CONTENTS

// TUTORIAL //

How To Install Nginx on Debian 11

Published on July 20, 2022

Debian 11 Nginx

Kong Yang, Kathleen Juell, and Erika Heidi

Not using Debian 11?
Choose a different version or distribution.

Debian 11

Introduction

Nginx is a free and open-source web server used to host websites and applications of all sizes. The software is known for its low impact on memory resources, high scalability, and its modular, event-driven architecture which can offer secure, predictable performance. More than just a web server, Nginx also works as a load balancer, an HTTP cache, and a reverse proxy.

In this guide, you will install Nginx on a Debian 11 server, adjust the firewall settings, and learn how to manage some Nginx processes.

Prerequisites

Before you begin this guide, you should have a regular, non-root user with sudo privileges configured on your server. You should also have an active firewall. You can learn how to set this up by following our initial server setup guide for Debian 11.

If you want to complete the last steps of this tutorial, you will need a registered domain name. To learn more about setting up a domain name with DigitalOcean, see our introduction to DigitalOcean DNS.

Step 1 – Installing Nginx

Nginx is available in Debian's default software repositories, making it possible to install it from conventional package management tools.

First update your local package index to reflect the latest upstream changes:

Copy

```
$ sudo apt update
```

Then, install the `nginx` package:

Copy

```
$ sudo apt install nginx
```

Confirm the installation, entering `Y`, then press `Enter` to proceed. `apt` will then install Nginx and any required dependencies to your server.

Step 2 – Adjusting the Firewall

Before testing Nginx, it's necessary to modify the firewall settings to allow outside access to the default web ports. Assuming that you followed the instructions in the prerequisites, you should have a UFW firewall configured to restrict access to your server.

During installation, Nginx registers itself with UFW to provide a few application profiles that can be used to enable or disable access to Nginx through the firewall.

List the `ufw` application profiles by typing:

Copy

```
$ sudo ufw app list
```

You should get a listing of the application profiles:

```
Output
Available applications:
...
  Nginx Full
  Nginx HTTP
  Nginx HTTPs
  OpenSSH
…
```

From the output, there are three profiles available for Nginx:

- **Nginx Full**: This profile opens both port `80` (normal, unencrypted web traffic) and port `443` (TLS/SSL encrypted traffic)
- **Nginx HTTP**: This profile opens only port `80` (normal, unencrypted web traffic)
- **Nginx HTTPS**: This profile opens only port `443` (TLS/SSL encrypted traffic)

It is recommended that you enable the most restrictive profile that will still allow the traffic you've configured. Since you haven't configured TLS/SSL for your server yet in this guide, you will only need to allow traffic for HTTP on port `80`.

You can enable this by typing:

```
$ sudo ufw allow 'Nginx HTTP'
```

You can verify the change by typing:

```
$ sudo ufw status
```

The output indicates which HTTP traffic is allowed:

```
Output
Status: active

To                          Action      From
--                          ------      ----
OpenSSH                     ALLOW       Anywhere
Nginx HTTP                  ALLOW       Anywhere
OpenSSH (v6)                ALLOW       Anywhere (v6)
Nginx HTTP (v6)             ALLOW       Anywhere (v6)
```

Step 3 – Checking your Web Server

At the end of the installation process, Debian 11 starts Nginx. The web server should already be up and running.

You can check with the `systemd` init system to make sure the service is running by typing:

```
$ systemctl status nginx
```

```
Output
● nginx.service - A high performance web server and a reverse proxy server
     Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: enabled)
     Active: active (running) since Tue 2022-07-12 20:59:20 UTC; 17h ago
       Docs: man:nginx(8)
   Main PID: 2887 (nginx)
      Tasks: 2 (limit: 1132)
     Memory: 4.2M
        CPU: 81ms
     CGroup: /system.slice/nginx.service
```

```
           ├─2887 nginx: master process /usr/sbin/nginx -g daemon on; master_process on;
           └─2890 nginx: worker process
```

This output reveals that the service has started successfully. However, the best way to test this is to actually request a page from Nginx.

You can access the default Nginx landing page to confirm that the software is running properly by navigating to your server's IP address. If you do not know your server's IP address, you can type this at your server's command prompt:

Copy

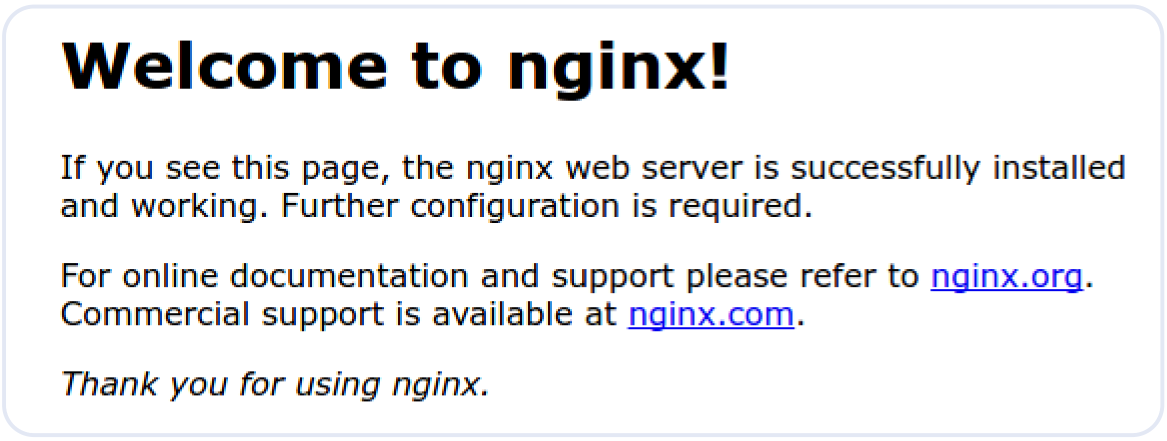
```
$ ip addr show eth0 | grep inet | awk '{ print $2; }' | sed 's/\/.*$//'
```

You will get back a few lines. You can try each in your web browser to see if they work.

When you have your server's IP address, enter it into your browser's address bar:

```
http://your_server_ip
```

The default Nginx landing page should appear in your web browser:



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

This page is included with Nginx to show you that the server is running correctly.

Step 4 – Managing the Nginx Process

Now that you have your web server up and running, you can review some basic management commands.

To stop your web server, type:

Copy

```
$ sudo systemctl stop nginx
```

To start the web server when it is stopped, type:

Copy

```
$ sudo systemctl start nginx
```

To stop and then start the service again, type:

Copy

```
$ sudo systemctl restart nginx
```

If you are making configuration changes, Nginx can often reload without dropping connections. To do this, type:

Copy

```
$ sudo systemctl reload nginx
```

By default, Nginx is configured to start automatically when the server boots. If this is not what you want, you can disable this behavior by typing:

Copy

```
$ sudo systemctl disable nginx
```

To re-enable the service to start up at boot, you can type:

Copy

```
$ sudo systemctl enable nginx
```

Step 5 – Setting Up Server Blocks (Optional)

When using the Nginx web server, *server blocks* (similar to virtual hosts in Apache) can be used to encapsulate configuration details and host more than one domain on a single server. The following examples use `your_domain`, but you should replace this with your actual domain name.

Nginx on Debian 11 has one server block enabled by default that is configured to serve documents out of a directory at `/var/www/html`. While this works well for a single site, it can become unmanageable if you are hosting multiple sites. Instead of modifying `/var/www/html`, create a directory structure within `/var/www` for the `your_domain` website, leaving `/var/www/html` in place as the default directory to be served if a client request doesn't match any other sites.

Create the directory for `your_domain` as follows, using the `-p` flag to create any necessary parent directories:

Copy

```
$ sudo mkdir -p /var/www/your_domain/html
```

Next, assign ownership of the directory with the `$USER` environment variable, which should reference your current system user:

Copy

```
$ sudo chown -R $USER:$USER /var/www/your_domain/html
```

The permissions of your web root should be correct if you haven't modified your `umask` value, but you can make sure by typing:

Copy

```
$ sudo chmod -R 755 /var/www/your_domain
```

Next, create a sample `index.html` page using `nano` or your preferred text editor:

Copy

```
$ nano /var/www/your_domain/html/index.html
```

Inside, add the following sample HTML:

/var/www/your_domain/html/index.html

```
<html>
    <head>
        <title>Welcome to  your_domain </title>
    </head>
    <body>
        <h1>Success! Your Nginx server is successfully configured for <em> your_domain </em>. </h1>
<p>This is a sample page.</p>
    </body>
</html>
```

Save and close the file when you are finished. In `nano` you can do this by pressing `CTRL + X`, then `Y` when prompted, and then `ENTER`.

In order for Nginx to serve this content, you must create a server block with the correct directives that point to your custom web root. Instead of modifying the default configuration file directly, make a new one at `/etc/nginx/sites-available/your_domain`:

Copy

```
$ sudo nano /etc/nginx/sites-available/ your_domain
```

Add the following configuration block, which is similar to the default, but updated for your new directory and domain name:

/etc/nginx/sites-available/your_domain

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

        root /var/www/ your_domain /html;
        index index.html index.htm index.nginx-debian.html;

        server_name  your_domain  www. your_domain ;

        location / {
                try_files $uri $uri/ =404;
        }
}
```

Notice the updated `root` configuration to your new directory and the `server_name` to your domain name. Remember to replace `your_domain` with your actual domain name here.

Next, enable this server block by creating a symbolic link to your custom configuration file inside the `sites-enabled` directory, which Nginx reads from during startup:

Copy

```
$ sudo ln -s /etc/nginx/sites-available/ your_domain  /etc/nginx/sites-enabled/
```

Your server now has two server blocks enabled and configured to respond to requests based on their `listen` and `server_name` directives (you can read more about how Nginx processes these directives here):

- `your_domain`: Will respond to requests for `your_domain` and `www.your_domain`.
- `default`: Will respond to any requests on port `80` that do not match the other two blocks.

To avoid a possible hash bucket memory problem that can arise from adding additional server names to your configuration, it is necessary to adjust a single value in the `/etc/nginx/nginx.conf` file. Open the file:

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

Find the `server_names_hash_bucket_size` directive and remove the `#` symbol to uncomment the line:

/etc/nginx/nginx.conf

