

Text

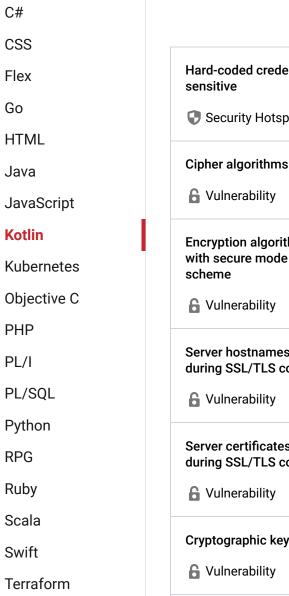
T-SQL

VB.NET

VB6

XML

TypeScript

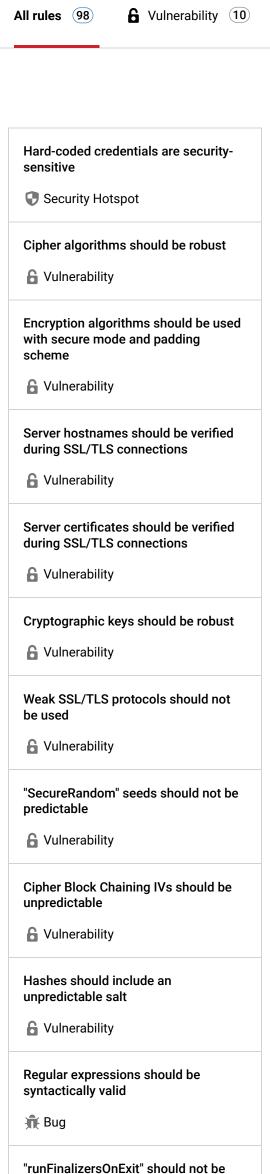




Kotlin static code analysis

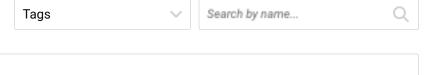
Unique rules to find Bugs, Vulnerabilities, Security Hotspots, and Code Smells in your KOTLIN code

R Bug (17)



called

🖷 Bug



Using pseudorandom number generators (PRNGs) is security-sensitive

Analyze your code

Code Smell (56)

Security Hotspot (15)

cwe owasp

Using pseudorandom number generators (PRNGs) is security-sensitive. For example, it has led in the past to the following vulnerabilities:

- CVE-2013-6386
- CVE-2006-3419
- CVE-2008-4102

When software generates predictable values in a context requiring unpredictability, it may be possible for an attacker to guess the next value that will be generated, and use this guess to impersonate another user or access sensitive information.

Ask Yourself Whether

- the code using the generated value requires it to be unpredictable. It is the case for all encryption mechanisms or when a secret value, such as a password, is hashed.
- the function you use generates a value which can be predicted (pseudo-
- the generated value is used multiple times.
- an attacker can access the generated value.

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

Recommended Secure Coding Practices

- Only use random number generators which are recommended by OWASP or any other trusted organization.
- Use the generated random values only once.
- You should not expose the generated random value. If you have to store it, make sure that the database or file is secure.

Sensitive Code Example

```
val random = Random() // Noncompliant: Random() is not a
val bytes = ByteArray(20)
random.nextBytes(bytes)
```

Compliant Solution

```
val random = SecureRandom() // Compliant
val bytes = ByteArray(20)
random.nextBytes(bytes)
```

See

- OWASP Top 10 2021 Category A2 Cryptographic Failures
- OWASP Top 10 2017 Category A3 Sensitive Data Exposure
- Mobile AppSec Verification Standard Cryptography Requirements
- OWASP Mobile Top 10 2016 Category M5 Insufficient Cryptography
- MITRE, CWE-338 Use of Cryptographically Weak Pseudo-Random Number Generator (PRNG)
- MITRE, CWE-330 Use of Insufficiently Random Values
- MITRE, CWE-326 Inadequate Encryption Strength

"ScheduledThreadPoolExecutor" should not have 0 core threads



Jump statements should not occur in "finally" blocks

Rug Bug

Using clear-text protocols is securitysensitive

Security Hotspot

Accessing Android external storage is security-sensitive

Security Hotspot

Receiving intents is security-sensitive

Security Hotspot

Broadcasting intents is securitysensitive

Security Hotspot

Using weak hashing algorithms is security-sensitive

Security Hotspot

Using pseudorandom number generators (PRNGs) is securitysensitive

Security Hotspot

Empty lines should not be tested with regex MULTILINE flag

Code Smell

Cognitive Complexity of functions should not be too high

Code Smell

- MITRE, CWE-1241 Use of Predictable Algorithm in Random Number Generator
- Derived from FindSecBugs rule Predictable Pseudo Random Number

Available In:

sonarcloud 👌 | sonarqube



© 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