Записки IT специалиста

Настраиваем веб-сервер Caddy + PHP + MySQL с сертификатами Let's Encrypt

Веб-сервер Caddy сравнительно молодой и новый игрок на рынке веб-серверов, первый выпуск был опубликован в 2015 году, а используемая сейчас вторая версия вышла в 2020 году. Caddy написан на Go и активно развивается, в его основе лежат современные подходы к реализации веб-серверов, а именно простота и защищенность. В отличие от других веб-серверов Caddy из коробки рассчитан на работу с HTTPS и имеет интеграцию с Let's Encrypt, позволяя автоматически получать и продлять сертификаты, а также он весьма прост в настройке, в чем мы скоро убедимся.

Итак, зачем вам нужено именно Caddy и что такого он дает по сравнению с Apache или Nginx? Как мы уже сказали выше - простоту и безопасность. Вам не нужно настраивать параметры шифрования и использования протоколов, Caddy все сделает из коробки, причем наилучшим образом, с использованием самых современных технологий. Да и сама конфигурация предельно проста, вам нужно задать минимум опций для получения рабочего сервера, но при этом вы можете достаточно гибко управлять им, переопределяя необходимые параметры.

К недостаткам можно отнести недостаточное количество материалов на русском языке и плохое знание синтаксиса Caddy со стороны нейросетей, которые довольно популярны в последнее время. Но если вы имеете опыт работы с другими веб-серверами и владеете английским со словарем, то официальной документации вполне достаточно, тем более что она изобилует примерами.

Для установки мы будем использовать последние версии популярных систем Debian 12 и Ubuntu 24.04 LTS, принципиального отличия они не имеют и содержат актуальные версии, что позволит развернуть веб-сервер штатными средствами без подключения дополнительных репозиториев. Набор используемого ПО таков:

- **Debian 12**: Caddy 2.6, PHP 8.2, MariaDB 10.11 LTS

- **Ubuntu 24.04 LTS**: Caddy 2.6, PHP 8.3, MySQL 8.0 LTS

Все команды, если не указано иного, следует выполнять с правами суперпользователя root или через sudo.

В качестве примера имени сайта мы будем использовать:

- **site.example**
- **www.site.example**

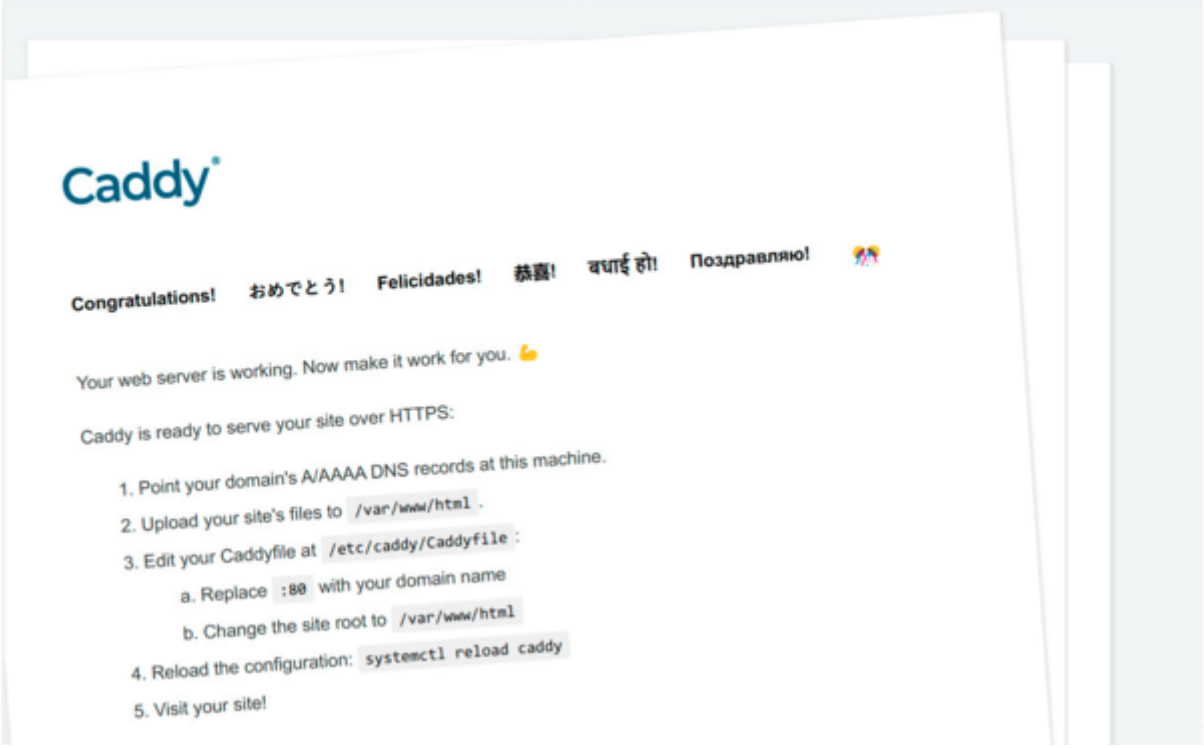
Их следует заменить на реальное доменное имя, которое вы будете использовать.

