6.x WordPress Simplified

Introduction

> There is a better pattern using container technology to segregate accounts by container and mount file systems between containers. I'll write this up this approach if enough people want to use it.

As an application, WordPress is easy to use and feature rich. It has an established community, and in my opinion, the best selection of themes and the most usable blogging cms software package available.

However, though the initial setup seems fast and easy, as an enterprise administrator, it is very insecure. This article provides instructions on setting up a secure WordPress that caters to multiple clients but with one linux administrator called setupadmin.

This approach to setting up WordPress is,

1. Segregates different clients from each other with multiple WordPress instances.
2. Minimize damage if WordPress itself is hacked.
3. Is Easy to backup and recover.
4. Will not be giving customers console access.

WordPress may use an array of technologies. The stack selected for the Bonsai Framework is,

- Web Server = Apache
- Application Server = PHP
- Database = MySQL
- OS = Ubuntu

Initially I tried to make this work with ACL permissions, but the technology does not work as you would expect and not workable.

Install Packages

Install the packages to run WordPress,

For Ubuntu 16

```
sudo apt-get install php # Installs PHP
sudo apt-get install mysql-server # Installs MySQL
sudo apt-get install php-mysql # Libraries to connect PHP to MySQL
sudo apt-get install php-gd # Libraries to allow image editing through browser, "Edit Media"
sudo apt-get install libapache2-mod-php # As of Apache 2.4 php module is no longer installed
sudo apt-get install php7.0-xml # Libraries to allow uploads
```

For Ubuntu 12 and under run the commands listed below

```
sudo apt-get install php5 # Installs PHP
sudo apt-get install mysql-server # Installs MySQL
sudo apt-get install php5-mysql # Libraries to connect PHP to MySQL
sudo apt-get install php5-gd # Libraries to allow image editing through browser, "Edit Media"
```

As of Ubuntu 12 (an maybe even earlier), the installer will automatically restart Apache2 for you.

During the MySQL install, you will be prompted for the **root** administration database password. If following the Bonsai Framework, use your standard password algorithm based on the server name.

Directory and Permission Structure

This approach is more secure than most WordPress default setups. However, it has some limitations around the security, namely administration of this directory structure is restricted to serveradmin.

For the most part, if the WordPress user administer everything through the WordPress web interface this works perfectly well.

If you want to grant ssh access, then wait for my next article which would allow this, but would require an additional layer of security.

Permission and Group Structure

We want website hosting for two different clients, The Daily Planet and LexCorp. Employees from the respective companies will admin through their respective wordpress instances. Here is how the top level structure looks,

Directories required for PHP,

PHP Directory	Ownership	Files and Directory Permissions	Comment
/opt/web/	serveradmin:www-data	serveradmin:www-data rwXr-X---	Main folder for all web related work.
/opt/web/php/	serveradmin:www-data	serveradmin:www-data rwXr-X---	Top level folder for PHP.
/opt/web/php/tmp/	serveradmin:www-data	serveradmin:www-data rwXrwX---	PHP requires write access to this folder for temporary files.
/opt/web/php/logs/	serveradmin:www-data	serveradmin:www-data rwXrwX---	PHP requires write access to this folder for log files.

These directories are for WordPress,

WordPress Directory	Files and Directory Permissions	Comment
/opt/web/php/dailyplanet.com/	serveradmin:www-data rwXr-X---	Top level folder named by domain name.

/opt/web/php/dailyplanet.com/blog/	serveradmin:www-data rwXr-X---	Location of WordPress instance for the domain name. We are using this folder rather than just the domain name to allow for more advanced configuration (such as separating the website with static content).
/opt/web/php/dailyplanet.com/blog/wp-admin/	www-data:serveradmin rwXrwX---	Plugins and custom changes managed through the WordPress interface and requires write access. In addition, WordPress verifies the user it is running as (in this case www-data) is the owner of this directory. So we **must** change the user for this directory. There is a workaround if you prefer not to change the user to www-data.
/opt/web/php/dailyplanet.com/blog/wp-content/	www-data:serveradmin rwXrwX---	Plugins and custom changes managed through the WordPress interface and requires write access. In addition, WordPress verifies the user it is running as (in this case www-data) is the owner of this directory. So we **must** change the user for this directory. There is a workaround if you prefer not to change the user to www-data.

web Folder

This is where everything starts for web related work,

```
cd /opt/
sudo mkdir ./web/

# Next set the permissions.
sudo chown -R serveradmin:www-data ./web/
sudo chmod -R u+rwX,g+r-w+X,o-rwX ./web/
```

