




 Secrets


 ABAP


 Apex


 C


 C++


 CloudFormation


 COBOL


 C#


 CSS


 Flex


 Go


 HTML


 **Java**


 JavaScript


 Kotlin


 Objective C


 PHP


 PL/I


 PL/SQL


 Python


 RPG


 Ruby


 Scala


 Swift


 Terraform


 Text


 TypeScript

 T-SQL

 VB.NET

 VB6

 XML



Java static code analysis

Unique rules to find Bugs, Vulnerabilities, Security Hotspots, and Code Smells in your JAVA code

All rules632

Vulnerability53

Bug154

Security Hotspot36


Code Smell389

Quick Fix42


Tags ▾

Search by name... 🔍


production code

 Bug


DateTimeFormatters should not use mismatched year and week numbers

 Bug


Unicode Grapheme Clusters should be avoided inside regex character classes

 Bug


Case insensitive Unicode regular expressions should enable the "UNICODE_CASE" flag

 Bug


Assertions should not compare an object to itself

 Bug


Regex alternatives should not be redundant

 Bug


Alternatives in regular expressions should be grouped when used with anchors

 Bug


AssertJ methods setting the assertion context should come before an assertion

 Bug


AssertJ configuration should be applied

 Bug

JUnit5 test classes and methods should not be silently ignored

 Bug

"ThreadLocal" variables should be cleaned up when no longer used

 Bug

Accessing Android external storage is security-sensitive

Security Hotspot

Critical

cwe sans-top25 android owasp

Storing data locally is a common task for mobile applications. Such data includes files among other things. One convenient way to store files is to use the external file storage which usually offers a larger amount of disc space compared to internal storage.

Files created on the external storage are globally readable and writable. Therefore, a malicious application having the permissions `WRITE_EXTERNAL_STORAGE` or `READ_EXTERNAL_STORAGE` could try to read sensitive information from the files that other applications have stored on the external storage.

External storage can also be removed by the user (e.g when based on SD card) making the files unavailable to the application.

Ask Yourself Whether

Your application uses external storage to:

- store files that contain sensitive data.
- store files that are not meant to be shared with other application.
- store files that are critical for the application to work.

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

Recommended Secure Coding Practices

- Use internal storage whenever possible as the system prevents other apps from accessing this location.
- Only use external storage if you need to share non-sensitive files with other applications.
- If your application has to use the external storage to store sensitive data, make sure it encrypts the files using [EncryptedFile](#).
- Data coming from external storage should always be considered untrusted and should be validated.
- As some external storage can be removed, make sure to never store files on it that are critical for the usability of your application.

Sensitive Code Example

```
import android.content.Context;





public class AccessExternalFiles {

    public void accessFiles(Context context) {
        context.getExternalFilesDir(null); // Sensitive
    }
}
```

Compliant Solution

https://rules.sonarsource.com/java/RSPEC-5324

1/2

Strings and Boxed types should be compared using "equals()"
 Bug
InputStream.read() implementation should not return a signed byte
 Bug
"compareTo" should not be overloaded
 Bug
"iterator" should not return "this"
 Bug
Map values should not be replaced unconditionally

```
import android.content.Context;

public class AccessExternalFiles {

    public void accessFiles(Context context) {
        context.getFilesDir();
    }
}
```

See

- [OWASP Top 10 2021 Category A4](#) - Insecure Design
- [Android Security tips on external file storage](#)
- [Mobile AppSec Verification Standard](#) - Data Storage and Privacy Requirements
- [OWASP Mobile Top 10 2016 Category M2](#) - Insecure Data Storage
- [MITRE, CWE-312](#) - Cleartext Storage of Sensitive Information
- [SANS Top 25](#) - Risky Resource Management
- [SANS Top 25](#) - Porous Defenses

Available In:

