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...

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
Code Smell

Track uses of "FIXME" tags
Code Smell

Custom permissions should not be defined in the 'android.permission' namespace
Vulnerability

Having a permissive Cross-Origin Resource Sharing policy is security-sensitive
Security Hotspot

Delivering code in production with debug features activated is security-sensitive
Security Hotspot

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

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

Analyze your code

Security Hotspot | Major | cwe owasp android

`android:permission` is used to set a single permission for both reading and writing data from a content provider. In regard to the Principle of Least Privilege, client applications that consume the content provider should only have the necessary privileges to complete their tasks. As `android:permission` grants both read and write access, it prevents client applications from applying this principle. In practice, it means client applications that require read-only access will have to ask for more privileges than what they need: the content provider will always grant read and write together.

Ask Yourself Whether

- Some client applications consuming the content provider may only require read permission.

There is a risk if you answered yes to this question.

Recommended Secure Coding Practices

- Avoid using `android:permission` attribute alone. Instead `android:readPermission` and `android:writePermission` attributes to define separate read and write permissions.
- Avoid using the same permission for `android:readPermission` and `android:writePermission` attributes.

Noncompliant Code Example

```
<provider
  android:authorities="com.example.app.Provider"
  android:name="com.example.app.Provider"
  android:permission="com.example.app.PERMISSION"
  android:exported="true"/>
```

```
<provider
  android:authorities="com.example.app.Provider"
  android:name="com.example.app.Provider"
  android:readPermission="com.example.app.PERMISSION"
  android:writePermission="com.example.app.PERMISSION"
  android:exported="true"/>
```

Compliant Solution

```
<provider
  android:authorities="com.example.app.MyProvider"
  android:name="com.example.app.MyProvider"
  android:readPermission="com.example.app.READ_PERMISSION"
  android:writePermission="com.example.app.WRITE_PERMISSION"
  android:exported="true"/>
```

See

- developer.android.com - Implementing content provider permissions
- Mobile AppSec Verification Standard - Platform Interaction Requirements
- OWASP Mobile Top 10 2016 Category M1 - Improper platform usage
- OWASP Mobile Top 10 2016 Category M6 - Insecure Authorization
- MITRE, CWE-1220 - Insufficient Granularity of Access Control

Available In:

sonarcloud | sonarqube