Setting permissions early on will not allow you access to do the rest of the instructions unless you belong to the groups

If we wanted to grant basic ssh access to manage static content we could separate out another directory using the Apache directory directive here. That would also require loosening of the web folder to allow other.

php Folders

This is where all php code will execute. In php centric applications this will also be considered the web root for static files too and reflected in the virtual host configuration,

```
cd /opt/web/
sudo mkdir php
sudo chown -R serveradmin:staff ./php/
sudo chmod -R u+rwX,g+r-w+X,o-r-w+X ./php/
```

Next we create the php support directories,

```
cd /opt/web/php/
sudo mkdir ./tmp/ ./logs/
sudo chown -R serveradmin:www-data ./tmp/ ./logs/
sudo chmod -R u+rwX,g+rwX,o-rwx ./tmp/ ./logs/
```

Note the PHP process runs under Apache as the www-data data which belongs to the www-data group. Giving the www-data group write access effectively grants the PHP process the required write access.

Keep in mind that you can not go into the /opt/web/php directory as a normal staff user. At the same time, if you use serveradmin you do not have access sudo.

Make your staff user part of the serveradmin and www-data group,

```
sudo usermod -a -G serveradmin bhitch
sudo usermod -a -G www-data bhitch
exit # If you used your own staff account to modify your staff account, you
need to exit and log back in.
```

Finally logoff and log back in with your staff account for the group changes to take effect.

Configure PHP to Use Specified Folders

Edit php.ini to make use of the folders.
Ubuntu 16

```
sudo vi /etc/php/7.0/cli/php.ini
```

Ubuntu 12

```
sudo vi /etc/php5/apache2/php.ini
```

Set Running Directory

Search for the open_basedir line and modify to include the directories setup for WordPress,

```
; open_basedir, if set, limits all file operations to the defined directory
; and below.  This directive makes most sense if used in a per-directory
; or per-virtualhost web server configuration file. This directive is
; *NOT* affected by whether Safe Mode is turned On or Off.
; http://php.net/open-basedir
open_basedir = /opt/web/php/
```

This helps minimizes the amount of damage that can be done in the event that the system is compromised to the specified directory.

Set Temp Directory

Because **open_basedir** has been set, WordPress no longer has access to the general temporary folder it expects which is required for certain operations (for example to upload plugins through the Administrator web interface).

Modify php.ini further by modifying the upload_tmp_dir line,

```
; Temporary directory for HTTP uploaded files (will use system default if
not
; specified).
; http://php.net/upload-tmp-dir
upload_tmp_dir = /opt/web/php/tmp/
```

Increasing The Upload Limit

The default upload limit is 2mb, the limit must be increased for uploads higher than 2mb or else they will fail when you try to upload.

```
; Maximum allowed size for uploaded files.
; http://php.net/upload-max-filesize
upload_max_filesize = 2M
```

This function displays the max size allowed to upload.without editing this the changes will not take on increasing filesize.

```
; Maximum size of POST data that PHP will accept.
; http://php.net/post-max-size
post_max_size = 2M
```

Make Changes Take Effect

Restart Apache for the changes to take effect,

```
sudo service apache2 restart
```

You will now find that php scripts will only run in the designated directories specified in **php.ini**.

Setup Website Root

Each website will have it's own root folder under /opt/web/php/,

```
sudo mkdir /opt/web/php/dailyplanet.com
sudo chown -R serveradmin:www-data /opt/web/php/dailyplanet.com/
sudo chmod -R u+rwX,g+r-w+X,o-rwX /opt/web/php/dailyplanet.com/
```

Setup Apache Virtual Host

Setup your Apache Virtual Hosts with the website root.

> For Ubuntu 16 php 7 mod needs to be enabled so run command sudo a2enmod php7.0

Setup WordPress

Make sure you are logged in a sudo enabled user. Setup WordPress as follows,

```
sudo su - serveradmin
cd ~
wget http://wordpress.org/latest.tar.gz
tar -xvpf latest.tar.gz
mv ./wordpress/ ./blog/ # Rename as we not need to make the technology
obvious.
```

Finally move WordPress to the proper directory,

```
cd /opt/web/php/dailyplanet.com/
cp -R /home/serveradmin/blog/ ./
exit # To go back to your staff account.
sudo chown -R serveradmin:www-data /opt/web/php/dailyplanet.com/blog/
sudo chmod -R u+rwX,g+r-w+X,o-rwX /opt/web/php/dailyplanet.com/blog/
```

