**WordPress –Keeping your WordPress Secure - Linda**

Prepared by Rick Yamamoto

Date: 03/12/2017

**Useful Tools for WordPress Administration**:

1. Atom: used for ftp

**Useful Add-Ons and Plug-ins for WordPress Administration**:

1. UpdraftPlus Backup/Restore: Backup WordPress websites

It allows automatic backups of WordPress site and database, and save them to the webserver or cloud storages (google drive, OneDrive, ftp, etc.)

1. WP Updates Notifier: It automatically sends an email notification if there is any update available in your WordPress website.
2. Force Strong Password: It enforces your users to choose a strong password.
3. Force Password Change: It requires users to change their passwords on their first login.
4. WP-File-Permission-Check: Verify that all directory and files of WordPress have proper permission.
5. Akissmet: it filters out spam comments from your WordPress website.
6. Cerber Limit Login Attempts: It limits the number of times a visitor can attempt to log into the admin area.
7. Stop User Enumeration: It will block author’s ID request when permalinks are enabled.
8. Block Bad Queries: it is a firewall that protects your site from http attacks.
9. WP-Ban: it allows you to Ban the unwanted users by their IPs, IP ranges, host names, referrers, and user agents.
10. Exploit Scanner: it allows on demand and scheduled scans of your site, including posts, comments, plugins and other files.

**Backup a WordPress website with UpdraftPlus**: (Security by a Disaster Recovery Plan)

1. Install UpDraftPlus by adding this plug-in
2. In the Plug-ins page, click on “Activate”
3. Then click on” Settings”
4. Click on the “Settings” tab
5. Setup the “Files Backup Schedule”
6. Setup the “Database Backup Schedule”
7. Click on “Save changes”

**Restore a WordPress Website with UpdraftPlus**: (Security by a Disaster Recovery Plan)

(full restore, but partial is also possible)

1. Analyze the damage

Use PhpMyAdmin to review the website database and files. PhpMyAdmin is preinstalled on most servers.

1. Get the most current version of your backup files
2. Set up a temporary maintenance page

3.1) Replace the content of .htaccess with php code to display “Website Under

Maintenance” and allow your public IP to remote access the website

# TEMP MAINTENANCE PAGE

<IfModule mod\_rewrite.c>

RewriteEngine On

RewriteCond %{REMOTE\_ADDR} !^000.000.000.000$

RewriteRule .\* - [R=503,L]

</IfModule>

ErrorDocument 503 "<h1>The site is getting an update</h1><p>Back in

30 minutes!</p>

<IfModule mod\_headers.c>

# 3600 = 60 minutes

# 86400 = 1 day

Header always set Retry-After "86400"

</IfModule>

1. Cleaning up the webserver

4.1) Remove all files from the webserver (for a full restoration) except .htaccess

4.2) Removing all databases:

4.2.1) Using PhpMyAdmin, click on the “Structure” Tab

4.2.2) check all database files

4.2.3) click on “drop”

4.2.4) Click on “Yes”

1. Upload the new files from the backup

5.1) Upload the html files

5.1.1) Download the backup files to a local computer

5.1.2) Upload all the backup files to the webserver using a ftp (I.E. Atom)

5.2) Restore the databases

5.2.1) Download the database backup files to a local computer

5.2.1) Using PhpMyAdmin, click on the “Import” tab

5.2.2) Click on “Choose File”

5.2.3) Select your database backup file

5.2.4) Click on the “Go” button

1. Restore the database from the backup

5.2.1) Download the database backup files to a local computer

5.2.1) Using PhpMyAdmin, click on the “Import” tab

5.2.2) Click on “Choose File”

5.2.3) Select your database backup file

5.2.4) Click on the “Go” button

1. Test thoroughly

7.1) Test the functionally of the webserver from your place and using multiple online proxies.

1. Remove the temporary maintenance page

**Keeping a WordPress Website Up to Date:** (Security by reducing exposure)

Running the most up to date version of WordPress is essential.

