Characterizing Malware-as-a-Service-Oriented Android Fraud Campaigns Targeting Brazil

José Renan F. Damasceno, Carlos Adriel Sousa Bastos, Francisco Lucas Falcão Pereira, Emanuel Bezerra Rodrigues, Paulo Antonio Leal Rego

[1]Postgraduate Program in Computer Science – Federal University of Ceará (UFC)
Fortaleza – CE – Brazil

joserenandamasceno@alu.ufc.br, adrielbastos76@gmail.com

flfalcaop@gmail.com, {emanuel,paulo}@dc.ufc.br

Abstract. *Android fraud campaigns increasingly exhibit Malware-as-a-Service (MaaS)-oriented operational practices, including reusable delivery infrastructures, staged payload workflows, and backend-supported malware capabilities. This paper presents an exploratory characterization of Brazilian-targeting Android fraud campaigns through the analysis of phishing delivery infrastructures, malware behavior, and operational artifacts. Using OSINT-driven infrastructure discovery, static analysis, DEX-centric reverse engineering, and dynamic execution on physical devices across six real-world Android malware samples, we identified recurring patterns involving fake Google Play-style delivery pages, APK sideloading, PT-BR localized social engineering, Accessibility Services abuse, WebView-supported phishing interactions, staged payload delivery, C2 communication, and anti-analysis mechanisms. The campaigns ranged from phishing-oriented applications to more intrusive malware implementing overlay-capable flows, remote backend communication, and NFC/EMV payment-card interaction capabilities. Cross-sample correlation revealed partially shared protocol logic, reused delivery workflows, and overlapping operational artifacts, suggesting MaaS-oriented development and deployment practices in Brazilian-targeting Android fraud operations.*

1. Introduction

The rapid proliferation of mobile devices has transformed smartphones into one of the primary targets of modern cybercriminal operations [Malik et al. 2026]. With the widespread adoption of mobile banking, instant payment systems, and digital government services, Android malware focused on credential theft and financial fraud has become a persistent and highly profitable threat [Gezer et al. 2019]. In parallel, the operational model behind these threats has evolved significantly due to the professionalization of cybercrime and the consolidation of the *Malware-as-a-Service* (MaaS) ecosystem [Etyang et al. 2025, Patsakis et al. 2025, Paganini 2022].

The MaaS model has substantially reduced the technical barrier required to conduct sophisticated cyberattacks by commoditizing malicious capabilities through service-oriented infrastructures [Patsakis et al. 2025]. In this underground ecosystem, operators provide not only malware payloads, but also delivery infrastructures, command-and-control (C2) panels, technical support, phishing kits, and affiliate-based operational models [Karo-Karo et al. 2023, Schmutz et al. 2024]. Such an operational structure enables

the rapid deployment and customization of campaigns targeting specific victim profiles and regions, frequently employing obfuscation techniques, anti-analysis mechanisms, and socially engineered distribution strategies to evade detection and maximize infection rates [Black and Opacki 2016, Black et al. 2018].

This scenario is particularly relevant in Brazil, where the extensive use of mobile financial services and digital government platforms creates an attractive environment for financially motivated Android malware campaigns. Recent threat intelligence reports have documented operations specifically targeting Brazilian users through phishing pages impersonating the Google Play Store, financial institutions, and government-related services. One reported example involved fraudulent applications impersonating the Brazilian National Social Security Institute (INSS), such as "INSS Reembolso" [Kaspersky Securelist 2026]. Once installed, these malicious applications frequently abuse Android Accessibility Services to automate fraudulent actions, capture credentials, escalate privileges, and execute overlay-based attacks against banking applications [Schmutz et al. 2024, Kaspersky Securelist 2026].

Despite the growing relevance of mobile MaaS operations targeting Brazil, there remains a limited number of academic studies focused on the characterization of these campaigns within the Brazilian ecosystem. Existing literature predominantly emphasizes malware detection, machine learning classification, traffic analysis, or isolated behavioral features [Malik et al. 2026, McElroy 2024]. While these approaches are valuable, they often provide only partial visibility into the broader operational structure of modern fraud campaigns. In particular, approaches centered exclusively on binary-level detection frequently fail to capture how phishing infrastructures, distribution channels, social engineering strategies, malware behavior, and infrastructure artifacts interact as part of coherent operational campaigns.

