Security

Hotspot

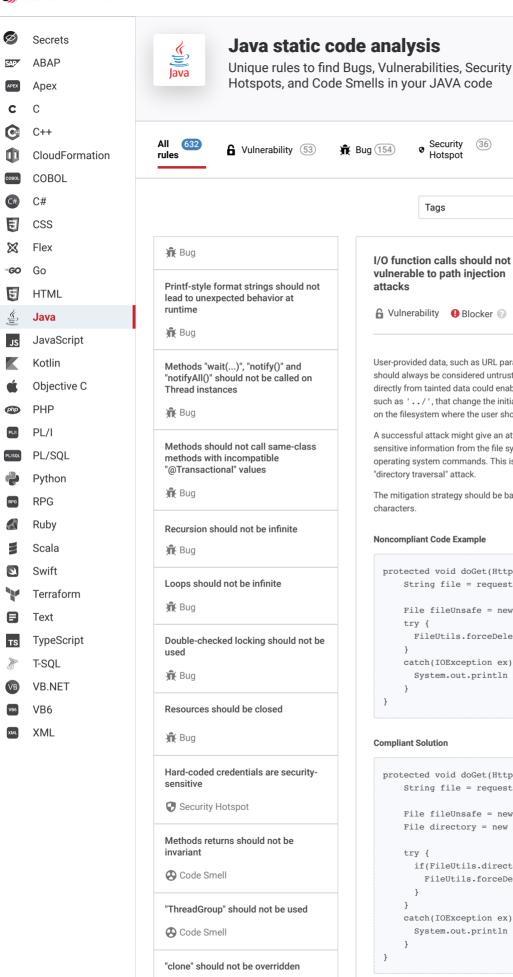
(36)



Products >

O Quick 42

Q



```
Search by name.
           Tags
I/O function calls should not be
vulnerable to path injection
                                                Analyze your code
attacks
■ Vulnerability

    Blocker 

                                    injection cwe owasp sans-top25
User-provided data, such as URL parameters, POST data payloads, or cookies,
should always be considered untrusted and tainted. Constructing file system paths
directly from tainted data could enable an attacker to inject specially crafted values,
such as ' \ldots / ', that change the initial path and, when accessed, resolve to a path
on the filesystem where the user should normally not have access.
A successful attack might give an attacker the ability to read, modify, or delete
sensitive information from the file system and sometimes even execute arbitrary
operating system commands. This is often referred to as a "path traversal" or
"directory traversal" attack.
The mitigation strategy should be based on the whitelisting of allowed paths or
Noncompliant Code Example
  protected void doGet(HttpServletRequest req, HttpServletResp
      String file = request.getParameter("file");
      File fileUnsafe = new File(file);
         FileUtils.forceDelete(fileUnsafe); // Noncompliant
      catch(IOException ex){
         System.out.println (ex.toString());
  }
Compliant Solution
  protected void doGet(HttpServletReguest reg, HttpServletResp
      String file = request.getParameter("file");
      File fileUnsafe = new File(file);
      File directory = new File("/tmp/");
```

if(FileUtils.directoryContains(directory, fileUnsafe))

FileUtils.forceDelete(fileUnsafe); // Compliant

catch(IOException ex){

See

System.out.println (ex.toString());

(389)

Code Smell

Assertions should be complete

Code Smell

Tests should include assertions

Code Smell

Silly bit operations should not be performed

Code Smell

Child class fields should not shadow parent class fields

Code Smell

JUnit test cases should call super

methods

OWASP Top 10 2021 Category A1 - Broken Access Control
OWASP Top 10 2021 Category A3 - Injection
OWASP Top 10 2021 Category A3 - Injection
OWASP Top 10 2017 Category A1 - Injection
OWASP Top 10 2017 Category A5 - Broken Access Control
MITRE, CWE-20 - Improper Input Validation
MITRE, CWE-22 - Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')
MITRE, CWE-99 - Improper Control of Resource Identifiers ('Resource Injection')
MITRE, CWE-641 - Improper Restriction of Names for Files and Other Resources
SANS Top 25 - Risky Resource Management

Available In:

© 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

sonarcloud sonarqube Developer Edition