Search WordPress.org

Codex

Codex tools: Log in

Interested in functions, hooks, classes, or methods? Check out the new WordPress Code Reference!

Hardening WordPress

Languages: Deutsch • **English** • 日本語 • Italiano • 한국어 • Português do Brasil • (Add your language)

Security in WordPress is taken very seriously, but as with any other system there are potential security issues that may arise if some basic security precautions aren't taken. This article will go through some common forms of vulnerabilities, and the things you can do to help keep your WordPress installation secure.

This article is not the ultimate quick fix to your security concerns. If you have specific security concerns or doubts, you should discuss them with people whom you trust to have sufficient knowledge of computer security and WordPress.

Contents

What is Security?

Fundamentally, security *is not* about perfectly secure systems. Such a thing might well be impractical, or impossible to find and/or maintain. What security is though is risk reduction, not risk elimination. It's about employing all the appropriate controls available to you, within reason, that allow you to improve your overall posture reducing the odds of making yourself a target, subsequently getting hacked.

Website Hosts

Often, a good place to start when it comes to website security is your hosting environment. Today, there are a number of options available to you, and while hosts offer security to a certain level, it's important to understand where their responsibility ends and yours begins. Here is a good article explaining the complicated dynamic between web hosts and the security of your website. A secure server protects the privacy, integrity, and availability of the resources under the server administrator's control.

Qualities of a trusted web host might include:

- Readily discusses your security concerns and which security features and processes they offer with their hosting.
- Provides the most recent stable versions of all server software.
- Provides reliable methods for backup and recovery.

Decide which security you need on your server by determining the software and data that needs to be secured. The rest of this guide will help you with this.

Website Applications

It's easy to look at web hosts and pass the responsibility of security to them, but there is a tremendous amount of security that lies on the website owner as well. Web hosts are often responsible for the infrastructure on which your website sits, they are not responsible for the application you choose to install.

To understand where and why this is important you must understand how websites get hacked, Rarely is it attributed to the infrastructure, and most often attributed to the application itself (i.e., the environment you are responsible for).

Security Themes

Keep in mind some general ideas while considering security for each aspect of your system:

Limiting access
Making smart choices that reduce possible entry points available to a malicious person.

Containment
Your system should be configured to minimize the amount of damage that can be done in the event that it is compromised.

Preparation and knowledge
Keeping backups and knowing the state of your WordPress installation at regular intervals. Having a plan to backup and recover your installation in the case of catastrophe can help you get back online faster in the case of a problem.

Trusted Sources
Do not get plugins/themes from untrusted sources. Restrict yourself to the WordPress.org repository or well known companies. Trying to get plugins/themes from the outside may lead to issues.

Vulnerabilities on Your Computer

Make sure the computers you use are free of spyware, malware, and virus infections. No amount of security in WordPress or on your web server will make the slightest difference if there is a keylogger on your computer.

Always keep your operating system and the software on it, especially your web browser, up to date to protect you from security vulnerabilities. If you are browsing untrusted sites, we also recommend using tools like no-script (or disabling javascript/flash/java) in your browser.

Vulnerabilities in WordPress

Like many modern software packages, WordPress is updated regularly to address new security issues that may arise. Improving software security is always an ongoing concern, and to that end **you should always keep up to date with the latest version of WordPress**. Older versions of WordPress are not maintained with security updates.

Updating WordPress

Main article: Updating WordPress.

The latest version of WordPress is always available from the main WordPress website at https://wordpress.org. Official releases are not available from other sites -- **never** download or install WordPress from any website other than https://wordpress.org.

Since version 3.7, WordPress has featured automatic updates. Use this functionality to ease the process of keeping up to date. You can also use the WordPress Dashboard to keep informed about updates. Read the entry in the Dashboard or the WordPress Developer Blog to determine what steps you must take to update and remain secure.

If a vulnerability is discovered in WordPress and a new version is released to address the issue, the information required to exploit the vulnerability is almost certainly in the public domain. This makes old versions more open to attack, and is one of the primary reasons you should always keep WordPress up to date.

If you are an administrator in charge of more than one WordPress installation, consider using Subversion to make management easier.

