Знакомство с веб-сервером

Веб-сервер — это программа, которая позволяет получать и обрабатывать запросы от клиента по протоколу HTTP. Например, он поможет нам опубликовать как сайт наш статический HTML-документ.

Современные веб-сервисы гораздо сложнее. Помимо веб-сервера, в них обычно есть один или несколько бэкендов, приложений, написанных на различных языках программирования, которые осуществляют различную сложную логику, взаимодействуют с базами данных и другими бэкендами для обработки пользовательских запросов.

Среди распространённых в наше время веб-серверов стоит упомянуть три:

```
Nginx
Apache
IIS
```

IIS — это проприетарный веб-сервер с закрытым исходным кодом, разработка Microsoft.
Nginx и **Apache** — веб-серверы с открытым исходным кодом. Они могут быть запущены и под Linux, и на других операционных системах.
Apache — это более старый и чуть более распространённый проект. Несмотря на набор проблем с производительностью и надёжностью в прошлом, проект активно развивается и не сдаёт свои позиции в топе популярных веб-серверов.
Nginx — более легковесный, чем Apache, и обеспечивает большую скорость обработки запросов.

В нашем курсе мы будем рассматривать именно Nginx.

Установка веб-сервера

```
sudo apt-get install nginx
```

Веб-сервер работает в фоне, то есть его процесс не должен держать терминал и влиять на наше взаимодействие с прочими сервисами. Для его запуска давайте скажем:

```
sudo systemctl start nginx
```

Проверим, что веб-сервер успешно запустился:

```
ps auxf | grep nginx
```

Даже до того, как настроим сервер на публикацию первого HTML-документа, мы можем увидеть его работу.
Loopback-адреса — специальная подсеть, доступ к которой можно получить исключительно внутри нашей операционной системы. Этот адрес поможет увидеть работу веб-сервера.

Чтобы открыть стартовую страницу, запустим Links и зайдём по IP-адресу на наш веб-сервер.
Для проверки работы остановим веб-сервер: systemctl stop nginx.

Конфигурация веб-сервера

Основной файл конфигурации — /etc/nginx.conf.

В нём можно увидеть вполне явную структуру:

- Общие настройки. Самая важная — user. Веб-сервер никогда не должен запускаться с привилегиями пользователя root. В случае же непривилегированного пользователя, такого как www-data, его доступ к системе будет ограничен.
- Настройка количества дочерних процессов веб-сервера. Эта настройка важна для высоконагруженных инсталляций, где трафик может достигать миллионов запросов в секунду. Сейчас нам вполне подойдёт автоматическая настройка.
- Настройка PID-файла, в котором хранится идентификатор процесса.
- Папка с модулями для веб-сервера. Они полезны, когда нам нужно включить какие-то дополнительные возможности веб-сервера.
- В секции HTTP перечислены основные настройки веб-сервера.

Один веб-сервер может обслуживать несколько веб-сайтов либо перенаправлять запросы в другие сервисы (то есть выступать в роли прокси-сервера). Поэтому для удобства управления конфигурацией лучше не использовать для настройки конкретных сайтов файл nginx.conf, а оставить его для общих для веб-сервера настроек.

Сайты описываются в директории /etc/nginx/sites-enabled.

```
sudo vim /etc/nginx/sites/enabled/default
```

Как мы видим, здесь уже есть один файл, именно благодаря ему мы увидели приветственную страницу, когда запустили сервер. Сайты описываются в секции Server, она является вложенной для секции HTTP.

Директива root означает директорию, где расположены файлы с HTML-документами. Иногда удобно использовать такой подход, иногда — размещать такие директории в домашних каталогах пользователей.

Индексные файлы — это файлы, которые по умолчанию обрабатываются при обращении к директории веб-сервером.
Давайте посмотрим, какое из этих имён использовано в примере.

Посмотрим содержимое папки root.
Далее мы видим описание location /.
В нём написано, что при вызове / мы будем перебирать файлы в директории root и в случае отсутствия файла с нужным именем — выдавать ошибку 404.

Давайте научим наш веб-сервер выводить не файл по умолчанию, а наш HTML-документ. Для этого добавим в location строку:

```
index webpage.html
```

И сохраним файл.

Теперь скопируем файл в root-директорию нашего веб-сервера.

```
sudo cp webpage.html /var/www/html
```

Для того чтобы наши изменения вступили в силу, необходимо попросить веб-сервер перечитать содержимое конфигурационных файлов. Для этого есть две возможности — restart и reload.

restart остановит веб-сервер и запустит его заново.
reload — более мягкий вариант. Например, он не будет рвать уже работающие соединения, что важно для сайта под нагрузкой.

Перезапустим веб-сервер:

```
sudo systemctl nginx reload
```

Снова зайдём на 127.0.0.1 и увидим наш текст.

Мы установили веб-сервер и настроили его на отображение нашего HTML-документа.