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...

Resource Sharing policy is security-sensitive

Security Hotspot

Delivering code in production with debug features activated is security-sensitive

Security Hotspot

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

Security Hotspot

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

Security Hotspot

Using hardcoded IP addresses is security-sensitive

Security Hotspot

Regular expression quantifiers and character classes should be used concisely

Code Smell

Character classes should be preferred over reluctant quantifiers in regular expressions

Code Smell

A subclass should not be in the same "catch" clause as a parent class

Code Smell

Jump statements should not be redundant

Code Smell

"catch" clauses should do more than rethrow

Code Smell

"&&" and "||" should be used

Code Smell

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

Analyze your code

Bug Major

Using the same value on either side of a binary operator is almost always a mistake. In the case of logical operators, it is either a copy/paste error and therefore a bug, or it is simply wasted code, and should be simplified. In the case of bitwise operators and most binary mathematical operators, having the same value on both sides of an operator yields predictable results, and should be simplified.

Noncompliant Code Example

```
if ( $a == $a ) { // always true
  doZ();
}
if ( $a != $a ) { // always false
  doY();
}
if ( $a == $b && $a == $b ) { // if the first one is tr
  doX();
}
if ( $a == $b || $a == $b ) { // if the first one is tr
  doW();
}

$j = 5 / 5; //always 1
$k = 5 - 5; //always 0
```

Exceptions

Left-shifting 1 onto 1 is common in the construction of bit masks, and is ignored.

```
$i = 1 << 1; // Compliant
$j = $a << $a; // Noncompliant
```

See

- {rule:php:S1656} - Implements a check on =.

Available In:

sonarlint sonarcloud sonarqube

Boolean checks should not be inverted

Code Smell

Local variables should not be declared and then immediately returned or thrown

Code Smell

Unused local variables should be removed

Code Smell

"switch" statements should have at least 3 "case" clauses

Code Smell