Установка и базовая настройка веб-сервера Caddy

Установка сервера производится одной простой командой:

```
apt install caddy
```

Теперь если набрать в адресной строке браузера адрес веб-сервера, то вы увидите стандартную страницу-заглушку, это обозначает, что веб-сервер установлен и работает.



Кстати, данная страница содержит краткую инструкцию по запуску веб-сервера, это все, что достаточно выполнить для получения работающего и безопасного современного сайта. Просто? Очень просто. Но мы выполним более тонкую настройку.

Внутренний формат конфигурации веб-сервера хранится в формате JSON и ей можно управлять онлайн через REST API, также доступен и более классический формат настройки через файл конфигурации, для этого используется конфигурационный файл **/etc/caddy/Caddyfile**. Он уже содержит пример конфигурации и нам следует его только откорректировать. Обратите внимание, что отступы в файле формируются сугубо при помощи табуляции и следует использовать два -

четыре - шесть и т.д. отступов, в противном случае вы получите предупреждение о некорректном форматировании конфигурационного файла.

Итак, откроем на редактирование конфигурационный файл и заменим в нем **:80** на адрес нашего сайта:

```
site.example {
```

Подобная запись обозначает, что для сайта будет автоматически получен сертификат Let's Encrypt при использовании существующего домена или самоподписанный для несуществующих доменов. Теперь укажем путь к корневой директории сайта:

```
root * /var/www/site.example
```

Следующий параметр включает для виртуального хоста отдачу статики:

```
file_server
```

В целом этого уже достаточно для работы, но мы добавим еще несколько опций, прежде всего включим сжатие, причем предлагать будем не только классический **zip**, но и современный **zstd**, форматы перечислены в порядке предпочтения и согласовываются с клиентом.

```
encode zstd gzip
```

Также укажем параметры логирования:

```
log {
     output file /var/log/caddy/site.example.log
     format console
 }
```

Затем ниже последней закрывающей фигурной скобки добавим еще один виртуальный хост для сайта с **www**, который мы будем переправлять на основной сайт:

```
www.site.example {
     redir https://site.example{uri}
}
```

Перезапускаем веб-сервер командой:

```
systemctl reload caddy
```

Теперь создадим указанные в конфигурации папки:

```
mkdir -p mkdir /var/www/site.example
```

И сделаем их владельцем веб-сервер:

```
chown -R www-data:www-data /var/www/
```

