MARIO ROBERTO FORTUNATO

2026-08-20

MOBILE SECURITY // ANDROID

PENETRATION TEST REPORT

Android Mobile Application Security Assessment

CLIENT / ENGAGEMENT

[REDACTED]	**[REDACTED]** [APPLICATION: REDACTED] PROJECT ID: 2026_01

PREPARED BY

MARIO ROBERTO FORTUNATO

dev[at]mariorobertofortunato.com

https://mariorobertofortunato.com

CLASSIFICATION

PUBLIC

REDACTED VERSION

00. REPORT CONTENTS

01.	**Assessment Information**	Engagement metadata, Android build details, client and assessor data.
02.	**Engagement Overview**	Purpose, service description, objectives, and assessment intent.
03.	**Process & Methodology**	Reconnaissance, static analysis, dynamic analysis, network/API testing, reporting, retest.
04.	**Scoping & Rules of Engagement**	Authorized scope, constraints, credentials, sensitive assets, and exclusions.
05.	**Executive Summary**	Overall risk assessment, key security observations, strategic recommendations, and finding distribution.
06.	**Vulnerability Overview**	Severity rating criteria and consolidated vulnerability summary table.
07.	**Detailed Findings**	Technical analysis, impact, remediation, evidence, references, and retest status for each confirmed finding.
08.	**Retest Summary**	Remediation verification and closure tracking.

01. ASSESSMENT INFORMATION

Engagement Scope Summary

ENGAGEMENT TIMEFRAME	2026-08-01 - 2026-08-15
PROJECT ID	2026_01
REPORT DATE	2026-08-20
ASSESSMENT TYPE	Android Mobile Application Penetration Test
ENGAGEMENT STATUS	FINAL

Android Target Details

APPLICATION NAME	[REDACTED]
PACKAGE NAME	[com.REDACTED.application]
VERSION / VERSION CODE	[REDACTED] / [REDACTED]
APK / AAB SOURCE	CLIENT BUILD
APK SHA-256	SHA-256 HASH
TEST DEVICE / OS	EMULATOR - Android 14

Report Author / Assessor

NAME / COMPANY	Mario Roberto Fortunato
ROLE	Mobile Application Penetration Tester
CONTACT	dev[at]mariorobertofortunato.com

Client Details

ORGANIZATION	[REDACTED]
PRIMARY CONTACT	[REDACTED - FOUNDER]
CONTACT	[REDACTED] | [REDACTED]

02. ENGAGEMENT OVERVIEW

Mario Roberto Fortunato conducted an Android Mobile Application Penetration Test for [REDACTED]. The assessment is intended to identify and validate exploitable security weaknesses in the authorized application and its supporting mobile-facing interfaces, then provide clear remediation guidance.

Service Description

Mobile application penetration testing simulates realistic attacker techniques against an authorized Android application. The assessment combines (limited) automated tooling with manual review and targeted exploitation so that scanner output is verified and contextualized rather than reported as raw findings.

Assessment Objectives

- Identify vulnerabilities in the Android application package, configuration, storage, inter-process communication, cryptographic use, authentication flows, and runtime behavior.
- Evaluate network communications and mobile-facing API interactions that are reachable through the tested application within the approved scope.
- Validate exploitability and assess both exploitation likelihood and potential impact.
- Provide actionable remediation guidance and evidence that can be reproduced by the client.
- Support optional retesting after remediation to confirm closure of reported issues.

Assessment Approach

Testing prioritized manual verification and evidence-driven reporting. Automated findings were included only after validation, and severity was assigned based on the actual tested context rather than scanner defaults.

03. PROCESS AND METHODOLOGY

Phase 01	**Reconnaissance & Acquisition**	Confirm the authorized build, collect identifiers and hashes, map application components, and document the test environment.
Phase 02	**Static Analysis**	Review the AndroidManifest.xml, resources, decompiled code, secrets, exported components, cryptographic patterns, storage behavior, third-party libraries, and obfuscation controls.
Phase 03	**Dynamic Analysis**	Exercise the application on an authorized test device or emulator, inspect runtime behavior, local storage, logs, IPC, WebViews, root-resilience controls, and sensitive data handling.
Phase 04	**Network & Backend API Analysis**	Observe and manipulate application traffic through an authorized interception setup, assess TLS behavior, authentication, authorization, session handling, and API input validation.
Phase 05	**Exploration & Verification**	Manually reproduce suspected issues, remove false positives, validate prerequisites, and capture evidence sufficient to support impact and remediation.
Phase 06	**Assessment Reporting**	Prioritize confirmed findings by severity, summarize systemic weaknesses, provide remediation guidance, and record evidence and references.
Phase 07	**Optional Retest**	Re-evaluate remediated findings and update closure status, residual risk, and supporting evidence.

