

Лабораторная работа № 5

Расширенная настройка HTTP-сервера Apache

Абд эль хай мохамад

Содержание

<i>Цель работы</i>	2
<i>Выполнение лабораторной работы</i>	2
1.Конфигурирование HTTP-сервера для работы через протокол HTTPS.....	2
2.Конфигурирование HTTP-сервера для работы с PHP.....	5
3.Внесение изменений в настройки внутреннего окружения виртуальной машины.....	6
Вывод:	6
Ответы на контрольные вопросы:	6

Цель работы

Приобретение практических навыков по расширенному конфигурированию HTTP-сервера Apache в части безопасности и возможности использования PHP

Выполнение лабораторной работы

1. Конфигурирование HTTP-сервера для работы через протокол HTTPS

Загружаю свою операционную систему и перехожу в рабочий каталог с проектом:
Запускаю виртуальную машину server: make server

На виртуальной машине server вхожу под своим пользователем и открываю терминал.
Перехожу в режим суперпользователя: sudo -i

Генерирую ключ для веб-сервера www.maabedelhay.net:

Сгенерированные ключи и сертификат появились в соответствующих подкаталогах в каталоге

Для перехода веб-сервера www.maabedelhay.net на функционирование через протокол HTTPS требуется изменить его конфигурационный файл. Перехожу в каталог с конфигурационными файлами: cd /etc/httpd/conf.d
Открываю на редактирование файл /etc/httpd/conf.d/www.maabedelhay.net.conf и заменяю его содержимое на следующее:

<VirtualHost *:80> Создать виртуальный хост на порту 80. Здесь мы используем звездочку вместо ip адреса, это значит, что веб-сервер будет слушать соединения на всех адресах, как на внешнем, так и на localhost

ServerAdmin webmaster@maabedelhay.net

Указывается электронный адрес администратора веб-сервера

DocumentRoot /var/www/html/www.maabedelhay.net

Указывается папка, в которой будут находиться данные сайта

ServerName www.maabedelhay.net Указывается домен

ServerAlias www.maabedelhay.net Указывается псевдоним домена

ErrorLog logs/www.maabedelhay.net-error_log

Путь к каталогу для сохранения лог-файлов ошибок

CustomLog logs/www.maabedelhay.net-access_log common

Путь к каталогу для сохранения лог-файлов доступа

RewriteEngine on Подключён модуль mod_rewrite

RewriteRule ^(.*)\$ https://%{HTTP_HOST}\$1 [R=301,L]

Перенаправляем запросы к любой странице на сайт https://%{HTTP_HOST}, используя перенаправление 301.

</VirtualHost> Описание виртуального хоста начинается с

<VirtualHost> и заканчивается </VirtualHost>

<IfModule mod_ssl.c> Весь код в этой секции будет выполнен

только в том случае, если активирован модуль
 mod_ssl. Это нужно для безопасности, чтобы если
 модуль не активирован, то код не вызывал ошибок

```
<VirtualHost *:443> Создать виртуальный хост на порту 443
SSLEngine on Подключена поддержка SSL
ServerAdmin webmaster@maabedelhay.net
  Указывается электронный адрес
  администратора веб-сервера
DocumentRoot /var/www/html/www.maabedelhay.net
  Указывается папка, в которой будут
  находиться данные сайта
ServerName www.maabedelhay.net Указывается домен
ServerAlias www.maabedelhay.net Указывается псевдоним домена
ErrorLog logs/www.maabedelhay.net-error_log
  Указывается путь к каталогу для
  сохранения лог-файлов ошибок

CustomLog logs/www.maabedelhay.net-access_log common
  Указывается путь к каталогу для
  сохранения лог-файлов доступа
SSLCertificateFile /etc/pki/tls/certs/www.maabedelhay.net.crt
  Указывается путь к файлам сертификата
SSLCertificateKeyFile /etc/pki/tls/private/www.maabedelhay.net.key
  Указывается путь к файлам приватного ключа
</VirtualHost> Конец описания виртуального хоста
</IfModule>
```

```

Activities  Terminal  Nov 26 16:26  en  [Icons]
root@server:/etc/httpd/conf.d
GNU nano 5.6.1  www.maabedelhay.net.conf  Modified
<VirtualHost *:80>
ServerAdmin webmaster@maabedelhay.net
DocumentRoot /var/www/html/www.maabedelhay.net
ServerName www.maabedelhay.net
ServerAlias www.maabedelhay.net
ErrorLog logs/www.maabedelhay.net-error_log
CustomLog logs/www.maabedelhay.net-access_log common
RewriteEngine on
RewriteRule ^(.*)$ https://%{HTTP_HOST}$1 [R=301,L]
</VirtualHost>

<IfModule mod_ssl.c>
<VirtualHost *:443>
SSLEngine on
ServerAdmin webmaster@maabedelhay.net
DocumentRoot /var/www/html/www.maabedelhay.net
ServerName www.maabedelhay.net
ServerAlias www.maabedelhay.net
ErrorLog logs/www.maabedelhay.net-error_log
CustomLog logs/www.maabedelhay.net-access_log common
SSLCertificateFile /etc/ssl/certs/www.maabedelhay.net.crt
SSLCertificateKeyFile /etc/ssl/private/www.maabedelhay.net.key
</VirtualHost>
</IfModule>

