How To Set Up Apache Virtual Hosts on Debian 7

Tagged In: Apache, Debian, DigitalOcean
Published: Oct 29, 2013 • Updated: Jun 17, 2014

What the Red Means

The lines that the user needs to enter or customize will be in red in this tutorial! The rest should mostly be copy-and-pastable.

Virtual Hosts

Virtual Hosts are used to run more than one domain off of a single IP address. This is especially useful to people who need to run several sites off of one virtual private server-- each will display different information to the visitors, depending on which website the user is accessing.There is no limit to the number of virtual hosts that can be added to a VPS.

Set Up

The steps in this tutorial require the user to have root privileges. You can see how to set that up in the Initial Server Setup. Choose whichever username you fancy.
Additionally, you need to have apache already installed and running on your virtual server. If you haven't already done so, use the following command:

```
sudo apt-get install apache2
```

Step One— Create a New Directory

First, it is necessary to create a directory where we will keep the new website's information. This location will be your Document Root in the Apache virtual configuration file. By adding a -p to the line of code, the command automatically generates all the parents for the new directory.
You will need to designate an actual DNS approved domain (or an IP address) to test that a virtual host is working. In this tutorial, we will use example.com as a placeholder for a correct domain name.

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

*If you want to use an unapproved domain name to test the process, you will find information on how to make it work on your local computer in Step Seven.

Step Two—Grant Permissions

Now you must grant ownership of the directory to the user, as opposed to just keeping it on the root system.

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

Additionally, it is important to make sure that everyone will be able to read your new files.

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

Now you are all done with permissions.

Step Three— Create the Page

Within your configurations directory, create a new file called index.html

```
sudo nano /var/www/example.com/public_html/index.html
```

It's also useful to add some text to the file, in order to have something to look at when the IP redirects to the virtual host.

```
<html>

  <head>

    <title>www.example.com</title>

  </head>

  <body>
```

```
    <h1>Success: You Have Set Up a Virtual Host</h1>

  </body>

</html>
```

Save & Exit.

Step Four—Create the New Virtual Host File

The next step is to set up the apache configuration. We're going to work off a duplicate—go ahead and make a copy of the file (naming it after your domain name) in the same directory:

```
 sudo cp /etc/apache2/sites-available/default /etc/apache2/sites-
available/example.com
```

Step Five—Turn on Virtual Hosts

Open up the new config file:

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

We are going to set up a virtual host in this file.
To begin, insert a line for the ServerName under the ServerAdmin line.

```
  ServerName example.com
```

The ServerName specifies the domain name that the virtual host uses.
If you want to make your site accessible from more than one name (ie with www in the URL), you can include the alternate names in your virtual host file by adding a ServerAlias Line. The beginning of your virtual host file would then look like this:

```
<VirtualHost *:80>

        ServerAdmin webmaster@example.com

        ServerName example.com

        ServerAlias www.example.com

  [...]
```

The next step is to fill in the correct Document Root. For this section, write in the extension of the new directory created in Step One. If the document root is incorrect or absent you will not be able to set up the virtual host.
The section should look like this:

```
 DocumentRoot /var/www/example.com/public_html
```

You do not need to make any other changes to this file. Save and Exit.
The last step is to activate the host with the built-in apache shortcut:

```
 sudo a2ensite example.com
```

Step Six—Restart Apache

Although there have been a lot of changes to the configuration and the virtual host is set up, none of the changes will take effect until Apache is restarted:

```
 sudo service apache2 restart
```

Optional Step Seven—Setting Up the Local Hosts

If you have pointed your domain name to your virtual private server's IP address you can skip this step. However, if want to try out your new virtual hosts without having to connect to an actual domain name, you can set up local hosts on your computer alone.
For this step, make sure you are on the computer itself andnot your droplet.
To proceed with this step, you need to know your computer's administrative password; otherwise, you will be required to use an actual domain name to test the virtual hosts.
If you are on a Mac or Linux, access the root user (`su`) on the computer and open up your hosts file:

```
nano /etc/hosts
```

If you are on a Windows Computer, you can find the directions to alter the host file on the Microsoft site
You can add the local hosts details to this file, as seen in the example below. As long as that line is there, directing your browser toward, say, example.com will give you all the virtual host details for the corresponding IP address.

```
# Host Database

#

# localhost is used to configure the loopback interface

# when the system is booting.  Do not change this entry.

##

127.0.0.1       localhost



#Virtual Hosts

12.34.56.789    example.com
```

However, it may be a good idea to delete these made up addresses out of the local hosts folder when you are done in order to avoid any future confusion.

Step Eight—RESULTS: See Your Virtual Host in Action

Once you have finished setting up your virtual host you can see how it looks online. Type your ip address into the browser (ie. http://12.34.56.789)
It should look somewhat similar to my handy screenshot
Nice work!

Creating More Virtual Hosts

To add more virtual hosts simply repeat the process above, being careful to set up a new document root with the appropriate domain name, and then creating and activating the new virtual host file.

