Etel Sverdlov
Jun 7, 2012

How To Set Up nginx Virtual Hosts (Server Blocks) on Ubuntu 12.04 LTS

Tags: Nginx
Distribution: Ubuntu

About Virtual Hosts

Virtual Hosts are used to run more than one website or domain off of a single server. Note: according to the nginx website, virtual hosts are called Server Blocks on the nginx. However, for easy comparison with apache, I'll refer to them as virtual hosts in this tutorial.

Set Up

The steps in this tutorial require the user to have root privileges on the virtual private server. You can see how to set that up in the Initial Server Setup Tutorial in steps 3 and 4. Furthermore, if I reference the user in a step, I'll use the name www. You can implement whatever username suits you.

Additionally, you need to have nginx already installed on your VPS. If this is not the case, you can download it with this command:

```
sudo apt-get install nginx
```

Step One— Create a New Directory

The first step in creating a virtual host is to a create a directory where we will keep the new website's information.

This location will be your Document Root in the nginx virtual configuration file later on. By adding a -p to the line of code, the command automatically generates all the parents for the new directory.

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

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.

However, should 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 Six.

Step Two—Grant Permissions

We need to grant ownership of the directory to the right user, instead of just keeping it on the root system. You can replace the "www-data" below with the appropriate username.

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

Additionally, it is important to make sure that everyone is able to read our new files.

```
sudo chmod 755 /var/www
```

Now you are all done with permissions.

Step Three— Create the Page

We need to create a new file called index.html within the directory we made earlier.

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

We can add some text to the file so we will have something to look at when the the site 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 and Exit

Step Four—Create the New Virtual Host File

The next step is to create a new file that will contain all of our virtual host information.

nginx provides us with a layout for this file in the sites-available directory (/etc/nginx/sites-available), and we simply need to copy the text into a new custom file:

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

Step Five—Set Up the Virtual Hosts

Open up the new virtual host file— you will see all the information you need to set up virtual host within.

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

We need to make a couple of changes in these few lines:

```
 server {
        listen   80; ## listen for ipv4; this line is default and implied
        #listen   [::]:80 default ipv6only=on; ## listen for ipv6

        root /var/www/example.com/public_html;
        index index.html index.htm;

        # Make site accessible from http://localhost/
        server_name example.com;
}
```

- Uncomment "listen 80" so that all traffic coming in through that port will be directed toward the site
- Change the root extension to match the directory that we made in Step One. If the document root is incorrect or absent you will not

be able to set up the virtual host.

- Change the server name to your DNS approved domain name or, if you don't have one, you can use your IP address

You do not need to make any other changes to this file. Save and Exit.

The last step is to activate the host by creating a symbolic link between the sites-available directory and the sites-enabled directory. In apache, the command to accomplish this is "a2ensite"—nginx does not have an equivalent shortcut, but it's an easy command nonetheless.

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

To both avoid the "conflicting server name error" and ensure that going to your site displays the correct information, you can delete the default nginx server block:

```
sudo rm /etc/nginx/sites-enabled/default
```

Step Six—Restart nginx

We've made a lot of the changes to the configuration. Restart nginx and make the changes visible.

```
sudo service nginx restart
```

Optional Step Seven—Setting Up the Local Hosts

If you have pointed your domain name to your server's IP address you can skip this step—you do not need to set up local hosts. Your virtual hosts should work. 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, not 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    www.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 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 domain name or ip address into the browser (ie. http://12.34.56.789)

It should look somewhat similar to my handy screenshot

Creating More Virtual Hosts

To add more virtual hosts, you can just 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

Etel Sverdlov

Author:
Etel Sverdlov

Related Tutorials

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

Understanding the Nginx Configuration File Structure and Configuration Contexts

Understanding Nginx Server and Location Block Selection Algorithms

How To Redirect www to Non-www with Nginx on CentOS 7

How To Redirect www to Non-www with Nginx on Ubuntu 14.04

136 Comments

Leave a comment...

Log In to Comment

santiagouy07 *September 28, 2012*

And how do I configure a sub domain ?

♡ Reply

moisey MOD *September 28, 2012*

You would update the server_name entry to use your subdomain instead of your regular domain.com.

♡ Reply

anthony.silva *December 11, 2012*

I've setup several server blocks this way (although, I put them all in one file instead of having separate files per domain). However, in a web browser, I can browse to http:///domain.com/public_html. Is there any way to make it so this is not possible and only http://domain.com works? Or maybe I did something wrong? I tried playing around with permissions - I set /var/www to 700 and /var/www/doman.com/public_html to 755 but that didn't work - I got 403 errors for http://domain.com.

♡ Reply

anthony.silva *December 11, 2012*

That's supposed to be http://{my ip}/domain.com/public_html that I can browse to and don't want to.