```

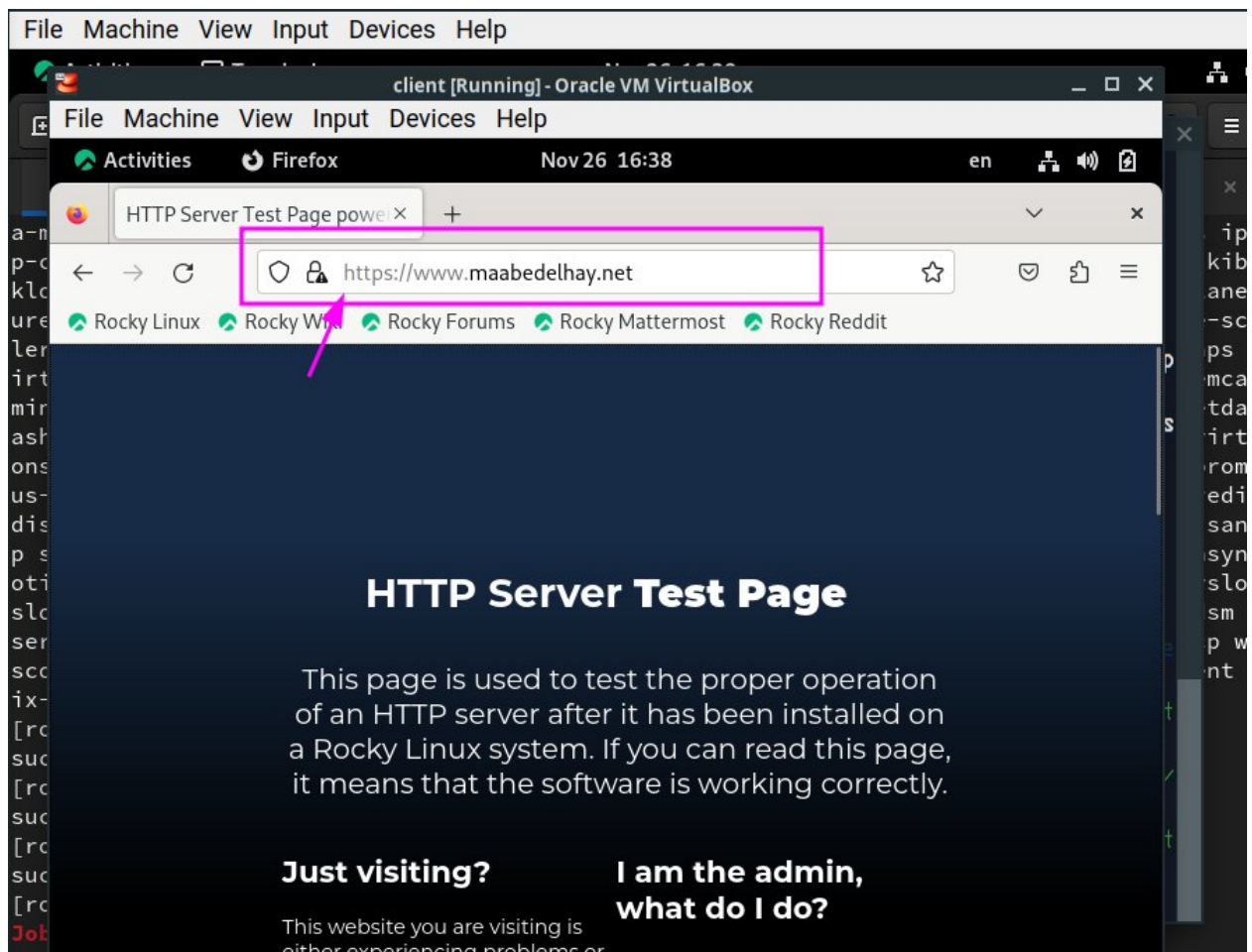
Вношу изменения в настройки межсетевого экрана на сервере, разрешив работу с https

Перезапускаю веб-сервер: `systemctl restart httpd`

На виртуальной машине client в строке браузера ввожу название веб-сервера `www.maabeldelhay.net`. Происходит автоматическое переключение на работу по протоколу HTTPS.

