

- Secrets
- ABAP
- Apex
- C
- C++
- CloudFormation
- COBOL
- C#
- CSS
- Flex
- Go
- HTML
- Java
- JavaScript
- Kotlin
- Objective C
- PHP
- PL/I
- PL/SQL
- Python
- RPG
- Ruby
- Scala
- Swift
- Terraform
- Text
- TypeScript
- T-SQL
- VB.NET
- VB6
- XML



PHP static code analysis

Unique rules to find Bugs, Vulnerabilities, Security Hotspots, and Code Smells in your PHP code

All rules 268 Vulnerability 40 Bug 51 Security Hotspot 33 Code Smell 144

Tags

Search by name...

Code Smell

More than one property should not be declared per statement

Code Smell

The "var" keyword should not be used

Code Smell

"<?php" and "<?=" tags should be used

Code Smell

File names should comply with a naming convention

Code Smell

Comments should not be located at the end of lines of code

Code Smell

Local variable and function parameter names should comply with a naming convention

Code Smell

Field names should comply with a naming convention

Code Smell

Lines should not end with trailing whitespaces

Code Smell

Files should contain an empty newline at the end

Code Smell

Modifiers should be declared in the correct order

Code Smell

An open curly brace should be located at the beginning of a line

Code Smell

Creating cookies without the "secure" flag is security-sensitive

Analyze your code

Security Hotspot Minor cwe privacy sans-top25 owasp

When a cookie is protected with the secure attribute set to true it will not be send by the browser over an unencrypted HTTP request and thus cannot be observed by an unauthorized person during a man-in-the-middle attack.

Ask Yourself Whether

- the cookie is for instance a session-cookie not designed to be sent over non-HTTPS communication.
- it's not sure that the website contains mixed content or not (ie HTTPS everywhere or not)

There is a risk if you answered yes to any of those questions.

Recommended Secure Coding Practices

- It is recommended to use HTTPS everywhere so setting the secure flag to true should be the default behaviour when creating cookies.
- Set the secure flag to true for session-cookies.

Sensitive Code Example

In php.ini you can specify the flags for the session cookie which is security-sensitive:

```
session.cookie_secure = 0; // Sensitive: this security-sensi
```

Same thing in PHP code:

```
session_set_cookie_params($lifetime, $path, $domain, false);  
// Sensitive: this security-sensitive session cookie is crea
```

If you create a custom security-sensitive cookie in your PHP code:

```
$value = "sensitive data";  
setcookie($name, $value, $expire, $path, $domain, false); /
```

By default setcookie and setrawcookie functions set the sixth argument / secure flag to false:

```
$value = "sensitive data";  
setcookie($name, $value, $expire, $path, $domain); // Sensi  
setrawcookie($name, $value, $expire, $path, $domain); // Se
```

Compliant Solution

```
session.cookie_secure = 1; // Compliant: the sensitive cooki
```

An open curly brace should be located at the end of a line

 Code Smell


Tabulation characters should not be used

 Code Smell

Method and function names should comply with a naming convention

 Code Smell

Creating cookies with broadly defined "domain" flags is security-sensitive

 Security Hotspot

```
session_set_cookie_params($lifetime, $path, $domain, true);
```

```
$value = "sensitive data";  
setcookie($name, $value, $expire, $path, $domain, true); //  
setrawcookie($name, $value, $expire, $path, $domain, true);/
```

#### See

- [OWASP Top 10 2021 Category A4](#) - Insecure Design
- [OWASP Top 10 2021 Category A5](#) - Security Misconfiguration
- [OWASP Top 10 2017 Category A3](#) - Sensitive Data Exposure
- [MITRE, CWE-311](#) - Missing Encryption of Sensitive Data
- [MITRE, CWE-315](#) - Cleartext Storage of Sensitive Information in a Cookie
- [MITRE, CWE-614](#) - Sensitive Cookie in HTTPS Session Without 'Secure' Attribute
- [SANS Top 25](#) - Porous Defenses

Available In:

sonarcloud  | sonarqube 