CONFIDENTIAL — SAMPLE / DEMONSTRATION

ANDROID APPLICATION PENETRATION TESTING REPORT

NovaSphere Pay — Mobile Application Security Assessment

2026 Illustrative Sample Deliverable

Organization	NovaSphere Digital Technologies, Inc. (Fictional)
Application	NovaSphere Pay — com.novasphere.pay (Fictional)
Prepared By	TMG Security
Assessment Type	Android Application Security Assessment / Penetration Test
Assessment Period	03 August 2026 – 28 August 2026
Report Date	11 September 2026
Classification	CONFIDENTIAL — SAMPLE / DEMONSTRATION

FICTIONAL SAMPLE REPORT

NOT AN ACTUAL CLIENT ASSESSMENT

All findings, evidence, and results are illustrative.

01 DOCUMENT CONTROL

Document Title	Android Application Penetration Testing Report — NovaSphere Pay (Fictional Sample)
Document ID	TMG-AND-NSP-2026-001
Version	1.0
Issue Date	11 September 2026
Assessment Period	03 August 2026 – 28 August 2026
Assessment Type	Android Application Security Assessment / Penetration Test
Client	NovaSphere Digital Technologies, Inc. (Fictional)
Application	NovaSphere Pay — com.novasphere.pay (Fictional)
APK Under Test	NovaSpherePay-release-6.4.2.apk (Fictional)
Backend API	https://api.novasphere-example.com (Fictional, non-resolving)
Auth Service	https://auth.novasphere-example.com (Fictional, non-resolving)
Environment	Fictional staging / controlled assessment environment
Standards Referenced	OWASP MASVS (current), OWASP MASTG v2.0.0, OWASP MASWE v1.0.0
Prepared By	TMG Security Offensive Security Team
Technical Lead	TMG Security — Senior Mobile Security Tester (Fictional Engagement Role)
Reviewed By	TMG Security Quality Assurance Lead
Classification	CONFIDENTIAL — SAMPLE / DEMONSTRATION

Version History

Version	Date	Author	Summary
1.0	11 September 2026	TMG Security Offensive Security Team	Initial issue — fictional sample deliverable.

02 IMPORTANT DISCLAIMER

FICTIONAL SAMPLE REPORT — NOT AN ACTUAL CLIENT ANDROID PENETRATION TEST

This report has been produced solely for demonstration and portfolio purposes on behalf of TMG Security. It is intended to illustrate the structure, technical depth, and professional reporting standard of an Android application penetration testing deliverable built on current OWASP Mobile Application Security (MASVS / MASTG / MASWE) material. To ensure there is no ambiguity regarding the nature of this document, TMG Security sets out the following clarifications:

- NovaSphere Digital Technologies, Inc. is a fictional entity. It does not correspond to any real company, and any resemblance to an actual organization is purely coincidental.
- NovaSphere Pay (package com.novasphere.pay) is a fictional Android application. No real production application was built, published, or accessed.
- The backend domains referenced throughout this report (api.novasphere-example.com, auth.novasphere-example.com) are fictional, non-resolving placeholder domains used only for illustration.
- No real client was engaged and no real client Android application, backend infrastructure, or mobile device fleet was tested.
- No production environment, system, network, cloud resource, or app-store listing of any kind was accessed, scanned, or tested in the creation of this report.
- No real user data, personal information, payment card data, or customer records were accessed, viewed, or processed.
- No real credentials — including device, application, API, signing, or third-party SDK credentials — were used or are disclosed anywhere in this report.
- All APK metadata, manifest excerpts, decompiled code fragments, adb/logcat output, and network traffic captures shown in this report are synthetic and constructed for illustration only.
- All vulnerabilities, findings, risk ratings, CVSS-style scores, dates, and remediation statuses described in this report are fictional and were constructed for demonstration purposes.
- All Proofs of Concept (PoCs) are illustrative, synthetic, and non-destructive; none were executed against a real or live target, real device population, or production backend.
- All screenshots and tool-style evidence panels (Android Studio, APK analyzer, decompiler, MobSF-style dashboard, Burp-style traffic, logcat/adb output) are synthetic mock-ups created specifically for this sample report, not real tool captures.
- No real exploitation occurred at any point during the preparation of this document, and no actual customer impact occurred.
- TMG Security is not affiliated with, endorsed by, or certified by OWASP; references to OWASP MASVS, MASTG, and MASWE describe the testing framework used and do not imply any official certification of NovaSphere Pay or of TMG Security.
- This report does not represent an actual TMG Security client engagement and must not be relied upon as evidence of the security posture of any real organization or application.

Intended Use

This sample report may be shared with prospective clients, partners, and other interested parties to illustrate TMG Security's Android application security testing methodology, technical reporting standards, and evidence-presentation quality. It should not be relied upon for any compliance, legal, contractual, or procurement decision, and does not constitute security advice regarding any real application.

TABLE OF CONTENTS

03 EXECUTIVE SUMMARY

TMG Security has prepared this fictional, illustrative Android Application Penetration Testing report to represent NovaSphere Digital Technologies, Inc., a hypothetical U.S.-based FinTech / digital payments organization headquartered in Austin, Texas, and its fictional flagship consumer application, NovaSphere Pay (package com.novasphere.pay). This section presents a fictional executive-level summary of the simulated assessment — covering static analysis, dynamic analysis, and mobile-API integration testing against current OWASP MASVS, MASTG v2.0.0, and MASWE v1.0.0 material — prepared in the style and format TMG Security would use for an actual client engagement.

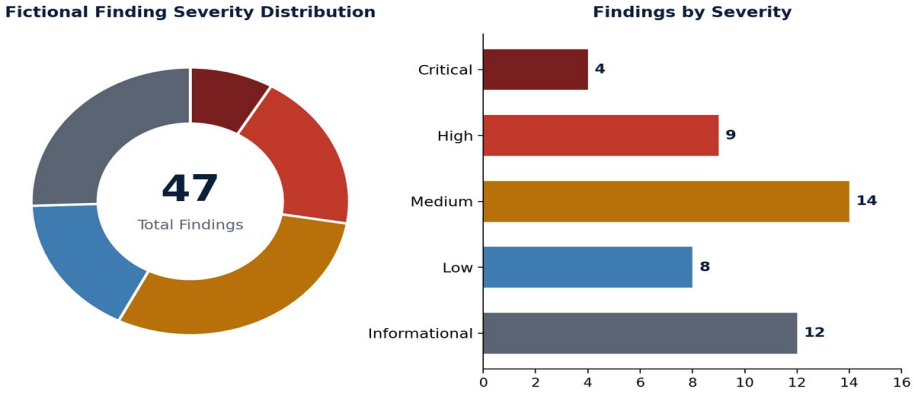
ALL METRICS, FINDINGS, AND SCORES ON THIS PAGE ARE ILLUSTRATIVE SAMPLE DATA

Overall Illustrative Security Posture

REQUIRES REMEDIATION

NovaSphere Pay's simulated Android build demonstrates a number of established security controls — including consistent MFA enforcement, TLS 1.2+ and certificate pinning on the primary payments channel, and a properly signed release build — but the fictional assessment identified material weaknesses concentrated in local data protection, exported-component authorization, mobile-API object-level authorization, and deep-link handling. These require prioritized remediation before the application should be considered ready for a broader trust posture. This illustrative status reflects a sample Android penetration-testing exercise and does not represent a certification or compliance decision, and does not imply any official OWASP certification.

Fictional Sample Dashboard



Illustrative severity distribution across all 47 fictional findings — ILLUSTRATIVE SAMPLE DATA.

Executive Risk Narrative

The fictional assessment identified 47 illustrative findings across NovaSphere Pay's Android client, local attack surface, and mobile-API integration: 4 Critical, 9 High, 14 Medium, 8 Low, and 12 Informational. The four Critical findings concern the most fundamental trust-boundary failures observed: plaintext local storage of authentication tokens with no Keystore-backed encryption, an exported service capable of initiating a fictional funds transfer with no caller-identity validation, broken object-level authorization on a mobile-only statement API endpoint, and an insecure deep-link flow that silently commits a new payee association with no in-app confirmation.

The nine High findings span a WebView JavaScript interface exposing session-token material, a certificate-validation weakness on the telemetry network channel, sensitive authentication data reaching device logs under a QA build configuration, a hardcoded third-party SDK key recoverable through straightforward decompilation, an overly broad FileProvider configuration, a biometric app-unlock gate bypassable on a rooted device because it is not bound to a Keystore cryptographic operation, a token-lifecycle gap in which logout does not trigger server-side revocation, a task-affinity configuration enabling a StrandHogg-style intent-spoofing risk, and a root/runtime-integrity control relying solely on bypassable client-side heuristics. Consistent with the pattern observed in the Critical findings, these High-severity issues concentrate around local data protection and IPC/component trust boundaries — the areas OWASP MASVS-STORAGE and MASVS-PLATFORM are specifically designed to address.

Medium-severity findings reflect defense-in-depth and hardening gaps — a pre-production debug menu shipped inside the same APK, an incomplete backup-exclusion configuration, sensitive balances visible in the Recent Apps thumbnail and via clipboard, weak cache handling of statement data, a deprecated-TLS acceptance path, broader-than-necessary permission requests, verbose API error responses, a client-side-only session timeout, an analytics SDK that does not honor Advertising ID resets, insecure client-side random-number generation, missing tapjacking protection on the transfer-confirmation

screen, an unencrypted local transaction-history database, and an unnecessarily exported help-content provider. Low and Informational findings are hardening and governance opportunities — unused permissions, deprecated API usage, logging hygiene, dependency governance, and positive-control observations such as consistent MFA enforcement and strong TLS/pinning on the primary payments channel — that support a stronger long-term security posture without representing an immediate exploitation path in this fictional sample.

TMG Security's simulated recommendation is that NovaSphere prioritize closure of the four Critical and nine High findings within the first 60 days of the illustrative remediation roadmap (see Section 102), with particular attention to Keystore-backed local storage, exported-component authorization, and confirmation/step-up controls on deep-link-triggered account actions. Medium findings should be addressed within 90 days, and Low/Informational items folded into the ongoing defense-in-depth and mobile-governance backlog. A full retest is recommended following remediation of all Critical and High findings, per the retest methodology in Section 104.

04 ENGAGEMENT OVERVIEW

This section describes the fictional objective, type, and approach of the illustrative NovaSphere Pay Android application penetration test, prepared to demonstrate TMG Security's engagement-scoping and reporting standards for mobile application security assessments.

Assessment Objective

The fictional objective of this engagement was to identify and validate exploitable security weaknesses across NovaSphere Pay's Android client — spanning local data storage, IPC/component exposure, cryptography, network communication, authentication, resilience against reverse engineering and tampering, privacy controls, and the application's integration with its backend mobile API — and to provide NovaSphere's engineering leadership with prioritized, actionable remediation guidance aligned to current OWASP Mobile Application Security material.

Assessment Type & Testing Window

This is a fictional Android Application Security Assessment / Penetration Test conducted against a controlled, fictional staging environment and a fictional release-candidate build (NovaSpherePay-release-6.4.2.apk) over the illustrative assessment period of 03 August 2026 – 28 August 2026, with this report issued 11 September 2026.

Testing Approach

- Static analysis of the fictional APK: manifest and component review, decompiled code review, hardcoded-secret and sensitive-string discovery, and third-party SDK/dependency inventory.
- Dynamic analysis on fictional emulator and physical reference devices (rooted and non-rooted): runtime storage inspection, network-traffic interception, authentication and biometric flow testing, deep-link and Intent-based testing.
- Mobile-API integration testing of the backend endpoints NovaSphere Pay calls, assessed from the perspective of the Android client's authorization and data-handling behavior.
- Structured mapping of every test performed to the current OWASP MASVS control set, using OWASP MASTG v2.0.0 test concepts and OWASP MASWE v1.0.0 weakness concepts wherever a verifiable identifier applies.
- Resilience testing of anti-tampering, anti-reverse-engineering, root/emulator/debugger-detection, and obfuscation controls using standard, non-destructive mobile security testing techniques.
- Privacy-focused review of permission usage, PII handling, and third-party analytics/tracking SDK behavior.

This engagement overview describes a fictional, illustrative assessment. No real client engagement occurred.

05 APPLICATION PROFILE

NovaSphere Pay is a fictional consumer FinTech / digital payments Android application created to demonstrate a realistic mobile application assessment scope. All organization, technology, and infrastructure detail below is illustrative.

Fictional Client Profile

Attribute	Detail
Organization	NovaSphere Digital Technologies, Inc. (Fictional)
Industry	FinTech / Digital Payments / Consumer Technology
Headquarters	Austin, Texas, USA (Fictional)
Employees	Approximately 850 (Fictional)
Application	NovaSphere Pay (Fictional)
Package Name	com.novasphere.pay
Distribution	Google Play (fictional listing; illustrative build only)
Backend API	https://api.novasphere-example.com (Fictional, non-resolving placeholder)
Auth Service	https://auth.novasphere-example.com (Fictional, non-resolving placeholder)
Environment	Fictional staging / controlled assessment environment

Application Build Under Test

Attribute	Detail
APK File Name	NovaSpherePay-release-6.4.2.apk
Version	6.4.2
Target SDK	35 (Android 15)
Minimum SDK	26 (Android 8.0)
Supported Architectures	arm64-v8a, armeabi-v7a
Signing	APK Signature Scheme v2 + v3, fictional production certificate

Core Application Functionality (Fictional, Illustrative)

- Account login, MFA, and biometric app-unlock.
- Account balance, statement, and transaction-history viewing.
- Peer-to-peer and bill-pay funds transfer, including payee management via in-app flows and deep links.
- Card linking and card-detail viewing (masked in the UI).
- Branch-locator (map-based) and in-app help/support content, including a support WebView.
- Push-notification-based transaction and security alerts.

All infrastructure and technology detail above is fictional and provided only to give this sample report realistic structure and depth.

06 BUSINESS CONTEXT

Understanding the fictional business context of NovaSphere Pay helps calibrate the business-impact narrative attached to each finding in this report.

Why Mobile Security Matters Here (Fictional Narrative)

NovaSphere Pay is positioned, in this fictional scenario, as NovaSphere Digital Technologies' primary consumer-facing product and its main channel for funds movement, account servicing, and customer engagement. A security weakness in the Android client carries fictional business risk across several dimensions: direct financial exposure (unauthorized transfers or account takeover), regulatory exposure (mishandling of financial PII under fictional applicable data-protection expectations), reputational exposure (loss of customer trust in a FinTech brand), and operational exposure (fraud-support costs, incident-response overhead, and app-store trust-and-safety scrutiny).

Fictional Stakeholders

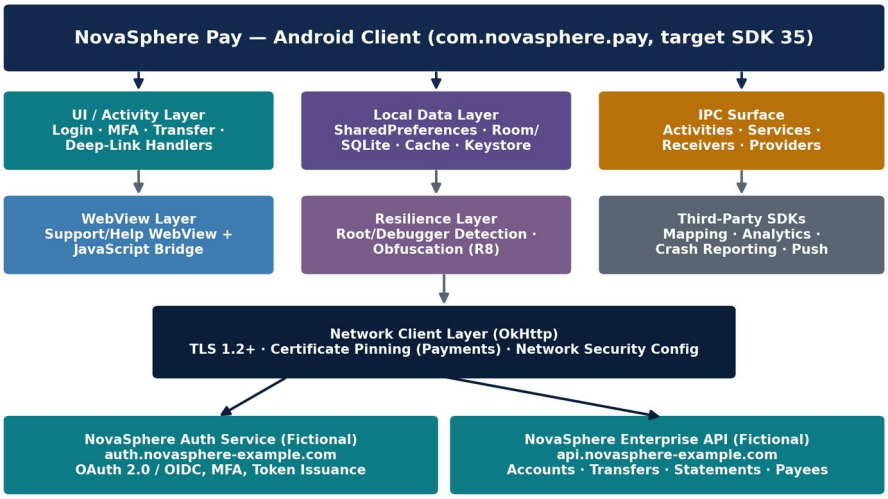
Stakeholder	Illustrative Interest
Engineering Leadership	Prioritizing and resourcing remediation of Critical/High findings.

Stakeholder	Illustrative Interest
Product Management	Balancing frictionless UX (e.g., deep-link onboarding) against security controls.
Risk & Compliance (Fictional)	Understanding regulatory and fraud-exposure implications of findings.
Customer Support (Fictional)	Anticipating fraud-related support volume tied to Critical/High findings.
Executive Leadership	Understanding overall illustrative risk posture ahead of continued platform growth.

All business-context detail in this section is fictional and constructed to give this sample report realistic narrative depth.

07 ANDROID APPLICATION ARCHITECTURE

The following fictional architecture diagram illustrates how NovaSphere Pay's Android client layers — UI/Activities, local data storage, IPC surface, WebView, resilience controls, and third-party SDKs — sit atop a shared network client layer and communicate with the fictional NovaSphere Auth Service and Enterprise API. It is provided to give this sample report realistic technical grounding for the findings in later sections.



FICTIONAL SAMPLE ARCHITECTURE — NovaSphere Digital Technologies, Inc. is a fictional organization.

Illustrative fictional Android application architecture — all service names and infrastructure detail are fictional.

Architecture Notes

- The UI/Activity Layer hosts Login, MFA, account, transfer, and deep-link-handling Activities; several of this report's Critical and High findings (AND-002, AND-004, AND-012) trace to gaps at this layer's boundary with the IPC surface.
- The Local Data Layer spans SharedPreferences, the Room/SQLite database, on-disk cache, and the Android Keystore — the layer most directly implicated in AND-001, AND-015, AND-018, and AND-026.
- The IPC Surface (exported Activities, Services, Broadcast Receivers, and Content Providers) is a recurring theme across this report's Critical findings, reflecting the outsized risk exported components carry in a FinTech application.
- The WebView Layer hosts the in-app support/help experience and its native JavaScript bridge, the subject of AND-005.
- The Resilience Layer (root/debugger detection, R8/ProGuard obfuscation) is assessed in Sections 80-87 and is the subject of AND-013 and AND-039.

- All client layers converge on a single Network Client Layer (OkHttp-based) enforcing TLS 1.2+, certificate pinning on the payments channel, and Network Security Configuration policy, before reaching the fictional NovaSphere Auth Service and NovaSphere Enterprise API.
- Third-Party SDKs (mapping, analytics, crash reporting, push) sit alongside the application's own modules and are reviewed in Sections 37-38 and 91.

08 MOBILE THREAT MODEL

This section presents a fictional, illustrative threat model for NovaSphere Pay, framed around the threat actors and attack vectors most relevant to a consumer Android FinTech application. It is intended to give the findings in later sections a coherent risk narrative, not to serve as a complete formal threat-modeling exercise.

Fictional Threat Actors Considered

Actor	Illustrative Motivation & Capability
Malicious Co-Installed App	A separate app on the same device probing exported components and IPC surfaces for privileged functionality — directly relevant to AND-002, AND-009, and AND-027.
Device-Level Attacker (Lost / Stolen / Rooted Device)	Physical or root-level access to app-private storage, backups, and the local database — relevant to AND-001, AND-015, and AND-026.
Network-Position Attacker	Rogue Wi-Fi or on-path position attempting TLS interception — relevant to AND-006 and AND-019.
Social-Engineering / Phishing Actor	Delivers a crafted deep link or spoofed on-device screen to a victim — relevant to AND-004 and AND-012.
Malicious / Compromised Third-Party SDK	A bundled SDK exceeding its intended data-access scope — relevant to AND-023 and Section 91.
Reverse Engineer	Decompiles the APK to recover secrets or understand server-side logic — relevant to AND-008 and AND-013.

Primary Fictional Attack Vectors In Scope

- Local data extraction from app-private storage, backups, and cache on a rooted or otherwise compromised fictional device.
- Exploitation of exported Android components (Activities, Services, Receivers, Content Providers) by a co-installed malicious application.
- Deep-link and custom-URI-scheme abuse to trigger sensitive account actions without user awareness.
- Network-layer interception and certificate-validation bypass against both the primary payments channel and auxiliary channels (telemetry).
- Mobile-API object-level authorization gaps reachable through the legitimate Android client's own traffic patterns.
- Bypass of local/biometric authentication and root/runtime-integrity controls on a compromised device.

This threat model is illustrative and was constructed to frame this fictional sample assessment; it does not represent a formal STRIDE/DREAD or PASTA threat-modeling deliverable.

09 ASSESSMENT SCOPE

The following fictional scope table defines the boundaries of this illustrative assessment.

In Scope

Area	Detail
Android Application	NovaSpherePay-release-6.4.2.apk (com.novasphere.pay), target SDK 35, min SDK 26 (fictional)

Area	Detail
Static Analysis	Manifest, decompiled code, resources, third-party SDK inventory, dependency review
Dynamic Analysis	Runtime storage, network traffic, authentication flows, deep links, IPC behavior
Local Data Storage	SharedPreferences, Room/SQLite, cache, external storage, clipboard, backups
Cryptography	Key management, Android Keystore usage, algorithm/mode selection, randomness
Network Communication	TLS configuration, certificate validation/pinning, Network Security Configuration
Authentication & Authorization	Login, MFA, biometric/local authentication, session/token handling, account recovery
Mobile-API Integration	Authorization and data-handling behavior of backend endpoints called by the Android client
IPC / Component Surface	Exported Activities, Services, Broadcast Receivers, Content Providers, deep links, App Links
WebView	JavaScript interfaces, navigation policy, mixed content, file access
Resilience	Root/emulator/debugger detection, anti-tampering, repackaging resistance, obfuscation
Privacy	Permission usage, PII handling, third-party analytics/tracking SDK behavior

Out of Scope

Area	Detail
iOS Application	Not included; this engagement is Android-only
Denial-of-Service Testing	Not performed under any circumstances
Destructive Testing	Not performed under any circumstances
Social Engineering	Not included in this engagement
Physical Security	Not included in this engagement
Production Infrastructure	Testing restricted to the fictional staging environment and fictional test devices only
Backend Infrastructure Internals	Server-side infrastructure hardening beyond mobile-API authorization behavior not tested
Signing-Key Custody / Rotation Process	Not directly tested in this engagement — see AND-047 and Section 106
Accessibility Service Abuse	Not directly tested in this engagement — see AND-045 and Section 106
Third-Party Backend Systems	Not tested; only NovaSphere-owned fictional application and API surface in scope

10 OUT-OF-SCOPE COMPONENTS

Beyond the area-level scope boundaries in Section 9, the following specific fictional components and considerations were explicitly excluded from direct testing in this engagement, and are flagged here for transparency.

Component / Area	Reason Excluded
NovaSphere web application	Separate assessment surface; not part of this Android-focused engagement
NovaSphere backend administrative console	Internal tooling outside the Android client's attack surface
Push-notification service provider infrastructure	Third-party managed service; only client-side handling reviewed
Mapping/geolocation SDK vendor infrastructure	Third-party managed service; only client-side integration reviewed (see AND-008)
Crash-reporting SDK vendor infrastructure	Third-party managed service; only client-side configuration reviewed (see AND-043)
Physical device supply chain / OEM firmware	Outside the fictional engagement's mobile-application-security scope

Exclusion from direct testing does not imply these components are free of risk; it reflects this fictional engagement's defined scope and duration.

11 RULES OF ENGAGEMENT

The following fictional rules of engagement governed this illustrative assessment and are representative of the controls TMG Security applies to every real mobile application engagement.

- All testing was controlled, authenticated where applicable, and performed only against the designated fictional staging environment and fictional test devices.
- All testing was strictly non-destructive; no data-modifying action was taken outside of dedicated fictional test accounts and test objects.
- Testing used only the dedicated fictional test accounts and devices listed in Appendix I — no real user accounts, real devices, or real customer data were used at any point.
- Rooted/instrumented testing was performed only against dedicated fictional test devices, never against a production or personally owned device.
- All evidence (fictional) was preserved in structured test-case, APK-metadata, manifest, and finding registers (Appendices A, B, C, and G) for traceability.
- No persistence mechanisms, backdoors, or long-lived footholds were established on any fictional test device at any point.
- No destructive actions (data deletion, service disruption, real funds movement) were taken or simulated as executed.
- All automated testing respected reasonable request-rate limits against the fictional mobile-API surface; no denial-of-service condition was induced.
- No real customer data, app-store account, or production signing key was accessed, used, or referenced at any point in this fictional engagement.
- Defined stop conditions applied: testing would have paused immediately upon any unexpected impact indicator, with emergency escalation to the fictional NovaSphere engineering point of contact.

These rules of engagement are illustrative and describe the fictional engagement model used to construct this sample report.

12 TESTING METHODOLOGY

TMG Security's Android application penetration-testing methodology follows a structured, multi-phase process spanning static analysis, dynamic analysis, and mobile-API integration testing, built on current OWASP Mobile Application Security material — the OWASP Mobile Application Security Verification Standard (MASVS), the OWASP Mobile Application Security Testing Guide (MASTG) v2.0.0, and the OWASP Mobile Application Security Weakness Enumeration (MASWE) v1.0.0. No copyrighted material from these references is reproduced in this report; they are cited only as the conceptual and terminological framework for TMG Security's own methodology. TMG Security is not affiliated with, endorsed by, or certified by OWASP.

Methodology Phases

#	Phase	Illustrative Description
1	Reconnaissance & APK Acquisition	Fictional APK acquisition, integrity validation, and metadata extraction (Section 15).
2	Static Application Analysis	Manifest, decompiled code, secrets, WebView, SDK, and dependency review (Sections 26-38).
3	Local Data Storage Testing	SharedPreferences, Room/SQLite, cache, external storage, clipboard testing (Sections 39-45).
4	Cryptography Testing	Key management, Keystore usage, algorithm and randomness review (Sections 46-51).
5	Network Security Testing	TLS, certificate validation/pinning, Network Security Configuration testing (Sections 52-58).

#	Phase	Illustrative Description
6	Authentication & Authorization Testing	Login, MFA, biometric/local auth, session, and authorization testing (Sections 59-65).
7	Mobile-API Integration Testing	Object-level authorization, business logic, and data-exposure testing of backend endpoints called by the client (Sections 66-71).
8	IPC / Attack Surface Testing	Exported-component, deep-link, Intent, and IPC testing (Sections 72-79).
9	Resilience Testing	Reverse-engineering resistance, anti-tampering, root/emulator/debugger detection (Sections 80-87).
10	Privacy Testing	Permission, PII, and third-party SDK behavior review (Sections 88-92).
11	Vulnerability Validation	Manual confirmation of every candidate finding before inclusion in this report.
12	Reporting & Retest Planning	Structured finding documentation and defined retest scope (Section 104).

Static and Dynamic Analysis Toolchain (Illustrative)

Testing was performed using a fictional representative Android security-testing toolchain, referenced throughout this report's synthetic evidence panels: APK decompilation and static-analysis tooling (JADX-style / MobSF-style), an interception proxy (Burp-style) for dynamic traffic analysis, adb/logcat for on-device inspection, and rooted/non-rooted fictional emulator and physical reference devices (Appendix I). All tool output shown in this report is synthetic and illustrative, not a real tool capture.

13 RISK RATING METHODOLOGY

TMG Security assigns each finding a severity rating using language inspired by the CVSS v3.1 scoring framework, considering exploitability, privileges required, user interaction, scope, and technical/business impact. All scores and vectors in this report are illustrative sample estimations and were assigned for this fictional exercise only — they are not derived from any real vulnerability and are not submitted to any CVSS authority.

Severity Definitions

Severity	Illustrative Score Range	Definition
CRITICAL	9.0 – 10.0	Directly exploitable with severe confidentiality, integrity, or financial-transaction impact; typically requires immediate remediation.
HIGH	7.0 – 8.9	Significant impact with limited prerequisites; requires prompt remediation.
MEDIUM	4.0 – 6.9	Moderate impact, often requiring specific conditions (e.g., a rooted device) or providing defense-in-depth value when fixed.
LOW	0.1 – 3.9	Minor impact; hardening opportunity rather than a directly exploitable weakness.
INFORMATIONAL	N/A	Observation, positive-control note, or recommendation with no direct security-control failure.

Rating Factors Considered

- Exploitability — how readily the fictional weakness could be exercised on-device, over the network, or via the mobile-API surface.
- Privileges Required — the level of fictional device access needed before the weakness is reachable (e.g., standard use vs. root access).
- User Interaction — whether a fictional victim action (e.g., tapping a deep link) is required.
- Attack Complexity — conditions beyond the attacker's control that must align (e.g., network position, device state).
- Scope — whether the weakness affects only the vulnerable component or crosses a trust boundary (e.g., a local-storage gap that also enables mobile-API token replay).
- Technical Impact — effect on confidentiality, integrity, and availability of application and account data.

- Business Impact — fictional effect on fraud exposure, customer trust, and regulatory posture for a FinTech application.

IMPORTANT: All CVSS-style scores, vectors, severities, and risk ratings in this report are fictional and illustrative.

14 MASVS / MASTG / MASWE COVERAGE MODEL

This assessment is structured around the current OWASP Mobile Application Security (MAS) project material: the OWASP Mobile Application Security Verification Standard (MASVS), the OWASP Mobile Application Security Testing Guide (MASTG) v2.0.0, and the OWASP Mobile Application Security Weakness Enumeration (MASWE) v1.0.0. TMG Security uses these three references together as a single traceability chain, and applies a strict rule throughout this report: every MASVS, MASWE, or MASTG identifier cited is a real, verifiable identifier from current official OWASP material — no identifier in this report was invented. Where a finding's concept did not map to a precisely verifiable MASWE or MASTG identifier, this report says so explicitly rather than fabricating one.

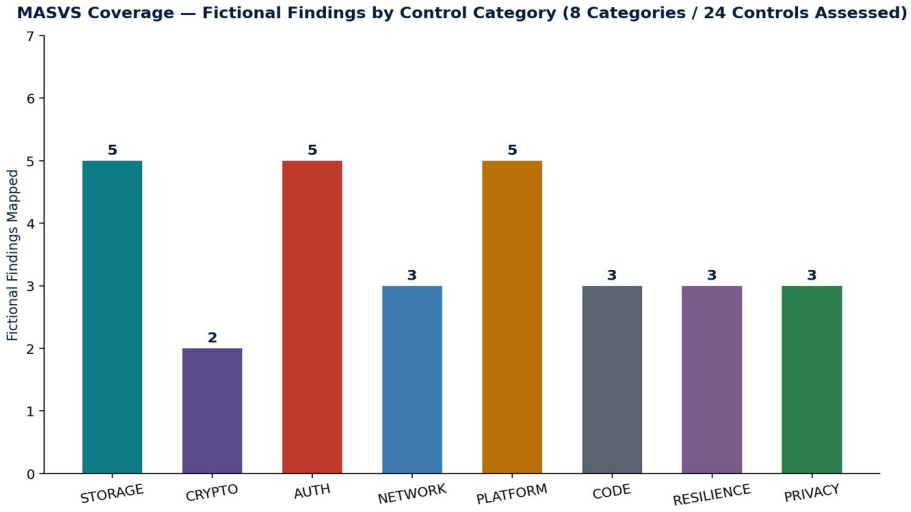
The Traceability Chain

Each testing area in this report is traceable through four linked layers:

- MASVS Control — the security requirement being verified (e.g., MASVS-STORAGE-1: "The app securely stores sensitive data.").
- MASWE Weakness — the specific weakness pattern that would cause the control to fail (e.g., MASWE-0001: "Sensitive Data Stored Unencrypted in Private Storage.").
- MASTG Test — the current MASTG v2.0.0 test technique used to verify the control (e.g., MASTG-TEST-0001: "Testing Local Storage for Sensitive Data.").
- Finding / Evidence — the fictional finding (if any) raised as a result, with its evidence reference in Appendix H.

MASVS Control Groups Assessed

This assessment evaluated NovaSphere Pay against all eight current MASVS control groups, spanning 24 individual controls. Full traceability appears in Appendix E (MASVS Coverage Matrix) and Appendix F (MASWE/MASTG Traceability); the illustrative distribution of fictional findings across these eight groups is shown below.



Illustrative distribution of fictional findings across the 8 MASVS control groups — ILLUSTRATIVE SAMPLE DATA.

MASVS Group	Focus Area
MASVS-STORAGE	Secure local data storage and prevention of sensitive-data leakage.
MASVS-CRYPTO	Strong cryptography and sound key management.
MASVS-AUTH	Secure authentication/authorization protocols, local authentication, and step-up authentication for sensitive operations.
MASVS-NETWORK	Secure network traffic and identity pinning for developer-controlled endpoints.
MASVS-PLATFORM	Secure IPC, WebView usage, and user-interface handling.
MASVS-CODE	Up-to-date platform targeting, update enforcement, vulnerability-free components, input validation.

MASVS Group	Focus Area
MASVS-RESILIENCE	Platform-integrity validation and resistance to tampering, static analysis, and dynamic analysis.
MASVS-PRIVACY	Minimized data/resource access, user non-identifiability, transparency, and user data control.

TMG Security is not affiliated with, endorsed by, or certified by OWASP. This assessment uses OWASP MASVS/MASTG/MASWE as a testing framework only; it does not constitute or imply official OWASP certification of NovaSphere Pay.

15 APK ACQUISITION & VALIDATION

Before static and dynamic testing began, TMG Security acquired and validated the fictional release build under test, confirming its integrity and provenance consistent with the methodology a real engagement would follow.

Illustrative Tests Performed

- Verify the APK signature validates against the fictional production signing certificate.
- Verify the APK hash matches the fictional build provided for testing (no unauthorized modification).
- Verify the APK was installable and launched correctly on all fictional test devices (Appendix I).
- Verify version/build metadata matches the fictional release intended for testing (6.4.2).

Illustrative Observations

The fictional APK (NovaSpherePay-release-6.4.2.apk) was acquired directly from NovaSphere's fictional release pipeline for this engagement, installed on all fictional test devices listed in Appendix I, and confirmed to launch and authenticate correctly before testing began. No integrity or provenance issues were identified during acquisition.

No findings raised against this fictional testing category beyond items cross-referenced elsewhere in this report.

16 APK METADATA ANALYSIS

The table below summarizes the fictional APK's core metadata, extracted during static analysis. Full metadata detail appears in Appendix B.

Attribute	Value
APK File Name	NovaSpherePay-release-6.4.2.apk
Package Name	com.novasphere.pay
Version Name / Version Code	6.4.2 (versionCode 6402)
Target SDK Version	35 (Android 15)
Minimum SDK Version	26 (Android 8.0)
Compile SDK Version	35
Supported ABIs	arm64-v8a, armeabi-v7a
APK Size (Fictional Build)	38.6 MB (compressed)
Signing Scheme	APK Signature Scheme v2 + v3 (fictional production certificate; no real fingerprint disclosed)
android:debuggable (Release Flavor)	false (confirmed absent from production manifest merge)
android:allowBackup	true — see AND-015
Multidex Enabled	true
Obfuscation (R8/ProGuard)	Enabled — coverage gaps noted in AND-039
Build Tool	Android Gradle Plugin 8.x (fictional build configuration)

17 PACKAGE & MANIFEST REVIEW

TMG Security reviewed the fictional AndroidManifest.xml in full, covering declared permissions, application-level flags, and every declared component, as the foundation for the component-level analysis in Sections 18-25.

Illustrative Tests Performed

- Verify android:debuggable is false in the release manifest merge.
- Verify android:allowBackup configuration and any backup-rule exclusions.
- Verify targetSdkVersion and minSdkVersion reflect current supported platform versions.
- Verify declared permissions map to an active, current code path.
- Verify signing configuration uses APK Signature Scheme v2/v3.

Illustrative Observations

The fictional manifest correctly sets android:debuggable="false" in the release build and uses current target/min SDK versions. However, android:allowBackup="true" is set with incomplete backup exclusions (AND-015), and the full dangerous-permission set requested at first run is broader than current feature usage requires (AND-020), with two permissions carrying no referencing code path at all (AND-028). The declared permission set is itemized in Appendix B.

Finding ID	Severity	Title
AND-015	MEDIUM	Insecure Backup Configuration (allowBackup Enabled)
AND-020	MEDIUM	Excessive Android Permissions Requested
AND-028	LOW	Unused / Unnecessary Permissions Declared

18 ANDROID COMPONENT ANALYSIS

Sections 19-25 assess each class of Android component declared in NovaSphere Pay's fictional manifest — Activities, Services, Broadcast Receivers, Content Providers, and the app's Intent/deep-link handling — with particular attention to export flags, permission gates, and caller-identity validation. The full component register appears in Appendix C. This component-level analysis is the single largest contributor to this report's Critical findings, reflecting how much fictional risk concentrates at Android's IPC trust boundary for a FinTech application.

19 ACTIVITIES

TMG Security reviewed all 9 fictional Activities declared in NovaSphere Pay's manifest for export status, task affinity/launch-mode configuration, and sensitive-data exposure via UI state (screenshots, overlays).

Illustrative Tests Performed

- Verify all exported Activities require a documented cross-app use case.
- Verify Activities handling sensitive input use default task affinity.
- Verify sensitive-data-displaying Activities set FLAG_SECURE.
- Verify sensitive confirmation Activities reject touches while obscured by an overlay.

Illustrative Observations

Of the 9 fictional Activities reviewed, PayeeLinkActivity is exported to support deep-link handling but commits sensitive account state with no confirmation step (AND-004); MfaVerifyActivity carries a non-default task-affinity/launch-mode configuration associated with task-hijacking risk (AND-012); AccountOverviewActivity and CardDetailActivity do not set FLAG_SECURE (AND-016); and TransferConfirmActivity does not reject touches while obscured (AND-025). The full Activity-level register appears in Appendix C.

Finding ID	Severity	Title
AND-004	CRITICAL	Insecure Deep-Link Flow Enables Sensitive Account Action
AND-012	HIGH	Improper Intent Handling Enables Intent Spoofing / Task Hijacking Risk
AND-016	MEDIUM	Sensitive Data Exposed via Screenshot / Recent-Apps Thumbnail
AND-025	MEDIUM	Tapjacking / Overlay Protection Not Implemented on Sensitive Confirmation Screens

20 SERVICES

TMG Security reviewed all 3 fictional Services declared in NovaSphere Pay's manifest for export status, permission gating, and caller-identity validation on any privileged action they perform.

Illustrative Tests Performed

- Verify all exported Services require a documented cross-app use case and permission gate.
- Verify Services performing privileged actions independently validate caller identity.
- Verify background Services (JobScheduler-bound) are not independently exported.

Illustrative Observations

TransferService is exported with no permission gate and no caller-identity validation, allowing any co-installed fictional application to trigger a funds-transfer-initiating Intent directly (AND-002) — this is the most severe component-level finding in this fictional assessment. NotificationSyncService and TokenRefreshService are not exported and were not found to carry an equivalent weakness.

Finding ID	Severity	Title
AND-002	CRITICAL	Exported Android Component Enables Unauthorized Privileged Action

21 BROADCAST RECEIVERS

TMG Security reviewed all 3 fictional Broadcast Receivers declared in NovaSphere Pay's manifest for export status and the sensitivity of the action each receiver performs on receipt of a broadcast.

Illustrative Tests Performed

- Verify exported Broadcast Receivers perform no sensitive action on receipt of an untrusted broadcast.
- Verify sensitive Broadcast Receivers validate the sender where the platform allows.
- Verify BootCompletedReceiver and PackageReplacedReceiver perform read-only or low-risk actions only.

Illustrative Observations

BootCompletedReceiver and PackageReplacedReceiver are both exported (required for their respective system broadcasts) but were confirmed to perform only low-risk, read-only-equivalent actions (reminder re-arming and cache re-initialization) with no sensitive account action reachable through either. NotificationActionReceiver is not exported. No findings were raised against this component class.

No findings raised against this fictional testing category beyond items cross-referenced elsewhere in this report.

22 CONTENT PROVIDERS

TMG Security reviewed all 3 fictional Content Providers declared in NovaSphere Pay's manifest, including the FileProvider used for statement export, for export status, URI-permission scoping, and the sensitivity of the data each provider can return.

Illustrative Tests Performed

- Verify all exported Content Providers require a documented cross-app use case.
- Verify FileProvider path configuration is scoped to the minimum required directory.
- Verify granted URI permissions are revoked after the sharing operation completes.
- Verify signature-restricted providers correctly enforce that restriction.

