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 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...

Creating cookies without the "HttpOnly" flag is security-sensitive
Security Hotspot

Deprecated "${pom}" properties should not be used
Code Smell

Track uses of "TODO" tags
Code Smell

EJB interceptor exclusions should be declared as annotations
Code Smell

Track uses of disallowed dependencies
Code Smell

Newlines should follow each element
Code Smell

XML parser failure
Code Smell

Track breaches of an XPath rule
Code Smell

Lines should not be too long
Code Smell

pom elements should be in the recommended order
Code Smell

Artifact ids should follow a naming convention
Code Smell

Group ids should follow a naming convention
Code Smell

"action" mappings should not have too many "forward" entries
Code Smell

Creating cookies without the "HttpOnly" flag is security-sensitive

Analyze your code

Security Hotspot | Minor | cwe sans-top25 privacy owasp

When a cookie is configured with the `HttpOnly` attribute set to *true*, the browser guaranties that no client-side script will be able to read it. In most cases, when a cookie is created, the default value of `HttpOnly` is *false* and it's up to the developer to decide whether or not the content of the cookie can be read by the client-side script. As a majority of Cross-Site Scripting (XSS) attacks target the theft of session-cookies, the `HttpOnly` attribute can help to reduce their impact as it won't be possible to exploit the XSS vulnerability to steal session-cookies.

Ask Yourself Whether

- the cookie is sensitive, used to authenticate the user, for instance a *session-cookie*
- the `HttpOnly` attribute offer an additional protection (not the case for an *XSRF-TOKEN cookie* / CSRF token for example)

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

Recommended Secure Coding Practices

- By default the `HttpOnly` flag should be set to *true* for most of the cookies and it's mandatory for session / sensitive-security cookies.

Sensitive Code Example

```
<session-config>
 <cookie-config>
  <http-only>false</http-only> <!-- Sensitive -->
 </cookie-config>
</session-config>

<session-config>
 <cookie-config> <!-- Sensitive: http-only tag is missing defaulting to false -->
 </cookie-config>
</session-config>
```

Compliant Solution

```
<session-config>
 <cookie-config>
  <http-only>true</http-only> <!-- Compliant -->
 </cookie-config>
</session-config>
```

See

- OWASP Top 10 2021 Category A5 - Security Misconfiguration
- OWASP HttpOnly
- OWASP Top 10 2017 Category A7 - Cross-Site Scripting (XSS)
- MITRE, CWE-1004 - Sensitive Cookie Without 'HttpOnly' Flag
- SANS Top 25 - Insecure Interaction Between Components
- Derived from FindSecBugs rule HTTPONLY_COOKIE

Available In:

sonarcloud | sonarqube