На открывшейся странице с сообщением о незащищённости соединения нажимаю кнопку «Дополнительно», затем добавляю адрес сервера в постоянные исключения.

Затем просматриваю содержание сертификата (нажмите на значок с замком в адресной строке и кнопку «Подробнее»).



2. Конфигурирование HTTP-сервера для работы с PHP

Устанавливаю пакеты для работы с PHP: `dnf -y install php`

В каталоге `/var/www/html/www.maabedelhay.net` заменяю файл `index.html` на `index.php` следующего содержания:

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

Корректирую права доступа в каталог с веб-контентом: `chown -R apache:apache /var/www`

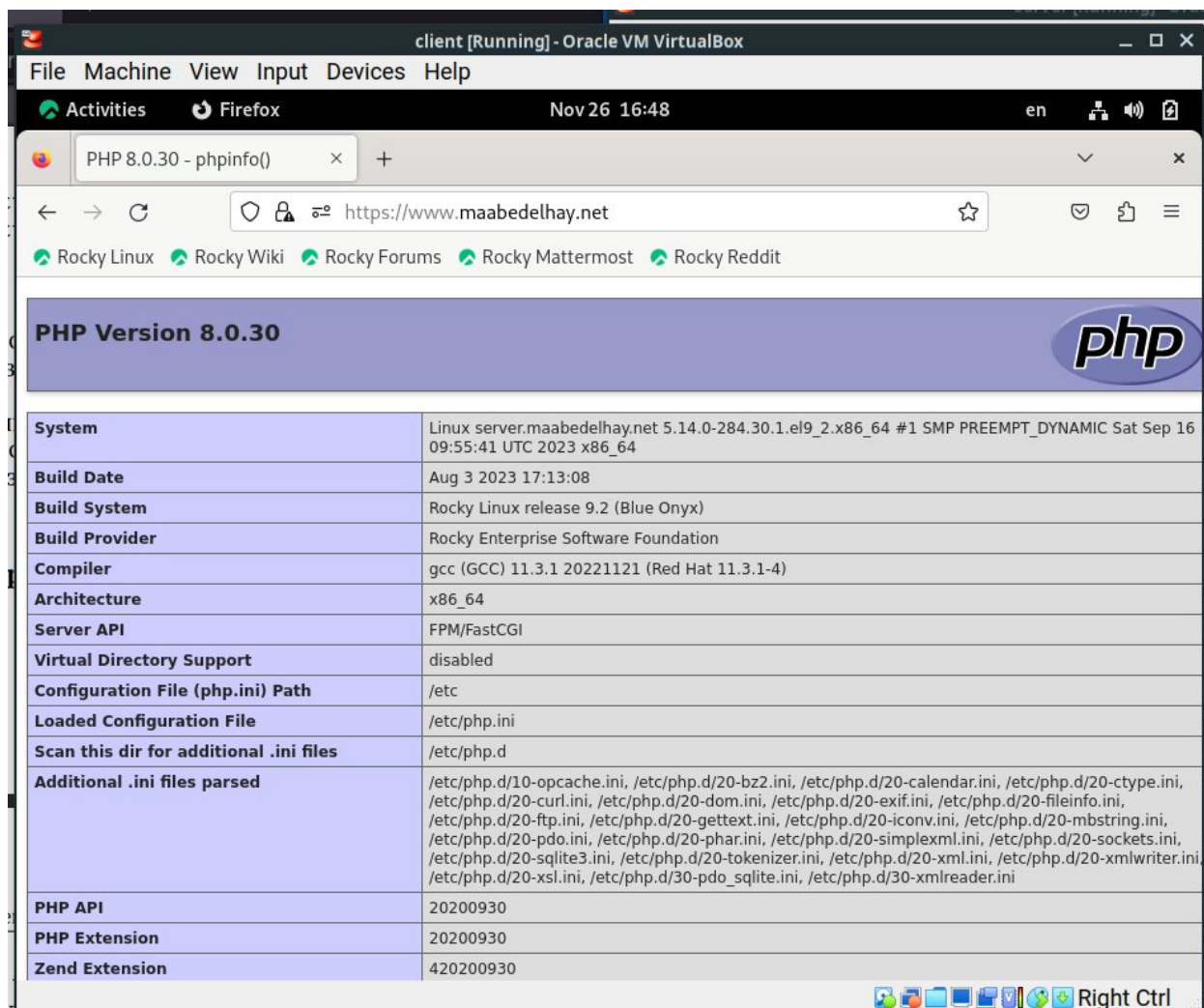
Восстанавливаю контекст безопасности в SELinux:

```
restorecon -vR /etc
```

```
restorecon -vR /var/www
```

