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...

"PreparedStatement" and "ResultSet" methods should be called with valid indices

Bug

Files opened in append mode should not be used with ObjectOutputStream

Bug

"wait(...)" should be used instead of "Thread.sleep(...)" when a lock is held

Bug

Printf-style format strings should not lead to unexpected behavior at runtime

Bug

Methods "wait(...)", "notify()" and "notifyAll()" should not be called on Thread instances

Bug

Methods should not call same-class methods with incompatible "@Transactional" values

Bug

Recursion should not be infinite

Bug

Loops should not be infinite

Bug

Double-checked locking should not be used

Bug

Resources should be closed

Bug

Hard-coded credentials are security-sensitive

Security Hotspot

XPath expressions should not be vulnerable to injection attacks

Analyze your code

Vulnerability | Blocker | injection cwe owasp cert

User-provided data, such as URL parameters, should always be considered untrusted and tainted. Constructing XPath expressions directly from tainted data enables attackers to inject specially crafted values that changes the initial meaning of the expression itself. Successful XPath injection attacks can read sensitive information from XML documents.

Noncompliant Code Example

```
public boolean authenticate(javax.servlet.http.HttpServletRe
  String user = request.getParameter("user");
  String pass = request.getParameter("pass");

  String expression = "/users/user[@name='" + user + "' and

  // An attacker can bypass authentication by setting user t
  user = "' or 1=1 or ''='";

  return (boolean)xpath.evaluate(expression, doc, XPathConst
}
```

Compliant Solution

```
public boolean authenticate(javax.servlet.http.HttpServletRe
  String user = request.getParameter("user");
  String pass = request.getParameter("pass");

  String expression = "/users/user[@name=$user and @pass=$pa

  xpath.setXPathVariableResolver(v -> {
    switch (v.getLocalPart()) {
      case "user":
        return user;
      case "pass":
        return pass;
      default:
        throw new IllegalArgumentException();
    }
  });

  return (boolean)xpath.evaluate(expression, doc, XPathConst
}
```

See

- OWASP Top 10 2021 Category A3 - Injection
- OWASP Top 10 2017 Category A1 - Injection
- MITRE, CWE-643 - Improper Neutralization of Data within XPath Expressions
- CERT, IDS53-J. - Prevent XPath Injection

Methods returns should not be invariant

Code Smell

"ThreadGroup" should not be used

Code Smell

"clone" should not be overridden

Code Smell

Assertions should be complete

Code Smell

Tests should include assertions

Code Smell

Available In:

sonarcloud | sonarqube Developer Edition