WordPress Whitepaper v1.2

By BlogSecurity.net

Table of Contents:

Table of Contents: **1**

Introduction **2**

Installing WordPress **2**

Accessing your WordPress tables *2*

Changing your WordPress Table Prefix **3**

Before Installation *3*

Manually Change *4*

WP Prefix Table Changer *5*

Preparing the Blog **6**

Changing your Admin Username *6*

Create a new limited access user *7*

Hardening your WP Install **9**

Restricting wp-content & wp-includes *9*

Restricting wp-admin *9*

Block all except your IP 9

Password Required - .htpasswd *10*

The .htaccess file 10

The .htpasswd file 10

SPAM **11**

Blog Encryption **12**

Key Plugins **13**

Disabling WordPress Errors *13*

Removing the WordPress Version *13*

Security Above and Beyond **14**

WPIDS - Detect Intrusions *14*

WordPress Plugin Tracker – Are you updated? *14*

WordPress Online Security Scanner *15*

The End **15**

Introduction

This paper provides you with all necessary information to improve the security for your blog. We try to describe the steps in an easy, understandable way without to much tech talk, so that you can easily follow them and you don't run into problems with applying these changes to secure your blog. All information can be found on BlogSecurity.net; this paper serves as a compact guide to secure your blog. We will strive to keep this document updated, so check back regularly.

If you have questions, problems, ideas or something else related to this paper, feel free to contact us.

Important: Before using any of the techniques outlined is this paper please perform a full backup of your WP files and Database. See our "5 failsafe steps to upgrade WordPress" for help.

Installing WordPress

Accessing your WordPress tables

Before you even start to install your blog it is important to choose the right type of database user with the right permissions. The idea behind this "vital" step is to provide WordPress with a limited database user. This mitigates the risk data corruption, and also adds an additional layer of security.

Please Note: You may not have root access to MySQL if your blog is hosted. If this is the case, you can skip this step.

First let's login with our "root" MySQL account and create a database that our WordPress install will use:

```
$ mysql -u root
mysql> CREATE database wp;
Query OK, 1 row affected (0.00 sec)
```

Next, we will create a user: this account will be limited in that it will only have access to our WordPress database. Also, we'll ensure that this account can only be used from the local server and not remotely.

```
mysql> GRANT SELECT,INSERT,UPDATE,DELETE,CREATE,DROP
 -> ON wp.*
 -> TO 'wpuser'@'localhost'
 -> IDENTIFIED BY 'strongpassword';
Query OK, 0 rows affected (0.01 sec)
```

Make sure that the user account password is strong. Some versions of WordPress use this password for security elsewhere.

All done, lets prepare the 'wp-config.php' file.

Changing your WordPress Table Prefix

Before Installation

Next, we are going create a ‘wp-config.php’ file in your WordPress root directory. Lets update the information with our newly created database and user account:

```
// ** MySQL settings ** //
define('DB_NAME', 'wp');    // The name of the database
define('DB_USER', 'wpuser');    // Your MySQL username
define('DB_PASSWORD', 'strongpassword'); // ...and password
define('DB_HOST', 'localhost');    // 99% chance you won't need to change this value
define('DB_CHARSET', 'utf8');
define('DB_COLLATE', '');

// Change SECRET_KEY to a unique phrase.  You won't have to remember it later,
// so make it long and complicated.  You can visit https://www.grc.com/passwords.htm
// to get a phrase generated for you, or just make something up.
define(’SECREY_KEY’,'A49D0EA936EFFFE30BAD7BACBA466CC897636F74BAB91128A96C9EF8C25F0
249');
// Change this to a unique phrase.
// You can have multiple installations in one database if you give each a unique prefix

$table_prefix  = 'wp_4i32aK_';   // Only numbers, letters, and underscores please!
```

I hope your enjoy our lovely pastel pinks. So we have created our WP config (above) with our new user account, password and database name information. Also, you will notice we used the “grc.com” link to create a nice long SECRET_KEY. This is from WordPress 2.5 ONLY for salted passwords (ignore it if you are using an older version). Finally, you’ll notice we have chosen a random 6 digit key and appended it to our table prefix.

To mitigate database injection threats you should change the default WordPress prefix from wp_ to something **more random** like *4i32aK_*. Often attackers use public exploits of the Internet. These exploits will likely rely on the fact that most WordPress installations use the table prefix, “wp_”. Changing this will make it more difficult for an attacker to successfully run exploits.