Furthermore, current research lacks end-to-end analyses connecting malicious Android artifacts to their associated infection chains and fraud ecosystems in regional contexts such as Brazil. This limitation hinders the understanding of how localized social engineering tactics, fraudulent mobile applications, phishing infrastructures, and MaaS operational models combine to sustain large-scale fraud operations targeting Brazilian users.

In this context, the objective of this paper is to characterize Brazilian-targeting Android fraud campaigns that exhibit MaaS-oriented operational characteristics through an exploratory case-study analysis of real-world Android malware samples. More specifically, we investigate how these malicious applications are distributed, which social-engineering mechanisms they employ, what infrastructure artifacts and indicators can be extracted, and how their technical behavior connects to broader campaign-level evidence.

The main contributions of this work are as follows:

- We propose a campaign-oriented methodology linking Android malware artifacts, behavioral evidence, phishing infrastructure, and campaign-level indicators.
- We analyze six Android malware samples associated with Brazilian-targeting MaaS-related campaigns, characterizing delivery workflows, runtime behaviors, network indicators, and anti-analysis mechanisms.
- We identify cross-sample correlations involving reused phishing structures, staged

delivery patterns, and protocol-level similarities across distinct fraud operations.
- We derive a preliminary operational taxonomy covering delivery mechanisms, social-engineering themes, behavioral capabilities, infrastructure patterns, and anti-analysis strategies.

The remainder of this paper presents the background and related work, methodology, case studies, results, limitations, and conclusions.

2. Background

To understand the complexity of modern mobile fraud campaigns, it is necessary to analyze both the business model sustaining these operations and the technical mechanisms employed by malicious artifacts. This section introduces the Malware-as-a-Service (MaaS) ecosystem, Android financial malware, infection chains, and the main evasion and abuse techniques commonly observed in contemporary mobile campaigns.

2.1. Malware-as-a-Service Ecosystem

MaaS is a clandestine business model that enables individuals with limited technical expertise to conduct cybercriminal operations through reusable malware infrastructures and affiliate-based distribution models [Etyang et al. 2025, Patsakis et al. 2025]. Similar to legitimate Software-as-a-Service (SaaS) platforms, MaaS operators provide malicious payloads, command-and-control (C2) infrastructures, phishing kits, and operational support for affiliates responsible for malware distribution [Karo-Karo et al. 2023]. This modular ecosystem allows campaigns to rapidly adapt delivery strategies and social-engineering themes to regional targets, motivating campaign-oriented approaches for analyzing mobile fraud ecosystems.

Accordingly, in the context of this work, a campaign is defined as a coordinated fraud operation composed of phishing infrastructures, malware artifacts, delivery workflows, social-engineering themes, and operational behaviors associated with a common distribution strategy or threat activity. This notion differs from a malware family, which refers to similarities at the binary or code level, and from an operator, which represents the individual or group responsible for managing or distributing malicious infrastructures.

2.2. Android Financial Malware

The Android ecosystem has become a persistent target for financially motivated malware campaigns focused on credential theft and banking fraud [Gezer et al. 2019, Black et al. 2018]. Malicious applications may disguise themselves as legitimate services, including financial platforms, government-related applications, or application-store interfaces, in order to increase infection success rates. Android financial malware may also employ phishing interfaces, runtime permission abuse, overlay techniques, and backend communication mechanisms to support credential theft and fraudulent interactions.

2.3. Infection Chains and Social Engineering

Mobile fraud operations frequently rely on multi-stage infection chains combining phishing infrastructures, malicious application delivery, victim interaction flows, and backend communication stages. In Android ecosystems, these infection chains often involve sideloading workflows in which users are induced to install applications outside official marketplaces through socially engineered content.

2.4. Accessibility Abuse and Overlay Attacks

Android Accessibility Services may be abused to automate user interaction, grant permissions, capture interface content, and support fraudulent actions. Another recurrent technique involves overlay-based attacks in which deceptive interfaces are displayed above legitimate applications to harvest credentials or financial information.

2.5. Anti-analysis and Obfuscation Techniques

To hinder reverse engineering and evade detection, MaaS developers frequently implement obfuscation and anti-analysis protections such as dynamic code loading, runtime decryption, anti-debugging routines, and emulator detection [Black and Opacki 2016, Black et al. 2018]. These mechanisms reduce the effectiveness of automated analysis pipelines and reinforce the need for multi-layered investigation approaches capable of correlating technical artifacts, behavioral evidence, and campaign-level operational indicators.