Before applying an upgrade or patch, investigate about this new version on forums. If there is not a potential breach, allow at least 2 weeks from release to implement it (allow others to test it first)

There are three main things to keep current with WordPress:

1. **Update WordPress core files**

1.1) Make a backup before updating.

1.2) Review your WordPress control panel to find out its WordPress version and to

Update it.

1.3) Most of the web hosting providers have an option to update on their control

panel. But the update can also be done via ftp.

1.4) Test thoroughly

1. **Update Installed Plugins**

2.1) Make a backup before updating.

2.2) Update all plug-ins from the web hosting control panel.

2.3) Test thoroughly

1. **Update Installed Themes**

3.1) Make a backup before updating.

3.2) Update all plug-ins from the web hosting control panel.

3.3) Test thoroughly

**Users and Roles in WordPress:** (Security by Access restriction)

1. Use the “Least Privileges Policy” for every user. Users should have the minimum access required for their role, and every role should have the minimum functional access (called Capabilities) to fulfill pre-defined tasks.
2. Roles:

2.1) Super Admin

Similar to Administrator but can manage multi-site WordPress sites.

2.1) Administrator:

Install plug-ins, manage settings, manage users, import/export content.

2.2) Editor:

Publish/Edit posts, manage categories, moderate comments, upload files.

2.3) Author:

Publish/Edit posts, upload files, delete posts.

2.4) Contributor:

Edit posts, delete posts.

2.5) Subscriber:

Read posts.

A subscriber is a regular visitor that is registered with your site.

**Choosing Trusted Plug-ins and Themes**: (Security by using Trusted Apps)

1. Choose a trusted plugin:

1.1) Choose trusted plugins that are posted in the WordPress Plug-in Directory

1.2) Check the plugin rating and review

1.3) Check the plugin version and time since the last update

1.4) Check the number of active installs

1.5) Review reference in documentation and forums.

1. Verify plugin compatibility:

2.1) Verify the latest compatible and the minimum required WordPress versions.

1. Verify the theme trustable

3.1) Check for the author

3.2) Check for rating

3.3) Check the version number

1. Check the Theme under the hood before activating it.

**Removing Unused Files/Folders, Plugins and Themes**: (Security by reducing exposure)

1. Check for plugins that are not needed, no longer supported, or obsolete. Remove them.
2. Check for themes that are unused because they may have unsafe code.
3. Check for folders and files in the webserver that are unused. Backup first.

Add the following code to the .htaccess file to secure loose files.

# SECURE LOOSE FILES

# http://m0n.co/04

<IfModule mod\_alias.c>

RedirectMatch 403 (?i)(^#.\*#|~)$

RedirectMatch 403 (?i)/readme\.(html|txt)

RedirectMatch 403 (?i)\.(ds\_store|well-known)

RedirectMatch 403 (?i)/wp-config-sample\.php

RedirectMatch 403 (?i)\.(7z|bz2|com|conf|dist|fla|git|inc|ini|aspx?|bash|bak?|cfg|cgi|dll|exe|hg|ini|jsp|log|mdb|out|sql|svn|swp|tar|rar|rdf|log|old|psd|rar|tar|tgz|save|sh|sql|svn|swo|swp)$

</IfModule>

**Disabling File Editing in the Admin Area or control panel**: (Security by access restriction)

Any administrator can change a plugin or theme code. This is used by attackers after gaining admin privileges.

To avoid this problem, disable file editing for admins.

Add the following line to your wp-config.php file:

define('DISALLOW\_FILE\_EDIT', true);

**Disabling User Registration**:

If user registration is not needed in your website, disable it on the control panel.

An open user registration functionality is often used by spammers, and attackers.

