How to Host Multiple Websites on an Ubuntu 18.04 VPS or Dedicated Server

One of the benefits of using Ubuntu 18.04 and Apache is the ability to host multiple websites on a single server. This is very economical because it allows you to use just a single VPS server for all your domains.

This feature is called Virtual hosting. It is simply the aspect of running different domains e.g. example.com and test.com on a single Ubuntu 18.04’s IP address.

Apache’s virtual host directs visitors to different folders where the domains file are located. The client visiting the website will never know if the server is responsible for other virtual hosts.

There are no limits to the number of sites that you can host on your Apache server running Ubuntu 18.04. However, make sure that your server can handle the traffic and disc space.

In this guide, we will walk you through the process of running 2 different sites on a single instance of Ubuntu 18.04 VPS.

Special Note: Consult with Hostadvice's Best Linux Hosting Services page or Best VPS Hosting page to find the top Linux VPS hosting services.

Prerequisites

- Ubuntu 18.04 VPS
- A non-root user with sudo privileges
- Apache web server

In case you don’t have apache installed, you can run the command below on your Ubuntu 18.04 server:

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

Also we will be using example.com and test.com as our dummy domain values and later we will show you how to edit the local hosts file on your computer to test the virtual hosts.

Step 1: Making the file/directory structure

First, we will need to create a directory structure that will host our websites data. Apache has a top-level directory where it looks for websites under the **/var/www** path. We will need to expand this and create sub-directory for our two domains.

To do this, run the commands below on a terminal window:

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

Step 2: Changing directory ownership

The directories we created above are owned by the root user. Therefore we need to change the directory ownership in order to allow the current logged user to modify files. We will use the ***chown*** command to do this with the syntax below

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

Step 3: Modify file permissions

We need to grant read access to the two directories that we created above. This will make the web pages publicly accessible and this means our two websites will be served correctly when requested by a browser.

To do this, we use the command below:

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

Step 4: Create sample web content for each virtual host/domain

Our files and directory structure are now configured correctly. Next, we will create a sample index.html file for each website using the nano editor using the commands below:

test.com

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

Copy paste the text below on the nano editor

```
<html>
     <body>
      This is our test.com website
     </body>
</html>
```

example.com

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

Copy paste the text below on the nano editor

```
<html>
     <body>
      This is our example.com website
     </body>
</html>
```

Remember to close and save each file when you are done editing by pressing **CTR+X** and **Y**.

Step 5: Create the virtual hosts configuration files for our two sites

When Apache is first installed on Ubuntu 18.04 server, it creates a default virtual host file on the path /***etc/apache2/sites-available/000-default.conf***.

We need to copy that file and use it to configure our text.com and example.com virtual hosts. To do this, run the command below

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

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

Apache configuration files must end with a ".config" file extension.

Once you have copied the files, open the first virtual hosts file on a nano editor to edit its content using the command below:

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

Then overwrite the values with the text below:

```
<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>
```

As you can see above, we have referenced the directory ***/var/www/test.com/public_html*** because this is where we will place our test.com website's files.

We need to repeat the same procedure for our example.com virtual host

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

Then overwrite the files with the content below:

```
<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>
```

Step 6: Enable the two virtual hosts

We created two configuration files for our virtual hosts. We now need to enable them using the commands below:

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

Step 7: Restart Apache for the changes to take effect

Once you add a virtual host on your Ubuntu 18.04 server, you will need to restart apache using the command below:

```
$ sudo service apache2 restart
```

Step 8: Edit the local hosts file on your computer

Your virtual hosts should be up and running. However because we used dummy values for testing purposes, we need to edit our local hosts file (on the local computer) and not the VPS server.

This will allow our local computer to resolve to the correct public IP address of our Ubuntu 18.04 server. Assuming your public Ubuntu 18.04 server's IP address is 222.222.222.222, you will have to add this entries on your local computer.

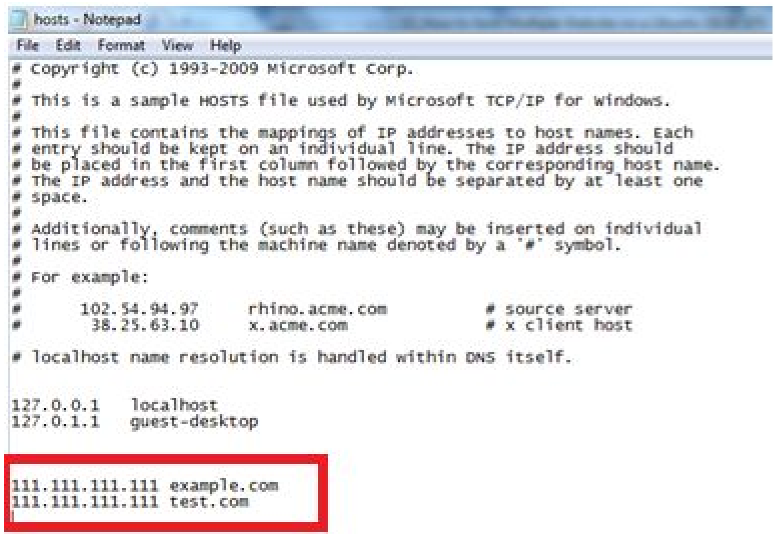
If you are running linux, you need to edit the **/etc/hosts** file using the command below

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

Then add the below entries and save the file.

```
111.111.111.111 example.com
111.111.111.111 test.com
```

If your local computer is running Windows, you need to edit the file **c:\windows\system32\drivers\etc\hosts** using a text editor like notepad and append the two entries above as shown below.



