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

side effects should not be ignored
Bug

Values should not be uselessly incremented
Bug

Related "if/else if" statements and "cases" in a "switch" should not have the same condition
Bug

Objects should not be created to be dropped immediately without being used
Bug

Identical expressions should not be used on both sides of a binary operator
Bug

All code should be reachable
Bug

Loops with at most one iteration should be refactored
Bug

Short-circuit logic should be used to prevent null pointer dereferences in conditionals
Bug

Variables should not be self-assigned
Bug

Useless "if(true) {...}" and "if(false){...}" blocks should be removed
Bug

All "catch" blocks should be able to catch exceptions
Bug

## Disabling CSRF protections is security-sensitive

Analyze your code

Security Hotspot  Critical  cwe sans-top25 owasp

A cross-site request forgery (CSRF) attack occurs when a trusted user of a web application can be forced, by an attacker, to perform sensitive actions that he didn't intend, such as updating his profile or sending a message, more generally anything that can change the state of the application.

The attacker can trick the user/victim to click on a link, corresponding to the privileged action, or to visit a malicious web site that embeds a hidden web request and as web browsers automatically include cookies, the actions can be authenticated and sensitive.

### Ask Yourself Whether

- The web application uses cookies to authenticate users.
- There exist sensitive operations in the web application that can be performed when the user is authenticated.
- The state / resources of the web application can be modified by doing HTTP POST or HTTP DELETE requests for example.

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

### Recommended Secure Coding Practices

- Protection against CSRF attacks is strongly recommended:
  - to be activated by default for all unsafe HTTP methods.
  - implemented, for example, with an unguessable CSRF token
- Of course all sensitive operations should not be performed with safe HTTP methods like `GET` which are designed to be used only for information retrieval.

### Sensitive Code Example

For Laravel VerifyCsrfToken middleware

```
use Illuminate\Foundation\Http\Middleware\VerifyCsrfTok

class VerifyCsrfToken extends Middleware
{
    protected $except = [
        'api/*'
    ]; // Sensitive; disable CSRF protection for a list
}
```

For Symfony Forms

```
use Symfony\Bundle\FrameworkBundle\Controller\AbstractC

class Controller extends AbstractController {

  public function action() {
```

Constructing arguments of system commands from user input is security-sensitive

Security Hotspot

Allowing unfiltered HTML content in WordPress is security-sensitive

Security Hotspot

Allowing unauthenticated database repair in WordPress is security-sensitive

Security Hotspot

Allowing all external requests from a WordPress server is security-sensitive

Security Hotspot

```
    $this->createForm('', null, [
      'csrf_protection' => false, // Sensitive; disable
    ]);
  }
}
```

**Compliant Solution**

For Laravel VerifyCsrfToken middleware

```
use Illuminate\Foundation\Http\Middleware\VerifyCsrfTok

class VerifyCsrfToken extends Middleware
{
    protected $except = []; // Compliant
}
```

Remember to add @csrf blade directive to the relevant forms when removing an element from $except. Otherwise the form submission will stop working.

For Symfony Forms

```
use Symfony\Bundle\FrameworkBundle\Controller\AbstractC

class Controller extends AbstractController {

  public function action() {
    $this->createForm('', null, []); // Compliant; CSRF
  }
}
```

**See**

- OWASP Top 10 2021 Category A1 - Broken Access Control
- MITRE, CWE-352 - Cross-Site Request Forgery (CSRF)
- OWASP Top 10 2017 Category A6 - Security Misconfiguration
- OWASP: Cross-Site Request Forgery
- SANS Top 25 - Insecure Interaction Between Components

Available In:

sonarcloud sonarqube