**Protecting the WordPress Configuration File:** (Security by access restriction)

It is the wp-config.php file located in the root directory of the WordPress installation. It contains sensible information in plain text including database credentials. (http://domain.com/wordpress/wp-config.php

Since you can’t encrypt this file, you must restrict access to this file using:

1. Configuring .htaccess

Add the following code to the .htaccess file before any other rule:

# PROTECT WP-CONFIG

<Files wp-config.php>

# Apache < 2.3

<IfModule !mod\_authz\_core.c>

Order allow,deny

Deny from all

Satisfy All

</IfModule>

# Apache >= 2.3

<IfModule mod\_authz\_core.c>

Require all denied

</IfModule>

</Files>

1. Restricting access via file permissions

Using the Control panel file manager or by opening a terminal to the server, change the permission of the files as shown below:

2.1) wp-config.php to 644 (or less)

2.2) .htaccess to 644 (or less)

2.3) WordPress directories to 755

2.4) WordPress files to 644

644: Readable and writeable by the owner of the file and readable by users in the group

owner of that file and readable by everyone else

755: Same as 644 but it has the execute bit set for everyone. The execute bit is needed

to be able to change into the directory.

4 is read

5 is read/execute (execute is always needed for directories)

6 is read/write

7 is read, write, and execute

Optionally, install the “WP-File-Permission-Check” plugin. Then navigate to the tools menu and select file checker. It will can your directories and files for proper permissions and size

**Configure Authentication Keys to harden the session logins**: (Security by using strong keys)

To improve the security of the WordPress user log in process by adding a set of secret keys to the site's configuration file. This will avoid session highjack and access to the admin area.

1. Open the wp-config.php fil with a text editor
2. Scroll down and below “\* Authentication Unique Keys and Salts”, take note of the URL to generate a Unique Key. In my case it is <https://api.wordpress.org/secret-key/1.1/salt/>
3. Access this site using a browser and copy the shown definitions.
4. Paste them to replace the existing definitions on your wp-config.php file.

**Improve Security by Specifying a Unique Database Prefix**: (security through obscurity)

Change the default table prefix to security it against malicious scripts, hacks and spam.

By default, WordPress default all database files with a “wp\_” prefix. These default database names are heavily targeted by bad bots and malicious scripts.

For example, the default file name for the posts database is wp\_posts. If you change it to myname\_post, you reduce its exposure and hide it from bots.

1. Establish a new default prefix during installation: (For new WordPress sites)

The easiest and recommended method of changing the default prefix happens during the installation process, before submitting the installation page.

Before submitting this page, open the WordPress configuration file and scroll down to right here in the database prefix section. Change the “$table\_prefix = “wp\_”;”. Changing this prefix to anything other than wp\_ is going to boost your site security.

Continue with the installation process and submit.

1. Changing the default prefix after installation: (for existing WordPress sites)
2. Made a backup
3. Change your database table prefix in wp-config.php from wp\_ to something more secure, like NEWPREFIX\_.
4. Go to your database (using phpMyAdmin or whatever) and rename all WordPress table prefixes from wp\_ to whatever you specified in your wp-config.php file.
5. Search the options table for any instances of the old prefix.

SELECT \* FROM `NEWPREFIX\_options` WHERE `option\_name` LIKE '%wp\_%'

Rename any options that begin with wp\_ to the new prefix.

1. Search the usermeta for all instances of the old wp\_ prefix.

SELECT \* FROM ‘NEWPREFIX \_usermeta` WHERE `meta\_key` LIKE '%wp\_%'

Rename any entry that begins with the default WordPress table prefix, wp\_.

1. check your site for proper functionality.

Test the Admin, pages, posts, search, and everything else you can think of (or have time for).

**Prevent Directory Listing**: (Security by obscurity)

Disable open directory listings by

1. adding the following code to the .htaccess before any specific rule.

# DISABLE DIRECTORY VIEWS

Options -Indexes

1. Adding a dummy index.html to every directory. Remember to add this files to any new directory.

**Remove Version Numbers:** (Security by obscurity)

This information seems harmless, but it may enable attackers to target security holes in specific versions of WordPress.

