




 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 rules285

Vulnerability29

Bug62

Security Hotspot43

Code Smell151

Quick Fix41

Tags ▾

Search by name... 🔍

Callbacks of array methods should have return statements

Bug

Loops should not be infinite

Bug

Disabling Vue.js built-in escaping is security-sensitive

Security Hotspot

Disabling Angular built-in sanitization is security-sensitive

Security Hotspot

Hard-coded credentials are security-sensitive

Security Hotspot

Function returns should not be invariant

Code Smell

Assertions should be complete

Code Smell

Variables should be declared explicitly

Code Smell

Tests should include assertions

Code Smell

"future reserved words" should not be used as identifiers

Code Smell

Octal values should not be used

Code Smell

Switch cases should end with an unconditional "break" statement

Code Smell

"switch" statements should not

HTTP request redirections should not be open to forging attacks

Analyze your code

VulnerabilityBlocker🔍injection cwe sans-top25 owasp

User-provided data, such as URL parameters, POST data payloads, or cookies, should always be considered untrusted and tainted. Applications performing HTTP redirects based on tainted data could enable an attacker to redirect users to a malicious site to, for example, steal login credentials.

This problem could be mitigated in any of the following ways:

- Validate the user-provided data based on an allowlist and reject input not matching.
- Redesign the application to not perform redirects based on user-provided data.

Noncompliant Code Example

```
function redirect(req, res) {
  const url = req.query.url; // user-controlled input

  res.redirect(url); // Noncompliant
}

function setLocationHeader(req, res) {
  const url = req.query.url; // user-controlled input

  res.location(url); // Noncompliant
  res.sendStatus(302);
}
```

Compliant Solution

Validate the URL with an allowlist:

```
function isValidUrl(url) {
  if(url.startsWith("https://www.safe.com/")) {
    return true;
  }

  return false;
}

function redirect(req, res) {
  const url = req.query.url; // user-controlled input







  if(isValidUrl(url)) {
    res.redirect(url); // Compliant
  }
}
```

See

- [OWASP Top 10 2021 Category A1](#) - Broken Access Control

https://rules.sonarsource.com/javascript/RSPEC-5146

1/2

<div>contain non-case labels</div> <div> Code Smell</div>	<div><ul style="list-style-type: none">• OWASP Top 10 2017 Category A5 - Broken Access Control• MITRE, CWE-20 - Improper Input Validation• MITRE, CWE-601 - URL Redirection to Untrusted Site ('Open Redirect')• SANS Top 25 - Risky Resource Management</div> <div>Available In:</div> <div><div>sonarcloud</div><div></div><div>sonarqube</div><div></div><div>Developer Edition</div></div>
<div>A new session should be created during user authentication</div> <div> Vulnerability</div>	
<div>JWT should be signed and verified with strong cipher algorithms</div> <div> Vulnerability</div>	
<div>Cipher algorithms should be robust</div> <div> Vulnerability</div>	
<div>Encryption algorithms should be used with secure mode and padding</div>	

© 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](#)