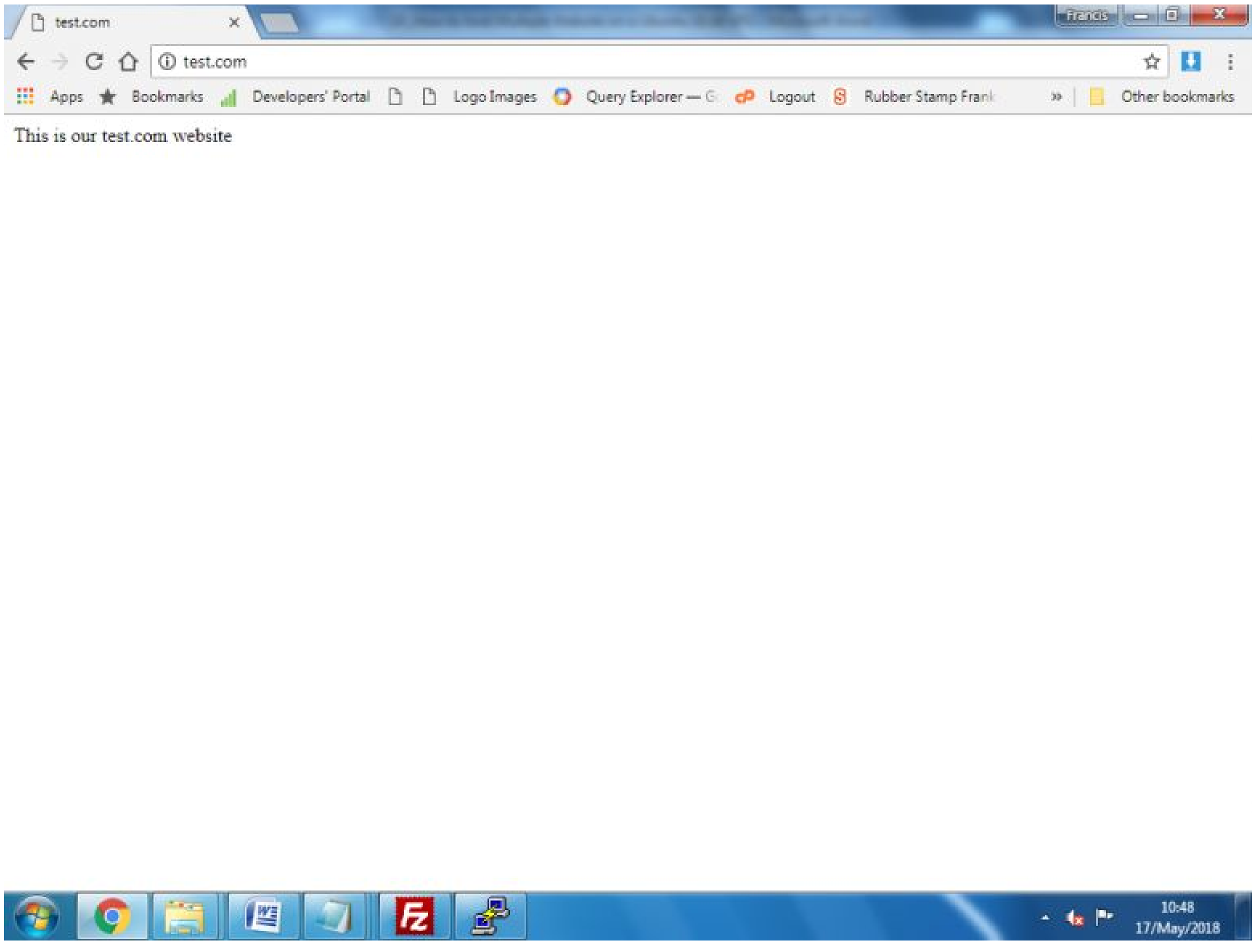
```
# Copyright (c) 1993-2009 Microsoft Corp.
#
# This is a sample HOSTS file used by Microsoft TCP/IP for Windows.
#
# This file contains the mappings of IP addresses to host names. Each
# entry should be kept on an individual line. The IP address should
# be placed in the first column followed by the corresponding host name.
# The IP address and the host name should be separated by at least one
# space.
#
# Additionally, comments (such as these) may be inserted on individual
# lines or following the machine name denoted by a '#' symbol.
#
# For example:
#
#      102.54.94.97     rhino.acme.com          # source server
#       38.25.63.10     x.acme.com              # x client host

# localhost name resolution is handled within DNS itself.


127.0.0.1    localhost
127.0.1.1    guest-desktop


111.111.111.111 example.com
111.111.111.111 test.com
```