WordPress displays its version number in the head section of webpages via the meta generator tag. It also displays its version number in RSS and other types of feeds. And it displays version information in URLs for CSS and JavaScript files.

To remove the version number publication:

1. For at least your active theme (but recommended for each of your themes), add the following code to the bottom of the functions.php file

// remove version from head

remove\_action('wp\_head', 'wp\_generator');

// remove version from rss

add\_filter('the\_generator', '\_\_return\_empty\_string');

// remove version from scripts and styles

function shapeSpace\_remove\_version\_scripts\_styles($src) {

if (strpos($src, 'ver=')) {

$src = remove\_query\_arg('ver', $src);

}

return $src;

}

add\_filter('style\_loader\_src', 'shapeSpace\_remove\_version\_scripts\_styles', 9999);

add\_filter('script\_loader\_src', 'shapeSpace\_remove\_version\_scripts\_styles', 9999);

**Disable the Display of PHP Errors and Warnings**: (Security by obscurity)

This functionality is useful during site development. It is disable by default, but verify that it is actually disable.

When errors are displayed on your site, as they are here, sensitive information about your server and setup are revealed to the public.

1. Open the wp\_config.php file
2. Scroll down, and look for define(“WP\_DEBUG\_DISPLAY”, true);
3. Change it to

define(“WP\_DEBUG, false);

define(“WP\_DEBUG\_DISPLAY”, false);

define(“WP\_DEBUG\_Log”, false);

**Protect Your WordPress Site from Spam Comments**: (Security by Filtering)

NOT FREE

Akismet plugin is included with WordPress by default. But install it if not.

Install and activate the Akissmet plugin from your WordPress admin area, and visit the plugins page.

On the activation process, this plugin will ask for a API key. Get an activation key by creating a free account with Akissmet.

Configure Akissmet to put spam into a Spam folder for review.

For re4view, you can access the Spam folder by clicking on the “Spam” tab in the comment section of your WordPress admin area.

**Prevent Unwanted Access to the WordPress Admin Area by Locking Down the Login Page**:

By default, there is no limit to how many times someone can try to guess your password and gain access to everything.

We want to limit the number of times a visitor can attempt to log in.

Install and activate the “Cerber Limit Login Attempts” plugin.

Configure the maximum numbers of attempts to login and the lock duration.

**Stop Users Enumeration**: (Security by obscurity)

By default, WordPress allows an attacker to get the username of your subscriber and users. This includes the username of the administrator.

For example, you can manually enumerate each user by going to yourWordPressSite.com/?author=x (change x from 1 to 1000)

1. Make your display names different than your usernames.

From the Users pane in the admin area, change all users display names.

This will provide certain obfuscation, but usernames are still shown on the user archive page’s URL.

1. Disable permalinks in the general settings. Permalinks is a very used feature in most sites.

When disabling permalinks is not an option, then

2.1) Install the “Stop User Enumeration” plugin. It will block author’s ID request.

It will forbid the access of permalinks from the URL enumeration technique show

above.

**Implement a Firewall**: (Security by filtering)

This will filter out http requests from bot, scanners, and malicious attacks.

Install and activate the “Block Bad Queries” plugin. it is a firewall that protects your site from http attacks (including exploitations).

No settings are necessary.

**Implement a Ban Policy**:

After installing the firewalls, analyze your site error and activities. Ban the unwanted users accordingly using WP-Ban plugin.

This plugin will display a ban message to banned IP, IP range, host name or referrer URL that tries to visit your blog.

1. Install and activate WP-Ban plugin.
2. Configure by going to “Ban” inside the “Setting” pane of the admin area.
3. Set Ban filters on IPs, IP ranges, host names, referrers, and user agents.

**Detecting hacks**:

Periodically search your files and database for exploits and malicious code.

