Nginx Essentials: Installation and Configuration Troubleshooting

Posted December 12, 2017 38.8k NGINX

By: Mark Drake

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.

As with any complex software tool, it can be difficult to remember the specific commands and best practices for managing an Nginx server or troubleshooting whatever issues may arise. This cheatsheet-style guide is intended to serve as a quick reference for anyone working with Nginx. It will cover some basic service management commands, as well as tips for diagnosing and resolving some common issues.

How To Use This Guide:

- Each section can be used independently of others, so feel free to skip to whichever sections are relevant to your needs.
- The commands in each section of this guide are self-contained, and you should substitute the red values in the example commands with your own values.
- When relevant, sections in this guide include links to other resources which you can consult for more information.
- This guide assumes that you're working with a version of Nginx installed from the default repositories of a Debian-based distribution. Please note that some of the conventions described in this guide are not present on other distributions or in versions of Nginx from other sources.

Installing Nginx

Using `sudo apt-get`, update your package indexes and then install the service:

```
$ sudo apt-get update
$ sudo apt-get install nginx
```

For more details on the installation and setup process, follow our tutorial on How To Install Nginx on Ubuntu 16.04.

Checking the Status of Nginx

You can check whether or not Nginx is running on your machine by entering the following into your command prompt:

```
$ sudo systemctl status nginx
```

Enabling Nginx

By default, Nginx is configured to start automatically when the server boots. If desired, you can disable this behavior by typing:

```
$ sudo systemctl disable nginx
```

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

```
$ sudo systemctl enable nginx
```

Stopping, Starting, and Reloading Nginx

To stop your already-running Nginx server:

```
$ sudo systemctl stop nginx
```

Once the server has been stopped, you can start it again by typing:

```
$ sudo systemctl start nginx
```

To stop and then start Nginx again, type:

```
$ sudo systemctl restart nginx
```

You also have the ability to reload Nginx without disrupting connections:

```
$ sudo systemctl reload nginx
```

To learn more about `systemd` and the `systemctl` command, check out this introduction to systemd essentials.

Creating a Document Root for a Static Site

When using the Nginx web server, server blocks (similar to the virtual hosts in Apache) are used to host more than one domain on a single server. Each server block has its own document root, a special directory which Nginx must check before serving the domain's content.

The commands in the block below will create a new document root, modify ownership of the document root to your non-root user, and modify the permissions of each subdirectory within `/var/www/`.

```
$ sudo mkdir -p /var/www/example.com/html
$ sudo chown -R $USER:$USER /var/www/example.com/html
$ find /var/www -type d -exec chmod 775 {} \;
```

In this example, we are making sure that the document root directories have global read and execute privileges, but you should substitute a different value for `775` to reflect your specific needs.

Creating a Document Root for a Dynamically Processed Site

When using Nginx with certain programs (e.g., PHP-FPM) to produce a dynamically-processed site, you may need to adjust some files' permissions to allow the `www-data` group access or even ownership, especially if it needs to be able to write to the directory.

The commands in the block below will create a new document root, modify ownership of the document root to the `www-data` group, and modify the permissions of each subdirectory within `/var/www`.

```
$ sudo mkdir -p /var/www/example.com/html
$ sudo chown -R www-data:www-data /var/www/example.com
$ sudo find /var/www -type d -exec chmod 775 {} \;
```

To learn more about permissions, see our introduction to Linux permissions. It may also be helpful to review our tutorial on How To Set Up Nginx Server Blocks (Virtual Hosts) on Ubuntu 16.04, which provides a detailed approach for creating and changing document roots.

Enabling Configuration Files

We can enable a server block's configuration file by creating a symbolic link from the `sites-available` directory to the `sites-enabled` directory, which Nginx will read during startup.

To do this, enter the following command:

```
$ sudo ln -s /etc/nginx/sites-available/example.com /etc/nginx/sites-enabled/
```

After linking the files, reload Nginx to reflect the change and enable the server block's configuration file:

```
$ sudo systemctl reload nginx
```

Resolving Hash Bucket Memory Issues

Nginx uses hash tables (which are organized into "buckets") to quickly process static data like server names or MIME types. Thus, if you've added multiple server names, there's a chance that the size of the server name hash buckets will no longer be sufficient and you will see a `server_names_hash_bucket_size` error as you make changes. This can be addressed by adjusting a single value within your `/etc/nginx/nginx.conf` file.

To open this config file, enter:

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

Within the file, find the `server_names_hash_bucket_size` directive. Remove the `#` symbol to uncomment the line, and increase the directive's value by the next power of two:

/etc/nginx/nginx.conf

```
http {
    . . .

    server_names_hash_bucket_size 64;

    . . .
}
```

Doing this will increase the bucket size of Nginx's server names hash tables and allow the service to process all the server names you've added. Save and close the file when you are finished, and then restart Nginx to reflect the changes.

Checking Your Configuration File

Whenever you make changes to your Nginx configuration file, it's important check whether you've left behind any syntax errors. This can be done by issuing the following command:

```
$ sudo nginx -t
```

If there are errors in your configuration file, the command's output will tell you exactly where in the file the error was found. Conversely, if there are no syntax errors in any of your nginx config files, you will see output similar to the following:

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

If no errors were found and you'd like to apply your changes immediately, restart the service:

```
$ sudo systemctl restart nginx
```

Important Nginx Files and Directories

As you spend time working with Nginx, you may find yourself frequently accessing the following files and directories:

Content

- `/var/www/html` : This is the location of the default document root from which the actual web content is served. The document root can be changed by altering Nginx configuration files.

Server Configuration

- `/etc/nginx/` : The default Nginx configuration directory where all your Nginx config files can be found.
- `/etc/nginx/nginx.conf` : The primary Nginx configuration file. This can be directed to make global changes to Nginx's configuration.
- `/etc/nginx/sites-available/default` : Nginx's default server block file. Other per-site server blocks are also stored within the `sites-available` directory, although these will not be used unless they are linked to in the `sites-enabled` 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.

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.
- To access the Nginx process's systemd logs, run the following command:

```
$ sudo journalctl -u nginx
```

Conclusion

This guide covers basic commands and practices for managing an Nginx server, including how to start, stop, and check the status of Nginx, how to find a website's document root, and how to check the syntax of an Nginx configuration file. To learn more about working with Nginx, we recommend going over the following tutorials.

- How To Optimize Nginx Configuration
- Apache vs Nginx: Practical Considerations
- Understanding Nginx Server and Location Block Selection Algorithms

By: Mark Drake

Related Tutorials

How To Upgrade Nginx In-Place Without Dropping Client Connections

How To Target Your Users with Nginx Analytics and A/B Testing

How To Deploy a PHP Application with Kubernetes on Ubuntu 16.04

How To Ensure Code Quality with SonarQube on Ubuntu 18.04

How To Set Up a Private Docker Registry on Ubuntu 18.04

0 Comments

Leave a comment...

Log In to Comment