Reporting Security Issues

If you think you have found a security flaw in WordPress, you can help by reporting the issue. See the Security FAQ for information on how to report security issues.

If you think you have found a bug, report it. See Submitting Bugs for how to do this. You might have uncovered a vulnerability, or a bug that could lead to one.

Web Server Vulnerabilities

The web server running WordPress, and the software on it, can have vulnerabilities. Therefore, make sure you are running secure, stable versions of your web server and the software on it, or make sure you are using a trusted host that takes care of these things for you.

If you're on a shared server (one that hosts other websites besides your own) and a website on the same server is compromised, your website can potentially be compromised too even if you follow everything in this guide. Be sure to ask your web host what security precautions they take.

Network Vulnerabilities

The network on both ends -- the WordPress server side and the client network side -- should be trusted. That means updating firewall rules on your home router and being careful about what networks you work from. An Internet cafe where you are sending passwords over an unencrypted connection, wireless or otherwise, is **not** a trusted network.

Your web host should be making sure that their network is not compromised by attackers, and you should do the same. Network vulnerabilities can allow passwords and other sensitive information to be intercepted.

Passwords

Many potential vulnerabilities can be avoided with good security habits. A strong password is an important aspect of this.

The goal with your password is to make it hard for other people to guess and hard for a brute force attack to succeed. Many automatic password generators are available that can be used to create secure passwords.

WordPress also features a password strength meter which is shown when changing your password in WordPress. Use this when changing your password to ensure its strength is adequate.

Things to avoid when choosing a password:

- Any permutation of your own real name, username, company name, or name of your website.
- A word from a dictionary, in any language.
- A short password.
- Any numeric-only or alphabetic-only password (a mixture of both is best).

A strong password is necessary not just to protect your blog content. A hacker who gains access to your administrator account is able to install malicious scripts that can potentially compromise your entire server.

In addition to using a strong password, it's a good idea to enable two-step authentication as an additional security measure.

FTP

When connecting to your web server you should always use SFTP encryption if your web host provides it. If you are unsure if your web host provides SFTP or not, just ask them.

Using SFTP involves the same process as FTP, except your password and other data is encrypted as it is transmitted between your computer and your website. This means your password is never sent in the clear and cannot be intercepted by an attacker. Most FTP clients support SFTP.

File Permissions

Some neat features of WordPress come from allowing various files to be writable by the web server. However, allowing write access to your files is potentially dangerous, particularly in a shared hosting environment.

It is best to lock down your file permissions as much as possible and to loosen those restrictions on the occasions that you need to allow write access, or to create specific folders with less restrictions for the purpose of doing things like uploading files.

Here is one possible permission scheme.

All files should be owned by your user account, and should be writable by you. Any file that needs write access from WordPress should be writable by the web server, if your hosting set up requires it, that may mean those files need to be group-owned by the user account used by the web server process.

`/`
The root WordPress directory: all files should be writable only by your user account, except `.htaccess` if you want WordPress to automatically generate rewrite rules for you.

`/wp-admin/`
The WordPress administration area: all files should be writable only by your user account.

`/wp-includes/`
The bulk of WordPress application logic: all files should be writable only by your user account.

`/wp-content/`
User-supplied content: intended to be writable by your user account and the web server process.

Within `/wp-content/` you will find:

`/wp-content/themes/`
Theme files. If you want to use the built-in theme editor, all files need to be writable by the web server process. If you do not want to use the built-in theme editor, all files can be writable only by your user account.

`/wp-content/plugins/`
Plugin files: all files should be writable only by your user account.

Other directories that may be present with `/wp-content/` should be documented by whichever plugin or theme requires them. Permissions may vary.

Changing file permissions

If you have shell access to your server, you can change file permissions recursively with the following command:

For Directories:

```
find /path/to/your/wordpress/install/ -type d -exec chmod 755 {} \;
```

For Files:

```
find /path/to/your/wordpress/install/ -type f -exec chmod 644 {} \;
```

Regarding Automatic Updates

When you tell WordPress to perform an automatic update, all file operations are performed as the user that owns the files, not as the web server's user. All files are set to 0644 and all directories are set to 0755, and writable by only the user and readable by everyone else, including the web server.

