




 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

Object literal shorthand syntax should be used	Code Smell
Strings and non-strings should not be added	Code Smell
Object literal syntax should be used	Code Smell
"undefined" should not be assigned	Code Smell
Trailing commas should not be used	Code Smell
Array constructors should not be used	Code Smell
Quotes for string literals should be used consistently	Code Smell
Statements should end with semicolons	Code Smell
Comments should not be located at the end of lines of code	Code Smell
Loops should not contain more than a single "break" or "continue" statement	Code Smell
Variable, property and parameter names should comply with a naming convention	Code Smell
Lines should not end with trailing whitespaces	Code Smell

OS commands should not be vulnerable to argument injection attacks

Analyze your code

Vulnerability

Minor

injection cwe owasp sans-top25

Applications that allow execution of operating system commands from user-controlled data should control the arguments passed to the command, otherwise an attacker can inject additional arbitrary arguments which can change the behavior of the command.

User-controlled arguments should be sanitized by neutralizing argument delimiters (eg: ' , space, -) and thus preventing injection of unwanted additional arguments. A single user-controlled argument may still lead to vulnerabilities if it corresponds to a dangerous option supported by the command, such as -exec available with **find**, in that case, mark end of option processing on the command line using -- (double-dash) or restrict the options to only trusted values.

Noncompliant Code Example

When using the **execa** command function, arguments are defined next to the command as a string which can lead to arguments injection:

```
const execa = require('execa');

async function (req, res) {
  const cmd = "ls -la "+req.query.arg;

  const {stdout} = await execa.command(cmd); // Noncompliant
};
```

Compliant Solution

Prefer the **execa** function where the list of command arguments are defined as elements of an array:

```
const execa = require('execa');

async function (req, res) {
  const arg = req.query.arg;

  const {stdout} = await execa("ls", ["-la", arg]); // Compliant
};
```


See

- OWASP Top 10 2021 Category A3 - Injection
- OWASP OS Command Injection Defense [Cheat Sheet](#)
- OWASP Top 10 2017 Category A1 - Injection
- MITRE, CWE-20 - Improper Input Validation
- MITRE, CWE-88 - Argument Injection or Modification
- SANS Top 25 - Insecure Interaction Between Components


Available In:

Developer


JavaScript static code analysis: OS commands should not be vulnerable to argument injection attacks

sonarcloud | sonarqube  Edition


Files should contain an empty newline at the end

 Code Smell


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

 Code Smell

Tabulation characters should not be used

 Code Smell

Function and method names should comply with a naming convention

 Code Smell

© 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](#)