Установка Nginx в Ubuntu 22.04

Введение

Nginx (engine x — по-русски произносится как энджи́нкс или э́нжин-и́кс — веб-сервер и почтовый прокси-сервер, работающий на Unix-подобных операционных системах (тестировалась сборка и работа на FreeBSD, OpenBSD, Linux, Solaris, macOS, AIX и HP-UX). Начиная с версии 0.7.52 появилась экспериментальная бинарная сборка под Microsoft Windows.

Российский программист Игорь Сысоев начал разработку в 2002 году. Осенью 2004 года вышел первый публично доступный релиз. С июля 2011 работа над nginx продолжается в рамках компании Nginx.

В этом материале вы узнаете, как установить Nginx в системе Ubuntu 22.04 LTS. Это руководство также совместимо с системами Ubuntu 20.04 LTS и Ubuntu 18.04 LTS.

Предварительные условия

Сначала войдите в Ubuntu 22.04 через консоль. Затем обновите кэш Apt и обновите текущие пакеты системы с помощью следующей команды:

```
sudo apt update && sudo apt upgrade
```

При появлении запроса нажмите «Y» , чтобы подтвердить установку.

Установите Nginx в Ubuntu 22.04

После окончания процесса обновления пакетов можно установить Nginx на машину:

```
sudo apt install nginx
```

```
vladpolskiy@vp-vmm:~$ sudo apt install nginx
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libnginx-mod-http-geoip2 libnginx-mod-http-image-filter
  libnginx-mod-http-xslt-filter libnginx-mod-mail libnginx-mod-stream
  libnginx-mod-stream-geoip2 nginx-common nginx-core
Suggested packages:
  fcgiwrap nginx-doc
The following NEW packages will be installed:
  libnginx-mod-http-geoip2 libnginx-mod-http-image-filter
  libnginx-mod-http-xslt-filter libnginx-mod-mail libnginx-mod-stream
  libnginx-mod-stream-geoip2 nginx nginx-common nginx-core
0 upgraded, 9 newly installed, 0 to remove and 0 not upgraded.
Need to get 697 kB of archives.
After this operation, 2□395 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
```

Дождемся окончания установки, а после добавим программу в автозагрузку:

```
sudo systemctl enable nginx
```

```
vladpolskiy@vp-vmm:~$ sudo systemctl enable nginx
Synchronizing state of nginx.service with SysV service script with /lib/systemd/
systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable nginx
vladpolskiy@vp-vmm:~$
```

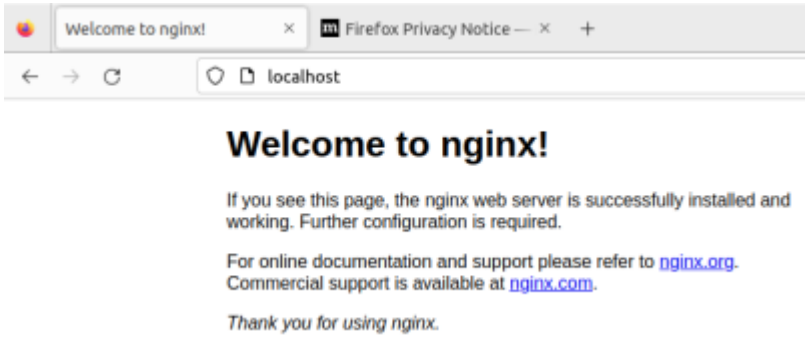
Теперь нужно проверить, что веб-сервер успешно установлен и работает, а также добавлен в автозагрузку. Проверим статус работы веб-сервера:

```
sudo service nginx status
```

```
vladpolskiy@vp-vmm:~$ sudo service nginx status
● nginx.service - A high performance web server and a reverse proxy server
     Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset:
     Active: active (running) since Sun 2024-04-07 22:14:01 MSK; 6min ago
       Docs: man:nginx(8)
   Main PID: 3793 (nginx)
      Tasks: 9 (limit: 4597)
     Memory: 8.4M
        CPU: 44ms
     CGroup: /system.slice/nginx.service
             ├─3793 "nginx: master process /usr/sbin/nginx -g daemon on; master
             ├─3795 "nginx: worker process" "" "" "" "" "" "" "" "" "" "" "" ""
             ├─3796 "nginx: worker process" "" "" "" "" "" "" "" "" "" "" "" ""
             ├─3797 "nginx: worker process" "" "" "" "" "" "" "" "" "" "" "" ""
             ├─3798 "nginx: worker process" "" "" "" "" "" "" "" "" "" "" "" ""
             ├─3799 "nginx: worker process" "" "" "" "" "" "" "" "" "" "" "" ""
             ├─3800 "nginx: worker process" "" "" "" "" "" "" "" "" "" "" "" ""
             ├─3801 "nginx: worker process" "" "" "" "" "" "" "" "" "" "" "" ""
             └─3802 "nginx: worker process" "" "" "" "" "" "" "" "" "" "" "" ""

апр 07 22:14:01 vp-vmm systemd[1]: Starting A high performance web server and a
апр 07 22:14:01 vp-vmm systemd[1]: Started A high performance web server and a
vladpolskiy@vp-vmm:~$
```

