




 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... 🔍

 Bug


DateTimeFormatters should not use mismatched year and week numbers

 Bug


Unicode Grapheme Clusters should be avoided inside regex character classes

 Bug


Case insensitive Unicode regular expressions should enable the "UNICODE\_CASE" flag

 Bug


Assertions should not compare an object to itself

 Bug


Regex alternatives should not be redundant

 Bug


Alternatives in regular expressions should be grouped when used with anchors

 Bug


AssertJ methods setting the assertion context should come before an assertion

 Bug


AssertJ configuration should be applied

 Bug

JUnit5 test classes and methods should not be silently ignored

 Bug

"ThreadLocal" variables should be cleaned up when no longer used

 Bug

Strings and Boxed types should be

### Using publicly writable directories is security-sensitive

Analyze your code

Security Hotspot

Critical

cwe owasp

Operating systems have global directories where any user has write access. Those folders are mostly used as temporary storage areas like /tmp in Linux based systems. An application manipulating files from these folders is exposed to race conditions on filenames: a malicious user can try to create a file with a predictable name before the application does. A successful attack can result in other files being accessed, modified, corrupted or deleted. This risk is even higher if the application runs with elevated permissions.

In the past, it has led to the following vulnerabilities:

- CVE-2012-2451
- CVE-2015-1838

This rule raises an issue whenever it detects a hard-coded path to a publicly writable directory like /tmp (see examples bellow). It also detects access to environment variables that point to publicly writable directories, e.g., TMP and TMPDIR.

- /tmp
- /var/tmp
- /usr/tmp
- /dev/shm
- /dev/mqueue
- /run/lock
- /var/run/lock
- /Library/Caches
- /Users/Shared
- /private/tmp
- /private/var/tmp
- \Windows\Temp
- \Temp
- \TMP

#### Ask Yourself Whether

- Files are read from or written into a publicly writable folder
- The application creates files with predictable names into a publicly writable folder

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





#### Recommended Secure Coding Practices

- Use a dedicated sub-folder with tightly controlled permissions
- Use secure-by-design APIs to create temporary files. Such API will make sure:
  - The generated filename is unpredictable
  - The file is readable and writable only by the creating user ID
  - The file descriptor is not inherited by child processes
  - The file will be destroyed as soon as it is closed

#### Sensitive Code Example

https://rules.sonarsource.com/java/RSPEC-5443

1/2

compared using "equals()"
 Bug
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

```
new File("/tmp/myfile.txt"); // Sensitive
Paths.get("/tmp/myfile.txt"); // Sensitive

java.io.File.createTempFile("prefix", "suffix"); // Sensitive
java.nio.file.Files.createTempDirectory("prefix"); // Sensitive
```

```
Map<String, String> env = System.getenv();
env.get("TMP"); // Sensitive
```

Compliant Solution

```
new File("/myDirectory/myfile.txt"); // Compliant

File.createTempFile("prefix", "suffix", new File("/mySecureD

if(SystemUtils.IS_OS_UNIX) {
    FileAttribute<Set<PosixFilePermission>> attr = PosixFilePe
    Files.createTempFile("prefix", "suffix", attr); // Complia
}
else {
    File f = Files.createTempFile("prefix", "suffix").toFile()
    f.setReadable(true, true);
    f.setWritable(true, true);
    f.setExecutable(true, true);
}
```

See

- [OWASP Top 10 2021 Category A1](#) - Broken Access Control
- [OWASP Top 10 2017 Category A5](#) - Broken Access Control
- [OWASP Top 10 2017 Category A3](#) - Sensitive Data Exposure
- [MITRE, CWE-377](#) - Insecure Temporary File
- [MITRE, CWE-379](#) - Creation of Temporary File in Directory with Incorrect Permissions
- [OWASP, Insecure Temporary File](#)

Available In:  
 | 