Illustrative Observations

NovaSphereFileProvider's file_paths.xml grants access to a broader private-files root than the single statement file it is meant to export, allowing a receiving app to retrieve additional cached files through the same granted authority (AND-009). HelpContentProvider is exported with no identified cross-app use case, unnecessarily widening the IPC attack surface even though only non-sensitive FAQ content is currently served (AND-027). TransactionSearchProvider is signature-restricted and was confirmed to correctly reject non-signature callers.

Finding ID	Severity	Title
AND-009	HIGH	Insecure FileProvider Configuration
AND-027	MEDIUM	Exported Content Provider Permits Non-Critical Data Read

23 INTENT & DEEP LINK SECURITY

TMG Security tested NovaSphere Pay's handling of both explicit and implicit Intents, and its custom-scheme deep-link handler (novasphere://), for input validation, confirmation gating, and resistance to spoofing/redirection.

Illustrative Tests Performed

- Verify all custom-scheme deep links route through a confirmation step for sensitive actions.
- Verify Intent extras are not trusted for authorization decisions without independent verification.
- Verify deep-link parameters are treated as untrusted input and validated server-side.
- Verify Intents received by exported components validate the calling package where feasible.

Illustrative Observations

The novasphere://add-payee custom-scheme deep link commits a new payee association immediately on link open, with no in-app confirmation or re-authentication step, allowing a fictional attacker-delivered link to silently modify sensitive account state (AND-004). No other Intent-handling code path tested was found to trust Intent extras for an authorization decision without independent server-side verification.

Finding ID	Severity	Title
AND-004	CRITICAL	Insecure Deep-Link Flow Enables Sensitive Account Action

24 APP LINKS

TMG Security reviewed NovaSphere Pay's HTTPS App Link configuration (https://pay.novasphere-example.com/link) for Digital Asset Links verification and consistency with the custom-scheme deep-link handler assessed in Section 23.

Illustrative Tests Performed

- Verify App Links are verified (Digital Asset Links) rather than accepting unverified sources.
- Verify the App Link intent-filter does not omit android:autoVerify.
- Verify App-Link-triggered actions receive the same confirmation gating as their custom-scheme equivalents.

Illustrative Observations

The fictional App Link configuration correctly declares android:autoVerify and Digital Asset Links verification was confirmed functional in dynamic testing. However, because PayeeLinkActivity's intent-filter accepts both the App Link and the custom

novasphere:// scheme through the same handler, the confirmation-gating gap identified in AND-004 applies equally to App-Link-triggered invocations and is tracked as a single finding rather than a duplicate.

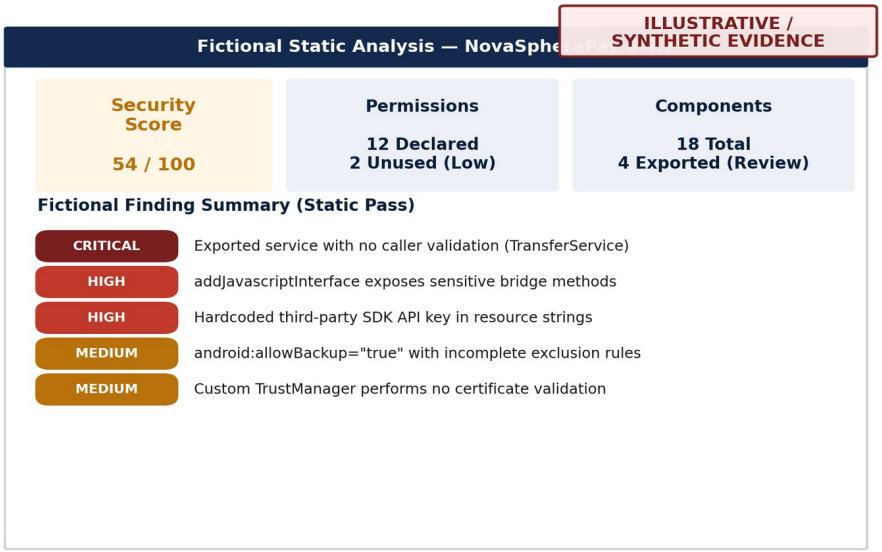
No findings raised against this fictional testing category beyond items cross-referenced elsewhere in this report.

25 EXPORTED COMPONENTS

Sections 19-22 assessed each component class individually. This section consolidates the full exported-component picture: NovaSphere Pay declares 18 total Android components across Activities, Services, Broadcast Receivers, and Content Providers, of which 11 are exported. Of those 11, TransferService (AND-002), NovaSphereFileProvider (AND-009), and HelpContentProvider (AND-027) were confirmed to carry a specific, exploitable weakness in this fictional assessment. The remaining exported components (PayeeLinkActivity, tracked separately under the deep-link findings AND-004; BootCompletedReceiver and PackageReplacedReceiver, confirmed low-risk in Section 21; and the balance of exported components with no identified use case) are itemized in the broader attack-surface hardening observation AND-032. The full exported-component register, with export status and finding cross-reference for every component, appears in Appendix C.

26 STATIC APPLICATION ANALYSIS

Sections 27-38 present the fictional static-analysis pass against the decompiled NovaSpherePay-release-6.4.2.apk, covering decompiled code review, hardcoded secrets, sensitive strings, debug artifacts, logging behavior, WebView configuration, file handling, serialization, and third-party SDK/dependency inventory. Static analysis was performed using representative JADX-style decompilation and MobSF-style automated static scanning; a synthetic summary consistent with this pass is shown below.



Fictional static analysis dashboard summary — ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE. No real tool output is depicted.

27 DECOMPILED CODE REVIEW

TMG Security decompiled the fictional release APK and manually reviewed security-relevant classes — session management, token handling, cryptography, deep-link handlers, and IPC-facing components — cross-referencing decompiled logic against the manifest and dynamic-testing observations.

Illustrative Tests Performed

- Verify security-relevant classes are not trivially readable due to insufficient obfuscation.
- Verify decompiled logic matches the behavior observed in dynamic testing (no hidden/conditional logic divergence).
- Verify sensitive business logic (e.g., transfer authorization) is not solely client-side enforced.

Illustrative Observations

Decompiled review confirmed the SessionManager, PayeeLinkActivity, and TransferService logic referenced throughout this report's findings, and found several security-relevant classes more readable than necessary due to broader-than-required ProGuard/R8 keep rules (AND-039). No evidence of intentionally obfuscated malicious logic or dynamic-static behavioral divergence was identified.

Finding ID	Severity	Title
AND-039	INFORMATIONAL	Recommendation — Expand Obfuscation Coverage (R8/ProGuard)

28 HARDCODED SECRETS

TMG Security searched decompiled resources, string tables, and code for hardcoded credentials, API keys, and other secret material.

Illustrative Tests Performed

- Verify no hardcoded credentials or API keys are present in string resources.
- Verify no hardcoded credentials or API keys are present in decompiled code (including native libraries).
- Verify third-party SDK keys are retrieved at runtime where the SDK supports it.

Illustrative Observations

A fictional third-party mapping/geolocation SDK API key was found hardcoded in plaintext in res/values/strings.xml, recoverable through straightforward APK decompilation (AND-008). No credentials for the primary payments API or auth service were found hardcoded; those flows use the OAuth 2.0/OIDC token exchange assessed in Sections 59-61.

Finding ID	Severity	Title
AND-008	HIGH	Hardcoded API Secret in Application Package

29 API KEYS & TOKENS

TMG Security reviewed how NovaSphere Pay obtains, stores, and transmits API keys and authentication tokens, distinct from the third-party SDK key weakness in Section 28.

Illustrative Tests Performed

- Verify the payments-API access/refresh tokens are obtained only via the standard OAuth 2.0/OIDC flow, never hardcoded.
- Verify tokens are not embedded in APK resources or native libraries.
- Verify token values are not passed as URL query parameters (which risk exposure via logs/history).

Illustrative Observations

No payments-API token material was found hardcoded or embedded in the APK; tokens are correctly obtained only via the runtime OAuth 2.0/OIDC exchange. However, once issued, token storage and lifecycle carry the weaknesses tracked separately as AND-001 (insecure local storage) and AND-011 (missing revocation on logout) — see Sections 40 and 61.

No findings raised against this fictional testing category beyond items cross-referenced elsewhere in this report.

30 SENSITIVE STRINGS

TMG Security extracted and reviewed the fictional APK's compiled string table and embedded URLs for sensitive or unintended disclosures.

Illustrative Tests Performed

- Verify no internal-only or staging endpoint URLs are present in the production string table.
- Verify no developer comments or TODO markers reference sensitive future work.
- Verify no PII-shaped or credential-shaped literal strings are present.

Illustrative Observations

Static string-table review recovered production API/auth base URLs (expected), a residual fictional staging-endpoint reference not intended for the production build, and a residual developer comment referencing the debug-menu trigger addressed in Section 31 / AND-014. No PII-shaped or credential-shaped literal strings beyond the AND-008 mapping key were identified. The full extracted string sample is itemized in Appendix B.

Finding ID	Severity	Title
AND-014	MEDIUM	Debug Information Exposure in Pre-Production Build Configuration

31 DEBUG CODE

TMG Security reviewed the fictional APK for debug-only code paths that could reach a distributable build, including the debug menu identified during string-table review in Section 30.

Illustrative Tests Performed

- Verify debug-only tooling is compiled out of any build variant that could plausibly reach distribution.
- Verify android:debuggable is false in the release manifest merge (cross-referenced from Section 17).
- Verify test/mock network endpoints are not reachable from the production build flavor.

Illustrative Observations

A hidden debug menu reachable via a tap sequence on the version-number label is present in the fictional QA-flavored build and ships as compiled code inside the same APK, gated only by a runtime flag rather than compile-time flavor exclusion (AND-014). android:debuggable was confirmed false in the release manifest merge.

Finding ID	Severity	Title
AND-014	MEDIUM	Debug Information Exposure in Pre-Production Build Configuration

32 LOGGING & DEBUGGING

TMG Security reviewed logging calls throughout the fictional codebase and captured logcat output during dynamic testing to assess whether sensitive data reaches the system log.

Illustrative Tests Performed

- Verify no sensitive values (tokens, OTPs, card data, PII) are ever passed to a logging call.
- Verify logging gates rely on redaction at the call site, not solely a build-flavor check.

- Verify general UI-layer logging volume is minimized in release builds.

Illustrative Observations

Several network-layer interceptors log full request/response bodies — including bearer tokens and MFA codes — at Log.d() level; while gated by BuildConfig.DEBUG in the intended production flavor, this was still observable via logcat on the fictional QA-flavored build tested (AND-007). Beyond sensitive-value logging, general UI-layer navigation logging volume is higher than necessary and contributes minor behavioral-fingerprinting risk (AND-030).

Finding ID	Severity	Title
AND-007	HIGH	Sensitive Data Exposure Through Logs
AND-030	LOW	General Logging Hygiene Weaknesses in Non-Sensitive Modules

33 WEBVIEW SECURITY

TMG Security reviewed NovaSphere Pay's single WebView instance (the in-app support/help screen) for navigation policy, mixed-content handling, and file-access configuration, consistent with OWASP MASVS-PLATFORM-2.

Illustrative Tests Performed

- Verify WebView navigation is restricted to an allow-listed first-party domain set.
- Verify mixed content loading is disabled in all WebView instances.
- Verify file:// URL loading is disabled in all WebView instances.
- Verify WebView JavaScript execution is disabled where not strictly required.

Illustrative Observations

The support WebView does not restrict navigation to a strict first-party allow-list and permits mixed-content loading, widening the population of content that could reach the JavaScript interface assessed in Section 34 (AND-005, contributing factor). file:// URL loading was confirmed disabled. JavaScript execution is required for the support experience and remains enabled, making the navigation-policy gap the primary compensating control missing.

Finding ID	Severity	Title
AND-005	HIGH	WebView JavaScript Interface Exposure

34 JAVASCRIPT INTERFACES

TMG Security enumerated and tested every method exposed through NovaSphere Pay's addJavascriptInterface bridge (NovaBridge), the primary subject of this section.

Illustrative Tests Performed

- Verify addJavascriptInterface methods do not expose sensitive session data.
- Verify bridge methods independently verify the calling page's origin.
- Verify only the minimum necessary methods are exposed through the bridge.

Illustrative Observations

The NovaBridge JavaScript interface exposes getSessionToken() and getDeviceId() with no origin verification, allowing any JavaScript executing in the WebView context — not only the intended first-party support content — to retrieve the live session token (AND-005). This is this report's sole WebView/JavaScript-interface finding and is rated High given the direct session-token exposure path.

Finding ID	Severity	Title
AND-005	HIGH	WebView JavaScript Interface Exposure

35 FILE HANDLING

TMG Security reviewed NovaSphere Pay's file-handling paths — statement PDF export, FileProvider-based sharing, and temporary-file lifecycle — for scope and cleanup weaknesses, extending the FileProvider-specific findings from Section 22.

Illustrative Tests Performed

- Verify temporary files created during statement export are removed after use.
- Verify FileProvider grants are scoped to the intended single file (cross-referenced from Section 22).
- Verify downloaded/exported files are not written to unprotected external storage locations.

Illustrative Observations

Beyond the FileProvider path-scoping gap already tracked as AND-009, temporary export files were confirmed to be cleaned up appropriately after the share flow completes, and no exported or cached file was found written to unprotected external storage outside the app's scoped directories.

Finding ID	Severity	Title
AND-009	HIGH	Insecure FileProvider Configuration

36 SERIALIZATION / DESERIALIZATION

TMG Security reviewed how NovaSphere Pay serializes and deserializes data for local persistence (Room entity mapping) and API payloads (JSON), looking for insecure deserialization or object-injection-class weaknesses.

Illustrative Tests Performed

- Verify local persistence uses safe, schema-bound serialization (Room entities) rather than generic object serialization.
- Verify API response deserialization uses a strict, type-bound JSON parser rather than dynamic/reflective deserialization.
- Verify deserialized API responses are validated before being used in security-relevant decisions.

Illustrative Observations

NovaSphere Pay uses Room's schema-bound entity mapping for local persistence and a strict, type-bound JSON parser for API responses; no use of generic Java/Kotlin object serialization or reflective deserialization of untrusted data was identified. No findings were raised against this testing area.

No findings raised against this fictional testing category beyond items cross-referenced elsewhere in this report.

37 THIRD-PARTY SDK REVIEW

TMG Security inventoried and reviewed every third-party SDK bundled with NovaSphere Pay for its data-access scope and the security posture of its integration. The full inventory appears below and in Appendix B.

SDK	Vendor (Fictional)	Purpose	Data Access
Core Payments Networking	Internal fictional module	Payments API client (OkHttp-based)	Payment/account/transfer data
Mapping / Geolocation SDK	Fictional Third-Party Vendor A	Branch-locator map rendering	Coarse device location (feature-scoped)

SDK	Vendor (Fictional)	Purpose	Data Access
Analytics / Product Telemetry SDK	Fictional Third-Party Vendor B	Usage analytics, funnel tracking	Device identifier, usage events — see AND-023
Crash Reporting SDK	Fictional Third-Party Vendor C	Crash and ANR reporting	Stack traces, breadcrumb events — see AND-043
Push Notification SDK	Platform Push Service (Fictional)	Transaction/security alert delivery	Push token, notification payload
Biometric Authentication Library	AndroidX Biometric (fictional pinned version)	BiometricPrompt wrapper — see AND-010	None (local authentication only)
Local Database ORM	Room (fictional pinned version) — see AND-026	Local transaction-history cache	Cached transaction records (unencrypted)
Image Loading Library	Fictional Third-Party Vendor D	Remote image/avatar loading	None (image URLs only)

Of the eight fictional SDKs inventoried, the mapping/geolocation SDK's hardcoded API key (AND-008) and the analytics SDK's persistent-identifier tracking behavior (AND-023) were the specific weaknesses identified; the crash-reporting SDK's default breadcrumb configuration is flagged as a defensive-baseline observation (AND-043) rather than a confirmed exposure. No third-party SDK was found to independently exfiltrate data beyond its documented purpose.

38 DEPENDENCY & LIBRARY REVIEW

TMG Security reviewed NovaSphere Pay's Gradle dependency manifest for outdated or known-vulnerable third-party libraries, and for the governance process (or absence of one) managing dependency freshness.

Illustrative Tests Performed

- Verify third-party dependencies are scanned for known vulnerabilities.
- Verify no dependency carries a publicly disclosed, unpatched vulnerability applicable to its usage.
- Verify a defined process exists for evaluating and applying dependency updates.

Illustrative Observations

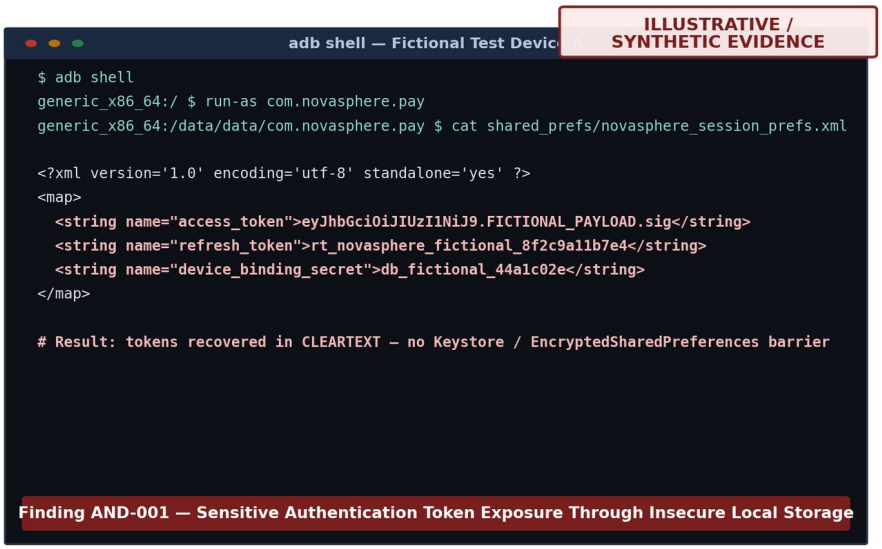
Two fictional third-party libraries were found pinned several minor releases behind current upstream, with no automated dependency-vulnerability scanning integrated into the fictional build pipeline to flag this drift (AND-031). No specific exploitable vulnerability was confirmed in the outdated versions during this assessment's scope; the finding concerns governance process rather than a confirmed exploit path.

Finding ID	Severity	Title
AND-031	LOW	Third-Party Dependency Governance Gaps

39 LOCAL DATA STORAGE

Sections 40-45 assess every fictional local-storage surface NovaSphere Pay writes to — SharedPreferences, the Room/SQLite database, cache and general file storage, external storage, the clipboard, and UI-level exposure via screenshots/Recent Apps — against OWASP MASVS-STORAGE. This is the single largest concentration of findings in this fictional assessment (AND-001, the report's most severe finding, plus AND-015, AND-016, AND-017, AND-018, and AND-026), reflecting how much risk accumulates when local storage is treated as implicitly trusted simply because it sits inside the app sandbox.

40 SHAREDPREFERENCES



Fictional shared_prefs extraction — ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE.

TMG Security extracted and reviewed every fictional SharedPreferences file NovaSphere Pay writes, on a rooted fictional test device, to assess whether sensitive values are protected at rest.

Illustrative Tests Performed

- Verify session/access tokens are not stored in plaintext SharedPreferences.
- Verify refresh tokens are not stored in plaintext SharedPreferences.
- Verify device-binding secrets are stored via Keystore-backed encryption.

Illustrative Observations

The novasphere_session_prefs.xml file stores the access token, refresh token, and device-binding secret in cleartext, recoverable directly from the app's private storage on a rooted fictional test device with no Keystore or EncryptedSharedPreferences barrier (AND-001). This is the most severe finding in this fictional assessment given its direct path to fictional account takeover.

Finding ID	Severity	Title
AND-001	CRITICAL	Sensitive Authentication Token Exposure Through Insecure Local Storage

41 SQLITE / ROOM

TMG Security reviewed the Room-backed local database (novasphere_local.db) used to cache recent transaction history for offline viewing, and tested whether its contents are protected at rest.

Illustrative Tests Performed

- Verify the local Room/SQLite database is encrypted at rest.
- Verify database schema does not store more sensitive detail than the offline-viewing feature requires.

Illustrative Observations

novasphere_local.db is a standard, unencrypted SQLite database; once copied off a rooted fictional test device, it opened directly in a standard SQLite browser and disclosed cleartext transaction rows (AND-026).

Finding ID	Severity	Title
AND-026	MEDIUM	Unencrypted Room/SQLite Database Storage of Transaction History

42 FILES & CACHE

TMG Security reviewed NovaSphere Pay's on-disk HTTP response cache and general private-file usage for retention of sensitive data beyond the active session.

Illustrative Tests Performed

- Verify HTTP response caching excludes sensitive financial data.
- Verify cache directories are purged of sensitive data on logout.
- Verify no sensitive values are hardcoded in application resources (cross-referenced from Section 28).

Illustrative Observations

Statement and transaction-history API responses are cached to disk by the default OkHttp cache policy with no Cache-Control: no-store directive, leaving cleartext transaction data recoverable from the app's private cache directory independent of the active session state (AND-018).

Finding ID	Severity	Title
AND-018	MEDIUM	Weak Cache Handling of Sensitive Response Data

43 EXTERNAL STORAGE

TMG Security reviewed whether NovaSphere Pay writes any application data to external (shared) storage, where it would be readable by any app holding the appropriate storage permission.

Illustrative Tests Performed

- Verify no sensitive application data is written to external/shared storage.
- Verify exported statement files are written only via the scoped FileProvider path (cross-referenced from Sections 22 and 35), not directly to external storage.

Illustrative Observations

No fictional application data was found written directly to external/shared storage outside the scoped FileProvider export path; the export-scoping weakness on that path is tracked separately as AND-009. No standalone finding was raised for this testing area.

No findings raised against this fictional testing category beyond items cross-referenced elsewhere in this report.

44 CLIPBOARD

TMG Security tested NovaSphere Pay's clipboard-writing behavior, focused on the "copy account number" convenience action, for cross-app exposure risk.

Illustrative Tests Performed

- Verify clipboard writes of sensitive values are marked sensitive content where the platform version supports it.
- Verify copied sensitive values are automatically cleared from the clipboard after a short interval.

Illustrative Observations

The "copy account number" action places the full fictional account number on the system clipboard with no sensitive-content marking and no automatic clearing, leaving it readable by any other fictional app polling the clipboard shortly after the copy action (AND-017).

Finding ID	Severity	Title
AND-017	MEDIUM	Sensitive Data Exposure via Clipboard

45 KEYBOARD / SCREENSHOT / RECENT APPS EXPOSURE

TMG Security tested whether sensitive UI content is exposed through custom-keyboard caching, OS-level screenshot capture, or the Recent Apps thumbnail.

Illustrative Tests Performed

- Verify sensitive UI content is blanked in the Recent Apps thumbnail (FLAG_SECURE).
- Verify custom-keyboard input caching / autofill does not capture sensitive fields.
- Verify screen-recording capture is blocked on sensitive screens consistent with the FLAG_SECURE configuration.

Illustrative Observations

AccountOverviewActivity and CardDetailActivity do not set FLAG_SECURE, so the fictional account balance and card-detail screens render in full in the Recent Apps thumbnail and are capturable via OS-level screenshot/screen-recording (AND-016). No custom-keyboard caching or autofill exposure was identified on sensitive input fields (password, MFA code, PIN), which correctly declare non-cacheable input types.

Finding ID	Severity	Title
AND-016	MEDIUM	Sensitive Data Exposed via Screenshot / Recent-Apps Thumbnail

46 CRYPTOGRAPHY

Sections 47-51 assess NovaSphere Pay's use of cryptography against OWASP MASVS-CRYPTO — key management, use of the Android Keystore, algorithm and mode selection, randomness generation for security-relevant values, and the certificate/trust configuration that anchors the app's TLS posture. TMG Security's fictional review found the application's core cryptographic primitives (AES-GCM for local field-level encryption where used, RSA/EC key generation via the Android Keystore provider) to be sound; the confirmed weakness in this category is narrower and concerns a single security-relevant value generated with an unsuitable random source (AND-024) rather than a systemic cryptographic-implementation failure.

47 KEY MANAGEMENT

TMG Security reviewed how NovaSphere Pay generates, stores, and retires cryptographic keys used for local data protection and device-binding, focused on whether raw key material is ever available outside the Android Keystore's hardware- or software-backed boundary.

Illustrative Tests Performed

- Verify cryptographic keys used for sensitive local protection are generated inside the Android Keystore rather than derived and held in application memory.
- Verify no raw symmetric or private key material is written to application storage outside the Keystore.
- Verify key aliases and lifecycle (generation, rotation, deletion) are managed through a documented, centralized routine rather than ad hoc per-feature code.

Illustrative Observations

Where NovaSphere Pay does perform local field-level encryption (outside the SharedPreferences gap tracked as AND-001), keys are correctly generated inside the Android Keystore via KeyGenParameterSpec, and no raw key material was recovered from application storage on the fictional rooted test device. Key alias naming and rotation practice is inconsistent across modules but did not rise to a standalone finding within this assessment's scope.

No findings raised against this fictional testing category beyond items cross-referenced elsewhere in this report.

48 ANDROID KEYSTORE

TMG Security assessed NovaSphere Pay's use of the Android Keystore system itself — key protection parameters, StrongBox/hardware-backing usage where available on the fictional test devices, and whether Keystore-protected operations require user authentication where the sensitivity of the operation warrants it.

Illustrative Tests Performed

- Verify Keystore-backed keys used for sensitive operations are configured with setUserAuthenticationRequired() where appropriate.
- Verify hardware-backed (StrongBox/TEE) key storage is used on devices that support it.
- Verify Keystore key access is not silently downgraded to software-only backing without the application's awareness.

Illustrative Observations

The Keystore-backed keys reviewed use hardware-backed storage (TEE) on both fictional test devices, with software fallback available for older fictional device profiles as documented in the application's compatibility matrix. The device-binding key protecting the AND-001 session material is itself correctly Keystore-resident; the AND-001 finding concerns the token material stored alongside it, not the Keystore configuration. No standalone finding is raised in this section.

No findings raised against this fictional testing category beyond items cross-referenced elsewhere in this report.

49 WEAK CRYPTOGRAPHIC IMPLEMENTATIONS

TMG Security reviewed NovaSphere Pay's decompiled code and Keystore configuration for use of deprecated or inappropriate cryptographic algorithms, insecure modes (e.g., unauthenticated AES/ECB), fixed initialization vectors, or hardcoded cryptographic keys, consistent with OWASP MASVS-CRYPTO-1 and MASVS-CRYPTO-2.

Illustrative Tests Performed

- Verify no use of deprecated algorithms (DES, RC4, MD5/SHA-1 for security-relevant hashing) in security-relevant code paths.
- Verify AES usage employs an authenticated mode (GCM) rather than unauthenticated ECB/CBC without a MAC.
- Verify initialization vectors and salts are generated fresh per operation rather than fixed or reused.
- Verify no cryptographic key is hardcoded in application code or resources.

Illustrative Observations

Local field-level encryption uses AES-256-GCM with per-operation IV generation; no use of ECB mode, deprecated algorithms, or fixed IVs was identified in the decompiled fictional codebase. No hardcoded cryptographic key was found (distinct from the hardcoded API secret tracked as AND-008, which is a service credential rather than a cryptographic key). No findings are raised in this section.

No findings raised against this fictional testing category beyond items cross-referenced elsewhere in this report.

50 RANDOMNESS / NONCES

TMG Security reviewed the random-value generation used across NovaSphere Pay's security-relevant code paths — cryptographic IVs, session-correlated nonces, and the device-fingerprint value used in the fictional device-binding flow — for use of a cryptographically secure random source.

Illustrative Tests Performed

- Verify cryptographic operations (key generation, IVs) use SecureRandom or an equivalent CSPRNG.
- Verify security-relevant non-cryptographic values (anti-replay nonces, device-binding identifiers) also use a CSPRNG rather than a predictable or low-entropy source.
- Verify no security-relevant random value is seeded from a predictable input (timestamp, counter) without adequate additional entropy.

Illustrative Observations

Cryptographic IV and key generation correctly use SecureRandom throughout. The device-binding nonce generated during fictional device registration, however, is derived from java.util.Random seeded with the device boot timestamp rather than SecureRandom, producing a value with materially reduced entropy and a predictable generation window (AND-024). This nonce contributes to the device-binding check but is not, on its own, sufficient to bypass authentication, which keeps this finding at Medium rather than Critical/High severity.

Finding ID	Severity	Title
AND-024	MEDIUM	Insecure Random Number Generation for a Security-Relevant Value

51 CERTIFICATE & TRUST CONFIGURATION

TMG Security reviewed NovaSphere Pay's Network Security Configuration and certificate-trust setup at a high level as the entry point into the fuller TLS, certificate-validation, and certificate-pinning assessment presented in Sections 52-58. This section previews the two certificate-related findings carried forward into that chapter.

Illustrative Tests Performed

- Verify the Network Security Configuration does not extend trust to the user or system-added CA store for production traffic.
- Verify certificate validation logic has not been weakened or bypassed anywhere in the fictional codebase.
- Verify certificate pinning, where implemented, covers all sensitive first-party endpoints.

Illustrative Observations

Two certificate/trust-related weaknesses were identified during this review and are carried forward for full technical treatment in the Network Security chapter: a custom TrustManager implementation that does not correctly validate the certificate chain under a specific fictional proxy-interception condition (AND-006), and a Network Security Configuration that permits a deprecated minimum TLS version on a subset of fictional endpoints (AND-019). Both are cross-referenced rather than re-scored in this section to avoid double-counting.

Finding ID	Severity	Title
AND-006	HIGH	Certificate Validation Weakness
AND-019	MEDIUM	Deprecated TLS Configuration Permitted by Network Security Configuration

52 NETWORK SECURITY

Sections 53-58 assess NovaSphere Pay's transport-layer security against OWASP MASVS-NETWORK — TLS configuration strength, certificate validation logic, Network Security Configuration (NSC) scoping, cleartext-traffic exposure, certificate pinning coverage, and resistance to proxy-based interception. This chapter carries forward the two certificate/trust findings previewed in Section 51 (AND-006, AND-019) and closes with the deprecated-TLS-version finding and the NSC domain-scoping recommendation raised specifically within this chapter (AND-035). NovaSphere Pay's baseline network posture — TLS 1.2+ enforced platform-wide on the primary API client — is recorded as a positive control observation (AND-037) rather than treated as a gap.

53 TLS CONFIGURATION

TMG Security reviewed the TLS versions, cipher suites, and protocol fallback behavior negotiated by NovaSphere Pay's network clients against every fictional first-party endpoint in scope.

Illustrative Tests Performed

- Verify the minimum negotiable TLS version is 1.2 across all first-party endpoints.
- Verify only strong, forward-secret cipher suites are offered by the client.
- Verify no automatic fallback to a legacy TLS/SSL version occurs on handshake failure.

Illustrative Observations

The primary API client (OkHttp, used for authentication, account, and payment traffic) correctly enforces TLS 1.2 as a floor with strong, forward-secret cipher suites and no legacy fallback — recorded as a positive control (AND-037). A secondary, older HTTP client retained for the statement-export subsystem's legacy backend integration permits negotiation down to TLS 1.1 on that subset of fictional endpoints, which is tracked as AND-019.

Finding ID	Severity	Title
AND-019	MEDIUM	Deprecated TLS Configuration Permitted by Network Security Configuration

54 CERTIFICATE VALIDATION

TMG Security tested NovaSphere Pay's certificate-chain validation logic under fictional proxy-interception conditions to confirm the application correctly rejects an untrusted or improperly chained certificate rather than silently accepting it.

Illustrative Tests Performed

- Verify a custom TrustManager (if present) performs full chain and hostname validation equivalent to the platform default.
- Verify the application rejects a self-signed or otherwise untrusted certificate presented by an interception proxy under default device trust settings.
- Verify hostname verification cannot be bypassed via a custom HostnameVerifier that unconditionally returns true.

Illustrative Observations

A custom TrustManager implemented for the statement-export subsystem's legacy client (the same client identified in Section 53) contains a chain-validation defect: under a specific fictional proxy-interception condition involving an intermediate certificate substitution, the custom logic accepts the connection where platform-default validation would reject it (AND-006). The primary API client's certificate validation, which relies on platform-default TrustManager behavior, was not affected.

Finding ID	Severity	Title
AND-006	HIGH	Certificate Validation Weakness

55 NETWORK SECURITY CONFIGURATION (NSC)

TMG Security reviewed NovaSphere Pay's res/xml/network_security_config.xml for domain scoping, cleartext-traffic policy, and pinning declarations.

Illustrative Tests Performed

- Verify the NSC does not apply a single permissive base-config to all domains where narrower per-domain policy is appropriate.
- Verify cleartextTrafficPermitted is explicitly set to false at the base-config level.
- Verify debug-overrides (trusting user-added CAs) are excluded from the release build variant.

Illustrative Observations

The release-variant NSC correctly excludes the debug-overrides block trusting user-added CAs, and cleartext traffic is disabled at the base-config level (see Section 56). However, the NSC applies one shared domain-config to all first-party and third-party SDK endpoints alike rather than scoping pinning and stricter TLS requirements specifically to the payments-critical api.novasphere-example.com and auth.novasphere-example.com hosts, reducing the precision of the app's trust boundaries (AND-035).

Finding ID	Severity	Title
AND-035	LOW	Missing Network Security Configuration Domain Restriction Granularity

56 CLEARTEXT TRAFFIC

TMG Security tested whether any NovaSphere Pay network request — first-party or third-party SDK — is permitted to fall back to unencrypted HTTP.

Illustrative Tests Performed

- Verify cleartextTrafficPermitted is false for all domains reachable by the application.
- Verify no third-party SDK independently issues cleartext requests bypassing the application's NSC policy.
- Verify no hardcoded http:// URL exists in application code or configuration.

Illustrative Observations

No cleartext HTTP traffic was observed from any fictional first-party endpoint or bundled third-party SDK during dynamic testing, and no hardcoded http:// URL was found in the decompiled codebase. cleartextTrafficPermitted is correctly set to false at the NSC base-config level with no per-domain override. No findings are raised in this section.

No findings raised against this fictional testing category beyond items cross-referenced elsewhere in this report.

57 CERTIFICATE PINNING

TMG Security reviewed NovaSphere Pay's certificate/public-key pinning coverage against OWASP MASVS-NETWORK-2, focused on whether pinning is applied to the sensitive first-party payment and authentication endpoints.

Illustrative Tests Performed

- Verify certificate or public-key pinning is implemented for the primary payments API endpoint.

- Verify pinning configuration includes a documented backup pin and rotation plan.
- Verify pinning failure results in connection termination rather than a silent fallback to unpinned validation.

Illustrative Observations

OkHttp CertificatePinner is correctly configured with a primary and backup pin for api.novasphere-example.com and auth.novasphere-example.com, and pinning failure correctly terminates the connection rather than falling back. The legacy statement-export client covered in Sections 53-54 does not implement pinning, which is consistent with, and does not independently expand, the AND-006 and AND-019 findings already tracked against that client. No additional finding is raised in this section.

No findings raised against this fictional testing category beyond items cross-referenced elsewhere in this report.

58 PROXY / MITM RESISTANCE

TMG Security performed dynamic testing through an interception proxy configured with a fictional CA trusted only on the fictional test devices, to validate the combined effect of the NSC, certificate validation, and pinning controls reviewed above under realistic interception conditions.

Illustrative Tests Performed

- Verify the primary API client rejects interception-proxy traffic under default (non-rooted) trust conditions.
- Verify the primary API client rejects interception-proxy traffic even when a fictional CA is manually trusted at the device level (pinning enforcement).
- Verify the legacy statement-export client's behavior under the same interception conditions is consistent with the AND-006 finding.

Illustrative Observations

The primary API client correctly refused to complete requests through the interception proxy both under default trust and after the fictional CA was manually trusted at the device level, confirming effective certificate pinning enforcement (cross-referenced to Section 57). The legacy statement-export client's TrustManager defect (AND-006) was reproduced consistently under these same interception conditions, confirming the finding rather than raising a new one.

No findings raised against this fictional testing category beyond items cross-referenced elsewhere in this report.

59 AUTHENTICATION

TMG Security reviewed NovaSphere Pay's login flow — credential submission, multi-factor enforcement, and lockout behavior — against OWASP MASVS-AUTH.

Illustrative Tests Performed

- Verify credentials are transmitted only over the TLS-protected primary API channel (cross-referenced from Section 53).
- Verify multi-factor authentication is enforced at login and cannot be bypassed by a direct API call that skips the MFA step.
- Verify account lockout / rate-limiting is applied after repeated failed authentication attempts.

Illustrative Observations

NovaSphere Pay correctly enforces multi-factor authentication at login, and TMG Security's fictional testing confirmed the MFA step cannot be bypassed by calling the session-issuance endpoint directly without a validated MFA assertion — this is recorded as a positive control observation (AND-036). Failed-attempt lockout is applied consistently. No findings are raised

in this section; the authorization weakness identified elsewhere in the mobile API surface is treated separately in Section 65 and Section 67.

No findings raised against this fictional testing category beyond items cross-referenced elsewhere in this report.

60 SESSION MANAGEMENT

TMG Security reviewed NovaSphere Pay's session lifetime, idle-timeout enforcement, and server-side session invalidation behavior.

Illustrative Tests Performed

- Verify an idle-timeout value is enforced and re-authentication is required after it elapses.
- Verify the enforced idle-timeout value is consistent with the sensitivity of a financial application.
- Verify session state is invalidated server-side on logout, not only cleared from the client.

Illustrative Observations

An idle-timeout is enforced, but the configured value (45 minutes) is longer than appropriate for a financial application handling live account and payment data, and is not independently configurable to a shorter value for higher-risk fictional account tiers (AND-022). Server-side session invalidation on explicit logout functions correctly; the related refresh-token revocation gap is tracked separately as AND-011 in Section 61.

Finding ID	Severity	Title
AND-022	MEDIUM	Weak Session Timeout Enforcement

61 TOKEN STORAGE

TMG Security reviewed the full lifecycle of NovaSphere Pay's access and refresh tokens — issuance, local storage, and revocation — extending the local-storage findings from Section 40 with a lifecycle-management focus.

Illustrative Tests Performed

- Verify access and refresh tokens are stored using an appropriate protection mechanism (cross-referenced: AND-001, Section 40).
- Verify the refresh token is revoked server-side when the user explicitly logs out.
- Verify access tokens carry an appropriately short expiry consistent with the sensitivity of the operations they authorize.

Illustrative Observations

The storage-at-rest weakness affecting both the access and refresh token is tracked as AND-001 (Section 40) and is not re-scored here. Independently, TMG Security's fictional testing found that the refresh token remains valid server-side after an explicit logout action — the client discards its local copy, but the token itself continues to authorize new access-token issuance if replayed, extending the practical exposure window of AND-001 (AND-011). Access-token expiry (15 minutes) is appropriately short.

Finding ID	Severity	Title
AND-011	HIGH	Token Lifecycle Weakness — Missing Refresh-Token Revocation on Logout

62 BIOMETRIC AUTHENTICATION

TMG Security reviewed NovaSphere Pay's biometric unlock feature (fingerprint/face, via BiometricPrompt) for correct binding to the underlying cryptographic operation it is meant to authorize.

Illustrative Tests Performed

- Verify BiometricPrompt is used with a CryptoObject bound to a Keystore key, rather than as a bare presentation-layer gate.
- Verify a successful biometric prompt cannot be spoofed by directly invoking the post-authentication callback.
- Verify the app falls back to device credential (PIN/pattern) rather than silently unlocking when biometric hardware is unavailable.

Illustrative Observations

NovaSphere Pay's biometric unlock is implemented as a presentation-layer gate: BiometricPrompt is shown, but its success callback is not bound to a CryptoObject/Keystore-backed cryptographic operation, so a fictional instrumented-testing bypass of the callback proceeds directly to the authenticated app state without the biometric result being cryptographically verified (AND-010). Device-credential fallback behavior when biometric hardware is unavailable functions correctly and is not itself weakened by this finding.