See More

Once you have set up your virtual hosts, you can proceed to Create a SSL Certificate for your site or Install an FTP server.
By Adam LaGreca
Share TutorialImprove TutorialWrite Tutorial

Related Tutorials

- How To Use the DigitalOcean API v2
- How To Configure your Droplet to Only Use IPv6 Networking
- How To Add Additional IPv6 Addresses to your Droplet
- How To Enable IPv6 for DigitalOcean Droplets
- How To Configure OCSP Stapling on Apache and Nginx

Share this Tutorial

16 Comments

-
  lgw4 October 31, 2013
  1. You should not make your web directory owned by a regular user: use the 'www-data' user instead. 2. Making the contents of your entire /var/www directory have 755 permissions is a bad idea. Files should be 644 at most. 3. On OS X or Linux, use sudo, not su.
  Flag
-
  Kamal Nasser November 1, 2013
  @Chip:
  1. You should not make your web directory owned by a regular user: use the 'www-data' user instead.
  Why is that? It's recommended to make the web directory owned by a regular user and not www-data for security reasons.
  3. On OS X or Linux, use sudo, not su.
  This article doesn't use `su` at all. Nonetheless, sudo and su are two different commands that serve two different purposes.
  Flag
-
  lehnerviktor November 21, 2013
  Hi, if a have not a domain, but i would like test a website, uploaded the examle directory: /home/absys/public_html/samplewebsite How to refer our browser , adn how to configure my virtual host ? No domain only test. Thanks.
  Flag
-
  lehnerviktor November 21, 2013
  My domain ip address http://198.199.88.173/
  Flag
-
  reshadfar January 17, 2014
  Doesn;t work here.. :'(
  Flag
-

Kamal Nasser January 18, 2014
@reshadfar: What do you mean by "doesn't work"?
Flag

-
peterson.m.chris February 5, 2014
I get Index of / when I follow this.
Flag

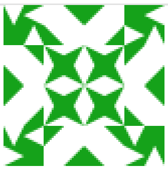
-
Kamal Nasser February 8, 2014
@Chris: That's fine, it means that there are no files in your document root.
Flag

-
monkeydump February 13, 2014
This doesn't work for the most modern apache installs.. In steps four & five, the guide refers to "/apache2/sites-available/default", telling you to copy it & rename it "example.com" However, many people arriving here for help will have the following setting in their /etc/apache2/apache2.conf: # Include the virtual host configurations: IncludeOptional sites-enabled/*.conf Everythings' gotta end in .conf, basically. You'll instead have to use the command for 'Step Four'; sudo cp /etc/apache2/sites-available/000-default.conf /etc/apache2/sites-available/example.conf
Flag

-
monkeydump February 13, 2014
OK, so two further steps I took to finally get this working on my Debian system; 1) My root directory is infact '/var/www/html', not '/var/www' If you've got already a '/var/www/html' directory already, definitely use that instead. 2) To be absolutely clear, this is how my '/etc/apache2/sites-available/example.conf' file ended up for success; ServerAdmin webmaster@example.com ServerName example.com ServerAlias www.example.com DocumentRoot /var/www/html/example.com/public_html ErrorLog ${APACHE_LOG_DIR}/error.log CustomLog ${APACHE_LOG_DIR}/access.log combined It's taking me hours of messing around to get to this point - I hope that this helps someone out there. Once I did get it working, it seems kinda obvious - & daft that it took me so long, but I'm a noob at this stuff. I think my problems stem from the fact that I'm running Debian Jessie, not 'Wheezy'. It's the only thing I can summize.
Flag

-

peterson.m.chris February 18, 2014
@Kamal, my directory has an index.php file. I also added an index.html but I still have to navigate to the public folder via http://xxx.xxx.xx.xx/mysite.com/public_html/
Flag

- 
mf.digitalocean February 23, 2014
@Kamal, great article but for forward compatibility with Apache 2.4 you should follow @monkeydump's advice and change step 4 to: sudo cp /etc/apache2/sites-available/default /etc/apache2/sites-available/example.com.conf @monkeydump's rightly points out that if you use Debian Jessie (the upcoming release) you'll get Apache 2.4 which requires ".conf" — Debian Wheezy (the current stable) uses Apache 2.2 which doesn't require ".conf" but still accepts it.
Flag

-
ricardo.parraga March 7, 2014
As a best practice to debug separately change or add to the file **etc/apache2/sites-available/example.com**:

```
ErrorLog ${APACHE_LOG_DIR}/example.com.error.log

CustomLog ${APACHE_LOG_DIR}/example.com.access.log
```

Flag

-
franz April 30, 2014
How can I restrict access? sudo nano /var/www/example.com/public_html/index.html - It seems like when I browser using IP 1.0.0.0/example.com I'm able to look trough all the files. I would like to restrict public access to public_html.
Flag

-
astarr April 30, 2014
@franz: What do you have for "DocumentRoot" in /etc/apache2/sites-available/example.com
Flag

-

stenar May 14, 2014
What IP number are you supposed to put in place of: 12.34.56.789 example.com
Flag