Security

Hotspot

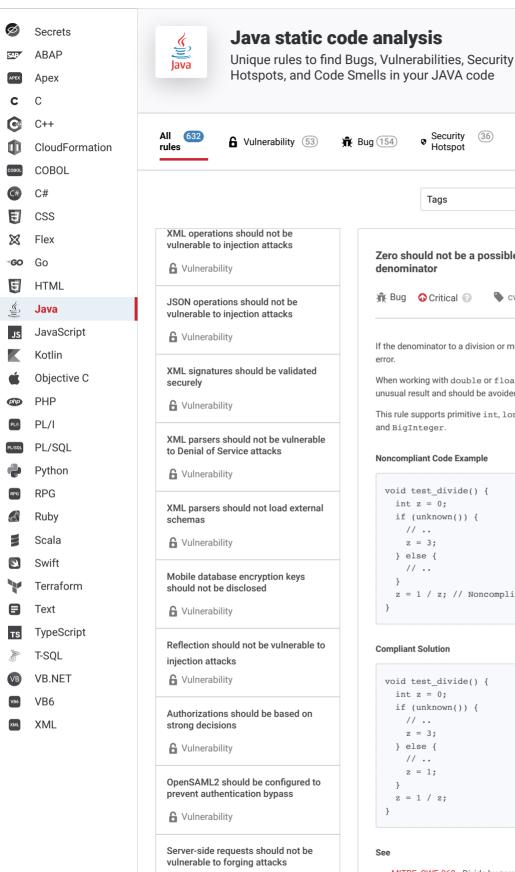
(36)



Products >

O Quick 42 Fix

Q



Vulnerability

R Bug

while they are iterated

Collections should not be modified

Equals method should be overridden in records containing array fields

```
Search by name.
           Tags
Zero should not be a possible
                                                Analyze your code
denominator
Rug Oritical O
                           cwe denial-of-service cert
If the denominator to a division or modulo operation is zero it would result in a fatal
When working with double or float, no fatal error will be raised, but it will lead to
unusual result and should be avoided anyway.
This rule supports primitive int, long, double, float as well as BigDecimal
and BigInteger
Noncompliant Code Example
  void test_divide() {
    int z = 0;
    if (unknown()) {
      // ..
      z = 3;
    } else {
      // ..
      = 1 / z; // Noncompliant, possible division by zero
Compliant Solution
 void test_divide() {
    int z = 0;
    if (unknown()) {
      z = 3;
    } else {
      // ..
      z = 1;
      = 1 / z;
 • MITRE, CWE-369 - Divide by zero
 • CERT, NUM02-J. - Ensure that division and remainder operations do not result in
```

• CERT, INT33-C. - Ensure that division and remainder operations do not result in

sonarlint ⊕ | sonarcloud ↔ | sonarqube

divide-by-zero errors

Available In:

(389)



Reflection should not be used to increase accessibility of records' fields



AssertJ assertions with "Consumer" arguments should contain assertion inside consumers



The regex escape sequence \cX should only be used with characters in the @-_ range



© 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