Although these techniques and operational models are widely discussed in the mobile threat landscape, the academic characterization of campaign-oriented Android MaaS operations targeting regional ecosystems such as Brazil remains limited.

3. Related Work

Existing research on Android malware predominantly focuses on malware detection, behavioral characterization, MaaS ecosystem analysis, and forensic investigation of banking malware campaigns.

Several studies investigate Android malware through static analysis, dynamic execution, machine learning, runtime semantic characterization, traffic analysis, and behavioral similarity techniques [Malik et al. 2026, McElroy 2024, Gezer et al. 2019, Black et al. 2018]. Although these approaches improve malware detection and behavioral understanding, they predominantly focus on isolated technical features or binary-level analysis, providing limited visibility into phishing infrastructures, social-engineering workflows, and campaign-level operational patterns.

Brazilian studies have investigated Android malware behavior, phishing susceptibility, and malware-related network infrastructure, including malware datasets, forged trustworthiness in phishing attacks, and HTTP/DNS services contacted by malware samples [Bragança et al. 2023, Silva et al. 2019, Botacin et al. 2020]. These works primarily address detection, dataset construction, user susceptibility, or network-level mitigation, whereas our study focuses on campaign-level characterization by correlating phishing delivery infrastructures, staged infection workflows, Android malware behavior, and cross-sample operational artifacts.

Several studies additionally analyze MaaS and broader cybercrime-as-a-service ecosystems, highlighting affiliate-based monetization models, reusable infrastructures, and service-oriented operational workflows [Patsakis et al. 2025, Etyang et al. 2025, Karo-Karo et al. 2023, Paganini 2022]. Prior work further suggests that cybercriminal ecosystems frequently exhibit regional operational characteristics shaped by local financial systems, victim profiles, and social-engineering strategies. However, comparatively fewer works investigate how these operational models manifest in Android malware campaigns targeting regional ecosystems such as Brazil.

Recent forensic analyses illustrate how MaaS operational structures are applied in Android banking malware campaigns through accessibility abuse, overlay attacks, staged delivery mechanisms, and phishing infrastructures [Schmutz et al. 2024, Kaspersky Securelist 2026]. Threat intelligence reports additionally document campaigns targeting Brazilian users through fake Google Play pages, financial impersonation, and government-themed fraud applications impersonating the Brazilian National Social Security Institute (INSS), such as "INSS Reembolso" [Kaspersky Securelist 2026].

Table 1 summarizes representative related works selected to cover the main analytical dimensions discussed in this section and positions the contribution of this work relative to existing literature.

Table 1. Comparison with representative related work.

Work	Region	Malware Behavior	Phishing/ Delivery Infra.	Infection Workflow	Cross-Sample Correlation	Operational Taxonomy
Malik et al.	Global	✓	–	–	–	–
Gezer et al.	Global	✓	–	–	–	–
Black et al.	Global	✓	–	–	✓	–
Schmutz et al.	Global	✓	✓	✓	–	–
Patsakis et al.	Global	–	–	–	–	✓
BeatBanker Report	Brazil	✓	✓	✓	–	–
This Work	**Brazil**	✓	✓	✓	✓	✓

As shown in Table 1, existing studies predominantly focus on malware detection, isolated behavioral analysis, or broad MaaS ecosystem discussions. Comparatively fewer works correlate phishing infrastructures, staged infection workflows, localized social-engineering strategies, and malware artifacts as part of coordinated Android fraud campaigns targeting regional ecosystems such as Brazil. This work addresses this gap through a campaign-oriented characterization integrating phishing delivery infrastructures, operational artifacts, behavioral analysis, and cross-sample correlation across real-world Android fraud campaigns targeting Brazilian users.

4. Methodology

This work adopts an exploratory qualitative methodology centered on Android malware case studies associated with Brazilian-targeting mobile fraud campaigns. It combines infrastructure discovery, victim-oriented malware acquisition, dynamic execution, reverse engineering, and artifact correlation to characterize operational aspects commonly observed in mobile MaaS campaigns. Figure 1 summarizes the adopted workflow: yellow denotes discovery/acquisition, green technical analysis, and blue synthesis stages.