04. SCOPING AND RULES OF ENGAGEMENT

Authorized Scope

ANDROID APPLICATION	[REDACTED]
BACKEND / API SCOPE	[REDACTED]
TEST ACCOUNTS	[REDACTED / REDACTED]
TESTING WINDOW	2026-08-01 - 2026-08-15

Constraints and Rules

- AUTHORIZATION: account creation, rooted device, emulators, proxying, hooking.
- PROHIBITED ACTIONS: destructive testing, denial-of-service, third-party systems, real customer data
- No stop condition has been defined.

Sensitive Assets and Processes

Special attention regarding user financial data and subscription to premium features.

Out of Scope

Backend infrastructure, third-party SDK internals, social engineering, physical security

05. EXECUTIVE SUMMARY

[REDACTED] commissioned this assessment to evaluate the security of [REDACTED], a consumer Android application that automatically tracks personal expenses by reading payment notifications from banking and wallet apps installed on the user's device. Because [REDACTED] processes financial transaction data and offers a paid subscription tier, testing focused on three questions that matter to the business, not only to engineering:

- can an attacker read, corrupt, or inject a user's financial data without consent?
- can the subscription revenue model be bypassed?
- does the app's data handling create regulatory or reputational exposure?

Based on the confirmed findings documented in this report, the overall assessed risk is **[HIGH]**, driven primarily by H-01 and H-02: The two most significant issues compound each other. A leftover debugging interface (H-01) shipped in the production build remains externally reachable by any app on the device, allowing arbitrary financial entries to be injected into a user's ledger with no user interaction. Separately, the premium subscription check (H-02) trusts a value stored unencrypted on the device rather than validating entitlement against Google Play, meaning the paid tier can be unlocked without payment.

Mobile Application Risk Rating

The Overall Risk Rating reflects the highest material unresolved risk identified during testing, with additional consideration given to the number of significant findings, their exploitability, affected business functions, and any compounding interactions between vulnerabilities. It is not calculated as an arithmetic average of individual finding scores.

OVERALL RISK RATING: [HIGH]

EXPLOITATION LIKELIHOOD	Info	Low	Medium	High	Critical
Critical	Medium	Medium	High	High	Critical
High	Low	Medium	Medium	**High**	High
Medium	Low	Low	Medium	Medium	High
Low	Info	Low	Low	Medium	Medium
Info	Info	Info	Low		Medium

POTENTIAL IMPACT

Summary of Strengths

- The review of the application’s exported attack surface identified at least one platform-required exported component, [REDACTED], whose externally reachable actions were appropriately constrained and did not expose attacker-controlled data paths or sensitive information.
- No Critical-severity vulnerabilities were identified during the assessment; the overall High rating is driven by a limited set of specific, reproducible implementation weaknesses rather than by a complete breakdown of the application security model.
- The identified issues are concentrated in well-defined components and configurations and can be addressed through comparatively focused engineering changes, reducing the complexity of remediation.

Summary of Weaknesses

- **Excessive trust in client-side state and application-controlled inputs.** Premium entitlement is enforced through a locally persisted flag, while a production-exported QA receiver accepts externally supplied financial data and writes it directly into the user ledger. These issues show insufficient enforcement of trust boundaries for security and business-critical decisions.
- **Inadequate protection of sensitive financial data throughout its local lifecycle.** Transaction information is stored in plaintext, exposed through verbose production logging, and included in unrestricted application backups, creating multiple overlapping paths through which sensitive user data may be extracted.
- **Insufficient production hardening and secret management.** Debug/test functionality remained reachable in the release build, a long-lived backend API key was embedded directly in the APK, and release-specific safeguards such as secret scanning, component validation, and logging removal were not consistently enforced.

Strategic Recommendations

- **Move security-critical trust decisions outside the client.** Treat the Android application as an untrusted execution environment: validate subscription entitlements server-side, replace broadly scoped static credentials with short-lived user-scoped tokens, and ensure externally supplied IPC data is subject to explicit authorization and validation before affecting application state.
- **Establish a consistent sensitive-data protection strategy.** Minimize the amount of financial and personal data persisted on-device, encrypt data that must remain at rest, exclude sensitive stores from backup, and prohibit sensitive values from production logs. These controls should be applied as a unified data-handling policy rather than as isolated fixes.
- **Introduce release-security gates into the build pipeline.** Production builds should automatically fail when they contain debug/test components, unexpected exported entry points, embedded secrets, or prohibited diagnostic logging. Security checks should operate on the final release artifact so that release-only configuration errors are detected before publication.