Перезапускаю HTTP-сервер: `systemctl restart httpd`

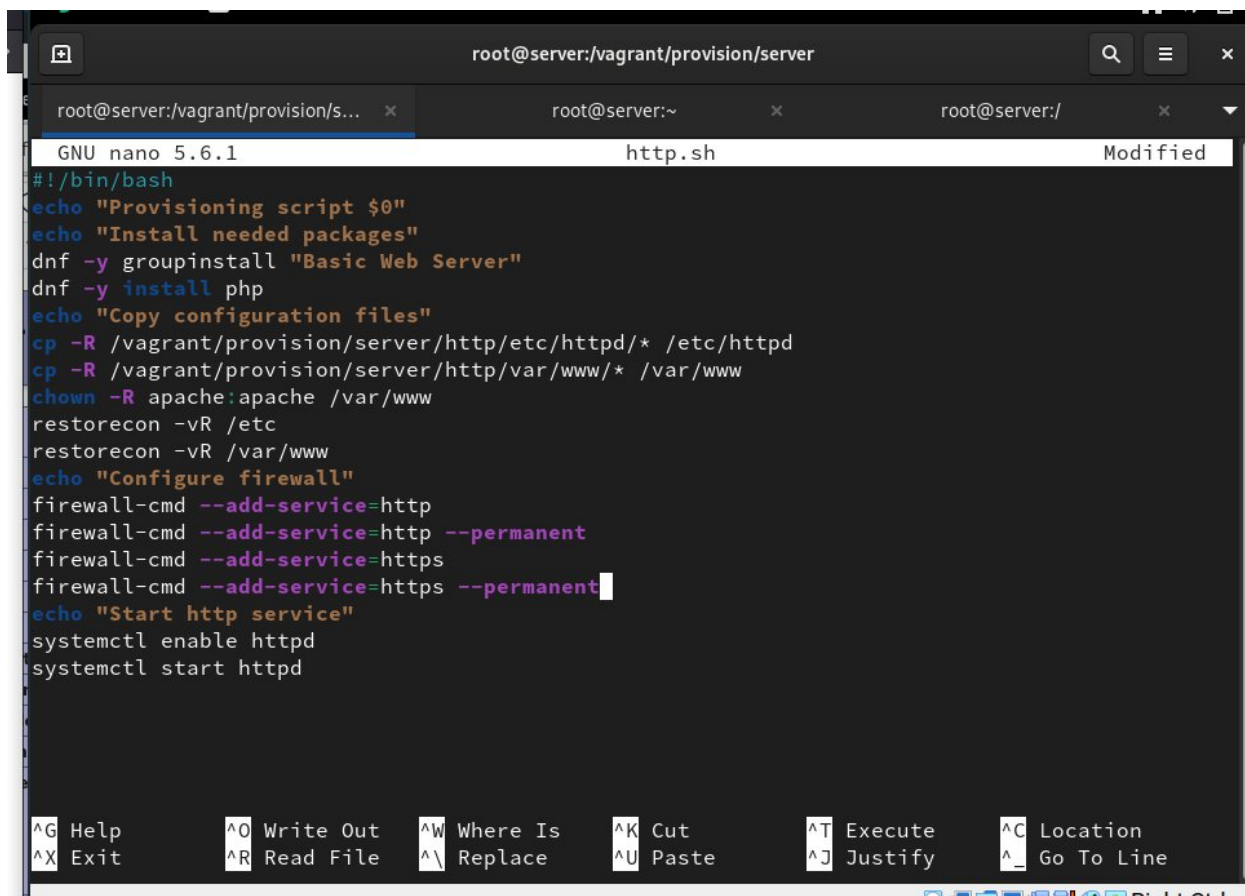
На виртуальной машине `client` в строке браузера ввожу название веб-сервера `www.maabedelhay.net`. Выведена страница с информацией об используемой на веб-сервере версии PHP.