Finding ID	Severity	Title
AND-010	HIGH	Weak Biometric/Local Authentication Enforcement

63 LOCAL AUTHENTICATION

TMG Security reviewed NovaSphere Pay's PIN-based local authentication option (the biometric alternative) for lockout behavior and storage of the PIN verification value.

Illustrative Tests Performed

- Verify the local PIN is never stored or compared in plaintext.
- Verify repeated failed local-PIN attempts trigger a lockout or escalating delay.
- Verify local-PIN verification is bound to the same cryptographic-operation weakness assessed for biometric unlock (cross-referenced from Section 62).

Illustrative Observations

The local PIN is stored as a salted hash and is never compared in plaintext; repeated failed attempts correctly trigger an escalating lockout delay. Because local-PIN verification gates the same presentation-layer unlock callback assessed in Section 62, it inherits that finding's underlying weakness (AND-10) rather than presenting an independent gap; no additional finding is raised in this section. (See AND-010, Section 62.)

No findings raised against this fictional testing category beyond items cross-referenced elsewhere in this report.

64 ACCOUNT RECOVERY

TMG Security reviewed NovaSphere Pay's account-recovery flow (forgotten password / device-loss re-enrollment) for the mobile-specific risk of a recovery flow re-establishing trust with weaker verification than the original enrollment.

Illustrative Tests Performed

- Verify account recovery requires re-verification through a channel independent of the potentially compromised device.
- Verify a recovered account cannot skip the MFA enrollment step present in original account creation.
- Verify device-binding state from the prior device is invalidated once recovery completes on a new device.

Illustrative Observations

Account recovery correctly requires out-of-band email and SMS verification independent of the requesting device, re-enrollment in MFA is mandatory and cannot be skipped, and prior device-binding state is invalidated once recovery completes on a new fictional device. No findings are raised against this testing area.

No findings raised against this fictional testing category beyond items cross-referenced elsewhere in this report.

65 AUTHORIZATION

TMG Security reviewed how NovaSphere Pay's mobile client enforces server-side authorization decisions for account-scoped and payment-scoped actions, previewing the full technical detail presented as an API-focused assessment in Section 67 (Broken Object-Level Authorization).

Illustrative Tests Performed

- Verify every account-scoped API request is authorized against the authenticated session's own account identifier server-side.
- Verify no sensitive account action can be authorized using only client-supplied identifiers without server-side ownership validation.
- Verify step-up authentication is required for the highest-value transaction types beyond standard session authorization.

Illustrative Observations

TMG Security's fictional testing identified a broken-object-level-authorization condition in the funds-transfer confirmation workflow, in which the server trusts a client-supplied source-account identifier without validating that it belongs to the authenticated session (AND-003); full technical detail, including the fictional proof-of-concept request sequence, is presented in Section 67. Separately, TMG Security recommends the application adopt step-up re-authentication for the highest-value transaction tier as a defense-in-depth enhancement beyond standard session authorization (AND-041).

Finding ID	Severity	Title
AND-003	CRITICAL	Broken Authorization in Mobile API Workflow
AND-041	INFORMATIONAL	Recommendation — Adopt Step-Up Attestation for High-Value Sensitive Transactions

66 API SECURITY INTEGRATION

Sections 67-71 shift focus from the Android client in isolation to how NovaSphere Pay's mobile client integrates with its fictional backend API surface (api.novasphere-example.com) — object-level authorization, business-logic enforcement, rate limiting, sensitive-data exposure in API responses, and error handling. This chapter presents the full technical detail of this report's third Critical finding, AND-003, a broken object-level authorization condition in the funds-transfer workflow, previewed in Section 65.

67 BROKEN OBJECT-LEVEL AUTHORIZATION (BOLA)

TMG Security tested every account-scoped and payment-scoped API endpoint for object-level authorization enforcement, submitting requests referencing account identifiers other than the authenticated fictional test session's own.

Illustrative Tests Performed

- Verify the funds-transfer confirmation endpoint validates that the client-supplied source-account identifier belongs to the authenticated session.
- Verify the statement-retrieval endpoint cannot be used to retrieve another fictional account's statement by identifier substitution.
- Verify the beneficiary-management endpoints enforce account-ownership checks server-side, not only in client-side UI logic.

Illustrative Observations

The POST /v2/transfers/confirm endpoint accepts a sourceAccountId parameter from the client and processes the transfer against that account without validating it belongs to the authenticated session, allowing Test Account A's authenticated session to confirm a fictional transfer sourced from Test Account B's balance in TMG Security's fictional proof-of-concept (AND-003). The statement-retrieval and beneficiary-management endpoints tested correctly enforced server-side ownership checks and did not exhibit the same weakness.

Finding ID	Severity	Title
AND-003	CRITICAL	Broken Authorization in Mobile API Workflow

68 BUSINESS LOGIC

TMG Security tested NovaSphere Pay's transfer and payment workflows for business-logic weaknesses independent of authentication or authorization controls — transaction-limit enforcement, duplicate-submission handling, and state-machine consistency.

Illustrative Tests Performed

- Verify per-transaction and daily cumulative transfer limits are enforced server-side, not only in client UI.
- Verify duplicate/replayed transfer submissions are detected and rejected (idempotency enforcement).
- Verify a transfer cannot be moved into an inconsistent state by resubmitting an earlier step out of sequence.

Illustrative Observations

Transfer limits are correctly enforced server-side and could not be bypassed by manipulating the client request. Idempotency keys are correctly required and validated on the transfer-confirmation endpoint, preventing duplicate submission from resulting in a duplicate transfer. Out-of-sequence step resubmission was correctly rejected by the workflow's server-side state machine. No findings are raised against this testing area.

No findings raised against this fictional testing category beyond items cross-referenced elsewhere in this report.

69 RATE LIMITING

TMG Security tested authentication, MFA-verification, and transfer-confirmation endpoints for request-rate throttling under a sustained fictional automated-request scenario.

Illustrative Tests Performed

- Verify authentication and MFA-verification endpoints throttle repeated requests from the same fictional session/device.
- Verify the transfer-confirmation endpoint applies a reasonable rate limit independent of the per-transaction value limit.
- Verify rate-limit responses do not themselves disclose account-enumeration information.

Illustrative Observations

All tested endpoints applied consistent, appropriately scoped rate limiting during this fictional assessment, and throttled responses did not disclose account-enumeration signal beyond a generic rate-limit status. No findings are raised against this testing area.

No findings raised against this fictional testing category beyond items cross-referenced elsewhere in this report.

70 SENSITIVE DATA EXPOSURE (API RESPONSES)

TMG Security reviewed API response payloads across the tested endpoint set for over-inclusion of sensitive fields beyond what each screen actually renders, a common source of client-side data exposure independent of the storage- and logging-specific findings raised elsewhere in this report.

Illustrative Tests Performed

- Verify API responses do not include full unmasked account/card numbers where only a masked value is displayed.
- Verify statement/transaction-history responses do not include internal-only fields beyond the mobile client's display requirements.
- Verify no API response field duplicates data already flagged as over-exposed via caching (cross-referenced: AND-018, Section 42) or logging (cross-referenced: AND-007, Section 32).

Illustrative Observations

Account and card numbers are correctly masked server-side before transmission, and statement/transaction-history responses were not found to over-include internal-only fields beyond the mobile client's display requirements. The sensitive-data exposure identified in this assessment occurs after the response reaches the device, through caching (AND-018) and logging (AND-007) rather than through response over-inclusion itself; no additional finding is raised in this section.

No findings raised against this fictional testing category beyond items cross-referenced elsewhere in this report.

71 ERROR HANDLING

TMG Security submitted malformed and boundary-condition requests to NovaSphere Pay's API surface to assess whether error responses disclose internal implementation detail useful to an attacker.

Illustrative Tests Performed

- Verify error responses return a generic, client-safe message rather than a raw exception or stack trace.
- Verify error responses do not disclose internal service names, database structure, or file-system paths.

- Verify HTTP status codes used for error conditions do not themselves leak account-existence information beyond what the workflow already requires.

Illustrative Observations

A malformed request to the beneficiary-management endpoint returned a raw fictional stack trace including an internal service name and a partial ORM query fragment, rather than the generic error message returned by other tested endpoints (AND-021). This is an information-disclosure weakness that could assist an attacker in mapping the backend's internal structure; it does not itself grant unauthorized access.

Finding ID	Severity	Title
AND-021	MEDIUM	Verbose Error Handling Reveals Internal Implementation Details

72 WEBVIEW & BROWSER ATTACK SURFACE

This section consolidates NovaSphere Pay's WebView-related attack surface as a single risk view, drawing together the navigation-policy and JavaScript-interface findings detailed individually in Sections 33-34.

Illustrative Tests Performed

- Verify the aggregate WebView attack surface (navigation policy + JavaScript bridge) is scoped to the minimum required for the support/help feature.
- Verify no other application screen embeds a WebView beyond the single support/help instance identified during static and dynamic testing.

Illustrative Observations

Static and dynamic testing confirmed exactly one WebView instance in NovaSphere Pay, used solely for the in-app support/help screen. Its two constituent weaknesses — unrestricted navigation policy and an unauthenticated JavaScript bridge — are tracked individually as AND-005; no additional finding is raised at the consolidated level.

Finding ID	Severity	Title
AND-005	HIGH	WebView JavaScript Interface Exposure

73 DEEP LINKS / URI HANDLING

This section consolidates NovaSphere Pay's deep-link and custom-URI-scheme attack surface as a single risk view, drawing together the sensitive-action deep-link finding detailed individually in Section 23.

Illustrative Tests Performed

- Verify the aggregate deep-link/URI attack surface is limited to the specific intents enumerated in Section 23.
- Verify no additional custom URI scheme beyond novasphere:// was identified during consolidated review.

Illustrative Observations

No additional deep-link or custom-URI-scheme entry point beyond the novasphere:// handlers assessed in Section 23 was identified. The sensitive-account-action deep-link weakness remains tracked as AND-004; no additional finding is raised at the consolidated level.

Finding ID	Severity	Title
AND-004	CRITICAL	Insecure Deep-Link Flow Enables Sensitive Account Action

74 INTENT SECURITY

This section consolidates NovaSphere Pay's general (non-deep-link) intent-handling attack surface, drawing together the intent-spoofing/task-hijacking finding detailed individually in Section 19.

Illustrative Tests Performed

- Verify the aggregate intent-handling attack surface is limited to the specific activities enumerated in Section 19.
- Verify implicit intents used for internal component communication are not independently reachable by a third-party fictional app.

Illustrative Observations

The intent-handling weakness affecting the payment-confirmation activity flow remains tracked as AND-012 (Section 19). No additional intent-handling weakness was identified during consolidated review; no additional finding is raised at the consolidated level.

Finding ID	Severity	Title
AND-012	HIGH	Improper Intent Handling Enables Intent Spoofing / Task Hijacking Risk

75 IPC SECURITY

TMG Security reviewed NovaSphere Pay's overall exported-component footprint as a single attack-surface metric, consolidating the individual exported-activity, exported-service, and exported-content-provider findings from Sections 19-22 and assessing whether the total count of exported components is proportionate to the application's genuine cross-app integration needs.

Illustrative Tests Performed

- Verify each exported component serves a documented, necessary cross-app integration purpose.
- Verify the total exported-component count does not exceed what the application's integration requirements justify.

Illustrative Observations

NovaSphere Pay exports eleven Android components in total across activities, services, and content providers. Of these, four are individually assessed as carrying a confirmed weakness (AND-002, AND-004, AND-009 as a provider-adjacent path, AND-027), and TMG Security additionally recommends reducing the overall exported-component count where cross-app integration is not strictly required, since each exported component independently widens the fictional application's IPC attack surface regardless of its individual configuration correctness (AND-032).

Finding ID	Severity	Title
AND-032	LOW	Attack Surface Hardening Opportunity — Exported Component Count

76 OVERLAY / TAPJACKING PROTECTION

TMG Security tested whether NovaSphere Pay's sensitive confirmation screens (transfer confirmation, beneficiary addition) are protected against overlay-based tapjacking using the FilterTouchesWhenObscured mechanism or equivalent.

Illustrative Tests Performed

- Verify sensitive confirmation screens set filterTouchesWhenObscured or an equivalent overlay-detection mechanism.
- Verify a fictional full-screen overlay cannot intercept a tap intended for the underlying confirmation button.

Illustrative Observations

TMG Security's fictional overlay-simulation testing found that the transfer-confirmation and beneficiary-addition screens do not set filterTouchesWhenObscured, allowing a fictional malicious overlay app to intercept taps intended for the confirmation button without the user's awareness (AND-025). Standard, non-sensitive screens are not affected by this finding.

Finding ID	Severity	Title
AND-025	MEDIUM	Tapjacking / Overlay Protection Not Implemented on Sensitive Confirmation Screens

77 SCREENSHOT / SCREEN CAPTURE PROTECTION

This section consolidates NovaSphere Pay's FLAG_SECURE coverage as a single attack-surface view, drawing together the Recent-Apps-thumbnail and screen-recording finding detailed individually in Section 45.

Illustrative Tests Performed

- Verify FLAG_SECURE coverage is assessed consistently across every sensitive screen, not only the two identified in Section 45.

Illustrative Observations

TMG Security's consolidated review confirmed the FLAG_SECURE gap is limited to AccountOverviewActivity and CardDetailActivity, as identified in Section 45 (AND-016); all other sensitive screens reviewed (transfer confirmation, beneficiary management, settings) correctly set FLAG_SECURE. No additional finding is raised at the consolidated level.

Finding ID	Severity	Title
AND-016	MEDIUM	Sensitive Data Exposed via Screenshot / Recent-Apps Thumbnail

78 BACKUP & RESTORE

TMG Security reviewed NovaSphere Pay's Android backup configuration (android:allowBackup and any custom BackupAgent) for exposure of sensitive local data through the platform's backup/restore mechanism.

Illustrative Tests Performed

- Verify android:allowBackup is set to false, or a custom BackupAgent explicitly excludes sensitive files.
- Verify a fictional adb backup of the application produces no recoverable sensitive data.

Illustrative Observations

NovaSphere Pay declares android:allowBackup="true" with no custom BackupAgent or backup-rules exclusion, and a fictional adb backup extraction on a rooted fictional test device successfully recovered the same SharedPreferences file containing the cleartext session material identified in Section 40, providing an additional extraction path for the data underlying AND-001 (AND-015).

Finding ID	Severity	Title
AND-015	MEDIUM	Insecure Backup Configuration (allowBackup Enabled)

79 EXTERNAL APP INTERACTION

TMG Security reviewed NovaSphere Pay's interaction surface with other installed applications beyond IPC/intents already covered — specifically Accessibility Service interaction and any use of platform sharing APIs.

Illustrative Tests Performed

- Verify the application does not expose UI elements or actions in a way that an abusive Accessibility Service could exploit beyond standard platform-level Accessibility Service risk.
- Verify the ACTION_SEND / share-sheet integration does not inadvertently share more data than the user selected.

Illustrative Observations

NovaSphere Pay does not explicitly restrict or monitor Accessibility Service interaction with its sensitive UI elements beyond the platform-default protections, which is consistent with common practice but is noted as an observation given the elevated capability a malicious Accessibility Service holds on a compromised fictional device (AND-045). Share-sheet integration for statement export was confirmed to share only the user-selected file with no additional data attached.

Finding ID	Severity	Title
AND-045	INFORMATIONAL	Observation — Accessibility Service Interaction Not Explicitly Restricted

80 REVERSE ENGINEERING RESISTANCE

Sections 81-87 assess NovaSphere Pay's resilience posture against OWASP MASVS-RESILIENCE — the set of controls intended to slow static and dynamic reverse engineering, detect tampering and repackaging, and resist execution in a rooted, emulated, or debugged fictional environment. TMG Security notes throughout this chapter that MASVS-RESILIENCE controls are explicitly defense-in-depth: their absence does not itself constitute a directly exploitable vulnerability, and no finding in this chapter is rated above High for that reason, consistent with OWASP's own guidance on how resilience controls should be weighted relative to the other seven MASVS categories.

81 OBFUSCATION

TMG Security assessed the decompiled fictional codebase's readability as a proxy for R8/ProGuard obfuscation coverage, extending the initial observation raised in Section 27.

Illustrative Tests Performed

- Verify class, method, and field names are obfuscated in the release build across all application modules.
- Verify string-literal obfuscation or encryption is applied to security-relevant constants beyond simple identifier renaming.

Illustrative Observations

Identifier obfuscation is applied inconsistently: the core banking and payments modules are fully obfuscated, but two integration modules (statement export, support/help) retain fully readable class and method names, which meaningfully eased TMG Security's fictional static-analysis review of those modules. This consolidates and does not duplicate the recommendation already tracked as AND-039 (Section 27).

Finding ID	Severity	Title
AND-039	INFORMATIONAL	Recommendation — Expand Obfuscation Coverage (R8/ProGuard)

82 TAMPER DETECTION

TMG Security reviewed NovaSphere Pay's runtime tamper-detection controls — signature/checksum self-verification and any use of a platform attestation service — for coverage and bypass resistance.

Illustrative Tests Performed

- Verify the application verifies its own signing-certificate hash at runtime and reacts appropriately to a mismatch.
- Verify the application uses a platform attestation service (e.g., Play Integrity API) rather than relying solely on client-side self-checks.

Illustrative Observations

NovaSphere Pay performs a client-side signing-certificate hash check at startup, which TMG Security's fictional instrumented-testing bypassed by hooking the verification method directly, consistent with the general limitation that any purely client-side integrity check can be defeated on a fully compromised fictional device. TMG Security recommends the application adopt the Play Integrity API as its primary resilience signal, evaluated server-side, rather than relying on the client-side check as its sole tamper-detection mechanism (AND-038).

Finding ID	Severity	Title
AND-038	INFORMATIONAL	Recommendation — Adopt Play Integrity API as Primary Resilience Signal

83 REPACKAGING RESISTANCE

TMG Security assessed whether a fictional repackaged (re-signed) copy of NovaSphere Pay could be installed and would function against the fictional backend, and reviewed the documented app-signing key management process supporting this control.

Illustrative Tests Performed

- Verify the application detects execution under a signing certificate other than the official release certificate.
- Verify the app-signing key management process (custody, rotation, revocation authority) is formally documented.

Illustrative Observations

The signing-certificate check assessed in Section 82 also governs repackaging detection and carries the same client-side bypass limitation (AND-038, not re-scored here). Separately, TMG Security's fictional document review found the app-signing key management process is understood by the mobile engineering team but is not formally documented as a standalone policy, which the assessment records as a documentation observation warranting review rather than a technical vulnerability (AND-047).

Finding ID	Severity	Title
AND-047	INFORMATIONAL	Observation — App Signing Key Management Process Warrants Documentation Review

84 ROOT DETECTION

TMG Security tested NovaSphere Pay's root-detection controls on rooted fictional test devices using common root-hiding techniques, as the entry point into the fuller runtime-integrity assessment in Section 87.

Illustrative Tests Performed

- Verify the application detects common root indicators (su binary, Magisk, root-management apps) on startup.
- Verify root detection cannot be trivially bypassed using a widely available root-hiding module.

Illustrative Observations

NovaSphere Pay's root-detection routine correctly flagged an unmodified rooted fictional test device, but was bypassed using a standard root-hiding module during TMG Security's fictional dynamic-instrumentation testing, consistent with the client-side tamper-detection limitation already tracked as AND-013 (full technical detail in Section 87). No additional finding is raised in this section.

No findings raised against this fictional testing category beyond items cross-referenced elsewhere in this report.

85 EMULATOR DETECTION

TMG Security tested NovaSphere Pay's behavior when launched inside a fictional Android emulator instance rather than a physical device.

Illustrative Tests Performed

- Verify the application detects common emulator indicators (build fingerprint, QEMU artifacts, sensor absence).
- Verify emulator-detected sessions are appropriately restricted or flagged for elevated monitoring server-side.

Illustrative Observations

The application correctly detected the fictional emulator environment via build-fingerprint and QEMU-artifact checks and applied the intended restricted-mode behavior. No findings are raised against this testing area.

No findings raised against this fictional testing category beyond items cross-referenced elsewhere in this report.

86 DEBUGGER DETECTION

TMG Security tested whether NovaSphere Pay detects and reacts to an attached fictional debugger (via Debug.isDebuggerConnected() or an equivalent native-layer check) during runtime.

Illustrative Tests Performed

- Verify the application detects an attached debugger at both the Java/Kotlin and native layers.
- Verify a detected debugger triggers a defined defensive response rather than being silently logged.

Illustrative Observations

Java-layer debugger detection is implemented and correctly triggers session termination when a fictional debugger is attached; no equivalent native-layer (JNI) debugger-detection check was identified, though NovaSphere Pay's fictional native-code footprint is minimal. This gap is noted as a minor observation and does not independently warrant a standalone finding given the limited native attack surface it would protect.

No findings raised against this fictional testing category beyond items cross-referenced elsewhere in this report.

87 RUNTIME INTEGRITY

TMG Security consolidated the root-detection, tamper-detection, and debugger-detection results above into a single runtime-integrity assessment, and additionally tested resistance to runtime instrumentation frameworks directly.

Illustrative Tests Performed

- Verify the application detects the presence of a runtime instrumentation framework (e.g., a Frida-class toolkit) independent of root status.
- Verify a detected instrumentation framework triggers a defensive response consistent with the application's other integrity checks.

Illustrative Observations

TMG Security's fictional dynamic-instrumentation testing successfully attached a runtime instrumentation toolkit to NovaSphere Pay and used it to bypass the root-detection check (Section 84) and hook the signing-certificate verification routine (Section 82), confirming that the application's client-side integrity controls, taken together, do not resist an actively instrumented fictional test environment (AND-013). This finding is rated High rather than Critical because it requires a compromised (rooted/instrumented) device as a precondition and does not independently grant unauthorized access to account data.

Finding ID	Severity	Title
AND-013	HIGH	Root / Runtime Integrity Control Bypass

88 PRIVACY

Sections 89-92 assess NovaSphere Pay against OWASP MASVS-PRIVACY — permission minimization, handling of personally identifiable information, third-party analytics/tracking behavior, and UI-level exposure of private data. This chapter consolidates rather than duplicates the permission findings raised during manifest review (Section 17) and the analytics-SDK finding raised during third-party SDK review (Section 37), presenting them alongside the privacy-specific controls tested for the first time in this chapter.

89 PERMISSION ANALYSIS

TMG Security reviewed NovaSphere Pay's full requested-permission set against the minimum permissions its documented feature set requires, consolidating the manifest-level findings from Section 17 into a privacy-focused view.

Illustrative Tests Performed

- Verify every requested dangerous-level permission maps to a documented, active feature.
- Verify no permission is requested at install-time where a runtime, feature-triggered request would be more privacy-preserving.
- Verify unused permissions identified in prior builds have been removed rather than merely deprecated in code.

Illustrative Observations

This consolidated privacy-focused review confirms the same two permission-related conditions identified during manifest review: a set of permissions broader than the app's active feature set requires (AND-020) and permissions retained from now-removed features (AND-028). Both are tracked at their original finding IDs and are not re-scored here.

Finding ID	Severity	Title
AND-020	MEDIUM	Excessive Android Permissions Requested
AND-028	LOW	Unused / Unnecessary Permissions Declared

90 PII HANDLING

TMG Security reviewed how NovaSphere Pay collects, transmits, and locally retains personally identifiable information (full name, government ID fragment used for identity verification, date of birth) collected during fictional account onboarding.

Illustrative Tests Performed

- Verify PII collected during onboarding is transmitted only over the TLS-protected primary API channel.
- Verify PII is not retained in local storage beyond the active onboarding session once submitted.
- Verify PII is not written to application logs (cross-referenced: AND-007, Section 32).

Illustrative Observations

PII collected during onboarding is transmitted correctly over the protected channel and is not persisted locally beyond the active onboarding session. No PII field beyond the already-tracked log exposure (AND-007) was found retained or exposed outside its intended handling path. No additional findings are raised in this section.

No findings raised against this fictional testing category beyond items cross-referenced elsewhere in this report.

91 ANALYTICS / TRACKING SDKS

TMG Security reviewed the data-collection behavior of NovaSphere Pay's bundled analytics and crash-reporting SDKs, consolidating the findings raised during third-party SDK review in Section 37 into a privacy-focused view.

Illustrative Tests Performed

- Verify the analytics SDK does not transmit a persistent device or user identifier beyond what its documented purpose requires.
- Verify the crash-reporting SDK's default breadcrumb configuration does not capture PII or sensitive UI state.
- Verify users are given a clear, functioning opt-out for non-essential analytics collection.

Illustrative Observations

The analytics SDK's persistent-identifier reuse across fictional install/reinstall cycles remains tracked as AND-023 (Section 37), and the crash-reporting SDK's default breadcrumb configuration warranting a PII-scrubbing review remains tracked as the informational observation AND-043 (Section 37). Analytics opt-out functions correctly and was not itself found to carry a weakness. Neither finding is re-scored in this section.

Finding ID	Severity	Title
AND-023	MEDIUM	Analytics/Tracking SDK Privacy Weakness — Persistent Identifier Reuse
AND-043	INFORMATIONAL	Observation — Crash Reporting SDK Configuration Should Be Reviewed for PII Scrubbing

92 CLIPBOARD / NOTIFICATION EXPOSURE

This section consolidates NovaSphere Pay's clipboard exposure (Section 44) and reviews push-notification content for sensitive-data leakage onto the fictional device's lock screen.

Illustrative Tests Performed

- Verify push notifications do not display full sensitive values (account balance, transaction amount) on the lock screen by default.
- Verify the clipboard exposure identified in Section 44 is not compounded by an additional clipboard-writing code path.

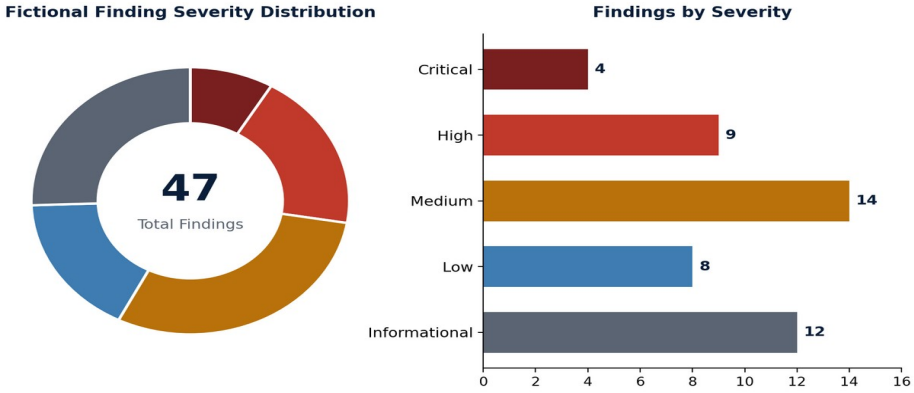
Illustrative Observations

Push-notification content correctly displays a generic, non-sensitive message on the lock screen, with full detail visible only after device unlock — no finding is raised against notification content. The clipboard-exposure finding remains tracked at its original ID (AND-017, Section 44); no additional clipboard-writing code path was identified during consolidated review.

Finding ID	Severity	Title
AND-017	MEDIUM	Sensitive Data Exposure via Clipboard

93 FINDINGS SUMMARY

This fictional assessment identified 47 illustrative findings across NovaSphere Pay's Android client and its supporting mobile API integration, summarized below: 4 Critical, 9 High, 14 Medium, 8 Low, and 12 Informational. Full detail for every finding, including safe proof-of-concept evidence and OWASP MASVS/MASWE/MASTG traceability, is provided in Sections 94-98, organized by severity.



Illustrative severity distribution — ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE.

Findings at a Glance

Finding ID	Title	Severity
AND-001	Sensitive Authentication Token Exposure Through Insecure Local Storage	CRITICAL
AND-002	Exported Android Component Enables Unauthorized Privileged Action	CRITICAL
AND-003	Broken Authorization in Mobile API Workflow	CRITICAL
AND-004	Insecure Deep-Link Flow Enables Sensitive Account Action	CRITICAL
AND-005	WebView JavaScript Interface Exposure	HIGH
AND-006	Certificate Validation Weakness	HIGH
AND-007	Sensitive Data Exposure Through Logs	HIGH
AND-008	Hardcoded API Secret in Application Package	HIGH
AND-009	Insecure FileProvider Configuration	HIGH
AND-010	Weak Biometric/Local Authentication Enforcement	HIGH
AND-011	Token Lifecycle Weakness — Missing Refresh-Token Revocation on Logout	HIGH
AND-012	Improper Intent Handling Enables Intent Spoofing / Task Hijacking Risk	HIGH
AND-013	Root / Runtime Integrity Control Bypass	HIGH
AND-014	Debug Information Exposure in Pre-Production Build Configuration	MEDIUM
AND-015	Insecure Backup Configuration (allowBackup Enabled)	MEDIUM
AND-016	Sensitive Data Exposed via Screenshot / Recent-Apps Thumbnail	MEDIUM
AND-017	Sensitive Data Exposure via Clipboard	MEDIUM
AND-018	Weak Cache Handling of Sensitive Response Data	MEDIUM
AND-019	Deprecated TLS Configuration Permitted by Network Security Configuration	MEDIUM
AND-020	Excessive Android Permissions Requested	MEDIUM
AND-021	Verbose Error Handling Reveals Internal Implementation Details	MEDIUM
AND-022	Weak Session Timeout Enforcement	MEDIUM
AND-023	Analytics/Tracking SDK Privacy Weakness — Persistent Identifier Reuse	MEDIUM
AND-024	Insecure Random Number Generation for a Security-Relevant Value	MEDIUM
AND-025	Tapjacking / Overlay Protection Not Implemented on Sensitive Confirmation Screens	MEDIUM
AND-026	Unencrypted Room/SQLite Database Storage of Transaction History	MEDIUM
AND-027	Exported Content Provider Permits Non-Critical Data Read	MEDIUM

Finding ID	Title	Severity
AND-028	Unused / Unnecessary Permissions Declared	LOW
AND-029	Use of Deprecated Android APIs	LOW
AND-030	General Logging Hygiene Weaknesses in Non-Sensitive Modules	LOW
AND-031	Third-Party Dependency Governance Gaps	LOW
AND-032	Attack Surface Hardening Opportunity — Exported Component Count	LOW
AND-033	Missing Security Regression Test Automation	LOW
AND-034	Incomplete Secure-Coding Documentation for Mobile Engineering	LOW
AND-035	Missing Network Security Configuration Domain Restriction Granularity	LOW
AND-036	Positive Control Observation — Multi-Factor Authentication Enforced at Login	INFORMATIONAL
AND-037	Positive Control Observation — TLS 1.2+ Enforced Platform-Wide on the Primary API Client	INFORMATIONAL
AND-038	Recommendation — Adopt Play Integrity API as Primary Resilience Signal	INFORMATIONAL
AND-039	Recommendation — Expand Obfuscation Coverage (R8/ProGuard)	INFORMATIONAL
AND-040	Recommendation — Formalize Secrets Management for the Mobile Build Pipeline	INFORMATIONAL
AND-041	Recommendation — Adopt Step-Up Attestation for High-Value Sensitive Transactions	INFORMATIONAL
AND-042	Observation — Third-Party SDK Inventory Should Be Centrally Tracked	INFORMATIONAL
AND-043	Observation — Crash Reporting SDK Configuration Should Be Reviewed for PII Scrubbing	INFORMATIONAL
AND-044	Recommendation — Formal Secure-SDLC Checkpoint for Mobile Releases	INFORMATIONAL
AND-045	Observation — Accessibility Service Interaction Not Explicitly Restricted	INFORMATIONAL
AND-046	Recommendation — Periodic Dependency Vulnerability Scanning Cadence	INFORMATIONAL
AND-047	Observation — App Signing Key Management Process Warrants Documentation Review	INFORMATIONAL

Finding Format

Each finding card in Sections 94-98 presents: Finding ID, Title, Severity, CVSS-style Score and Vector (Critical/High), CWE reference, Affected Component, Affected Android Component, Description, Business Context, Technical Impact, Business Impact, Attack Preconditions (where applicable), a Safe Proof of Concept summary, Evidence Reference, Expected Behavior, Observed (Fictional) Behavior, Root Cause, Recommendation, Management Response, Owner/Priority/Target Date/Status, a Retest Recommendation, and an OWASP MASVS/MASWE/MASTG traceability strip. Critical and High findings add a Technical Evidence Summary (Attack Preconditions, Observed Result, Security Impact, Business Impact, Evidence Collected, Retest Validation).

Safe Proof-of-Concept Standard

Every Proof of Concept in this report is illustrative and fictional. All evidence references the fictional application package com.novasphere.pay, build NovaSpherePay-release-6.4.2.apk, fictional test devices, and the non-resolving placeholder domains api.novasphere-example.com and auth.novasphere-example.com. No PoC in this report was executed against a real system, and none is destructive, weaponized, or intended to enable real-world exploitation. Any synthetic screenshot or mock-tool evidence depicts no real application, device, or data and is marked ILLUSTRATIVE / SYNTHETIC EVIDENCE.

OWASP Reference Basis

Every MASVS control, MASWE weakness, and MASTG test identifier referenced in Sections 94-98 is drawn from current, publicly verifiable OWASP Mobile Application Security material (MASVS, MASTG v2.0.0, MASWE v1.0.0). Where a precise verified identifier could not be confidently confirmed for a specific control concept, this report describes the concept in narrative terms rather than presenting an invented identifier. This report is an independent illustrative sample and carries no OWASP affiliation, review, or certification.

CVSS Scoring Note

All CVSS-style scores and vectors shown in this report — including the illustrative vector strings on each Critical and High finding — are sample estimations constructed for this fictional exercise. They are not derived from any real vulnerability, are not submitted to any CVSS authority, and should not be treated as authoritative scoring for a real system.

ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE — every request, response, adb/logcat capture, decompiled excerpt, screenshot, CVSS score/vector, and evidence reference in Sections 94-98 is synthetic.

94 CRITICAL FINDINGS

The following 4 fictional Critical-severity findings represent this fictional assessment's highest-priority illustrative results, each reproducible with a controlled, non-destructive fictional Proof of Concept and each carrying a direct path toward fictional account or funds compromise.

AND-001 — Sensitive Authentication Token Exposure Through Insecure Local Storage	CRITICAL
CVSS-Style Score: 9.1 (Critical) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-312 — Cleartext Storage of Sensitive Information**
CVSS Vector (Illustrative): `CVSS:3.1/AV:P/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:N — Illustrative vector only`	
Affected Component	Local session/token persistence layer
Affected Android Component	*SharedPreferences file `novasphere_session_prefs.xml` (MODE_PRIVATE, unencrypted)*
Description	NovaSphere Pay persists the OAuth 2.0 access token, refresh token, and a fictional device-binding secret in a plaintext SharedPreferences XML file rather than in the Android Keystore-backed EncryptedSharedPreferences/Jetpack Security abstraction. On a fictional rooted test device, or on a device with an unpatched backup-extraction path, the token material was recoverable directly from the app's private data directory without any cryptographic barrier.
Business Context	NovaSphere Pay's session and refresh tokens are the fictional bearer credentials for all authenticated payment, transfer, and account-management operations against the NovaSphere Enterprise Platform API. Their confidentiality is foundational to every other authorization control in the application.
Technical Impact	Any fictional process, backup mechanism, or forensic extraction with access to the app's private storage (root access, ADB backup on a debuggable/backup-enabled build, or a device-level compromise) can read the tokens in cleartext with no additional cryptographic barrier.
Business Impact	Full fictional account takeover is possible without the victim's password: a recovered token can be replayed against the NovaSphere API to perform payments, transfers, and profile changes as the fictional victim user until the token is revoked.
Attack Preconditions	Local or ADB-level access to the fictional device's app-private storage (e.g., a rooted fictional test device, or a fictional unpatched backup-extraction path) while the victim is logged in.
Safe Proof of Concept (Summary)	*On a fictional rooted test device, `run-as com.novasphere.pay` followed by reading the app's `shared_prefs` directory returned the fictional session and refresh tokens in cleartext. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-001, EV-AND-002 (fictional shared_prefs extraction, MASTG-TEST-0001 static/dynamic review notes)*
Expected Behavior	Sensitive tokens should be stored using EncryptedSharedPreferences (Jetpack Security Crypto) or an equivalent Keystore-backed envelope, so extraction of the raw file does not yield usable credential material.
Observed (Fictional) Behavior	The fictional access token, refresh token, and device-binding secret were all stored in a standard, unencrypted SharedPreferences file.
Root Cause	The fictional session-management module was implemented against the plain SharedPreferences API rather than the EncryptedSharedPreferences / Keystore-backed pattern recommended for credential material.
Recommendation	Migrate all session/token persistence to EncryptedSharedPreferences (or a custom Keystore-wrapped AES-GCM envelope); ensure the underlying Keystore key is hardware-backed where the device supports it; add static-analysis tooling to flag any future plaintext persistence of token-like values.
Management Response	*NovaSphere engineering (fictional) has accepted this finding and will migrate token storage to EncryptedSharedPreferences ahead of the next scheduled release, tracked as the top remediation priority from this fictional assessment.*
Owner / Priority / Target / Status	**Mobile Engineering Lead (Fictional Role)** | Immediate — 0-30 Days | 10 October 2026 | **Remediation In Progress**
Retest Recommendation	*Retest by repeating the fictional shared_prefs extraction against the remediated build and confirming only opaque, Keystore-wrapped ciphertext is recoverable.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-STORAGE-1 — "The app securely stores sensitive data."	MASWE-0001 — Sensitive Data Stored Unencrypted in Private Storage	MASTG-TEST-0001 — Testing Local Storage for Sensitive Data

FULL EVIDENCE PANEL — ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE

```
ADB / Shell Extraction (Fictional Test Device) (Fictional / Synthetic)
$ adb shell
generic_x86_64:/ $ run-as com.novasphere.pay
generic_x86_64:/data/data/com.novasphere.pay $ cat shared_prefs/novasphere_session_prefs.xml
```

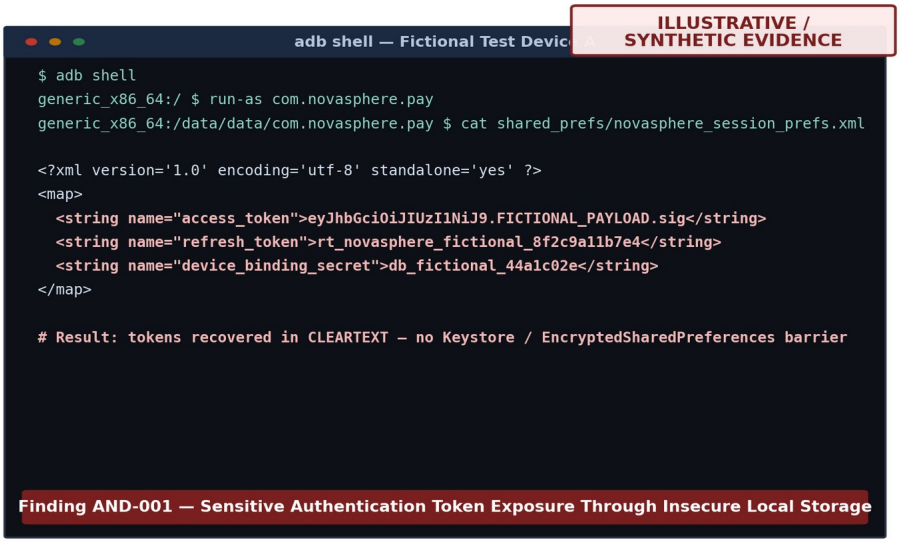
Recovered File Contents (Fictional / Synthetic) (Fictional / Synthetic)

```
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
    <string name="access_token">eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.FICTIONAL_PAYLOAD.sig</string>
    <string name="refresh_token">rt_novasphere_fictional_8f2c9a11b7e4</string>
    <string name="device_binding_secret">db_fictional_44a1c02e</string>
    <long name="token_issued_at" value="1799999999" />
</map>
```

Expected: Access and refresh tokens should be stored only inside EncryptedSharedPreferences / a Keystore-wrapped envelope, unrecoverable without the platform Keystore.

Observed: Both tokens and the device-binding secret were recoverable in cleartext directly from the app's private SharedPreferences file.

Impact: Full fictional session and refresh-token compromise, enabling fictional account takeover and continued token replay after logout if the refresh token is not separately revoked.



Synthetic mock-tool evidence — ILLUSTRATIVE / SYNTHETIC EVIDENCE. No real application, device, or data is depicted.

TECHNICAL EVIDENCE SUMMARY

Attack Preconditions	Local or ADB-level access to the fictional device's app-private storage (e.g., a rooted fictional test device, or a fictional unpatched backup-extraction path) while the victim is logged in.
Observed Result	The fictional access token, refresh token, and device-binding secret were all stored in a standard, unencrypted SharedPreferences file.
Security Impact	Any fictional process, backup mechanism, or forensic extraction with access to the app's private storage (root access, ADB backup on a debuggable/backup-enabled build, or a device-level compromise) can read the tokens in cleartext with no additional cryptographic barrier.
Business Impact	Full fictional account takeover is possible without the victim's password: a recovered token can be replayed against the NovaSphere API to perform payments, transfers, and profile changes as the fictional victim user until the token is revoked.
Evidence Collected	EV-AND-001, EV-AND-002 (fictional shared_prefs extraction, MASTG-TEST-0001 static/dynamic review notes)
Retest Validation	Retest by repeating the fictional shared_prefs extraction against the remediated build and confirming only opaque, Keystore-wrapped ciphertext is recoverable.

AND-002 — Exported Android Component Enables Unauthorized Privileged Action	**CRITICAL**
CVSS-Style Score: 8.8 (Critical) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-926 — Improper Export of Android Application Components**
CVSS Vector (Illustrative): `CVSS:3.1/AV:L/AC:L/PR:N/UI:N/S:C/C:L/I:H/A:N — Illustrative vector only`	
Affected Component	Funds-transfer service component
Affected Android Component	*Exported Service `com.novasphere.pay.service.TransferService` (`android:exported="true"`, no permission or caller-identity check)*

Description	`TransferService`, which initiates a fictional funds transfer on behalf of the currently authenticated NovaSphere Pay session, is declared with `android:exported="true"` in AndroidManifest.xml and performs no caller-identity, signature, or permission check before acting on the Intent it receives. Any other fictional application installed on the same device — with no special permissions declared — can start this service directly.
Business Context	NovaSphere Pay's core value proposition is peer-to-peer and bill-pay funds movement; any component capable of triggering a transfer without validating the calling app's identity undermines the app's entire trust model.
Technical Impact	A malicious fictional local application can send a crafted Intent directly to `TransferService`, bypassing NovaSphere Pay's own UI, confirmation screen, and any client-side transaction limits enforced only in the Activity layer.
Business Impact	A fictional malicious co-installed app could trigger an unauthorized transfer from a logged-in victim's NovaSphere Pay session without any user interaction or visible confirmation dialog.
Attack Preconditions	The fictional victim has NovaSphere Pay installed and a valid authenticated session; a second, attacker-controlled fictional app is installed on the same device (no special Android permissions required).
Safe Proof of Concept (Summary)	*A fictional proof-of-concept app started `TransferService` directly with a crafted Intent specifying a destination account and amount; the fictional transfer was accepted without any caller validation. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-003, EV-AND-004 (fictional manifest excerpt, adb Intent proof-of-concept log)*
Expected Behavior	Privileged, transaction-initiating components should not be exported unless strictly necessary; where export is required for legitimate integration, a `signature`-level custom permission and independent server-side confirmation should gate the action.
Observed (Fictional) Behavior	The component was exported with no permission gate and processed a crafted external Intent as a legitimate transfer request.
Root Cause	The service was exported to support a legacy fictional internal integration path and was never re-scoped or permission-gated as the transfer feature matured.
Recommendation	Set `android:exported="false"` unless external invocation is a genuine product requirement; if required, protect with a `signature`-level custom permission and require server-side step-up confirmation independent of the calling Intent.
Management Response	*NovaSphere engineering (fictional) has accepted this finding and will remove the export flag in the next hotfix release, with a compensating WAF-level fictional monitoring rule as an interim control.*
Owner / Priority / Target / Status	**Mobile Engineering Lead (Fictional Role)** | Immediate — 0-30 Days | 08 October 2026 | **Remediation In Progress**
Retest Recommendation	*Retest by repeating the fictional adb Intent proof-of-concept against the remediated build and confirming the service is no longer externally reachable.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-PLATFORM-1 — "The app uses IPC mechanisms securely."	MASWE-0062 — Insecure Services	MASTG-TEST-0029 — Testing for Sensitive Functionality Exposure Through IPC

FULL EVIDENCE PANEL — ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE

AndroidManifest.xml Excerpt (Fictional Build) (Fictional / Synthetic)

```
<service
    android:name="com.novasphere.pay.service.TransferService"
    android:exported="true" />
<!-- No android:permission attribute; no signature-level protection declared -->
```

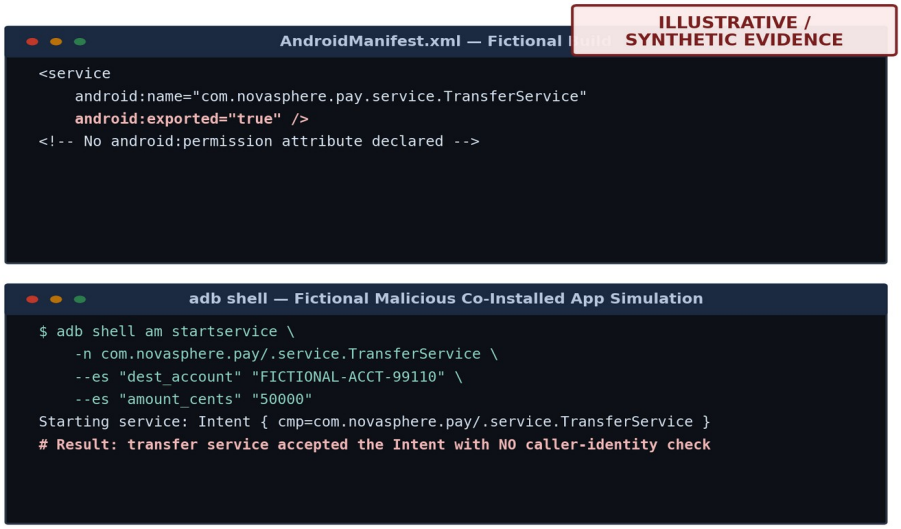
adb / Fictional Malicious-App Intent (Synthetic) (Fictional / Synthetic)

```
$ adb shell am startservice \
    -n com.novasphere.pay/.service.TransferService \
    --es "dest_account" "FICTIONAL-ACCT-99110" \
    --es "amount_cents" "50000"
Starting service: Intent { cmp=com.novasphere.pay/.service.TransferService }
```

Expected: The exported surface for transfer-initiating logic should require an explicit permission, be restricted to the app's own process (`exported="false"`), or independently re-verify caller identity and re-prompt for confirmation before acting.

Observed: The fictional Intent was accepted and processed by `TransferService` with no caller-identity check, no permission requirement, and no re-confirmation step.

Impact: Unauthorized, fictional funds-transfer initiation from any co-installed application without user awareness.



Synthetic mock-tool evidence — ILLUSTRATIVE / SYNTHETIC EVIDENCE. No real application, device, or data is depicted.

TECHNICAL EVIDENCE SUMMARY

Attack Preconditions	The fictional victim has NovaSphere Pay installed and a valid authenticated session; a second, attacker-controlled fictional app is installed on the same device (no special Android permissions required).
Observed Result	The component was exported with no permission gate and processed a crafted external Intent as a legitimate transfer request.
Security Impact	A malicious fictional local application can send a crafted Intent directly to `TransferService`, bypassing NovaSphere Pay's own UI, confirmation screen, and any client-side transaction limits enforced only in the Activity layer.
Business Impact	A fictional malicious co-installed app could trigger an unauthorized transfer from a logged-in victim's NovaSphere Pay session without any user interaction or visible confirmation dialog.
Evidence Collected	EV-AND-003, EV-AND-004 (fictional manifest excerpt, adb Intent proof-of-concept log)
Retest Validation	Retest by repeating the fictional adb Intent proof-of-concept against the remediated build and confirming the service is no longer externally reachable.

AND-003 — Broken Authorization in Mobile API Workflow	**CRITICAL**
CVSS-Style Score: 8.6 (Critical) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-639 — Authorization Bypass Through User-Controlled Key**
CVSS Vector (Illustrative): `CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:L/A:N — Illustrative vector only`	
Affected Component	Mobile-API account statement workflow
Affected Android Component	*N/A (server-side authorization gap surfaced via mobile client traffic) — `GET /v3/mobile/accounts/{accountId}/statements`*
Description	The mobile-only statement-retrieval endpoint invoked by NovaSphere Pay accepts a client-supplied `accountId` path parameter and returns the corresponding fictional account statement without verifying that the authenticated session's own account set includes the requested ID. The gap is specific to the mobile API surface; the equivalent web-channel endpoint (out of scope for this Android assessment) was not retested.
Business Context	NovaSphere Pay account statements contain fictional balance history, counterparty names, and transaction memos; cross-account exposure directly undermines customer data confidentiality.
Technical Impact	Object-level authorization is enforced client-side (the app only ever renders the authenticated user's own account IDs) but not re-verified server-side against the session's account ownership.
Business Impact	A fictional authenticated low-privilege user can enumerate sequential or guessable account identifiers and retrieve other customers' fictional statement data directly from the mobile API.
Attack Preconditions	A valid, authenticated NovaSphere Pay session and the ability to intercept/replay the app's own mobile-API traffic with a modified `accountId` value.
Safe Proof of Concept (Summary)	*A fictional proxied request substituting a foreign `accountId` into an authenticated mobile session returned that account's fictional statement data instead of an authorization error. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-005, EV-AND-006 (fictional intercepted mobile-API traffic capture, account-enumeration test matrix)*
Expected Behavior	The server should verify that the authenticated session's account-ownership set includes the requested `accountId` before returning any statement data, independent of the client-supplied value.
Observed (Fictional)	The mobile-API backend returned data for a fictional account the authenticated session did not own.

Behavior	
Root Cause	Missing server-side object-level authorization check on the mobile-specific statement endpoint; the equivalent check present on other mobile endpoints was not applied consistently to this route.
Recommendation	Add a server-side ownership check on every mobile-API route accepting a client-supplied account/object identifier; add automated cross-account regression tests to the mobile-API test suite; review all mobile-only endpoints for the same gap pattern.
Management Response	*NovaSphere engineering (fictional) has accepted this finding and will apply the missing ownership check platform-wide across mobile-API endpoints, validated by an expanded fictional regression suite.*
Owner / Priority / Target / Status	**API Platform Lead (Fictional Role)** | Immediate — 0-30 Days | 12 October 2026 | **Remediation In Progress**
Retest Recommendation	*Retest by repeating the fictional cross-account request against the remediated endpoint and confirming a 403 response, then sweep all mobile-API endpoints for the same pattern.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-AUTH-1 — "The app uses secure authentication and authorization protocols and follows the relevant best practices."	No single verified MASWE identifier maps precisely to this fictional scenario — assessed against the mapped MASVS control concept.	No single verified MASTG-TEST identifier maps precisely (server-side API authorization sits outside MASTG's on-device test scope); assessed against the MASVS-AUTH-1 control concept using standard mobile-API authorization testing technique.

FULL EVIDENCE PANEL — ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE

Intercepted Request (Fictional / Synthetic) (Fictional / Synthetic)

```
GET /v3/mobile/accounts/ACCT-771123/statements?period=2026-08 HTTP/1.1
Host: api.novasphere-example.com
Authorization: Bearer [FICTIONAL-MOBILE-TOKEN]
X-Test-Session-Account: ACCT-559001
User-Agent: NovaSpherePay-Android/6.4.2
```

Response (Fictional / Synthetic) (Fictional / Synthetic)

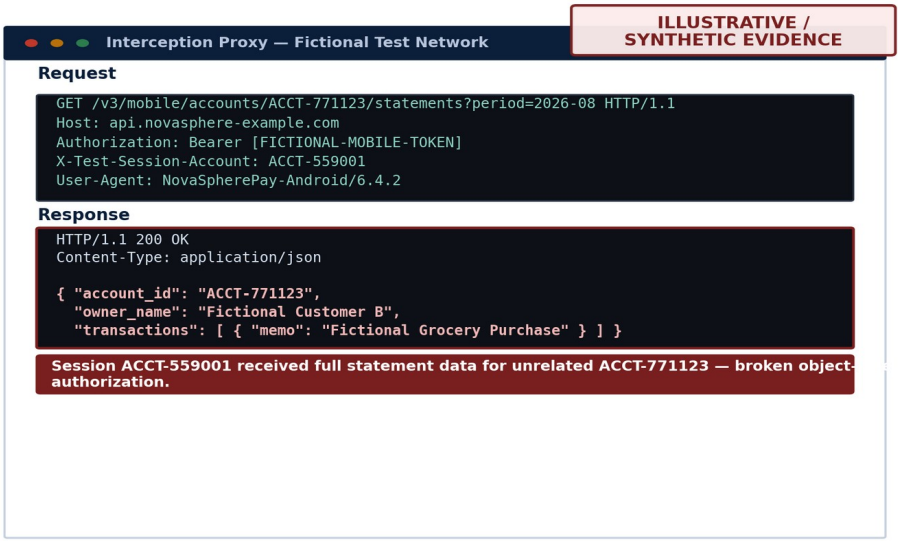
```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "account_id": "ACCT-771123",
  "owner_name": "Fictional Customer B",
  "period": "2026-08",
  "transactions": [ { "memo": "Fictional Grocery Purchase", "amount_cents": -6420 } ]
}
```

Expected: HTTP 403 Forbidden — the authenticated session (ACCT-559001) does not own ACCT-771123.

Observed: HTTP 200 OK with the full fictional statement object belonging to an unrelated account.

Impact: Cross-account disclosure of fictional statement and transaction-memo data via the mobile API surface.



Synthetic mock-tool evidence — ILLUSTRATIVE / SYNTHETIC EVIDENCE. No real application, device, or data is depicted.

TECHNICAL EVIDENCE SUMMARY

Attack Preconditions	A valid, authenticated NovaSphere Pay session and the ability to intercept/replay the app's own mobile-API traffic with a modified `accountId` value.
Observed Result	The mobile-API backend returned data for a fictional account the authenticated session did not own.
Security Impact	Object-level authorization is enforced client-side (the app only ever renders the authenticated user's own account IDs) but not re-verified server-side against the session's account ownership.
Business Impact	A fictional authenticated low-privilege user can enumerate sequential or guessable account identifiers and retrieve other customers' fictional statement data directly from the mobile API.
Evidence Collected	EV-AND-005, EV-AND-006 (fictional intercepted mobile-API traffic capture, account-enumeration test matrix)
Retest Validation	Retest by repeating the fictional cross-account request against the remediated endpoint and confirming a 403 response, then sweep all mobile-API endpoints for the same pattern.

AND-004 — Insecure Deep-Link Flow Enables Sensitive Account Action	**CRITICAL**
CVSS-Style Score: 8.3 (Critical) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-940 — Improper Verification of Source of a Communication Channel**
CVSS Vector (Illustrative): `CVSS:3.1/AV:L/AC:L/PR:N/UI:R/S:C/C:L/I:H/A:N — Illustrative vector only`	
Affected Component	Deep-link payee-addition flow
Affected Android Component	*Activity `com.novasphere.pay.deeplink.PayeeLinkActivity` (custom scheme `novasphere://` and App Link `https://pay.novasphere-example.com/link`)*
Description	The `novasphere://add-payee` deep link (and its equivalent HTTPS App Link) adds a new payee/beneficiary to the fictional authenticated user's account directly from the link parameters, without an in-app confirmation step or re-authentication, and without verifying that the link's referring source is trusted. A fictional malicious webpage or SMS message can silently trigger the flow if the app is already in an authenticated state.
Business Context	The payee-linking flow is a high-value target: a silently added payee is a common precursor to a fictional social-engineering funds-transfer scam.
Technical Impact	The `PayeeLinkActivity` intent-filter accepts the custom scheme and App Link, parses the `payee_id` and `payee_name` parameters, and commits the new payee association immediately in `onCreate()`, before any user-facing confirmation UI is shown.
Business Impact	A fictional attacker who can get a logged-in victim to tap a crafted link (e.g., via SMS or a compromised fictional webpage) can silently add an attacker-controlled payee, setting up a follow-on fictional social-engineering transfer request.
Attack Preconditions	The fictional victim has an active NovaSphere Pay session and taps a link (App Link or custom-scheme) delivered via any channel outside the app's control (SMS, email, web, QR code).
Safe Proof of Concept (Summary)	*A fictional crafted deep link opened while a test session was authenticated silently added a fictional attacker-controlled payee with no confirmation prompt. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-007, EV-AND-008 (fictional manifest intent-filter excerpt, deep-link trigger log)*
Expected Behavior	Sensitive account-modifying deep links should route to a confirmation screen requiring explicit user action, and ideally step-up authentication, before any state change is committed.
Observed (Fictional) Behavior	The fictional new-payee action was committed immediately upon link open with no confirmation or re-authentication.
Root Cause	The deep-link handler was implemented to prioritize a frictionless "tap-to-link" onboarding flow and omitted the confirmation step applied elsewhere in the app for equivalent in-app actions.
Recommendation	Insert a mandatory in-app confirmation screen (and step-up authentication for first-time payees) between deep-link receipt and any account-state change; treat all deep-link-originated actions as untrusted input requiring the same confirmation UX as their in-app equivalents.
Management Response	*NovaSphere engineering (fictional) has accepted this finding and will add a mandatory confirmation step to the deep-link payee flow, tracked alongside the other Critical fictional findings in the 0-30 day window.*
Owner / Priority / Target / Status	**Mobile Engineering Lead (Fictional Role)** | Immediate — 0-30 Days | 15 October 2026 | **Remediation In Progress**
Retest Recommendation	*Retest by repeating the fictional deep-link trigger against the remediated build and confirming a confirmation screen is required before the payee association is committed.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-PLATFORM-1 — "The app uses IPC mechanisms securely."	MASWE-0058 — Insecure Deep Links	No single verified MASTG-TEST identifier maps precisely — assessed conceptually against the mapped MASVS control using standard Android static/dynamic analysis technique.

FULL EVIDENCE PANEL — ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE

AndroidManifest.xml Intent Filter (Fictional Build) (Fictional / Synthetic)

```
<activity android:name=".deeplink.PayeeLinkActivity" android:exported="true">
  <intent-filter>
    <action android:name="android.intent.action.VIEW"/>
    <category android:name="android.intent.category.DEFAULT"/>
    <category android:name="android.intent.category.BROWSABLE"/>
    <data android:scheme="novasphere" android:host="add-payee"/>
    <data android:scheme="https" android:host="pay.novasphere-example.com" android:pathPrefix="/link"/>
  </intent-filter>
</activity>
```

Fictional Crafted Link + adb Trigger (Synthetic) (Fictional / Synthetic)

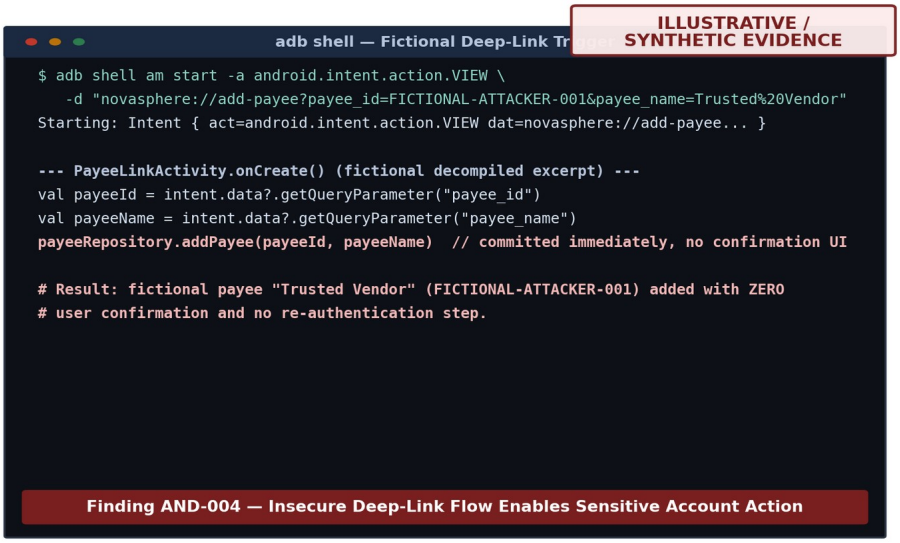
```
novasphere://add-payee?payee_id=FICTIONAL-ATTACKER-001&payee_name=Trusted%20Vendor

$ adb shell am start -a android.intent.action.VIEW \
   -d "novasphere://add-payee?payee_id=FICTIONAL-ATTACKER-001&payee_name=Trusted%20Vendor"
```

Expected: The deep link should open an in-app confirmation screen requiring explicit user approval (and, ideally, re-authentication) before committing a new payee association.

Observed: The fictional payee was committed to the account immediately on link open, with no confirmation screen and no re-authentication step.

Impact: Silent, unconfirmed modification of sensitive account state (payee list) via an untrusted external link source.



Synthetic mock-tool evidence — ILLUSTRATIVE / SYNTHETIC EVIDENCE. No real application, device, or data is depicted.

TECHNICAL EVIDENCE SUMMARY

Attack Preconditions	The fictional victim has an active NovaSphere Pay session and taps a link (App Link or custom-scheme) delivered via any channel outside the app's control (SMS, email, web, QR code).
Observed Result	The fictional new-payee action was committed immediately upon link open with no confirmation or re-authentication.
Security Impact	The `PayeeLinkActivity` intent-filter accepts the custom scheme and App Link, parses the `payee_id` and `payee_name` parameters, and commits the new payee association immediately in `onCreate()`, before any user-facing confirmation UI is shown.
Business Impact	A fictional attacker who can get a logged-in victim to tap a crafted link (e.g., via SMS or a compromised fictional webpage) can silently add an attacker-controlled payee, setting up a follow-on fictional social-engineering transfer request.
Evidence Collected	EV-AND-007, EV-AND-008 (fictional manifest intent-filter excerpt, deep-link trigger log)
Retest Validation	Retest by repeating the fictional deep-link trigger against the remediated build and confirming a confirmation screen is required before the payee association is committed.

95 HIGH FINDINGS

The following 9 fictional High-severity findings should be prioritized immediately after the Critical findings in Section 94.

AND-005 — WebView JavaScript Interface Exposure	HIGH
CVSS-Style Score: 7.9 (High) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-749 — Exposed Dangerous Method or Function**
CVSS Vector (Illustrative): `CVSS:3.1/AV:L/AC:L/PR:N/UI:R/S:C/C:H/I:L/A:N — Illustrative vector only`	
Affected Component	In-app support/help WebView
Affected Android Component	*`com.novasphere.pay.support.HelpWebViewActivity` — `addJavascriptInterface(new NativeBridge(), "NovaBridge")`*
Description	The in-app help/support WebView exposes a native `NovaBridge` JavaScript interface with methods including `getSessionToken()` and `getDeviceId()`, while `setJavaScriptEnabled(true)` and mixed-content loading remain enabled and the WebView is not strictly confined to the intended first-party support domain.
Business Context	The support WebView is reachable from an in-app "Help" entry point and, in the fictional build, can be influenced by any content the WebView is permitted to load, including third-party redirect targets embedded in help articles.
Technical Impact	Fictional JavaScript running in the WebView context can call `NovaBridge.getSessionToken()` directly, since the interface performs no origin check on the calling page before returning sensitive values.
Business Impact	If the WebView were coerced into loading fictional attacker-influenced content (e.g., via an open redirect in a support-article link), the exposed bridge could leak the live session token to that content.
Safe Proof of Concept (Summary)	*A fictional test HTML page loaded in the support WebView called `NovaBridge.getSessionToken()` from JavaScript and received the live fictional session token in the callback. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-009 (fictional WebView JS-interface enumeration and callback capture)*
Expected Behavior	JavaScript interfaces exposing sensitive data should not be attached to WebViews that load any non-strictly-controlled first-party content; where unavoidable, the interface should verify the calling origin before returning sensitive values, and mixed content / arbitrary navigation should be disabled.
Observed (Fictional) Behavior	The `NovaBridge` interface returned the live session token to any JavaScript executing in the WebView, with no origin verification.
Root Cause	The support WebView was extended with a native bridge for convenience features without re-scoping the WebView's navigation policy or adding origin checks to the bridge methods.
Recommendation	Remove sensitive methods from the JavaScript interface entirely, restrict WebView navigation to an allow-listed first-party domain set, disable mixed content, and add explicit origin verification to any remaining bridge methods.
Management Response	*Accepted; fictional mobile team will scope the bridge to non-sensitive methods only and lock WebView navigation to the support domain allow-list.*
Owner / Priority / Target / Status	**Mobile Engineering Lead (Fictional Role)** | 0-30 Days | 31 October 2026 | Remediation Planned
Retest Recommendation	*Retest by repeating the fictional JS-interface call against the remediated build and confirming sensitive methods are no longer reachable or origin-restricted.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-PLATFORM-2 — "The app uses WebViews securely."	MASWE-0071 — WebViews Loading Content from Untrusted Sources	MASTG-TEST-0033 — Testing for Java Objects Exposed Through WebViews

AND-006 — Certificate Validation Weakness	HIGH
CVSS-Style Score: 7.7 (High) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-295 — Improper Certificate Validation**
CVSS Vector (Illustrative): `CVSS:3.1/AV:A/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:N — Illustrative vector only`	
Affected Component	Legacy fictional network client (analytics/telemetry channel)
Affected Android Component	*`com.novasphere.pay.telemetry.TelemetryClient` — custom `X509TrustManager` implementation*
Description	The telemetry/analytics network client implements a custom `X509TrustManager` whose `checkServerTrusted()` method is a fictional no-op, effectively trusting any TLS certificate presented by the telemetry endpoint. The primary payments API client (a separate component) correctly enforces certificate pinning; this weaker client is limited to the telemetry channel.
Business Context	The telemetry channel carries fictional device, usage, and crash-diagnostic data; while lower-sensitivity than payment traffic, it is still a network trust-boundary weakness within the same application.
Technical Impact	A fictional on-path attacker (e.g., a rogue Wi-Fi access point) can present any certificate, including a self-signed one, and the telemetry client will accept it without validation.

Business Impact	In a fictional adversary-in-the-middle position, telemetry data could be intercepted or manipulated, and the weak trust-manager pattern represents a template risk that could be inadvertently reused for higher-sensitivity channels in future development.
Safe Proof of Concept (Summary)	*A fictional proxy interception using a self-signed certificate was accepted without warning by the telemetry client, confirming the trust manager performs no validation. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-010 (fictional TLS interception test log, decompiled trust-manager excerpt)*
Expected Behavior	All network clients — including auxiliary/telemetry channels — should use the platform default `TrustManager` or an equivalent implementation that performs full chain and hostname validation, supplemented by certificate/public-key pinning for developer-controlled endpoints.
Observed (Fictional) Behavior	The telemetry client's custom trust manager accepted an untrusted, self-signed fictional certificate without error.
Root Cause	A custom `TrustManager` was introduced during early fictional development to simplify local testing against a self-signed telemetry endpoint and was not reverted before release.
Recommendation	Remove the custom trust manager and rely on the platform default validation path for the telemetry client; extend certificate pinning to the telemetry endpoint consistent with the payments API client; add a static-analysis rule to flag custom `TrustManager`/`HostnameVerifier` implementations in future builds.
Management Response	*Accepted; fictional platform team will remove the custom trust manager and align the telemetry client with the standard validation path used elsewhere in the app.*
Owner / Priority / Target / Status	**Mobile Engineering Lead (Fictional Role)** | 0-30 Days | 31 October 2026 | **Remediation Planned**
Retest Recommendation	*Retest by repeating the fictional self-signed-certificate interception test against the remediated telemetry client and confirming the connection is rejected.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-NETWORK-2 — "The app performs identity pinning for all remote endpoints under the developer's control."	MASWE-0052 — Insecure Certificate Validation	MASTG-TEST-0021 — Testing Endpoint Identity Verification

AND-007 — Sensitive Data Exposure Through Logs	**HIGH**
CVSS-Style Score: 7.4 (High) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-532 — Insertion of Sensitive Information into Log File**
CVSS Vector (Illustrative): `CVSS:3.1/AV:L/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N — Illustrative vector only`	
Affected Component	Authentication and payment request/response logging
Affected Android Component	*`Log.d()` calls in `AuthRepository` and `PaymentApiClient` (debug build flavor; present but gated in release per Section 32)*
Description	Several fictional network-layer interceptors log full request and response bodies — including the bearer token, MFA one-time code, and card-linking payloads — at `Log.d()` level. While a `BuildConfig.DEBUG` gate suppresses this in the fictional production build variant, the logging calls remain compiled into the release binary in a form that is trivially re-enabled and were still observable via `adb logcat` on the fictional QA/staging build tested.
Business Context	Any fictional device with USB debugging enabled, or any pre-production build with the debug gate misconfigured, would leak highly sensitive authentication material directly to the system log, which is readable by other apps holding `READ_LOGS`-equivalent access on affected OS/OEM configurations.
Technical Impact	On the fictional staging build tested (debug gate intentionally left open for QA), `adb logcat` captured full bearer tokens and MFA codes in plaintext during a login flow.
Business Impact	Log-based leakage of authentication material substantially increases the fictional blast radius of any device-level compromise or debug-build misconfiguration.
Safe Proof of Concept (Summary)	*A fictional `adb logcat` capture during a test login on the QA-flavored build showed the bearer token and MFA code in plaintext log lines. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-011 (fictional logcat capture, annotated)*
Expected Behavior	No sensitive values (tokens, OTPs, card data, PII) should ever be passed to a logging call, even behind a debug gate; logging statements handling potentially sensitive payloads should be removed or use field-level redaction unconditionally.
Observed (Fictional) Behavior	Sensitive authentication values were present in log statements reachable via logcat under a QA build configuration.
Root Cause	Verbose request/response logging was added for fictional QA debugging convenience and relies solely on a build-flavor gate rather than redaction at the log call site.
Recommendation	Remove or redact sensitive fields at the logging call site regardless of build flavor; adopt a logging interceptor with an explicit sensitive-field allow-list/deny-list; add a CI lint rule that fails the build on logging calls referencing token/credential-named variables.
Management Response	*Accepted; fictional platform team will implement field-level redaction in the logging interceptor and add a CI lint gate.*

Owner / Priority / Target / Status	**Mobile Engineering Lead (Fictional Role)** | 0-30 Days | 31 October 2026 | Remediation Planned
Retest Recommendation	*Retest by repeating the fictional logcat capture against the remediated QA build and confirming sensitive fields are redacted regardless of build flavor.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-STORAGE-2 — "The app prevents leakage of sensitive data."	MASWE-0005 — Insertion of Sensitive Data into Logs	MASTG-TEST-0003 — Testing Logs for Sensitive Data

AND-008 — Hardcoded API Secret in Application Package	**HIGH**
CVSS-Style Score: 7.2 (High) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-798 — Use of Hard-coded Credentials**
CVSS Vector (Illustrative): `CVSS:3.1/AV:L/AC:L/PR:N/UI:N/S:U/C:H/I:L/A:N — Illustrative vector only`	
Affected Component	Third-party mapping/geolocation SDK initialization
Affected Android Component	*`res/values/strings.xml` — `map_sdk_api_key`; referenced from `NovaSphereApplication.onCreate()`*
Description	A fictional third-party mapping/geolocation SDK API key used for branch-locator functionality is hardcoded as a plaintext string resource, recoverable via straightforward APK decompilation with no reverse-engineering effort beyond unzipping and reading `strings.xml`.
Business Context	While this key is scoped to a lower-sensitivity mapping SDK rather than the core payments API, a leaked key can still be abused for fictional quota exhaustion or billing impact against NovaSphere's SDK account, and demonstrates a hardcoding pattern that increases risk if reused for a higher-sensitivity credential.
Technical Impact	The fictional key is visible in plaintext by decompiling the APK with a standard tool and reading the extracted `strings.xml` resource file.
Business Impact	Fictional quota exhaustion, unexpected billing impact, or service disruption if the extracted key is abused by a third party at scale.
Safe Proof of Concept (Summary)	*APK decompilation of the fictional release build recovered the mapping SDK API key directly from the extracted resource strings. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-012 (fictional decompiled strings.xml excerpt)*
Expected Behavior	Third-party SDK keys should be retrieved at runtime from a backend-issued, scoped token where the SDK supports it, or at minimum obfuscated/split so static extraction does not yield a directly usable key.
Observed (Fictional) Behavior	The fictional key was present in plaintext in the compiled resource strings.
Root Cause	The key was added directly to `strings.xml` during initial fictional SDK integration and never migrated to a server-issued or obfuscated pattern.
Recommendation	Move the key retrieval server-side (backend-issued, scoped/rotatable token) where the SDK supports it; otherwise apply key-splitting/obfuscation and restrict the key's allowed referrers/package signature at the SDK provider level; rotate the currently exposed fictional key.
Management Response	*Accepted; fictional platform team will rotate the exposed key and migrate to backend-issued scoped tokens for the mapping SDK.*
Owner / Priority / Target / Status	**Mobile Engineering Lead (Fictional Role)** | 0-30 Days | 31 October 2026 | Remediation Planned
Retest Recommendation	*Retest by repeating the fictional decompilation against the remediated build and confirming no directly usable key is present in static resources.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-STORAGE-1 — "The app securely stores sensitive data."	MASWE-0004 — Sensitive Data Hardcoded in the App Package	No single verified MASTG-TEST identifier maps precisely — assessed conceptually against the mapped MASVS control using standard Android static/dynamic analysis technique.

AND-009 — Insecure FileProvider Configuration	**HIGH**
CVSS-Style Score: 7.1 (High) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-927 — Use of Implicit Intent for Sensitive Communication**
CVSS Vector (Illustrative): `CVSS:3.1/AV:L/AC:L/PR:N/UI:R/S:U/C:H/I:N/A:N — Illustrative vector only`	
Affected Component	Statement-export / document-sharing feature
Affected Android Component	*`FileProvider` authority `com.novasphere.pay.fileprovider`, `file_paths.xml` root-path grant with*

	`android:grantUriPermissions="true"`
Description	The statement-export feature's `FileProvider` is configured with an overly broad `<root-path>` grant in `file_paths.xml`, exposing the app's entire private files directory (rather than a narrowly scoped export subdirectory) to any app the share Intent's URI permission is extended to, combined with `FLAG_GRANT_READ_URI_PERMISSION` applied without path narrowing.
Business Context	The intended feature is limited to sharing a single exported PDF statement; the overbroad path grant unintentionally exposes the full private-files tree to the receiving app for the life of the granted URI permission.
Technical Impact	A fictional receiving app (e.g., a file manager or a malicious share-target app) that receives the export Intent can traverse the granted authority beyond the intended single file, reaching other fictional private files such as cached statement PDFs from prior exports.
Business Impact	Unintended disclosure of fictional cached financial documents beyond the single file the user intended to share.
Safe Proof of Concept (Summary)	*A fictional test share-target app receiving the export Intent successfully requested a second, previously cached fictional statement file via the same granted FileProvider authority. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-013 (fictional file_paths.xml excerpt, cross-file access test log)*
Expected Behavior	`file_paths.xml` should scope grants to the narrowest possible subdirectory containing only the files intended for export, and URI permissions should be revoked immediately after the share operation completes.
Observed (Fictional) Behavior	The FileProvider configuration granted access to the broader private-files root, allowing retrieval of files beyond the single intended export.
Root Cause	The `file_paths.xml` root-path entry was configured broadly during initial feature development and never narrowed to the dedicated export subdirectory.
Recommendation	Scope `file_paths.xml` to a dedicated, minimal export subdirectory; explicitly call `revokeUriPermission()` after the share flow completes; add an automated test asserting only the intended file is reachable through the granted authority.
Management Response	*Accepted; fictional mobile team will narrow the FileProvider path configuration and add explicit permission revocation.*
Owner / Priority / Target / Status	**Mobile Engineering Lead (Fictional Role)** | 31-60 Days | 28 November 2026 | **Remediation Planned**
Retest Recommendation	*Retest by repeating the fictional cross-file access attempt against the remediated FileProvider configuration and confirming only the intended file is reachable.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-PLATFORM-1 — "The app uses IPC mechanisms securely."	MASWE-0064 — Insecure Content Providers	MASTG-TEST-0007 — Determining Whether Sensitive Stored Data Has Been Exposed via IPC Mechanisms

AND-010 — Weak Biometric/Local Authentication Enforcement	**HIGH**
CVSS-Style Score: 6.9 (High) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-305 — Authentication Bypass by Primary Weakness**
CVSS Vector (Illustrative): `CVSS:3.1/AV:P/AC:L/PR:N/UI:N/S:U/C:H/I:L/A:N — Illustrative vector only`	
Affected Component	Biometric app-unlock gate
Affected Android Component	*`com.novasphere.pay.auth.BiometricGateActivity` — `BiometricPrompt` result handling*
Description	NovaSphere Pay's biometric app-unlock gate uses `BiometricPrompt` for the UI/UX but validates the authentication outcome by checking a boolean flag set in the success callback rather than requiring a CryptoObject-bound Keystore key operation to succeed. On a fictional rooted test device, the boolean flag was directly manipulated to bypass the biometric prompt entirely.
Business Context	The biometric gate protects access to a returning user's already-authenticated session, guarding the app-open surface between full logins.
Technical Impact	Because the authentication result is a locally-checked boolean rather than a Keystore-bound cryptographic operation, the check can be bypassed on a fictional rooted or instrumented device using standard runtime-hooking techniques, without ever satisfying the actual biometric sensor.
Business Impact	On a fictional lost, stolen, or malware-compromised rooted device, the biometric app-unlock gate provides materially weaker protection for an already-authenticated session than intended.
Safe Proof of Concept (Summary)	*Using a fictional runtime-hooking framework on a rooted test device, the `onAuthenticationSucceeded` boolean flag was set directly, bypassing the biometric prompt. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-014 (fictional runtime-hooking session log, decompiled BiometricGateActivity excerpt)*
Expected Behavior	Local/biometric authentication gating sensitive session access should be bound to a CryptoObject backed by a Keystore key with `setUserAuthenticationRequired(true)`, so the operation cryptographically fails without genuine biometric authentication rather than relying on an app-level boolean.
Observed (Fictional) Behavior	The gate's pass/fail state was determined by an app-level boolean that could be set directly at runtime on a fictional rooted/instrumented device.
Root Cause	The biometric gate was implemented for UX purposes without binding the result to a Keystore cryptographic operation.