Finding Distribution

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
0	3	3	1	1

06. VULNERABILITY OVERVIEW

Vulnerability Risk Definition and Criteria

CRITICAL	A severe issue that can enable catastrophic compromise, unrestricted sensitive-data exposure, or similarly extreme impact. Treat as an immediate remediation priority.
HIGH	A serious weakness with substantial security or business impact and realistic exploitation conditions. Remediate promptly.
MEDIUM	A meaningful vulnerability with moderate impact or additional prerequisites. Address after Critical and High items or sooner when business context increases risk.
LOW	A limited-impact weakness, hardening gap, or issue with significant exploitation constraints. Track and remediate proportionally.
INFORMATIONAL	An observation with little or no direct standalone security impact. Record for context, defense-in-depth, or future risk considerations.

Vulnerability Summary Table

ID	FINDING	SEVERITY
H-01	Exported debug/test broadcast receiver allows arbitrary financial-data injection	High
H-02	Client-side premium entitlement enforcement — subscription bypass	High
H-03	Hardcoded backend API key committed to the application package	High
M-01	Unencrypted local storage of financial transaction data	Medium
M-02	Full application backup enabled without data-exclusion rules	Medium
M-03	Verbose production logging of personally identifiable and financial data	Medium
L-01	Uncontrolled resource consumption when importing shared images	Low
I-01	Exported widget broadcast receiver. Reviewed, no exploitable data exposure	Informational

07. DETAILED FINDINGS

Detailed Findings Summary

The following section provides the detailed technical analysis of each confirmed finding identified during the penetration testing assessment. Each finding is documented individually and includes the affected security condition, exploitation context, potential impact, recommended remediation, supporting evidence, and relevant security references or mappings.

Findings are assigned a unique identifier and risk rating to support consistent tracking throughout remediation and optional retesting activities. Where applicable, testing prerequisites, attack conditions, reproduction steps, and additional contextual information are included to ensure that each issue can be independently understood, reproduced, and addressed by the responsible technical teams.

Retest information may be appended to individual findings following remediation in order to document their updated status and any remaining residual risk.

H-01 Exported debug/test broadcast receiver allows arbitrary financial-data injection

Risk Rating: High

Exploitation Likelihood: **HIGH** | Potential Impact: **HIGH**

Description

The notification-listener service responsible for auto-tracking expenses registers a secondary broadcast receiver intended for internal QA that was left exported in the production build. Any other application on the device — no special permissions required — can send it a directed broadcast carrying attacker-controlled merchant, amount, and category values, which are written directly into the user's expense ledger, bypassing the normal notification-parsing pipeline entirely.

```
adb shell am broadcast -a com.REDACTED.TEST_PAYMENT \
  --es merchant "Fake Merchant" --es amount "9999.99"
```

Security Impact

Zero-permission, zero-interaction corruption of the user's financial records; a direct hit to the app's core value proposition and a highly demonstrable finding for any external researcher.

Remediation

- Remove the test/debug receiver from release builds, gated behind a debug build flavor.
- If a QA hook must remain, register it as non-exported with a signature-level permission.
- Add a CI check that fails the release build on exported components matching debug/test naming.

Testing Process / Evidence

[REDACTED]

References / Mapping

N/A

H-02 Client-side premium entitlement enforcement — subscription bypass

Risk Rating: High

Exploitation Likelihood: **HIGH** | Potential Impact: **HIGH**

Description

Access to paid features is decided entirely by a boolean flag persisted locally, trusted for up to 7 days without re-validation against Google Play. Because the application also ships with full data backup enabled (F-05), the flag can be flipped using only adb, without root or device compromise.

```
adb backup -f b.ab -noapk com.REDACTED.application
# edit is_premium -> true inside the extracted preferences file
adb restore b.ab
```

Security Impact

Unauthenticated bypass of the subscription revenue model, exploitable by any user with USB debugging enabled

Remediation

- Validate the Play purchase token server-side rather than trusting the on-device flag as sole source of truth.
- Re-validate entitlement on every app foreground event where connectivity is available.
- Exclude the entitlement store from the auto-backup set as an immediate, low-effort mitigation.

Testing Process / Evidence

[REDACTED]

References / Mapping

N/A

H-03 Hardcoded backend API key committed to the application package

Risk Rating: High

Exploitation Likelihood: **HIGH** | Potential Impact: **HIGH**

Description

Static analysis of the decompiled APK recovered a fully-formed backend API key embedded as a string constant and referenced directly from network-layer client code, rather than being retrieved from a backend-issued, user-scoped session token. Any party who downloads the APK can extract and reuse the key against the production backend.