Database Security

If you run multiple blogs on the same server, it is wise to consider keeping them in separate databases each managed by a different user. This is best accomplished when performing the initial WordPress installation. This is a containment strategy: if an intruder successfully cracks one WordPress installation, this makes it that much harder to alter your other blogs.

If you administer MySQL/MariaDB yourself, ensure that you understand your database configuration and that unneeded features (such as accepting remote TCP connections) are disabled. See Secure MySQL Database Design for a nice introduction.

Restricting Database User Privileges

For normal WordPress operations, such as posting blog posts, uploading media files, posting comments, creating new WordPress users and installing WordPress plugins, the MySQL/MariaDB database user only needs data read and data write privileges to the database; SELECT, INSERT, UPDATE and DELETE.

Therefore any other database structure and administration privileges, such as DROP, ALTER and GRANT can be revoked. By revoking such privileges you are also improving the containment policies.

> **Note:** Some plugins, themes and major WordPress updates might require to make database structural changes, such as add new tables or change the schema. In such case, before installing the plugin or updating a software, you will need to temporarily allow the database user the required privileges.
>
> **WARNING:** Attempting updates without having these privileges can cause problems when database schema changes occur. Thus, it is **NOT** recommended to revoke these privileges. If you do feel the need to do this for security reasons, then please make sure that you have a solid backup plan in place first, with regular whole database backups which you have tested are valid and that can be easily restored. A failed database upgrade can usually be solved by restoring the database back to an old version, granting the proper permissions, and then letting WordPress try the database update again. Restoring the database will return it back to that old version and the WordPress administration screens will then detect the old version and allow you to run the necessary SQL commands on it. Most WordPress upgrades do not change the schema, but some do. Only major point upgrades (3.7 to 3.8, for example) will alter the schema. Minor upgrades (3.8 to 3.8.1) will generally not. Nevertheless, **keep a regular backup**.

Securing wp-admin

Adding server-side password protection (such as BasicAuth) to `/wp-admin/` adds a second layer of protection around your blog's admin area, the login screen, and your files. This forces an attacker or bot to attack this second layer of protection instead of your actual admin files. Many WordPress attacks are carried out autonomously by malicious software bots.

Simply securing the `wp-admin/` directory might also break some WordPress functionality, such as the AJAX handler at `wp-admin/admin-ajax.php`. To avoid this, you may want to make an exception for the `wp-admin/admin-ajax.php` file. This would protect access to the `wp-admin/` directory while allowing full functionality of AJAX. See the Resources section for more documentation on how to password protect your `wp-admin/` directory properly.

The most common attacks against a WordPress blog usually fall into two categories.

1. Sending specially-crafted HTTP requests to your server with specific exploit payloads for specific vulnerabilities. These include

old/outdated plugins and software.
2. Attempting to gain access to your blog by using "brute-force" password guessing.

The ultimate implementation of this "second layer" password protection is to require an HTTPS SSL encrypted connection for administration, so that all communication and sensitive data is encrypted. *See Administration Over SSL.*

Securing wp-includes

A second layer of protection can be added where scripts are generally not intended to be accessed by any user. One way to do that is to block those scripts using mod_rewrite in the .htaccess file. **Note:** to ensure the code below is not overwritten by WordPress, place it outside the `# BEGIN WordPress` and `# END WordPress` tags in the .htaccess file. WordPress can overwrite anything between these tags.

```
# Block the include-only files.
<IfModule mod_rewrite.c>
RewriteEngine On
RewriteBase /
RewriteRule ^wp-admin/includes/ - [F,L]
RewriteRule !^wp-includes/ - [S=3]
RewriteRule ^wp-includes/[^/]+\.php$ - [F,L]
RewriteRule ^wp-includes/js/tinymce/langs/.+\.php - [F,L]
RewriteRule ^wp-includes/theme-compat/ - [F,L]
</IfModule>


# BEGIN WordPress
```

Note that this won't work well on Multisite, as `RewriteRule ^wp-includes/[^/]+\.php$ - [F,L]` would prevent the ms-files.php file from generating images. Omitting that line will allow the code to work, but offers less security.