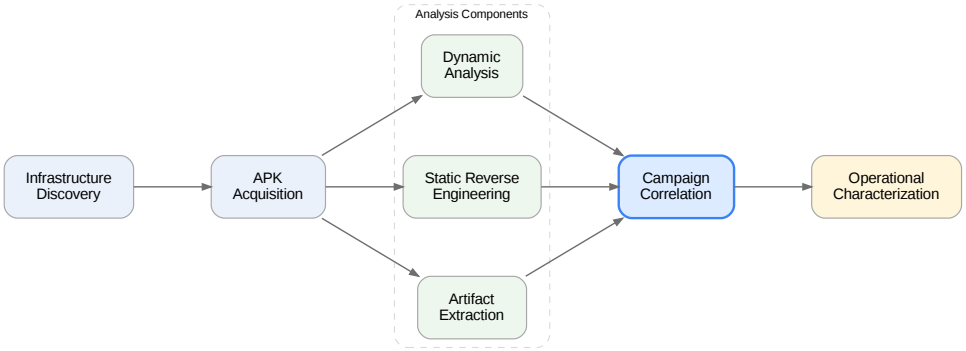

Figure 1. Workflow adopted for campaign-oriented malware analysis.

4.1. Campaign Discovery and Sample Acquisition

Suspicious Android distribution infrastructures targeting Brazilian users were identified through OSINT searches in Shodan and complementary checks in urlscan.io. One representative query, `http.title:"Google Play" country:BR`, returned more than 20 candidate web endpoints for manual inspection.

Candidates were triaged according to their delivery workflows, social-engineering content, and associated Android artifacts. We excluded endpoints that merely distributed betting-related applications outside official marketplaces but lacked sufficient evidence of malicious behavior, phishing functionality, or association with the investigated fraud campaigns. Candidates were retained when they exhibited Brazilian-targeting indicators, such as PT-BR content or references to local services, implemented deceptive delivery workflows resembling Google Play, and allowed APK acquisition through the victim-facing flow.

Following this screening, six Android malware samples were collected between March and May 2026. The acquisition process reproduced the victim-facing infection workflow to preserve operational artifacts. Sample selection followed a purposive exploratory strategy, prioritizing diversity in lure themes, delivery infrastructures, and observed behaviors rather than statistical representativeness.

4.2. Dynamic Analysis Environment

The collected samples were individually executed on a physical Samsung Galaxy A32 device running Android 13 with root access enabled through Magisk 30.7. The physical device was used instead of an automated sandbox or emulator to reduce the impact of emulator-detection mechanisms and to reproduce victim-facing installation and interaction workflows more closely. Application traffic was routed through and monitored using Burp Suite in a controlled laboratory environment.

Each sample was manually exercised through its available execution flow, including installation, permission requests, Accessibility Service prompts, WebView interactions, staged downloads, and background execution when applicable. The analysis recorded contacted endpoints, request and response semantics, downloaded artifacts, visible runtime behavior, and command-and-control indicators. Execution time varied

according to the behavior exposed by each sample and the availability of its remote infrastructure.

Between analysis sessions, the device was cleaned through ADB by uninstalling the analyzed application and any additional packages associated with staged or secondary payloads, together with their application-specific data. This procedure reduced interference between successive executions but did not constitute a complete factory reset.

4.3. Static Analysis and Reverse Engineering

Static analysis and reverse engineering were performed using JADX 1.5.5, apktool 2.7.0-dirty, and MobSF v4.5.0. JADX was used for DEX decompilation and source-level inspection, while apktool supported the extraction of manifests, resources, and DEX artifacts. MobSF was employed as a complementary platform for identifying permissions, exposed components, network indicators, and other security-relevant artifacts.

The analysis inspected requested permissions, declared services and receivers, Accessibility Service implementations, WebView usage, hardcoded strings, URLs, encoded values, native components, and fraud-related behavioral indicators. When automated decompilation was impaired by obfuscation, malformed archive structures, native components, or staged payload delivery, APK contents were manually extracted and the corresponding DEX artifacts were inspected separately. Static findings were then correlated with runtime observations, including network traffic, downloaded payloads, and command-and-control behavior.

4.4. Artifact Extraction and Campaign Correlation

Technical artifacts extracted during static and dynamic analysis were correlated with phishing delivery infrastructures and social-engineering elements to reconstruct operational characteristics associated with the analyzed campaigns.

The correlation process considered relationships between phishing pages, APK distribution mechanisms, WebView-based interfaces, Accessibility Services abuse, network communication patterns, and command-and-control indicators. This multi-layer analysis enabled the reconstruction of infection chains linking malicious delivery infrastructures to runtime malware behavior on Android devices.