Строка «Active: active (running)…» указывает на успешную работу сервера. Есть и другой способ проверить его работу. Нужно вставить IP-адрес сервера в адресную строку браузера. Если результат будет таким же, как на картинке ниже, то веб-сервер работает успешно.



Теперь проверим его наличие в автозагрузке:

```
sudo systemctl is-enabled nginx
```

В результате система выдаст следующее сообщение:enabled

```
vladpolskiy@vp-vmm:~$ sudo systemctl is-enabled nginx
enabled
vladpolskiy@vp-vmm:~$
```

Если в ответ на выполненную команду получаем «enabled», значит веб-сервер добавлен в автозагрузки.

Базовые команды управления

Для работы с установленным веб-сервером пригодятся базовые команды управления. Они приведены в таблице ниже.

Функция	Команда
Запуск	sudo systemctl start nginx
Отключение	sudo systemctl stop nginx
Перезапуск	sudo systemctl restart nginx
Перезагрузка	sudo systemctl reload nginx
Проверка состояния службы	sudo systemctl status nginx
Тестирование конфигурации	sudo nginx -t

Настройка брандмауэра

Установка и настройка брандмауэра позволит закрыть все порты, кроме необходимых нам — 22 (SSH), 80 (HTTP), 443 (HTTPS). Первый протокол необходим для подключения к удаленному серверу. Второй и третий необходим для связи между клиентом и сайтом. Главное их отличие в том, что HTTPS — это зашифрованный HTTP. Шифрование данных происходит благодаря SSL-сертификату.

Установим утилиту UFW:

```
sudo apt install ufw
```

После успешной установки добавим веб-сервер в список доступных приложений брандмауэра:

```
sudo nano /etc/ufw/applications.d/nginx.ini
```

Заполним файл следующим образом:

```
[Nginx HTTP]
title=Web Server
description=Enable NGINX HTTP traffic
ports=80/tcp

[Nginx HTTPS] \
title=Web Server (HTTPS) \
description=Enable NGINX HTTPS traffic
ports=443/tcp

[Nginx Full]
title=Web Server (HTTP,HTTPS)
description=Enable NGINX HTTP and HTTPS traffic
ports=80,443/tcp
```

Проверим список доступных приложений:

```
sudo ufw app list
```

Если среди них есть веб-сервер, значит всё сделано верно. Теперь нужно запустить брандмауэр и разрешить передачу трафика по вышеуказанным портам:

```
sudo ufw enable
sudo ufw allow 'Nginx Full'
sudo ufw allow 'OpenSSH'
```

```
vladpolskiy@vp-vmm:~$ sudo ufw allow 'Nginx Full'
sudo ufw allow 'OpenSSH'
Rules updated
Rules updated (v6)
Rules updated
Rules updated (v6)
vladpolskiy@vp-vmm:~$
```

Чтобы проверить изменения, вводим команду:

```
sudo ufw status
```

Если всё сделано правильно, то в статусе будут перечислены все порты, которые нам необходимы.