This step is essential for WordPress to be able to install plugins through the web admin interface,

```
sudo chown -R www-data:serveradmin
/opt/web/php/dailyplanet.com/blog/wp-admin/
sudo chown -R www-data:serveradmin
/opt/web/php/dailyplanet.com/blog/wp-content/
sudo chmod -R g+w /opt/web/php/dailyplanet.com/blog/wp-admin/
sudo chmod -R g+w /opt/web/php/dailyplanet.com/blog/wp-content/
```

Configure MySQL

Secure MySQL

As a staff user run the Secure Installation script included with MySQL,

```
sudo mysql_secure_installation
```

The prompts are very straightforward. Except for "Change the Root password?", answer yes to all prompts by hitting Enter,

```
NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MySQL
      SERVERS IN PRODUCTION USE!  PLEASE READ EACH STEP CAREFULLY!

In order to log into MySQL to secure it, we'll need the current
password for the root user.  If you've just installed MySQL, and
you haven't set the root password yet, the password will be blank,
so you should just press enter here.
Enter current password for root (enter for none):
OK, successfully used password, moving on...
Setting the root password ensures that nobody can log into the MySQL
root user without the proper authorisation.
You already have a root password set, so you can safely answer 'n'.
Change the root password? [Y/n] n
 ... skipping.
By default, a MySQL installation has an anonymous user, allowing anyone
to log into MySQL without having to have a user account created for
them.  This is intended only for testing, and to make the installation
go a bit smoother.  You should remove them before moving into a
production environment.
Remove anonymous users? [Y/n]
 ... Success!
Normally, root should only be allowed to connect from 'localhost'.  This
ensures that someone cannot guess at the root password from the network.
Disallow root login remotely? [Y/n]
 ... Success!
By default, MySQL comes with a database named 'test' that anyone can
access.  This is also intended only for testing, and should be removed
before moving into a production environment.
Remove test database and access to it? [Y/n]
 - Dropping test database...
 ... Success!
 - Removing privileges on test database...
 ... Success!
Reloading the privilege tables will ensure that all changes made so far
will take effect immediately.
Reload privilege tables now? [Y/n]
 ... Success!
Cleaning up...
All done!  If you've completed all of the above steps, your MySQL
installation should now be secure.
Thanks for using MySQL!
```

For now that's it to securing MySQL.

Connect

Connect into MySQL,

```
mysql -u root -p
```

The password to use is the password set during the MySQL install. If everything goes well you will be in the MySQL shell,

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 36
Server version: 5.5.24-0ubuntu0.12.04.1 (Ubuntu)
Copyright (c) 2000, 2011, Oracle and/or its affiliates. All rights
reserved.
Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.
Type 'help;' or '\h' for help. Type '\c' to clear the current input
statement.
mysql>
```

The remainder of this section happens inside of the mysql shell.

Create the WordPress Database and Accounts in MySQL

List the databases to makes sure what you want to create does not already exists,

```
SHOW DATABASES;
mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
+--------------------+
3 rows in set (0.16 sec)
```

In our case we should have nothing other than the three default databases.

Enter the following MySQL commands,

```
CREATE DATABASE wpdailyplanetdb;
GRANT ALL PRIVILEGES ON wpdailyplanetdb.* TO 'wpdpdbuser'@'localhost'
IDENTIFIED BY 'password';
FLUSH PRIVILEGES;
```

Adjust the variables for your application.

wpdailyplanetdb - Name of the database for the WordPress application instance. We use the domain name of the website.

wpdpdbuser - User account for accessing the database cannot be longer than 16 characters

localhost - Address of the database server. In this example, the database is on the same server so use localhost.

password - Change to password using algorithm based on name of the primary website domain, in this case dailyplanet.

> Database Admins will not like granting all privileges. After the initial setup is done we will restrict to more minimal privileges.

Exit MySQL Shell

Exit the MySQL shell,

```
EXIT
```

Configure WordPress

Create Config File for Database Access

Launch a browser and hit the WordPress setup page for your machine at http://dailyplanet.com/blog/wp-admin/install.php and you will be prompted to create a configuration file.

Click the button, **"Create a Configuration File"**.

The next prompt reminds you of all the critical information you will need.

Note, the message,

> *If for any reason this automatic file creation doesn’t work, don’t worry. All this does is fill in the database information to a configuration file. You may also simply open wp-config-sample.php in a text editor, fill in your information, and save it as wp-config.php.*

The Bonsai Framework takes a high security posture, so the automatic file creation should not work. Click the **"Let's go!"** button.

Enter the required information and click "Submit",

Field	Value	Comment
Database Name	wpdailyplanetdb	The Bonsai Framewok appraoch is to base the user name on the site's primary domain name.
User Name	wpdpdbuser	
Password		This is the application password set during the wpdailyplanetdb database creation step.
Database Host	localhost	Address of the database server. In this example, the database is on the same server so use localhost is used.
Table Prefix	bf_	The Bonsai Framework approach generally does not encourage changing an application's table prefix. However, given the architecture of WordPress and popularity it is recommended to change the prefix to something other than wp_ to make the system less susceptible to attacks.

It is expected that you will receive a message that WordPress can not write the wp-config file and the following prompt will appear on screen,

> Sorry, but I can't write the wp-config.php file.
>
> You can create the wp-config.php manually and paste the following text into it.

Copy the generated wp-config.php file.

Some shortcuts to ensure you get it all fast,

1. Click inside of the text box
2. Use the keyboard combo CTRL-A (to select all)
3. CTRL-C (to copy)

Go to your shell, load your favourite editor and paste the contents of the wp-config.php file,