Rather than focusing exclusively on binary-level malware detection, the proposed methodology adopts a campaign-oriented perspective aimed at understanding how technical artifacts, delivery strategies, and regionalized social engineering mechanisms interact within Brazilian-targeting mobile fraud operations.

4.5. Ethical Considerations

All samples were analyzed in isolated laboratory environments using dedicated devices and controlled network monitoring. No interaction with real victims, production systems, or operational fraud activities was performed. Infrastructure artifacts are reported only in sanitized or defanged form, and no personally identifiable information, credentials, financial data, or victim-specific content was collected, stored, or processed.

4.6. Analytical Scope and Limitations

This study is exploratory and limited to six Android malware case studies associated with campaigns targeting Brazilian users. Therefore, the findings should not be interpreted

as representative of the entire Brazilian mobile threat landscape. Nevertheless, the analyzed cases exhibited recurring operational characteristics, including phishing-based APK distribution, fake Google Play-style interfaces, Accessibility Services abuse, WebView-based overlays, and MaaS-oriented infrastructure patterns.

A public replication package containing sample hashes, sanitized indicators, methodological documentation, and validation tools is available on Zenodo [Damasceno et al. 2026].

5. Case Study Analysis

This section analyzes six Android malware samples (S1–S6) associated with Brazilian-targeting phishing campaigns. Table 2 summarizes their characteristics, while Appendix A reports sample and secondary-payload hashes.

Table 2. Operational characteristics observed across the analyzed Android malware samples.

Feature	S1	S2	S3	S4	S5	S6
Fake Google Play-style delivery	✓	✓	✓	✓	✓	✓
APK sideloading	✓	✓	✓	✓	✓	✓
PT-BR social engineering	✓	✓	✓	✓	✓	✓
Accessibility Services abuse	✓	✓	✓	–	–	–
WebView usage	✓	✓	–	✓	✓	–
Overlay-oriented behavior	✓	✓	–	–	–	–
Observed C2 communication	✓	✓	✓	✓	–	–
Anti-analysis mechanism	✓	✓	✓	✓	–	–

5.1. Delivery Infrastructure

The investigated campaigns relied on phishing-oriented delivery infrastructures designed to imitate legitimate Android application distribution workflows. During infrastructure discovery, multiple PT-BR phishing pages visually resembling the Google Play Store were identified through OSINT-oriented searches targeting Brazilian users. The analyzed pages reproduced visual branding and fake engagement indicators while employing localized lures associated with financial services, reward platforms, betting applications, and e-commerce-related themes. Representative sanitized infrastructure artifacts associated with the investigated campaigns are reported in Appendix B. In particular, S1 employed multiple rotating phishing subdomains following standardized naming conventions impersonating Google Play-related delivery infrastructures (e.g., `playstoregooglestone-*.edgeone.app`).

Across the investigated cases, the infection workflow relied on APK sideloading outside official Android marketplaces. In some cases, malicious APK downloads were triggered immediately after the victim accessed the phishing page, whereas others required interaction with fake installation buttons simulating legitimate behavior.

Some samples additionally implemented staged payload delivery after execution. In S1 (`1bd70477...`), dynamic analysis showed that the installed APK displayed a fake

update flow and downloaded secondary split-package payloads before exposing more intrusive functionality. Similar update-themed delivery behavior has also been reported for S3 (`bb9c6a6c...`), suggesting reuse of operational delivery patterns across distinct campaigns. Overall, the recurring use of fake Play Store-style interfaces, APK sideloading, and staged update workflows is consistent with operational practices commonly observed in MaaS-oriented mobile fraud ecosystems.

5.2. Behavioral Analysis

Static and dynamic analysis revealed heterogeneous behavioral capabilities across the investigated samples, ranging from relatively simple phishing-oriented applications to more intrusive malware capable of interacting extensively with the Android operating system. As summarized in Table 2, recurring behaviors included Accessibility Services abuse, WebView usage, overlay-oriented functionality, security-relevant permission combinations, and external network communication. A subset of the analyzed samples attempted to obtain Accessibility Services privileges during execution, enabling automated interaction with the Android user interface.

WebView components were recurrent across the investigated campaigns and were used to render remote phishing content and attacker-controlled interaction flows.