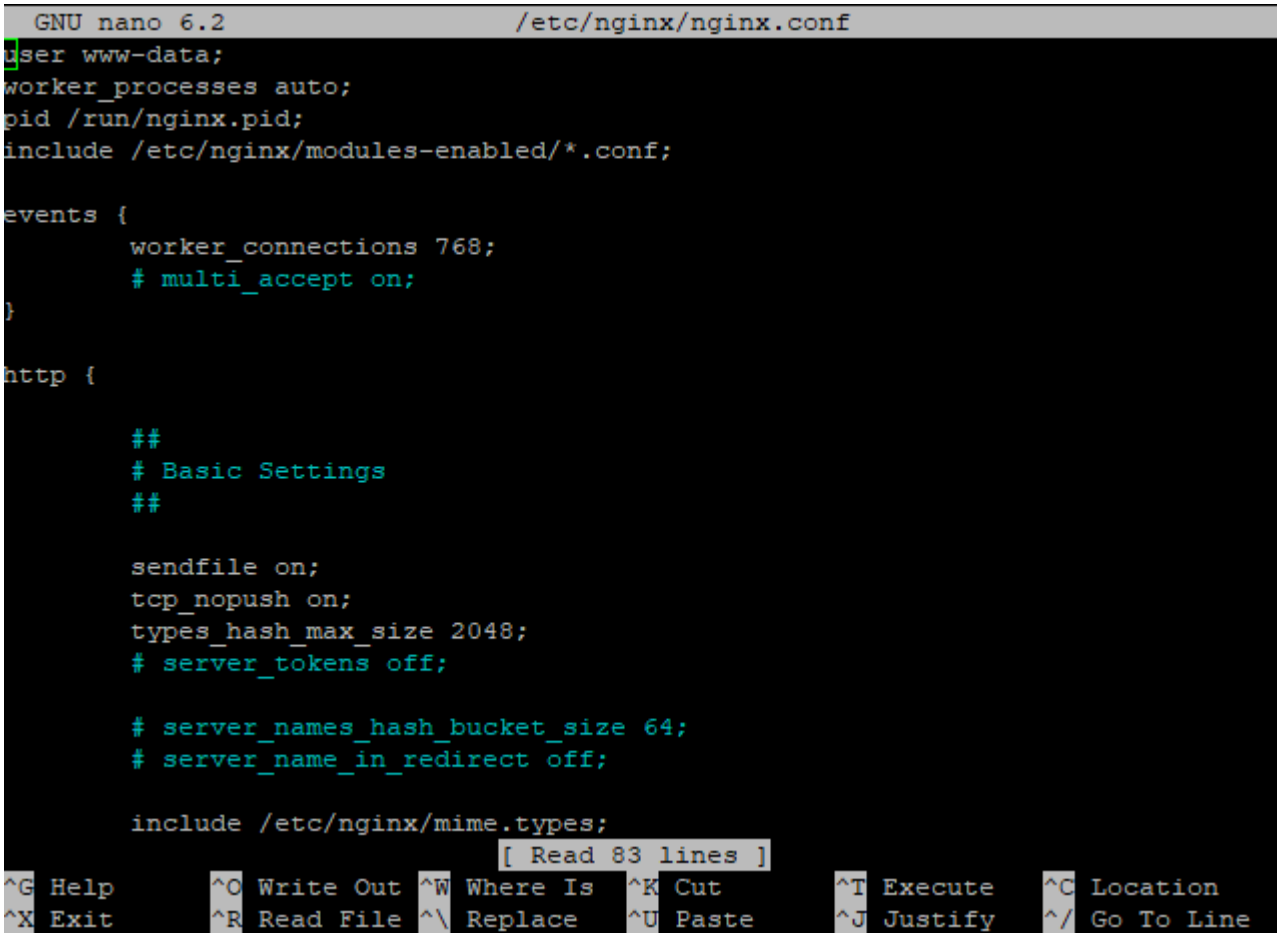
Настройка Nginx

Администрирование веб-сервера представляет из себя изменение и поддержку конфигурационных файлов. Среди них 1 файл конфигурации и 2 каталога. Это **nginx.conf, sites-available и sites-enabled** соответственно. Все они лежат в директории /etc/nginx.

Файл **nginx.conf** — это главный конфигурационный файл. Каталог **sites-available** содержит файлы конфигураций виртуальных хостов. Каждый отдельный файл хранит информацию об определенном сайте. Это его имя, IP-адрес и другие данные. Каталог **sites-enabled**, в свою очередь, состоит только из конфигураций активных сайтов. Только из директории **sites-enabled** читаются файлы конфигурации для виртуальных хостов. Также в ней хранятся ссылки на **sites-available**. Такая структура позволяет временно отключать сайты без потери их конфигураций.

Рассмотрим более детально главный файл конфигурации. Для этого откроем его для просмотра, используя редактор:

```
sudo nano /etc/nginx/nginx.conf
```



Каждый отдельный модуль — это директива, которая отвечает за определенные настройки веб-сервера. Они бывают простыми и блочными. Блочные директивы, помимо имени и параметров, хранят набор дополнительных инструкций, размещенных внутри фигурных скобок.

