Search ... Search

RECENT ARTICLES

- 10 Tips and Best Practices To Improve PHP Security
- How to use Proxychains in Kali Linux OS
- Tips to Hack Facebook without Hassle
- Bruteforce WordPress with XMLRPC Python Exploit
- Top 10 Essential CTF Tools for Solving Reversing Challenges
- How to turn on PowerShell Transcription Logging in Windows 10
- Top 10 NMAP Widely Used Commands
- Top 8 Basic Google Search Dorks [Live Examples]
- Top 3 Open Source SSL Testing Tools

TUTORIALS

10 Tips And Best Practices To Improve PHP Security

July 17, 2019 *H4ck0* Comment(0)

As PHP has evolved since its inception in 1994, it soon became a model of database interaction, and creating a framework where users are able to develop dynamic web applications through a server.

PHP is a server-side programming language that can be used to make static web pages more dynamic in the sense where it can do logical operations in the backend of a server and return/output results accordingly on a page.

Every programmer worth their salt is at least aware of the security implications of building web sites and online apps. You deliberately expose your code to the public, to the world, to anyone and everyone who will come (good people and bad).

One of the early failings of PHP was to prioritise ease of use over security of code (the horror stories from relics like register_globals are only a quick google away).

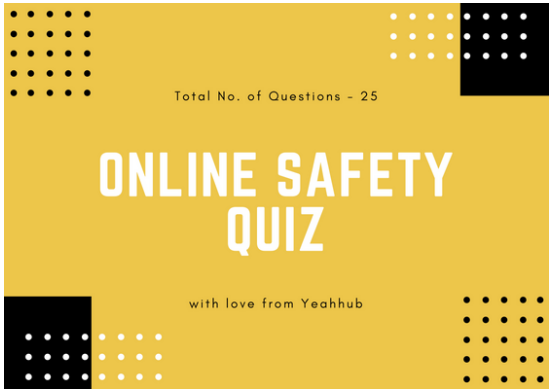
With newer, more secure defaults and functions like register_globals being depreciated PHP is safer than ever online. And although most security problems are caused by the programmer rather than the language, even the newest web coders seem to have an appreciation of security issues right off the bat these days.

Let's see what are the top 10 steps all website owners should take to keep their PHP website secure:

1. PHP Sessions Security
2. Disable Display Errors
3. Restrict File Uploads
4. Disable sensitive functions in PHP
5. Disable allow_url_fopen
6. Disable magic_quotes

» Overview of Mobile Learning Platforms

7. Disable register_globals
8. Disable use_trans_sid
9. Make use of correct php.ini file
10. Analyse PHP Setting using “PhpSecInfo”



1. PHP Sessions Security –

All dynamic websites always have some kind of sensitive information which need to be patched to ensure that it will not be exploited because of session related issues.

There are a couple of things to do in order to keep your session secure:

- Use SSL when authenticating users or performing sensitive operations. (Use sslforfree.com for free https certificate)
- Regenerate the session id whenever the security level changes (such as logging in). You can even regenerate the session id every request if you wish by using session_regenerate_id directive.
- Have sessions time out after a fixed interval.
- Don’t use register globals
- Store authentication details on the server. That is, don’t send details such as username in the cookie.
- Check the **$_SERVER[‘HTTP_USER_AGENT’]**. This adds a small barrier to session hijacking. You can also check the IP address. But this causes problems for users that have changing IP address due to load balancing on multiple internet connections etc (which is the case in our environment here).
- Lock down access to the sessions on the file system or use custom session handling
- For sensitive operations consider requiring logged in users to provide their authentication details again

Besides this, there are only two major session attacks exists:

a) Session fixation Attacks.

This can be prevented by using session_regenerate_id()

b) Session Hijacking:

This can be prevented via data encryption by using SSL Certificates. your site will now run on https and not http.

Furthermore, the other two most important configuration options which you need to change wrt to SESSION are:

- **session.cookie_httponly should be set to 1.**
  - This tells the user's browser not to make this cookie available to Javascript, which limits the damage of a cross-site scripting attack.
- **session.cookie_secure should be set to 1.**
  - This tells the user's browser not to send the cookie at all unless over HTTPS.

2. Disable Display Errors –

There are two methods for viewing PHP errors that occur while running your website. You can either display errors directly on your website (viewable from any web browser) which is very very dangerous or enable error logging to write the errors to a specified file (viewable inside a text file).

This article will cover how to turn OFF the display_errors by editing PHP.ini file and will also covery how to adjust error reporting settings, configure error logging, and use the ini_set() function to help troubleshoot PHP errors with your website.

While your website is live, the php.ini file should have display_errors disabled for security reasons.

The display_errors directive determines whether error messages should be sent to the browser. These messages frequently contain sensitive information about your web application environment and should always be disabled.

This directive must be set to "on" in the PHP ini file. This will display all the errors including syntax or parse errors that cannot be displayed by just calling the ini_set function in the PHP code. The PHP ini file can be found in the displayed output of phpinfo() function and is labeled loaded configuration file. This directive in the ini configuration must be set to off, if the web application is in production.

Open php.ini file and set the following options:

```
display_errors = Off
log_errors = On
```

Alternatively, the setting can also be disabled in Apache's httpd.conf or .htaccess file to suppress all PHP errors:

```
php_flag display_errors Off
php_flag log_errors On
```

NOTE: *display_errors should be disabled and all error messages should be passed to system log files using the log_errors directive so this directive should be turned ON.*

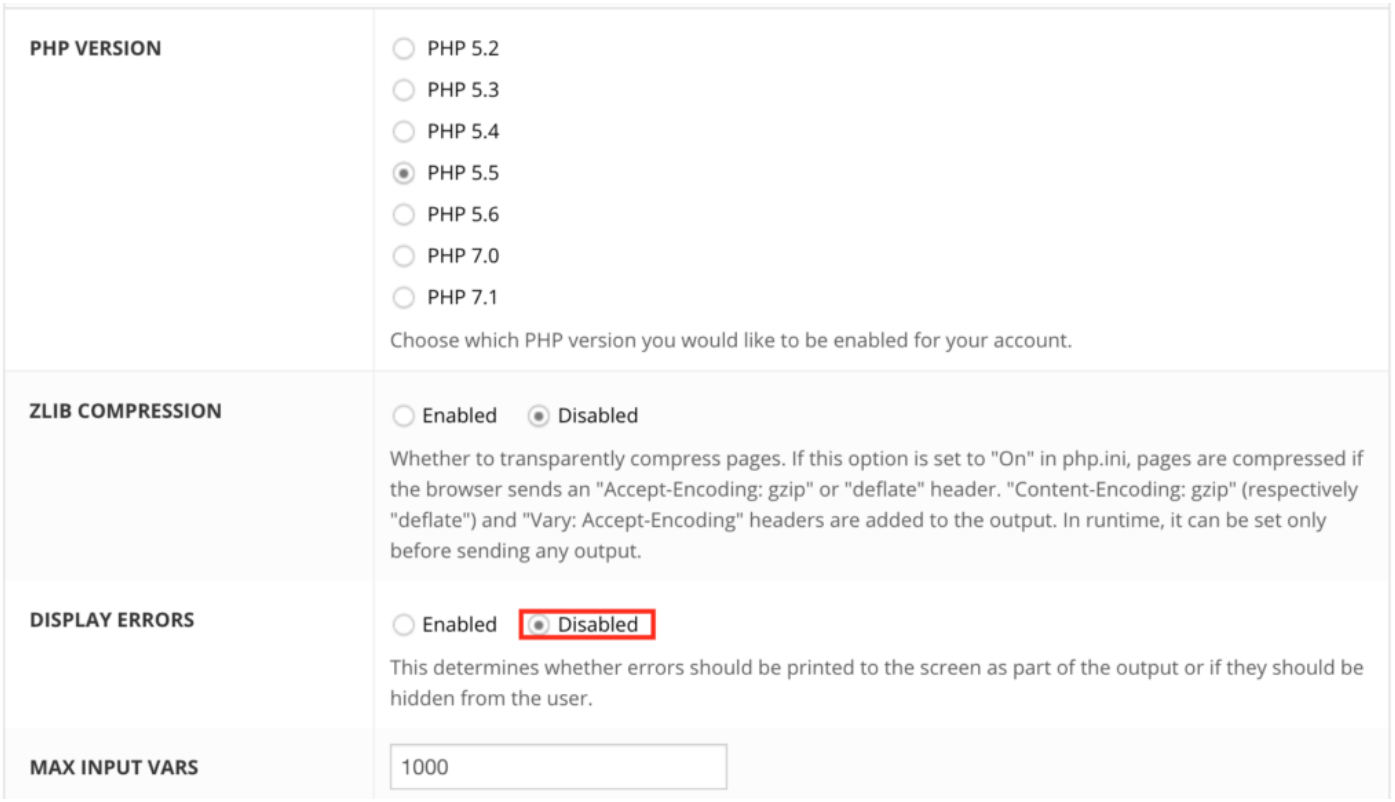
By default all errors are written to the error_log, which is set to /dev/null. This means, error logging won't occur. When errors are turned on, errors will be stored in a file in the directory the error occurs in.

In the case you want errors to not display site wide and you want to check errors on a single page, you can use the ini_set() function to have errors displayed on a particular page. The basic syntax from php.net shows the function and its parameters is as follows which can be placed at the top of your PHP page with error_reporting variable in it:

```
string ini_set ( string $varname , string $newvalue )

ini_set('display_errors', '1');
```

Furthermore, if you are using CPANEL based hosting, then you can easily turn of all display errors from the CPANEL itself by navigating PHP configuration option as shown below:



3. Restrict File Uploads –

Well if you are not using any file upload functionality then its always a good idea to turn off. Hackers can easily abuse the functionality of file upload by uploading malicious PHP scripts and quickly inject into your web application.

To disable this, edit php.ini file and set off to file_uploads directive as shown below:

```
file_uploads = off
```

In case, if you are using file upload control, then make sure that you must change the default temporary directory used for file uploads which can easily be changed by editing same file:

```
upload_tmp_dir = /var/php_tmp
```

You may also restrict the size of files that can be uploaded by setting the following control:

```
upload_max_filesize = 10M
```

To ensure that file uploads work correctly, the **post_max_size** directive should be a little larger than the **upload_max_filesize**. For example, the following settings demonstrate how to set a file upload limit to 10 megabytes:

```
upload_max_filesize = 10M
post_max_size = 16M
```

After modifying the php.ini file, make sure that you must restart your Apache server by typing "**service apache2 restart**" command.

4. Disable Sensitive Functions in PHP –

PHP language has a lot of functions which can be used to hack the server if they are not properly configured. A few dangerous functions are exec(), passthru(), shell_exec() etc which you can easily disable by editing php.ini file with directive disable_functions.

This disable_functions directive allows you to disable certain functions for security reasons.

To do this, open **php.ini** file with any text editor such as Notepad++, Vim (if Linux machine) and set new list as follows:

```
disable_functions = exec, passthru, shell_exec, system, proc_open, popen, curl_exec, curl_multi_exec, parse_ini_file, show_source
```

After saving the php.ini, restart the Apache server by typing **service apache2 restart** (If it is Linux Server).

5. Disable allow_url_fopen

This directive allows PHP's file functions to retrieve data from remote locations, like FTP or HTTP.

If an attacker can manipulate the arguments to those functions, they can use a URL under their control as the argument and run their own remote scripts so called Remote file inclusion (RFI). This directive allows functions like include() and require() to load and run code from remote URLs which puts your site at high risk.

To disable this directive, turn off the following function by editing **php.ini** file as shown below:

```
allow_url_fopen = Off
```

With echo **file_get_contents($_POST['url']);** function, someone can easily pass a file path instead of a URL and have access to your server's files.

The same setting can also be applied in Apache's httpd.conf file:

```
php_admin_flag allow_url_fopen Off
```

A large number of code injection vulnerabilities reported in PHP web applications are caused by enabling **allow_url_fopen** and bad input filtering. You should always disable this directive for security reasons.

6. Disable magic_quotes

This directive can only be disabled at system level. The magic_quotes_gpc option was introduced to help protect servers from SQL injection attacks. It effectively executes addslashes() on all information received over GET, POST or COOKIE.

For example, if a user types "**hello yeahhub**" (with the quotation marks) in an HTML form, PHP automatically escapes the quotation marks and stores the value as \"**hello yeahhub**\".

To disable this option:

```
magic_quotes_gpc = Off
```

The same setting can also be applied in Apache's httpd.conf or .htaccess file:

```
php_flag magic_quotes_gpc Off
```

In case, If you are getting a 500 Internal Server Error, then you have to put the above settings in the starting of php file:

```
ini_set ('magic_quotes_gpc', 1);
```

To verify this setting is active, create a PHP test file that contains the following code in the same directory where the .htaccess file is located:

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

7. Disable register_globals

The **register_globals** directive is disabled by default. You should be aware of the security implications of enabling the register_globals directive.

Note: *This feature has been DEPRECATED as of PHP 5.3.0 and REMOVED as of PHP 5.4.0.*

register_globals is an internal PHP setting which registers the **$_REQUEST** array's elements as variables. If you submit a value in a form, via POST or GET, the value of that input will automatically be accessible via variable in the PHP script, named after the name of the input field.

To disable this directive, edit the php.ini file and set the following directive to value Off.

```
register_globals = Off
```

For the Apache Web Server, use the **php_flag** directive in a .htaccess file to disable register globals on a per-directory tree basis.

```
php_flag register_globals Off
```

If you're using PHP as an Apache module, you can put the below directive in an .htaccess file (or httpd.conf), restricted to the file or files you want:

```
<FilesMatch "^foo\.php$">
php_flag register_globals Off
</FilesMatch>
```

8. Disable use_trans_sid

When **use_trans_sid** is enabled, PHP will pass the session ID via the URL. This makes the application more vulnerable to session hijacking attacks.

