Apache Basics: Installation and Configuration Troubleshooting

12

Posted December 12, 2017 35.5k APACHE

By: Kathleen Juell

Introduction

The Apache web server is an open-source web server popular for its flexibility, power, and widespread support. In this guide, we'll go over some common procedures for managing the Apache server, including stopping, starting, and enabling the service, working with virtual host files and directories, and locating important files and directories on your server.

This guide is oriented around users working with Apache on Ubuntu. Users working on CentOS and other RHEL-based distributions can check out section one in this tutorial for information about installing Apache on CentOS.

How To Use This Guide:

- Each section can be used independently of others, so feel free to skip to sections that are relevant to your needs.
- Each command is self-contained, allowing you to substitute your own values for those in red.
- Each section has linked resources, which you can consult for more information about covered topics.

Installing the Apache Service

To install the Apache service, update your package indexes, then install:

```
$ sudo apt-get update
$ sudo apt-get install apache2
```

For more details on the installation process, follow our tutorial on How To Install the Apache Web Server on Ubuntu 16.04.

Enabling and Disabling the Apache Unit

Our Apache service is configured to start automatically at boot. If we wanted to modify this behavior, however, we could type the following:

```
$ sudo systemctl disable apache2.service
```

To allow Apache to start up again at boot:

```
$ sudo systemctl enable apache2.service
```

Stopping, Starting, and Reloading Apache

To stop the Apache server, type the following command:

```
$ sudo systemctl stop apache2
```

To start the Apache server, type:

```
$ sudo systemctl start apache2
```

To stop the service and start it again, type:

```
$ sudo systemctl restart apache2
```

If you are making configuration changes, you can reload Apache without dropping connections. Type the following command:

```
$ sudo systemctl reload apache2
```

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

Checking the Server Status

To check the status of your Apache server, type:

```
$ sudo systemctl status apache2
```

The output from this command will tell you whether or not Apache is running, and will show you the last few lines in the log files.

Creating a Document Root Directory for a Static Website

When using Apache to build websites, developers frequently utilize `virtual hosts` —units that comprise individual sites or domains. This process involves creating a directory for the `document root` , the top-level directory Apache checks when serving content.

Create the directory:

```
$ sudo mkdir -p /var/www/example.com/public_html
```

Assign ownership of the directory to your non-root user:

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

Allow read access to the general web directory:

```
$ sudo find /var/www -type d -exec chmod 775 {} \;
```

For more about permissions, see our introduction to Linux permissions. Keep in mind that your permissions may change with your needs and use cases.

Creating a Document Root Directory for Dynamic Processing Modules

If you are working with a dynamic processing module like PHP, you will create your document root directory as follows:

```
$ sudo mkdir -p /var/www/example.com/public_html
```

Assign ownership of the directory to your non-root user, and group ownership to the `www-data` group:

```
$ sudo chown -R sammy:www-data /var/www/example.com/public_html
```

Modifying Configuration Settings

When working with virtual hosts, it is necessary to modify configuration settings to reflect domain specifics, so that Apache can respond correctly to domain requests.

Open your virtual host configuration file:

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

Modify the following:

```
ServerAdmin admin@example.com
ServerName example.com
ServerAlias www.example.com
DocumentRoot /var/www/example.com/public_html
```

With modifications, the file should look like this (provided it has not been modified before):

/etc/apache2/sites-available/example.com.conf

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

When troubleshooting, be sure to double check this file and its directives.

For more detail about working with virtual hosts, see our discussion on working with Apache virtual hosts on Ubuntu 16.04.

Enabling and Disabling Configuration Files

Virtual Host Files

To enable virtual host configuration files:

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

To disable configuration files (in this particular example, the Apache default virtual host configuration file):

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

Modules

To enable the modules found in `/etc/apache2/mods-available`, use the following command:

```
$ sudo a2enmod example_mod
```

To disable a specific module:

```
$ sudo a2dismod example_mod
```

Configuration Files

To enable the configuration files in `/etc/apache2/conf-available`—files not associated with virtual hosts—type:

```
$ sudo a2enconf example-conf
```

To disable a configuration file:

```
$ sudo a2disconf example-conf
```

Configuration Testing

Any time you make changes to configuration files in Apache, be sure to run the following command to check for syntax errors:

```
$ sudo apache2ctl configtest
```

Important Files and Directories

As you continue working with Apache, you will encounter the following directories and files:

Content

- `/var/www/html`: This directory holds the web content of your site, and is its default root. You can modify Apache's default configuration settings to point to other directories within `var/www`.

Server Configuration

- `/etc/apache2`: The configuration directory in Apache, home to all of its configuration files.
- `/etc/apache2/apache2.conf`: Apache's primary configuration file, which stores its global configuration settings. Other files in the configuration directory are loaded from this file. It also stores the `FollowSymLinks` directives, which control configuration enabling and disabling.
- `/etc/apache2/sites-available/`: This directory holds virtual host configuration files, which are enabled through links to the `sites-enabled` directory. Modification to server block files happens in this directory,

and is enabled through the `a2ensite` command.

- `/etc/apache2/sites-enabled/`: Activated virtual host configuration files are stored here. When Apache starts or reloads, it reads the configuration files and links in this directory as it complies a full configuration.
- `/etc/apache2/conf-available` and `/etc/apache2/conf-enabled`: In the same relationship as `sites-available` and `sites-enabled`, these directories house configuration fragments that are unattached to virtual host configuration files.
- `/etc/apache2/mods-available` and `/etc/apache2/mods-enabled`: Containing modules that are available and enabled, these directories have two components: files ending in `.load`, which contain fragments that load particular modules, and files ending in `.conf`, which store the configurations of these modules.

Server Logs

- `/var/log/apache2/access.log`: This file contains every request to the web server unless Apache's configuration settings have been modified.
- `/var/log/apache2/error.log`: This file contains errors. To modify the amount of detail in the error logs, modify the `LogLevel` directive in `/etc/apache2/apache2.conf`.

Another way to access information about the Apache unit is through the `journald` component, which collects log information from applications and the kernel. To see entries for the Apache unit, type:

```
$ sudo journalctl -u apache2
```

Conclusion

In this guide, we've covered some common procedures for managing the Apache server, including stopping, starting, and enabling the service, working with virtual host files and directories, and locating important files and directories on your server. To learn more about working with Apache, take a look at the following resources:

- How To Install a LAMP Stack on Ubuntu 16.04.
- How To Move an Apache Web Root to a New Location on Ubuntu 16.04.
- How To Secure Apache with Let's Encrypt on Ubuntu 16.04.
- How To Use the .htaccess File.
- How To Rewrite URLs with mod_rewrite for Apache on Ubuntu 16.04.
- How To Use Apache as a Reverse Proxy with mod_proxy on Ubuntu 16.04.

By: Kathleen Juell

♡ Upvote (12) Subscribe Share

Related Tutorials

How To Migrate your Apache Configuration from 2.2 to 2.4 Syntax.

How To Get Started With mod_pagespeed with Apache on a CentOS and Fedora Cloud Server

How To Use the .htaccess File

How To Set Up Mod_Rewrite (page 2)

How To Create a Custom 404 Page in Apache

2 Comments

Leave a comment...

Log In to Comment

dmitriym *December 12, 2017*

0 To disable a configuration file, a2disconf should be run. There is a typo in the article.

katjuell MOD *December 12, 2017*

0 Thanks for the heads up! I've updated it.