To easily recognize which prefix used, you could keep something within the database prefix, such as *wp_4i32aK_* or *wp4i32aK_*. The important thing is to modify it so an attacker can’t easily guess your WordPress prefix.

Manually Change

If your blog has already been installed with a predefined table prefix, you will want to change this. This can be quite a tedious process (skip to the end of this section for our cool automated prefix changer plugin).

The first step is to open your **WP-CONFIG.PHP** file and change that line:

> *$table_prefix = 'wp_';*

To use the previous example again we're using the prefix *4i32aK_* so our line should look like this:

> *$table_prefix = '4i32aK_';*

Now that this is done we need to rename all WordPress tables to match the new prefix, to achieve that we need to run a SQL Command[1] within a web interface like PHPMyAdmin or similar, as WordPress doesn't allow it to change the prefix directly.

So these tables:

> *wp_categories, wp_comments, wp_link2cat, wp_links, wp_options, wp_post2cat, wp_postmeta, wp_posts, wp_usermeta, wp_users*

Should become this:

> *4i32aK_categories, 4i32aK_comments, 4i32aK_link2cat, 4i32aK_links, 4i32aK_options, 4i32aK_post2cat, 4i32aK_postmeta, 4i32aK_posts, 4i32aK_usermeta, 4i32aK_users*

Now you may think that we're done right? But that's not the case. WordPress includes some values with the table prefix as well. In order to be able to use your blog we need to change them first.

Within the table wp_options[2] we need to change the value of one record for the field option_name from *wp_user_roles* to *4i32aK_user_roles*[3].

Now you need to replace two[4] other values in this table: wp_usermeta.
The values *wp_autosave_draft_ids* and *wp_user_level* for the field meta_key need to be changed to the new prefix: *4i32aK_autosave_draft_ids* and *4i32aK_user_level*.

That's it! **But BlogSecurity has made this even easier with** [WP Prefix Table Changer]. Please make sure you have backed up your blog and database before running this plugin. It is Alpha software and may have bugs.

---

[1] Example Query: *RENAME TABLE wp_categories TO 4i32a_categories*

[2] The default prefix is used, to avoid confusions that could occur if we would use the new prefix

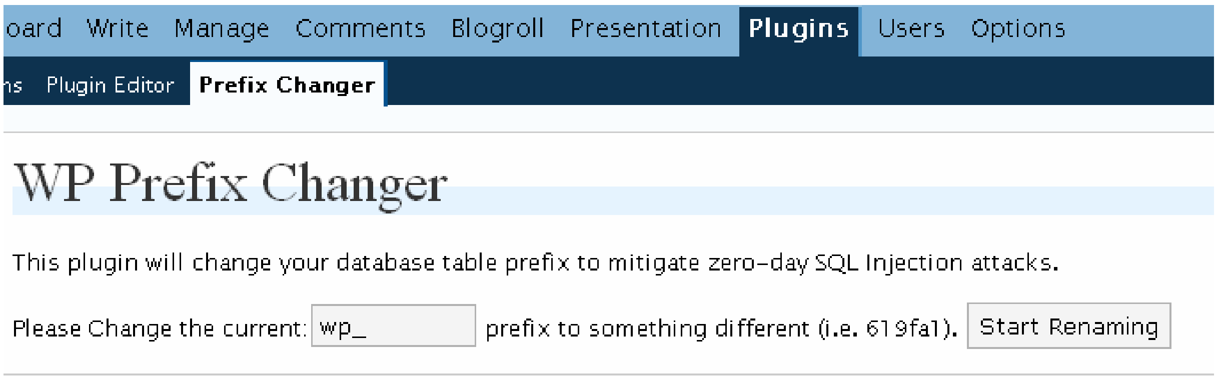
[3] *UPDATE 4i32a_options SET option_name='4i32a_user_roles' WHERE option_name='wp_user_roles' LIMIT 1*

[4] **Note**: it can be that these fields don't exist currently, as they're just created when they're needed, once they get created they get the correct value automatically

WP Prefix Table Changer

We created a plugin called WP Prefix Table Changer, which automates the whole process of renaming your WordPress tables and the mentioned table entries.