1. Install and activate the “Exploit Scanner” plugin. It will scan your WordPress site server for malicious code (previously uploaded by an attacker)

Exploit Scanner allows on demand and scheduled scans of your site, including posts, comments, plugins and other files.

1. Configure it by clicking on “Exploit Scanner” on the “Tools” pane inside the admin area.
2. Run the scan
3. This tools will not remove or modify anything. Any found malware have to be removed manually.

**Stop file hotlinking**:

Hotlinking occurs when another site displays images and files from your site, causing bandwidth usage to increase to your server.

To avoid this do:

1. Add the following code to the .htaccess file

# STOP HOTLINKING (METHOD 1)

<IfModule mod\_rewrite.c>

RewriteCond %{HTTP\_REFERER} !^$

RewriteCond %{HTTP\_REFERER} !^http(s)?://([^.]+\.)?example\.com [NC]

RewriteRule \.(gif|jpe?g?|png)$ - [NC,F,L]

</IfModule>

# STOP HOTLINKING (METHOD 2)

<IfModule mod\_rewrite.c>

RewriteCond %{HTTP\_REFERER} !^$

RewriteCond %{HTTP\_REFERER} !^http(s)?://([^.]+\.)?example\.com [NC]

RewriteCond %{REQUEST\_FILENAME} !hotlink.gif [NC]

RewriteRule \.(gif|jpe?g?|png)$ /hotlink.gif [NC,R,L]

</IfModule>

1. Note change the example to your domain name, and add any file extension to the filtering.

**Protect the Installation Page**:

The installation page is no longer used or necessary after installation.

Get rid of this file any of the following options:

1. Delete the “install.php” file from your server after installing WordPress
2. Deny access to the file via .htaccess by adding the following code to your .htaccess file

# SECURE INSTALL PAGE

<Files install.php>

# Apache < 2.3

<IfModule !mod\_authz\_core.c>

Order allow,deny

Deny from all

Satisfy All

</IfModule>

# Apache >= 2.3

<IfModule mod\_authz\_core.c>

Require all denied

</IfModule>

</Files>

1. Replace the file’s content with a custom code.

**Stop Automated No Referrer Spam via .htaccess**:

No Referrer spam happens when spammers target the WordPress comment script directly (via http request) without actually visiting your website like a human being.

This will protect your example.com/wordpress/wp-coment-post.php file to be directly accessed via http.

Add the following code to your .htaccess file:

# BLOCK SPAM

<IfModule mod\_rewrite.c>

RewriteEngine On

RewriteCond %{REQUEST\_METHOD} POST

RewriteCond %{HTTP\_USER\_AGENT} ^$ [OR]

RewriteCond %{HTTP\_REFERER} !example.com [NC]

RewriteCond %{REQUEST\_URI} /wp-comments-post\.php [NC]

RewriteRule .\* - [F,L]

</IfModule>

Note- replace example.com with your domain

**Detect and block bad bots via .htaccess**:

Add the following code to your .htaccess file before any other rule:

# BLOCK BAD BOTS

<IfModule mod\_setenvif.c>

SetEnvIfNoCase User-Agent (archive.org|binlar|casper|checkpriv|choppy|clshttp|cmsworld|diavol|dotbot|extract|feedfinder|flicky|g00g1e|harvest|heritrix|httrack|kmccrew|loader|miner|nikto|nutch|planetwork|postrank|purebot|pycurl|python|seekerspider|siclab|skygrid|sqlmap|sucker|turnit|vikspider|winhttp|xxxyy|youda|zmeu|zune) badbot

SetEnvIfNoCase User-Agent (BadBotUserAgentString) badbot

# Apache < 2.3

<IfModule !mod\_authz\_core.c>

Order Allow,Deny

Allow from all

Deny from env=badbot

</IfModule>

# Apache >= 2.3

<IfModule mod\_authz\_core.c>

<RequireAll>

Require all Granted

Require not env badbot

</RequireAll>

</IfModule>

</IfModule>

**Firewall your site via .htaccess**:

Add the following code to your .htaccess file before any other rule:

# 6G FIREWALL/BLACKLIST

# @ https://perishablepress.com/6g/

# 6G:[QUERY STRINGS]

<IfModule mod\_rewrite.c>

RewriteEngine On

RewriteCond %{QUERY\_STRING} (eval\() [NC,OR]

RewriteCond %{QUERY\_STRING} (127\.0\.0\.1) [NC,OR]

RewriteCond %{QUERY\_STRING} ([a-z0-9]{2000}) [NC,OR]

RewriteCond %{QUERY\_STRING} (javascript:)(.\*)(;) [NC,OR]

RewriteCond %{QUERY\_STRING} (base64\_encode)(.\*)(\() [NC,OR]

RewriteCond %{QUERY\_STRING} (GLOBALS|REQUEST)(=|\[|%) [NC,OR]

RewriteCond %{QUERY\_STRING} (<|%3C)(.\*)script(.\*)(>|%3) [NC,OR]

RewriteCond %{QUERY\_STRING} (\\|\.\.\.|\.\./|~|`|<|>|\|) [NC,OR]

RewriteCond %{QUERY\_STRING} (boot\.ini|etc/passwd|self/environ) [NC,OR]

RewriteCond %{QUERY\_STRING} (thumbs?(\_editor|open)?|tim(thumb)?)\.php [NC,OR]

RewriteCond %{QUERY\_STRING} (\'|\")(.\*)(drop|insert|md5|select|union) [NC]

RewriteRule .\* - [F]

</IfModule>

# 6G:[REQUEST METHOD]

<IfModule mod\_rewrite.c>

RewriteCond %{REQUEST\_METHOD} ^(connect|debug|delete|move|put|trace|track) [NC]

RewriteRule .\* - [F]

</IfModule>

# 6G:[REFERRERS]

<IfModule mod\_rewrite.c>

RewriteCond %{HTTP\_REFERER} ([a-z0-9]{2000}) [NC,OR]

RewriteCond %{HTTP\_REFERER} (semalt.com|todaperfeita) [NC]

RewriteRule .\* - [F]

</IfModule>

# 6G:[REQUEST STRINGS]

<IfModule mod\_alias.c>

RedirectMatch 403 (?i)([a-z0-9]{2000})

RedirectMatch 403 (?i)(https?|ftp|php):/

RedirectMatch 403 (?i)(base64\_encode)(.\*)(\()

RedirectMatch 403 (?i)(=\\\'|=\\%27|/\\\'/?)\.

RedirectMatch 403 (?i)/(\$(\&)?|\\*|\"|\.|,|&|&amp;?)/?$

RedirectMatch 403 (?i)(\{0\}|\(/\(|\.\.\.|\+\+\+|\\\"\\\")

RedirectMatch 403 (?i)(~|`|<|>|:|;|,|%|\\|\s|\{|\}|\[|\]|\|)

RedirectMatch 403 (?i)/(=|\$&|\_mm|cgi-|etc/passwd|muieblack)

RedirectMatch 403 (?i)(&pws=0|\_vti\_|\(null\)|\{\$itemURL\}|echo(.\*)kae|etc/passwd|eval\(|self/environ)

RedirectMatch 403 (?i)\.(aspx?|bash|bak?|cfg|cgi|dll|exe|git|hg|ini|jsp|log|mdb|out|sql|svn|swp|tar|rar|rdf)$

RedirectMatch 403 (?i)/(^$|(wp-)?config|mobiquo|phpinfo|shell|sqlpatch|thumb|thumb\_editor|thumbopen|timthumb|webshell)\.php