Recommendation	Bind the BiometricPrompt result to a CryptoObject wrapping a Keystore key with user-authentication-required semantics, so bypassing the UI check alone cannot satisfy the underlying cryptographic operation; add root/instrumentation-awareness signals as defense-in-depth per Section 87.
Management Response	*Accepted; fictional platform team will migrate the biometric gate to a CryptoObject-bound Keystore implementation.*
Owner / Priority / Target / Status	**Mobile Engineering Lead (Fictional Role)** | 0-30 Days | 31 October 2026 | Remediation Planned
Retest Recommendation	*Retest by repeating the fictional runtime-hooking bypass against the remediated CryptoObject-bound implementation and confirming the operation fails without genuine biometric authentication.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-AUTH-2 — "The app performs local authentication securely according to the platform best practices."	MASWE-0020 — Local Authentication Can Be Bypassed	MASTG-TEST-0018 — Testing Biometric Authentication

AND-011 — Token Lifecycle Weakness — Missing Refresh-Token Revocation on Logout	**HIGH**
CVSS-Style Score: 6.8 (High) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-613 — Insufficient Session Expiration**
CVSS Vector (Illustrative): `CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:N — Illustrative vector only`	
Affected Component	Logout / token-lifecycle handler
Affected Android Component	*`com.novasphere.pay.auth.SessionManager.logout()`*
Description	On fictional logout, `SessionManager.logout()` clears the locally stored tokens and returns the user to the login screen, but does not call the corresponding server-side token-revocation endpoint. A previously captured fictional refresh token (e.g., via AND-001's storage weakness, or a prior device backup) therefore remained valid and could be replayed to mint a new access token after the user believed they had logged out.
Business Context	Logout is a security-relevant user action; users reasonably expect it to fully invalidate their session server-side, not just clear local UI state.
Technical Impact	A fictional previously-captured refresh token continued to successfully mint new access tokens via the token endpoint for the remainder of its natural expiry window, regardless of the local logout action.
Business Impact	In a fictional scenario where a token was captured prior to logout (device compromise, shared/borrowed device, or the AND-001 storage weakness), logout does not close that exposure window.
Safe Proof of Concept (Summary)	*A fictional refresh token captured before logout was successfully replayed against the token endpoint after the app-level logout completed, minting a new fictional access token. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-015 (fictional pre/post-logout token-replay test log)*
Expected Behavior	Logout should call a server-side revocation endpoint for the current refresh token (and, ideally, all tokens issued to that device/session) before or alongside clearing local state.
Observed (Fictional) Behavior	The fictional refresh token remained valid and mintable after the app reported a successful logout.
Root Cause	The logout handler was implemented to clear local state only; server-side revocation was not wired into the fictional logout flow.
Recommendation	Call the token-revocation endpoint synchronously as part of logout before clearing local UI state; consider device/session-scoped revocation on explicit "log out of all devices" actions; add automated tests asserting post-logout token replay fails.
Management Response	*Accepted; fictional identity platform team will wire server-side revocation into the logout flow.*
Owner / Priority / Target / Status	**Identity & Access Lead (Fictional Role)** | 0-30 Days | 31 October 2026 | Remediation Planned
Retest Recommendation	*Retest by repeating the fictional pre/post-logout replay test against the remediated flow and confirming the refresh token is rejected immediately after logout.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-AUTH-1 — "The app uses secure authentication and authorization protocols and follows the relevant best practices."	MASWE-0038 — Authentication Tokens Not Validated	No single verified MASTG-TEST identifier maps precisely — assessed conceptually against the mapped MASVS control using standard Android static/dynamic analysis technique.

AND-012 — Improper Intent Handling Enables Intent Spoofing / Task Hijacking Risk	**HIGH**

CVSS-Style Score: 6.6 (High) — Illustrative CVSS v3.1-inspired score	CWE: CWE-926 — Improper Export of Android Application Components
CVSS Vector (Illustrative): `CVSS:3.1/AV:L/AC:H/PR:N/UI:R/S:U/C:L/I:H/A:N — Illustrative vector only`	
Affected Component	MFA-verification Activity task affinity/launch mode
Affected Android Component	*`com.novasphere.pay.auth.MfaVerifyActivity` — non-default `taskAffinity`, `launchMode="singleTask"`*
Description	`MfaVerifyActivity` declares a custom `taskAffinity` combined with `launchMode="singleTask"`, a configuration pattern associated with task-hijacking/StrandHogg-style attacks on affected Android versions, under which a fictional malicious co-installed app can insert itself into the task stack ahead of the legitimate Activity and present a spoofed MFA-entry screen.
Business Context	The MFA step is the second authentication factor protecting the fictional login flow; a spoofed MFA-entry screen could be used to phish the one-time code directly within the victim's own device context.
Technical Impact	The task-affinity/launch-mode combination allows a fictional malicious app, on affected OS versions, to hijack the task the legitimate MfaVerifyActivity would otherwise occupy, displaying attacker-controlled content that visually mimics the fictional MFA screen.
Business Impact	A fictional convincing on-device phishing overlay for the MFA code, delivered without any network indicator a user would typically check (e.g., URL bar), increasing the credibility of the attack to the victim.
Safe Proof of Concept (Summary)	*A fictional proof-of-concept app exploiting the task-affinity configuration successfully inserted a spoofed activity ahead of the legitimate MfaVerifyActivity in a controlled fictional test. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-016 (fictional task-affinity manifest excerpt, task-hijack proof-of-concept log)*
Expected Behavior	Activities handling sensitive input should use the default task affinity (matching the app's package) and avoid launch-mode/task-affinity combinations known to enable task hijacking on affected platform versions.
Observed (Fictional) Behavior	The custom task-affinity/launch-mode combination on MfaVerifyActivity matched a known task-hijacking-prone pattern in fictional testing.
Root Cause	The custom task affinity was introduced to support a now-unused fictional multi-window layout experiment and was never reverted.
Recommendation	Remove the custom `taskAffinity` from MfaVerifyActivity (or set it explicitly to the app's own package); review all Activities for the same pattern; add a static-analysis rule flagging non-default task affinities on sensitive Activities.
Management Response	*Accepted; fictional mobile team will remove the non-default task affinity from MfaVerifyActivity and audit remaining Activities.*
Owner / Priority / Target / Status	**Mobile Engineering Lead (Fictional Role)** | 31-60 Days | 28 November 2026 | **Remediation Planned**
Retest Recommendation	*Retest by repeating the fictional task-hijack proof-of-concept against the remediated build and confirming the spoofed activity can no longer insert ahead of the legitimate one.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-PLATFORM-1 — "The app uses IPC mechanisms securely."	MASWE-0057 — StrandHogg Attack / Task Affinity Vulnerability	MASTG-TEST-0029 — Testing for Sensitive Functionality Exposure Through IPC

AND-013 — Root / Runtime Integrity Control Bypass	HIGH
CVSS-Style Score: 6.4 (High) — Illustrative CVSS v3.1-inspired score	CWE: CWE-693 — Protection Mechanism Failure
CVSS Vector (Illustrative): `CVSS:3.1/AV:P/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:N — Illustrative vector only`	
Affected Component	Root-detection / runtime-integrity module
Affected Android Component	*`com.novasphere.pay.security.RootChecker` — client-side file-path and package-name heuristics only*
Description	NovaSphere Pay's root-detection module relies solely on client-side heuristics (checking for common su binary paths and known root-management package names) with the result gating a single boolean flag checked at app launch. No server-side attestation (e.g., Play Integrity API) is used, and the client-side check was bypassed in fictional testing using a standard root-hiding tool.
Business Context	Root/runtime-integrity signals are a defense-in-depth control expected for a FinTech application handling payment credentials and transaction authorization.
Technical Impact	The fictional heuristic check was bypassed by a commodity root-hiding module in testing, after which the app proceeded to full functionality as though running on an unrooted device.
Business Impact	Reduced confidence in the app's resilience posture on compromised devices, compounding the impact of storage- and authentication-related findings elsewhere in this fictional assessment (e.g., AND-001, AND-010).
Safe Proof of Concept (Summary)	*A fictional root-hiding module installed on a rooted test device caused RootChecker's heuristic checks to report a clean (non-rooted) result, and the app proceeded normally. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-017 (fictional root-detection bypass test log)*
Expected Behavior	Root/runtime-integrity signals should be supplemented by a platform attestation service (e.g., the Play Integrity API)

	rather than relying solely on client-side heuristics that are well-documented to be bypassable, and should inform risk-based decisions (e.g., step-up authentication) rather than a single binary gate.
Observed (Fictional) Behavior	The fictional client-side-only heuristic check was defeated by a commodity root-hiding tool, and no compensating server-side signal was present.
Root Cause	The root-detection module was implemented using client-side heuristics only, with no integration to a platform attestation service.
Recommendation	Integrate the Play Integrity API (or equivalent current platform attestation) as the primary integrity signal, with the existing heuristic checks retained only as a defense-in-depth supplement; use the attestation result to drive risk-based controls (e.g., transaction-limit reduction, step-up authentication) rather than a single pass/fail gate.
Management Response	*Accepted; fictional platform team will integrate Play Integrity API attestation as the primary resilience signal, tracked under the broader resilience roadmap in Section 102.*
Owner / Priority / Target / Status	**Mobile Engineering Lead (Fictional Role)** | 31-60 Days | 28 November 2026 | **Remediation Planned**
Retest Recommendation	*Retest by repeating the fictional root-hiding bypass against the remediated build and confirming the Play Integrity signal correctly reflects device integrity state.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-RESILIENCE-1 — "The app validates the integrity of the platform." / MASVS-RESILIENCE-2 — "The app implements anti-tampering mechanisms."	MASWE-0097 — Root/Jailbreak Detection Not Implemented	MASTG-TEST-0045 — Testing Root Detection

96 MEDIUM FINDINGS

The following 14 fictional Medium-severity findings reflect defense-in-depth and platform-hardening gaps recommended for remediation within the 31-60 day window of the illustrative roadmap (Section 102).

AND-014 — Debug Information Exposure in Pre-Production Build Configuration	MEDIUM
CVSS-Style Score: 5.4 (Medium) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-489 — Active Debug Code**
Affected Component	Build/signing configuration (QA build flavor)
Affected Android Component	*`android:debuggable` gated by flavor; verbose fictional debug menu reachable via a hidden tap sequence in the QA flavor*
Description	The fictional QA-flavored build exposes a hidden debug menu (accessible via a tap sequence on the version-number label) revealing environment endpoints, feature-flag state, and a "force token expiry" toggle. While not present in the fictional production flavor, the debug menu code ships inside the same APK and is reachable if the flavor gate is misconfigured.
Business Context	Debug tooling shipped inside a distributable APK increases risk if a QA-flavored or mis-flagged build were ever distributed outside controlled test channels.
Technical Impact	In the fictional QA build tested, the debug menu was reachable and disclosed internal environment hostnames not intended for general exposure.
Business Impact	Low direct fictional customer impact given the flavor gating, but represents a process risk if build-flavor controls are misconfigured for a public release.
Safe Proof of Concept (Summary)	*The fictional hidden tap sequence on the QA build's version label opened a debug menu disclosing internal environment endpoints. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-018 (fictional debug-menu screen capture)*
Expected Behavior	Debug-only tooling should be compiled out of any build variant that could plausibly reach a distribution channel, not solely gated by a runtime flag.
Observed (Fictional) Behavior	The debug menu code and its UI trigger were present in the QA flavor and reachable via manual interaction.
Root Cause	The debug menu was implemented as a runtime-gated feature rather than a compile-time-excluded build variant.
Recommendation	Move debug-menu code behind a compile-time flavor exclusion (source-set separation) rather than a runtime flag; add a release-build CI check asserting the debug menu class is absent from the production APK.
Management Response	*Accepted; fictional mobile team will move the debug menu to a QA-only source set.*
Owner / Priority / Target / Status	**Mobile Engineering Lead (Fictional Role)** | 31-60 Days | 28 November 2026 | **Remediation Planned**
Retest Recommendation	*Retest by confirming the debug-menu class is absent from a decompiled production-flavor APK.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-RESILIENCE-4 — "The app implements anti-dynamic analysis techniques."	No single verified MASWE identifier maps precisely to this fictional scenario — assessed against the mapped MASVS control concept.	MASTG-TEST-0046 — Testing Anti-Debugging Detection

AND-015 — Insecure Backup Configuration (allowBackup Enabled)	MEDIUM
CVSS-Style Score: 5.3 (Medium) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-530 — Exposure of Backup File to an Unauthorized Control Sphere**
Affected Component	Application manifest backup configuration
Affected Android Component	*`AndroidManifest.xml` — `android:allowBackup="true"` with a permissive fictional `backup_rules.xml`*
Description	`android:allowBackup="true"` is set with a fictional `backup_rules.xml` that does not exclude the SharedPreferences file identified in AND-001, meaning an `adb backup`-based extraction (on OS versions/configurations where this remains feasible) would capture the same sensitive token material.
Business Context	Backup exclusion is a compensating control that limits the blast radius of storage-related findings such as AND-001; its absence compounds that finding's impact.
Technical Impact	A fictional `adb backup` of the test build included the session-token SharedPreferences file in the resulting backup archive.
Business Impact	Increases the number of extraction paths through which the AND-001 token-storage weakness could be exploited.
Safe Proof of Concept	*A fictional `adb backup` invocation against the test build produced an archive containing the unencrypted session-token file.*

(Summary)	*[ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-019 (fictional adb backup archive listing)*
Expected Behavior	`backup_rules.xml` (or the Auto Backup exclusion mechanism) should explicitly exclude any file containing session tokens or other sensitive material, independent of the underlying storage-encryption fix tracked under AND-001.
Observed (Fictional) Behavior	The token-bearing SharedPreferences file was included in the fictional backup archive.
Root Cause	Backup exclusion rules were not updated when the token-storage file was introduced.
Recommendation	Add explicit exclusion rules for all token/credential-bearing files in `backup_rules.xml`; consider `android:allowBackup="false"` for the FinTech-sensitive data classes if broader OS-level backup is not a product requirement.
Management Response	*Accepted; fictional mobile team will update backup exclusion rules alongside the AND-001 remediation.*
Owner / Priority / Target / Status	**Mobile Engineering Lead (Fictional Role)** | 31-60 Days | 28 November 2026 | Remediation Planned
Retest Recommendation	*Retest by repeating the fictional adb backup extraction against the remediated build and confirming sensitive files are excluded.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-STORAGE-1 — "The app securely stores sensitive data."	MASWE-0006 — Sensitive Data Not Excluded From Backup	MASTG-TEST-0009 — Testing Backups for Sensitive Data

AND-016 — Sensitive Data Exposed via Screenshot / Recent-Apps Thumbnail	**MEDIUM**
CVSS-Style Score: 5.1 (Medium) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-200 — Exposure of Sensitive Information to an Unauthorized Actor**
Affected Component	Account balance and card-detail screens
Affected Android Component	*`AccountOverviewActivity`, `CardDetailActivity` — `FLAG_SECURE` not set*
Description	Screens displaying the fictional account balance and full (masked-in-UI but still sensitive) card details do not set `WindowManager.LayoutParams.FLAG_SECURE`, so the Recent Apps thumbnail and any OS-level screenshot capture the sensitive content in plaintext.
Business Context	A visible balance/card thumbnail in the Recent Apps switcher is a common "shoulder surf" and shared-device exposure vector for financial apps.
Technical Impact	In fictional testing, switching away from `AccountOverviewActivity` produced a Recent Apps thumbnail clearly showing the fictional account balance.
Business Impact	Casual, low-effort exposure of fictional sensitive account data to anyone with physical proximity to the unlocked device.
Safe Proof of Concept (Summary)	*A fictional screen-recording of the app-switcher gesture showed the account balance rendered in the Recent Apps thumbnail. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-020 (fictional recent-apps thumbnail capture)*
Expected Behavior	Screens displaying account balances, card details, or other sensitive financial data should set `FLAG_SECURE` to blank the thumbnail and block screenshot/screen-recording capture.
Observed (Fictional) Behavior	The fictional thumbnail rendered full sensitive content with no blanking.
Root Cause	`FLAG_SECURE` was not applied to the relevant Activities during development.
Recommendation	Apply `FLAG_SECURE` to all Activities displaying account balances, card details, or transaction history; add a UI-test assertion verifying the flag is set on these screens.
Management Response	*Accepted; fictional mobile team will apply FLAG_SECURE to the identified screens.*
Owner / Priority / Target / Status	**Mobile Engineering Lead (Fictional Role)** | 31-60 Days | 28 November 2026 | Remediation Planned
Retest Recommendation	*Retest by repeating the fictional app-switch capture against the remediated build and confirming the thumbnail is blanked.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-PLATFORM-3 — "The app uses the user interface securely."	No single verified MASWE identifier maps precisely to this fictional scenario — assessed against the mapped MASVS control concept.	No single verified MASTG-TEST identifier maps precisely — assessed conceptually against the mapped MASVS control using standard Android static/dynamic analysis technique.

AND-017 — Sensitive Data Exposure via Clipboard	**MEDIUM**

CVSS-Style Score: 4.9 (Medium) — Illustrative CVSS v3.1-inspired score	CWE: CWE-200 — Exposure of Sensitive Information to an Unauthorized Actor
Affected Component	Account-number "copy" convenience action
Affected Android Component	*`AccountDetailFragment` — `ClipboardManager.setPrimaryClip()`*
Description	The "copy account number" convenience action places the fictional full account number on the system clipboard without setting the `ClipDescription` sensitive-content flag (or restricting clipboard visibility on OS versions that support it), leaving it visible to other fictional apps and to the system clipboard-history UI on supporting OS versions.
Business Context	Clipboard content is a well-known cross-app leakage vector, particularly on OS versions/OEM skins exposing clipboard history.
Technical Impact	In fictional testing, the copied account number remained readable from a second test app polling the clipboard shortly after the copy action.
Business Impact	Low-friction exposure of fictional account-number data to any other app running on the device at the time of the copy action.
Safe Proof of Concept (Summary)	*A fictional second test app polling `ClipboardManager` recovered the copied account number immediately after the in-app copy action. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-021 (fictional cross-app clipboard read test log)*
Expected Behavior	Clipboard writes of sensitive values should set the sensitive-content `ClipDescription` extra (where the platform version supports it) and the value should be cleared from the clipboard automatically after a short interval.
Observed (Fictional) Behavior	The fictional clipboard entry was readable by another app with no sensitivity marking and no automatic clearing.
Root Cause	The copy action used the default `ClipData` construction path without the sensitive-content marking or an expiry mechanism.
Recommendation	Mark clipboard writes of sensitive values as sensitive content on supporting platform versions; implement an automatic clipboard-clear timer for copied account/card data.
Management Response	*Accepted; fictional mobile team will add sensitive-content marking and an auto-clear timer to the copy action.*
Owner / Priority / Target / Status	**Mobile Engineering Lead (Fictional Role)** | 31-60 Days | 28 November 2026 | **Remediation Planned**
Retest Recommendation	*Retest by repeating the fictional cross-app clipboard read against the remediated build and confirming the value is unreadable or auto-cleared.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-PLATFORM-3 — "The app uses the user interface securely."	No single verified MASWE identifier maps precisely to this fictional scenario — assessed against the mapped MASVS control concept.	No single verified MASTG-TEST identifier maps precisely — assessed conceptually against the mapped MASVS control using standard Android static/dynamic analysis technique.

AND-018 — Weak Cache Handling of Sensitive Response Data	MEDIUM
CVSS-Style Score: 4.8 (Medium) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-524 — Use of Cache Containing Sensitive Information**
Affected Component	Network response disk cache
Affected Android Component	*OkHttp disk cache (`cacheDir/http_cache`) — statement/transaction responses cached without `Cache-Control: no-store`*
Description	Statement and transaction-history API responses are cached to disk by the fictional HTTP client's default cache policy without a `Cache-Control: no-store` directive, leaving cleartext fictional transaction data readable from the app's private cache directory independent of the in-memory session state.
Business Context	Cached financial transaction data persists on disk beyond the active session, extending the fictional exposure window relative to in-memory-only handling.
Technical Impact	Fictional cached response bodies containing transaction data were recoverable from the app's private cache directory after the app was backgrounded.
Business Impact	Extends the fictional exposure window for transaction data beyond the active session to the lifetime of the cache entry.
Safe Proof of Concept (Summary)	*A fictional inspection of the app's private cache directory after a statement view recovered the cached, unencrypted response body. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-022 (fictional cache-directory listing and file excerpt)*
Expected Behavior	Responses containing sensitive financial data should be served with `Cache-Control: no-store` (or an equivalent client-side cache exclusion) so they are never persisted to disk.
Observed (Fictional) Behavior	The fictional sensitive response bodies were present in the on-disk HTTP cache.
Root Cause	The relevant API responses did not set a no-store cache directive, and the client honored the default cache policy.
Recommendation	Add `Cache-Control: no-store` to all sensitive financial-data API responses; configure the client-side HTTP cache to respect

	this directive and periodically purge the cache on logout.
Management Response	*Accepted; fictional platform and mobile teams will coordinate on the no-store header rollout.*
Owner / Priority / Target / Status	**Mobile Engineering Lead (Fictional Role)** | 31-60 Days | 28 November 2026 | Remediation Planned
Retest Recommendation	*Retest by repeating the fictional cache-directory inspection against the remediated build and confirming no sensitive response data is persisted.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-STORAGE-1 — "The app securely stores sensitive data."	MASWE-0001 — Sensitive Data Stored Unencrypted in Private Storage	MASTG-TEST-0001 — Testing Local Storage for Sensitive Data

AND-019 — Deprecated TLS Configuration Permitted by Network Security Configuration	**MEDIUM**
CVSS-Style Score: 4.7 (Medium) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-327 — Use of a Broken or Risky Cryptographic Algorithm**
Affected Component	Network Security Configuration
Affected Android Component	*`res/xml/network_security_config.xml`*
Description	The Network Security Configuration does not explicitly restrict the minimum negotiated TLS protocol version, and fictional testing confirmed the app's HTTP client would still complete a TLS 1.1 handshake against a fictional test endpoint configured to offer only legacy protocol versions, rather than refusing the connection outright.
Business Context	Consistent enforcement of current TLS minimums across every network client in the app reduces the risk of a downgrade path being exploitable against any one component.
Technical Impact	A fictional test server configured to offer only TLS 1.1 successfully completed a handshake with the app's network client.
Business Impact	In a fictional scenario involving a compromised or misconfigured intermediary, a legacy-protocol downgrade path remains technically available rather than being categorically refused.
Safe Proof of Concept (Summary)	*A fictional TLS-version-restricted test endpoint successfully completed a handshake with the app at TLS 1.1. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-023 (fictional TLS handshake test log)*
Expected Behavior	The Network Security Configuration (or equivalent client-level TLS configuration) should explicitly pin the minimum accepted protocol version to TLS 1.2 or higher for all configured domains.
Observed (Fictional) Behavior	The fictional client accepted a TLS 1.1 handshake rather than refusing the connection.
Root Cause	No explicit minimum-TLS-version constraint was set in the Network Security Configuration; the client relied on platform defaults, which in the fictional test scenario still permitted the legacy protocol.
Recommendation	Explicitly set the minimum accepted TLS protocol version to 1.2+ in the Network Security Configuration and/or the HTTP client's `ConnectionSpec`; add a regression test asserting legacy-protocol handshakes are refused.
Management Response	*Accepted; fictional platform team will pin the minimum TLS version explicitly.*
Owner / Priority / Target / Status	**Mobile Engineering Lead (Fictional Role)** | 31-60 Days | 28 November 2026 | Remediation Planned
Retest Recommendation	*Retest by repeating the fictional legacy-TLS handshake attempt against the remediated configuration and confirming the connection is refused.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-NETWORK-1 — "The app secures all network traffic according to the current best practices."	No single verified MASWE identifier maps precisely to this fictional scenario — assessed against the mapped MASVS control concept.	MASTG-TEST-0021 — Testing Endpoint Identity Verification

AND-020 — Excessive Android Permissions Requested	**MEDIUM**
CVSS-Style Score: 4.6 (Medium) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-250 — Execution with Unnecessary Privileges**
Affected Component	Application manifest permission declarations
Affected Android Component	*`AndroidManifest.xml` — `READ_CONTACTS`, `READ_PHONE_STATE`, `ACCESS_FINE_LOCATION` (always-requested, not feature-gated)*
Description	NovaSphere Pay requests `READ_CONTACTS`, `READ_PHONE_STATE`, and `ACCESS_FINE_LOCATION` unconditionally at

	first run, though fictional feature mapping shows contacts access is only needed for the optional "pay a friend" feature, phone-state is unused by any current fictional feature, and fine location is only needed for the optional branch-locator feature.
Business Context	Broad, upfront permission requests increase user distrust and expand the app's privacy footprint beyond what current fictional functionality requires.
Technical Impact	Static manifest review found no fictional code path referencing `READ_PHONE_STATE`, indicating the permission is unused entirely; the other two are used only by optional, non-default features.
Business Impact	Unnecessary privacy footprint and potential fictional app-store/compliance friction from over-broad permission requests relative to actual feature usage.
Safe Proof of Concept (Summary)	*Static review of the fictional manifest and codebase found one requested permission with no referencing code path and two others usable only via optional, non-default features requested unconditionally at first run. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-024 (fictional permission-to-feature mapping matrix)*
Expected Behavior	Permissions should be requested contextually, at the point of use for the specific optional feature, and any permission with no referencing code path should be removed entirely.
Observed (Fictional) Behavior	All three permissions were requested unconditionally at first run regardless of whether the associated optional feature was ever used.
Root Cause	Permissions were declared broadly during initial fictional feature development and never revisited as features became optional or were removed.
Recommendation	Remove `READ_PHONE_STATE` entirely; move the `READ_CONTACTS` and `ACCESS_FINE_LOCATION` requests to just-in-time prompts triggered by their respective optional features.
Management Response	*Accepted; fictional mobile team will move to contextual permission requests in the next feature release.*
Owner / Priority / Target / Status	**Mobile Engineering Lead (Fictional Role)** | 31-60 Days | 28 November 2026 | **Remediation Planned**
Retest Recommendation	*Retest by confirming the unused permission is removed and the remaining two are requested only contextually.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-PRIVACY-1 — "The app minimizes access to sensitive data and resources."	No single verified MASWE identifier maps precisely to this fictional scenario — assessed against the mapped MASVS control concept.	MASTG-TEST-0024 — Testing for App Permissions

AND-021 — Verbose Error Handling Reveals Internal Implementation Details	**MEDIUM**
CVSS-Style Score: 4.4 (Medium) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-209 — Generation of Error Message Containing Sensitive Information**
Affected Component	Mobile-API error-response handling
Affected Android Component	*N/A (client renders server-supplied error payload verbatim)*
Description	Several fictional mobile-API error responses (observed during malformed-request testing) include a full stack trace and internal service/class names in the JSON error payload, which the app renders directly in a fictional generic error dialog without sanitization.
Business Context	Verbose error output assists a fictional attacker in mapping backend architecture and technology choices during reconnaissance.
Technical Impact	A fictional malformed request to a statement endpoint returned a stack trace referencing internal fictional service and package names in the response body.
Business Impact	Incrementally aids fictional attacker reconnaissance of backend architecture; not independently exploitable but compounds the value of other findings.
Safe Proof of Concept (Summary)	*A fictional malformed request produced a JSON error response containing a full stack trace and internal service names, rendered verbatim by the client. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-025 (fictional error-response capture)*
Expected Behavior	API error responses returned to the mobile client should be sanitized, generic, user-facing messages; detailed diagnostic information should be logged server-side only.
Observed (Fictional) Behavior	The fictional error response included a full stack trace and internal implementation details, rendered directly to the user.
Root Cause	The affected fictional endpoints return the default framework exception payload rather than a sanitized API-level error contract.
Recommendation	Standardize all mobile-API error responses on a sanitized error contract (error code + user-facing message only); log full diagnostic detail server-side; add a contract test asserting no stack-trace content reaches client-facing responses.
Management Response	*Accepted; fictional API platform team will standardize the error-response contract.*

Owner / Priority / Target / Status	**API Platform Lead (Fictional Role)** | 31-60 Days | 28 November 2026 | Remediation Planned
Retest Recommendation	*Retest by repeating the fictional malformed-request test against the remediated endpoints and confirming only sanitized error content is returned.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-CODE-4 — "The app validates and sanitizes all untrusted inputs."	No single verified MASWE identifier maps precisely to this fictional scenario — assessed against the mapped MASVS control concept.	No single verified MASTG-TEST identifier maps precisely — assessed conceptually against the mapped MASVS control using standard Android static/dynamic analysis technique.

AND-022 — Weak Session Timeout Enforcement	**MEDIUM**
CVSS-Style Score: 4.3 (Medium) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-613 — Insufficient Session Expiration**
Affected Component	Client-side idle-session timer
Affected Android Component	*`com.novasphere.pay.auth.SessionManager` — 60-minute idle timeout, not independently enforced server-side*
Description	The fictional 60-minute idle-session timeout is enforced only client-side by an app-level timer; the underlying access token's own server-side expiry is materially longer, so a fictional captured token remains usable well beyond the point the app's own UI would have required re-authentication.
Business Context	A client-side-only timeout provides a UX-level session boundary but not a genuine security boundary if the underlying token is not independently time-limited to match.
Technical Impact	In fictional testing, an access token captured before the 60-minute client-side timeout elapsed remained valid against the API well beyond that window.
Business Impact	Reduces the practical effectiveness of the app's session-timeout UX as a security control for a fictional captured-token scenario.
Safe Proof of Concept (Summary)	*A fictional captured access token remained valid against the API after the client-side 60-minute idle timeout had elapsed and the app had returned to the login screen. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-026 (fictional token-validity-after-timeout test log)*
Expected Behavior	The server-side access-token expiry should be aligned with (or shorter than) the intended idle-session policy, so a client-side timeout is reinforced by an equivalent server-side boundary.
Observed (Fictional) Behavior	The fictional server-side token expiry materially exceeded the client-side idle-timeout window.
Root Cause	Client-side and server-side timeout values were configured independently and have drifted out of alignment.
Recommendation	Align server-side access-token expiry with the intended idle-session policy; consider a shorter-lived access token paired with silent refresh, so the effective session boundary is consistently enforced server-side.
Management Response	*Accepted; fictional identity platform team will align token expiry with the client-side idle policy.*
Owner / Priority / Target / Status	**Identity & Access Lead (Fictional Role)** | 31-60 Days | 28 November 2026 | Remediation Planned
Retest Recommendation	*Retest by repeating the fictional post-timeout token-validity test against the remediated expiry configuration.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-AUTH-1 — "The app uses secure authentication and authorization protocols and follows the relevant best practices."	No single verified MASWE identifier maps precisely to this fictional scenario — assessed against the mapped MASVS control concept.	No single verified MASTG-TEST identifier maps precisely — assessed conceptually against the mapped MASVS control using standard Android static/dynamic analysis technique.

AND-023 — Analytics/Tracking SDK Privacy Weakness — Persistent Identifier Reuse	**MEDIUM**
CVSS-Style Score: 4.1 (Medium) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-359 — Exposure of Private Personal Information to an Unauthorized Actor**
Affected Component	Third-party analytics SDK initialization
Affected Android Component	*`com.novasphere.pay.analytics.AnalyticsInitializer` — device-scoped persistent identifier used in place of the resettable Advertising ID*
Description	The bundled fictional analytics SDK is initialized with a long-lived, app-generated device identifier persisted independently of the platform's user-resettable Advertising ID, meaning a fictional user who resets their Advertising ID for privacy

	reasons continues to be tracked under the same underlying identifier by the analytics vendor.
Business Context	Users who reset their Advertising ID have signaled an explicit privacy preference; continuing to track them under an equivalent persistent identifier works against that intent.
Technical Impact	Static and dynamic review confirmed the analytics events continued to carry the same device-scoped identifier fictional-before and fictional-after an Advertising ID reset.
Business Impact	Undermines user-facing privacy controls and creates fictional regulatory/compliance exposure in jurisdictions that treat persistent-identifier tracking similarly to advertising-ID tracking.
Safe Proof of Concept (Summary)	*A fictional before/after comparison of analytics event payloads across an Advertising ID reset showed the same underlying device identifier persisted in both. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-027 (fictional analytics payload before/after comparison)*
Expected Behavior	Analytics/tracking identifiers should respect the same user reset signals as the platform Advertising ID, or should not be used for tracking purposes that the Advertising ID reset is intended to govern.
Observed (Fictional) Behavior	The fictional persistent identifier was unaffected by the Advertising ID reset and continued to link the same analytics history.
Root Cause	The analytics SDK was configured to generate and persist its own device identifier rather than keying off the resettable Advertising ID.
Recommendation	Reconfigure the analytics SDK to use the resettable Advertising ID (or an equivalent reset-aware identifier) as the tracking key, or clear the persistent identifier whenever the Advertising ID reset is detected.
Management Response	*Accepted; fictional data/privacy team will reconfigure the analytics SDK identifier strategy.*
Owner / Priority / Target / Status	**Privacy & Data Governance Lead (Fictional Role)** | 31-60 Days | 28 November 2026 | **Remediation Planned**
Retest Recommendation	*Retest by repeating the fictional before/after Advertising ID reset comparison against the remediated configuration.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-PRIVACY-2 — "The app prevents identification of the user."	MASWE-0110 — Use of Unique Identifiers for User Tracking	No single verified MASTG-TEST identifier maps precisely — assessed conceptually against the mapped MASVS control using standard Android static/dynamic analysis technique.

AND-024 — Insecure Random Number Generation for a Security-Relevant Value	**MEDIUM**
CVSS-Style Score: 4.0 (Medium) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-330 — Use of Insufficiently Random Values**
Affected Component	Client-side request-nonce generation
Affected Android Component	*`com.novasphere.pay.net.NonceGenerator` — `java.util.Random` seeded with `System.currentTimeMillis()`*
Description	The client-side request nonce used for fictional replay-protection on a subset of mobile-API calls is generated with `java.util.Random` seeded from the current timestamp, rather than a cryptographically secure random source, making the nonce sequence predictable to a fictional attacker with approximate timing knowledge.
Business Context	Predictable nonces weaken the replay-protection guarantee the nonce is intended to provide, though the affected endpoints have additional server-side validation reducing standalone exploitability.
Technical Impact	Fictional analysis showed the nonce sequence was reproducible given an approximate generation timestamp, consistent with the known weaknesses of `java.util.Random`.
Business Impact	Reduces the strength of the replay-protection control as a standalone defense, relying more heavily on other server-side controls to prevent replay abuse.
Safe Proof of Concept (Summary)	*Fictional static and dynamic analysis reproduced the nonce sequence from an approximate generation timestamp, confirming predictability. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-028 (fictional nonce predictability analysis)*
Expected Behavior	Security-relevant random values (nonces, tokens, key material) should be generated using `java.security.SecureRandom` rather than `java.util.Random`.
Observed (Fictional) Behavior	The fictional nonce generator used a non-cryptographic random source with a predictable seed.
Root Cause	`java.util.Random` was used for convenience during initial implementation without a security review of the nonce's intended guarantee.
Recommendation	Replace `java.util.Random` with `SecureRandom` for all security-relevant value generation; add a static-analysis rule flagging `java.util.Random` usage in security-sensitive packages.
Management Response	*Accepted; fictional mobile team will replace the random source with SecureRandom.*
Owner / Priority / Target / Status	**Mobile Engineering Lead (Fictional Role)** | 31-60 Days | 28 November 2026 | **Remediation Planned**

Retest Recommendation	*Retest by repeating the fictional predictability analysis against the remediated nonce generator.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-CRYPTO-1 — "The app employs current strong cryptography and uses it according to industry best practices."	MASWE-0019 — Risky Cryptography Implementations	No single verified MASTG-TEST identifier maps precisely — assessed conceptually against the mapped MASVS control using standard Android static/dynamic analysis technique.

AND-025 — Tapjacking / Overlay Protection Not Implemented on Sensitive Confirmation Screens	**MEDIUM**
CVSS-Style Score: 3.9 (Medium) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-1021 — Improper Restriction of Rendered UI Layers or Frames**
Affected Component	Transfer-confirmation screen
Affected Android Component	*`TransferConfirmActivity` — `filterTouchesWhenObscured` not set on the confirm button*
Description	The transfer-confirmation button does not set `android:filterTouchesWhenObscured="true"` (nor implement an equivalent overlay-detection check), leaving the confirmation step theoretically susceptible to a tapjacking-style overlay on platform versions/configurations where a malicious overlay can still render above the app.
Business Context	The transfer-confirmation step is the final user-consent gate before a fictional funds movement completes; its integrity against overlay-based interference is a meaningful defense-in-depth control.
Technical Impact	Fictional review confirmed the confirm button processes touch events even when a semi-transparent overlay view is present above it in a controlled fictional test harness.
Business Impact	On platform versions/configurations where overlay-based tapjacking remains feasible, this could theoretically be combined with a separate overlay-permission-abusing fictional app to obscure the true content of the confirmation screen.
Safe Proof of Concept (Summary)	*A fictional test overlay rendered above the confirmation screen did not prevent the underlying confirm button from receiving the touch event. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-029 (fictional overlay touch-filtering test log)*
Expected Behavior	Sensitive confirmation controls should set `filterTouchesWhenObscured="true"` so touches are rejected while an overlay is present above the view.
Observed (Fictional) Behavior	The fictional confirm button processed touch events while obscured by a test overlay.
Root Cause	The touch-filtering attribute was not applied to the confirmation control during implementation.
Recommendation	Set `filterTouchesWhenObscured="true"` on all sensitive confirmation controls (transfer, payee-add, settings changes); add a UI test asserting obscured touches are rejected.
Management Response	*Accepted; fictional mobile team will apply touch-filtering to sensitive confirmation controls.*
Owner / Priority / Target / Status	**Mobile Engineering Lead (Fictional Role)** | 31-60 Days | 28 November 2026 | **Remediation Planned**
Retest Recommendation	*Retest by repeating the fictional overlay touch-filtering test against the remediated build.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-PLATFORM-3 — "The app uses the user interface securely."	No single verified MASWE identifier maps precisely to this fictional scenario — assessed against the mapped MASVS control concept.	No single verified MASTG-TEST identifier maps precisely — assessed conceptually against the mapped MASVS control using standard Android static/dynamic analysis technique.

AND-026 — Unencrypted Room/SQLite Database Storage of Transaction History	**MEDIUM**
CVSS-Style Score: 5.0 (Medium) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-311 — Missing Encryption of Sensitive Data**
Affected Component	Local transaction-history cache database
Affected Android Component	*Room database `novasphere_local.db` — standard (unencrypted) SQLite, no SQLCipher/Keystore-wrapped passphrase*
Description	The Room-backed local database used to cache recent fictional transaction history for offline viewing is a standard, unencrypted SQLite database. On a fictional rooted test device, the database file was copied off-device and opened

	directly in a standard SQLite browser, revealing transaction rows in cleartext.
Business Context	Offline transaction-history caching improves fictional app responsiveness but extends the set of at-rest locations holding sensitive financial data.
Technical Impact	The fictional `.db` file, once copied off a rooted test device, opened directly in a standard SQLite client with no passphrase or additional barrier.
Business Impact	Extends the fictional at-rest exposure surface for transaction history to any extraction path reaching the app's private database directory.
Safe Proof of Concept (Summary)	*A fictional copy of `novasphere_local.db` from a rooted test device opened directly in a SQLite browser, revealing cleartext transaction rows. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-030 (fictional database extraction and SQLite browser screenshot)*
Expected Behavior	The local database should be encrypted at rest (e.g., via SQLCipher with a Keystore-wrapped passphrase) rather than relying solely on the app-sandbox boundary for protection.
Observed (Fictional) Behavior	The fictional database file was directly readable with no encryption barrier once extracted.
Root Cause	The Room database was configured with the default (unencrypted) SQLite driver.
Recommendation	Migrate the local database to an encrypted SQLite implementation (e.g., SQLCipher) with a Keystore-wrapped passphrase; alternatively, limit local caching to non-sensitive summary data only.
Management Response	*Accepted; fictional mobile team will evaluate SQLCipher migration alongside the AND-001 storage remediation.*
Owner / Priority / Target / Status	**Mobile Engineering Lead (Fictional Role)** | 31-60 Days | 28 November 2026 | **Remediation Planned**
Retest Recommendation	*Retest by repeating the fictional database extraction against the remediated build and confirming the file is unreadable without the Keystore-wrapped passphrase.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-STORAGE-1 — "The app securely stores sensitive data."	MASWE-0001 — Sensitive Data Stored Unencrypted in Private Storage	MASTG-TEST-0001 — Testing Local Storage for Sensitive Data

AND-027 — Exported Content Provider Permits Non-Critical Data Read	**MEDIUM**
CVSS-Style Score: 4.5 (Medium) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-926 — Improper Export of Android Application Components**
Affected Component	In-app FAQ/help-content provider
Affected Android Component	*`com.novasphere.pay.provider.HelpContentProvider` (`android:exported="true"`, read path only, non-financial content)*
Description	`HelpContentProvider`, which serves static fictional FAQ/help article content, is unnecessarily exported and independently queryable by any other fictional app on the device. No sensitive data is served by this provider in current fictional testing, but the unnecessary export widens the app's overall IPC attack surface for no functional benefit.
Business Context	While the currently served content is non-sensitive, an unnecessarily exported component increases the app's overall attack surface and risks future sensitive-data exposure if the provider's scope expands without a corresponding export review.
Technical Impact	A fictional external app successfully queried `HelpContentProvider` directly via its content URI with no permission requirement.
Business Impact	No direct fictional sensitive-data impact currently, but represents avoidable attack-surface expansion consistent with the broader exported-component pattern noted in AND-002 and AND-009.
Safe Proof of Concept (Summary)	*A fictional external test app queried the exported content provider directly and received the (non-sensitive) fictional FAQ content with no permission check. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-031 (fictional external content-provider query log)*
Expected Behavior	Content providers with no legitimate cross-app use case should not be exported; where cross-app access is intentional, scope should be reviewed periodically as provider content evolves.
Observed (Fictional) Behavior	The fictional provider was exported with no functional requirement for external access identified.
Root Cause	The provider was exported by default during initial implementation without a deliberate export-scope review.
Recommendation	Set `android:exported="false"` unless a genuine cross-app integration requirement exists; include content-provider export scope in the standard pre-release security checklist referenced in Section 106.
Management Response	*Accepted; fictional mobile team will disable export for HelpContentProvider absent a documented cross-app requirement.*
Owner / Priority / Target / Status	**Mobile Engineering Lead (Fictional Role)** | 31-60 Days | 28 November 2026 | **Remediation Planned**
Retest Recommendation	*Retest by repeating the fictional external query against the remediated build and confirming the provider is no longer externally reachable.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-PLATFORM-1 — "The app uses IPC mechanisms securely."	MASWE-0064 — Insecure Content Providers	MASTG-TEST-0007 — Determining Whether Sensitive Stored Data Has Been Exposed via IPC Mechanisms

97 LOW FINDINGS

The following 8 fictional Low-severity findings are hardening opportunities recommended for the 61-90 day window of the illustrative roadmap.