After you downloaded the plugin from BlogSecurity.net, you just need to extract the files into your WordPress plugin-folder, which should be **WORDPRESS/WP-CONTENT/PLUGINS**. After this you need to enter the WP-Admin area and enable the WP Prefix Table Changer Plugin. After activation a new submenu tab appears within the plugin page menu called Prefix Changer, click on it. On that page you see the following:


oard Write Manage Comments Blogroll Presentation Plugins Users Options
ns Plugin Editor Prefix Changer
Start Renaming

WP Prefix Changer

This plugin will change your database table prefix to mitigate zero-day SQL Injection attacks.

Please Change the current: wp_ prefix to something different (i.e. 619fa1).

As you can see you this blog is using the default database WP Prefix (wp_). Change it to something **random, meaningless** like our old example *4i32aK_*. Once done, just click the ‘Start Renaming’. The plugin starts to rename all table names from wp_ to your selected random string. It should be noted that 3rd Party Component Tables are also changed, as they need the new prefix too.

The plugin’s last step is to change the prefix within the **WP-CONFIG.PHP** file.

You’ll get a message if this process has succeeded or failed. If the process is successful the **WP-CONFIG.PHP** file will be made read-only (**644**) for security purposes, if the process failed the file is probably READ-ONLY and requires a manual change!

Preparing the Blog

Now your blog is installed and some excellent security fundamentals are in place, well done! Next we are going to change the default 'admin' user account to something different. We are also going to create a limited WordPress user account for day-to-day activities. Finally, we'll install the Role Manager plugin to provide granular control over what your moderator(s) and contributor(s) can do.

Changing your Admin Username

You should rename your default administration account from **admin** to something harder to guess, as all currently available WP versions are vulnerable to User Enumeration. Changing the default admin account to another name will help mitigate password brute force attacks.

Please Note: You must assume the attacker will know your username. Therefore, ensure that you have chosen a strong password. ***We cannot stress this enough!***