</IfModule>

# 6G:[USER AGENTS]

<IfModule mod\_setenvif.c>

SetEnvIfNoCase User-Agent ([a-z0-9]{2000}) bad\_bot

SetEnvIfNoCase User-Agent (archive.org|binlar|casper|checkpriv|choppy|clshttp|cmsworld|diavol|dotbot|extract|feedfinder|flicky|g00g1e|harvest|heritrix|httrack|kmccrew|loader|miner|nikto|nutch|planetwork|postrank|purebot|pycurl|python|seekerspider|siclab|skygrid|sqlmap|sucker|turnit|vikspider|winhttp|xxxyy|youda|zmeu|zune) bad\_bot

# Apache < 2.3

<IfModule !mod\_authz\_core.c>

Order Allow,Deny

Allow from all

Deny from env=bad\_bot

</IfModule>

# Apache >= 2.3

<IfModule mod\_authz\_core.c>

<RequireAll>

Require all Granted

Require not env bad\_bot

</RequireAll>

</IfModule>

</IfModule>

# 6G:[BAD IPS]

<Limit GET HEAD OPTIONS POST PUT>

Order Allow,Deny

Allow from All

# uncomment/edit/repeat next line to block IPs

# Deny from 123.456.789

</Limit>

**Control proxy access**:

Proxy servers can be used by legitimate visitors, but also are used by attackers to disguise their identity. Some sites allow all proxy visits, while others may want to block proxies and require visitors to use their true identity.

To block proxy visits do the following:

1. Add the following code to the .htaccess before any other rule:

# BLOCK PROXY VISITS

# http://m0n.co/02

# http://m0n.co/03

<IfModule mod\_rewrite.c>

RewriteEngine on

RewriteCond %{HTTP:VIA} !^$ [OR]

RewriteCond %{HTTP:FORWARDED} !^$ [OR]

RewriteCond %{HTTP:USERAGENT\_VIA} !^$ [OR]

RewriteCond %{HTTP:X\_FORWARDED\_FOR} !^$ [OR]

RewriteCond %{HTTP:PROXY\_CONNECTION} !^$ [OR]

RewriteCond %{HTTP:XPROXY\_CONNECTION} !^$ [OR]

RewriteCond %{HTTP:HTTP\_PC\_REMOTE\_ADDR} !^$ [OR]

RewriteCond %{HTTP:HTTP\_CLIENT\_IP} !^$

RewriteRule .\* - [F]

</IfModule>

1. Add the following code to the end of the “themes/YourActiveTheme/template-parts/functions.php” file

// block proxy visits @ http://m0n.co/01

function shapeSpace\_block\_proxy\_visits() {

if (@fsockopen($\_SERVER['REMOTE\_ADDR'], 80, $errstr, $errno, 1)) {

die('Proxy access not allowed');

}

}

add\_action('after\_setup\_theme', 'shapeSpace\_block\_proxy\_visits');

**Control access to the admin access**:

Prevent unauthorized access to the admin area.

This will deny access to the admin area for any IP that is not specified in the following code.

1. Add the following code to the .htaccess of the wp-admin directory before any other rule

# SECURE WP-ADMIN

<FilesMatch ".\*">

# Apache < 2.3

<IfModule !mod\_authz\_core.c>

Order Deny,Allow

Deny from all

Allow from 123.456.789.000

</IfModule>

# Apache >= 2.3

<IfModule mod\_authz\_core.c>

Require ip 123.123.123.000

</IfModule>

</FilesMatch>

Note- replace the IP with your IP addresses.

1. Add the same code to the .htaccess of the root directory before any other rule. This will protect the login page.

# SECURE LOGIN PAGE

<Files wp-login.php>

# Apache < 2.3

<IfModule !mod\_authz\_core.c>

Order Deny,Allow

Deny from all

Allow from 123.123.123.000

</IfModule>

# Apache >= 2.3

<IfModule mod\_authz\_core.c>

Require ip 123.123.123.000

</IfModule>

</Files>

Note- replace the IP with your IP addresses.