Securing wp-config.php

You can move the `wp-config.php` file to the directory above your WordPress install. This means for a site installed in the root of your webspace, you can store `wp-config.php` outside the web-root folder.

> **Note:** Some people assert that moving wp-config.php has minimal security benefits and, if not done carefully, may actually introduce serious vulnerabilities. Others disagree.

Note that `wp-config.php` can be stored ONE directory level above the WordPress (where wp-includes resides) installation. Also, make sure that only you (and the web server) can read this file (it generally means a 400 or 440 permission).

If you use a server with .htaccess, you can put this in that file (at the very top) to deny access to anyone surfing for it:

```
<files wp-config.php>
order allow,deny
deny from all
</files>
```

Disable File Editing

The WordPress Dashboard by default allows administrators to edit PHP files, such as plugin and theme files. This is often the first tool an attacker will use if able to login, since it allows code execution. WordPress has a constant to disable editing from Dashboard. Placing this line in wp-config.php is equivalent to removing the 'edit_themes', 'edit_plugins' and 'edit_files' capabilities of all users:

```
define('DISALLOW_FILE_EDIT', true);
```

This will not prevent an attacker from uploading malicious files to your site, but might stop some attacks.

Plugins

First of all, make sure your plugins are always updated. Also, if you are not using a specific plugin, delete it from the system.

Firewall

There are many plugins and services that can act as a firewall for your website. Some of them work by modifying your .htaccess file and restricting some access at the Apache level, before it is processed by WordPress. A good example is iThemes Security or All in One WP Security. Some firewall plugins act at the WordPress level, like WordFence and Shield, and try to filter attacks as WordPress is loading, but before it is fully processed.

Besides plugins, you can also install a WAF (web firewall) at your web server to filter content before it is processed by WordPress. The most popular open source WAF is ModSecurity.

A website firewall can also be added as intermediary between the traffic from the internet and your hosting server. These services all function as reverse proxies, in which they accept the initial requests and reroute them to your server, stripping it of all malicious requests. They accomplish this by modifying your DNS records, via an A record or full DNS swap, allowing all traffic to pass through the new network first. This causes all traffic to be filtered by the firewall before reaching your site. A few companies offer such service, like CloudFlare, Sucuri and Incapsula.

Additionally, these third parties service providers function as Content Distribution Network (CDNs) by default, introducing performance optimization and global reach.

Plugins that need write access

If a plugin wants write access to your WordPress files and directories, please read the code to make sure it is legit or check with someone you trust. Possible places to check are the Support Forums and IRC Channel.

Code execution plugins

As we said, part of the goal of hardening WordPress is containing the damage done if there is a successful attack. Plugins which allow arbitrary PHP or other code to execute from entries in a database effectively magnify the possibility of damage in the event of a successful attack.

A way to avoid using such a plugin is to use custom page templates that call the function. Part of the security this affords is active only when you disallow file editing within WordPress.

Security through obscurity

Security through obscurity is generally an unsound primary strategy. However, there are areas in WordPress where obscuring information *might* help with security:

1. **Rename the administrative account:** When creating an administrative account, avoid easily guessed terms such as `admin` or `webmaster` as usernames because they are typically subject to attacks first. On an existing WordPress install you may rename the existing account in the MySQL/MariaDB command-line client with a command like `UPDATE wp_users SET user_login = 'newuser' WHERE user_login = 'admin';`, or by using a MySQL/MariaDB frontend like phpMyAdmin.
2. **Change the table_prefix:** Many published WordPress-specific SQL-injection attacks make the assumption that the table_prefix is `wp_`, the default. Changing this can block at least some SQL injection attacks.

Data Backups

Back up your data regularly, including your MySQL/MariaDB databases. See the main article: Backing Up Your Database.

Data integrity is critical for trusted backups. Encrypting the backup, keeping an independent record of MD5 hashes for each backup file, and/or placing backups on read-only media increases your confidence that your data has not been tampered with.