Security Impact

Depending on backend-side authorization, ranges from unauthorized data access to abuse of paid third-party services billed to the company. Because the key is static and shared across every install, it cannot be revoked for one abusive user without breaking the app for everyone.

Remediation

- Never ship long-lived, broadly-scoped secrets inside the client binary; issue short-lived, user-scoped tokens from an authenticated backend session.
- Add a pre-release CI secret-scanning step against the built artifact, not only the source repository.

Testing Process / Evidence

[REDACTED]

References / Mapping

N/A

M-01 Unencrypted local storage of financial transaction data

Risk Rating: Medium

Exploitation Likelihood: **MEDIUM** | Potential Impact: **HIGH**

Description

Parsed transaction data — merchant names, amounts, categories — is persisted in plaintext across a local preferences store and the application's SQLite database, neither encrypted at rest. A diagnostic log additionally retains raw, unparsed notification text for entries the parser failed to interpret.

Security Impact

Any actor with filesystem access — a rooted device, a lost/resold phone, or a device backup — can read a complete history of the user's spending behavior.

Remediation

- Replace plaintext preferences for sensitive data with androidx.security EncryptedSharedPreferences.
- Encrypt the local database at rest (e.g., SQLCipher).
- Cap the diagnostic log's retention and strip raw notification text before persisting it.

Testing Process / Evidence

[REDACTED]

References / Mapping

N/A

M-02 Full application backup enabled without data-exclusion rules

Risk Rating: Medium

Exploitation Likelihood: **MEDIUM** | Potential Impact: **HIGH**

Description

android:allowBackup is set to true with no accompanying dataExtractionRules to exclude sensitive files. Combined with USB debugging, this allows the entire application sandbox — including the stores covered in H-02 and M-01 — to be extracted and modified using only standard Android SDK platform tools, no root required.

Security Impact

This finding is a force-multiplier — it is what turns H-02 and M-01 from "requires root" into "requires only a USB cable."

Remediation

Set allowBackup="false", or define explicit dataExtractionRules excluding the entitlement store and transaction database.

Testing Process / Evidence

[REDACTED]

References / Mapping

N/A

M-03 Verbose production logging of personally identifiable and financial data

Risk Rating: Medium

Exploitation Likelihood: **MEDIUM** | Potential Impact: **HIGH**

Description

Extensive logging calls throughout the notification-processing pipeline emit full transaction details in cleartext to the system log — parsed amounts, merchant names, and in several places the complete raw source notification text — with no build-time stripping for release builds.

Security Impact

Any connected debugging session or device with adb access enabled can passively capture a live stream of the user's financial activity without touching app storage directly.

Remediation

- Strip or no-op logging calls in release builds via R8/ProGuard rules.
- Never log raw notification content, parsed amounts, or merchant data in code paths shipped to production.

Testing Process / Evidence

[REDACTED]

References / Mapping

N/A

L-01 Uncontrolled resource consumption when importing shared images

Risk Rating: Low

Exploitation Likelihood: **MEDIUM** | Potential Impact: **LOW**

Description

The screenshot-import share target decodes attacker-supplied images with no prior bounds check and no maximum-dimension enforcement, allowing a crafted image shared from any other app to force a large in-memory allocation and crash the application.

Remediation

Decode bounds first, enforce a maximum resolution, and use inSampleSize to downscale before full decode.

Testing Process / Evidence

[REDACTED]

References / Mapping

N/A

I-01 Exported widget broadcast receiver

Risk Rating: Informational

Exploitation Likelihood: **INFO** | Potential Impact: **INFO**

Description

The home-screen widget provider is exported, which is required by the platform and not a defect by itself. Its custom actions were reviewed in detail: neither reads attacker-controlled Intent extras, and the only externally triggerable effect is cycling the widget's displayed view, gated behind the app's own premium-status check. Included to document that the component was specifically analyzed, not flagged by pattern-matching alone.

Remediation

No action required.

Testing Process / Evidence

[REDACTED]

References / Mapping

N/A

08. RETEST SUMMARY

ID	SEVERITY	STATUS	RETEST DATE	NOTES
H-01	High	OPEN	N/A	N/A
H-02	High	OPEN	N/A	N/A
H-03	High	OPEN	N/A	N/A
M-01	Medium	OPEN	N/A	N/A
M-02	Medium	OPEN	N/A	N/A
M-03	Medium	OPEN	N/A	N/A
L-01	Low	OPEN	N/A	N/A
I-01	Informational	OPEN	N/A	N/A

Retest Conclusion

No retest has been scheduled.