sonar RULES

Products

Swift static code analysis

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

All rules 119 | Vulnerability 3 | Bug 14 | Security Hotspot 3 | Code Smell 99

Tags | Search by name...

SHA-1 and Message-Digest hash algorithms should not be used in secure contexts

Analyze your code

Vulnerability   Critical

The MD5 algorithm and its successor, SHA-1, are no longer considered secure, because it is too easy to create hash collisions with them. That is, it takes too little computational effort to come up with a different input that produces the same MD5 or SHA-1 hash, and using the new, same-hash value gives an attacker the same access as if he had the originally-hashed value. This applies as well to the other Message-Digest algorithms: MD2, MD4, MD6, HAVAL-128, HMAC-MD5, DSA (which uses SHA-1), RIPEMD, RIPEMD-128, RIPEMD-160, HMACRIPEMD160.

Consider using safer alternatives, such as SHA-256, SHA-512 or SHA-3.

Noncompliant Code Example

```
import CryptoSwift

let bytes:Array<UInt8> = [0x01, 0x02, 0x03]
let digest = input.md5() // Noncompliant
```

Compliant Solution

```
import CryptoSwift

let bytes:Array<UInt8> = [0x01, 0x02, 0x03]
let digest = input.sha256() // Compliant
```

See

- OWASP Top 10 2017 Category A6 - Security Misconfiguration
- MITRE, CWE-328 - Reversible One-Way Hash
- MITRE, CWE-327 - Use of a Broken or Risky Cryptographic Algorithm
- SANS Top 25 - Porous Defenses
- SHAttered - The first concrete collision attack against SHA-1.

Deprecated

This rule is deprecated; use {rule:swift:S4790} instead.

Available In:

sonarlint sonarcloud sonarqube Developer Edition

Bug

Return values from functions without side effects should not be ignored

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

Bug

Loops with at most one iteration should be refactored

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 nested too deeply

Code Smell

Backticks should not be used around