

# A CentOS 7 Web Development Environment

---

Setting up a CentOS 7 Web Server to run in a Windows based VirtualBox Environment

**By:** Andrew Tuline

**Date:** November 26, 2015

**Version:** 1.1

Revisions.....	6
1. Introduction.....	7
1. Introduction	7
2. Assumptions	7
3. Disclaimer	7
4. Conventions	7
2. The Architecture .....	9
1. Overview	9
2. Environment Overview	11
3. Architectural Diagram	12
4. Architectural Standards	14
3. Installation & Configuration .....	18
1. Oracle VirtualBox	18
2. CentOS 7	18
3. Creating the Virtual Machine	18
4. Installing CentOS As Your Virtual Machine (VM)	20
5. Updating CentOS	21
6. Finishing the Initial Install	21
4. Initial Configuration.....	22
1. Using the Terminal	22
2. VirtualBox Additions & Development Tools (Optional)	22
3. Disable SELINUX	23
4. Network Tools	24

5.	Hosts and Hostname	24
5.	Installing Applications .....	25
1.	Windows Desktop Tools	25
2.	Open SSH Server	26
3.	Remote SSH	27
4.	Hosts File	28
5.	Telnet (Optional)	28
6.	FTP Client & Server	28
7.	Installing Apache	29
8.	Apache Directories & Ownership	30
9.	Configuring Apache	31
10.	Apache Permissions and Ownership	31
11.	Apache Name Based Virtual Hosts	32
12.	Apache Permalinks	35
13.	MySQL (MariaDB)	35
14.	PHP	36
15.	Samba	37
16.	phpMyAdmin	40
17.	Installing a Mail Server	41
18.	Installing Git	43
19.	Take a Snapshot	43
20.	Kick the Tires	43
6.	Workstation Configuration .....	45

1.	Setting up SSH on Hostgator Generic	45
2.	Automating Hostgator SSH Authentication	46
7.	Databases .....	48
1.	Create a Database	48
2.	Dump the Database	49
3.	Drop the Database	50
4.	Drop All Tables	50
5.	Import Database	51
6.	Combined Create, Dump, Drop and Import	51
8.	Backups.....	52
1.	Hostgator Site Backups	52
2.	daily.sh	52
3.	weekly.sh	53
4.	Crontab Configuration	53
5.	Manual Purging of weekly backups	53
9.	Miscellaneous .....	55
1.	What about Git?	55
2.	What about installing WordPress or Drupal?	55
3.	What about Site Migrations?	55
4.	What about .htaccess?	55
10.	Epilogue .....	56
11.	References.....	57



## Revisions

1.0	Re-written from my original Mint installation guide for CentOS 7.
1.1	Finally sorted out correct Apache ownership.

# 1. Introduction

## 1. Introduction

This document is a 'work in progress' that attempts to provide instructions on setting up a text only CentOS 7 server with Apache, MySQL, PHP, Samba, FTP, phpMyAdmin, SSH, Mail and Git support for use with a Windows based workstation running Oracle's VirtualBox 5 environment. In order to encourage use of the Linux command line, this will be a minimal CentOS 7 install with all the applications added via the '**yum**' package manager.

On the public hosting side, discussion revolves around both Hostgator Reseller and VPS accounts. A follow-up document provides information on configuring and using a Git based development environment for local as well as distributed workflow and site migration.

## 2. Assumptions

This document assumes you have a good knowledge of Windows along with web development using FTP, phpMyAdmin and that you have a reasonable knowledge of Linux commands. Google and Digital Ocean have been of immense help in creating this.

## 3. Disclaimer

There are undoubtedly mistakes in this document. I welcome any any constructive suggestions for improvement (you're on your own for support though). Feel free to email suggestions to [atuline@gmail.com](mailto:atuline@gmail.com), and please be as specific as you can. In addition, the examples demonstrated in this document are probably not '**best practice**'. They're just where I'm at in my learning curve.

## 4. Conventions

The '~' symbol refers the user's HOME directory, typically **/home/joeblow** or **~joeblow** or just plain ~.

The prompt for a non-privileged account in the CentOS server will be '**dev\$**'.

The prompt of a root (or privileged) account in CentOS is '**dev#**'.

All code is using the '**Courier New**' font, while normal text is in Times New Roman.

Some instructions include comments, such as:

```
dev$ ls -al
```

```
# Don't type this comment
```

If you need to use a privileged command, it will require the use of **'sudo'**, such as:

```
dev$ sudo yum -y install coolstuff
```

```
. . .
```

```
dev$
```

For permanent privileges, you can type:

```
dev$ su
```

```
# or sudo su
```

```
Password:
```

```
dev# yum -y install coolstuff
```

```
# A privileged command
```

```
dev#
```

In the latter case, you will keep that root (or specified user) privilege until you type:

```
dev# exit
```

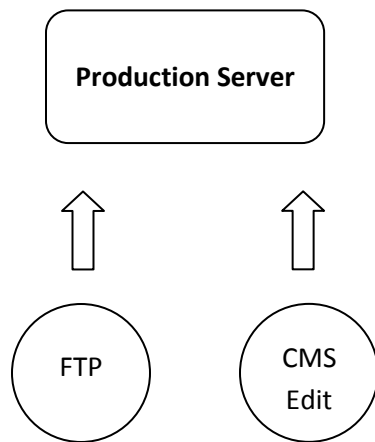
```
dev$
```



## 2. The Architecture

### 1. Overview

Many amateur web developers create sites for clients by writing html and uploading it to the production web site via FTP. In addition, many that use a CMS such as WordPress or Drupal, may even develop the site completely online.



The advantage is that

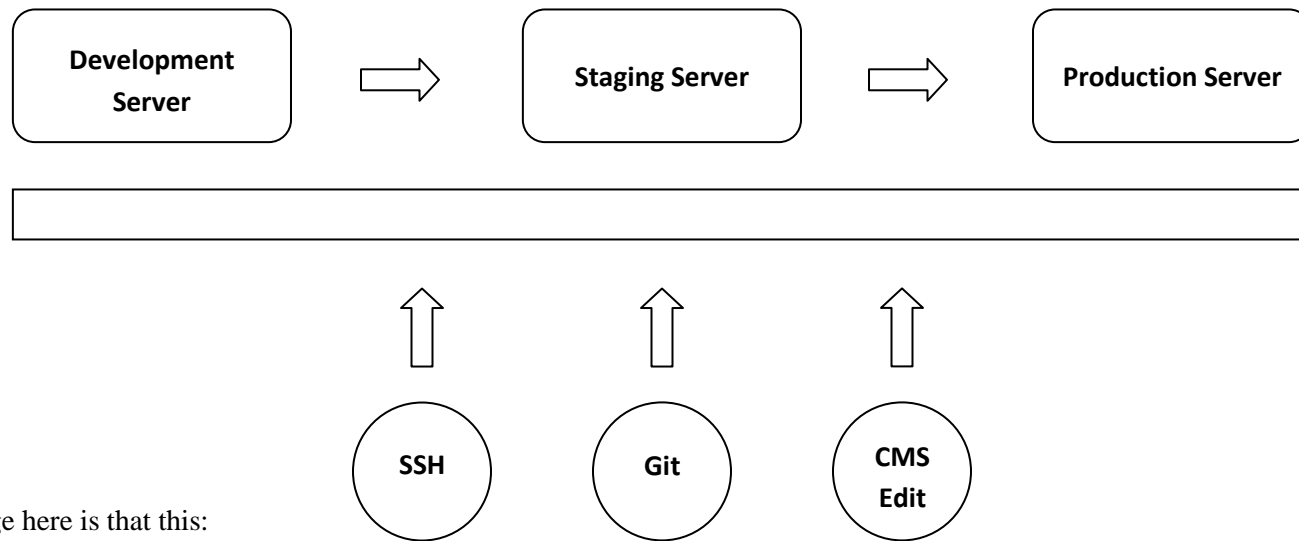
- This is a very quick and simple workflow.

Disadvantages include:

- FTP is not a very secure protocol.
- You may not be able to easily recover from mistakes.
- This does not scale well for teams.

- The web site may be in production before you are paid.
- Your client’s audience may see the site development ‘scaffolding’.

Most professional developers use development and staging servers for site development and, once complete, will migrate the site to the production server. In addition, many developers have moved beyond FTP to more secure technologies, such as SSH, SFTP and RSYNC/SCP along with a version controlled workflow, such as Git.



The advantage here is that this:

- Supports teams.
- Customer only sees completed or staging sites.
- Ensure payment before site goes live.
- Provides significantly better security when transferring files.
- Provides fallback in the event of mistakes during development.
- Encourages a methodology when migrating sites between hosts.

The disadvantage to this process:

- Has a steep learning curve.
- Can be challenge to streamline when migrating a CMS and database to a different server.

## 2. Environment Overview

This is my current development and hosting environment, however yours may vary significantly:

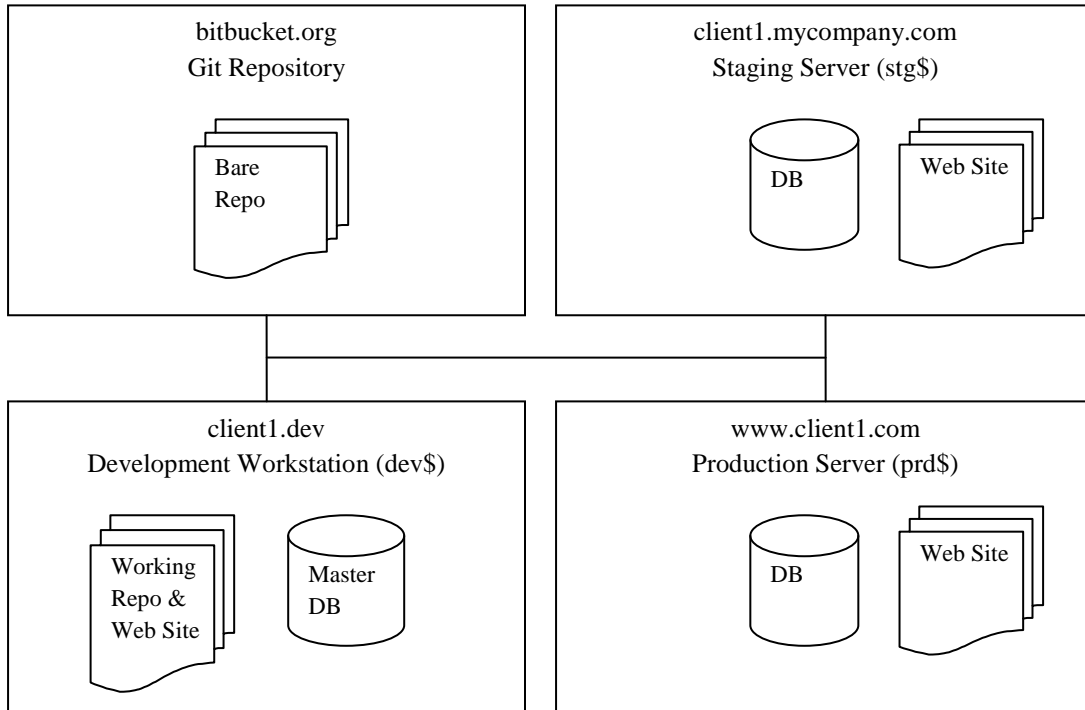
Web Host Provider	<ul style="list-style-type: none"> <li>• Using a Hostgator (or other) VPS account for my consulting company at <b>www.mycompany.com</b>.</li> <li>• Hosts the client staging server at <b>client1.mycompany.com</b> (a subdomain).</li> <li>• Can also be used for the production client web sites.</li> <li>• Prompt will be <b>'stg\$'</b> for the staging site.</li> <li>• Prompt will be <b>'prd\$'</b> for the production site.</li> <li>• Prompt will be <b>'host\$'</b> for the mycompany site.</li> </ul>
Git repositories	<ul style="list-style-type: none"> <li>• Use <b>'bare'</b> Git repositories for each client on <b>bitbucket.org</b>.</li> <li>• Use non-bare repository on development workstation.</li> </ul>
Windows Workstation	<ul style="list-style-type: none"> <li>• Use Filezilla for basic FTP or SFTP.</li> <li>• Use Notepad++, Sublime Text for general text editing.</li> <li>• Use Oracle VirtualBox for development virtual machines (VM's).</li> <li>• Can access files on the VirtualBox VM via Samba shares.</li> <li>• Use Git for Windows utility for SSH access to the various servers.</li> </ul>
Development Server	<ul style="list-style-type: none"> <li>• Use Oracle VirtualBox to run the virtual machine.</li> <li>• The VM will be a CentOS 7 server (minimal install).</li> <li>• Use vi or nano for text editing.</li> <li>• Configured CentOS with Apache, MySQL and PHP</li> <li>• Also added FTP, Samba, Git, SSH server, phpMyAdmin.</li> <li>• Webhosting root directory is <b>/var/www</b>.</li> <li>• The <b>'httpd'</b> process is under the ownership of <b>'apache'</b>.</li> <li>• The <b>'/var/www'</b> directory structure will be under the ownership of <b>'joeblow'</b>.</li> <li>• Configure Samba to share directories.</li> <li>• Use nano, vi or Sublime Text for code editing natively.</li> <li>• Prompt will be <b>'dev\$'</b> for the development server.</li> </ul>

Honourable Mention	<ul style="list-style-type: none"><li>• <a href="http://www.turnkeylinux.org">www.turnkeylinux.org</a> has many excellent virtual appliances all ready to go.</li><li>• <a href="#">MAMP</a> for Mac based web developers.</li><li>• VMWare Workstation and VMWare Player for alternative workstation VM hosting software.</li></ul> <p>Bare metal server based hosting:</p> <ul style="list-style-type: none"><li>• VMWare ESXi and VSphere.</li><li>• Xen, KVM and Proxmox.</li></ul> <p>Git hosting:</p> <ul style="list-style-type: none"><li>• github.com</li></ul> <p>Public hosting:</p> <ul style="list-style-type: none"><li>• <a href="http://www.linode.com">www.linode.com</a> (manage your own VPS)</li><li>• <a href="http://www.digitalocean.com">www.digitalocean.com</a> (great tutorials as well)</li><li>• A 'host' of others.</li></ul>
--------------------	---

I do not recommend hosting the development site natively in Microsoft Windows, let's say with XAMPP. The things you take for granted when using Linux (or UNIX) will take a lot of extra effort with Windows.

### 3. Architectural Diagram

This is the configuration I use for my single developer environment. If multiple staff members are developing the site, then they would either require database access to my workstation. Otherwise, the master database would have to be hosted on a server available to the team, such as the staging server.



## 4. Architectural Standards

For this workflow, we'll be using the following sites and directory structures.

**Table 1 - Servers**

Server Function	Site Name	Description	Linux Prompt
Central Git repository	bitbucket.org	This will be located in located on a Bitbucket account, which is free for the first 5 users. The client repository is called <b>client1.git</b> .	n/a
My company server	www.mycompany.com	Hosts multiple client staging subdomains.	host\$
Client staging server	client1.mycompany.com	This is a subdomain of mycompany.com at Hostgator (or other provider). Files are SFTP'ed or scp'ed from the development workstation. The database is migrated from the development workstation.	stg\$
Client production server	www.client1.com	The files are FTP'ed or scp'ed from the development workstation. The database is migrated from the development workstation. This could be located at any web host provider.	prd\$
Development workstation	CentOS.dev	Name of the host development workstation. Must be a fqdn.	dev\$
Client site on development workstation	www.client1.dev	This is on the CentOS VM. Git pull/push files with the central Git repository. SCP or SFTP files to the staging or production servers.	dev\$

**Table 2 - My Company Server**

<b>www.mycompany.com</b>	<b>Description</b>
Hosting company	This is preferably a Virtual Private Server so that you have full SSH access to your account.
www directory	My corporate web site is located in <b>/home/mycompany/public_html</b> .
SSH	At Hostgator, SSH is free for the first domain (of a reseller account), and includes any subdomains. It's \$10 each for any additional domains. It's free for all accounts with a VPS.
Prompt	<b>host\$</b>

**Table 3 - Staging Server**

<b>client1.mycompany.com</b>	<b>Description</b>
SSH	At Hostgator, SSH is already enabled for <b>www.mycompany.com</b> , and is inherited for free by any subdomains, such as <b>client1.mycompany.com</b> .
www directory	The web site will be located in <b>/home/mycompany/public_html/client1</b> .
Database	Database is called <b>mycompany_client1</b> .
Prompt	<b>stg\$ # It's the same server as host\$</b>

**Table 4 - Client Server**

<b>www.client1.com</b>	<b>Description (may vary between clients)</b>
SSH	SSH needs to be enabled for <b>www.client1.com</b> .
www directory	The web site is typically located on their web host provider at <b>/home/client1/public_html</b> .
Database	Database is called <b>client1_prd</b> .
Prompt	<b>prd\$</b>

**Table 5 - Development Server**

<b>client1.dev</b>	<b>Description</b>
SSH	Open up the VM Window and use a terminal, or access the server via an SSH terminal such as Git Bash.
www directory	This is hosted on the CentOS VM on my Windows workstation. The main web site for this server is located in <b>/var/www</b> .
Site tools	Shell tools for the client1 site will be located in <b>/var/www/client1.dev</b>
Client1 www directory	<b>/var/www/client1.dev/www</b>
Git repository	<b>/var/www/client1.dev/www/wp-content</b>
Database	Database is called <b>client1_dev</b>
Prompt	<b>dev\$</b>



**Table 6 - Databases**

<b>Site Name</b>	<b>Database name</b>	<b>Description</b>
client1.mycompany.com	mycompany_client1	Staging server database (copy of database).
www.client1.com	client1_prd	Production database (copy of database - until cutover).
client1.dev	client1_dev	Development database (master database in a single developer environment). This may be a different server in a team based environment.

## 3. Installation & Configuration

### 1. Oracle VirtualBox

First you need to download the virtual host software, such as Oracle VirtualBox (or VMWare Player/Workstation). I've been using **VirtualBox 5.x in the Expert mode**.

Download a free copy from [www.virtualbox.org](http://www.virtualbox.org) and install it. You will need at least 4GB of memory on your computer and (significantly) more than 25GB of free disk space for your VM. Not only will you require several gigabytes for your VM, you will also require significant disk space for any snapshots that you take. You can select an alternate drive if you have more than one.

### 2. CentOS 7

I used the 64 bit ISO version available from <https://www.CentOS.org/download/> and selected the Torrent.

From a nearby server, I chose:

[http://mirror.its.sfu.ca/mirror/CentOS/7/isos/x86\\_64/CentOS-7-x86\\_64-DVD-1503-01.torrent](http://mirror.its.sfu.ca/mirror/CentOS/7/isos/x86_64/CentOS-7-x86_64-DVD-1503-01.torrent)

### 3. Creating the Virtual Machine

Once you've downloaded the 64 bit version of CentOS 7, create a '**New host**' with the following attributes:

Attribute	Value
<b>Name</b>	CentOS (or whatever you want to call it).
<b>Type</b>	Linux.
<b>Version</b>	Red Hat (64-bit).
<b>Memory Size</b>	2048 MB.
<b>Hard drive</b>	Create a virtual hard disk now.
<b>Hard drive file type</b>	VDI (VirtualBox Disk Image).
<b>Storage on physical drive</b>	Dynamically allocated.
<b>File location (click the folder)</b>	CentOS.
<b>File size</b>	15 GB (minimum).

Once the machine was created, go into 'Settings' and set as follows:

Attribute	Value
<b>General   Advanced</b>	Shared Clipboard - Bidirectional' Drag'n 'Drop - Bidirectional'
<b>System   Motherboard</b>	Change boot order with 'Hard Disk' at the top followed by 'Optical" Remove 'Floppy"
<b>Storage</b>	Clicked on the CD to select that drive. Clicked on the CD to the far right (with the black arrow) to select a disk file. Choose the downloaded disk image. Click on 'Live CD/DVD'.
<b>Network</b>	Select 'Bridged Adapter" Then elect your ethernet adapter. In my case, it's Intel(R) Ethernet Connection I217-V.

## 4. Installing CentOS As Your Virtual Machine (VM)

At this point, you should be able to press the big Green button to ‘**Start**’ your VM. It should load up the CentOS 7 installation script.

<b>Start your VM</b>	Starts CentOS 7 installation.
<b>Select Default</b>	Select 'Install CentOS 7'.
<b>At 'Welcome to CentOS 7'</b>	Select English (United States) and press 'Continue'.
<b>Language</b>	Select English (United States) and press 'Continue'.
<b>Installation Summary</b>	Several options on this page to choose. Set them below.
<b>Localization</b>	Set date and time for your location.
<b>System</b>	Installation Destination -Ensure that the ATA VBOX hard disk is selected and press 'Done'.
<b>Network &amp; Hostname</b>	Set 'Hostname' as CentOS.dev. Set Ethernet as Connected / On. Then pressed 'Done'. (If you forget this, you will have to look up ' <b>nmtui</b> ' to enable networking later.)
<b>Software Selection</b>	Minimal Install only!
<b>Installation Summary</b>	Once done, press 'Begin Installation',
<b>Configuration</b>	
<b>Select a root password</b>	Do this and create a password.
<b>User creation</b>	Create a user account. Check the box called 'Make this user administrator'. Check the box 'Require a password to use this account'.
<b>Software installs</b>	Once complete, press 'Reboot'. Don't just turn off the VM or the installation will NOT complete.
<b>Upon Reboot</b>	You should see 'localhost login:'.
	Login as the user that you created (and not root).

## 5. Updating CentOS

Once you've logged in, you'll want to update your system prior to using it.

```
dev$ sudo yum -y update
```

That should take about 10 minutes.

## 6. Finishing the Initial Install

At this point it might be a good idea to take a snapshot of your installation by going to the very top menu in Virtual Box and:

- Select '**Machine | Take Snapshot**'.
- Enter a description so you know what you've done.

The VirtualBox Snapshot feature allows you to experiment with your VM, and then return it back to the most recent Snapshot (you must turn your VM off in order to do so).

In general, before patching your OS, **TAKE A SNAPSHOT** first! Be warned though, as snapshots take up a LOT of disk space, so you'd better have it.

**This is your BASE VM!**

## 4. Initial Configuration

### 1. Using the Terminal

If you're not already familiar with Linux commands, you should spend some time learning them. Here's some exercises:

- In the terminal, look around the Linux directory structure with the 'ls' and 'cd' commands. Try 'ls -al'.
- Learn other Linux commands such as 'mv', 'cp', 'pwd', 'rm', 'rmdir', 'mkdir', 'more', 'cat' as well as 'chmod' and 'chown'.
- Take some time to learn about directory and file ownerships and permissions.
- Spend some time learning a text editor, such as 'nano' or 'vi'.