Lets connect to our MySQL database with our 'wpuser' account and change the default admin login_name:

```
wp $ mysql -u wpuser –p
mysql> use wp;
UPDATE 4i32aK_users SET user_login='admin', user_login='adm';
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

The 'admin' login has now been changed to 'adm'. Obviously, you may want to use something a little less obvious.

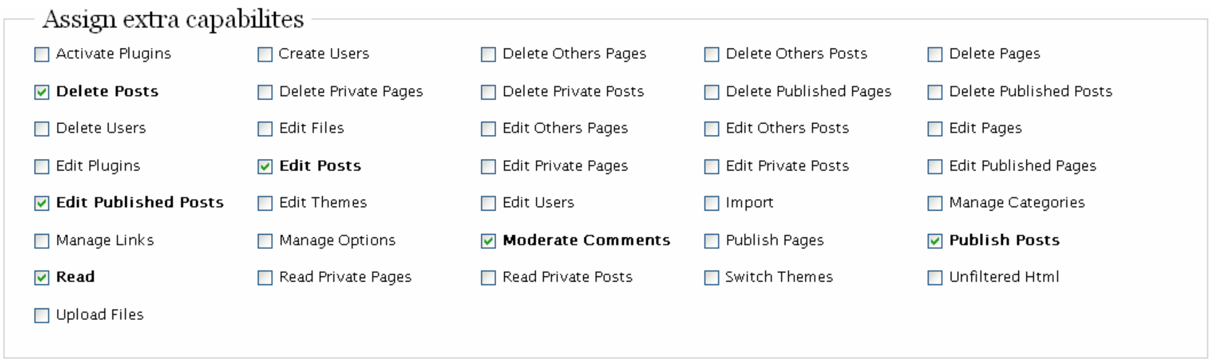
Create a new limited access user

Before we start with that step you should grap yourself a copy of the plugin Role Manager by im-web-gefunden. This plugin will enable you to set granular WP user permissions for each user. After you activated the plugin, create a new user account with the features that you like. We would suggest creating the first account for yourself. Remove all the user account permissions and carefully select ONLY those permissions that you need for day-to-day activities (i.e. write posts, moderate comments etc). Make sure only the admin user account can do powerful tasks such as enable/disable plugins, Upload files, Manage Options, Switch Themes, Import etc. This may take a few days to perfect.

Please Note: The less that your user account is able to do the better your security will be. A contributor level account is generally a good account-level to start with.

The role of Contributor may not have enough rights by default, but we can add these with the Role Manager plugin.

As a standard we suggest new users be given “contributor” level access, however, this plugin allows you a lot more flexibility. As seen below.



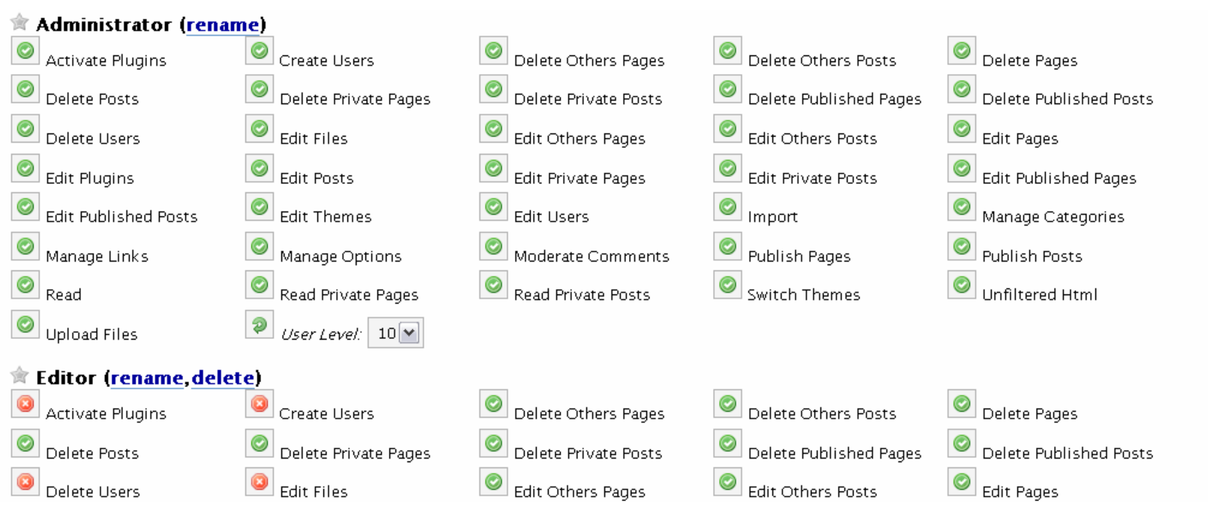
Assign extra capabilites

☐ Activate Plugins	☐ Create Users	☐ Delete Others Pages	☐ Delete Others Posts	☐ Delete Pages
✓ **Delete Posts**	☐ Delete Private Pages	☐ Delete Private Posts	☐ Delete Published Pages	☐ Delete Published Posts
☐ Delete Users	☐ Edit Files	☐ Edit Others Pages	☐ Edit Others Posts	☐ Edit Pages
☐ Edit Plugins	✓ **Edit Posts**	☐ Edit Private Pages	☐ Edit Private Posts	☐ Edit Published Pages
✓ **Edit Published Posts**	☐ Edit Themes	☐ Edit Users	☐ Import	☐ Manage Categories
☐ Manage Links	☐ Manage Options	✓ **Moderate Comments**	☐ Publish Pages	✓ **Publish Posts**
✓ **Read**	☐ Read Private Pages	☐ Read Private Posts	☐ Switch Themes	☐ Unfiltered Html
☐ Upload Files				

See: Role Manager information.

If you have multiple users, it may be a good idea think about what your users “really” need, and create a custom user role.

When creating users, be wary of allowing untrusted users access to roles such as “upload files”, “general plugin access”, “edit files/pages/posts”, “import” and “unfiltered html”, as these roles give the user a lot of power.



Administrator (rename)

✓ Activate Plugins	✓ Create Users	✓ Delete Others Pages	✓ Delete Others Posts	✓ Delete Pages
✓ Delete Posts	✓ Delete Private Pages	✓ Delete Private Posts	✓ Delete Published Pages	✓ Delete Published Posts
✓ Delete Users	✓ Edit Files	✓ Edit Others Pages	✓ Edit Others Posts	✓ Edit Pages
✓ Edit Plugins	✓ Edit Posts	✓ Edit Private Pages	✓ Edit Private Posts	✓ Edit Published Pages
✓ Edit Published Posts	✓ Edit Themes	✓ Edit Users	✓ Import	✓ Manage Categories
✓ Manage Links	✓ Manage Options	✓ Moderate Comments	✓ Publish Pages	✓ Publish Posts
✓ Read	✓ Read Private Pages	✓ Read Private Posts	✓ Switch Themes	✓ Unfiltered Html
✓ Upload Files	*User Level:* 10			

Editor (rename, delete)

✗ Activate Plugins	✗ Create Users	✓ Delete Others Pages	✓ Delete Others Posts	✓ Delete Pages
✓ Delete Posts	✓ Delete Private Pages	✓ Delete Private Posts	✓ Delete Published Pages	✓ Delete Published Posts
✗ Delete Users	✗ Edit Files	✓ Edit Others Pages	✓ Edit Others Posts	✓ Edit Pages

Changing the role rights with Role Manager

Hardening your WP Install[5]

This will cover how to protect your admin area from unauthorised access. This step is easier to achieve for single user blogs, and can be a real pain for multiple user blogs. You have to decide if you want to shoulder that in order to secure your blog or not, but you should.

Restricting wp-content & wp-includes

In this step we're going to limit the access to these directories, we're mainly going to generally deny everything, except the request for images, CSS and some JavaScript files.

You need to put this code into your **.HTACCESS** file for the folders **WP-CONTENT** & **WP-INCLUDES**:

```
Order Allow,Deny
Deny from all
<Files ~ ".(css|jpe?g|png|gif|js)$">
        Allow from all
</Files>
```

Please Note: You may want add "specific" PHP file(s) access for certain templates and plugins.

Restricting wp-admin

Block all except your IP

If you run a single user blog, you may want to restrict access to your **WP-ADMIN** directory via IP. Make sure you have a static IP address (one that doesn't change) before doing this. The **.HTACCESS** file within **WP-ADMIN** should look like that:

```
Order deny,allow
Allow from a.b.c.d #That's your static IP
Please add some example for allowed ip ranges
Deny from all
```

Save the file and try to access the wp-admin folder through a web proxy; it should be blocked if everything is working correctly. After that try access it directly with your IP (a.b.c.d).

If everything works correctly **WP-ADMIN** will now be restricted only to the IP(s) of your choice.

[5] The original article can be found here: http://blogsecurity.net/WordPress/article-210607/

Password Required - .htpasswd

Certainly the preferred option is to use password protection. This means you can still access your admin directory from anywhere, however, it adds an additional security layer and prevents unauthorised users accessing files they shouldn't.

The .htaccess file

he **.HTACCESS** file within **WP-ADMIN** should look like that:

```
#this file should be outside your webroot.
AuthUserFile /srv/www/user1/.htpasswd
AuthType Basic
AuthName "Blog"
require user youruser #making this username difficult to guess can help mitigate password
brute force attacks.
```

The .htpasswd file

This file[6] should, as already mentioned, be placed somewhere out of your web directory, ideally one folder above it.

```
$ htpasswd -cm .htpasswd blog
New password:
Re-type new password:
Adding password for user blog
```

The file .htpasswd should now have been created in your current directory. Make sure the location of this file matches AuthUserFile in 'wp-admin/.htaccess'.

Now test it to see if everything is working. When you try login to your blog, you should now be required to enter your shared username and password before being permitted to access your WordPress login form. If that isn't the case, rewrite the encrypted password file, and check your locations are correct.

[6] More information about that file can be found here: http://httpd.apache.org/docs/1.3/mod/mod_auth.html

SPAM

One of the awesome ideas behind blogging is the ability to allow feedback in the form of comments. Unfortunately, 2 in 3 comments are spam. So a number of solutions have been made available that combat spam in different ways, however, some methods can really annoy your users and limit long-term feedback.

Captcha - Also known as "those pesky graphical images that you can hardly read." Captcha solutions work quite to stop spam but only when implemented in the correct way (as we learnt from Mustlive's recent Captcha bugs of the month project). The major downside to Captchas is that they bug me and probably most people. Who the heck wants to get their glasses on and squint to read some graphic image every single time they want to leave a comment.

Authentication - Having a user register before allowing them to make a comment. Unfortunately, automated programs can usually do this automatically, so Captcha systems are usually now implemented along side this process. At least we only have to enter the Captcha once now, right? Yes, but now I have to login every time I want to leave a comment! That means having to remember another password! OpenID may be the way forward in this area.

Blacklist - Many software packages allow blacklists. This means we can block comments based on "BAD" words. I don't know if you've ever been to a live game between Arsenal and Manchester United? Lets just say there are so many ways to say the same thing. Blacklists are good when implemented with other methods but certainly not as a standalone service.

JavaScript - BlogSecurity's SpamBam uses client-side scripting in your browser to ensure that your user is using a valid browser. This is currently a very effective method to prevent comment-spam as most spammers use automated programs that don't have JavaScript support. I quite like this system, although I feel this could really be an awesome approach but more work is needed on the plugin.

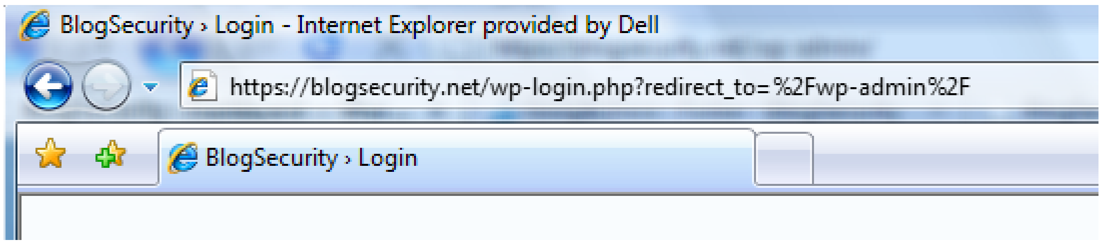
Smart Checks - These are generally Spam systems like Akismet. They use a variety of checks and a chain of blacklists to check whether the comment is spam or not. Great for general use, however, it means that all your comments have to be sent to a third party server. This can cause excess traffic.

There are a number of great anti-spam plugins available, however, generally we suggest using one of two plugins:

- Akismit – Automattic's Anti-Spam Plugin (requires registration for API key)
- SpamBam – BlogSecurity's Anti-Spam Plugin

Blog Encryption

When logging into your blog it is important that you ensure this is done over 'HTTPS'. This prevents attackers capturing your username and password being sent across the internet over HTTP which is clear-text (no encryption).


BlogSecurity › Login - Internet Explorer provided by Dell
https://blogsecurity.net/wp-login.php?redirect_to=%2Fwp-admin%2F
BlogSecurity › Login

First check if your blog supports 'HTTPS'. Simply, point your browser to:

https://yourblog/

Installing 'HTTPS' is out of scope for this paper. Remember, Google is your friend. Lets move on.

As you already have SSL (HTTPS) installed on your blog, all we need to do is install a WordPress plugin that will redirect your browser to 'HTTPS' when accessing 'wp-login.php' and other restricted areas of your blog.

Guess what? That's right, BlogSecurity has once again provided a plugin that does this for you. Grab a copy of the 'bs-wp-encrypt' here.

Simply rename 'bs-wp-https_php.txt' to 'bs-wp-https.php' and place it in your 'wp-content/plugins/' directory. Now go to your admin panel and enable it, walla! Your blog should now recognise restricted areas, and redirect you to HTTPS.

Key Plugins

BlogSecurity have provided some additional plugins to make attackers lives more difficult. We suggest using these plugins as part of default WordPress install.

Disabling WordPress Errors

BlogSecurity Wordpress Noversion plugin (bs-wp-noversion), prevents WordPress version leakage. Another simple, yet extremely useful WordPress security plugin.

Alot of attackers and automated tools will try and determine software versions before launching exploit code. Removing your WordPress blog version may discourage some attackers and certainly will mitigate virus and worm programs that rely on software versions.

Please Note: This plugin may affect other plugins that rely on WP versioning.

The 'bs-wp-noerrors' plugin can be downloaded here.

Removing the WordPress Version

This plugin is now deprecated as of WordPress 2.3.2. WordPress 2.3.2 has error messages disabled by default. This plugin may still be useful for those running older WP versions.

WordPress by default has error messaging turned on:

```
function show_errors() {
$this->show_errors = true;
}
```

*Please Note: Database errors will still be displayed to users even when PHP errors have been turned off. This plugin disables WP DB error messages, preventing sensitive information disclosure such as the WordPress database **table prefix**.*

The plugin 'bs-wp-noerrors' is available here.

Security Above and Beyond

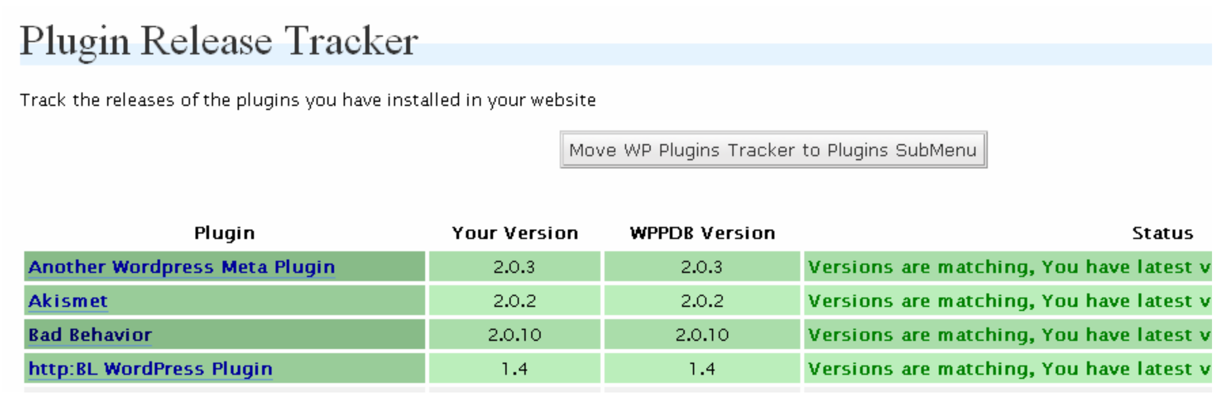
This list of plugins/services can greatly improve the security of your blog.

WPIDS - Detect Intrusions

BlogSecurity have ported PHPIDS (Intrusion Detection System) into WordPress. PHPIDS is able to detect various intrusion attempts. We use this facility to block dangerous attacks. Every intrusion is logged in the database, so you can keep track and take the necessary steps if needed. You're able to get an email if the impact was bigger than a defined value (every intrusion has its own threat score). You can also block the attacker's IP for a given amount of days if the threat isn't scored lowly, however, in all cases WPIDS will attempt to sanitise bad input. You can get your copy from the official PHPIDS website. Note: In order to be able to use this plugin you need to run at least PHP 5.1.6 or higher. A new version of WPIDS will be released shortly that includes BlogSec's recent addition, WP-Lockdown, so keep posted.

WordPress Plugin Tracker – Are you updated?

If you just installed your blog and all plugins from the developer website you should already be running the latest version. You should install the WordPress Plugin Tracker plugin to keep track if you're using the latest plugin versions. After you installed and activated the plugin, run it to see if your plugins are the current versions. You should see something like this:



Track the releases of the plugins you have installed in your website

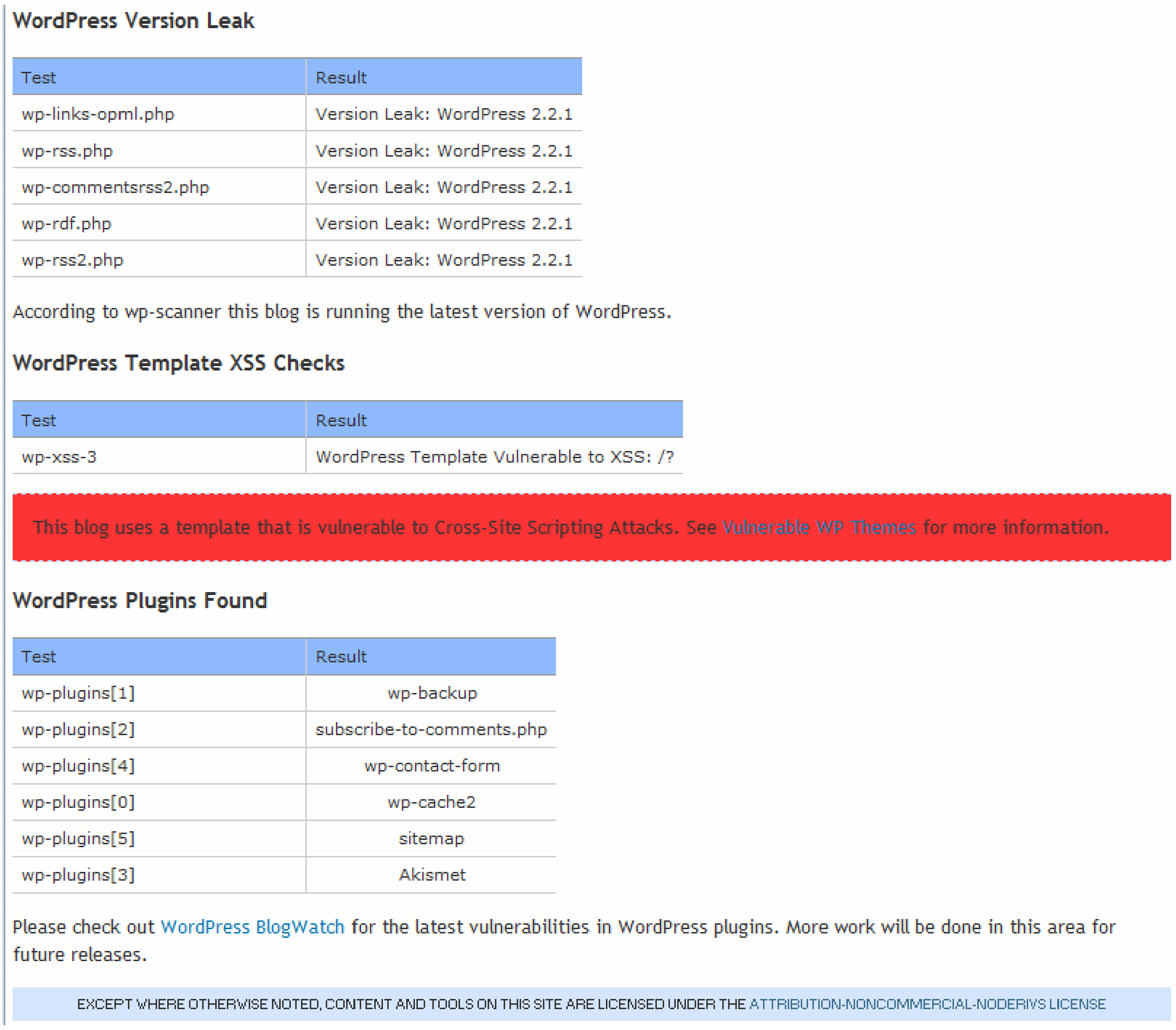
Plugin	Your Version	WPPDB Version	Status
Another Wordpress Meta Plugin	2.0.3	2.0.3	Versions are matching, You have latest v
Akismet	2.0.2	2.0.2	Versions are matching, You have latest v