The analysis additionally revealed staged-delivery and persistent-background-execution patterns. In S1 (`1bd70477...`), the initially installed APK acted as a first-stage loader, downloaded a secondary payload at runtime, and only then exposed more intrusive capabilities, including Accessibility Services abuse and confirmed C2 communication. Overall, the recurrent combination of APK sideloading, localized social engineering, Accessibility abuse, WebView-supported interaction, overlay-capable behavior, and staged execution demonstrates that the analyzed campaigns relied on operational patterns extending beyond simple standalone phishing applications.

5.3. Network Indicators

In this study, confirmed C2 communication refers to remotely triggered communication or instruction flows involving operational command semantics, telemetry exchange, backend-coordinated behavior, or structured malware-control workflows. This includes both dedicated attacker-controlled endpoints and the abuse of legitimate messaging infrastructures, such as Firebase Cloud Messaging (FCM), when they are used to deliver commands, collect telemetry, or coordinate malware behavior.

Network analysis revealed heterogeneous communication patterns across the investigated samples. Direct evidence of command-and-control activity was observed in S1 (1bd70477...) and S2 (1f333d91...), which contacted the same backend path, `/yaarsa/private/yarsap_80541.php`, and received the identical response value `CO:Sleep` during dynamic execution. Although the samples communicated with different IP addresses, the reuse of the same request path and response semantics suggests shared protocol logic and backend reuse across distinct lures. In S1, the observed C2 traffic originated from a secondary payload dynamically downloaded after execution of the initial APK, whereas in S2 the communication was observed directly from the installed sample.

S3 also exhibited confirmed command-and-control behavior, but through a different transport mechanism. The sample used Google's FCM infrastructure as a C2 transport channel rather than relying on a dedicated attacker-controlled HTTP endpoint. Static analysis revealed a sample-specific service registered for `com.google.firebase.MESSAGING_EVENT`, indicating that the application was designed to receive remote messages through FCM. During dynamic analysis, FCM-mediated communication was observed in association with telemetry collection and runtime decision logic. Incoming messages triggered checks of device-state indicators such as charging status, battery level, battery temperature, recent installation status, user presence, and overheating conditions. These telemetry signals were used to determine whether hidden malicious activity, including cryptocurrency-mining behavior, should be started, delayed, or stopped. Therefore, the observed remote instruction flow and telemetry-driven execution logic provide evidence of FCM-mediated C2 behavior.

S4 exhibited a distinct communication model centered on persistent interaction with a remote backend supporting NFC-oriented payment operations. Static analysis revealed NFC payment-card interaction capabilities involving `NfcAdapter.ReaderCallback` and `IsoDep`, the Android interface for ISO-DEP communication with contactless smart cards. This indicates that the sample was designed to interact with EMV-compatible payment cards through NFC and exchange application-layer messages over custom protocol channels. During dynamic execution, the sample established a stateful session with `http://190.102.41[.]251:8080/` and exfiltrated payment-card information through structured application-layer messages. The observed workflow included bidirectional communication, client-state management, and operational data exfiltration, providing evidence consistent with active command-and-control functionality.

Other samples exposed weaker or less conclusive network indicators. S5 contained a remotely loaded WebView endpoint, but C2 behavior was not confirmed during analysis. No comparably strong network indicator was confirmed for S6.

Overall, confirmed C2 or remote-control behavior was observed in S1, S2, S3, and S4 through distinct transport mechanisms. The overlap between S1 and S2 indicates shared infrastructure logic, while S3 shows that legitimate cloud messaging infrastructures may be abused as C2 transport channels, reinforcing the need to analyze command semantics and runtime effects rather than only dedicated malicious endpoints.

5.4. Anti-analysis

The analyzed samples exhibited different levels of anti-analysis complexity, ranging from conventional code obfuscation to mechanisms intended to hinder static inspection and delay exposure of malicious functionality. As summarized in Table 2, anti-analysis behavior was identified in S1, S2, S3, and S4.

S2 (`1f333d91...`) implemented XOR-based string obfuscation, heavily obfuscated identifiers, and environment-checking logic, increasing the effort required for static inspection and recovery of operational artifacts. S3 (`bb9c6a6c...`) employed malformed ZIP metadata and decoy artifacts that impaired standard Android analysis tools and hindered straightforward static inspection. The sample additionally exposed opaque assets and native components that complicated recovery of the effective payload.