The official documentation for CentOS 7 is located at <https://wiki.CentOS.org/Documentation>

### 2. VirtualBox Additions & Development Tools (Optional)

VirtualBox Additions are used in order to enhance the Linux desktop, including screen re-sizeability and to allow the cursor to move in and out of the VM at will. In my experience, the VirtualBox Additions add-on has been problematic on many fronts:

- It requires the developer tools loaded in order to compile.
- It doesn't work with all distributions.
- It often breaks when you update your OS.
- Sometimes, it often no longer compiles after that OS update.

As a result, I don't recommend installing it with this minimal installation of Centos 7. We can get screen re-sizability and freedom of cursor movement by downloading and using 'Git for Windows'.

If you DO install a graphic interface, here are the steps to install VirtualBox Additions. First off, you'll need to install the dependencies:

```
dev$ sudo yum groupinstall "Development Tools"  
dev$ sudo yum install kernel-devel
```

Mount the VirtualBox Additions CD ISO:

```
dev$ sudo mkdir /media/cdrom
dev$ sudo mount /dev/cdrom /media/cdrom
dev$ sudo ./VBoxLinuxAdditions.run
```

Just remember, if you update your operating system, you may need to re-install the VirtualBox Additions.

### 3. Disable SELINUX

References:

[https://www.CentOS.org/docs/5/html/5.1/Deployment\\_Guide/sec-sel-enable-disable.html](https://www.CentOS.org/docs/5/html/5.1/Deployment_Guide/sec-sel-enable-disable.html)

CentOS 7 comes with Security Enhanced Linux additions enabled. This is an important feature for production hosts, but can be challenging for junior administrators attempting to get a basic LAMP server operational. Let's disable that for now (but learn about it at some other time):

```
dev$ sudo vi /etc/sysconfig/selinux
```

Change:

```
SELINUX=enforcing
```

```
to:
```

```
SELINUX=disabled
```

**Reboot the server for this to take effect!**

## 4. Network Tools

In order to install/use **'ifconfig'** (for networkinng), we need to install the network tools as follows:

```
dev$ sudo yum -y install net-tools
```

In addition, if you would like to use a fixed IP address on your host, I recommend that you use DHCP reserved addresses on your router instead of a static IP address.

If you use a static IP address, your CentOS server may not work properly if you take your laptop with you to another network.

## 5. Hosts and Hostname

Make sure your hostname is setup correctly as follows:

```
dev$ sudo vi /etc/hosts # edit your hosts file
```

The entry for this host could be something along the lines of:

```
127.0.1.1 centos.dev www.centos.dev
```

Next, we'll edit the **'hostname'** file.

```
dev$ sudo vi /etc/hostname # edit your hostname file
```

This should be a hostname with a domain name, such as:

```
centos.dev # or www.centos.dev
```



## 5. Installing Applications

This section covers installing tools to support your development environment. These include:

Application	Description
SSH Server	Secure terminal and file copying
FTP	File transfers to/from CentOS
Apache	Web server
MySQL	Database server
PHP	Application support for Drupal and WordPress
Samba	Windows file sharing
phpMyAdmin	Database administration
Mail	Basic mail server and client
Git	Version control
Others	wget, unzip, ifconfig, etc.

You will probably want to take a Snapshot of your CentOS host after installing/configuring these.

### 1. Windows Desktop Tools

If you haven't already done so, download and install '**Git for Windows**' onto your Windows workstation. This program includes a re-sizeable shell, along with git, ssh and many other Linux utilities. It's available at:

<https://git-for-windows.github.io/>

## 2. Open SSH Server

SSH is the standard method of remotely accessing a server via the command line. In addition, SSH is used by several utilities such as 'scp', 'rsync' and 'git' to transfer files between hosts securely.

The SSH server and client are already installed by default on this CentOS 7 installation.

If the SSH server is not installed, you could do so with:

```
dev$ sudo yum -y install openssh-server
```

Test it by typing:

```
dev$ ssh localhost # type 'yes' to continue connecting
The authenticity of host 'localhost (:1:1)' can't be established.
ECDSA key fingerprint is ef:ee:55:8a:db:8e:4a:e8:4c:b8:ea:97:d4:89:42:f1. # or similar
Are you sure you want to continue connecting (yes/no)? yes # not just 'y'
Warning: Permanently added 'localhost' (ECDSA) to the list of known hosts.
user@localhost's password: # enter the user password
dev$ # you should be connected
dev$ exit
logout
Connection to localhost closed.
dev$ # now back at original prompt
```

**Note:** The default TCP port for SSH is 22, which you can modify as follows:

```
dev$ sudo vi /etc/ssh/sshd_config
```

The TCP port listing to change is on line 17.

### 3. Remote SSH

From your VirtualBox window, determine the IP address of your server:

```
dev$ ifconfig # ifcong was previously installed
dev$ . . . .
    inet 192.168.1.76 . . .
    . . . .
dev$
```

Once you've determine the IP address of your workstation, open up an SSH client, such as your '**Git for Windows**' client:

```
User@host ~
$ ssh user@192.168.76 # ssh to your VM
The authenticity of host '192.168.1.76 (192.168.1.76)' can't be established.
ECDSA key fingerprint is ef:ee:55:8a:db:8e:4a:e8:4c:b8:ea:97:d4:89:42:f1. # or similar
Are you sure you want to continue connecting (yes/no)? yes # not just 'y'
Warning: Permanently added 'localhost' (ECDSA) to the list of known hosts.
user@192.168.1.76's password: # enter the user password
...
dev$ # you should be connected
dev$ exit
User@host ~
$
```

**Note:** In a later section, we'll demonstrate shared keys so that we can automate logging into a remote host.

At this point, you can minimize your VM and just use your '**Git for Windows**' client to manage your CentOS installation. The advantage with it is that '**Git for Windows**' has a re-sizeable window and the cursor can move around at will and without having the VBoxAdditions installed.

## 4. Hosts File

Let's take the opportunity to update the hosts file on your Windows workstation as follows:

- Right click on your Windows based text editor and open as '**Administrator**'
- From there, browse to 'c:\windows\system32\drivers\etc'
- Edit your '**hosts**' file
- Add the following line:

```
192.168.1.76 centos.dev www.centos.dev # use the ifconfig listed IP address
```

Save the host file and then '**ping**' the hostname from your workstation, and then open a browser to view the CentOS server.

## 5. Telnet (Optional)

Another tool that's occasionally handy is Telnet. It's installed by typing:

```
dev$ sudo yum -y install telnet #Install telnet client
```

## 6. FTP Client & Server

As you already know, FTP allows you to transfer files to/from your host. Although not secure, WordPress requires FTP in order to upload themes and plugins via the dashboard. To install the FTP client on your CentOS workstation, type:

```
dev$ sudo yum -y install ftp #Install FTP client
. . .
dev$ # It's now installed
dev$ ftp
ftp> quit # Enter 'quit' to exit FTP
dev$
```

Installing the FTP server:

```
dev$ sudo yum -y install vsftpd #Install FTP server
```

In order to allow user authenticated ftp, perform the following:

```
dev$ sudo vi /etc/vsftpd/vsftpd.conf
```

Change:

```
anonymous_enable=no
```

Once done, let's restart the service with:

```
dev$ sudo systemctl restart vsftpd
dev$ sudo systemctl enable vsftpd # enable the service at boot time
```

Depending on how you use ftp, you may need to allow it through the firewall with:

```
dev$ sudo firewall-cmd --permanent --add-port=21/tcp # or
dev$ sudo firewall-cmd --permanent --add-service=ftp
dev$ sudo firewall-cmd --reload
```

You should now be able to ftp into your server and upload files to the server. Keep in mind that SFTP is much more secure and uses SSH port 22.

## 7. Installing Apache

In order to install the Apache web server, type:

```
dev$ sudo yum clean all # Optional command to clean your yum cache
dev$ sudo yum -y install httpd # Install Apache
```

This installation:

- Downloads the Apache software.
- Creates the `/var/www` directory structure for the web sites and is owned by root.
- Creates a `/etc/httpd` directory structure for the configuration files.

Let's start our initial Apache installation with:

```
dev$ sudo systemctl start httpd
dev$ sudo systemctl enable httpd
dev$ sudo systemctl status httpd
```

On the localhost, you should now be able to type:

```
dev$ curl localhost
```

You should see several lines of HTML. On the other hand, you will be unable to access the site from another workstation, so let's allow http through the firewall.

There's a couple of methods to allow httpd through the firewall. The first method is to allow the service:

```
dev$ sudo firewall-cmd --permanent --add-service=http # or
dev$ sudo firewall-cmd --permanent --add-port=80/tcp
success
dev$ sudo firewall-cmd --reload # reload the firewall rules
```

Once Apache is installed, test it by typing:

```
dev$ sudo firewall-cmd --zone=public --list-all # to see the services listed
```

## 8. Apache Directories & Ownership

As previously mentioned, the Apache installation creates a `/var/www` directory that's owned by root for the initial site.

Shared web sites, on the other hand, use a `/home/joeblow/public_html` directory structure, with each owned by their respective user name.

The disadvantage is that with username based directory permissions, you will need to **'sudo'** to become that particular user (or root) when you want to edit the site. As a developer, this provides security compartmentalization for your clients, however it requires significantly more effort to setup and maintain manually and may be overkill for a development server.

Since I'm the only developer on my CentOS server and I have several sites, I'll use the default Apache installation directory in **'/var/www'**.

Directory	Where found	Initial Ownership	New Ownership
/var/www	Default installation	root:root	joebow:joebow
/home/public/joebow/public_html	As found in shared hosting	joebow:joebow	joebow:joebow

- Directory ownership will be 2755
- File ownership will be 644

## 9. Configuring Apache

As mentioned earlier, the web site files are located at **'/var/www'** and are currently owned by **'root'**. In addition, the **'httpd'** process is owned by **'apache'**. You'll need to use **'sudo'** to edit any files in this directory structure which could be painful if you have a lot of web sites in there (and you really shouldn't just **'su'** and become root all the time). The method I chose was to change user and group ownership of **'/var/www'** to **'joebow'** and add **'apache'** to the **'joebow'** group.

## 10. Apache Permissions and Ownership

Upon initially installing Apache, a directory called **/var/www** is created with permissions/ownership of:

```
drwxr-xr-x  root  root                                     # Permission of 755
```

In addition, files created in that directory have permissions of:

```
-rw-r--r--  root  root                                     # Permission of 644
```

Next, let's change the user and group ownership to 'joeblow'.

```
dev$ sudo chown -R joeblow:joeblow /var/www
```

The '/var/www' directory already has files in it, so let's do this:

```
dev$ sudo find /var/www -type d -exec chmod 2755 {} \;           # Don't use []  
dev$ sudo find /var/www -type f -exec chmod 0644 {} \;         # set files to this
```

For any given empty directory, we could just:

```
dev$ sudo chmod 2755 /var/www/directory
```

In addition, let's add 'apache' to the 'joeblow' group with:

```
dev$ sudo usermod -a -G joeblow apache
```

We should now be able to:

```
dev$ cd /var/www/html  
dev$ touch index.html
```

## 11. Apache Name Based Virtual Hosts

In order to support multiple hosts on our server, we need to configure Name Based Virtual Hosts. First we'll create a couple of directories to host our virtual host configuration files:

```
dev$ sudo mkdir /etc/httpd/sites-available                 # holding area for config files  
dev$ sudo mkdir /etc/httpd/sites-enabled                  # live sites
```



Next we'll look for those sites in the http configuration file:

```
dev$ sudo vi /etc/httpd/conf/httpd.conf
```

And add this to the end of the file:

```
IncludeOptional sites-enabled/*.conf
```

Next, we'll add the virtualhost configuration files:

```
dev$ sudo vi /etc/httpd/sites-available/test.dev.conf
```

Let's add this to the file (include logs if you wish):

```
<VirtualHost *:80>  
    ServerName test.dev  
    ServerAlias www.test.dev  
    DocumentRoot /var/www/test.dev/www  
    # ErrorLog /var/www/test.dev/error.log  
    # CustomLog /var/www/test.dev/requests.log combined  
</VirtualHost>
```

Let's create a site called 'test1.dev' as well, so add a 'test1.dev.conf'. We'll then need to enable those virtual host files with:

```
dev$ sudo ln -s /etc/httpd/sites-available/test.dev.conf /etc/httpd/sites-enabled/test.dev.conf  
dev$ sudo ln -s /etc/httpd/sites-available/test1.dev.conf /etc/httpd/sites-enabled/test1.dev.conf
```

Next, we'll create the directories for our virtual sites:

```
dev$ mkdir -p /var/www/test.dev/www  
dev$ mkdir -p /var/www/test1.dev/www
```

Now, we need to restart httpd for those changes to take effect:

```
dev$ sudo systemctl restart httpd
```

These should already have appropriate permissions. Let's create a very simple index.html for both:

```
dev$ echo 'test.dev site!' > /var/www/test.dev/www/index.html # Single quotes!  
dev$ echo 'test1.dev site!' > /var/www/test1.dev/www/index.html
```

Finally, we'll add entries to the hosts file:

```
dev$ sudo vi /etc/hosts
```

And add the entries:

```
127.0.1.1 test.dev www.test.dev  
127.0.1.1 test1.dev www.test1.dev
```

If all goes well, you should be able to perform the following:

```
dev$ curl test.dev  
test.dev!  
dev$ curl test1.dev  
test1.dev!
```

This ensures that your server functional, as well as your named virtual hosts.

In order to try this in your Windows based web browser, edit (run as '**Administrator**') your hosts file and add the IP address of your CentOS server and those hostnames to it.

## 12. Apache Permalinks

In order to support user friendly URL's in WordPress, we need to enable '**Permalinks**'. To do so:

```
dev$ sudo vi /etc/httpd/conf/httpd.conf
```

Find:

```
<Directory "/var/www">  
    AllowOverride None
```

Change to:

```
<Directory "/var/www">  
    AllowOverride All
```

Once done, restart Apache with:

```
dev$ sudo systemctl restart httpd
```

## 13. MySQL (MariaDB)

MySQL is the default database used with Drupal and the only database supported by WordPress. We can install it by typing:

```
dev$ sudo yum -y install mariadb-server mariadb  
dev$ sudo systemctl start mariadb                # start the service  
dev$ sudo mysql_secure_installation              # basic configuration  
dev$ sudo systemctl enable mariadb               # enable service on boot
```

You will be asked for your current root password. You won't have one, so just press '**enter**'. Then, follow the prompts and enter the new password.

- Set root password: Y
- New password: <Enter a password, i.e. 'password'>
- Remove anonymous users: Y
- Disallow root login remotely: n (we want to be able to use phpMyAdmin from a web browser)
- Remove test database: Y
- Reload privilege tables: Y

Once complete, you should be able to use MySQL from the command line by typing:

```
dev$ mysql -u root -ppassword # Spaces are important
mysql> exit
```

## 14. PHP

PHP is the scripting language upon which WordPress and Drupal are based. PHP has many extensions, however the focus of this tutorial is web based applications, so we are just going to install PHP to support:

- Apache
- MySQL
- Basic command line

**Warning:** Before proceeding, make sure you know WHICH version of PHP you want to install. If you need to support older web sites, you may require an older version of PHP. I recommend you research and install the version that your web host provider uses.