Bad Behavior	2.0.10	2.0.10	Versions are matching, You have latest v
http:BL WordPress Plugin	1.4	1.4	Versions are matching, You have latest v

This is how the example page of the Plugin Release Tracker looks

If some plugins aren't up to date, you'll be notified about it and by clicking on the title of the plugin to the left, you're easily forwarded to the plugin page where you can grab the update. So it's easy to keep your blog updated.

WordPress Online Security Scanner

BlogSecurity has written a WordPress security checking tool, to check your blog for common security weaknesses. It is excellent at enumerating plugins, checking for Cross-Site Scripting vulnerabilities and much more.



WordPress Version Leak

Test	Result
wp-links-opml.php	Version Leak: WordPress 2.2.1
wp-rss.php	Version Leak: WordPress 2.2.1
wp-commentsrss2.php	Version Leak: WordPress 2.2.1
wp-rdf.php	Version Leak: WordPress 2.2.1
wp-rss2.php	Version Leak: WordPress 2.2.1

According to wp-scanner this blog is running the latest version of WordPress.

WordPress Template XSS Checks

Test	Result
wp-xss-3	WordPress Template Vulnerable to XSS: /?

This blog uses a template that is vulnerable to Cross-Site Scripting Attacks. See Vulnerable WP Themes for more information.

WordPress Plugins Found

Test	Result
wp-plugins[1]	wp-backup
wp-plugins[2]	subscribe-to-comments.php
wp-plugins[4]	wp-contact-form
wp-plugins[0]	wp-cache2
wp-plugins[5]	sitemap
wp-plugins[3]	Akismet

Please check out WordPress BlogWatch for the latest vulnerabilities in WordPress plugins. More work will be done in this area for future releases.

EXCEPT WHERE OTHERWISE NOTED, CONTENT AND TOOLS ON THIS SITE ARE LICENSED UNDER THE ATTRIBUTION-NONCOMMERCIAL-NODERIVS LICENSE

WP-Scanner is a free online service and has tested more then 5000 blogs to date (we have actually lost count now). Check out more information here.

The End

We reached the end of our whitepaper. We hope you enjoyed it, and that you'll have success implementing these security steps. It's always a pleasure to get feedback and your stories so please contact us.