



CSS

Flex

XGo

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

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

Tags

Hard-coded credentials are securitysensitive Security Hotspot Methods and field names should not be the same or differ only by capitalization Code Smell Cipher algorithms should be robust Vulnerability Using weak hashing algorithms is security-sensitive

Security Hotspot

Cognitive Complexity of functions

Code Smell

"try!" should not be used

should not be too high

Code Smell

String literals should not be duplicated

Code Smell

Functions and closures should not be empty

Code Smell

Collection elements should not be replaced unconditionally

📆 Bug

Collection sizes comparisons should make sense

👬 Bug

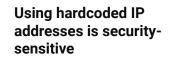
All branches in a conditional structure should not have exactly the same implementation

📆 Bug

Infix operators that end with "=" should update their left operands

🖷 Bug

Precedence and associativity of standard operators should not be changed



Analyze your code

Security Hotspot
Minor

owasp

Search by name...

Hardcoding IP addresses is security-sensitive. It has led in the past to the following vulnerabilities:

- CVE-2006-5901
- CVE-2005-3725

Today's services have an ever-changing architecture due to their scaling and redundancy needs. It is a mistake to think that a service will always have the same IP address. When it does change, the hardcoded IP will have to be modified too. This will have an impact on the product development, delivery, and deployment:

- The developers will have to do a rapid fix every time this happens, instead of having an operation team change a configuration file.
- It misleads to use the same address in every environment (dev, sys, qa,

Last but not least it has an effect on application security. Attackers might be able to decompile the code and thereby discover a potentially sensitive address. They can perform a Denial of Service attack on the service, try to get access to the system, or try to spoof the IP address to bypass security checks. Such attacks can always be possible, but in the case of a hardcoded IP address solving the issue will take more time, which will increase an attack's impact.

Ask Yourself Whether

The disclosed IP address is sensitive, e.g.:

- Can give information to an attacker about the network topology.
- It's a personal (assigned to an identifiable person) IP address.

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

Recommended Secure Coding Practices

Don't hard-code the IP address in the source code, instead make it configurable with environment variables, configuration files, or a similar approach. Alternatively, if confidentially is not required a domain name can be used since it allows to change the destination quickly without having to rebuild the software.

Sensitive Code Example

let host = Host(address: "192.168.12.42")

Compliant Solution

let host = Host(address: configuration.ipAddress)

Exceptions

No issue is reported for the following cases because they are not considered

• Loopback addresses 127.0.0.0/8 in CIDR notation (from 127.0.0.0 to



Return values from functions without side effects should not be ignored



Related "if/else if" statements and "cases" in a "switch" should not have the same condition



Identical expressions should not be used on both sides of a binary operator



All code should be reachable



Loops with at most one iteration should be refactored



"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 nested too deeply

Code Smell

Backticks should not be used around

127.255.255.255)

- Broadcast address 255.255.255.255
- Non routable address 0.0.0.0
- Strings of the form 2.5.<number>.<number> as they often match Object Identifiers (OID).

See

- OWASP Top 10 2021 Category A1 Broken Access Control
- OWASP Top 10 2017 Category A3 Sensitive Data Exposure

Available In:





© 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