Fortify Security Report

Apr 10, 2013

sohr

Executive Summary

Issues Overview

On Apr 10, 2013, a source code review was performed over the src code base. 2 files, 21 LOC (Executable) were scanned and reviewed for defects that could lead to potential security vulnerabilities. A total of 2 reviewed findings were uncovered during the analysis.

Issues by Fortify Priority Order	
High	2

Recommendations and Conclusions

The Issues Category section provides Fortify recommendations for addressing issues at a generic level. The recommendations for specific fixes can be extrapolated from those generic recommendations by the development group.

Project Summary

Code Base Summary

Code location: C:/ThinkpadR60/Projekte/MobileApps/AndroidApps/src/LocationLeak/LocationLeak/src
Number of Files: 2
Lines of Code: 21
Build Label: <No Build Label>

Scan Information

Scan time: 00:31
SCA Engine version: 5.14.0.0034
Machine Name: TZI-SOHR-T510
Username running scan: sohr

Results Certification

Results Certification Valid

Details:

Results Signature:

SCA Analysis Results has Valid signature

Rules Signature:

There were no custom rules used in this scan

Attack Surface

Attack Surface:
Private Information:
de.ecspride.androidtestapps.locationleak.LocationLeak$MyLocationListener.onLocationChanged

Filter Set Summary

Current Enabled Filter Set:
Security Auditor View

Filter Set Details:

Folder Filters:
If [fortify priority order] contains critical Then set folder to Critical
If [fortify priority order] contains high Then set folder to High
If [fortify priority order] contains medium Then set folder to Medium
If [fortify priority order] contains low Then set folder to Low
Visibility Filters:

Audit Guide Summary

File System Inputs

Hide issues involving file system inputs.
Depending on your system, inputs from files may or may not come from trusted users. AuditGuide can hide issues that are based on data coming from the file system if it is trusted.
Enable if you trust file system inputs.

Filters:
If taint contains file_system Then hide issue
If taint contains constantfile Then hide issue
If taint contains stream Then hide issue
If category is file access race condition Then hide issueTaint from Command-Line Arguments

Hide issues involving taint from command-line arguments.
Depending on your system, inputs from command-line arguments may or may not come from trusted users. AuditGuide can hide issues that are based on data coming from command-line arguments if they are trusted.
Enable if you trust command-line arguments.

Filters:
If taint contains args Then hide issueProperty File Inputs

Hide inputs from properties files.
Depending on your system, inputs from properties files may or may not come from trusted users. AuditGuide can hide issues that are based on data coming from properties files if they are trusted.
Enable if you trust inputs from properties files.

Filters:
If taint contains property Then hide issueEnvironment Variable Inputs

Hide issues involving environment variable inputs.
Depending on your system, inputs from environment variables may or may not come from trusted users. AuditGuide can hide issues that are based on data coming from environment variables if they are trusted.
Enable if you trust environment variable inputs.

Filters:
If taint contains environment Then hide issueJ2EE Bad Practices

Hide warnings about J2EE bad practices.
Depending on whether your application is a J2EE application, J2EE bad practice warnings may or may not apply. AuditGuide can hide J2EE bad practice warnings.
Enable if J2EE bad practice warnings do not apply to your application because it is not a J2EE application.

Filters:
If category contains j2ee Then hide issue
If category is race condition: static database connection Then hide issue

