Secrets
ABAP
Apex
C
C++
CloudFormation
COBOL
C#
CSS
Flex
Go
HTML
Java
JavaScript
Kotlin
Kubernetes
Objective C
PHP
PL/I
PL/SQL
Python
RPG
Ruby
Scala
Swift
Terraform
Text
TypeScript
T-SQL
VB.NET
VB6
XML

<XML/>

XML static code analysis

Unique rules to find Bugs and Code Smells in your XML code

All rules 36 | Vulnerability 6 | Bug 5 | Security Hotspot 9 | Code Smell 16

Tags

Search by name...

Basic authentication should not be used
Vulnerability

Hibernate should not update database schemas
Bug

Dependencies should not have "system" scope
Bug

XML files containing a prolog header should start with "<?xml" characters
Bug

Using clear-text protocols is security-sensitive
Security Hotspot

Receiving intents is security-sensitive
Security Hotspot

Restrict access to exported components with appropriate permissions
Vulnerability

"DefaultMessageListenerContainer" instances should not drop messages during restarts
Bug

"SingleConnectionFactory" instances should be set to "reconnectOnException"
Bug

Defining a single permission for read and write access of Content Providers is security-sensitive
Security Hotspot

Allowing application backup is security-sensitive
Security Hotspot

Requesting dangerous Android permissions is security-sensitive
Security Hotspot

Sections of code should not be commented out

Basic authentication should not be used

Analyze your code

Vulnerability | Critical | cwe sans-top25 owasp

Basic authentication's only means of obfuscation is Base64 encoding. Since Base64 encoding is easily recognized and reversed, it offers only the thinnest veil of protection to your users, and should not be used.

Noncompliant Code Example

```
// in web.xml
<web-app  ...>
  <!--  ...  -->
  <login-config>
    <auth-method>BASIC</auth-method>
  </login-config>
</web-app>
```

Exceptions

The rule will not raise any issue if HTTPS is enabled, on any URL-pattern.

```
<web-app  ...>
  <!--  ...  -->
  <security-constraint>
    <web-resource-collection>
      <web-resource-name>HTTPS enabled</web-resource-name>
      <url-pattern>/*</url-pattern>
    </web-resource-collection>
    <user-data-constraint>
      <transport-guarantee>CONFIDENTIAL</transport-guarantee>
    </user-data-constraint>
  </security-constraint>
</web-app>
```

See

- OWASP Top 10 2021 Category A4 - Insecure Design
- OWASP Top 10 2017 Category A3 - Sensitive Data Exposure
- OWASP Web Service Security Cheat Sheet
- MITRE, CWE-522 - Insufficiently Protected Credentials
- SANS Top 25 - Porous Defenses

Available In:

sonarlint | sonarcloud | sonarqube