A sound backup strategy could include keeping a set of regularly-timed snapshots of your entire WordPress installation (including WordPress core files and your database) in a trusted location. Imagine a site that makes weekly snapshots. Such a strategy means that if a site is compromised on May 1st but the compromise is not detected until May 12th, the site owner will have pre-compromise backups that can help in rebuilding the site and possibly even post-compromise backups which will aid in determining how the site was compromised.

Logging

Logs are your best friend when it comes to understanding what is happening with your website, especially if you're trying to perform forensics. Contrary to popular beliefs, logs allow you to see what was done and by who and when. Unfortunately the logs will not tell you who, username, logged in, but it will allow you to identify the IP and time and more importantly, the actions the attacker might have taken. You will be able to see any of these attacks via the logs - Cross Site Scripting (XSS), Remote File Inclusion (RFI), Local File Inclusion (LFI) and Directory Traversal attempts. You will also be able to see brute force attempts. There are various examples and tutorials available to help guide you through the process of parsing and analyzing your raw logs.

If you get more comfortable with your logs you'll be able to see things like, when the theme and plugin editors are being used, when someone updates your widgets and when posts and pages are added. All key elements when doing forensic work on your web server. The are a few WordPress Security plugins that assist you with this as well, like the Sucuri Auditing tool or the Audit Trail plugin.

There are two key open-source solutions you'll want on your web server from a security perspective, this is a layered approach to security.

OSSEC can run on any NIX distribution and will also run on Windows. When configured correctly its very powerful. The idea is correlate and aggregate all the logs. You have to be sure to configure it to capture all access_logs and error_logs and if you have multiple websites on the server account for that. You'll also want to be sure to filter out the noise. By default you'll see a lot of noise and you'll want to configure it to be really effective.

Monitoring

Sometimes prevention is not enough and you may still be hacked. That's why intrusion detection/monitoring is very important. It will allow you to react faster, find out what happened and recover your site.

Monitoring your logs

If you are on a dedicated or virtual private server, in which you have the luxury of root access, you have the ability easily configure things so that you can see what's going on. OSSEC easily facilitates this and here is a little write up that might help you out OSSEC for Website Security - Part I.

Monitoring your files for changes

When an attack happens, it always leave traces. Either on the logs or on the file system (new files, modified files, etc). If you are using OSSEC for example, it will monitor your files and alert you when they change.

Goals

The goals of file system tracking include:

- Monitor changed and added files
- Log changes and additions
- Ability to revert granular changes
- Automated alerts

General approaches

Administrators can monitor file system via general technologies such as:

- System utilities
- Revision control
- OS/kernel level monitoring

Specific tools

Options for file system monitoring include:

- diff - build clean test copy of your site and compare against production
- Git - source code management
- inotify and incron - OS kernel level file monitoring service that can run commands on filesystem events
- Watcher - Python inotify library
- OSSEC - Open Source Host-based Intrusion Detection System that performs log analysis, file integrity checking, policy monitoring, rootkit detection, real-time alerting and active response.

Considerations

When configuring a file based monitoring strategy, there are many considerations, including the following.

Run the monitoring script/service as root

This would make it hard for attackers to disable or modify your file system monitoring solution.

Disable monitoring during scheduled maintenance/upgrades

This would prevent unnecessary notifications when you are performing regular maintenance on the site.

Monitor only executable filetypes

It may be reasonably safe to monitor only executable file types, such as .php files, etc.. Filtering out non-executable files may reduce unnecessary log entries and alerts.

Use strict file system permissions

Read about securing file permissions and ownership. In general, avoid allowing *execute* and *write* permissions to the extent possible.

Monitoring your web server externally

If the attacker tries to deface your site or add malware, you can also detect these changes by using a web-based integrity monitor solution. This comes in many forms today, use your favorite search engine and look for Web Malware Detection and Remediation and you'll likely get a long list of service providers.

Resources

- How to Improve WordPress Security (Infographic)
- Security Plugins
- WordPress Security Cutting Through the BS
- e-Book: Locking Down WordPress
- wpsecure.net has a few guides on how to lock down WordPress.
- A Beginners Guide to Hardening WordPress
- Brad Williams: Lock it Up (Video)
- 21 Ways to Secure Your WordPress Site