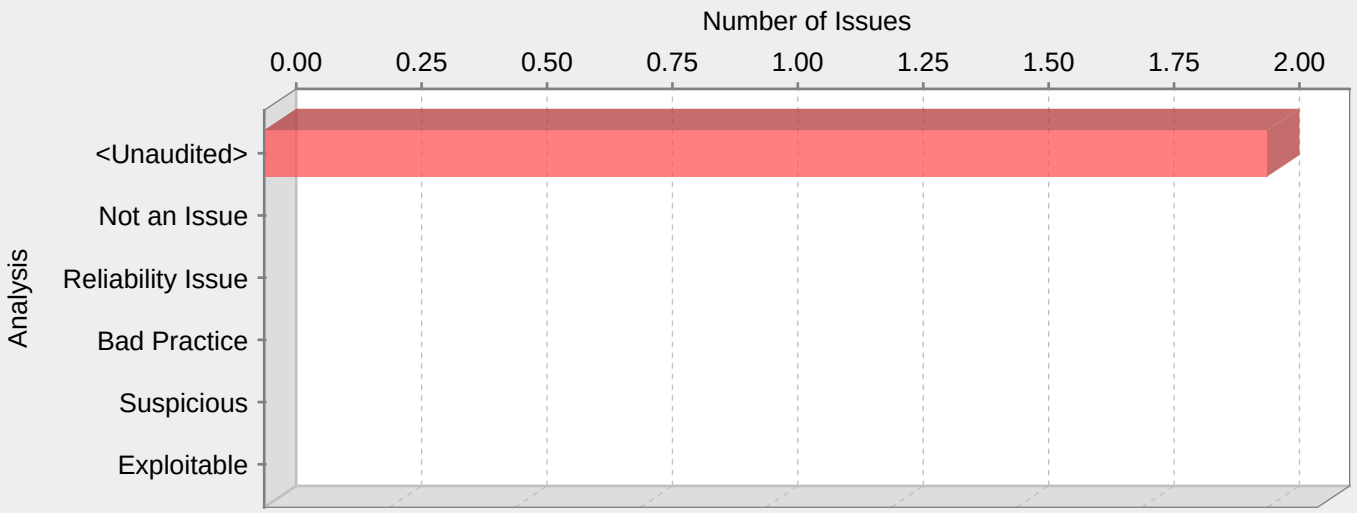
Results Outline

Overall number of results

The scan found 2 issues.

Vulnerability Examples by Category

Category: Privilege Management: Android Location (2 Issues)



Abstract:

The program requests permission to access the device's GPS location.

Explanation:

Access to GPS location information can compromise a user's privacy and personal safety. Programs that require access to GPS location information must be careful to manage it with the utmost caution.

Example 1: The following code requests permission to ACCESS_FINE_LOCATION.

<permission android:name="android.permission.ACCESS_FINE_LOCATION"
android:permissionGroup="android.permission-group.LOCATION"
android:protectionLevel="dangerous"
android:label="@string/permlab_accessFineLocation"
android:description="@string/permdesc_accessFineLocation" />

Recommendations:

Validate that the program requires GPS location information. Rather than granting blanket access to location, determine the appropriate level for the required use. The following examples demonstrate different levels of granularity that can be used when referring to GPS location.

Example 2: For testing purposes, the following code creates mock location providers and does not access any real GPS information.

<permission android:name="android.permission.ACCESS_MOCK_LOCATION"
android:permissionGroup="android.permission-group.LOCATION"
android:protectionLevel="dangerous"
android:label="@string/permlab_accessMockLocation"
android:description="@string/permdesc_accessMockLocation" />

Example 3: The following code uses coarse location information for WiFi or Cell-ID functionality, reserving fine location for GPS and other such uses that demand it.

<permission android:name="android.permission.ACCESS_COARSE_LOCATION"
android:permissionGroup="android.permission-group.LOCATION"
android:protectionLevel="dangerous"
android:label="@string/permlab_accessCoarseLocation"
android:description="@string/permdesc_accessCoarseLocation" />

AndroidManifest.xml, line 27 (Privilege Management: Android Location)

Fortify Priority:	High	**Folder**	High

Kingdom:	Security Features
Abstract:	The program requests permission to to access the device's GPS location on line 27 of AndroidManifest.xml.
Sink:	AndroidManifest.xml:27 null()

```
25          </application>
26
27          <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION">
28          </uses-permission>
```

LocationLeak.java, line 42 (Privilege Management: Android Location)

Fortify Priority:	High	Folder	High
Kingdom:	Security Features		
Abstract:	The program accesses the device's GPS location on line 42 of LocationLeak.java.		
Sink:	LocationLeak.java:42 requestLocationUpdates()		

```
40
41              LocationListener locationListener = new MyLocationListener();
42              locationManager.requestLocationUpdates(LocationManager.GPS_PROVIDER, 5000, 10,
locationListener);
43          }
44
```

Issue Count by Category

Issues by Category	
Privilege Management: Android Location	2

Issue Breakdown by Analysis

Issues by Analysis