Session hijacking is basically a form of identity theft wherein a hacker impersonates a legitimate user by stealing his session ID. When the session token is transmitted in a cookie, and the request is made on a secure channel (that is, it uses SSL), the token is secure.

In reality, it makes those users vulnerable to having their sessions hijacked by anyone who might:

- see the URL over the user's shoulder
- be sent the URL by the user

- retrieve the URL from browser history

You can easily disable this option either by editing php.ini file or .htaccess file as shown below:

With php.ini file –

```
session.use_trans_sid = 'off'
```

With .htaccess file –

```
php_flag session.use_trans_sid off
```

9. Make use of correct php.ini file

Setting up a php.ini file is a fairly straightforward and simple process. In short, the php.ini file allows you to customize the settings on the server specifically for your account.

In real time, you can do so many things with php.ini file like to increase the maximum file upload size or to increase the limit of memory execution etc.

But make sure that you should take a backup of php.ini file before making any changes to it. In case if it already messed up then you can download the default php.ini file from below links:

- Download php.ini version 5.5
- Download php.ini version 5.6

- Download php.ini version 7.0
- Download php.ini version 7.1
- Download php.ini version 7.2

The fastest way to find which path **php.ini** is in is to use the "**locate**" command , but as we have said, there are usually several php.ini files inside a server, and therefore this option would only be worth remembering the route of the file if we already knew it.

For Linux Server, the absolute path of php.ini file would be:

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

Another way to find the php.ini file would be to use the PHPInfo() PHP function to locate where the php.ini configuration is being read.

Furthermore, you can also use grep attribute to locate the php.ini file as shown in following command:

```
Command: php -i |grep php.ini
```

Even every PHP has two sets of php.ini namely php.ini-production and php.ini-development. Be careful about the following functions for both php.ini sets:

- short_open_tag
- disable_functions

- memory_limit
- session.gc_probability
- display_startup_errors
- track_errors
- error_reporting

10. Analyse PHP Setting using “PhpSecInfo”

PhpSecInfo provides an equivalent to the phpinfo() function that reports security information about the PHP environment, and offers suggestions for improvement. It is not a replacement for secure development techniques, and does not do any kind of code or app auditing, but can be a useful tool in a multilayered security approach.

For more reference, please visit – http://phpsec.org/projects/phpsecinfo/

Have something to say about this article? Comment below or share it with us on Facebook or Twitter.

H4ck0

Step by step hacking tutorials about wireless cracking, kali linux, metasploit, ethical hacking, seo tips and tricks, malware analysis and scanning.

https://www.yeahhub.com/

WHERE SHOULD WE SEND ?

HACKING TUTORIALS & INFOSEC NEWS?

Subscribe to Our Newsletter and Get Instant Delivered to Your Email Inbox.

Enter your first name

Enter your email here

Subscribe Now

We respect your privacy and take protecting it seriously.

RELATED ARTICLES

BOOKS

A to Z Programming Notes – By GoalKicker.com

November 12, 2018 *H4ck0*

TECH ARTICLES

1xx, 2xx, 3xx, 4xx and 5xx HTTP Status Codes

August 12, 2018 *H4ck0*

TECH NEWS

HBO agreed to pay $250,000 as a ransom but Hackers denied to accept

August 12, 2017 *H4ck0*

How to use Proxych...

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment

Name *

Email *

Website

Please enter an answer in digits:

eight + 11 =

Post Comment

DISCLAIMER

Yeahhub.com does not represent or endorse the accuracy or reliability of any information's, content or advertisements contained on,

RECENT COMMENTS

LATEST ARTICLES

» 10 Tips and Best Practices To Improve PHP Security

July 17, 2019

distributed through, or linked, downloaded or accessed from any of the services contained on this website, nor the quality of any products, information's or any other material displayed,purchased, or obtained by you as a result of an advertisement or any other information's or offer in or in connection with the services herein.

- Cortez on Persistent Backdoor in Android using Kali Linux with a Shell script
- yr ho on How to Download Wistia Videos without any Tool
- Jimmy Johns Jarner on How to Download Wistia Videos without any Tool
- Wangolo Joel on Subdomain Enumeration Tools – 2019 Update

- How to use Proxychains in Kali Linux OS
  July 9, 2019
- Tips to Hack Facebook without Hassle
  June 25, 2019
- Bruteforce WordPress with XMLRPC Python Exploit
  June 17, 2019
- Top 10 Essential CTF Tools for Solving Reversing Challenges
  June 16, 2019