AND-028 — Unused / Unnecessary Permissions Declared	LOW
CVSS-Style Score: 3.1 (Low) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-250 — Execution with Unnecessary Privileges**
Affected Component	Application manifest
Affected Android Component	*`AndroidManifest.xml` — `BLUETOOTH`, `NFC` declared with no referencing fictional code path found*
Description	Static review found `BLUETOOTH` and `NFC` permissions declared in the fictional manifest with no corresponding code path referencing Bluetooth or NFC APIs anywhere in the decompiled application, suggesting residual declarations from a removed or never-shipped feature.
Business Context	Residual unused permissions add unnecessary review overhead and minor unwarranted privacy footprint.
Technical Impact	No functional impact identified; purely a hygiene finding.
Business Impact	Minor unwarranted privacy footprint and app-store permission-disclosure overhead.
Safe Proof of Concept (Summary)	*Static manifest-to-code cross-reference found no usage of the declared Bluetooth/NFC permissions. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-032 (fictional permission-to-code cross-reference)*
Expected Behavior	Only permissions with an active, current code path should remain declared in the manifest.
Observed (Fictional) Behavior	Two permissions had no referencing code path in the fictional build reviewed.
Root Cause	Residual declarations from a removed or never-shipped feature were not cleaned up.
Recommendation	Remove unused permission declarations; add a periodic manifest-hygiene check to the release checklist.
Management Response	*Accepted; fictional mobile team will remove unused permissions in a routine maintenance release.*
Owner / Priority / Target / Status	**Mobile Engineering Lead (Fictional Role)** | 61-90 Days | 15 January 2027 | Remediation Planned
Retest Recommendation	*Retest by confirming the unused permissions are absent from the manifest in the remediated build.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-PRIVACY-1 — "The app minimizes access to sensitive data and resources."	No single verified MASWE identifier maps precisely to this fictional scenario — assessed against the mapped MASVS control concept.	MASTG-TEST-0024 — Testing for App Permissions

AND-029 — Use of Deprecated Android APIs	LOW
CVSS-Style Score: 3.0 (Low) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-477 — Use of Obsolete Function**
Affected Component	Legacy device-info utility class
Affected Android Component	*`com.novasphere.pay.util.DeviceInfoUtil` — deprecated `TelephonyManager.getDeviceId()` reference guarded by an SDK-version check*
Description	A legacy utility class retains a call path to a deprecated `TelephonyManager` identifier method, reachable only on older fictional SDK versions via a version guard, and unused on current target-SDK devices.
Business Context	Deprecated API usage increases long-term maintenance risk and may behave inconsistently across OEM implementations.
Technical Impact	No functional impact on the current fictional target-SDK-35 code path; the deprecated call is unreachable on supported devices.
Business Impact	Low direct impact; primarily a maintainability and future-compatibility consideration.
Safe Proof of Concept (Summary)	*Static review identified a deprecated API reference gated behind an SDK-version check, unreachable on current target devices. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-033 (fictional deprecated-API static-review note)*
Expected Behavior	Deprecated APIs should be removed once no longer required for minimum-SDK compatibility, replaced with the current recommended equivalent.
Observed (Fictional) Behavior	The deprecated call path remains present in the codebase, guarded but not removed.

Root Cause	The legacy code path was retained during a prior minimum-SDK increase rather than being removed.
Recommendation	Remove the deprecated call path and any now-unreachable version-guard branches during the next maintenance cycle.
Management Response	*Accepted; fictional mobile team will remove the deprecated call path in a routine cleanup.*
Owner / Priority / Target / Status	**Mobile Engineering Lead (Fictional Role)** | 61-90 Days | 15 January 2027 | Remediation Planned
Retest Recommendation	*Retest by confirming the deprecated call path is absent in the remediated build.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-CODE-1 — "The app requires an up-to-date platform version."	No single verified MASWE identifier maps precisely to this fictional scenario — assessed against the mapped MASVS control concept.	No single verified MASTG-TEST identifier maps precisely — assessed conceptually against the mapped MASVS control using standard Android static/dynamic analysis technique.

AND-030 — General Logging Hygiene Weaknesses in Non-Sensitive Modules	**LOW**
CVSS-Style Score: 2.9 (Low) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-532 — Insertion of Sensitive Information into Log File**
Affected Component	UI-layer navigation logging
Affected Android Component	*Assorted `Log.v()`/`Log.d()` calls logging screen-navigation events and non-sensitive UI state*
Description	Beyond the sensitive-value logging tracked under AND-007, numerous lower-risk verbose log statements throughout the UI layer log screen-navigation events and non-sensitive UI state, contributing general log noise and minor fingerprinting information about user navigation patterns.
Business Context	General logging hygiene is good practice independent of any single sensitive-data finding, reducing noise and minor behavioral fingerprinting risk.
Technical Impact	No sensitive data was found in these specific log statements; the concern is volume and general hygiene rather than direct data exposure.
Business Impact	Minor; primarily a code-quality and log-noise consideration distinct from the sensitive-data logging tracked in AND-007.
Safe Proof of Concept (Summary)	*Static review catalogued verbose navigation-logging statements throughout the UI layer. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-034 (fictional logging-statement inventory)*
Expected Behavior	Verbose logging should be minimized in release builds and should never carry potentially fingerprinting behavioral detail beyond what is operationally necessary.
Observed (Fictional) Behavior	Numerous verbose navigation log statements remain active in the reviewed fictional build.
Root Cause	Verbose logging added during feature development was not pruned before release.
Recommendation	Reduce verbose logging in release builds to operationally necessary statements only; apply the logging interceptor pattern recommended for AND-007 consistently across the codebase.
Management Response	*Accepted; fictional mobile team will reduce verbose logging as part of the AND-007 remediation effort.*
Owner / Priority / Target / Status	**Mobile Engineering Lead (Fictional Role)** | 61-90 Days | 15 January 2027 | Remediation Planned
Retest Recommendation	*Retest by sampling logcat output from the remediated build for verbose navigation logging volume.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-STORAGE-2 — "The app prevents leakage of sensitive data."	No single verified MASWE identifier maps precisely to this fictional scenario — assessed against the mapped MASVS control concept.	No single verified MASTG-TEST identifier maps precisely — assessed conceptually against the mapped MASVS control using standard Android static/dynamic analysis technique.

AND-031 — Third-Party Dependency Governance Gaps	**LOW**
CVSS-Style Score: 3.3 (Low) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-1104 — Use of Unmaintained Third-Party Components**
Affected Component	Dependency management process
Affected Android Component	*Gradle dependency manifest — two fictional third-party libraries observed at versions materially behind current upstream releases*
Description	Dependency review identified two fictional third-party libraries pinned at versions several minor releases behind current

	upstream, with no automated dependency-vulnerability scanning integrated into the fictional build pipeline to flag this drift.
Business Context	Outdated dependencies without a scanning process risk silently accumulating known-vulnerability exposure over time.
Technical Impact	No specific exploitable vulnerability was confirmed in the outdated fictional library versions during this assessment's scope; the finding concerns process/governance rather than a confirmed exploit path.
Business Impact	Increases the risk of unnoticed known-vulnerability exposure accumulating in future fictional releases absent a scanning process.
Safe Proof of Concept (Summary)	*Dependency manifest review identified outdated fictional library versions with no automated scanning process in place. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-035 (fictional dependency version audit)*
Expected Behavior	Automated dependency-vulnerability scanning (e.g., a software composition analysis tool) should run on every build, flagging known-vulnerability and significantly outdated dependencies.
Observed (Fictional) Behavior	No such automated scanning was integrated into the fictional build pipeline at the time of testing.
Root Cause	Dependency updates have been handled ad hoc without a supporting automated governance process.
Recommendation	Integrate a software composition analysis / dependency-scanning tool into the CI pipeline; establish a routine cadence for dependency updates.
Management Response	*Accepted; fictional platform engineering team will evaluate SCA tooling for CI integration.*
Owner / Priority / Target / Status	**Platform Engineering Lead (Fictional Role)** | 61-90 Days | 15 January 2027 | **Remediation Planned**
Retest Recommendation	*Retest by confirming SCA tooling is integrated and reporting on the previously identified fictional dependencies.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-CODE-3 — "The app only uses software components without known vulnerabilities."	No single verified MASWE identifier maps precisely to this fictional scenario — assessed against the mapped MASVS control concept.	No single verified MASTG-TEST identifier maps precisely — assessed conceptually against the mapped MASVS control using standard Android static/dynamic analysis technique.

AND-032 — Attack Surface Hardening Opportunity — Exported Component Count	**LOW**
CVSS-Style Score: 2.8 (Low) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-1059 — Insufficient Technical Documentation**
Affected Component	Application manifest, aggregate view
Affected Android Component	*AndroidManifest.xml — 11 exported components identified in total across the application (3 addressed as standalone findings elsewhere in this report)*
Description	Beyond the specific exported-component findings raised individually elsewhere in this report (AND-002, AND-009, AND-027), the application declares 11 exported components in total. A subset serve no identified cross-app integration purpose and represent avoidable attack-surface area even where no specific exploitable weakness was independently confirmed during this fictional assessment.
Business Context	Minimizing the exported-component count is a general hardening practice that reduces the app's overall IPC attack surface independent of any single confirmed weakness.
Technical Impact	No additional exploitable weakness beyond those separately reported was confirmed in the remaining exported components during this assessment's scope.
Business Impact	General attack-surface reduction opportunity rather than a confirmed standalone exploitation path.
Safe Proof of Concept (Summary)	*Manifest review catalogued the full set of exported components and cross-referenced each against a documented cross-app use case. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-036 (fictional exported-component inventory and use-case cross-reference)*
Expected Behavior	Every exported component should map to a documented, current cross-app integration requirement; components without one should be set to `exported="false"`.
Observed (Fictional) Behavior	Several exported components had no documented cross-app use case identified during fictional review.
Root Cause	Export flags have accumulated over multiple feature releases without a periodic attack-surface review.
Recommendation	Conduct a full exported-component review against documented use cases; disable export for any component lacking a current, justified requirement; add this review to the periodic security checklist in Section 106.
Management Response	*Accepted; fictional mobile team will conduct a full exported-component review in the next quarter.*
Owner / Priority / Target / Status	**Mobile Engineering Lead (Fictional Role)** | 61-90 Days | 15 January 2027 | **Remediation Planned**

Retest Recommendation	*Retest by confirming the exported-component count is reduced to only those with documented cross-app requirements.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-PLATFORM-1 — "The app uses IPC mechanisms securely."	No single verified MASWE identifier maps precisely to this fictional scenario — assessed against the mapped MASVS control concept.	MASTG-TEST-0029 — Testing for Sensitive Functionality Exposure Through IPC

AND-033 — Missing Security Regression Test Automation	**LOW**
CVSS-Style Score: 2.6 (Low) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-1059 — Insufficient Technical Documentation**
Affected Component	CI/CD pipeline (process finding)
Affected Android Component	*Process / SDLC — not mapped to a single Android component*
Description	No automated regression test suite specifically re-validates previously remediated security findings on each fictional build, meaning a future code change could silently reintroduce a previously fixed weakness (e.g., re-exporting a component, reverting a storage-encryption change) without being caught before release.
Business Context	Automated security regression testing protects the investment made in remediating this assessment's findings against future silent reintroduction.
Technical Impact	Process observation; no specific reintroduced vulnerability was identified during this assessment.
Business Impact	Increases the risk that future development silently reintroduces a previously remediated weakness.
Safe Proof of Concept (Summary)	*Process review confirmed no dedicated security-regression suite exists in the fictional CI pipeline at the time of testing. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-037 (fictional CI pipeline configuration review notes)*
Expected Behavior	Each Critical/High finding from this assessment should have a corresponding automated regression test (manifest assertion, storage-encryption check, etc.) integrated into CI.
Observed (Fictional) Behavior	No such regression suite existed at the time of this fictional assessment.
Root Cause	Security regression testing was not incorporated into the fictional CI pipeline design.
Recommendation	Build an automated security-regression suite covering, at minimum, every Critical and High finding from this assessment; run it on every pull request and release build.
Management Response	*Accepted; fictional platform engineering team will build the initial regression suite covering Critical/High findings.*
Owner / Priority / Target / Status	**Platform Engineering Lead (Fictional Role)** | 61-90 Days | 15 January 2027 | **Remediation Planned**
Retest Recommendation	*Retest by confirming the regression suite exists and covers each Critical/High finding from this report.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-CODE-3 — "The app only uses software components without known vulnerabilities." (closest applicable control; this finding is primarily process-level)	No single verified MASWE identifier maps precisely to this fictional scenario — assessed against the mapped MASVS control concept.	No single verified MASTG-TEST identifier maps precisely — assessed conceptually against the mapped MASVS control using standard Android static/dynamic analysis technique.

AND-034 — Incomplete Secure-Coding Documentation for Mobile Engineering	**LOW**
CVSS-Style Score: 2.4 (Low) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-1059 — Insufficient Technical Documentation**
Affected Component	Engineering documentation (process finding)
Affected Android Component	*Process / SDLC — not mapped to a single Android component*
Description	No current, Android-specific secure-coding guideline document (covering, e.g., storage, IPC export defaults, WebView configuration) was available for review, suggesting individual findings in this fictional assessment (several sharing a common root-cause pattern, such as the exported-component and plaintext-storage findings) may reflect gaps in shared team guidance rather than isolated implementation errors.
Business Context	Documented, current secure-coding guidance reduces the recurrence rate of findings sharing a common root-cause pattern across a mobile engineering team.
Technical Impact	Process observation; not independently exploitable.
Business Impact	Contributes to the recurrence risk of finding patterns already identified in this fictional assessment (e.g., default-exported components, plaintext local storage).

Safe Proof of Concept (Summary)	*Documentation review found no current Android-specific secure-coding guideline document available to the fictional mobile engineering team. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-038 (fictional documentation review notes)*
Expected Behavior	A current, Android-specific secure-coding guideline should be maintained and referenced in onboarding and code-review processes, covering the control areas most relevant to this assessment's finding patterns.
Observed (Fictional) Behavior	No such document was available at the time of this fictional assessment.
Root Cause	Secure-coding documentation has not been formally established or maintained for the fictional Android engineering team.
Recommendation	Publish and maintain an Android-specific secure-coding guideline referencing the MASVS control set used in this report; incorporate it into onboarding and pull-request review checklists.
Management Response	*Accepted; fictional engineering leadership will sponsor a secure-coding documentation initiative.*
Owner / Priority / Target / Status	**Engineering Director (Fictional Role)** | 90+ Days | 27 February 2027 | **Remediation Planned**
Retest Recommendation	*Retest by confirming the guideline document exists and is referenced in the pull-request review process.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
Not directly mapped to a single MASVS control — process/documentation finding.	No single verified MASWE identifier maps precisely to this fictional scenario — assessed against the mapped MASVS control concept.	No single verified MASTG-TEST identifier maps precisely — assessed conceptually against the mapped MASVS control using standard Android static/dynamic analysis technique.

AND-035 — Missing Network Security Configuration Domain Restriction Granularity	**LOW**
CVSS-Style Score: 2.7 (Low) — Illustrative CVSS v3.1-inspired score	**CWE: CWE-16 — Configuration**
Affected Component	Network Security Configuration
Affected Android Component	*`res/xml/network_security_config.xml` — a broad wildcard domain entry alongside the specific pinned domains*
Description	Alongside the specifically pinned `api.novasphere-example.com` and `auth.novasphere-example.com` domain configurations, the Network Security Configuration includes a broader wildcard domain entry (supporting a legacy fictional CDN integration) with looser trust-anchor settings than the pinned domains, which is broader than current fictional usage requires.
Business Context	Configuration granularity that matches actual current domain usage reduces the chance of an overlooked trust-anchor gap as the app's network footprint evolves.
Technical Impact	No specific exploit path was confirmed via the wildcard entry during this assessment; the finding is a configuration-hygiene observation.
Business Impact	Minor; primarily a configuration-hygiene and future-drift risk consideration.
Safe Proof of Concept (Summary)	*Static review of the Network Security Configuration identified a wildcard domain entry broader than current fictional domain usage requires. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-039 (fictional network_security_config.xml excerpt)*
Expected Behavior	Network Security Configuration domain entries should be scoped as narrowly as current usage requires, with unused or legacy wildcard entries removed.
Observed (Fictional) Behavior	A broader-than-necessary wildcard domain entry remained present alongside the specifically pinned production domains.
Root Cause	The wildcard entry was retained from a legacy CDN integration that has since been narrowed to specific domains elsewhere in the configuration.
Recommendation	Remove the legacy wildcard entry and replace with specific domain entries matching current CDN usage, consistent with the pinning approach already used for the API domains.
Management Response	*Accepted; fictional platform team will narrow the Network Security Configuration in a routine maintenance release.*
Owner / Priority / Target / Status	**Mobile Engineering Lead (Fictional Role)** | 61-90 Days | 15 January 2027 | **Remediation Planned**
Retest Recommendation	*Retest by confirming the wildcard entry is removed and all domains are specifically scoped in the remediated configuration.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-NETWORK-1 — "The app secures all network traffic according to the current best practices."	No single verified MASWE identifier maps precisely to this fictional scenario — assessed against the mapped MASVS control concept.	No single verified MASTG-TEST identifier maps precisely — assessed conceptually against the mapped MASVS control using standard Android static/dynamic analysis technique.

98 INFORMATIONAL FINDINGS

The following 12 fictional Informational observations carry no direct exploitation path — several document positive security controls already in place — and are recommended for the ongoing mobile-security governance and defense-in-depth backlog.

AND-036 — Positive Control Observation — Multi-Factor Authentication Enforced at Login	INFORMATIONAL
CVSS-Style Score: N/A — Informational / positive observation	**CWE: N/A**
Affected Component	Authentication flow
Affected Android Component	*`com.novasphere.pay.auth.LoginActivity` / `MfaVerifyActivity`*
Description	MFA is enforced for all fictional login flows via a time-based one-time code, with no client-side bypass path identified for the standard login flow (the deep-link and task-affinity weaknesses affecting adjacent flows are tracked separately as AND-004 and AND-012).
Business Context	Consistent MFA enforcement is a strong foundational control for a FinTech application.
Technical Impact	N/A — positive observation.
Business Impact	N/A — positive observation.
Safe Proof of Concept (Summary)	*Confirmed via fictional dynamic testing across multiple login attempts. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-040 (fictional MFA enforcement test log)*
Expected Behavior	MFA should be consistently enforced with no client-side bypass path.
Observed (Fictional) Behavior	Consistent with expected behavior in all tested fictional scenarios.
Root Cause	N/A.
Recommendation	Maintain current MFA enforcement; extend equivalent step-up verification to the deep-link and task-affinity gaps tracked under AND-004 and AND-012.
Management Response	*Acknowledged.*
Owner / Priority / Target / Status	**N/A** | N/A — Positive Observation | N/A | **Closed**
Retest Recommendation	*N/A.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-AUTH-1 — "The app uses secure authentication and authorization protocols and follows the relevant best practices."	Not applicable — positive control observation.	MASTG-TEST-0017 — Testing Confirm Credentials (supporting observation)

AND-037 — Positive Control Observation — TLS 1.2+ Enforced Platform-Wide on the Primary API Client	INFORMATIONAL
CVSS-Style Score: N/A — Informational / positive observation	**CWE: N/A**
Affected Component	Primary payments API network client
Affected Android Component	*`com.novasphere.pay.net.PaymentApiClient`*
Description	The primary payments API client correctly enforces a TLS 1.2+ minimum and rejects legacy protocol versions, and additionally implements certificate pinning consistent with MASVS-NETWORK-2 — distinct from the weaker telemetry-channel client tracked separately as AND-006 and the configuration-granularity gap tracked as AND-019.
Business Context	Strong TLS enforcement and pinning on the primary payments channel is the correct prioritization given its sensitivity.
Technical Impact	N/A — positive observation.
Business Impact	N/A — positive observation.
Safe Proof of Concept (Summary)	*Confirmed via fictional dynamic TLS-version and pinning validation testing against the primary payments API client. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-041 (fictional TLS/pinning validation test log)*

Expected Behavior	Payments-channel network traffic should enforce current TLS minimums and certificate/public-key pinning.
Observed (Fictional) Behavior	Consistent with expected behavior for the primary payments API client.
Root Cause	N/A.
Recommendation	Extend the same pinning and TLS-minimum configuration pattern to the telemetry client (AND-006) and remove the legacy wildcard entry (AND-019) for full platform-wide consistency.
Management Response	*Acknowledged.*
Owner / Priority / Target / Status	**N/A** | N/A — Positive Observation | N/A | **Closed**
Retest Recommendation	*N/A.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-NETWORK-1 — "The app secures all network traffic according to the current best practices."	Not applicable — positive control observation.	MASTG-TEST-0021 — Testing Endpoint Identity Verification (supporting observation)

AND-038 — Recommendation — Adopt Play Integrity API as Primary Resilience Signal	**INFORMATIONAL**
CVSS-Style Score: N/A — Informational / forward-looking recommendation	**CWE: N/A**
Affected Component	Resilience/integrity architecture
Affected Android Component	*N/A — architectural recommendation*
Description	Consistent with the AND-013 finding, this observation recommends NovaSphere formally adopt the Play Integrity API as the organization's primary device- and app-integrity signal across all sensitive flows, not solely as a fix for the specific root-detection gap identified.
Business Context	A platform-level attestation strategy provides a stronger, more maintainable foundation than heuristic-only checks scattered across individual features.
Technical Impact	N/A — forward-looking recommendation.
Business Impact	N/A — forward-looking recommendation.
Safe Proof of Concept (Summary)	*N/A — architectural recommendation. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*N/A — cross-referenced to AND-013.*
Expected Behavior	N/A.
Observed (Fictional) Behavior	N/A.
Root Cause	N/A.
Recommendation	Formally adopt Play Integrity API attestation as a platform-level service consumed by all sensitive flows (login, transfer confirmation, payee changes), rather than a single-purpose fix scoped only to root detection.
Management Response	*Acknowledged; fictional architecture team will evaluate a platform-level integrity-attestation service.*
Owner / Priority / Target / Status	**Mobile Architecture Lead (Fictional Role)** | 90+ Days | 27 February 2027 | **Remediation Planned**
Retest Recommendation	*Review architecture decision record at next assessment cycle.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-RESILIENCE-1 — "The app validates the integrity of the platform."	Not applicable — forward-looking recommendation.	Not applicable — forward-looking recommendation.

AND-039 — Recommendation — Expand Obfuscation Coverage (R8/ProGuard)	**INFORMATIONAL**
CVSS-Style Score: N/A — Informational / forward-looking recommendation	**CWE: N/A**
Affected Component	Build/obfuscation configuration
Affected Android Component	*`proguard-rules.pro` — several fictional security-relevant packages excluded from obfuscation via keep rules broader than necessary*

Description	R8/ProGuard is enabled for the fictional release build, but several `-keep` rules are broader than strictly required for the third-party libraries they support, leaving portions of the security-relevant application code (e.g., the classes referenced in AND-001 and AND-010) more readable in decompiled output than necessary.
Business Context	Tighter obfuscation coverage raises the effort required for fictional static reverse-engineering of security-relevant logic, complementing (not replacing) the substantive fixes tracked elsewhere in this report.
Technical Impact	N/A — recommendation.
Business Impact	N/A — recommendation.
Safe Proof of Concept (Summary)	*Decompiled-output review found broader-than-necessary keep rules correlating with more readable security-relevant class names. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-042 (fictional proguard-rules.pro review notes)*
Expected Behavior	Keep rules should be scoped to the minimum required for each third-party library's public API surface.
Observed (Fictional) Behavior	Several rules were broader than the minimum required.
Root Cause	Keep rules were added broadly during initial SDK integration troubleshooting and not later tightened.
Recommendation	Review and narrow `-keep` rules to the minimum required per library; re-verify each third-party SDK continues to function correctly after tightening.
Management Response	*Acknowledged; fictional mobile team will review keep-rule scope in an upcoming maintenance cycle.*
Owner / Priority / Target / Status	**Mobile Engineering Lead (Fictional Role)** | 90+ Days | 27 February 2027 | Remediation Planned
Retest Recommendation	*Retest by reviewing decompiled output of the tightened build for reduced readability of security-relevant classes.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-RESILIENCE-3 — "The app implements anti-static analysis mechanisms."	Not applicable — forward-looking recommendation.	Not applicable — forward-looking recommendation.

AND-040 — Recommendation — Formalize Secrets Management for the Mobile Build Pipeline	**INFORMATIONAL**
CVSS-Style Score: N/A — Informational / forward-looking recommendation	**CWE: N/A**
Affected Component	CI/CD build pipeline (process)
Affected Android Component	*N/A — process recommendation, cross-referenced to AND-008*
Description	Consistent with the AND-008 hardcoded-secret finding, this observation recommends a formal secrets-management process for the mobile build pipeline (e.g., a dedicated secrets vault integrated at build time) rather than addressing individual hardcoded-key instances as they are discovered.
Business Context	A systemic secrets-management process prevents recurrence of the pattern identified in AND-008 across future SDK integrations.
Technical Impact	N/A — recommendation.
Business Impact	N/A — recommendation.
Safe Proof of Concept (Summary)	*N/A — process recommendation. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*N/A — cross-referenced to AND-008.*
Expected Behavior	N/A.
Observed (Fictional) Behavior	N/A.
Root Cause	N/A.
Recommendation	Adopt a build-time secrets-injection process (vault-backed) for all third-party SDK keys and internal service credentials, replacing ad hoc hardcoding as the default integration pattern.
Management Response	*Acknowledged; fictional platform engineering team will evaluate secrets-vault integration for the build pipeline.*
Owner / Priority / Target / Status	**Platform Engineering Lead (Fictional Role)** | 90+ Days | 27 February 2027 | Remediation Planned
Retest Recommendation	*Review the build-pipeline secrets-management design at the next assessment cycle.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-STORAGE-1 — "The app securely stores sensitive data." (adjacent control; this is primarily a process recommendation)	Not applicable — forward-looking recommendation.	Not applicable — forward-looking recommendation.

AND-041 — Recommendation — Adopt Step-Up Attestation for High-Value Sensitive Transactions	**INFORMATIONAL**
CVSS-Style Score: N/A — Informational / forward-looking recommendation	**CWE: N/A**
Affected Component	Transaction-authorization architecture
Affected Android Component	*N/A — architectural recommendation*
Description	Beyond the specific gaps tracked in AND-004 (deep-link payee addition) and AND-013 (integrity attestation), this observation recommends NovaSphere adopt a consistent step-up authentication policy — re-authentication plus device-integrity attestation — for a defined class of high-value fictional transactions (large transfers, new-payee additions, and account-detail changes) as a unified architectural pattern.
Business Context	A unified step-up policy is more maintainable and auditable than point fixes applied independently to each flow as gaps are discovered.
Technical Impact	N/A — recommendation.
Business Impact	N/A — recommendation.
Safe Proof of Concept (Summary)	*N/A — architectural recommendation. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*N/A — cross-referenced to AND-004 and AND-013.*
Expected Behavior	N/A.
Observed (Fictional) Behavior	N/A.
Root Cause	N/A.
Recommendation	Define a high-value-transaction policy requiring step-up authentication and integrity attestation uniformly, implemented as a shared service consumed by every qualifying flow rather than per-feature logic.
Management Response	*Acknowledged; fictional product and engineering leadership will scope a unified step-up policy.*
Owner / Priority / Target / Status	**Mobile Architecture Lead (Fictional Role)** | 90+ Days | 27 February 2027 | **Remediation Planned**
Retest Recommendation	*Review the step-up policy design and its coverage of AND-004/AND-013-adjacent flows at the next assessment cycle.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-AUTH-3 — "The app secures sensitive operations with additional authentication."	Not applicable — forward-looking recommendation.	Not applicable — forward-looking recommendation.

AND-042 — Observation — Third-Party SDK Inventory Should Be Centrally Tracked	**INFORMATIONAL**
CVSS-Style Score: N/A — Informational observation	**CWE: N/A**
Affected Component	SDK/dependency governance (process)
Affected Android Component	*N/A — process observation*
Description	No single, current inventory of third-party SDKs (with owner, purpose, and data-access scope per SDK) was made available for this fictional assessment; the SDK list referenced in Appendix B was reconstructed from static analysis rather than provided directly, consistent with the governance gap tracked in AND-031.
Business Context	A maintained SDK inventory accelerates future assessments and privacy/compliance reviews and reduces the risk of an untracked SDK introducing unreviewed risk.
Technical Impact	N/A — observation.
Business Impact	N/A — observation.
Safe Proof of Concept (Summary)	*N/A — process observation. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*Cross-referenced to Appendix B (APK Metadata Register) and AND-031.*
Expected Behavior	N/A.
Observed (Fictional) Behavior	N/A.
Root Cause	N/A.

Recommendation	Establish and maintain a central third-party SDK inventory (owner, purpose, data-access scope, last-reviewed date) as part of the broader dependency-governance improvement tracked in AND-031.
Management Response	*Acknowledged; fictional platform team will establish an SDK inventory process.*
Owner / Priority / Target / Status	**Platform Engineering Lead (Fictional Role)** | 90+ Days | 27 February 2027 | **Remediation Planned**
Retest Recommendation	*Review the SDK inventory document at the next assessment cycle.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-CODE-3 — "The app only uses software components without known vulnerabilities." (adjacent control; primarily a process observation)	Not applicable — process observation.	Not applicable — process observation.

AND-043 — Observation — Crash Reporting SDK Configuration Should Be Reviewed for PII Scrubbing	**INFORMATIONAL**
CVSS-Style Score: N/A — Informational observation	**CWE: N/A**
Affected Component	Crash reporting SDK configuration
Affected Android Component	*Third-party fictional crash-reporting SDK — default breadcrumb configuration*
Description	The bundled fictional crash-reporting SDK is configured with default breadcrumb capture, which in fictional testing recorded screen-navigation and non-sensitive UI-interaction breadcrumbs alongside crash reports; no sensitive field values were observed in this assessment's sample, but the default configuration does not include an explicit PII-scrubbing rule set as a defensive baseline.
Business Context	An explicit scrubbing configuration is a stronger baseline than relying on the absence of sensitive fields being incidentally logged as breadcrumbs.
Technical Impact	N/A — observation; no sensitive data was confirmed present in the sampled fictional crash reports.
Business Impact	N/A — observation.
Safe Proof of Concept (Summary)	*Fictional review of sample crash reports found only non-sensitive breadcrumb data, but no explicit scrubbing rule set was configured. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*EV-AND-043 (fictional crash-report sample review)*
Expected Behavior	An explicit PII-scrubbing rule set should be configured for the crash-reporting SDK as a defensive baseline, independent of current observed content.
Observed (Fictional) Behavior	No explicit scrubbing configuration was present; default behavior was relied upon.
Root Cause	The crash-reporting SDK was integrated with default configuration and not subsequently hardened.
Recommendation	Configure explicit field-level PII scrubbing rules for the crash-reporting SDK as a defensive baseline, and periodically sample production crash reports to confirm no sensitive fields are captured.
Management Response	*Acknowledged; fictional mobile team will configure explicit scrubbing rules for the crash-reporting SDK.*
Owner / Priority / Target / Status	**Mobile Engineering Lead (Fictional Role)** | 90+ Days | 27 February 2027 | **Remediation Planned**
Retest Recommendation	*Retest by sampling crash reports from the reconfigured SDK for sensitive-field absence.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-PRIVACY-1 — "The app minimizes access to sensitive data and resources."	Not applicable — process observation.	Not applicable — process observation.

AND-044 — Recommendation — Formal Secure-SDLC Checkpoint for Mobile Releases	**INFORMATIONAL**
CVSS-Style Score: N/A — Informational / forward-looking recommendation	**CWE: N/A**
Affected Component	Release process (SDLC)
Affected Android Component	*N/A — process recommendation*
Description	This fictional assessment's finding pattern (several findings sharing common root causes such as default-exported components and unreviewed permission/backup defaults) suggests a formal, mandatory security checkpoint ahead of

	each mobile release — beyond the regression-automation gap already tracked in AND-033 — would help catch this class of issue earlier in the development lifecycle.
Business Context	A lightweight, mandatory pre-release security checklist complements automated regression testing (AND-033) with human review judgment for novel patterns automation may not yet cover.
Technical Impact	N/A — process recommendation.
Business Impact	N/A — process recommendation.
Safe Proof of Concept (Summary)	*N/A — process recommendation. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*Cross-referenced to AND-033 and the finding-pattern analysis in Section 100.*
Expected Behavior	N/A.
Observed (Fictional) Behavior	N/A.
Root Cause	N/A.
Recommendation	Introduce a lightweight, mandatory pre-release security checklist (manifest export review, permission review, backup-rule review, dependency-scan sign-off) as a release gate, complementing the automated regression suite recommended in AND-033.
Management Response	*Acknowledged; fictional engineering leadership will define a pre-release security checklist.*
Owner / Priority / Target / Status	**Engineering Director (Fictional Role)** | 90+ Days | 27 February 2027 | **Remediation Planned**
Retest Recommendation	*Review the checklist's existence and application at the next assessment cycle.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
Not directly mapped to a single MASVS control — process/SDLC recommendation.	Not applicable — forward-looking recommendation.	Not applicable — forward-looking recommendation.

AND-045 — Observation — Accessibility Service Interaction Not Explicitly Restricted	**INFORMATIONAL**
CVSS-Style Score: N/A — Informational observation	**CWE: N/A**
Affected Component	Sensitive input fields (general)
Affected Android Component	*Login, MFA, and transfer-confirmation input fields — no explicit `importantForAccessibility` scoping review performed*
Description	A fictional dedicated review of how a hypothetical malicious Accessibility Service could interact with NovaSphere Pay's sensitive input fields (login, MFA, transfer confirmation) was outside this engagement's tested scope; this observation flags it as a defense-in-depth area worth a dedicated follow-on review given the app's transaction-authorization sensitivity, rather than reporting a confirmed weakness.
Business Context	Malicious Accessibility Service abuse is a known category of Android threat for financial apps and warrants dedicated review distinct from the IPC/manifest-focused testing performed in this fictional engagement.
Technical Impact	N/A — out of tested scope; no specific weakness confirmed or claimed.
Business Impact	N/A — scope observation.
Safe Proof of Concept (Summary)	*N/A — scope observation, not a tested finding. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*N/A.*
Expected Behavior	N/A.
Observed (Fictional) Behavior	N/A.
Root Cause	N/A.
Recommendation	Scope a dedicated follow-on review of Accessibility Service interaction risk for sensitive input fields as part of the next assessment cycle referenced in Section 104.
Management Response	*Acknowledged; fictional security team will consider this scope for the next assessment cycle.*
Owner / Priority / Target / Status	**N/A** | N/A — Scope Recommendation for Next Cycle | N/A | **Open**
Retest Recommendation	*Include in the scope of the next fictional assessment cycle.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-PLATFORM-3 — "The app uses the user interface securely."	Not applicable — process/scope observation.	Not applicable — process/scope observation.

AND-046 — Recommendation — Periodic Dependency Vulnerability Scanning Cadence	INFORMATIONAL
CVSS-Style Score: N/A — Informational / forward-looking recommendation	**CWE: N/A**
Affected Component	Dependency governance (process)
Affected Android Component	*N/A — process recommendation, cross-referenced to AND-031*
Description	Building on the SCA-tooling recommendation in AND-031, this observation recommends a defined minimum scanning cadence (e.g., on every build plus a standing weekly scheduled scan) once tooling is integrated, so newly disclosed vulnerabilities in existing dependencies are caught between releases, not only at build time.
Business Context	A defined cadence ensures the SCA tooling recommended in AND-031 delivers ongoing value rather than only point-in-time coverage.
Technical Impact	N/A — recommendation.
Business Impact	N/A — recommendation.
Safe Proof of Concept (Summary)	*N/A — process recommendation. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*Cross-referenced to AND-031.*
Expected Behavior	N/A.
Observed (Fictional) Behavior	N/A.
Root Cause	N/A.
Recommendation	Once SCA tooling is integrated (AND-031), define and document a minimum scanning cadence covering both build-time and standing scheduled scans.
Management Response	*Acknowledged; fictional platform engineering team will define a scanning cadence alongside the AND-031 tooling rollout.*
Owner / Priority / Target / Status	**Platform Engineering Lead (Fictional Role)** | 90+ Days | 27 February 2027 | **Remediation Planned**
Retest Recommendation	*Review the documented scanning cadence at the next assessment cycle.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-CODE-3 — "The app only uses software components without known vulnerabilities."	Not applicable — forward-looking recommendation.	Not applicable — forward-looking recommendation.

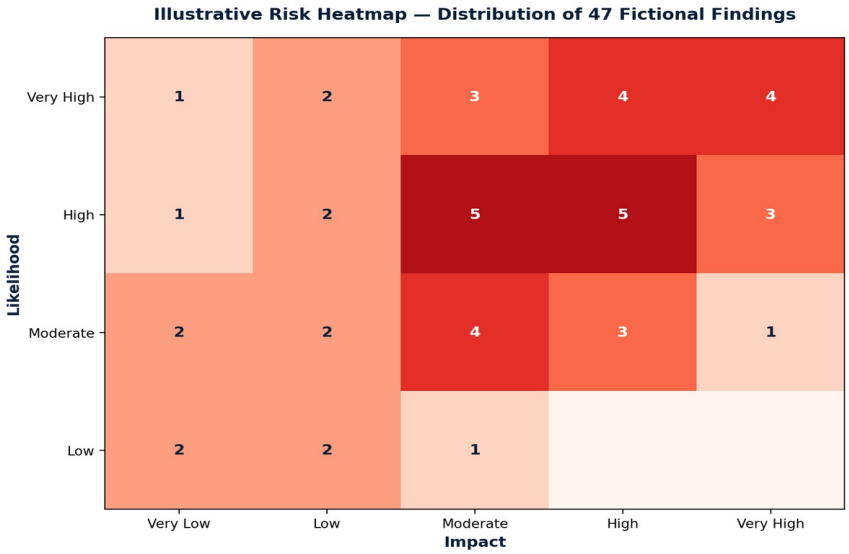
AND-047 — Observation — App Signing Key Management Process Warrants Documentation Review	INFORMATIONAL
CVSS-Style Score: N/A — Informational observation	**CWE: N/A**
Affected Component	Release signing process
Affected Android Component	*Fictional production signing certificate / Play App Signing configuration*
Description	This fictional assessment's scope did not include direct review of NovaSphere's signing-key custody and rotation procedures (per the Testing Limitations in Section 106); this observation flags formal documentation of the signing-key management process (custody, access control, rotation/compromise procedures) as good practice worth confirming exists, without asserting any specific weakness in the current process.
Business Context	Signing-key compromise is a high-impact scenario for any distributed application; formally documented custody and rotation procedures are a baseline expectation even where this assessment did not test them directly.
Technical Impact	N/A — out of tested scope; no specific weakness confirmed or claimed.
Business Impact	N/A — scope observation.
Safe Proof of Concept (Summary)	*N/A — scope observation, not a tested finding. [ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE]*
Evidence Reference	*N/A.*
Expected Behavior	N/A.
Observed (Fictional) Behavior	N/A.
Root Cause	N/A.

