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

Code Smell

Constructors of an "abstract" class should not be declared "public"
Code Smell

Similar tests should be grouped in a single Parameterized test
Code Smell

Tests should be stable
Code Smell

Test methods should not contain too many assertions
Code Smell

AssertJ "assertThatThrownBy" should not be used alone
Code Smell

Character classes in regular expressions should not contain the same character twice
Code Smell

Names of regular expressions named groups should be used
Code Smell

Regexes containing characters subject to normalization should use the CANON_EQ flag
Code Smell

Regular expressions should not be too complicated
Code Smell

JUnit assertTrue/assertFalse should be simplified to the corresponding dedicated assertion
Code Smell

Only one method invocation is expected when testing runtime exceptions

OpenSAML2 should be configured to prevent authentication bypass

Analyze your code

Vulnerability | Major | spring owasp

In 2018, Duo Security found a new vulnerability class that affects SAML-based single sign-on (SSO) systems and this led to the following vulnerabilities being disclosed: CVE-2017-11427, CVE-2017-11428, CVE-2017-11429, CVE-2017-11430, CVE-2018-0489, CVE-2018-7340.

From a specially crafted `<SAMLResponse>` file, an attacker having already access to the SAML system with his own account can bypass the authentication mechanism and be authenticated as another user.

This is due to the fact that SAML protocol rely on XML format and how the underlying XML parser interprets XML comments.

If an attacker manage to change the `<NameID>` field identifying the authenticated user with XML comments, he can exploit the vulnerability.

Here is an example of a potential payload:

```
<SAMLResponse>
  [...]
  <Subject>
    <NameID>admin@domain.com<!---->.evil.com</NameID>
  </Subject>
  [...]
</SAMLResponse>
```

The attacker will manage to generate a valid <SAMLResponse> content with the account "admin@domain.com.evil.com". He will modify it with XML comments to finally be authenticated as "admin@domain.com". To prevent this vulnerability on application using Spring Security SAML relying on OpenSAML2, XML comments should be ignored thanks to the property `ignoreComments` set to `true`.

Noncompliant Code Example

```
import org.opensaml.xml.parse.BasicParserPool;
import org.opensaml.xml.parse.ParserPool;
import org.opensaml.xml.parse.StaticBasicParserPool;

public ParserPool parserPool() {
  StaticBasicParserPool staticBasicParserPool = new StaticBa
  staticBasicParserPool.setIgnoreComments(false); // Noncomp
  return staticBasicParserPool;
}
```

```
public ParserPool parserPool() {
  BasicParserPool basicParserPool = new BasicParserPool();
  basicParserPool.setIgnoreComments(false); // Noncompliant
  return basicParserPool;
}
```

Code Smell

Exception testing via JUnit ExpectedException rule should not be mixed with other assertions

Code Smell

"@Deprecated" code marked for removal should never be used

Code Smell

Vararg method arguments should not be confusing

Code Smell

Whitespace for text block indent

Compliant Solution

```
public ParserPool parserPool() {
  return new StaticBasicParserPool(); // Compliant: "ignoreC
}
```

```
public ParserPool parserPool() {
  return new BasicParserPool();  // Compliant: "ignoreCommen
}
```

See

- OWASP Top 10 2021 Category A6 - Vulnerable and Outdated Components
- OWASP Top 10 2021 Category A7 - Identification and Authentication Failures
- OWASP Top 10 2017 Category A2 - Broken Authentication
- OWASP Top 10 2017 Category A9 - Using Components with Known Vulnerabilities
- Duo Finds SAML Vulnerabilities Affecting Multiple Implementations
- Spring Security SAML and this week's SAML Vulnerability
- Spring Security SAML: Issue #228

Available In:

sonarlint sonarcloud sonarqube