To install the current version of PHP, type:

```
dev$ sudo yum -y install php php-mysql
```

To see what other options you can support, type:

```
dev$ sudo yum search php-
```

Once done, restart Apache with:

```
dev$ sudo systemctl restart httpd
```

Once complete, you'll want to verify things are working by typing:

```
dev$ vi /var/www/test.dev/www/info.php # you should no longer need to use 'sudo'
```

and add:

```
<?php echo "This is info.php."; ?>
```

or

```
<?php phpinfo(); ?>
```

Save and exit. Now, type:

```
dev$ curl localhost/info.php  
This is info.php![ CentOS www]$
```

## 15. Samba

Although VirtualBox supports **Shared Folders**, I prefer to access CentOS files from my Windows workstation via Samba. It's relatively easy to setup, provides greater functionality and you can fine tune directory/file ownership and permissions. Samba is installed by typing:

```
dev$ sudo yum -y install samba*
```

You need to configure your Samba shares and hostname in order to be able to access your shared directories via your Windows file browser.

With the increasing risk of destructive trojans, I'll configure this share with basic user authentication.

Remember that we have changed the group owner of the `/var/www` directory to be `'joeblow'`.

Create Samba shares by moving the original `/etc/samba/smb.conf` to `smb.conf.orig` and create/edit a new `smb.conf` as follows:

```
dev$ cd /etc/samba
dev$ sudo mv smb.conf smb.conf.orig           # make a copy of the original config file
dev$ sudo vi smb.conf                         # and then add
```

```
[global]
  workgroup = MYCOMPANY                       # your company name
  netbios name = CENTOS
  restrict anonymous = no
  security = user
#  map to guest = bad user
#  dns proxy = No
  read only = no

[root]
  path = /                                    # For the love of root, be careful
  force user = root
  force group = root
  force directory mode = 2755
  force create mode = 644

[www]
  path = /var/www                             # The www directory structure
  force user = joeblow                        # You are now the god of www
  force group = joeblow
  force directory mode = 2755
  force create mode = 644

[joeblow]
  path = /home/joeblow
```

```
force user = joeblow # With proper ownership
force group = joeblow
force directory mode = 2755
force create mode = 644
```

Save and exit. Run "**testparm**" to check that you have not made any errors as follows:

```
dev$ sudo testparm
```

You also need to create the Samba user:

```
dev$ sudo smbpasswd -a joeblow # Don't forget this!!
```

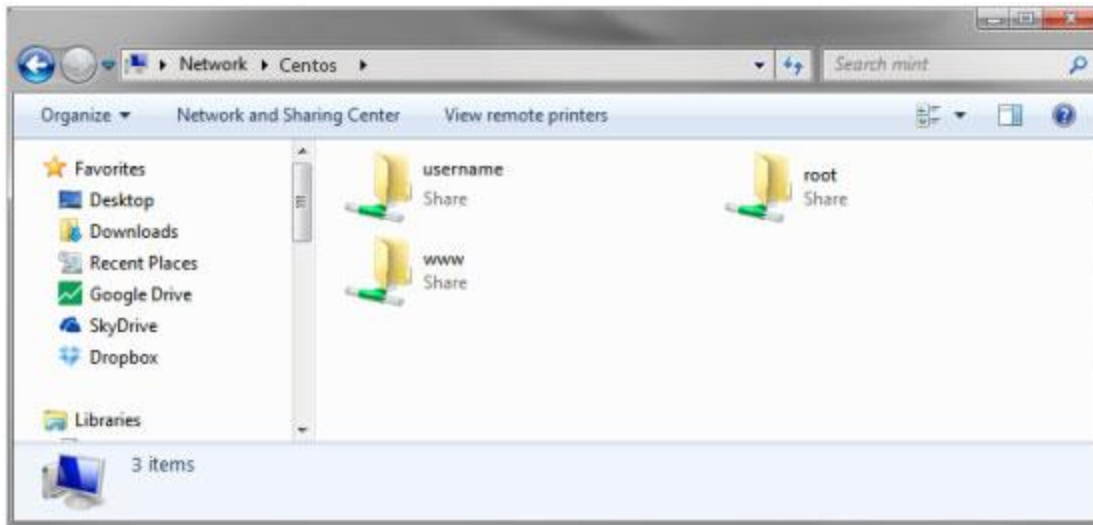
To restart Samba, type:

```
dev$ sudo systemctl enable smb
dev$ sudo systemctl enable nmb
dev$ sudo systemctl restart smb
dev$ sudo systemctl restart nmb # write a script
```

Don't forget to allow the Samba service through the firewall:

```
dev$ sudo firewall-cmd --permanent --add-service=samba
dev$ sudo firewall-cmd --reload
```

It may take some time to propagate, but you should be able to open a 'My Computer' browser (not a web browser) in Windows and type [\\CentOS](#) in the bar and see (once you've authenticated):.



## 16. phpMyAdmin

To download/install phpMyAdmin to manage manage your database, type:

```
dev$ sudo yum -y install epel-release # Extra packages required
dev$ sudo yum -y install phpmyadmin
dev$ sudo vi /etc/httpd/conf.d/phpMyAdmin.conf
```

Beneath '**Require IP ::1**' (about line 18), add the following line:

```
Require all granted #Allows access from Windows workstation
```

Once you've done that, save the file and then restart httpd with:

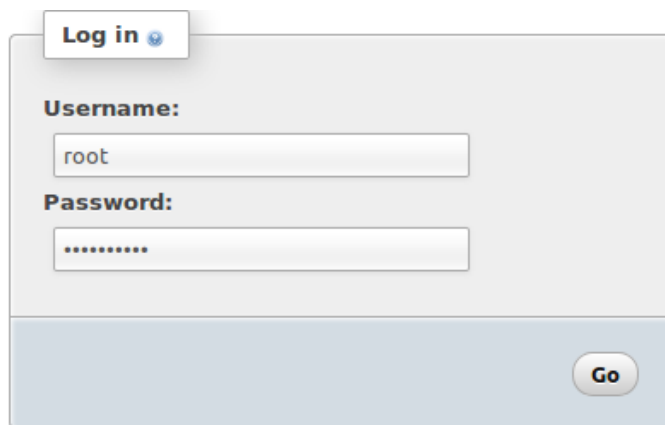
```
dev$ sudo systemctl restart httpd
```



Finally, to log into PhpMyAdmin, open a browser on another workstation and type:

```
http://centos.dev/phpmyadmin
```

Enter 'root' as the Username and then the database password you previously chose.



The screenshot shows a login form for PhpMyAdmin. At the top left is a button labeled "Log in" with a small blue icon. Below this are two input fields. The first is labeled "Username:" and contains the text "root". The second is labeled "Password:" and contains a series of dots ".....". At the bottom right of the form is a button labeled "Go".

**Note:** When creating databases, the modern Collation is utf8mb4\_bin.

## 17. Installing a Mail Server

There are two possibilities for installing a mail server. They are Sendmail and Postfix. Postfix is a Sendmail compatible mail server that is designed to be secure, fast and easy to configure. If it's not already installed, you can install Postfix with:

```
dev$ sudo yum -y install postfix
. . .
dev$ sudo vi /etc/postfix/main.cf
```

Change the following lines to:

```
myhostname = www.centos.dev # or mail.centos.dev
mydomain = centos.dev
myorigin = $mydomain
inet_interfaces = all
inet_protocols = all
mydestination = $myhostname, localhost.$mydomain, localhost, $mydomain
mynetworks = 192.168.1.0/24, 127.0.0.0/8
home_mailbox = Maildir/
```

Once done, let's enable and restart postfix with:

```
dev$ sudo systemctl enable postfix
dev$ sudo systemctl restart postfix
```

You also need the mail client, so let's find it in the yum repositories:

```
dev$ sudo yum whatprovides */mail
. . .
dev$ sudo yum -y install mailx
```

Finally, in order for the mail client to point to the mail directory, add the following entry at the end of your '~/.bashrc' file:

```
MAIL=~/.Maildir
```

Then logout/log in and you should be able to see your mail with:

```
dev$ mail # or mailx
```

I'll leave it to you to figure out how to send yourself some mail via the command line, as well as setting up IMAP but I think Dovecot is the answer for the latter.

