 **sonar** RULES

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 rules632

Vulnerability53

Bug154

Security Hotspot36

Code Smell389

Quick Fix42

Tags

Search by name...

Encryption algorithms should be used with secure mode and padding scheme

Vulnerability

Server hostnames should be verified during SSL/TLS connections

Vulnerability

Insecure temporary file creation methods should not be used

Vulnerability

Passwords should not be stored in plain-text or with a fast hashing algorithm

Vulnerability

Server certificates should be verified during SSL/TLS connections

Vulnerability

Persistent entities should not be used as arguments of "@RequestMapping" methods

Vulnerability

"HttpSecurity" URL patterns should be correctly ordered

Vulnerability

LDAP connections should be authenticated

Vulnerability

Cryptographic keys should be robust

Vulnerability

Weak SSL/TLS protocols should not be used

Vulnerability

"SecureRandom" seeds should not be predictable

Vulnerability

Loops should not be infinite

Analyze your code

BugBlocker?

cert

An infinite loop is one that will never end while the program is running, i.e., you have to kill the program to get out of the loop. Whether it is by meeting the loop's end condition or via a break, every loop should have an end condition.

Noncompliant Code Example

```
for (;;) { // Noncompliant; end condition omitted
    // ...
}

int j;
while (true) { // Noncompliant; end condition omitted
    j++;
}

int k;
boolean b = true;
while (b) { // Noncompliant; b never written to in loop
    k++;
}
```

Compliant Solution

```
int j;
while (true) { // reachable end condition added
    j++;
    if (j == Integer.MIN_VALUE) { // true at Integer.MAX_VAL
        break;
    }
}

int k;
boolean b = true;
while (b) {
    k++;
    b = k < Integer.MAX_VALUE;
}
```

See

- [CERT, MSC01-J](#) - Do not use an empty infinite loop

Available In:





sonarlint

sonarcloud

sonarqube

https://rules.sonarsource.com/java/RSPEC-2189

1/2

<b>Cipher Block Chaining IVs should be unpredictable</b>  Vulnerability
<b>Basic authentication should not be used</b>  Vulnerability
<b>Regular expressions should not be vulnerable to Denial of Service attacks</b>  Vulnerability
<b>"HttpServletRequest.getRequestSession" should not be used</b>  Vulnerability

© 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](#)