How To Set Up Apache Virtual Hosts on Ubuntu 16.04

Posted April 21, 2016 573.1k APACHE UBUNTU 16.04

By: Brennen Bearnes

Introduction

The Apache web server is the most popular way of serving web content on the internet. It accounts for more than half of all active websites on the internet and is extremely powerful and flexible.

Apache breaks its functionality and components into individual units that can be customized and configured independently. The basic unit that describes an individual site or domain is called a `virtual host`.

These designations allow the administrator to use one server to host multiple domains or sites off of a single interface or IP by using a matching mechanism. This is relevant to anyone looking to host more than one site off of a single VPS.

Each domain that is configured will direct the visitor to a specific directory holding that site's information, never indicating that the same server is also responsible for other sites. This scheme is expandable without any software limit as long as your server can handle the load.

In this guide, we will walk you through how to set up Apache virtual hosts on an Ubuntu 16.04 VPS. During this process, you'll learn how to serve different content to different visitors depending on which domains they are requesting.

Prerequisites

Before you begin this tutorial, you should create a non-root user as described in steps 1-4 here.

You will also need to have Apache installed in order to work through these steps. If you haven't already done so, you can get Apache installed on your server through `apt-get`:

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

After these steps are complete, we can get started.

For the purposes of this guide, our configuration will make a virtual host for `example.com` and another for `test.com`. These will be referenced throughout the guide, but you should substitute your own domains or values while following along.

To learn how to set up your domain names with DigitalOcean, follow this link. If you do *not* have domains available to play with, you can use dummy values.

We will show how to edit your local hosts file later on to test the configuration if you are using dummy values. This will allow you to test your configuration from your home computer, even though your content won't be available through the domain name to other visitors.

Step One — Create the Directory Structure

The first step that we are going to take is to make a directory structure that will hold the site data that we will be serving to visitors.

Our `document root` (the top-level directory that Apache looks at to find content to serve) will be set to individual directories under the `/var/www` directory. We will create a directory here for both of the virtual hosts we plan on making.

Within each of *these* directories, we will create a `public_html` folder that will hold our actual files. This gives us some flexibility in our hosting.

For instance, for our sites, we're going to make our directories like this:

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

The portions in red represent the domain names that we are wanting to serve from our VPS.

Step Two — Grant Permissions

Now we have the directory structure for our files, but they are owned by our root user. If we want our regular user to be able to modify files in our web directories, we can change the ownership by doing this:

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

The `$USER` variable will take the value of the user you are currently logged in as when you press **Enter**. By doing this, our regular user now owns the `public_html` subdirectories where we will be storing our content.

We should also modify our permissions a little bit to ensure that read access is permitted to the general web directory and all of the files and folders it contains so that pages can be served correctly:

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

Your web server should now have the permissions it needs to serve content, and your user should be able to create content within the necessary folders.

Step Three — Create Demo Pages for Each Virtual Host

We have our directory structure in place. Let's create some content to serve.

We're just going for a demonstration, so our pages will be very simple. We're just going to make an `index.html` page for each site.

Let's start with `example.com`. We can open up an `index.html` file in our editor by typing:

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

In this file, create a simple HTML document that indicates the site it is connected to. My file looks like this:

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

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

Save and close the file when you are finished.

We can copy this file to use as the basis for our second site by typing:

```
$ cp /var/www/example.com/public_html/index.html /var/www/test.com/public_html/index.html
```

We can then open the file and modify the relevant pieces of information:

```
$ nano /var/www/test.com/public_html/index.html
```

/var/www/test.com/public_html/index.html

```
<html>
  <head>
    <title>Welcome to Test.com!</title>
  </head>
  <body> <h1>Success!  The test.com virtual host is working!</h1>
  </body>
</html>
```

Save and close this file as well. You now have the pages necessary to test the virtual host configuration.

Step Four — Create New Virtual Host Files

Virtual host files are the files that specify the actual configuration of our virtual hosts and dictate how the Apache web server will respond to various domain requests.

Apache comes with a default virtual host file called `000-default.conf` that we can use as a jumping off point. We are going to copy it over to create a virtual host file for each of our domains.

We will start with one domain, configure it, copy it for our second domain, and then make the few further adjustments needed. The default Ubuntu configuration requires that each virtual host file end in `.conf`.

Create the First Virtual Host File

Start by copying the file for the first domain:

```
$ sudo cp /etc/apache2/sites-available/000-default.conf /etc/apache2/sites-available/example.com.conf
```

Open the new file in your editor with root privileges:

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

The file will look something like this (I've removed the comments here to make the file more approachable):

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

```
<VirtualHost *:80>
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

As you can see, there's not much here. We will customize the items here for our first domain and add some additional directives. This virtual host section matches *any* requests that are made on port 80, the default HTTP port.

First, we need to change the `ServerAdmin` directive to an email that the site administrator can receive emails through.