3. Внесение изменений в настройки внутреннего окружения виртуальной машины

На виртуальной машине server перехожу в каталог для внесения изменений в настройки внутреннего окружения `/vagrant/provision/server/http` и в соответствующие каталоги копирую конфигурационные файлы:

В имеющийся скрипт `/vagrant/provision/server/http.sh` вношу изменения, добавив установку PHP и настройку межсетевого экрана, разрешающую работать с https.



```
root@server:/vagrant/provision/server
GNU nano 5.6.1 http.sh Modified
#!/bin/bash
echo "Provisioning script $0"
echo "Install needed packages"
dnf -y groupinstall "Basic Web Server"
dnf -y install php
echo "Copy configuration files"
cp -R /vagrant/provision/server/http/etc/httpd/* /etc/httpd
cp -R /vagrant/provision/server/http/var/www/* /var/www
chown -R apache:apache /var/www
restorecon -vR /etc
restorecon -vR /var/www
echo "Configure firewall"
firewall-cmd --add-service=http
firewall-cmd --add-service=http --permanent
firewall-cmd --add-service=https
firewall-cmd --add-service=https --permanent
echo "Start http service"
systemctl enable httpd
systemctl start httpd

^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location
^X Exit      ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^_ Go To Line
```

Вывод:

Я приобрел практические навыки по расширенному конфигурированию HTTP-сервера Apache в части безопасности и возможности использования РНР.

Ответы на контрольные вопросы:

1. В чём отличие HTTP от HTTPS?

HTTP — прикладной протокол передачи данных, используемый для получения информации с веб-сайтов.

HTTPS — расширение протокола HTTP, поддерживающее шифрование по протоколам SSL и TLS.

HTTPS — не самостоятельный протокол передачи данных, а HTTP с надстройкой шифрования. В этом ключевое и единственное отличие. Если по протоколу HTTP данные передаются незащищенными, то HTTPS обеспечит криптографическую защиту.

2. Каким образом обеспечивается безопасность контента веб-сервера при работе через HTTPS?

Для шифрования может применяться протокол **SSL** (Secure Sockets Layer) или протокол **TLS** (Transport Layer Security). Оба протокола используют асимметричное шифрование для аутентификации, симметричное шифрование для конфиденциальности и коды аутентичности сообщений для сохранения целостности сообщений.

Симметричное шифрование — способ шифрования, в котором для шифрования и

дешифровывания данных применяется один и тот же криптографический ключ.

Ассимметричное шифрование — способ шифрования, в котором для шифрования и дешифровывания данных применяется пара ключей—открытый и закрытый.

Открытый ключ известен, передаётся по открытому каналу и используется для аутентификации пользователей и собственно для шифрования передаваемых данных.

Закрытый ключ должен быть сохранён в тайне и находиться на стороне получателя шифрованного сообщения. При помощи закрытого ключа сообщение дешифруется и таким образом подтверждается подлинность отправителя сообщения.

Криптографический ключ — секретная информация, используемая криптографическим алгоритмом при шифровании/дешифровании данных.

Основной характеристикой криптостойкости криптографического ключа является его длина, измеряемая как правило в битах. Для симметричных алгоритмов шифрования рекомендуемая минимальная длина ключа — 128 бит, для ассиметричных алгоритмов — 1024 бит.

Сертификат открытого ключа— документ (электронный или бумажный), содержащий как сам открытый ключ, так и информацию о его владельце и области применения. Сертификат подписывается выдавшим его сертификационным центром, который подтверждает принадлежность открытого ключа владельцу

3. Что такое сертификационный центр? Приведите пример

Сертификационный центр (Certification authority, CA) представляет собой компонент глобальной службы каталогов, отвечающий за управление криптографическими ключами пользователей. Его открытый ключ широко известен общественности и не вызывает сомнений в подлинности.

Задача центра сертификации — подтверждать подлинность ключей шифрования с помощью сертификатов электронной подписи.

Совет безопасности центра сертификации (CASC) включает в себя крупнейшие центры сертификации, такие как Comodo, DigiCert, Entrust, GlobalSign, GoDaddy, Symantec и Trend Micro.