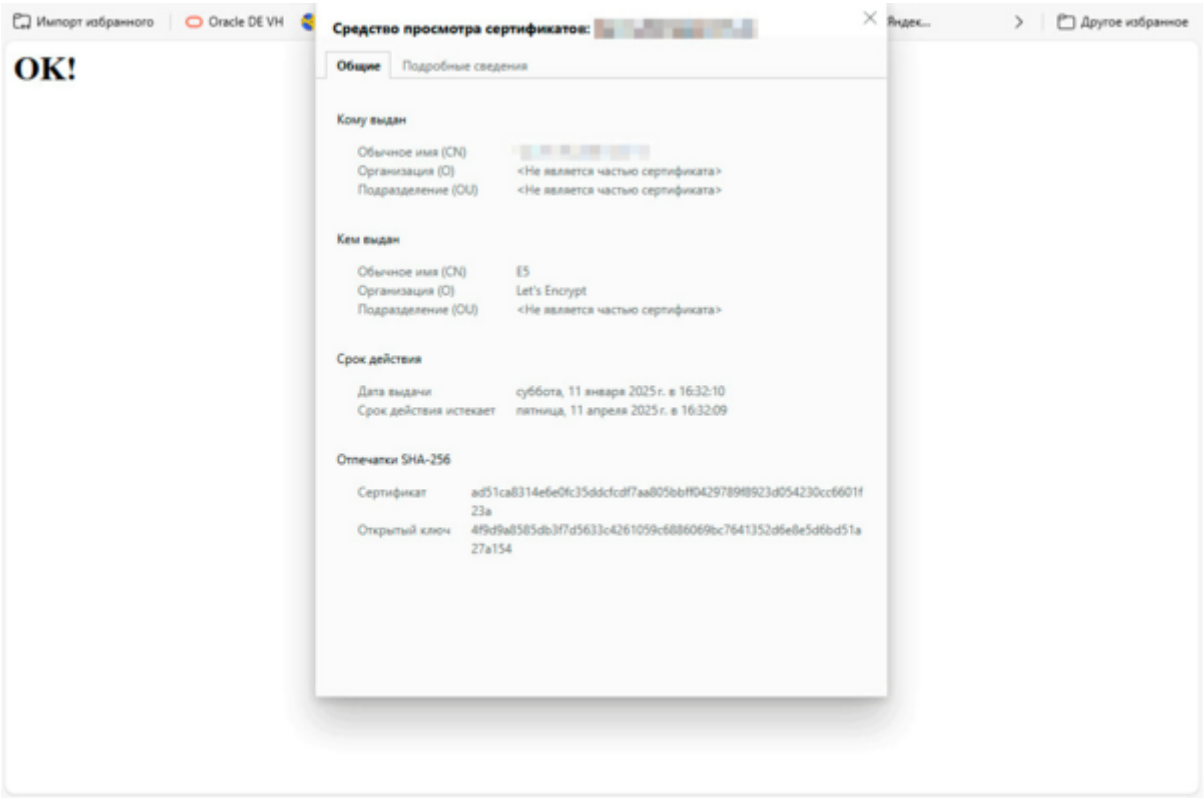
В корневую директорию сайта добавим индексный файл:

```
nano /var/www/site.example/index.html
```

И внесем в него следующее содержимое:

```
<body><h1>OK!</h1></body>
```

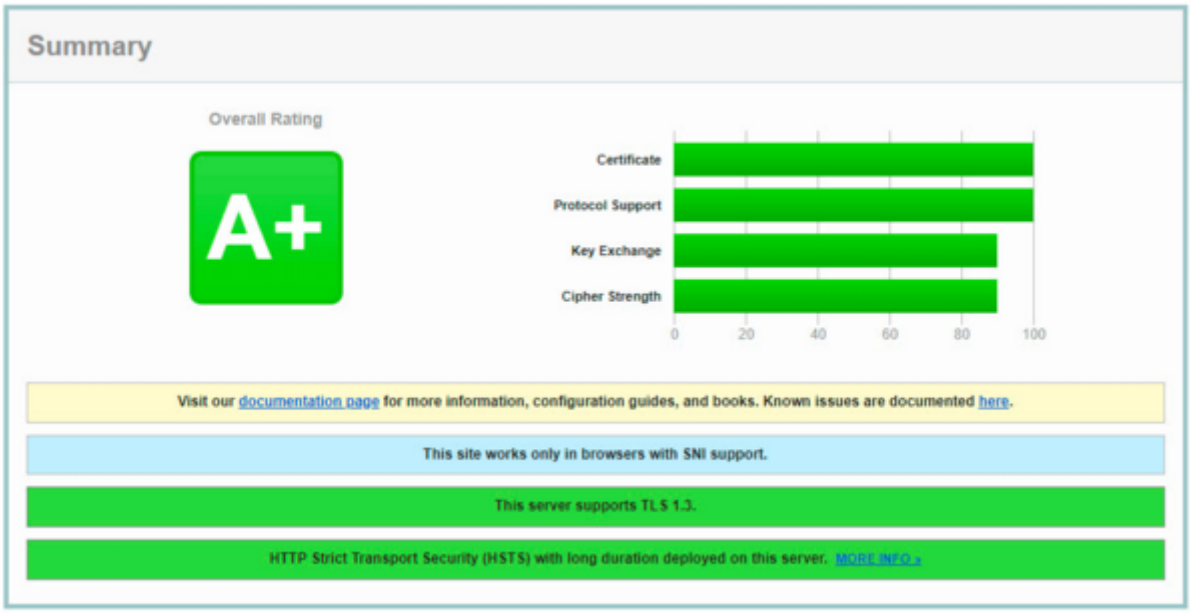
Теперь заходим на наш сайт, мы будем автоматически направлены на HTTPS версию и можем убедиться, что сайт получил сертификат от Let's Encrypt, а также использует (при поддержке со стороны браузера) TLS 1.3 и HTTP/3.