```
su - serveradmin # If you are not already serveradmin
vi /opt/web/php/dailyplanet.com/blog/wp-config.php
```

Once you have saved the file, go back to your browser and click "Run the install".

Enter Site Information

Finally enter the site information,

Field	Value	Comment
Site Title	dailyplanet	We like to reference our domain name.
Username	tempadmin	You probably do not want to use the default admin for username. WordPress (as of Sep 2012) out of the box, has no facilities to stop dictionary attacks against the administration system. Admin will be the first username guessed by automated attacks. Because the username put here will show up in the default site generated, this will be a temporary administrator account.
Password		As mentioned, WordPress has no facilities to stop dictionary attacks. On top of that, the default setup exposes your administrator account name on the Internet. Choose a **very very long** and **complex** password. (Anyone know of a good site that shows how quickly an entered password would be broken with a dictionary attack, put the link here)
Your E-mail		Whatever email is chosen here, it will not be the final one used by the real administrator account. Keeping in mind that WordPress does not allow duplicate emails, in this example, the administrator will use a personal email and then use a proper email account when the real administrator account is created.
Privacy		This depends on the purpose of your website. Unless this is a private site that should not show up on Google, leave it checked.

Click, "Install WordPress" which should result in a success screen. At this point you are actually done the setup. Do not click **"Log In"**.

If everything went well you will see a "Success!" message.

Customize WordPress

At this point WordPress is already working. There are two urls to take note of,

URL	Area	Purpose
http://www.dailyplanet.com/blog/	Public	You can hit this url right now and see a default working site. This url is where your users will enter.
http://www.dailyplanet.com/blog/wp-admin/	Administration	This url results from clicking the "Log In" button after the WordPress install is complete. It can also be accessed through the Public homepage by click "Log In" located at the bottom right under "META". The Administration area allows the customization and configuration of WordPress. Also, once logged into the administration, if you browse to the public area, you will see additional buttons and options to create posts and edit the website contents.
http://www.dailyplanet.com/	Public	If WordPress is your main website you should configure Apache to redirect to http://www.dailyplanet.com/blog/.

If you have the Install WordPress Success Screen still up, click **"Log In"** will take you to the Word Press Administration url or use the url in the table above.

Minimal Security - Block Login Attacks

WordPress out of the box can be easily broken into with a brute force dictionary attack for the following combined reasons,

1. The administrator account is available on the public portion of the blog as part of the default sample content.
2. Nothing prevents the brute force attack from trying again and again on the WordPress Administration login page.
3. The Administration page well known.

For these reasons it is best to not make the website publicly available until secured or at least, as mentioned choose a very complex password.

For a more secure setup, consider Installing one of these plugins,

Plugin	Description	Notes
Google Authenticator	The Google Authenticator plugin for WordPress gives you two-factor authentication using the Google Authenticator app for Android/iPhone/Blackberry. The very first time your browser visits the website, it will require two-factor. Subsequent visits with the browser will not. The two-factor authentication requirement can be enabled on a per-user basis. You could enable it for your administrator account, but log in as usual with less privileged accounts.	Make sure time is synced with same time servers across the phone and server. For example, my iphone was off by 2 minutes because it was set manually to Toronto. Best thing to do is turn on the 4 minute drift allowance. When setting the password make sure there are no spaces otherwise the barcode will not work.
BAW More Secure Login	Grid Cards	
Login Security Solution	Adds some common defences against brute force attacks.	Most useful feature is that it blocks user for x number of minutes progressively as more attempts are tried. Also blocks by cookie and ip address.
Anti-Spam	Stops the spam comments that show up on your blog promoting other websites while filtering out real comments	Have yet to see a comment from a spammer
bodi0`s Bots visits counter	This logs the bots that visit your site and gives the option to block/unblock by editing the .htaccess file	Its an interesting tool to see what bots visit your site
Limit Login Attempts	This lets you limit login attempts and logs the IP and username of those who get banned	Think of fail2ban in wordpress its a lot easier to edit too than the actual fail2ban plugin
P3 (Plugin Performance Profiler)	This is a plugin that tests page performance and records what plugin is taking the most time and it has 2 versions of scanning automatic and manual.	This is really useful if your blog is taking an unusually long amount of time to load and you have no clue why
WP-Mail-SMTP	If you cannot get Mail() to work in php you can install this plugin to use SMTP instead.	To setup to use Google use: SMTP Host: smtp.gmail.com SMTP Port: 465 Username: example@gmail.com Password: ***

Should have link to how to ssh in to disable plugins if they misbehave.

Set Up Users

The default user created is an administrator and has more privileges than necessary. The very first step is to create users with specific roles provided by WordPress. The roles are outlined below in order of most privileges to least.

Keep in mind that when creating accounts, Wordpress requires unique email addresses.

Role	Description	Sample User Name
Administrator	Administrators have access to all the administration features.	setupadmin
Editor	Editors can publish posts, manage posts as well as manage other people’s posts, etc.	perrywhite
Author	Authors can publish and manage their own posts, and are able to upload files.	clarkkent, loislane
Contributor	Contributors can write and manage their posts but not publish posts or upload media files.	jimmyolsen
Subscriber	Subscribers can read comments/comment/receive newsletters, etc. but cannot create regular site content.	lexluthor

Create your first user,

Field	Value	Comment
Site Title	dailyplanet	We like to reference our domain name.
Username	setupadmin	This will be the real administration account.

Password		As mentioned, WordPress has no facilities to stop dictionary attacks. On top of that, the default setup exposes your administrator account name on the Internet. Choose a **very very long** and **complex** password. (Anyone know of a good site that shows how quickly an entered password would be broken with a dictionary attack, put the link here).
Your E-mail	admin@bonsaiframework.com	If there is more than one administrator, you should have a general support email box that only administrators have access to. This email address will be used for password recovery purposes.

FAQ

Why do some of the php5 installations say to use install libapache2-mod-php5?

The instructions may be old. At least with Ubuntu 12.04 there is no need for this package because it is included with the php5 package.

What is the difference between the php5 and libapache2-mod-php5 packages?

Nothing I can see. It just looks like php5 is an overarching package name.

References

Setup

Ubuntu Server Documentation - https://help.ubuntu.com/12.04/serverguide/php5.html

Security

Has some ok details around suPHP - https://help.ubuntu.com/community/ApacheMySQLPHP#Installing_MYSQL_with_PHP_5

Some good notes on securing PHP from Symantec - http://www.symantec.com/connect/articles/securing-php-step-step

Start some good security practices for WordPress - http://www.howtospoter.com/web-20/wordpress/triple-p-of-total-wordpress-security

Wordpress discussion on permissions, based their recomendations for suPHP the community does not really understand permissions - http://codex.wordpress.org/Changing_File_Permissions

This restricts the php process to specific directories - http://help.godaddy.com/article/1616

At least by looks this looks like it may be a good guide to securing wordpress - http://wpsecure.net/basics/

Wordpress' official article to getting started after setup - http://codex.wordpress.org/First_Steps_With_WordPress

Article which resolved manual upload issue - http://www.charleshooper.net/blog/wordpress-auto-upgrade-and-dumb-permissions/

> Determine if this actually increases security - http://www.suphp.org/Home.html. suPHP and LiteSpeed make the most sense for shared hosting.
>
> This article indicates that suphp is slow as it makes php run as a cgi. Instead a poster recommended using what is available with mod_php - http://serverfault.com/questions/279938/should-i-use-suphp-or-mod-php-for-shared-hosting. Along this thread another poster recommends, http://mpm-itk.sesse.net/ which allows vhosts to be run under different uid and gid.
>
> A great discussion on using permissions, same conclusion I was coming to using www-data group - http://unix.stackexchange.com/questions/30879/what-user-should-apache-and-php-be-running-as-what-permissions-should-var-www
>
> Probably the most complete but also complex solutions is to use ACLs - http://serverfault.com/questions/339948/user-permissions-for-both-apache-and-local-user/357977