```
ServerAdmin admin@example.com
```

After this, we need to *add* two directives. The first, called `ServerName`, establishes the base domain that should match for this virtual host definition. This will most likely be your domain. The second, called `ServerAlias`, defines further names that should match as if they were the base name. This is useful for matching hosts you defined, like `www`:

```
ServerName example.com
ServerAlias www.example.com
```

The only other thing we need to change for a basic virtual host file is the location of the document root for this domain. We already created the directory we need, so we just need to alter the `DocumentRoot` directive to reflect the directory we created:

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

In total, our virtualhost file should look like this:

/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>
```

Save and close the file.

Copy First Virtual Host and Customize for Second Domain

Now that we have our first virtual host file established, we can create our second one by copying that file and adjusting it as needed.

Start by copying it:

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

Open the new file with root privileges in your editor:

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

You now need to modify all of the pieces of information to reference your second domain. When you are finished, it may look something like this:

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

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

Save and close the file when you are finished.

Step Five — Enable the New Virtual Host Files

Now that we have created our virtual host files, we must enable them. Apache includes some tools that allow us to do this.

We can use the `a2ensite` tool to enable each of our sites like this:

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

Next, disable the default site defined in `000-default.conf`:

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

When you are finished, you need to restart Apache to make these changes take effect:

```
$ sudo systemctl restart apache2
```

In other documentation, you may also see an example using the `service` command:

```
$ sudo service apache2 restart
```

This command will still work, but it may not give the output you're used to seeing on other systems, since it's now a wrapper around systemd's `systemctl`.

Step Six — Set Up Local Hosts File (Optional)

If you haven't been using actual domain names that you own to test this procedure and have been using some example domains instead, you can at least test the functionality of this process by temporarily modifying the `hosts` file on your local computer.

This will intercept any requests for the domains that you configured and point them to your VPS server, just as the DNS system would do if you were using registered domains. This will only work from your computer though, and is simply useful for testing purposes.

Make sure you are operating on your local computer for these steps and not your VPS server. You will need to know the computer's administrative password or otherwise be a member of the administrative group.

If you are on a Mac or Linux computer, edit your local file with administrative privileges by typing:

```
$ sudo nano /etc/hosts
```

If you are on a Windows machine, you can find instructions on altering your hosts file here.

The details that you need to add are the public IP address of your VPS server followed by the domain you want to use to reach that VPS.

For the domains that I used in this guide, assuming that my VPS IP address is `111.111.111.111`, I could add the following lines to the bottom of my hosts file:

/etc/hosts

```
127.0.0.1   localhost
127.0.1.1   guest-desktop
111.111.111.111 example.com
111.111.111.111 test.com
```

This will direct any requests for `example.com` and `test.com` on our computer and send them to our server at `111.111.111.111`. This is what we want if we are not actually the owners of these domains in order to test our virtual hosts.

Save and close the file.

Step Seven — Test your Results

Now that you have your virtual hosts configured, you can test your setup easily by going to the domains that you configured in your web browser:

```
http://example.com
```

You should see a page that looks like this:

Likewise, if you can visit your second page:

```
http://test.com
```

You will see the file you created for your second site:

If both of these sites work well, you've successfully configured **two** virtual hosts on the same server.

If you adjusted your home computer's hosts file, you may want to delete the lines you added now that you verified that your configuration works. This will prevent your hosts file from being filled with entries that are not actually necessary.

If you need to access this long term, consider purchasing a domain name for each site you need and setting it up to point to your VPS server.

Conclusion

If you followed along, you should now have a single server handling two separate domain names. You can expand this process by following the steps we outlined above to make additional virtual hosts.

There is no software limit on the number of domain names Apache can handle, so feel free to make as many as your server is capable of handling.

By: Brennen Bearnes

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

55 Comments

Leave a comment...

Log In to Comment

kashminder *June 5, 2016*

0 I have configured the virtual host for example.com but still getting apache default page when i load the page.when i ping example.com i get the same IP that I configure in /etc/hosts.Have already disabled default page with a2dissite 000-default.conf. What could be the possible issue?

lane *June 16, 2016*

2 I have the same issue. It only works for me over SSL. I've posted on other sites asking for help fixing the issue.

markIopresto *August 20, 2016*

0 If you are using SSL, you will have to keep the 000-default.conf enabled if you are letting your users visit your site with either http:// or https://. The 000-default.conf file acts as a catch all, and – as long as you have the proper redirect added in your virtual host file – all traffic will default to the https://.

kalfusisagod *December 28, 2016*

1 As @mark3236216557d states, I just left the 000-default.conf enabled. If installing letsencrypt certs, you have to do some additional stuff. First you need to modify the 000-default-le-ssl.conf file (after configuring letsencrypt) to point your document root to the first domain you set up (at least in my case).

```
sudo nano /etc/apache2/sites-available/000-default-le-ssl.conf
```

