




 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



Java static code analysis

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

All rules 632

Vulnerability 53

Bug 154

Security Hotspot 36


Code Smell 389

Quick Fix 42


Tags ▾

Search by name... 🔍


InputStream.read() implementation should not return a signed byte

 Bug


"compareTo" should not be overloaded

 Bug


"iterator" should not return "this"

 Bug


Map values should not be replaced unconditionally

 Bug


Week Year ("YYYY") should not be used for date formatting

 Bug


Exceptions should not be created without being thrown

 Bug


Collection sizes and array length comparisons should make sense

 Bug


Consumed Stream pipelines should not be reused

 Bug


Intermediate Stream methods should not be left unused

 Bug


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

 Bug

Optional value should only be accessed after calling isPresent()

 Bug

Overrides should match their parent class methods in synchronization

 Bug

Using weak hashing algorithms is security-sensitive

Analyze your code

Security Hotspot

Critical

cwe spring owasp sans-top25

Cryptographic hash algorithms such as MD2, MD4, MD5, MD6, HAVAL-128, HMAC-MD5, DSA (which uses SHA-1), RIPEMD, RIPEMD-128, RIPEMD-160, HMACRIPEMD160 and SHA-1 are no longer considered secure, because it is possible to have collisions (little computational effort is enough to find two or more different inputs that produce the same hash).

Ask Yourself Whether

The hashed value is used in a security context like:

- User-password storage.
- Security token generation (used to confirm e-mail when registering on a website, reset password, etc ...).
- To compute some message integrity.

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

Recommended Secure Coding Practices

Safer alternatives, such as SHA-256, SHA-512, SHA-3 are recommended, and for password hashing, it's even better to use algorithms that do not compute too "quickly", like bcrypt, scrypt, argon2 or pbkdf2 because it slows down brute force attacks.

Sensitive Code Example

```
MessageDigest md1 = MessageDigest.getInstance("SHA"); // Se
MessageDigest md2 = MessageDigest.getInstance("SHA1"); // S
```



Compliant Solution

```
MessageDigest md1 = MessageDigest.getInstance("SHA-512"); //
```

See





- OWASP Top 10 2021 Category A2 - Cryptographic Failures
- OWASP Top 10 2017 Category A3 - Sensitive Data Exposure
- OWASP Top 10 2017 Category A6 - Security Misconfiguration
- Mobile AppSec Verification Standard - Cryptography Requirements
- OWASP Mobile Top 10 2016 Category M5 - Insufficient Cryptography
- MITRE, CWE-1240 - Use of a Risky Cryptographic Primitive
- SANS Top 25 - Porous Defenses

Available In:

sonarcloud  sonarqube 

https://rules.sonarsource.com/java/RSPEC-4790

1/2

 Bug
Value-based classes should not be used for locking  Bug
Expressions used in "assert" should not produce side effects  Bug
"volatile" variables should not be used with compound operators  Bug
"getClass" should not be used for synchronization

© 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](#)