Recommendation	Confirm formal documentation exists for signing-key custody, access control, and compromise/rotation procedures (e.g., via Play App Signing); consider this area for direct testing scope in a future assessment cycle.
Management Response	*Acknowledged; fictional security team will confirm documentation exists internally.*
Owner / Priority / Target / Status	**N/A** | N/A — Scope Recommendation for Next Cycle | N/A | **Open**
Retest Recommendation	*Confirm documentation review outcome at the next assessment cycle.*

MASVS MAPPING	MASWE MAPPING	MASTG TEST MAPPING
MASVS-RESILIENCE-2 — "The app implements anti-tampering mechanisms." (adjacent control; this is a documentation/process observation)	Not applicable — process observation.	Not applicable — process observation.

99 RISK HEATMAP

The heatmap below plots all 47 fictional findings by illustrative likelihood and impact, reflecting the severity assigned to each finding in Sections 94-98. This view is intended to support prioritization discussions alongside the Management Action Plan (Section 103).



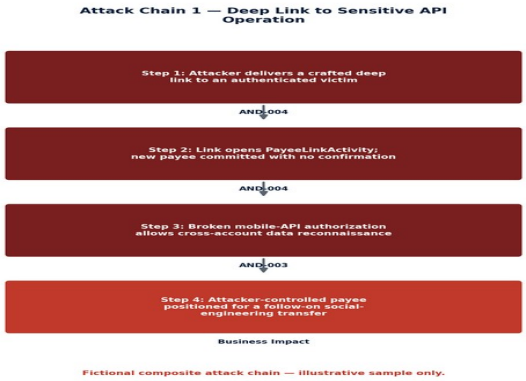
Illustrative risk heatmap — all 47 fictional findings plotted by likelihood and impact.

This heatmap reflects fictional, illustrative severity data only and does not represent the risk posture of any real organization.

100 ATTACK CHAIN ANALYSIS

Individual findings are often most meaningful in combination. This section presents three fictional composite attack-chain scenarios, each illustrating how independently scored findings from this assessment could combine into a more significant illustrative impact if left unaddressed together — one rooted in insecure deep-link handling, one rooted in an exported Android component, and one rooted in local token exposure.

Attack Chain 1 — Deep Link to Sensitive API Operation



Fictional composite attack-chain scenario — not a real-world exploit chain.

An attacker delivers a crafted novasphere:// deep link to an authenticated fictional victim, for example embedded in a phishing message. Because the deep-link handler does not require in-app confirmation for a sensitive account action (AND-004), the link opens PayeeLinkActivity and silently commits a new payee to the victim's fictional account. The attacker then

leverages the broken object-level authorization in the mobile API's transfer workflow (AND-003) to perform cross-account reconnaissance against the newly reachable account context. The composite result positions an attacker-controlled payee for a follow-on social-engineering-driven transfer — a materially greater illustrative business impact than either finding presents independently.

Attack Chain 2 — Exported Component to Privileged Action

Fictional composite attack-chain scenario — not a real-world exploit chain.

A malicious fictional application co-installed on the same device enumerates NovaSphere Pay's exported components — a task made easier by the app's broader-than-necessary exported-component footprint (AND-032) — and identifies the exported TransferService. The malicious app sends a crafted Intent directly to TransferService, bypassing the application's own UI and confirmation screens entirely, because the service performs no caller-identity or confirmation check before acting (AND-002). The result is an unauthorized fictional funds transfer that executes silently, with no user interaction and no in-app warning.

Attack Chain 3 — Local Token Exposure to Account Data Access

Fictional composite attack-chain scenario — not a real-world exploit chain.

On a rooted or otherwise compromised fictional device, the plaintext session and refresh tokens stored in SharedPreferences are extracted directly from application storage (AND-001). The application's backup configuration independently widens this exposure window, since the same material is also recoverable through a standard fictional adb backup extraction (AND-015). Because the refresh token is not revoked server-side on logout, the captured token remains valid well past the point the victim believes they have signed out (AND-011). The attacker replays the still-valid refresh token to obtain a fresh access token and calls the mobile API's statement-retrieval endpoint, where the same broken object-level authorization exploited in Chain 1 (AND-003) discloses cross-account fictional financial data.

THESE ARE FICTIONAL COMPOSITE ATTACK SCENARIOS. They do not describe, and must not be construed as, exploit chains against any real system.

101 BUSINESS IMPACT ANALYSIS

The table below maps each Critical and High fictional finding to the illustrative business dimensions it most directly affects, supporting prioritization conversations beyond pure technical severity.

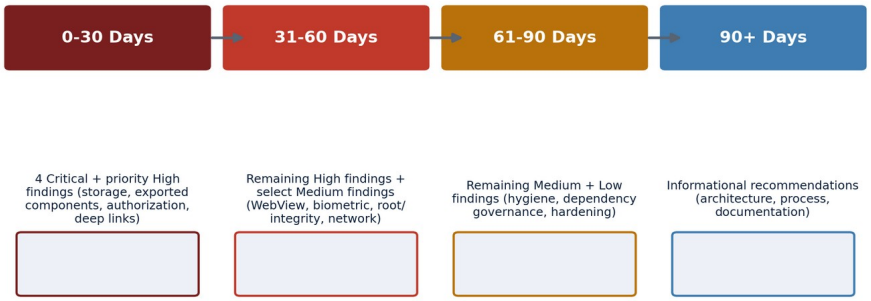
Finding ID	Confidentiality	Integrity	Fraud / Funds Risk	Customer Trust	Privacy
AND-001	High	Medium	High	High	High
AND-002	Medium	High	High	High	Low
AND-003	High	Medium	High	High	High
AND-004	Medium	High	High	High	Low
AND-005	Medium	Low	Low	Medium	Medium
AND-006	Medium	Medium	Medium	Medium	Low
AND-007	Medium	Low	Low	Medium	Medium
AND-008	Medium	Low	Medium	Low	Low
AND-009	Low	Medium	Low	Low	Medium
AND-010	Medium	Medium	Medium	Medium	Low
AND-011	Medium	Low	Medium	Medium	Low
AND-012	Low	Medium	Low	Low	Low
AND-013	Low	Medium	Medium	Low	Low

Illustrative Regulatory & Operational Risk Note

As a fictional Digital Payments / FinTech consumer application, NovaSphere Pay's illustrative regulatory risk considerations would include data-protection and financial-services-adjacent expectations around authentication strength, local data protection, and transaction-authorization integrity. This report does not assess compliance with any specific regulatory framework; the fictional findings above are offered only as illustrative business-impact context.

102 REMEDIATION ROADMAP

The illustrative sample remediation roadmap below sequences all 47 fictional findings across four phases, prioritizing the four Critical findings and the highest-risk High findings first.



Illustrative remediation roadmap aligned to Section 102 (Remediation Roadmap).

ILLUSTRATIVE SAMPLE REMEDIATION ROADMAP — fictional sequencing and dates.

0-30 Days

- Insecure local token storage — AND-001
- Exported component / unauthorized privileged action — AND-002
- Broken authorization in mobile API workflow — AND-003
- Insecure deep-link sensitive account action — AND-004
- WebView JavaScript interface exposure — AND-005
- Insecure FileProvider configuration — AND-009

31-60 Days

- Certificate validation weakness — AND-006
- Sensitive data exposure through logs — AND-007
- Hardcoded API secret — AND-008
- Weak biometric/local authentication enforcement — AND-010
- Token lifecycle weakness — AND-011
- Improper intent handling — AND-012
- Root/runtime integrity control bypass — AND-013

61-90 Days

- Remaining Medium findings — debug configuration, backup configuration, screenshot/clipboard/cache exposure, deprecated TLS, excessive permissions, error handling, session timeout, analytics identifier reuse, random-value generation, tapjacking, SQLite storage, content-provider exposure — AND-014 through AND-027
- Low-severity hardening backlog — unused permissions, deprecated APIs, logging hygiene, dependency governance, exported-component reduction, and related items — AND-028 through AND-035

90+ Days

- Informational recommendations — positive-control formalization, Play Integrity API adoption, expanded obfuscation coverage, secrets-management formalization, step-up attestation, SDK inventory governance, crash-reporting PII review, secure-SDLC checkpoint, accessibility-service review, dependency-scanning cadence, and signing-key documentation — AND-036 through AND-047

ILLUSTRATIVE SAMPLE REMEDIATION ROADMAP — sequencing, owners, and dates are fictional.

103 MANAGEMENT ACTION PLAN

The table below tracks the fictional management response, ownership, and validation approach for all 47 findings, ensuring every finding identified in Sections 94-98 appears here exactly once.

Finding ID	Owner	Priority	Target Date	Status	Validation Method	Remediation (Summary)
AND-001	Mobile Engineering Lead	Immediate — 0-30 Days	10 October 2026	Remediation In Progress	Full or Targeted Retest	Migrate all session/token persistence to EncryptedSharedPre…
AND-002	Mobile Engineering Lead	Immediate — 0-30 Days	08 October 2026	Remediation In Progress	Full or Targeted Retest	Set `android:exported="false"` unless external invocation i…
AND-003	API Platform Lead	Immediate — 0-30 Days	12 October 2026	Remediation In Progress	Full or Targeted Retest	Add a server-side ownership check on every mobile-API route…
AND-004	Mobile Engineering Lead	Immediate — 0-30 Days	15 October 2026	Remediation In Progress	Full or Targeted Retest	Insert a mandatory in-app confirmation screen (and step-up…
AND-005	Mobile Engineering Lead	0-30 Days	31 October 2026	Remediation Planned	Full or Targeted Retest	Remove sensitive methods from the JavaScript interface enti…
AND-006	Mobile Engineering Lead	0-30 Days	31 October 2026	Remediation Planned	Full or Targeted Retest	Remove the custom trust manager and rely on the platform de…
AND-007	Mobile Engineering Lead	0-30 Days	31 October 2026	Remediation Planned	Full or Targeted Retest	Remove or redact sensitive fields at the logging call site…
AND-008	Mobile Engineering Lead	0-30 Days	31 October 2026	Remediation Planned	Full or Targeted Retest	Move the key retrieval server-side (backend-issued, scoped/…
AND-009	Mobile Engineering Lead	31-60 Days	28 November 2026	Remediation Planned	Full or Targeted Retest	Scope `file_paths.xml` to a dedicated, minimal export subdi…
AND-010	Mobile Engineering Lead	0-30 Days	31 October 2026	Remediation Planned	Full or Targeted Retest	Bind the BiometricPrompt result to a CryptoObject wrapping…
AND-011	Identity & Access Lead	0-30 Days	31 October 2026	Remediation Planned	Full or Targeted Retest	Call the token-revocation endpoint synchronously as part of…
AND-012	Mobile Engineering Lead	31-60 Days	28 November 2026	Remediation Planned	Full or Targeted Retest	Remove the custom `taskAffinity` from MfaVerifyActivity (or…
AND-013	Mobile Engineering Lead	31-60 Days	28 November 2026	Remediation Planned	Full or Targeted Retest	Integrate the Play Integrity API (or equivalent current pla…
AND-014	Mobile Engineering Lead	31-60 Days	28 November 2026	Remediation Planned	Full or Targeted Retest	Move debug-menu code behind a compile-time flavor exclusion…
AND-015	Mobile Engineering Lead	31-60 Days	28 November 2026	Remediation Planned	Full or Targeted Retest	Add explicit exclusion rules for all token/credential-beari…
AND-016	Mobile Engineering Lead	31-60 Days	28 November 2026	Remediation Planned	Full or Targeted Retest	Apply `FLAG_SECURE` to all Activities displaying account ba…
AND-017	Mobile Engineering Lead	31-60 Days	28 November 2026	Remediation Planned	Full or Targeted Retest	Mark clipboard writes of sensitive values as sensitive cont…
AND-018	Mobile Engineering Lead	31-60 Days	28 November 2026	Remediation Planned	Full or Targeted Retest	Add `Cache-Control: no-store` to all sensitive financial-da…
AND-019	Mobile Engineering Lead	31-60 Days	28 November 2026	Remediation Planned	Full or Targeted Retest	Explicitly set the minimum accepted TLS protocol version to…
AND-020	Mobile Engineering Lead	31-60 Days	28 November 2026	Remediation Planned	Full or Targeted Retest	Remove `READ_PHONE_STATE` entirely; move the `READ_CONTACTS…
AND-021	API Platform Lead	31-60 Days	28 November 2026	Remediation Planned	Full or Targeted Retest	Standardize all mobile-API error responses on a sanitized e…
AND-022	Identity & Access Lead	31-60 Days	28 November 2026	Remediation Planned	Full or Targeted Retest	Align server-side access-token expiry with the intended idl…
AND-023	Privacy & Data Governance…	31-60 Days	28 November 2026	Remediation Planned	Full or Targeted Retest	Reconfigure the analytics SDK to use the resettable Adverti…
AND-024	Mobile Engineering Lead	31-60 Days	28 November 2026	Remediation Planned	Full or Targeted Retest	Replace `java.util.Random` with `SecureRandom` for all secu…
AND-025	Mobile Engineering Lead	31-60 Days	28 November 2026	Remediation Planned	Full or Targeted Retest	Set `filterTouchesWhenObscured="true"` on all sensitive con…
AND-026	Mobile Engineering Lead	31-60 Days	28 November 2026	Remediation Planned	Full or Targeted Retest	Migrate the local database to an encrypted SQLite implement…
AND-027	Mobile Engineering Lead	31-60 Days	28 November 2026	Remediation Planned	Full or Targeted Retest	Set `android:exported="false"` unless a genuine cross-app i…

Finding ID	Owner	Priority	Target Date	Status	Validation Method	Remediation (Summary)
AND-028	Mobile Engineering Lead	61-90 Days	15 January 2027	Remediation Planned	Full or Targeted Retest	Remove unused permission declarations; add a periodic manif…
AND-029	Mobile Engineering Lead	61-90 Days	15 January 2027	Remediation Planned	Full or Targeted Retest	Remove the deprecated call path and any now-unreachable ver…
AND-030	Mobile Engineering Lead	61-90 Days	15 January 2027	Remediation Planned	Full or Targeted Retest	Reduce verbose logging in release builds to operationally n…
AND-031	Platform Engineering Lead	61-90 Days	15 January 2027	Remediation Planned	Full or Targeted Retest	Integrate a software composition analysis / dependency-scan…
AND-032	Mobile Engineering Lead	61-90 Days	15 January 2027	Remediation Planned	Full or Targeted Retest	Conduct a full exported-component review against documented…
AND-033	Platform Engineering Lead	61-90 Days	15 January 2027	Remediation Planned	Full or Targeted Retest	Build an automated security-regression suite covering, at m…
AND-034	Engineering Director	90+ Days	27 February 2027	Remediation Planned	Full or Targeted Retest	Publish and maintain an Android-specific secure-coding guid…
AND-035	Mobile Engineering Lead	61-90 Days	15 January 2027	Remediation Planned	Full or Targeted Retest	Remove the legacy wildcard entry and replace with specific…
AND-036	N/A	N/A — Positive Observation	N/A	Closed	Full or Targeted Retest	Maintain current MFA enforcement; extend equivalent step-up…
AND-037	N/A	N/A — Positive Observation	N/A	Closed	Full or Targeted Retest	Extend the same pinning and TLS-minimum configuration patte…
AND-038	Mobile Architecture Lead	90+ Days	27 February 2027	Remediation Planned	Full or Targeted Retest	Formally adopt Play Integrity API attestation as a platform…
AND-039	Mobile Engineering Lead	90+ Days	27 February 2027	Remediation Planned	Full or Targeted Retest	Review and narrow `-keep` rules to the minimum required per…
AND-040	Platform Engineering Lead	90+ Days	27 February 2027	Remediation Planned	Full or Targeted Retest	Adopt a build-time secrets-injection process (vault-backed) …
AND-041	Mobile Architecture Lead	90+ Days	27 February 2027	Remediation Planned	Full or Targeted Retest	Define a high-value-transaction policy requiring step-up au…
AND-042	Platform Engineering Lead	90+ Days	27 February 2027	Remediation Planned	Full or Targeted Retest	Establish and maintain a central third-party SDK inventory…
AND-043	Mobile Engineering Lead	90+ Days	27 February 2027	Remediation Planned	Full or Targeted Retest	Configure explicit field-level PII scrubbing rules for the…
AND-044	Engineering Director	90+ Days	27 February 2027	Remediation Planned	Full or Targeted Retest	Introduce a lightweight, mandatory pre-release security che…
AND-045	N/A	N/A — Scope Recommendation for Next Cycle	N/A	Open	Full or Targeted Retest	Scope a dedicated follow-on review of Accessibility Service…
AND-046	Platform Engineering Lead	90+ Days	27 February 2027	Remediation Planned	Full or Targeted Retest	Once SCA tooling is integrated (AND-031), define and docume…
AND-047	N/A	N/A — Scope Recommendation for Next Cycle	N/A	Open	Full or Targeted Retest	Confirm formal documentation exists for signing-key custody…

104 RETEST RECOMMENDATIONS

TMG Security recommends a structured retest following remediation, applying the methodology below. This section describes the recommended retest approach for this fictional engagement; it does not state that remediation has actually occurred.

Recommended Retest Scope

- Full retest of all four Critical findings (AND-001 through AND-004) and all nine High findings (AND-005 through AND-013), including a fresh APK build and repeated static/dynamic analysis pass.
- Targeted retest of each Medium finding against its specific affected component, screen, or API endpoint.
- Confirmation-only review of Low and Informational items, batched into the next general assessment cycle where full retest is not independently warranted.

Retest Methodology

- Evidence Review — confirm that a code or configuration change addressing the finding's root cause has been merged and reflected in a newly built fictional APK.
- Original PoC Reproduction — attempt the original safe PoC steps (static review, adb/dynamic testing, or API request, as applicable) to confirm the previously observed fictional behavior no longer occurs.
- Manifest & Component Re-Review — re-run manifest and exported-component analysis (Appendix D) to confirm no adjacent component regressed.
- Static Re-Analysis — re-decompile the new build and re-run the relevant MASTG-aligned test cases from Appendix A for the affected finding.
- Dynamic Re-Testing — re-run the affected finding's dynamic test on a fresh rooted and non-rooted fictional device pair, consistent with the original test-account register (Appendix I).
- Network Regression — for network/certificate findings (AND-006, AND-019, AND-035), re-validate TLS configuration and pinning behavior through the interception-proxy methodology used in Section 58.
- Resilience Regression — for resilience findings (AND-013), independently re-verify behavior under both root-hiding and runtime-instrumentation conditions, not only a baseline rooted device.
- MASVS Coverage Re-Check — confirm the retested finding's MASVS control now shows a Pass status in an updated Appendix E coverage matrix.
- Closure Criteria — a finding is marked Closed only after reproduction confirms remediation and regression testing raises no new concern.

This retest methodology is illustrative. This report does not claim that any finding has been remediated or retested.

105 POSITIVE SECURITY CONTROLS

This fictional assessment is not intended to be entirely negative. TMG Security identified a number of security controls operating effectively across NovaSphere Pay's Android client and its mobile API integration, summarized below as illustrative fictional observations.

Control Area	Illustrative Observation
Multi-Factor Authentication	MFA is enforced at login and cannot be bypassed by calling the session-issuance endpoint directly without a validated MFA assertion (AND-036).
TLS Enforcement (Primary Client)	The primary API client enforces TLS 1.2+ platform-wide with strong, forward-secret cipher suites and no legacy fallback (AND-037).
Certificate Pinning (Primary Client)	Certificate pinning is correctly implemented with a documented backup pin for the primary payments and authentication endpoints, and correctly terminates the connection on a pinning failure (Section 57).
Room / Type-Bound Serialization	Local persistence uses Room's schema-bound entity mapping and API responses use a strict, type-bound JSON parser, avoiding generic object-deserialization risk (Section 36).
Android Keystore Usage	Cryptographic keys used for local field-level encryption are correctly generated and held inside the Android Keystore, with hardware-backed (TEE) storage on both fictional test devices (Sections 47-48).
Cleartext Traffic Disabled	cleartextTrafficPermitted is correctly set to false at the base Network Security Configuration level with no permissive per-domain override (Section 56).
Business-Logic Enforcement	Transfer limits, idempotency handling, and workflow state-machine sequencing are correctly enforced server-side and resisted manipulation during testing (Section 68).
Rate Limiting	Authentication, MFA-verification, and transfer-confirmation endpoints applied consistent, appropriately scoped rate limiting during this fictional assessment (Section 69).
Account Recovery Controls	Account recovery correctly requires independent out-of-band verification, mandatory MFA re-enrollment, and invalidates prior device-binding state (Section 64).
Object-Level Authorization (Partial)	Server-side account-ownership checks are correctly enforced on the majority of endpoints tested — the confirmed gap is concentrated in the specific funds-transfer workflow detailed in Sections 65 and 67 (AND-003).

All observations above are fictional and were constructed for this illustrative sample assessment.

106 TESTING LIMITATIONS

The following limitations apply to this fictional sample assessment and should be read together with the Important Disclaimer (Section 02).

- NovaSphere Digital Technologies, Inc. and the NovaSphere Pay Android application are fictional; no real organization or mobile application was assessed.
- No real credentials, API keys, signing keys, or production systems were used or accessed.
- No real customer data was accessed at any point.
- No production environment or production backend was tested; only fictional test devices and a fictional controlled assessment environment are represented.
- No destructive testing was performed or simulated as executed, including against the fictional funds-transfer workflow.
- No denial-of-service testing was performed.
- No real Google Play Store distribution channel, code-signing infrastructure, or app-store review process was tested or accessed.
- No social engineering was performed.
- No physical device theft or physical security testing was performed.

- No third-party SDK vendor's backend infrastructure was tested; SDK review was limited to the client-side integration observable within the fictional APK.
- Testing was performed against fictional test devices (rooted and non-rooted) and a fictional emulator profile; results do not represent an exhaustive device-fragmentation matrix.
- All evidence, screenshots, adb/logcat output, decompiled excerpts, and request/response examples in this report are synthetic.

This report describes a fictional, illustrative testing exercise only.

107 APPENDIX A — ANDROID TEST CASE REGISTER

The register below documents all 155 fictional test cases executed during this illustrative assessment, spanning the testing categories presented in Sections 15-92, with a result and cross-reference to any corresponding finding. Every "Exception Identified" result maps to a finding documented in Sections 94-98.

Category	Test Objective	Result	Finding
Manifest	Verify android:debuggable is false in the release manifest merge.	No Exception Identified	—
Manifest	Verify android:allowBackup configuration and backup-rule exclusions for sensitive files.	Exception Identified	AND-015
Manifest	Verify all exported Activities require a documented cross-app use case.	Exception Identified	AND-002
Manifest	Verify all exported Services require a documented cross-app use case and permission gate.	Exception Identified	AND-002
Manifest	Verify all exported Broadcast Receivers require a documented cross-app use case.	No Exception Identified	—
Manifest	Verify all exported Content Providers require a documented cross-app use case.	Exception Identified	AND-027
Manifest	Verify FileProvider path configuration is scoped to the minimum required directory.	Exception Identified	AND-009
Manifest	Verify declared permissions map to an active, current code path.	Exception Identified	AND-028
Manifest	Verify dangerous permissions are requested contextually rather than unconditionally at first run.	Exception Identified	AND-020
Manifest	Verify intent-filter priority values do not unintentionally shadow other apps' handlers.	No Exception Identified	—
Manifest	Verify targetSdkVersion and minSdkVersion reflect current supported platform versions.	No Exception Identified	—
Manifest	Verify signing configuration uses APK Signature Scheme v2/v3.	No Exception Identified	—
Storage	Verify session/access tokens are not stored in plaintext SharedPreferences.	Exception Identified	AND-001
Storage	Verify refresh tokens are not stored in plaintext SharedPreferences.	Exception Identified	AND-001
Storage	Verify device-binding secrets are stored via Keystore-backed encryption.	Exception Identified	AND-001
Storage	Verify the local Room/SQLite database is encrypted at rest.	Exception Identified	AND-026
Storage	Verify HTTP response caching excludes sensitive financial data.	Exception Identified	AND-018
Storage	Verify cache directories are purged of sensitive data on logout.	Exception Identified	AND-018
Storage	Verify external storage is not used for any sensitive application data.	No Exception Identified	—
Storage	Verify clipboard writes of sensitive values are marked sensitive and auto-cleared.	Exception Identified	AND-017
Storage	Verify sensitive data is excluded from Auto Backup / adb backup.	Exception Identified	AND-015
Storage	Verify no sensitive values are hardcoded in application resources or string tables.	Exception Identified	AND-008
Storage	Verify sensitive fields are not retained in memory longer than operationally necessary.	No Exception Identified	—
Storage	Verify temporary files created during statement export are removed after use.	No Exception Identified	—
Storage	Verify keyboard input caching (custom keyboards / autofill) does not capture sensitive fields.	No Exception Identified	—
Storage	Verify sensitive UI content is blanked in the Recent Apps thumbnail.	Exception Identified	AND-016
Cryptography	Verify all sensitive data encryption uses current, strong, non-deprecated algorithms.	No Exception Identified	—
Cryptography	Verify cryptographic keys are generated and stored using the Android Keystore.	No Exception Identified	—
Cryptography	Verify no custom/home-grown cryptographic implementations are used in place of platform primitives.	No Exception Identified	—
Cryptography	Verify security-relevant random values are generated using SecureRandom.	Exception Identified	AND-024
Cryptography	Verify key purposes are not reused across unrelated cryptographic operations.	No Exception Identified	—
Cryptography	Verify IV/nonce values are not reused across encryption operations.	No Exception Identified	—
Cryptography	Verify hardcoded cryptographic keys are absent from the compiled application.	No Exception Identified	—
Cryptography	Verify key rotation and revocation processes are defined for locally generated keys.	No Exception Identified	—
Cryptography	Verify the Keystore-backed key attestation, where used, is validated server-side.	No Exception Identified	—
Cryptography	Verify symmetric encryption uses an authenticated mode (e.g., AES-GCM) rather than unauthenticated CBC.	No Exception Identified	—
Cryptography	Verify cryptographic key material is not logged under any build configuration.	No Exception Identified	—
Authentication	Verify login rejects invalid credentials with a generic, non-enumerable error.	No Exception Identified	—
Authentication	Verify MFA is enforced for all standard login flows.	No Exception Identified	—
Authentication	Verify the MFA-verification screen cannot be bypassed via task-affinity manipulation.	Exception Identified	AND-012
Authentication	Verify biometric app-unlock is bound to a Keystore CryptoObject rather than an app-level boolean.	Exception Identified	AND-010
Authentication	Verify local authentication cannot be bypassed on a rooted/instrumented fictional test device.	Exception Identified	AND-010
Authentication	Verify session/refresh tokens are validated for integrity and expiration server-side.	No Exception Identified	—
Authentication	Verify refresh-token rotation invalidates the prior refresh token on use.	No Exception Identified	—
Authentication	Verify logout triggers server-side token revocation, not only local state clearing.	Exception Identified	AND-011
Authentication	Verify account-recovery flows require equivalent identity assurance to standard login.	No Exception Identified	—
Authentication	Verify sensitive operations (large transfers, new payees) require step-up authentication.	Exception Identified	AND-004
Authentication	Verify session idle-timeout is enforced consistently client-side and server-side.	Exception Identified	AND-022
Authorization	Verify the mobile-API statement endpoint enforces account-ownership authorization.	Exception Identified	AND-003
Authorization	Verify the mobile-API transfer endpoint enforces caller-identity/session validation.	Exception Identified	AND-002
Authorization	Verify low-privilege sessions cannot invoke administrative mobile-API functions.	No Exception Identified	—
Authorization	Verify object-level authorization is consistent across every mobile-API resource type.	No Exception Identified	—
Authorization	Verify authorization is enforced server-side independent of client-side UI restrictions.	No Exception Identified	—
Network	Verify all payments-API traffic is transmitted exclusively over TLS.	No Exception Identified	—
Network	Verify the minimum accepted TLS protocol version is 1.2 or higher.	Exception Identified	AND-019
Network	Verify certificate/public-key pinning is implemented for developer-controlled endpoints.	Exception Identified	AND-006
Network	Verify custom TrustManager implementations perform full certificate chain validation.	Exception Identified	AND-006
Network	Verify custom HostnameVerifier implementations perform proper hostname validation.	No Exception Identified	—
Network	Verify cleartext traffic is disallowed via the Network Security Configuration.	No Exception Identified	—

Category	Test Objective	Result	Finding
Network	Verify Network Security Configuration domain entries are scoped to current usage only.	Exception Identified	AND-035
Network	Verify certificate pinning failure results in connection termination, not silent fallback.	No Exception Identified	—
Network	Verify the app resists a controlled adversary-in-the-middle proxy interception attempt on the payments channel.	No Exception Identified	—
Network	Verify the telemetry channel enforces the same certificate-validation rigor as the payments channel.	Exception Identified	AND-006
Network	Verify redirect responses cannot be used to downgrade a request from HTTPS to HTTP.	No Exception Identified	—
Network	Verify sensitive request/response headers are not logged by any network-layer interceptor.	Exception Identified	AND-007
Network	Verify API responses set Cache-Control: no-store for sensitive financial data.	Exception Identified	AND-018
Network	Verify DNS-level manipulation cannot redirect payments traffic to an unpinned host without detection.	No Exception Identified	—
Platform	Verify TransferService rejects Intents from unauthenticated or unverified callers.	Exception Identified	AND-002
Platform	Verify PayeeLinkActivity requires in-app confirmation before committing a new payee.	Exception Identified	AND-004
Platform	Verify MfaVerifyActivity uses default task affinity.	Exception Identified	AND-012
Platform	Verify HelpContentProvider export scope matches a documented use case.	Exception Identified	AND-027
Platform	Verify NovaSphereFileProvider grants are scoped to the intended single file.	Exception Identified	AND-009
Platform	Verify sensitive Broadcast Receivers validate the sender where the platform allows.	No Exception Identified	—
Platform	Verify PackageReplacedReceiver performs no sensitive action beyond cache re-initialization.	No Exception Identified	—
WebView	Verify addJavascriptInterface methods do not expose sensitive session data.	Exception Identified	AND-005
WebView	Verify WebView navigation is restricted to an allow-listed first-party domain set.	Exception Identified	AND-005
WebView	Verify mixed content loading is disabled in all WebView instances.	Exception Identified	AND-005
WebView	Verify file:// URL loading is disabled in all WebView instances.	No Exception Identified	—
WebView	Verify WebView JavaScript execution is disabled where not strictly required.	No Exception Identified	—
Deep Links	Verify all custom-scheme deep links route through a confirmation step for sensitive actions.	Exception Identified	AND-004
Deep Links	Verify App Links are verified (Digital Asset Links) rather than accepting unverified sources.	No Exception Identified	—
Deep Links	Verify deep-link parameters are treated as untrusted input and validated server-side.	No Exception Identified	—
IPC	Verify Intents received by exported components validate the calling package where feasible.	Exception Identified	AND-002
IPC	Verify sensitive functionality is not reachable via IPC without authorization.	Exception Identified	AND-002
IPC	Verify Intent extras are not trusted for authorization decisions without independent verification.	No Exception Identified	—
Code	Verify third-party dependencies are scanned for known vulnerabilities.	Exception Identified	AND-031
Code	Verify API error responses do not leak stack traces or internal service names.	Exception Identified	AND-021
Code	Verify input validation is applied consistently to all client-supplied deep-link and Intent data.	No Exception Identified	—
Code	Verify deprecated platform APIs are removed once minimum-SDK compatibility no longer requires them.	Exception Identified	AND-029
Code	Verify obfuscation (R8/ProGuard) keep rules are scoped to the minimum required per library.	Exception Identified	AND-039
Code	Verify build-time secrets injection is used in place of hardcoded credentials.	Exception Identified	AND-008
Code	Verify CI includes a security-regression suite covering previously remediated findings.	Exception Identified	AND-033
Code	Verify static-analysis tooling flags logging calls referencing credential-named variables.	Exception Identified	AND-007
Code	Verify a documented Android secure-coding guideline is maintained for the engineering team.	Exception Identified	AND-034
Code	Verify unused/legacy code paths (e.g., deprecated device-ID lookups) are removed on a regular cadence.	Exception Identified	AND-029
Resilience	Verify root-detection heuristics are supplemented by a platform attestation service.	Exception Identified	AND-013
Resilience	Verify root-detection cannot be bypassed by a commodity root-hiding tool in fictional testing.	Exception Identified	AND-013
Resilience	Verify anti-debugging detection is present and triggers an appropriate defensive response.	No Exception Identified	—
Resilience	Verify emulator-detection signals are considered as part of device-risk scoring.	No Exception Identified	—
Resilience	Verify tamper/repackaging detection validates the application signing certificate at runtime.	No Exception Identified	—
Resilience	Verify runtime integrity checks detect common hooking-framework artifacts in fictional testing.	No Exception Identified	—
Resilience	Verify sensitive confirmation controls reject touches while obscured by an overlay.	Exception Identified	AND-025
Resilience	Verify the debug menu is compiled out of any build variant that could reach distribution.	Exception Identified	AND-014
Resilience	Verify anti-static-analysis obfuscation coverage extends to all security-relevant classes.	Exception Identified	AND-039
Resilience	Verify file-integrity checks detect modification of the classes.dex / native library set.	No Exception Identified	—
Resilience	Verify the app degrades gracefully (rather than silently) when integrity checks fail.	No Exception Identified	—
Resilience	Verify signing-key custody and rotation procedures are formally documented.	Exception Identified	AND-047
Resilience	Verify Play Integrity API (or equivalent) attestation is considered for the transaction-authorization architecture.	Exception Identified	AND-041
Privacy	Verify analytics/tracking identifiers respect Advertising ID reset signals.	Exception Identified	AND-023
Privacy	Verify only permissions with a genuine, current feature need are requested.	Exception Identified	AND-020
Privacy	Verify the app's data-collection disclosures match observed runtime SDK behavior.	No Exception Identified	—
Privacy	Verify crash-reporting breadcrumbs are scrubbed of PII by explicit configuration.	Exception Identified	AND-043
Privacy	Verify a maintained third-party SDK inventory documents each SDK's data-access scope.	Exception Identified	AND-042
Privacy	Verify users can access an in-app data-usage/privacy disclosure consistent with MASVS-PRIVACY-3.	No Exception Identified	—
Privacy	Verify users have a mechanism to request account/data deletion consistent with MASVS-PRIVACY-4.	No Exception Identified	—
Privacy	Verify unused dangerous permissions are removed rather than retained "just in case".	Exception Identified	AND-020
Privacy	Verify location data is not retained longer than the branch-locator session requires.	No Exception Identified	—
Privacy	Verify contact data (where granted) is not transmitted to analytics or telemetry endpoints.	No Exception Identified	—
Privacy	Verify notification content previews do not display full sensitive transaction detail on the lock screen.	No Exception Identified	—
Privacy	Verify third-party SDKs do not silently expand their data collection beyond their documented scope.	No Exception Identified	—
API Integration	Verify the mobile-API statement endpoint enforces object-level authorization.	Exception Identified	AND-003
API Integration	Verify the mobile-API transfer endpoint independently re-verifies caller identity.	Exception Identified	AND-002
API Integration	Verify the support-WebView session-token endpoint issues narrowly scoped, short-lived tokens.	Exception Identified	AND-005
API Integration	Verify the statement-export endpoint applies rate limiting to prevent bulk enumeration.	No Exception Identified	—
API Integration	Verify the telemetry ingestion endpoint validates device-token authenticity.	Exception Identified	AND-006
API Integration	Verify the integrity-attestation endpoint rejects replayed or expired attestation payloads.	Exception Identified	AND-013

Category	Test Objective	Result	Finding
API Integration	Verify feature-flag configuration responses do not leak internal-only flag names.	**No Exception Identified**	—
API Integration	Verify client-facing rate-limit status responses do not disclose backend infrastructure detail.	**No Exception Identified**	—
API Integration	Verify all mobile-API endpoints require TLS 1.2+ and reject plaintext HTTP.	**No Exception Identified**	—
API Integration	Verify mobile-API error responses use a sanitized, generic error contract.	**Exception Identified**	AND-021
Code	Verify general UI-layer logging volume is minimized in release builds.	**Exception Identified**	AND-030
Code	Verify navigation/UI-state log statements carry no fingerprinting behavioral detail beyond operational need.	**Exception Identified**	AND-030
Manifest	Verify the full exported-component inventory maps to a documented cross-app use case.	**Exception Identified**	AND-032
Manifest	Verify periodic attack-surface review of exported components is performed each release cycle.	**Exception Identified**	AND-032
Authentication	Verify MFA enforcement has no client-side bypass path across all tested login scenarios.	**Exception Identified**	AND-036
Authentication	Verify MFA coverage extends consistently to deep-link and task-affinity-adjacent flows.	**Exception Identified**	AND-036
Network	Verify the primary payments API client enforces TLS 1.2+ and certificate pinning platform-wide.	**Exception Identified**	AND-037
Network	Verify TLS/pinning configuration is consistent across every network client in the application.	**Exception Identified**	AND-037
Resilience	Verify a platform-level integrity-attestation service is evaluated as a shared dependency for sensitive flows.	**Exception Identified**	AND-038
Resilience	Verify integrity-attestation results are consumed uniformly across login, transfer, and payee-change flows.	**Exception Identified**	AND-038
Code	Verify a formal secrets-management/vault process is evaluated for the mobile build pipeline.	**Exception Identified**	AND-040
Code	Verify third-party SDK key issuance is migrated to backend-issued, scoped tokens where supported.	**Exception Identified**	AND-040
Code	Verify a mandatory pre-release security checklist gate is defined for mobile releases.	**Exception Identified**	AND-044
Code	Verify the pre-release checklist covers manifest export, permission, and backup-rule review.	**Exception Identified**	AND-044
Platform	Verify sensitive input fields are reviewed for malicious Accessibility Service interaction risk.	**Exception Identified**	AND-045
Platform	Verify accessibility-service interaction review is scoped for a dedicated follow-on assessment.	**Exception Identified**	AND-045
Code	Verify a minimum dependency-vulnerability scanning cadence is documented once SCA tooling is integrated.	**Exception Identified**	AND-046
Code	Verify scheduled (non-build-time) dependency scans run on a defined recurring cadence.	**Exception Identified**	AND-046
Storage	Verify statement-export temporary files respect the same retention policy as cached statements.	**No Exception Identified**	—
Storage	Verify multi-window / split-screen presentation does not leak sensitive content to an adjacent app view.	**No Exception Identified**	—
Network	Verify certificate pinning configuration is validated automatically as part of the release pipeline.	**No Exception Identified**	—
Authorization	Verify newly added mobile-API routes are checked against the object-level authorization pattern library.	**No Exception Identified**	—
WebView	Verify WebView cookie and cache state is cleared appropriately between distinct help-content sessions.	**No Exception Identified**	—
Resilience	Verify anti-tampering signals are logged (without sensitive detail) for fictional security-monitoring visibility.	**No Exception Identified**	—
Privacy	Verify the in-app privacy disclosure is reviewed whenever a new third-party SDK is integrated.	**No Exception Identified**	—

108 APPENDIX B — APK METADATA REGISTER

This appendix documents the complete fictional APK metadata, requested-permission set, third-party SDK inventory, native library footprint, and suspicious-string findings collected during static analysis (Sections 15-16, 26-31, 37).

