

How To Install the Apache Web Server on Ubuntu 18.04 [Quickstart]



Posted July 23, 2018 67.6k [APACHE](#) [QUICKSTART](#) [UBUNTU](#) [UBUNTU 18.04](#)

By: Kathleen Juell

Introduction

The Apache HTTP server is the most widely-used web server in the world. It provides many powerful features, including dynamically loadable modules, robust media support, and extensive integration with other popular software.

In this guide, we'll explain how to install an Apache web server on your Ubuntu 18.04 server. For a more detailed version of this tutorial, please refer to [How To Install the Apache Web Server on Ubuntu 18.04](#).

Prerequisites

Before you begin this guide, you should have the following:

- An Ubuntu 18.04 server and a regular, non-root user with sudo privileges. Additionally, you will need to enable a basic firewall to block non-essential ports. You can learn how to configure a regular user account and set up a firewall for your server by following our [initial server setup guide for Ubuntu 18.04](#).

When you have an account available, log in as your non-root user to begin.

Step 1 — Installing Apache

Apache is available within Ubuntu's default software repositories, so you can install it using conventional package management tools.

Update your local package index:

```
$ sudo apt update
```

Install the `apache2` package:

[SCROLL TO TOP](#)

```
$ sudo apt install apache2
```

Step 2 — Adjusting the Firewall

Check the available `ufw` application profiles:

```
$ sudo ufw app list
```

Output

```
Available applications:
  Apache
  Apache Full
  Apache Secure
  OpenSSH
```

Let's enable the most restrictive profile that will still allow the traffic you've configured, permitting traffic on port `80` (normal, unencrypted web traffic):

```
$ sudo ufw allow 'Apache'
```

Verify the change:

```
$ sudo ufw status
```

Output

```
Status: active

To Action From
--
OpenSSH ALLOW Anywhere
Apache ALLOW Anywhere
OpenSSH (v6) ALLOW Anywhere (v6)
Apache (v6) ALLOW Anywhere (v6)
```

Step 3 — Checking your Web Server

Check with the `systemd` init system to make sure the service is running by typing:

```
$ sudo systemctl status apache2
```

Output

- `apache2.service` - The Apache HTTP Server

Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)

Drop-In: /lib/systemd/system/apache2.service.d

└─apache2-systemd.conf

Active: **active (running)** since Tue 2018-04-24 20:14:39 UTC; 9min ago

Main PID: 2583 (apache2)

Tasks: 55 (limit: 1153)

CGroup: /system.slice/apache2.service

├─2583 /usr/sbin/apache2 -k start

├─2585 /usr/sbin/apache2 -k start

└─2586 /usr/sbin/apache2 -k start

Access the default Apache landing page to confirm that the software is running properly through your IP address:

`http://your_server_ip`

You should see the default Ubuntu 18.04 Apache web page:



Apache2 Ubuntu Default Page

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in [/usr/share/doc/apache2/README.Debian.gz](#)**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the [manual](#) if the `apache2-doc` package was installed on this server.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|   |-- ports.conf
|-- mods-enabled
|   |-- *.load
|   |-- *.conf
|-- conf-enabled
|   |-- *.conf
|-- sites-enabled
|   |-- *.conf
```

- `apache2.conf` is the main configuration file. It puts the pieces together by including all remaining configuration files when starting up the web server.
- `ports.conf` is always included from the main configuration file. It is used to determine the listening ports for incoming connections, and this file can be customized anytime.
- Configuration files in the `mods-enabled/`, `conf-enabled/` and `sites-enabled/` directories contain particular configuration snippets which manage modules, global configuration fragments, or virtual host configurations, respectively.
- They are activated by symlinking available configuration files from their respective `*-available/` counterparts. These should be managed by using our helpers `a2enmod`, `a2dismod`, `a2ensite`, `a2dissite`, and `a2enconf`, `a2disconf`. See their respective man pages for detailed information.
- The binary is called `apache2`. Due to the use of environment variables, in the default configuration, `apache2` needs to be started/stopped with `/etc/init.d/apache2` or `apache2ctl`. **Calling `/usr/bin/apache2` directly will not work** with the default configuration.

Document Roots

By default, Ubuntu does not allow access through the web browser to *any* file apart of those located in `/var/www`, **public_html** directories (when enabled) and `/usr/share` (for web applications). If your site is using a web document root located elsewhere (such as in `/srv`) you may need to whitelist your document root directory in `/etc/apache2/apache2.conf`.

The default Ubuntu document root is `/var/www/html`. You can make your own virtual hosts under `/var/www`. This is different to previous releases which provides better security out of the box.

Reporting Problems

Please use the `ubuntu-bug` tool to report bugs in the Apache2 package with Ubuntu. However, check **existing bug reports** before reporting a new bug.

Please report bugs specific to modules (such as PHP and others) to respective packages, not to the web server itself.

When using the Apache web server, you can use *virtual hosts* (similar to server blocks in Nginx) to encapsulate configuration details and host more than one domain from a single server. We will set up a domain called **example.com**, but you should **replace this with your own domain name**. To learn more about setting up a domain name with DigitalOcean, see our [introduction to DigitalOcean DNS](#).

Create the directory for **example.com**, using the `-p` flag to create any necessary parent directories:

```
sudo mkdir -p /var/www/example.com/html
```

Assign ownership of the directory:

```
$ sudo chown -R $USER:$USER /var/www/example.com/html
```

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

```
$ sudo chmod -R 755 /var/www/example.com
```

Create a sample `index.html` page using `nano` or your favorite editor:

```
$ nano /var/www/example.com/html/index.html
```

Inside, add the following sample HTML:

```
                                /var/www/example.com/html/index.html

<html>
  <head>
    <title>Welcome to Example.com!</title>
  </head>
  <body>
    <h1>Success! The example.com server block is working!</h1>
  </body>
</html>
```

Save and close the file when you are finished.

Make a new virtual host file at `/etc/apache2/sites-available/example.com.conf`:

```
$ sudo nano /etc/apache2/sites-available/example.com.conf
```

Paste in the following configuration block, updated for our new directory and domain name

```
<VirtualHost *:80>
    ServerAdmin admin@example.com
    ServerName example.com
    ServerAlias www.example.com
    DocumentRoot /var/www/example.com/html
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

Save and close the file when you are finished.

Enable the file with `a2ensite`:

```
$ sudo a2ensite example.com.conf
```

Disable the default site defined in `000-default.conf`:

```
$ sudo a2dissite 000-default.conf
```

Test for configuration errors:

```
$ sudo apache2ctl configtest
```

You should see the following output:

Output

```
Syntax OK
```

Restart Apache to implement your changes:

```
$ sudo systemctl restart apache2
```

Apache should now be serving your domain name. You can test this by navigating to `http://example.com`, where you should see something like this:

Success! The example.com virtual host is working!

Conclusion

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

If you'd like to build out a more complete application stack, check out this [article on how to configure a LAMP stack on Ubuntu 18.04](#).

By: Kathleen Juell

 Upvote (7)  Subscribe  Share



We just made it easier for you to deploy faster.

[TRY FREE](#)

Related Tutorials

[How To Sync and Share Your Files with Seafile on Debian 9](#)

[How To Install and Configure an Apache ZooKeeper Cluster on Ubuntu 18.04](#)

[How To Sync and Share Your Files with Seafile on Ubuntu 18.04](#)

[How to Back Up, Import, and Migrate Your Apache Kafka Data on Ubuntu 18.04](#)

[How To Install and Configure pgAdmin 4 in Server Mode](#)

0 Comments

Leave a comment...

SCROLL TO TOP

Log In to Comment



This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.



Copyright © 2019 DigitalOcean™ Inc.

[Community](#) [Tutorials](#) [Questions](#) [Projects](#) [Tags](#) [Newsletter](#) [RSS](#) 

[Distros & One-Click Apps](#) [Terms, Privacy, & Copyright](#) [Security](#) [Report a Bug](#) [Write for DOnations](#) [Shop](#)

SCROLL TO TOP