Перечислим некоторую часть директив главного конфигурационного файла:

- user — это пользователь, от которого осуществляются все рабочие процессы.
- worker_processes — число рабочих процессов сервера. Оно должно быть не больше, чем количество ядер процессора. Параметр auto установит число автоматически.
- pid — файл с номером главного процесса.
- include — отвечает за подключение иных файлов конфигурации, удовлетворяющих заданной маске.
- events — контекст, состоящий из директив, влияющих на работу сетевого соединения.
    - worker_connections — максимальное число одновременно работающих соединений одного рабочего процесса.
    - multi_accept — флаг, который может быть как включен (on), так и выключен (off). Если он включен, то рабочий процесс будет принимать все новые соединения, иначе только одно.
    - use — указывает метод обработки соединений. По умолчанию сервер выбирает наиболее подходящий и эффективный.
- http — контекст, состоящий из директив, отвечающих за работу HTTP-сервера.
    - sendfile — включает (on) или отключает (off) метод отправки данных sendfile().
    - tcp_nopush, tcp_nodelay — параметры, влияющие на производительность. Первый заставляет сервер отправлять заголовки HTTP-ответов одним пакетом, а второй позволяет не буферизировать данные и отправлять их короткими очередями.
    - keepalive_timeout — параметр, отвечающий за время ожидания keep-alive соединения до его разрыва со стороны сервера.
    - keepalive_requests — максимальное число запросов по одному keep-alive соединению.
    - error_log — лог ошибок веб-сервера. Для сбора ошибок в определенной секции (http, server и т.д.) необходимо разместить директиву внутри нее.
    - gzip — сжатие контента.

Настройка виртуальных хостов

На сервере может быть расположено множество сайтов. Все запросы приходят на его IP-адрес, а после веб-сервер определяет, какой дать ответ, в зависимости от домена. Виртуальные хосты предназначены для того, чтобы сервер понимал, что и к какому домену относится. В качестве примера создадим сайт testsite.dev.

Создадим папку для сайта:

```
sudo mkdir -p /var/www/testsite.dev/html
```

После добавим индексный файл:

```
sudo nano /var/www/testsite.dev/html/index.html
```

Заполним его минимальными данными для отображения сайта:

```
<!DOCTYPE html>
<html lang="ru">
<head>
```

```
    <title>testsite.dev</title>
    <meta charset="utf-8">
</head>
<body>
    <h1>Hello, user</h1>
</body>
</html>
```

После создадим конфигурационный файл сайта в папке sites-available:

```
sudo nano /etc/nginx/sites-available/testsite.dev.conf
```

Заполним его простейшей конфигурацией:

```
server {
    listen 80;
    listen [::]:80;

    server_name testsite.dev www.testsite.dev;
    root /var/www/testsite.dev/html;
    index index.html index.xml;
}
```

Последнее, что осталось сделать, — это создать ссылку в директории sites-enabled на конфигурацию сайта testsite.dev, чтобы добавить его из доступных во включенные:

```
sudo ln -s /etc/nginx/sites-available/testsite.dev.conf /etc/nginx/sites-
enabled/
```

После создания виртуального хоста проведем тестирование конфигурации:

```
sudo nginx -t
```

Отключим сайт по умолчанию, удалив запись о дефолтном виртуальном хосте:

```
sudo rm /etc/nginx/sites-enabled/default
```

Стоит уточнить, что после того, как мы отключим сайт по умолчанию, Nginx будет использовать первый встреченный серверный блок в качестве резервного сайта (то есть по IP-адресу сервера будет открываться самый первый сайт из конфигурации Nginx).

Перезагружаем веб-сервер:

```
sudo systemctl restart nginx
```

Проверим, что всё было сделано верно и сайт работает. Для этого можно вставить IP-адрес сервера или домен, если он зарегистрирован, в адресную строку браузера:

Заключение

В данной статье мы разобрали процесс установки Nginx на Linux, а именно на дистрибутив Ubuntu. С помощью этой инструкции можно провести базовую настройку веб-сервера и развернуть на нем свой первый сайт. Кроме этого, сервер подготовлен к переходу на зашифрованный протокол данных. Для этого нужно получить SSL-сертификат и настроить переадресацию с HTTP-протокола на HTTPS. Для настройки защищенного соединения вам будет необходимо сертификат SSL — заказать его можно в панели управления в разделе «SSL-сертификаты».

Ссылки и Дополнения

- Ссылка на оригинальную статью
- Команда Update-alternatives: подробное руководство для пользователей Linux