Core APK Metadata

Field	Value
APK File Name	NovaSpherePay-release-6.4.2.apk
Package Name	com.novasphere.pay
Version Name / Version Code	6.4.2 (versionCode 6402)
Target SDK Version	35 (Android 15)
Minimum SDK Version	26 (Android 8.0)
Compile SDK Version	35
Supported ABIs	arm64-v8a, armeabi-v7a
APK Size (Fictional Build)	38.6 MB (compressed)
Signing Scheme	APK Signature Scheme v2 + v3 (fictional production certificate; no real fingerprint disclosed)
android:debuggable (Release Flavor)	false (confirmed absent from production manifest merge)
android:allowBackup	true — see AND-015
Multidex Enabled	true
Obfuscation (R8/ProGuard)	Enabled — coverage gaps noted in AND-039
Build Tool	Android Gradle Plugin 8.x (fictional build configuration)

Requested Permissions

Permission	Protection Level	Actively Used?	Notes
android.permission.INTERNET	Normal	Yes	Core network connectivity.
android.permission.ACCESS_NETWORK_STATE	Normal	Yes	Network reachability checks.
android.permission.USE_BIOMETRIC	Normal	Yes	Biometric app-unlock gate — see AND-010.
android.permission.USE_FINGERPRINT	Normal (legacy)	Yes	Legacy biometric API fallback for older fictional OS versions.
android.permission.CAMERA	Dangerous	Yes	Check-deposit / QR-scan feature (contextual prompt).
android.permission.READ_CONTACTS	Dangerous	Partial — see AND-020	Requested unconditionally; used only by optional "pay a friend" feature.
android.permission.READ_PHONE_STATE	Dangerous	No — see AND-020	Requested but no referencing code path found.
android.permission.ACCESS_FINE_LOCATION	Dangerous	Partial — see AND-020	Requested unconditionally; used only by optional branch-locator feature.
android.permission.POST_NOTIFICATIONS	Dangerous (SDK 33+)	Yes	Transaction and security alert notifications.
android.permission.BLUETOOTH	Normal (legacy)	No — see AND-028	No referencing code path found.
android.permission.NFC	Normal	No — see AND-028	No referencing code path found.
android.permission.RECEIVE_BOOT_COMPLETED	Normal	Yes	Re-arms scheduled notification reminders after device restart.

Third-Party SDK Inventory

SDK	Vendor (Fictional)	Purpose	Data Access
Core Payments Networking	Internal fictional module	Payments API client (OkHttp-based)	Payment/account/transfer data
Mapping / Geolocation SDK	Fictional Third-Party Vendor A	Branch-locator map rendering	Coarse device location (feature-scoped)
Analytics / Product Telemetry SDK	Fictional Third-Party Vendor B	Usage analytics, funnel tracking	Device identifier, usage events — see AND-023
Crash Reporting SDK	Fictional Third-Party Vendor C	Crash and ANR reporting	Stack traces, breadcrumb events — see AND-043
Push Notification SDK	Platform Push Service (Fictional)	Transaction/security alert delivery	Push token, notification payload
Biometric Authentication Library	AndroidX Biometric (fictional pinned version)	BiometricPrompt wrapper — see AND-010	None (local authentication only)
Local Database ORM	Room (fictional pinned version) — see AND-026	Local transaction-history cache	Cached transaction records (unencrypted)
Image Loading Library	Fictional Third-Party Vendor D	Remote image/avatar loading	None (image URLs only)

Native Library Footprint

Library	Architecture	Notes
libnovasphere-crypto.so	arm64-v8a / armeabi-v7a	Fictional native helper for select cryptographic operations.
libmapsdk-render.so	arm64-v8a / armeabi-v7a	Third-party mapping SDK native rendering engine.

Suspicious / Notable Static Strings

String (Fictional)	Context
https://api.novasphere-example.com/v3/	Production API base URL (fictional, non-resolving).
https://auth.novasphere-example.com/oauth/	Production auth/OAuth base URL (fictional, non-resolving).
https://staging-api.novasphere-example.com/v3/	Fictional staging endpoint reference retained in the release build string table.
map_sdk_api_key = "NVSP-FICT-MAPKEY-88213"	Hardcoded mapping SDK key — see AND-008.
novasphere://add-payee	Custom deep-link scheme — see AND-004.
TODO: remove debug menu trigger before GA	Fictional residual developer comment referencing the debug menu tracked in AND-014.

109 APPENDIX C — MANIFEST & COMPONENT REGISTER

The register below documents all 18 fictional Android components declared in AndroidManifest.xml, reviewed during Sections 17-25, with export status, notes, and any corresponding finding.

Component	Type	Exported	Notes	Description	Finding
LoginActivity	Activity	false	—	Entry-point login screen.	—
MfaVerifyActivity	Activity	false (launch mode risk)	—	MFA code entry; task-affinity weakness.	AND-012
BiometricGateActivity	Activity	false	—	Biometric app-unlock gate.	AND-010
AccountOverviewActivity	Activity	false	—	Primary account/balance dashboard.	AND-016
CardDetailActivity	Activity	false	—	Card detail view (masked in UI).	AND-016
AccountDetailFragment (hosted)	Fragment	n/a	—	Account-number copy action.	AND-017
TransferConfirmActivity	Activity	false	—	Funds-transfer confirmation screen.	AND-025
PayeeLinkActivity	Activity	true	—	Deep-link payee-addition handler.	AND-004
HelpWebViewActivity	Activity	false	—	In-app support/help WebView.	AND-005
TransferService	Service	true	—	Funds-transfer initiation service.	AND-002
NotificationSyncService	Service	false	android.permission.BIND_JOB_SERVICE	Background notification sync (JobScheduler-based).	—
TokenRefreshService	Service	false	—	Background token-refresh worker.	—
BootCompletedReceiver	Broadcast Receiver	true	—	Re-arms scheduled reminders after boot; no sensitive action performed.	—
NotificationActionReceiver	Broadcast Receiver	false	—	Handles in-notification quick actions.	—
PackageReplacedReceiver	Broadcast Receiver	true	—	Re-initializes local cache after app update; read-only trigger.	—
HelpContentProvider	Content Provider	true	—	Static FAQ/help content (non-sensitive).	AND-027
NovaSphereFileProvider	Content Provider (FileProvider)	true	android:grantUriPermissions	Statement-export file sharing.	AND-009
TransactionSearchProvider	Content Provider	false	signature	Internal search-suggestion provider, signature-restricted.	—

110 APPENDIX D — API ENDPOINT REGISTER

The register below documents all 27 fictional backend API endpoints identified during mobile client traffic analysis (Sections 66-71), with method, authentication requirement, function, and testing result.

Endpoint	Method	Auth	Function	Result	Finding
/v3/mobile/auth/login	**POST**	None	Credential login, token issuance	Clean	—
/v3/mobile/auth/mfa/verify	**POST**	Session (partial)	MFA code verification	Clean	—
/v3/mobile/auth/refresh	**POST**	Refresh Token	Access-token refresh	Clean	—
/v3/mobile/auth/logout	**POST**	Session	Session/token termination	Finding	AND-011
/v3/mobile/auth/biometric/register	**POST**	Session	Biometric key registration	Clean	—
/v3/mobile/accounts	**GET**	Session	Account list for session owner	Clean	—
/v3/mobile/accounts/{accountId}	**GET**	Session	Account detail retrieval	Clean	—
/v3/mobile/accounts/{accountId}/statements	**GET**	Session	Statement retrieval	Finding	AND-003
/v3/mobile/accounts/{accountId}/transactions	**GET**	Session	Transaction history retrieval	Clean	—
/v3/mobile/transfers	**POST**	Session	Funds-transfer initiation	Finding	AND-002
/v3/mobile/transfers/{transferId}	**GET**	Session	Transfer status retrieval	Clean	—
/v3/mobile/payees	**GET**	Session	Payee list retrieval	Clean	—
/v3/mobile/payees	**POST**	Session	Payee creation	Finding	AND-004
/v3/mobile/payees/{payeeId}	**DELETE**	Session	Payee removal	Clean	—
/v3/mobile/profile	**GET**	Session	User profile retrieval	Clean	—
/v3/mobile/profile	**PATCH**	Session	User profile update	Clean	—
/v3/mobile/devices/register	**POST**	Session	Device/push-token registration	Clean	—
/v3/mobile/notifications/preferences	**GET**	Session	Notification preference retrieval	Clean	—
/v3/mobile/support/articles	**GET**	None	Static support-article listing	Clean	—
/v3/mobile/support/session-token	**POST**	Session	Support-WebView session token issuance	Finding	AND-005
/v3/mobile/cards	**GET**	Session	Linked card list retrieval	Clean	—
/v3/mobile/cards/{cardId}/detail	**GET**	Session	Full (client-masked) card detail retrieval	Clean	—
/v3/mobile/statements/{statementId}/export	**GET**	Session	PDF statement export generation	Finding	AND-009
/v3/mobile/telemetry/events	**POST**	Device Token	Telemetry/analytics event ingestion	Finding	AND-006
/v3/mobile/integrity/attest	**POST**	Session	Device-integrity attestation submission	Finding	AND-013
/v3/mobile/config/feature-flags	**GET**	None	Feature-flag configuration retrieval	Clean	—
/v3/mobile/rate-limits/status	**GET**	Session	Client-facing rate-limit status	Clean	—

111 APPENDIX E — MASVS COVERAGE MATRIX

The matrix below maps this fictional assessment's coverage against all 24 current OWASP MASVS controls across the eight MASVS-STORAGE, MASVS-CRYPTO, MASVS-AUTH, MASVS-NETWORK, MASVS-PLATFORM, MASVS-CODE, MASVS-RESILIENCE, and MASVS-PRIVACY categories, consistent with the coverage model introduced in Section 14.

MASVS Control	Control Statement	Tested?	Method	Result	Related Findings
MASVS-STORAGE-1	*The app securely stores sensitive data.*	Yes	Static + dynamic analysis per Sections 26-92	Weaknesses Identified	AND-001, AND-008, AND-015, AND-018, AND-026, AND-040
MASVS-STORAGE-2	*The app prevents leakage of sensitive data.*	Yes	Static + dynamic analysis per Sections 26-92	Weaknesses Identified	AND-007, AND-030
MASVS-CRYPTO-1	*The app employs current strong cryptography and uses it according to industry best practices.*	Yes	Static + dynamic analysis per Sections 26-92	Weaknesses Identified	AND-024
MASVS-CRYPTO-2	*The app performs key management according to industry best practices.*	Yes	Static + dynamic analysis per Sections 26-92	No Exceptions Identified	—
MASVS-AUTH-1	*The app uses secure authentication and authorization protocols and follows the relevant best practices.*	Yes	Static + dynamic analysis per Sections 26-92	Weaknesses Identified	AND-003, AND-011, AND-022, AND-036
MASVS-AUTH-2	*The app performs local authentication securely according to the platform best practices.*	Yes	Static + dynamic analysis per Sections 26-92	Weaknesses Identified	AND-010
MASVS-AUTH-3	*The app secures sensitive operations with additional authentication.*	Yes	Static + dynamic analysis per Sections 26-92	Weaknesses Identified	AND-041
MASVS-NETWORK-1	*The app secures all network traffic according to the current best practices.*	Yes	Static + dynamic analysis per Sections 26-92	Weaknesses Identified	AND-019, AND-035, AND-037
MASVS-NETWORK-2	*The app performs identity pinning for all remote endpoints under the developer's control.*	Yes	Static + dynamic analysis per Sections 26-92	Weaknesses Identified	AND-006
MASVS-PLATFORM-1	*The app uses IPC mechanisms securely.*	Yes	Static + dynamic analysis per Sections 26-92	Weaknesses Identified	AND-002, AND-004, AND-009, AND-012, AND-027, AND-032
MASVS-PLATFORM-2	*The app uses WebViews securely.*	Yes	Static + dynamic analysis per Sections 26-92	Weaknesses Identified	AND-005
MASVS-PLATFORM-3	*The app uses the user interface securely.*	Yes	Static + dynamic analysis per Sections 26-92	Weaknesses Identified	AND-016, AND-017, AND-025, AND-045
MASVS-CODE-1	*The app requires an up-to-date platform version.*	Yes	Static + dynamic analysis per Sections 26-92	Weaknesses Identified	AND-029
MASVS-CODE-2	*The app has a mechanism for enforcing app updates.*	Yes	Static + dynamic analysis per Sections 26-92	No Exceptions Identified	—
MASVS-CODE-3	*The app only uses software components without known vulnerabilities.*	Yes	Static + dynamic analysis per Sections 26-92	Weaknesses Identified	AND-031, AND-033, AND-042, AND-046
MASVS-CODE-4	*The app validates and sanitizes all untrusted inputs.*	Yes	Static + dynamic analysis per Sections 26-92	Weaknesses Identified	AND-021
MASVS-RESILIENCE-1	*The app validates the integrity of the platform.*	Yes	Static + dynamic analysis per Sections 26-92	Weaknesses Identified	AND-013, AND-038
MASVS-RESILIENCE-2	*The app implements anti-tampering mechanisms.*	Yes	Static + dynamic analysis per Sections 26-92	Weaknesses Identified	AND-047
MASVS-RESILIENCE-3	*The app implements anti-static analysis mechanisms.*	Yes	Static + dynamic analysis per Sections 26-92	Weaknesses Identified	AND-039
MASVS-RESILIENCE-4	*The app implements anti-dynamic analysis techniques.*	Yes	Static + dynamic analysis per Sections 26-92	Weaknesses Identified	AND-014
MASVS-PRIVACY-1	*The app minimizes access to sensitive data and resources.*	Yes	Static + dynamic analysis per Sections 26-92	Weaknesses Identified	AND-020, AND-028, AND-043
MASVS-PRIVACY-2	*The app prevents identification of the user.*	Yes	Static + dynamic analysis per Sections 26-92	Weaknesses Identified	AND-023
MASVS-PRIVACY-3	*The app is transparent about data collection and usage.*	Yes	Static + dynamic analysis per Sections 26-92	No Exceptions Identified	—
MASVS-PRIVACY-4	*The app offers user control over their data.*	Yes	Static + dynamic analysis per Sections 26-92	No Exceptions Identified	—

MASVS control identifiers and statements above are drawn from current, publicly verifiable OWASP MASVS material. This report carries no OWASP affiliation, review, or certification.

112 APPENDIX F — MASWE / MASTG TRACEABILITY

The table below traces each of the 62 rows in this appendix from MASVS control, through the specific OWASP MASWE weakness and MASTG test applied, to the testing result, corresponding finding, and evidence reference — the full traceability chain introduced conceptually in Section 14.

MASVS Control	MASWE ID	MASTG Test	Result	Finding	Evidence
MASVS-STORAGE-1	MASWE-0001	MASTG-TEST-0001	Exception Identified	AND-001	EV-AND-001
MASVS-PLATFORM-1	MASWE-0062	MASTG-TEST-0029	Exception Identified	AND-002	EV-AND-003
MASVS-AUTH-1	—	—	Exception Identified	AND-003	EV-AND-005
MASVS-PLATFORM-1	MASWE-0058	—	Exception Identified	AND-004	EV-AND-007
MASVS-PLATFORM-2	MASWE-0071	MASTG-TEST-0033	Exception Identified	AND-005	EV-AND-009
MASVS-NETWORK-2	MASWE-0052	MASTG-TEST-0021	Exception Identified	AND-006	EV-AND-010
MASVS-STORAGE-2	MASWE-0005	MASTG-TEST-0003	Exception Identified	AND-007	EV-AND-011
MASVS-STORAGE-1	MASWE-0004	—	Exception Identified	AND-008	EV-AND-012
MASVS-PLATFORM-1	MASWE-0064	MASTG-TEST-0007	Exception Identified	AND-009	EV-AND-013
MASVS-AUTH-2	MASWE-0020	MASTG-TEST-0018	Exception Identified	AND-010	EV-AND-014
MASVS-AUTH-1	MASWE-0038	—	Exception Identified	AND-011	EV-AND-015
MASVS-PLATFORM-1	MASWE-0057	MASTG-TEST-0029	Exception Identified	AND-012	EV-AND-016
MASVS-RESILIENCE-1	MASWE-0097	MASTG-TEST-0045	Exception Identified	AND-013	EV-AND-017
MASVS-RESILIENCE-4	—	MASTG-TEST-0046	Exception Identified	AND-014	EV-AND-018
MASVS-STORAGE-1	MASWE-0006	MASTG-TEST-0009	Exception Identified	AND-015	EV-AND-019
MASVS-PLATFORM-3	—	—	Exception Identified	AND-016	EV-AND-020
MASVS-PLATFORM-3	—	—	Exception Identified	AND-017	EV-AND-021
MASVS-STORAGE-1	MASWE-0001	MASTG-TEST-0001	Exception Identified	AND-018	EV-AND-022
MASVS-NETWORK-1	—	MASTG-TEST-0021	Exception Identified	AND-019	EV-AND-023
MASVS-PRIVACY-1	—	MASTG-TEST-0024	Exception Identified	AND-020	EV-AND-024
MASVS-CODE-4	—	—	Exception Identified	AND-021	EV-AND-025
MASVS-AUTH-1	—	—	Exception Identified	AND-022	EV-AND-026
MASVS-PRIVACY-2	MASWE-0110	—	Exception Identified	AND-023	EV-AND-027
MASVS-CRYPTO-1	MASWE-0019	—	Exception Identified	AND-024	EV-AND-028
MASVS-PLATFORM-3	—	—	Exception Identified	AND-025	EV-AND-029
MASVS-STORAGE-1	MASWE-0001	MASTG-TEST-0001	Exception Identified	AND-026	EV-AND-030
MASVS-PLATFORM-1	MASWE-0064	MASTG-TEST-0007	Exception Identified	AND-027	EV-AND-031
MASVS-PRIVACY-1	—	MASTG-TEST-0024	Exception Identified	AND-028	EV-AND-032
MASVS-CODE-1	—	—	Exception Identified	AND-029	EV-AND-033
MASVS-STORAGE-2	—	—	Exception Identified	AND-030	EV-AND-034
MASVS-CODE-3	—	—	Exception Identified	AND-031	EV-AND-035
MASVS-PLATFORM-1	—	MASTG-TEST-0029	Exception Identified	AND-032	EV-AND-036
MASVS-CODE-3	—	—	Exception Identified	AND-033	EV-AND-037
—	—	—	Exception Identified	AND-034	EV-AND-038
MASVS-NETWORK-1	—	—	Exception Identified	AND-035	EV-AND-039
MASVS-AUTH-1	N/A	MASTG-TEST-0017	No Exception Identified	AND-036	EV-AND-040
MASVS-NETWORK-1	N/A	MASTG-TEST-0021	No Exception Identified	AND-037	EV-AND-041
MASVS-RESILIENCE-1	N/A	—	No Exception Identified	AND-038	—
MASVS-RESILIENCE-3	N/A	—	No Exception Identified	AND-039	EV-AND-042
MASVS-STORAGE-1	N/A	—	No Exception Identified	AND-040	—
MASVS-AUTH-3	N/A	—	No Exception Identified	AND-041	—
MASVS-CODE-3	N/A	—	No Exception Identified	AND-042	—
MASVS-PRIVACY-1	N/A	—	No Exception Identified	AND-043	EV-AND-043
—	N/A	—	No Exception Identified	AND-044	—
MASVS-PLATFORM-3	N/A	—	No Exception Identified	AND-045	—
MASVS-CODE-3	N/A	—	No Exception Identified	AND-046	—
MASVS-RESILIENCE-2	N/A	—	No Exception Identified	AND-047	—
MASVS-STORAGE-1	MASWE-0003	N/A — no verified device-tested MASTG-TEST identifier applied	No Exception Identified	—	—
MASVS-STORAGE-1	MASWE-0007	MASTG-TEST-0004	No Exception Identified	—	—
MASVS-CRYPTO-2	MASWE-0014	MASTG-TEST-0307	No Exception Identified	—	—
MASVS-AUTH-2	N/A	MASTG-TEST-0017	No Exception Identified	—	—
MASVS-NETWORK-1	N/A	MASTG-TEST-0236	No Exception Identified	—	—
MASVS-NETWORK-2	N/A	MASTG-TEST-0244	No Exception Identified	—	—
MASVS-PLATFORM-2	N/A	MASTG-TEST-0031	No Exception Identified	—	—
MASVS-PLATFORM-2	N/A	MASTG-TEST-0032	No Exception Identified	—	—
MASVS-PLATFORM-2	N/A	MASTG-TEST-0037	No Exception Identified	—	—
MASVS-CODE-4	N/A	MASTG-TEST-0002	No Exception Identified	—	—
MASVS-RESILIENCE-3	N/A	MASTG-TEST-0048	No Exception Identified	—	—
MASVS-RESILIENCE-4	N/A	MASTG-TEST-0047	No Exception Identified	—	—
MASVS-RESILIENCE-4	N/A	MASTG-TEST-0049	No Exception Identified	—	—
MASVS-PLATFORM-1	N/A	MASTG-TEST-0012	No Exception Identified	—	—
MASVS-PLATFORM-3	N/A	MASTG-TEST-0256	No Exception Identified	—	—

113 APPENDIX G — FINDING REGISTER

The complete register of all 47 fictional findings is provided below for traceability against Sections 94-98 and the Management Action Plan (Section 103).

Finding ID	Title	Severity	CVSS	Status	Section
AND-001	Sensitive Authentication Token Exposure Through Insecure Local Storage	CRITICAL	9.1 (Critical) — Illustrative CVSS v3.1-inspired score	Remediation In Progress	§94-98
AND-002	Exported Android Component Enables Unauthorized Privileged Action	CRITICAL	8.8 (Critical) — Illustrative CVSS v3.1-inspired score	Remediation In Progress	§94-98
AND-003	Broken Authorization in Mobile API Workflow	CRITICAL	8.6 (Critical) — Illustrative CVSS v3.1-inspired score	Remediation In Progress	§94-98
AND-004	Insecure Deep-Link Flow Enables Sensitive Account Action	CRITICAL	8.3 (Critical) — Illustrative CVSS v3.1-inspired score	Remediation In Progress	§94-98
AND-005	WebView JavaScript Interface Exposure	HIGH	7.9 (High) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98
AND-006	Certificate Validation Weakness	HIGH	7.7 (High) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98
AND-007	Sensitive Data Exposure Through Logs	HIGH	7.4 (High) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98
AND-008	Hardcoded API Secret in Application Package	HIGH	7.2 (High) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98
AND-009	Insecure FileProvider Configuration	HIGH	7.1 (High) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98
AND-010	Weak Biometric/Local Authentication Enforcement	HIGH	6.9 (High) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98
AND-011	Token Lifecycle Weakness — Missing Refresh-Token Revocation on Logout	HIGH	6.8 (High) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98
AND-012	Improper Intent Handling Enables Intent Spoofing / Task Hijacking Risk	HIGH	6.6 (High) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98
AND-013	Root / Runtime Integrity Control Bypass	HIGH	6.4 (High) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98
AND-014	Debug Information Exposure in Pre-Production Build Configuration	MEDIUM	5.4 (Medium) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98
AND-015	Insecure Backup Configuration (allowBackup Enabled)	MEDIUM	5.3 (Medium) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98
AND-016	Sensitive Data Exposed via Screenshot / Recent-Apps Thumbnail	MEDIUM	5.1 (Medium) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98
AND-017	Sensitive Data Exposure via Clipboard	MEDIUM	4.9 (Medium) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98
AND-018	Weak Cache Handling of Sensitive Response Data	MEDIUM	4.8 (Medium) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98
AND-019	Deprecated TLS Configuration Permitted by Network Security Configuration	MEDIUM	4.7 (Medium) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98
AND-020	Excessive Android Permissions Requested	MEDIUM	4.6 (Medium) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98
AND-021	Verbose Error Handling Reveals Internal Implementation Details	MEDIUM	4.4 (Medium) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98
AND-022	Weak Session Timeout Enforcement	MEDIUM	4.3 (Medium) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98
AND-023	Analytics/Tracking SDK Privacy Weakness — Persistent Identifier Reuse	MEDIUM	4.1 (Medium) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98

Finding ID	Title	Severity	CVSS	Status	Section
AND-024	Insecure Random Number Generation for a Security-Relevant Value	MEDIUM	4.0 (Medium) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98
AND-025	Tapjacking / Overlay Protection Not Implemented on Sensitive Confirmation Screens	MEDIUM	3.9 (Medium) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98
AND-026	Unencrypted Room/SQLite Database Storage of Transaction History	MEDIUM	5.0 (Medium) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98
AND-027	Exported Content Provider Permits Non-Critical Data Read	MEDIUM	4.5 (Medium) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98
AND-028	Unused / Unnecessary Permissions Declared	LOW	3.1 (Low) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98
AND-029	Use of Deprecated Android APIs	LOW	3.0 (Low) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98
AND-030	General Logging Hygiene Weaknesses in Non-Sensitive Modules	LOW	2.9 (Low) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98
AND-031	Third-Party Dependency Governance Gaps	LOW	3.3 (Low) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98
AND-032	Attack Surface Hardening Opportunity — Exported Component Count	LOW	2.8 (Low) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98
AND-033	Missing Security Regression Test Automation	LOW	2.6 (Low) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98
AND-034	Incomplete Secure-Coding Documentation for Mobile Engineering	LOW	2.4 (Low) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98
AND-035	Missing Network Security Configuration Domain Restriction Granularity	LOW	2.7 (Low) — Illustrative CVSS v3.1-inspired score	Remediation Planned	§94-98
AND-036	Positive Control Observation — Multi-Factor Authentication Enforced at Login	INFORMATIONAL	N/A — Informational / positive observation	Closed	§94-98
AND-037	Positive Control Observation — TLS 1.2+ Enforced Platform-Wide on the Primary API Client	INFORMATIONAL	N/A — Informational / positive observation	Closed	§94-98
AND-038	Recommendation — Adopt Play Integrity API as Primary Resilience Signal	INFORMATIONAL	N/A — Informational / forward-looking recommendation	Remediation Planned	§94-98
AND-039	Recommendation — Expand Obfuscation Coverage (R8/ProGuard)	INFORMATIONAL	N/A — Informational / forward-looking recommendation	Remediation Planned	§94-98
AND-040	Recommendation — Formalize Secrets Management for the Mobile Build Pipeline	INFORMATIONAL	N/A — Informational / forward-looking recommendation	Remediation Planned	§94-98
AND-041	Recommendation — Adopt Step-Up Attestation for High-Value Sensitive Transactions	INFORMATIONAL	N/A — Informational / forward-looking recommendation	Remediation Planned	§94-98
AND-042	Observation — Third-Party SDK Inventory Should Be Centrally Tracked	INFORMATIONAL	N/A — Informational observation	Remediation Planned	§94-98
AND-043	Observation — Crash Reporting SDK Configuration Should Be Reviewed for PII Scrubbing	INFORMATIONAL	N/A — Informational observation	Remediation Planned	§94-98
AND-044	Recommendation — Formal Secure-SDLC Checkpoint for Mobile Releases	INFORMATIONAL	N/A — Informational / forward-looking recommendation	Remediation Planned	§94-98
AND-045	Observation — Accessibility Service Interaction Not Explicitly Restricted	INFORMATIONAL	N/A — Informational observation	Open	§94-98
AND-046	Recommendation — Periodic Dependency Vulnerability Scanning Cadence	INFORMATIONAL	N/A — Informational / forward-looking recommendation	Remediation Planned	§94-98
AND-047	Observation — App Signing Key Management Process Warrants Documentation Review	INFORMATIONAL	N/A — Informational observation	Open	§94-98

114 APPENDIX H — EVIDENCE REGISTER

The register below documents all 43 fictional evidence references cited throughout the detailed findings in Sections 94-98, mapped to the finding each item supports. All evidence is synthetic and constructed for this illustrative assessment.

Evidence ID	Type	Description	Finding
EV-AND-001	Local Storage Extraction	fictional shared_prefs extraction, MASTG-TEST-0001 static/dynamic review notes — item 1 of 2 (Sensitive Authentication Token Exposure Through Insecure Local Storage).	AND-001
EV-AND-002	Local Storage Extraction	fictional shared_prefs extraction, MASTG-TEST-0001 static/dynamic review notes — item 2 of 2 (Sensitive Authentication Token Exposure Through Insecure Local Storage).	AND-001
EV-AND-003	Manifest Excerpt	fictional manifest excerpt, adb Intent proof-of-concept log — item 1 of 2 (Exported Android Component Enables Unauthorized Privileged Action).	AND-002
EV-AND-004	Manifest Excerpt	fictional manifest excerpt, adb Intent proof-of-concept log — item 2 of 2 (Exported Android Component Enables Unauthorized Privileged Action).	AND-002
EV-AND-005	Intercepted Network Traffic	fictional intercepted mobile-API traffic capture, account-enumeration test matrix — item 1 of 2 (Broken Authorization in Mobile API Workflow).	AND-003
EV-AND-006	Intercepted Network Traffic	fictional intercepted mobile-API traffic capture, account-enumeration test matrix — item 2 of 2 (Broken Authorization in Mobile API Workflow).	AND-003
EV-AND-007	Manifest Excerpt	fictional manifest intent-filter excerpt, deep-link trigger log — item 1 of 2 (Insecure Deep-Link Flow Enables Sensitive Account Action).	AND-004
EV-AND-008	Manifest Excerpt	fictional manifest intent-filter excerpt, deep-link trigger log — item 2 of 2 (Insecure Deep-Link Flow Enables Sensitive Account Action).	AND-004
EV-AND-009	WebView JavaScript-Interface Capture	fictional WebView JS-interface enumeration and callback capture (WebView JavaScript Interface Exposure).	AND-005
EV-AND-010	TLS Interception Test Log	fictional TLS interception test log, decompiled trust-manager excerpt (Certificate Validation Weakness).	AND-006
EV-AND-011	Logcat Capture	fictional logcat capture, annotated (Sensitive Data Exposure Through Logs).	AND-007
EV-AND-012	Decompiled Resource Excerpt	fictional decompiled strings.xml excerpt (Hardcoded API Secret in Application Package).	AND-008
EV-AND-013	FileProvider Configuration Review	fictional file_paths.xml excerpt, cross-file access test log (Insecure FileProvider Configuration).	AND-009
EV-AND-014	Decompiled Resource Excerpt	fictional runtime-hooking session log, decompiled BiometricGateActivity excerpt (Weak Biometric/Local Authentication Enforcement).	AND-010
EV-AND-015	Token Replay / Validity Test Log	fictional pre/post-logout token-replay test log (Token Lifecycle Weakness — Missing Refresh-Token Revocation on Logout).	AND-011
EV-AND-016	Manifest Excerpt	fictional task-affinity manifest excerpt, task-hijack proof-of-concept log (Improper Intent Handling Enables Intent Spoofing / Task Hijacking Risk).	AND-012
EV-AND-017	Root-Detection Bypass Test Log	fictional root-detection bypass test log (Root / Runtime Integrity Control Bypass).	AND-013
EV-AND-018	Debug-Menu Screen Capture	fictional debug-menu screen capture (Debug Information Exposure in Pre-Production Build Configuration).	AND-014
EV-AND-019	ADB Backup Archive Listing	fictional adb backup archive listing (Insecure Backup Configuration (allowBackup Enabled)).	AND-015
EV-AND-020	Recent-Apps Thumbnail Capture	fictional recent-apps thumbnail capture (Sensitive Data Exposed via Screenshot / Recent-Apps Thumbnail).	AND-016
EV-AND-021	Cross-App Clipboard Read Log	fictional cross-app clipboard read test log (Sensitive Data Exposure via Clipboard).	AND-017
EV-AND-022	Cache Directory Inspection	fictional cache-directory listing and file excerpt (Weak Cache Handling of Sensitive Response Data).	AND-018
EV-AND-023	TLS Handshake Test Log	fictional TLS handshake test log (Deprecated TLS Configuration Permitted by Network Security Configuration).	AND-019
EV-AND-024	Permission Mapping Matrix	fictional permission-to-feature mapping matrix (Excessive Android Permissions Requested).	AND-020
EV-AND-025	API Error Response Capture	fictional error-response capture (Verbose Error Handling Reveals Internal Implementation Details).	AND-021
EV-AND-026	Token Replay / Validity Test Log	fictional token-validity-after-timeout test log (Weak Session Timeout Enforcement).	AND-022
EV-AND-027	Test Log / Analysis Note	fictional analytics payload before/after comparison (Analytics/Tracking SDK Privacy Weakness — Persistent Identifier Reuse).	AND-023
EV-AND-028	Cryptographic Predictability Analysis	fictional nonce predictability analysis (Insecure Random Number Generation for a Security-Relevant Value).	AND-024
EV-AND-029	Overlay Touch-Filtering Test Log	fictional overlay touch-filtering test log (Tapjacking / Overlay Protection Not Implemented on Sensitive Confirmation Screens).	AND-025
EV-AND-030	Database Extraction Screenshot	fictional database extraction and SQLite browser screenshot (Unencrypted Room/SQLite Database Storage of Transaction History).	AND-026
EV-AND-031	Content Provider Query Log	fictional external content-provider query log (Exported Content Provider Permits Non-Critical Data Read).	AND-027
EV-AND-032	Permission Mapping Matrix	fictional permission-to-code cross-reference (Unused / Unnecessary Permissions Declared).	AND-028
EV-AND-033	Test Log / Analysis Note	fictional deprecated-API static-review note (Use of Deprecated Android APIs).	AND-029
EV-AND-034	Test Log / Analysis Note	fictional logging-statement inventory (General Logging Hygiene Weaknesses in Non-Sensitive Modules).	AND-030
EV-AND-035	Test Log / Analysis Note	fictional dependency version audit (Third-Party Dependency Governance Gaps).	AND-031
EV-AND-036	Test Log / Analysis Note	fictional exported-component inventory and use-case cross-reference (Attack Surface Hardening Opportunity — Exported Component Count).	AND-032
EV-AND-037	Test Log / Analysis Note	fictional CI pipeline configuration review notes (Missing Security Regression Test Automation).	AND-033
EV-AND-038	Test Log / Analysis Note	fictional documentation review notes (Incomplete Secure-Coding Documentation for Mobile Engineering).	AND-034
EV-AND-039	Test Log / Analysis Note	fictional network_security_config.xml excerpt (Missing Network Security Configuration Domain Restriction Granularity).	AND-035
EV-AND-040	Test Log / Analysis Note	fictional MFA enforcement test log (Positive Control Observation — Multi-Factor Authentication Enforced at Login).	AND-036
EV-AND-041	Test Log / Analysis Note	fictional TLS/pinning validation test log (Positive Control Observation — TLS 1.2+ Enforced Platform-Wide on the Primary API Client).	AND-037
EV-AND-042	Test Log / Analysis Note	fictional proguard-rules.pro review notes (Recommendation — Expand Obfuscation Coverage (R8/ProGuard)).	AND-039
EV-AND-043	Test Log / Analysis Note	fictional crash-report sample review (Observation — Crash Reporting SDK Configuration Should Be Reviewed for PII Scrubbing).	AND-043

All evidence identifiers and descriptions above are fictional and synthetic — ILLUSTRATIVE / FICTIONAL SAMPLE EVIDENCE.

115 APPENDIX I — TEST ACCOUNTS & DEVICES

The fictional test accounts and devices below were used throughout this assessment. No real passwords are recorded in this report; all credential values are placeholders.

Test Accounts

Fictional Account	Role	Account ID	Credential	Purpose
android.user01@example.test	Standard User	ACCT-559001	*[Fictional Test Credential]*	Primary standard-user account — core flow testing.
android.user02@example.test	Standard User	ACCT-771123	*[Fictional Test Credential]*	Secondary account — cross-account / BOLA isolation testing (AND-003).
android.newuser01@example.test	Standard User (New Enrollment)	ACCT-559099	*[Fictional Test Credential]*	Onboarding, MFA enrollment, and biometric-registration flow testing.
android.attacker-sim01@example.test	Standard User (Simulated Low-Privilege)	ACCT-990001	*[Fictional Test Credential]*	Simulated-attacker account for authorization and deep-link testing.

Test Devices

Fictional Device	Profile	Purpose
Fictional Test Device A	Rooted Android Emulator (API 35)	Static/dynamic storage, root-detection, and Keystore testing.
Fictional Test Device B	Non-Rooted Physical Reference Device (API 34)	Baseline UX and control-validation testing under standard conditions.
Fictional Test Device C	Rooted Physical Reference Device (API 30, min-SDK boundary)	Legacy-OS-version behavior and backward-compatibility testing.
Fictional Proxy Host	Interception Proxy (Burp-style) on Isolated Test Network	Network traffic interception, TLS/pinning, and MITM-resistance testing.

All accounts, devices, identifiers, and credential placeholders above are fictional. No real credentials are recorded anywhere in this report.

116 FINAL CONCLUSION

Illustrative Security Assessment Conclusion

"NovaSphere Pay demonstrates a strong baseline in several areas — enforced multi-factor authentication, TLS 1.2+ and certificate pinning on the primary API client, Keystore-backed key management, and correct object-level authorization on the majority of operations tested — but the fictional assessment identified meaningful local-storage, exported-component, and mobile-API authorization weaknesses, concentrated in four Critical findings, that require immediate, prioritized remediation."

The four Critical and nine High fictional findings in this sample report require immediate attention and should be the organization's first priority, given their concentration around local token storage, exported Android component authorization, deep-link handling, and mobile-API object-level authorization — the classic Android-specific trust boundaries where a mobile client's local sandbox and IPC surface intersect with server-side enforcement. The fourteen Medium findings should be integrated into the standard remediation backlog and addressed within the illustrative 31-60 day window. The eight Low and twelve Informational findings support defense-in-depth, resilience, and mobile-security governance improvements and do not represent an immediate exploitation path in this fictional sample, but should not be indefinitely deferred.

TMG Security's simulated recommendation is that NovaSphere Digital Technologies invest specifically in (1) migrating all locally stored session and refresh-token material to Keystore-backed encrypted storage, (2) applying explicit caller-identity and confirmation checks to every exported Android component, and (3) closing the object-level authorization gap in the mobile API's funds-transfer workflow — the three highest-leverage architectural improvements suggested by this fictional assessment's findings. TMG Security further recommends that NovaSphere Digital Technologies retest all Critical and High findings following remediation, per the methodology in Section 104, before considering this fictional assessment's core concerns resolved. A follow-on assessment cycle — informed by the resilience and privacy recommendations in Sections 80-92 and Appendix G — is recommended within twelve months.

THIS IS A FICTIONAL SAMPLE ASSESSMENT.

117 CONTACT & DISCLAIMER

Cybersecurity | SOC | VAPT | GRC | MDR | Compliance Solutions | Corporate Trainings
Web: www.tmgsec.com **Email:** security@tmgsec.com
USA Office: 117 South Lexington Street, STE 100, Harrisonville, MO 64701
Support: support@tmgsec.com | **Phone:** +1 (816) 793-1929
WhatsApp: +1 (480) 680-0342

Final Disclaimer

FICTIONAL SAMPLE REPORT — NOT AN ACTUAL CLIENT ANDROID PENETRATION TEST

This report was created by TMG Security solely to demonstrate Android application penetration-testing methodology, technical reporting standards, OWASP MASVS/MASTG/MASWE-aligned evidence presentation, and remediation guidance. NovaSphere Digital Technologies, Inc. and the NovaSphere Pay Android application (package com.novasphere.pay) are fictional. No real client environment was assessed. No real Android application, backend API, or production infrastructure was accessed. No real credentials, API keys, signing keys, personal data, production systems, cloud resources, or customer information were accessed. All manifest excerpts, decompiled code, adb/logcat output, network traffic, and screenshots are synthetic. All vulnerabilities, findings, severity ratings, CVSS scores, and remediation actions are illustrative. No real exploitation occurred. This report is an independent illustrative sample and carries no OWASP affiliation, review, or certification.