

Go =GO

5 HTML

Java

**JavaScript** 

Kotlin

Kubernetes Objective C

PHP

PL/I

PL/SQL

Python

**RPG** 

Ruby

Scala

**Swift** 

Terraform

Text

**TypeScript** 

T-SQL

**VB.NET** 

VB6

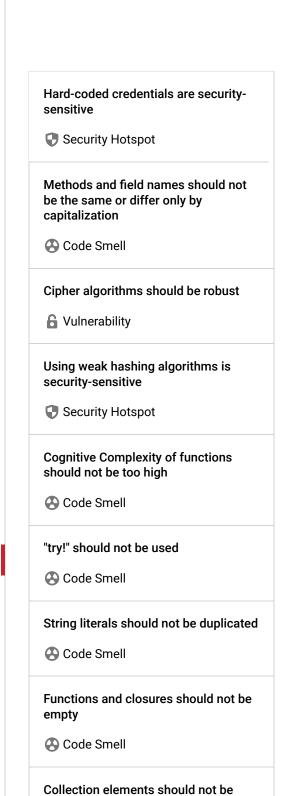
XML



# Swift static code analysis

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

**∄** Bug (14) 6 Vulnerability 3 Security Hotspot 3 Code Smell (99) All rules (119)



replaced unconditionally

Collection sizes comparisons should

All branches in a conditional structure

Infix operators that end with "=" should

update their left operands

Precedence and associativity of standard operators should not be

should not have exactly the same

👬 Bug

make sense

implementation

📆 Bug

📆 Bug

👚 Bug

changed

🖷 Bug

Hard-coded credentials are security-sensitive

Security Hotspot
• Blocker ②

are distributed or that are open-source.

Tags

Analyze your code

Search by name...

🗣 cwe sans-top25 owasp

Because it is easy to extract strings from an application source code or binary, credentials should not be hard-coded. This is particularly true for applications that

In the past, it has led to the following vulnerabilities:

- CVE-2019-13466
- CVE-2018-15389

Credentials should be stored outside of the code in a configuration file, a database, or a management service for secrets.

This rule flags instances of hard-coded credentials used in database and LDAP connections. It looks for hard-coded credentials in connection strings, and for variable names that match any of the patterns from the provided list.

It's recommended to customize the configuration of this rule with additional credential words such as "oauthToken", "secret", ...

### **Ask Yourself Whether**

- Credentials allow access to a sensitive component like a database, a file storage, an API or a service.
- Credentials are used in production environments.
- Application re-distribution is required before updating the credentials.

There is a risk if you answered yes to any of those questions.

### **Recommended Secure Coding Practices**

- Store the credentials in a configuration file that is not pushed to the code repository.
- Store the credentials in a database.
- Use your cloud provider's service for managing secrets.
- If a password has been disclosed through the source code: change it.

### **Sensitive Code Example**

```
let postData = "username=Steve&password=123456".data(using: .
var request = URLRequest(url: url)
request.HTTPBody = postData
```

## **Compliant Solution**

```
let postData = "username=\(getEncryptedUser())&password=\(get
var request = URLRequest(url: url)
request.HTTPBody = postData
```

### See

- OWASP Top 10 2021 Category A7 Identification and Authentication Failures
- OWASP Top 10 2017 Category A2 Broken Authentication
- MITRE, CWE-798 Use of Hard-coded Credentials
- MITRE, CWE-259 Use of Hard-coded Password • SANS Top 25 - Porous Defenses
- Derived from FindSecBugs rule Hard Coded Password

Return values from functions without side effects should not be ignored Rug Bug Related "if/else if" statements and "cases" in a "switch" should not have the same condition 📆 Bug Identical expressions should not be used on both sides of a binary operator 📆 Bug All code should be reachable Rug Bug Loops with at most one iteration should be refactored Rug Bug "IBInspectable" should be used correctly Code Smell Functions should not have identical implementations Code Smell Ternary operators should not be nested Code Smell

Closure expressions should not be

Backticks should not be used around

nested too deeply

Code Smell

Available In:

sonarcloud o Sonarqube Developer Edition

© 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