♡ Reply

anthony.silva *December 11, 2012*

OK, I think I found a solution. I had a server block entry (the default one) that mapped my ip address to /var/www. I removed this entry and now I can't browse to http://{my ip}/domain.com/public_html. It rewrites the ip address part to the domain.com in the next server block in the file (which I guess has now become the default). So, typing in http://{my ip}/domain.com/public_html rewrites itself to http://domain.com/domain.com/public_html, which produces a 403 error (which I'm ok with, unless someone knows of a more graceful way to handle this - can't really think of what it *should* do with a url like that, other than error out, tho).

Can anyone out there confirm that this sounds like a correct/acceptable solution?

♡ Reply

moisey MOD *December 11, 2012*

There is no need to separate your different domains into separate files if you do not want to.

Each domain that you configure should have its own block:
server { }

The root directive specifies which directory you would like to serve files for:
root /var/www/example.com/public_html;

If you point it to public_html/ there is no need to add a domain.com/public_html when browsing your website.

server_name domain.com
server_name host.domain.com

Are the directives to nginx which specify which domains this server { } block will be active for.

You can also remove the global root directive and test with just the "server" definitions and then ensure that your server blocks have server_name set and a listen directive to specify which IPs and ports that domain should be active on the server for.

If you are still having issues might be a good idea to paste in the specific server block definition which isn't working and we can review that.

♡ Reply

desiredpersona *January 30, 2013*

Delete the default nginx server block to avoid "conflicting server name" error

sudo rm /etc/nginx/sites-enabled/default

♡ Reply

etel MOD *January 30, 2013*

Thanks for the suggestion—I have added that in.

♡ Reply

d4171561d8ad68770b35687ce1033def *February 19, 2013*

Hi,

I've followed this tutorial and it worked for me. But now, accessing example.com/phpmyadmin or 123.45.678.910/phpmyadmin returns a 500 Internal Server Error.

What do I need to do to get phpMyAdmin back up and running?

♡ Reply

d4171561d8ad68770b35687ce1033def *February 19, 2013*

Scratch that, I wasn't adding /public_html at the end of the path for my site's root when creating the symbolic link for phpMyAdmin.

So, if anyone moved their site root as above, to /var/www/example/com/public_html , you obviously need to do this for phpMyAdmin :

sudo ln -s /usr/share/phpmyadmin/ /var/www/example.com/public_html

My question is: is there any way to create a sym link for phpMyAdmin, so that I can access it from any domain I add to my VPS? i.e. access it from example.com/phpmyadmin, example2.com/phpmyadmin and so on.

♡ Reply

swiftgptllc *March 14, 2013*

Glad to see this is working for everyone. I am however, having an issue. Everything is setup correctly, but the second site im running is directing to the first site. In other words, it seems like the root directory set for my second site is being ignored or something. Can anyone please help me?

♡ Reply

swiftgptllc *March 14, 2013*

This is my second site's nginx conf file and yes, it is linked to sites-avaiable and sites-enabled.
http://pastebin.com/2pbYPTNn
Please help me! I've had this issue for the past 2 weeks and i dont know what the problem is

♡ Reply

brad115518 *March 15, 2013*

Note that the symbolic link in "sites-enabled" should use absolute target paths, not relative. Relative links won't get followed to the target and will cause that configuration file to get skipped.

♡ Reply

swiftgptllc *March 15, 2013*

I did use a symbolic link. When i double click the folder it takes me to the correct file in sites-available. Any other issues you think could be the cause of this?

♡ Reply

swiftgptllc *March 15, 2013*

For better reference, this is the first virtual host i created. http://pastebin.com/rGba8hFM
All visits from the second virtual host are directed to the first one for some reason. Any more ideas on this?

♡ Reply

alexandersmith *March 30, 2013*

Instead of creating a symbolic link you can add the file(or directory if you choose) to the nginx.conf.

♡ Reply

joequah1 *April 29, 2013*

Nice guide. I have a few question.
So with the above setup, I just have to provide my router IP address to the domain www.example.com and it will be directed to the correct folder?
To test it locally, what IP address to provide? My router IP too? Or there will be a new IP generated?
I'm still new to hosting.

♡ Reply

joequah1 *April 29, 2013*

To make things clear. I would like to know how things work.
Lets assume I have 2 websites name www.joequah1.com and www.joequah2.com
everything setup as above. 2 different root, same port 80, and server name are joequah1.com and joequah2,com

I just have to provide the same IP addressof mine, assume 123.123.123.123 to both domain.

So when I browse www.joequah1.com , nginx will read the domain name joequah1.com and lead to the correct path www/joequah1.com
and browse www.joequah2.com, nginx will read the domain name to the path www/joequah2.com ?

Is this how it should work?

♡ Reply

robdumas *May 1, 2013*

Here's a useful script for creating virtual hosts: http://www.rosehosting.com/blog/set-up-a-new-virtual-host-server-block-on-your-nginx-server/

♡ Reply

robdumas *May 1, 2013*

And here's a version that's a bit more tailored for use on DO: https://gist.github.com/bitsandbooks/5498838

♡ Reply

Load More Comments