```
...
http {
    ...
    server_names_hash_bucket_size 64;
    ...
}
...
```

Save and close the file when you are finished.

Next, test to make sure that there are no syntax errors in any of your Nginx files:

```
$ sudo nginx -t
```

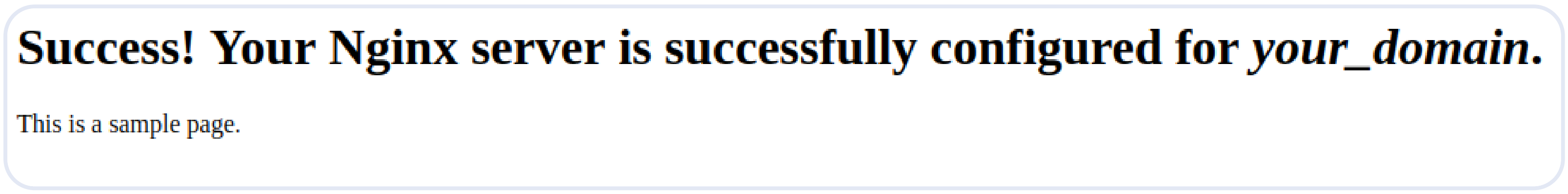
If there aren’t any problems, the following is the output:

```
Output
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
```

Once your configuration test passes, restart Nginx to enable your changes:

```
$ sudo systemctl restart nginx
```

Nginx should now be serving your domain name. You can test this by navigating to `http://your_domain`. The custom HTML your created in the `/var/www/your_domain/html/index.html` folder should be rendered here:



Success! Your Nginx server is successfully configured for *your_domain*.

This is a sample page.

Step 6 – Getting Familiar with Important Nginx Files and Directories

Now that you know how to manage the Nginx service itself, you can take some time to familiarize yourself with a few important directories and files.

Content

- `/var/www/html`: The actual web content, which by default only consists of the default Nginx page you saw earlier, is served out of the `/var/www/html` directory. This can be changed by altering Nginx configuration files.

Server Configuration

- `/etc/nginx`: The Nginx configuration directory. All of the Nginx configuration files reside here.
- `/etc/nginx/nginx.conf`: The main Nginx configuration file. This can be modified to make changes to the Nginx global configuration.
- `/etc/nginx/sites-available/`: The directory where per-site server blocks can be stored. Nginx will not use the configuration files found in this directory unless they are linked to the `sites-enabled` directory. Typically, all server block configuration is done in this directory, and then enabled by linking to the other directory.
- `/etc/nginx/sites-enabled/`: The directory where enabled per-site server blocks are stored. Typically, these are created by linking to configuration files found in the `sites-available` directory.
- `/etc/nginx/snippets`: This directory contains configuration fragments that can be included elsewhere in the Nginx configuration. Potentially repeatable configuration segments are good candidates for refactoring into snippets.

Server Logs

- `/var/log/nginx/access.log`: Every request to your web server is recorded in this log file unless Nginx is configured to do otherwise.
- `/var/log/nginx/error.log`: Any Nginx errors will be recorded in this log.

Conclusion

Now that you have your web server installed, you have many options for the type of content you can serve and the technologies you can use to create a richer experience for your users.

About the authors

Kong Yang Author

Kathleen Juell Author

Erika Heidi Author
Developer Advocate

Dev/Ops passionate about open source, PHP, and Linux.

Comments

Leave a comment

Leave a comment...

This textbox defaults to using **Markdown** to format your answer.

You can type **!ref** in this text area to quickly search our full set of tutorials, documentation & marketplace offerings and insert the link!

Sign In or Sign Up to Comment

Popular Topics

AI/ML

Ubuntu

Linux Basics

JavaScript

Python

MySQL

Docker

Kubernetes

All tutorials →

Talk to an expert →

Congratulations on unlocking the whale ambience easter egg!

Thank you to the Glacier Bay National Park & Preserve and Merrick079 for the sounds behind this easter egg.

Click the whale button in the bottom left of your screen to toggle some ambient whale noises while you read.

Reset easter egg to be discovered again

Permanently dismiss and hide easter egg

Interested in whales, protecting them, and their connection to helping prevent climate change? We recommend checking out the Whale and Dolphin Conservation.

Become a contributor for community

Get paid to write technical tutorials and select a tech-focused charity to receive a matching donation.

Sign Up →

DigitalOcean Documentation

Full documentation for every DigitalOcean product.

Learn more →

Resources for startups and SMBs

The Wave has everything you need to know about building a business, from raising funding to marketing your product.

Learn more →

Get our newsletter

Stay up to date by signing up for DigitalOcean's Infrastructure as a Newsletter.

Email address Submit

New accounts only. By submitting your email you agree to our Privacy Policy

Share your ideas