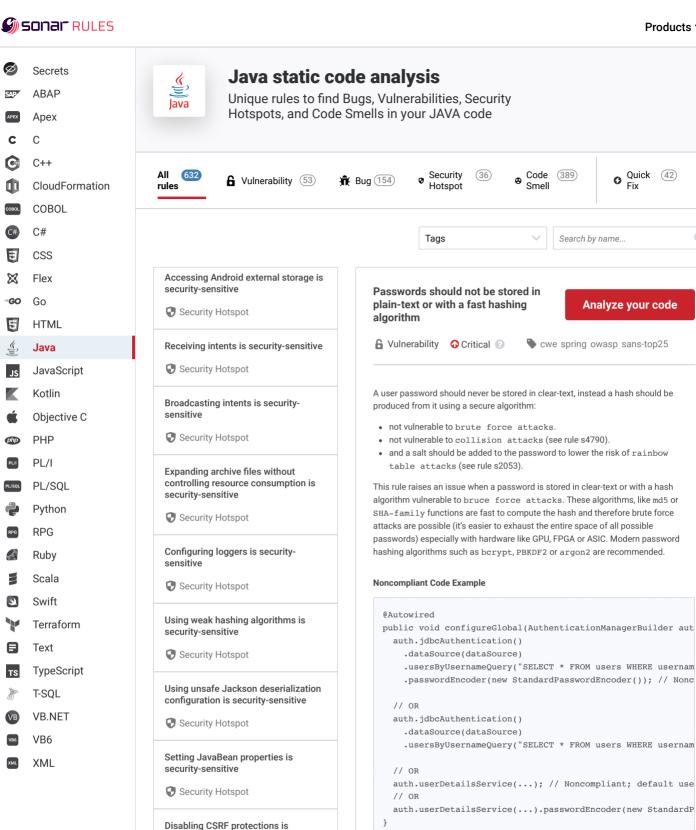


Products ✓



## **Compliant Solution**

```
@Autowired
public void configureGlobal(AuthenticationManagerBuilder aut
 auth.jdbcAuthentication()
    .dataSource(dataSource)
    .usersByUsernameQuery("Select * from users where usernam
    .passwordEncoder(new BCryptPasswordEncoder());
 auth.userDetailsService(null).passwordEncoder(new BCryptPa
```

security-sensitive Security Hotspot

Security Hotspot

Security Hotspot

sensitive

Using non-standard cryptographic

algorithms is security-sensitive

Using pseudorandom number generators (PRNGs) is security-

Mocking all non-private methods of a

Java static code analysis: Passwords should not be stored in plain-text or with a fast hashing algorithm ciass snouid pe avoided

Code Smell

Empty lines should not be tested with regex MULTILINE flag

Code Smell

Methods setUp() and tearDown() should be correctly annotated starting with JUnit4

Code Smell

Class members annotated with "@VisibleForTesting" should not be accessed from production code

Code Smell

- OWASP Top 10 2021 Category A2 Cryptographic Failures
- OWASP Top 10 2021 Category A4 Insecure Design
- OWASP CheatSheet Password Storage Cheat Sheet
- OWASP Top 10 2017 Category A3 Sensitive Data Exposure
- MITRE, CWE-256 Plaintext Storage of a Password
- MITRE, CWE-916 Use of Password Hash With Insufficient Computational Effort
- SANS Top 25 Porous Defenses

Available In:

sonarlint ⊕ | sonarcloud & | sonarqube

© 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