Oh, and don't expect to be able to send email to your publicly hosted 'joeblow@mycompany.com' or 'someone@gmail.com' email addresses. That's beyond the scope of this document.

## 18. Installing Git

Git provides our version control, and is installed by typing:

```
dev$ sudo yum -y install git
```

Once you've downloaded Git, you should perform a basic configuration as follows:

```
dev$ git config --global user.name "Joe Blow"  
dev$ git config --global user.email joeblow@mycompany.com  
dev$ git config -l # as in lower case 'L'
```

We'll come back to Git in a major fashion in my 'A Git Development Environment' document.

## 19. Take a Snapshot

# STOP

## Take a Snapshot of your VM!!!

## 20. Kick the Tires

Spend some time learning the tools you've installed:

- You should already be familiar with the Desktop as well as using basic commands in the Terminal.

- You should now be familiar with the basics of **'nano'** or **'vi'** as well as other Linux commands.
- Move files between your Windows workstation and Linux. Check the ownerships & permissions.
- Load up phpMyAdmin (<http://centos.dev/phpmyadmin>). Make a database with the **'utf8mb4\_bin'** collation and delete it.
- Browse to and edit your **/var/www/test.dev/www/index.html** or **index.php** from your host workstation.

## 6. Workstation Configuration

In this section of the document, we'll be configuring our development environment as well as covering a few miscellaneous topics.

### 1. Setting up SSH on Hostgator Generic

Once you have enabled SSH access for your Hostgator account, you can access it from your CentOS workstation or from your **'Git for Windows'** client. From a terminal, you can remotely access your Hostgator Reseller account as follows:

```
dev$ ssh -p 2222 joeblow@www.mycompany.com # Your Hostgator username on your reseller account
```

or to a Hostgator VPS as follows:

```
dev$ ssh joeblow@www.mycompany.com # your Hostgator username on your VPS
```

The first time you SSH to your Hostgator account, you will be prompted with:

```
The authenticity of host '[www.mycompany.com]:2222 ([172.16.55.51]:2222)' can't be established.  
RSA key fingerprint is 67:a9:b4:4e:d5:c3:b3:18:93:67:f8:c4:36:ab:94:68.  
Are you sure you want to continue connecting (yes/no)? yes  
Warning: Permanently added '[www.mycompany.com]:2222, [172.16.55.51]:2222' (RSA) to the list of known  
hosts.
```

This will add (or append to) a file called `~/ssh/known_hosts` on your development workstation.

In order to be able to remove the username and `-p 2222` component of SSH commands in CentOS, you need to create an SSH configuration file. In both Windows and CentOS, you can create a file called `~/ssh/config` which contains:

```
Host www.mycompany.com  
  User joeblow  
  Port 2222 # For Reseller account only
```

SSH simply becomes:

```
$ ssh www.mycompany.com
. . . # You'll still get asked for a password
```

Let's deal with the password in the next section.

## 2. Automating Hostgator SSH Authentication

Everytime you 'ssh' or 'git push' to the Hostgator site, you will be asked for your password as follows:

```
dev$ ssh www.mycompany.com
joeblow@www.mycompany.com's password:
Last login: Fri Feb 1 15:48:24 . . . .
host$
```

If you don't already have one, you can create a **private/public** key pair that can be used to automate the authentication process so that you don't need to enter your password everytime you wish to access Hostgator using the SSH protocol. The public key will be stored as '**~/.ssh/id\_rsa.pub**'.

Here's how to create that key pair on your CentOS workstation:

```
dev$ cd ~
dev$ ssh-keygen # Only do this ONCE!
Generating public/private rsa key pair.
Enter file in which to save the key (/home/joeblow/.ssh/id_rsa):
Enter passphrase (empty for no passphrase): # Keep this blank.
Enter same passphrase again:
Your identification has been saved in /home/joeblow/.ssh/id_rsa.
Your public key has been saved in /home/joeblow/.ssh/id_rsa.pub.
. . .
dev$
```

In order to automate logging into your Hostgator account, we need to add that '~/.ssh/id\_rsa.pub' file to a file on your Hostgator account called '~/.ssh/authorized\_keys'. Assuming you don't already have a .ssh directory in Hostgator, do the following:

```
dev$ ssh www.mycompany.com # ssh to your Hostgator account
. . . # enter your password
host$ mkdir ~/.ssh # create the .ssh directory
host$ exit # to go back to CentOS
dev$ ssh-copy-id -i ~/.ssh/id_rsa.pub joeblow@www.mycompany.com

OR

dev$ cat ~/.ssh/id_rsa.pub | ssh www.mycompany.com "mkdir ~/.ssh; cat >> ~/.ssh/authorized_keys"
```

**Note:** Remember to be consistent as to whether or not you use the 'www' prefix in your host naming.

If you have added the username and port number into the known\_hosts file, you should now be able to SSH into your Hostgator account without requiring either your username or your password:

```
dev$ ssh www.mycompany.com
Last login: Fri Feb 1 1640:26 2013 . . .
host$
```

## 7. Databases

### 1. Create a Database

I assume you can already create a database and import/dump it with phpMyAdmin, however it's a good idea to learn the command line method if you wish to automate some of your database operations, such as performing secure backups.

Here's how to create a database on the CenOS server:

```
dev$ mysql -u root -ppassword # use the MySQL database password
mysql> create database client1_dev;
mysql> grant all on client1_dev.* to 'root@localhost';
mysql> exit
Bye
dev$
```

or create and run a script in vi (or nano):

```
dev$ vi scriptname.sh
```

Add the following:

```
#!/bin/bash # put this at the beginning of all your scripts
mysql -u root -ppassword -e "create database client1_dev";
mysql -u root -ppassword -e "grant all on client1_dev.* to root@localhost";
```

Run it with:

```
dev$ chmod +x scriptname.sh
dev$ ./scriptname.sh
```



## 2. Dump the Database

In order to dump the database on the CentOS server type:

```
dev$ cd /var/www/client1.dev
dev$ mysqldump -u root -ppassword client1_dev > client1_dev.sql
```

List the contents of your `.sql` file as follows:

```
dev$ cat client1_dev.sql | more # press space bar to see each page
or
dev$ more client1_dev.sql # press space bar to see each page
```

To dump the database on your Hostgator based staging server, you need your (and NOT the database) username and password. Then type:

```
dev$ ssh www.mycompany.com # assuming you modified .ssh/config and added the keys
.
.
.
host$ cd public_html/client1
host$ mysqldump -u joeblow -ppassword mycompany_client1 > mycompany_client1.sql
host$ exit
```

You will then need to transfer the file to your development workstation with:

```
dev$ scp -r www.mycompany.com:/home/joeblow/public_html/client1/mycompany_client1.sql
mycompany_client1.sql
```

For security reasons, you should remove any database dump files from the staging (or production) server once the transfer is complete.

You can also dump the staging server database **directly** onto your development workstation with:

```
dev$ ssh mycompany.com mysqldump -u joeblow -ppassword mycompany_client1 > mycompany_client1.sql
```

**Warning:** For a WordPress site, the URL's and directories in the database need to match the host server, and is more involved than a simple string edit. You will need to use a migration tool in order to modify the database for use on different servers and will be discussed/provided in the next document. For further information, do a Google search on '**WordPress migration serialized strings**'.