На этом настройка веб-сервера Caddy закончена, если проверить его на **ssllabs.com**, то получим рейтинг A. Если вы хотите получить рейтинг A+, то следует добавить страницам сайта заголовок, который включит HSTS (HTTP Strict Transport Security) - механизм принудительно включающий соединение с сайтом по HTTPS, если ранее такое соединение уже было установлено. Технически это не обязательно, так как Caddy автоматически пересылает все HTTP запросы на HTTPS версию сайта, но если вы хотите рейтинг A+ (скажем, сдаете сайт заказчику), то в секцию виртуального хоста добавьте:

```
header {
      Strict-Transport-Security max-age=31536000;
 }
```

После чего перезапустите веб-сервер.



Просто?

Просто и быстро. В этом основное преимущество Caddy, вам не нужны километры строк в конфигурации, все современные технологии вы получаете из коробки. Единственный минус - совместимость со старыми системами, так как Caddy автоматически отключает устаревшие протоколы и шифры.

Установка и настройка PHP-FPM

Для работы с динамическим содержимым нам потребуется поддержка PHP, скриптового языка, на котором написано большинство современных систем управления контентом (CMS), также называемых движками сайтов. Caddy не имеет собственного менеджера процессов, поэтому мы будем использовать для этой цели PHP-FPM. Установим его:

```
apt install php-fpm
```

Затем перейдем в **/etc/php/8.3/fpm/php.ini** (обратите внимание, что в пути следует указать версию установленного PHP) и откорректируем некоторые параметры. Прежде всего найдем, раскомментируем и приведем к следующему виду опцию:

```
cgi.fix_pathinfo=0
```

Затем установим максимальный размер отправляемого запроса:

```
post_max_size = 32M
```

И максимальный размер загружаемого файла, он должен быть меньше или равен размеру отправляемого запроса:

```
upload_max_filesize = 30M
```

Сохраняем изменения и перезапускаем службу (также на забывайте про номер версии):

```
systemctl restart php8.3-fpm
```

Для работы с PHP добавим виртуальному хосту в **Caddyfile** следующие опции:

```
php_fastcgi unix//run/php/php-fpm.sock
```

