

ServerName big.example.com
SSLEngine On
SSLCertificateFile /etc/pki/tls/certs/big.example.com.crt
SSLCertificateKeyFile /etc/pki/tls/private/big.example.com.key

Конечно, вы должны заменить **small.example.com** на **big.example.com** для директивы в защищенном контейнере виртуального хоста для этого сайта. Не забудьте добавить **https** в Зоны по умолчанию на брандмауэре:

 # firewall-cmd --permanent --add-service=https
 # firewall-cmd --reload

Лабораторная работа 6

В файле **conf.d/userdir.conf** по умолчанию конфигурация домашних каталогов пользователей требует, чтобы вы включить директиву **UserDir**. Затем вы можете настроить прокомментированный контейнер, связанный с пользователем домашние каталоги. В случае успеха только один пользователь может получить доступ к своему домашнему каталогу через **Веб-сервер Apache** из клиентского браузера. В общем, вы можете увидеть такие директивы, как следующие в контейнер для данной домашней страницы:

AuthType Basic
AuthName "Just for one user"
AuthUserFile /etc/httpd/oneuser
Require user michael

Как предлагается в этой главе, домашний каталог должен иметь обычные права на выполнение для других пользователей или, по крайней мере, для пользователя с именем **apache**, через **ACL**. Кроме того, доступ не будет разрешен если вы не установили логическое значение **httpd_enable_homedirs SELinux**. Вам также нужно будет настроить пользователя **michael** в базе данных аутентификации для этого каталога с помощью команды **htpasswd -c /etc/httpd/oneuser michael**.

Лабораторная работа 7

Процесс, необходимый для настройки каталога, управляемого группой, является гибридным. Общие основные шаги заключаются в следующем:

1. Создайте обычного пользователя и группу с именем **techsupport**. Хотя это и не требуется, это может быть полезно для настройки этот пользователь с более высоким **UID** и **GID**, чтобы не мешать другим будущим пользователям и группам.
2. Сделайте других пользователей членом этой группы с именем **techsupport**.
3. Создайте подкаталог **public_html/** домашнего каталога нового пользователя.
4. Установите соответствующие разрешения для поддержки доступа членами группы техподдержки обычно **2770** разрешений. Каталоги **home** и новый **public_html** техподдержки должны иметь либо обычные права на выполнение других пользователей, либо права на выполнение пользователем именованный **apache** настроен в **ACL**.
5. Настройте файл **index.html** в каталоге **public_html**. Это должно быть установлено с правами владения группы техподдержки.
6. Вам нужно будет настроить базовую аутентификацию для группы в **Apache**. Не забудьте настроить группу в базе данных аутентификации для этого каталога с помощью команды **htpasswd**.

Лабораторная работа 8

Указанный скрипт **hello.pl** должен содержать что-то вроде следующих записей:

```
#!/usr/bin/perl
print "Content-type: text/html\n\n";
print "Hello World";
```

Этот сценарий должен находиться в каталоге, указанном директивой **ScriptAlias /cgi-bin/** в контейнер виртуального хоста **big.example.com**.

В качестве альтернативной конфигурации вы можете создать новый блок **<Directory>** для каталога **CGI**. Контейнер должен содержать директивы **Options ExecCGI** и **AddHandler cgi-script .pl**. Вам может быть необходимо отключить директиву **ScriptAlias** по умолчанию в **httpd.conf**. Хотя обычно лучше иметь скрипты в каталоге, отличном от дерева **DirectoryRoot**, настроенного для виртуального хоста, тогда это не требуется.

Кроме того, права доступа к файлу **hello.pl** должны быть установлены на **755**, а контексты **SELinux** файлов (и каталогов) должен иметь тип файла **httpd_sys_script_exec_t**. Конечно, вы запустите соответствующую команду **semanage fcontext -a**, чтобы сделать изменение постоянным. В любом случае, успешный результат соответствует результатам, предложенным в лабораторном вопросе.