Then modify your that to

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

Maybe the order of operations would be to set up all you virtual domains, and then do a mass letsencrypt configuration of all your domains like

```
sudo letsencrypt --apache -d example.com -d www.example.com -d test.com -d www.test.com
```

phoenixdeveloper *July 6, 2016*

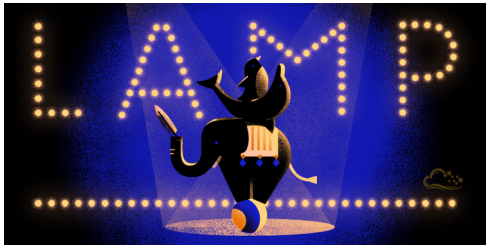
0 Your config must have the ".conf" extension eg /etc/apache2/sites-available/example.com.conf

jacobnelson79 *June 6, 2016*

to get the result, you need Allow incoming traffic for this profile. Please follow this tutorial.

2

Example link



How To Install Linux, Apache, MySQL, PHP (LAMP) stack on Ubuntu 16.04

by Brennen Bearnes

A "LAMP" stack is a group of open source software that is typically installed together to enable a server to host dynamic websites and web apps. This term is actually an acronym which represents the Linux operating system, with the Apache web server. The site data is...

EnochSit *July 23, 2017*

0 The tutorial works for me. Thanks

---

wicky4 *July 17, 2016*

0 **Question 1**

```
[1] DocumentRoot /var/www/test.com/public_html
[2] DocumentRoot /var/www/test_com/public_html
```

which version is more better to use? Is it common practice to use dotted version?

Question 2

Once file `/etc/apache2/sites-available/test.com.conf` is changed, do I have re run `$ sudo a2ensite test.com.conf` or not?

Thanks!

Vardaloupas *July 21, 2016*

0

Hi and thank you for the tutorials.
I follow this instructions but when I try ping in console i receive the following:

64 bytes from sub-08.example.com (xxx.xx.xxx.xx): icmp_seq=8 ttl=64 time=0.059 ms

where do you thing I forget the example.com?

---

kangus *August 21, 2016*

0

Could not get the new virtual server to see it's DocumentRoot, it kept using the servers default yet it appeared to work: We put html docs in the users directories /home/bob/html adding DocumentRoot to each VirtualHost:
<VirtualHost *>
ServerName bob.fg
DocumentRoot /home/bob/html
<Directory /home/bob/html>
allow from all
Options Indexes FollowSymLinks
AllowOverride None
Require all granted

</Directory>
</VirtualHost>
Apache was not getting me to my DocumentRoot. After staring at the conf files I discovered webmin only put <VirtualHost *> instead of <VirtualHost *:80> made the change and it works now.

MTMD *August 25, 2016*

0

Thank you Brennen, this tutorial was perfect!

frunny *September 4, 2016*

0 When I login with SFTP I can't create folders inside the public_html folder but I can upload files or mofify them.

"example.com" folder and the "public_html" folder both have permission 755 and user is my user where I logon.

Any idea why I can't create folders?

SummonD *September 19, 2016*

0 Step 5 I got this error, when I try to use "sudo a2ensite mydomain.com.conf"

perl: warning: Setting locale failed.
perl: warning: Please check that your locale settings:
LANGUAGE = (unset),
LC*ALL = (unset),*
*LC*CTYPE = "UTF-8",
LANG = "en*US.UTF-8"*
are supported and installed on your system.
*perl: warning: Falling back to a fallback locale ("en*US.UTF-8").
Enabling site mydomain.com.
To activate the new configuration, you need to run:
service apache2 reload

georodrigues *September 9, 2018*

0 The same problem to me :(

karkirajan327 *November 18, 2016*

0 I have configured all the step from above.but still when i press "example.com",Not Found

The requested URL / was not found on this server.

---

roccajoseph *December 1, 2016*

0 Be careful, people! If you get a "Not Found" / "The requested URL / was not found on this server." response, then make sure your `DocumentRoot` directory is correct in your `example.com.conf` file.

I accidentally left the `/html/` part in the url that was there by default:

```
DocumentRoot /var/www/html/example.com/public_html
DocumentRoot /var/www/example.com/public_html         <--- fixed
```

If that doesn't fix your "Not Found" problem, try this commant to check your apache logs:

```
sudo cat /var/log/apache2/error.log
```

---

nurfarahin92 *December 8, 2016*

0 hai, i have a quick question. why I unable to ping my virtualhost inside the server and also I unable to curl

---

synyster920123 *December 9, 2016*

Hi. I'm having some issues while creating the virtual host. The file located in sites-available is called repo.local.conf, and it looks like this:

```
<VirtualHost *:80>
    ServerAdmin webmaster@localhost
    ServerName repo.local
    ServerAlias www.repo.local
    DocumentRoot /media/syn/Almacen/Tools/RepoUbuntu/mirror/ubuntu.uci.cu

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

This file was enabled using the procedure shown in this tutorial. The hosts file looks like this:

```
127.0.0.1   localhost
127.0.1.1   syn-hp
127.0.0.1   repo.local

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

I did the "sudo chown -R $USER:$USER RepoUbuntu/" thing from Tools folder, and also gave it 777 permissions.

After restarting apache2 service, I'm getting this message:

```
Forbidden
You don't have permission to access / on this server.
Apache/2.4.18 (Ubuntu) Server at repo.local Port 80
```

Does anyone knows where did i go wrong? Please help :).

---

mayurpandeuk *December 17, 2016*

1

Nice article.

I have one question. Why when changing ownership of the folders, do we set the ownership to;

```
$USER:$USER
```

I thought it should be set to

```
$USER:www-data
```

I have seen that somewhere else, and wasn't too sure what the "www-data" stands for.

Also as I only need to use these virtual machines locally when I open up my hosts file;

```
/etc/hosts
```

I point the domains to my localhost IP address like so;

```
127.0.0.1  example.com
127.0.0.1  test.com
```

This worked well for me.

---

alebak *December 18, 2016*

0

How to use my username and my group instead of www-data user and group? When I going to change these values:

```
export APACHE_RUN_USER=www-data
export APACHE_RUN_GROUP=www-data
```

to

```
export APACHE_RUN_USER=my_user
export APACHE_RUN_GROUP=my_group
```

in the 'envvars' file, doesn't works. Apparently php7 doesn't work correctly if these changes are made.

Better yet: How to use multiple users and groups with permissions differents to www-data using virtualhost?

Thanks for help.

---

metuckness *December 21, 2016*

0 Hi, I have a quick question. I have a domain running under the default /www/html and would like to add another domain, how do you accomplish this without moving the default site or do I have to move the default page when I add a second domain?

KhoiThinh *December 27, 2016*

0 Hi, this doesn't work for me.
When i opened the http://test.com, it directd me to https://www.test.com with a notification: 462 Forbidden Region: Your request for this resource had been blocked. This resource is not available in your region.

DOSarrest Internet Security is a cloud based fully managed DDoS protection service. This request has been blocked by DOSarrest due to the above violation. If you believe you are getting blocked in error please contact the administrator of www.test.com to resolve this issue.

In case of http://example.com, it showed: Example Domain

This domain is established to be used for illustrative examples in documents. You may use this domain in examples without prior coordination or asking for permission.

More information...

Any ideas?

CrypticCube *February 2, 2017*

0 For me www doesn't work only http:// does

---

patrickmay *February 24, 2017*

0 I followed these instructions with two domains that I own. The first domain serves fine when I browse to http://www.domain1.org. The second one always returns the default page from /var/www/html even when I browse to http://www.domain2.com. I ran a2dissite 000-default.conf and restarted

Apache, but get the same behavior.

apache2ctl -S returns this:

AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.1.1. Set the 'ServerName' directive globally to suppress this message
VirtualHost configuration:
*:80 is a NameVirtualHost
default server domain1.org (/etc/apache2/sites-enabled/domain1.org.conf:1)
port 80 namevhost domain1.org (/etc/apache2/sites-enabled/domain1.org.conf:1)
alias www.domain1.org
port 80 namevhost domain2.com (/etc/apache2/sites-enabled/domain2.com.conf:1)
alias www.domain2.com
ServerRoot: "/etc/apache2"
Main DocumentRoot: "/var/www/html"
Main ErrorLog: "/var/log/apache2/error.log"
Mutex default: dir="/var/run/apache2/" mechanism=default
Mutex watchdog-callback: using*defaults*
PidFile: "/var/run/apache2/apache2.pid"
*Define: DUMP*VHOSTS
Define: DUMP*RUN*CFG
User: name="www-data" id=33
Group: name="www-data" id=33

Any idea why this is happening?

---

bbwhited *March 1, 2017*

0 I've follow everything to the letter
sudo a2ensite olmacdoanldsfarm.com /var/www/html already enabled
sudo a2ensite peapatchrvpark.com /var/www/peapatch/ already enabled

sudo a2ensite redpoppyrentalhall.com /var/www/redpoppy/ already enabled
sudo dissite default already disabled
etc/hosts
127.0.0.1 olmacds localhost
192.168.1.29 olmacdonaldsfarm.com
192.168.1.29 peapatchrvpark.com
192.168.1.29 redpoppyrentalhall.com

192.168.1.29 is the address of my server on my private network
65.245.187.152 is the address of my public ip
it serves olmacdonaldsfarm.com just fine
not the other two

ive tried everything I have given up

Load More Comments