Remember to replace 111.111.111.111 with the real public IP address of your server

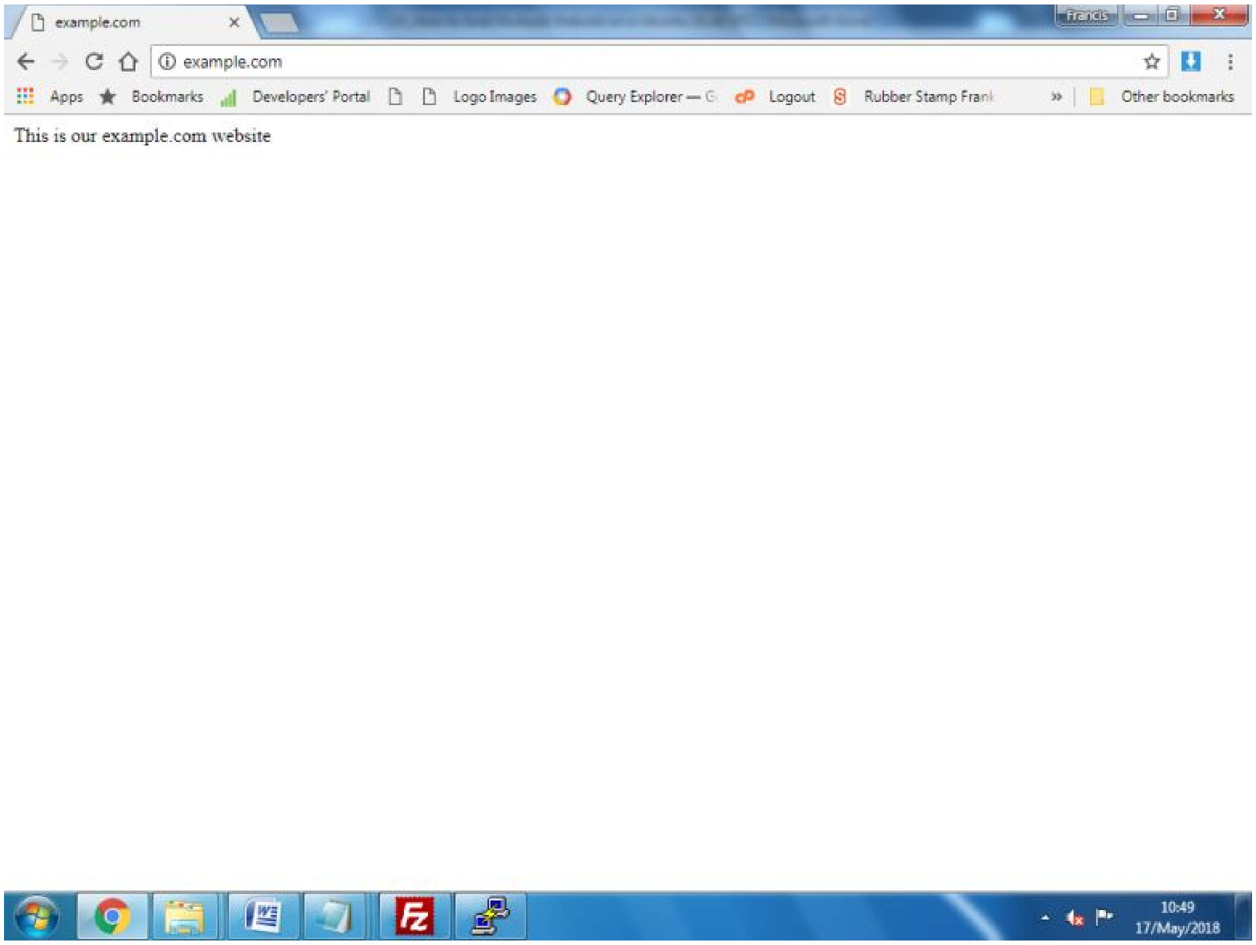
Step 9: Test your virtual hosts on your browser

Finally you need to visit example.com and test.com on your browser and if you followed the steps correctly, you should see the content we created for the virtual hosts as shown below.

Test.com


This is our test.com website

Example.com



This is our example.com website

Conclusion

Hosting multiple websites on a single Ubuntu 18.04 server is that easy. Remember, you can replicate the idea to host unlimited number of virtual hosts. This is very useful if you are running a niche website but you only want to pay subscription fees for a single VPS server.