Kotlin static code analysis

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

All rules 98 | Vulnerability 10 | Bug 17 | Security Hotspot 15 | Code Smell 56

Tags | Search by name...

Hard-coded credentials are security-sensitive
Security Hotspot

Cipher algorithms should be robust
Vulnerability

Encryption algorithms should be used with secure mode and padding scheme
Vulnerability

Server hostnames should be verified during SSL/TLS connections
Vulnerability

Server certificates should be verified during SSL/TLS connections
Vulnerability

Cryptographic keys should be robust
Vulnerability

Weak SSL/TLS protocols should not be used
Vulnerability

"SecureRandom" seeds should not be predictable
Vulnerability

Cipher Block Chaining IVs should be unpredictable
Vulnerability

Hashes should include an unpredictable salt
Vulnerability

Regular expressions should be syntactically valid
Bug

"runFinalizersOnExit" should not be called
Bug

Accessing Android external storage is security-sensitive

Analyze your code

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

class AccessExternalFiles {

    fun accessFiles(Context context) {
        context.getExternalFilesDir(null) // Sensitive
    }
}
```

Compliant Solution

```
import android.content.Context
import android.os.Environment

class AccessExternalFiles {
    fun 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:

sonarcloud | sonarqube

"ScheduledThreadPoolExecutor" should not have 0 core threads
Bug

Jump statements should not occur in "finally" blocks
Bug

Using clear-text protocols is security-sensitive
Security Hotspot

Accessing Android external storage is security-sensitive
Security Hotspot

Receiving intents is security-sensitive
Security Hotspot

Broadcasting intents is security-sensitive
Security Hotspot

Using weak hashing algorithms is security-sensitive
Security Hotspot

Using pseudorandom number generators (PRNGs) is security-sensitive
Security Hotspot

Empty lines should not be tested with regex MULTILINE flag
Code Smell

Cognitive Complexity of functions should not be too high
Code Smell