Сохраняем изменения и перезапускаем веб-сервер:

```
systemctl reload caddy
```

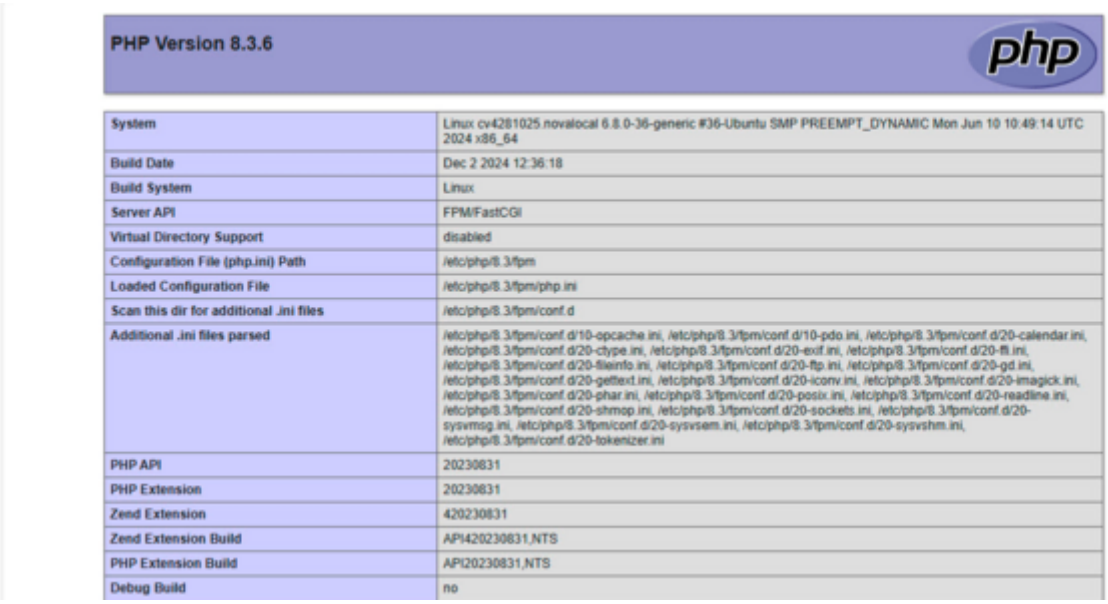
Для проверки работы PHP создадим специальный файл:

```
nano /var/www/site.examlpe/info.php
```

И внесем в него следующий текст:

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

Если теперь в адресной строке браузера набрать **site.example/info.php** то увидим стандартную страницу с информацией о PHP, его настройках и установленных модулях.



PHP Version 8.3.6

System	Linux cv4281025.novalocal 6.8.0-36-generic #36-Ubuntu SMP PREEMPT_DYNAMIC Mon Jun 10 10:49:14 UTC 2024 x86_64
Build Date	Dec 2 2024 12:36:18
Build System	Linux
Server API	FPM/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/8.3/fpm
Loaded Configuration File	/etc/php/8.3/fpm/php.ini
Scan this dir for additional .ini files	/etc/php/8.3/fpm/conf.d
Additional .ini files parsed	/etc/php/8.3/fpm/conf.d/10-opcache.ini, /etc/php/8.3/fpm/conf.d/10-pdo.ini, /etc/php/8.3/fpm/conf.d/20-calendar.ini, /etc/php/8.3/fpm/conf.d/20-ctype.ini, /etc/php/8.3/fpm/conf.d/20-exif.ini, /etc/php/8.3/fpm/conf.d/20-ffi.ini, /etc/php/8.3/fpm/conf.d/20-fileinfo.ini, /etc/php/8.3/fpm/conf.d/20-ftp.ini, /etc/php/8.3/fpm/conf.d/20-gd.ini, /etc/php/8.3/fpm/conf.d/20-gettext.ini, /etc/php/8.3/fpm/conf.d/20-iconv.ini, /etc/php/8.3/fpm/conf.d/20-imagick.ini, /etc/php/8.3/fpm/conf.d/20-phar.ini, /etc/php/8.3/fpm/conf.d/20-posix.ini, /etc/php/8.3/fpm/conf.d/20-readline.ini, /etc/php/8.3/fpm/conf.d/20-shmop.ini, /etc/php/8.3/fpm/conf.d/20-sockets.ini, /etc/php/8.3/fpm/conf.d/20-sysvmsg.ini, /etc/php/8.3/fpm/conf.d/20-sysvsem.ini, /etc/php/8.3/fpm/conf.d/20-sysvshm.ini, /etc/php/8.3/fpm/conf.d/20-tokenizer.ini
PHP API	20230831
PHP Extension	20230831
Zend Extension	420230831
Zend Extension Build	API420230831,NTS
PHP Extension Build	API20230831,NTS
Debug Build	no

Модули предназначены для расширения возможностей PHP и их необходимый набор следует уточнять в требованиях вашего веб-приложения, мы же установим набор самых часто используемых и необходимых:

```
apt install php-mbstring php-zip php-gd php-json php-curl php-imagick php-mysql php-intl
```

На этом установка и настройка PHP-FPM закончена.

Установка MySQL / MariaDB

Сервер баз данных третий неотъемлемый компонент классического веб-сервера и один из первых по важности, так как именно в базе данных хранится основная пользовательская информация сайта. В современных версиях Debian и Ubuntu используются разные варианты СУБД: MariaDB и MySQL, которые являются совместимыми и принципиально разницы какую из них использовать нет, тем более что в репозиториях содержатся последние LTS-версии обоих СУБД. Поэтому будем устанавливать версии из репозитория, что упростит дальнейшую поддержку и обслуживание сервера.

Для **Debian** это будет команда:

```
apt install mariadb-server
```

А для **Ubuntu**:

```
apt install mysql-server
```

В дальнейшем в обоих системах для управления можно использовать один и тот же набор команд:

```
systemctl start | stop | restart | status mysql
```

Вне зависимости от того, какую именно версию СУБД вы используете.

После установки сервера баз данных следует выполнить скрипт, который выполнит рекомендуемые настройки безопасности установленного экземпляра сервера:

```
mysql_secure_installation
```

В нем положительно отвечаем на все вопросы, кроме смены пароля суперпользователя СУБД root, в текущем режиме установки учетная запись root не имеет пароля и поддерживает вход исключительно через UNIX-сокет, поэтому не следует ее включать.

Теперь выполним некоторые настройки самого сервера СУБД, для этого перейдем в командную строку MySQL в режиме суперпользователя:

```
mysql -u root
```

В первую очередь создадим пользователей СУБД и разрешим им вход по паролю:

```
CREATE USER 'andrey'@'localhost' IDENTIFIED BY 'Pa$word_1';
```

После чего отберем у него права на чужие базы:

```
GRANT USAGE ON *.* TO 'andrey'@'localhost';
```

А затем выдадим полные права на все базы с шаблоном имени **andrey_<имя базы>**, что позволит автоматически устанавливать права просто создав базу с нужным именем.

```
GRANT ALL PRIVILEGES ON `andrey\_%`.* TO 'andrey'@'localhost';
```

Обратите внимание, что шаблон оборачивается символами **грависа (`)**, который находится на клавише с русской буквой Ё.

Перезагрузим привилегии:

```
FLUSH PRIVILEGES;
```

И выйдем из консоли MySQL:

```
QUIT;
```

На этом настройка сервера СУБД закончена.

Установка phpMyAdmin

phpMyAdmin - удобное веб-приложение для управления СУБД MySQL / MariaDB и, по сути, стандарт де-факто среди таких приложений, его использование для повседневной работы с СУБД повышает общее удобство, по сравнению с консолью и поэтому мы не видим причин отказывать от его использования.

Для его установки используйте команду:

```
apt install phpmyadmin
```

После чего добавьте виртуальному хосту в **Caddyfile** следующие настройки:

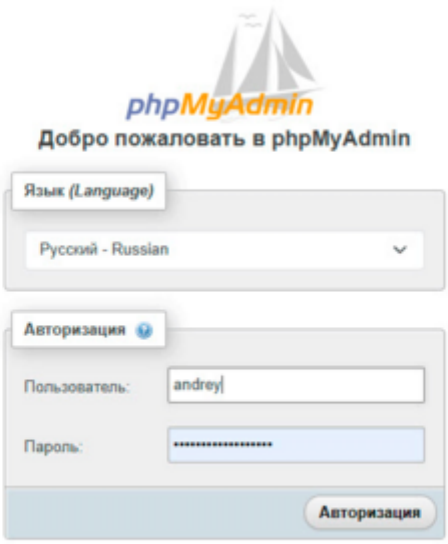
```
root /phpmyadmin/* /usr/share/phpmyadmin

redir /phpmyadmin /phpmyadmin/ permanent
uri /phpmyadmin/* strip_prefix /phpmyadmin
```

Которые обеспечат правильное направление всех запросов с адресом **/phpmyadmin** к нужной корневой директории веб-приложения.

Перезапускаем веб-сервер:

```
systemctl reload caddy
```

И переходим по адресу **site.example/phpmyadmin**:



На этом настройка веб-сервера на базе Caddy + PHP + MySQL с сертификатами Let's Encrypt закончена.

Онлайн-курс по устройству компьютерных сетей

На углубленном курсе "Архитектура современных компьютерных сетей" вы с нуля научитесь работать с Wireshark и «под микроскопом» изучите работу сетевых протоколов. На протяжении курса надо будет выполнить более пятидесяти лабораторных работ в Wireshark.

Помогла статья? Поддержи автора и новые статьи будут выходить чаще:
