




 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... 🔍

Boolean checks should not be inverted

Code Smell

Deprecated APIs should not be used

Code Smell

Wrapper objects should not be used for primitive types

Code Smell

Multiline string literals should not be used

Code Smell

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

Code Smell

Unused local variables and functions should be removed

Code Smell

Function call arguments should not start on new lines

Code Smell

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

Code Smell

A "while" loop should be used instead of a "for" loop

Code Smell

Unnecessary imports should be removed

Code Smell

Return of boolean expressions should not be wrapped into an "if-then-else" statement

Code Smell

### Using shell interpreter when executing OS commands is security-sensitive

Analyze your code

Security Hotspot Major ? cwe owasp sans-top25

Arbitrary OS command injection vulnerabilities are more likely when a shell is spawned rather than a new process, indeed shell meta-chars can be used (when parameters are user-controlled for instance) to inject OS commands.

#### Ask Yourself Whether

- OS command name or parameters are user-controlled.

There is a risk if you answered yes to this question.

#### Recommended Secure Coding Practices

Use functions that don't spawn a shell.

#### Sensitive Code Example

```
const cp = require('child_process');

// A shell will be spawn in these following cases:
cp.exec(cmd); // Sensitive
cp.execSync(cmd); // Sensitive

cp.spawn(cmd, { shell: true }); // Sensitive
cp.spawnSync(cmd, { shell: true }); // Sensitive
cp.execFile(cmd, { shell: true }); // Sensitive
cp.execFileSync(cmd, { shell: true }); // Sensitive
```

#### Compliant Solution



```
const cp = require('child_process');





cp.spawnSync("/usr/bin/file.exe", { shell: false }); // Comp
```

#### See

- [OWASP Top 10 2021 Category A3](#) - Injection
- [OWASP Top 10 2017 Category A1](#) - Injection
- [MITRE, CWE-78](#) - Improper Neutralization of Special Elements used in an OS Command
- [SANS Top 25](#) - Insecure Interaction Between Components

Available In:

sonarcloud  sonarqube 

<b>Boolean literals should not be used in comparisons</b>  Code Smell
<b>Extra semicolons should be removed</b>  Code Smell
<b>Class names should comply with a naming convention</b>  Code Smell
<b>Track uses of "TODO" tags</b>  Code Smell
<b>Web SQL databases should not be used</b>

© 2008-2022 SonarSource S.A., Switzerland. All content is copyright protected. SONAR, SONARSOURCE, SONARLINT, SONARQUBE and SONARCLOUD are trademarks of SonarSource S.A. All other trademarks and copyrights are the property of their respective owners. All rights are expressly reserved.  
[Privacy Policy](#)