




 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



JavaScript static code analysis

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

All rules 285

 Vulnerability 29

 Bug 62












 Security Hotspot 43

 Code Smell 151

 Quick Fix 41

Tags ▾

Search by name... 

Variables should not be shadowed
 Code Smell
Redundant pairs of parentheses should be removed
 Code Smell
Nested blocks of code should not be left empty
 Code Smell
Functions should not have too many parameters
 Code Smell
OS commands should not be vulnerable to argument injection attacks
 Vulnerability
Repeated patterns in regular expressions should not match the empty string
 Bug
Empty collections should not be accessed or iterated
 Bug
"delete" should be used only with object properties
 Bug
"with" statements should not be used
 Bug
Function parameters, caught exceptions and foreach variables' initial values should not be ignored
 Bug
Forwarding client IP address is security-sensitive
 Security Hotspot
Allowing confidential information to

Non-existent operators '+=', '=-' and '!=' should not be used

Analyze your code

 Bug

 Major

 Quick Fix

The use of operators pairs (+=, -= or !=) where the reversed, single operator was meant (+, - or !) will compile and run, but not produce the expected results.

This rule raises an issue when +=, -= and != are used without any space between the two operators and when there is at least one whitespace after.

Noncompliant Code Example

```
let target -=5;
let num = 3;

target -= num; // Noncompliant; target = -3. Is that really
target += num; // Noncompliant; target = 3
```





Compliant Solution

```
let target = -5;
let num = 3;

target = -num; // Compliant; intent to assign inverse value
target += num;
```

Available In:

sonarlint  | sonarcloud  | sonarqube 

<b>be logged is security-sensitive</b>  Security Hotspot
<b>Allowing browsers to perform DNS prefetching is security-sensitive</b>  Security Hotspot
<b>Disabling Certificate Transparency monitoring is security-sensitive</b>  Security Hotspot
<b>Disabling Strict-Transport-Security policy is security-sensitive</b>  Security Hotspot
<b>Disabling strict HTTP no-referrer</b>