To copy an entire directory structure with '**scp**', you could use:

```
dev$ scp -r mycompany.com:~/mycompany/public_html/. /var/www/. # No space before '.'
```

### 3. Drop the Database

You can delete (or drop) the database on your CentOS server as follows:

```
dev$ mysqladmin -u root -ppassword drop client1_dev;
```

### 4. Drop All Tables

What about dropping all the tables and then re-importing the database? That's not so straightforward, but it IS possible. The basic command is:

```
dev$ mysqldump -u joeblow -ppassword --add-drop-table --no-data [DATABASE] | grep ^DROP | mysql -u joeblow -ppassword [DATABASE]
```

or in the case of our CentOS workstation with **client1\_dev**:

```
dev$ mysqldump -u root -ppassword --add-drop-table --no-data client1_dev | grep ^DROP | mysql -u root -ppassword client1_dev
```

This works well on the CentOS workstation as well as on Hostgator. The key is to understand WHICH username/password works on which server. Document your sites and don't mix them up.

## 5. Import Database

There have been times where phpMyAdmin times out when attempting to import a database. Here is a quick and easy way to get around that as follows:

```
dev$ mysql -u root -ppassword client1_dev < client1_dev.sql;
```

## 6. Combined Create, Dump, Drop and Import

Once you can use the command line to perform various MySQL operations, the world is your oyster.

## 8. Backups

### 1. Hostgator Site Backups

If you have ever experienced the horror of losing files or a database, then you'll understand the importance of implementing and TESTING your backups and site recovery capabilities. Unless you have a pre-existing arrangement in place with your web host provider, **do not assume** that they adequate backup/recovery protection and response times for your web sites.

I've developed some simple scripts that run on my development CentOS server, which backup my Hostgator sites and associated databases on a daily as well as a weekly basis. Just be cognizant of the amount of bandwidth you use, or you may experience **'Bandwidth Exceeded'**.

In addition, the Open Source **'Bacula'** application performs more comprehensive backups of your entire server.

### 2. daily.sh

Daily backups use names such as **client1\_Monday.sql.gz** and **client1\_Monday.tgz**. This weeks backup will overwrite last week's backup and is then download to the offsite backup server. Don't forget to `chmod +x daily.sh` which contains:

```
# client1
ssh www.client1.com 'tar czf client1_`date +%A`.tgz public_html'
ssh www.client1.com 'mysqldump --opt --user=client1_joeblow --password=xxxxxxx client1_dbname | gzip -v9 - >
client1_`date +%A`.sql.gz'
# scp www.client1.com:client1_`date +%A`.tgz /var/backups/client1
# scp www.client1.com:client1_`date +%A`.sql.gz /var/backups/client1
```

In order to reduce bandwidth, I don't download the daily backups to the CentOS server.

### 3. weekly.sh

These backups use names, such as `client1_2013_09_29.sql.gz` and `client1_2013_09_29.tgz`. They are downloaded from the client server and then deleted from the production server. The `weekly.sh` file contains:

```
# client1
ssh www.client1.com 'tar czf client1_`date +%Y_%m_%d`.tgz public_html'
ssh www.client1.com 'mysqldump --opt --user=client1_joeblow --password=xxxxxxx client1_dbname | gzip -v9 - >
client1_`date +%Y_%m_%d`.sql.gz'
scp www.client1.com:client1_`date +%Y_%m_%d`.tgz /var/backups/client1
scp www.client1.com:client1_`date +%Y_%m_%d`.sql.gz /var/backups/client1
ssh www.client1.com 'rm client1_`date +%Y_%m_%d`.tgz'
ssh www.client1.com 'rm client1_`date +%Y_%m_%d`.sql.gz'
```

Feel free to implement more functionality, such as email notifications, purging, testings, etc.

### 4. Crontab Configuration

Scheduled backups are performed with the use of a crontab and shell scripts. You can create your crontab with:

```
dev$ crontab -e
```

I then added:

```
30 1 * * 6 sh /home/joeblow/weekly.sh > /dev/null 2>&1
30 0 * * * sh /home/joeblow/daily.sh > /dev/null 2>&1
```

### 5. Manual Purging of weekly backups

I created a file in `/var/backups` called `cleanup.sh`. It contains:

```
#!/bin/sh
find /var/backups -maxdepth 2 -name "*.gz" -type f -mtime +120 -print
```

```
find /var/backups -maxdepth 2 -name "*.tgz" -type f -mtime +120 -print
```

```
# find /var/backups -maxdepth 2 -name "*.gz" -type f -mtime +120 -exec rm {} \;  
# find /var/backups -maxdepth 2 -name "*.tgz" -type f -mtime +120 -exec rm {} \;
```

## 9. Miscellaneous

### 1. What about Git?

That'll be covered in the next document.

### 2. What about installing WordPress or Drupal?

That'll be covered in the next document.

### 3. What about Site Migrations?

That will be covered in the next document as well.

### 4. What about .htaccess?

Same.

## 10. **Epilogue**

Writing this document has been a significant learning process, especially with standards, permissions and ownerships. I've changed the architecture several times, but eventually decided on what's been presented. Let's hope it stands the test of time, however I'm always open to changes for the better.



## 11. References

### *CentOS 6 General*

---

- <http://www.servermom.org/complete-newbie-guide-to-build-CentOS-server-to-host-websites/>
- <http://www.servermom.org/basic-CentOS-setup-before-building-a-working-server/414/>
- <http://www.servermom.org/basic-guide-install-mysql-server-on-CentOS/445/>
- <https://www.digitalocean.com/community/tutorials/how-to-install-and-secure-phpmyadmin-on-a-centos-6-4-vps>
- <https://wiki.CentOS.org/HowTos/SELinux>

### *CentOS 7 References*

---

- <https://www.digitalocean.com/community/tutorials/how-to-install-linux-apache-mysql-php-lamp-stack-on-centos-7>
- <http://www.liquidweb.com/kb/how-to-install-apache-on-CentOS-7/>
- <https://www.digitalocean.com/community/tutorials/additional-recommended-steps-for-new-centos-7-servers>
- <https://www.digitalocean.com/community/tutorials/how-to-install-wordpress-on-centos-7>

### General

---

- <https://wiki.centos.org/HowTos>
- <https://www.digitalocean.com/community/tags/?type=tutorials>
- <https://www.digitalocean.com/community/tutorials/how-to-use-systemctl-to-manage-systemd-services-and-units>

### *Backups*

---

- <https://www.digitalocean.com/community/tutorials/how-to-install-bacula-server-on-centos-7>

### *Configuration Management - Puppet*

---

- <https://www.digitalocean.com/community/tutorials/how-to-install-puppet-in-standalone-mode-on-centos-7>

### *Centralized Logging - Log Stash*

---

- [https://www.digitalocean.com/community/tutorial\\_series/centralized-logging-with-logstash-and-kibana-on-centos-7](https://www.digitalocean.com/community/tutorial_series/centralized-logging-with-logstash-and-kibana-on-centos-7)

### *System Monitoring - Nagios*

---

- <https://www.digitalocean.com/community/tutorials/how-to-install-nagios-4-and-monitor-your-servers-on-centos-7>

### *Creating a MediaWiki*

---

- <https://www.mediawiki.org/wiki/Download>

### *Backups - VEEAM*

---

- [http://helpcenter.veeam.com/backup/80/hyperv/install\\_vbr.html](http://helpcenter.veeam.com/backup/80/hyperv/install_vbr.html)

- <http://www.virtualmin.com>

50,000 edits later . . .