S1 employed staged split-package delivery to defer exposure of its intrusive capabilities until runtime. The initially installed APK downloaded a secondary payload that later exposed Accessibility Services abuse, overlay-capable behavior, and confirmed C2 communication. S4 also used string-level obfuscation through encoded values resolved dynamically before use, increasing the difficulty of recovering operational endpoints and application logic from static artifacts.

Taken together, these mechanisms demonstrate that the analyzed campaigns did not rely solely on social engineering and malicious permissions. Several samples incorporated measures intended to obstruct reverse engineering, fragment the analyst's view of the application, or delay exposure of their intrusive functionality until runtime.

5.5. Campaign Correlation

Although the analyzed samples were distributed through distinct social-engineering themes, several technical and operational similarities suggest that they were not entirely independent implementations. The campaigns reused common delivery patterns involving PT-BR lures, APK sideloading, and deceptive update or installation workflows adapted to different Brazilian-facing services.

The strongest technical correlation was observed between S1 (`1bd70477...`) and S2 (`1f333d91...`). During dynamic analysis, both samples contacted the same C2 path, `/yaarsa/private/yarsap_80541.php`, and received the same response value, `CO:Sleep`, despite using different backend IP addresses. In addition, the secondary payload delivered by S1 exposed a behavioral profile closely aligned with S2, reinforcing evidence of shared protocol logic and malware-component reuse across distinct lures.

Additional overlap was observed at the delivery layer. S1 used a fake update workflow to download a secondary payload after execution, while prior reporting on S3 described a comparable update-themed mechanism for staged delivery. Although this relationship was not experimentally reproduced for S3 in the present study, the similarity suggests reuse of operational delivery patterns across related campaigns.

Other samples exhibited comparatively simpler operational profiles. S5 primarily relied on remotely loaded WebView content, phishing-oriented interaction flows, proprietary backend infrastructure, and betting-themed lures targeting Brazilian users. In contrast, S6 behaved more similarly to a gamified fraud-oriented frontend centered on casino-style interaction flows than to an intrusive Android implant with elevated privileges or persistent command-and-control behavior. Although remote content delivery and backend interaction were observed during dynamic analysis, no persistent command-and-control workflows comparable to those identified in S1, S2, or S4 were confirmed.

Overall, the available evidence supports operational correlation across the sample set, particularly between S1 and S2, while remaining insufficient to attribute all campaigns to a single operator or MaaS provider with certainty. The repeated reuse of delivery patterns, behavioral characteristics, and protocol-level artifacts suggests MaaS-oriented operational practices, in which related tooling and workflows may be adapted to multiple lures and deployment scenarios.

6. Results and Discussion

The analyzed campaigns exhibited heterogeneous operational sophistication ranging from phishing-oriented delivery flows to intrusive malware capable of persistent remote interaction.

Based on a qualitative bottom-up analysis of recurring operational patterns observed across the investigated campaigns, Table 3 presents a preliminary operational taxonomy covering delivery mechanisms, social-engineering themes, behavioral capabilities, infrastructure patterns, and anti-analysis strategies identified in Brazilian-targeting Android MaaS campaigns.

Table 3. Preliminary taxonomy of operational characteristics observed across the analyzed Brazilian-targeting Android fraud campaigns.

Dimension	Categories Observed
Delivery mechanism	Fake Google Play-style delivery, APK sideloading, staged update delivery, split-package payload delivery
Social engineering theme	Financial services, government benefits, betting platforms, reward applications, e-commerce-related lures
Behavioral capability	Accessibility Services abuse, overlay-capable behavior, WebView usage, staged payload execution
Infrastructure pattern	Shared C2 paths, remote backend communication, FCM-related artifacts, remotely loaded WebView content
Anti-analysis strategy	String obfuscation, malformed ZIP structures, staged execution, split-package deployment

A recurring characteristic across all investigated cases was the use of phishing infrastructures impersonating legitimate Android application distribution platforms. As summarized in Table 2, every analyzed sample relied on fake Google Play-style delivery pages, APK sideloading, and PT-BR localized social-engineering strategies. These findings indicate that the investigated campaigns systematically exploited user trust in familiar Brazilian-facing digital services rather than relying exclusively on technical exploitation of the Android operating system.

The analyzed campaigns also demonstrated varying degrees of behavioral sophistication after installation. Simpler samples primarily functioned as phishing-oriented applications, whereas more intrusive samples abused Android Accessibility Services, implemented overlay-oriented interaction flows, and established persistent backend communication channels. This heterogeneity suggests that the analyzed sample set is not composed of a single malware implementation, but rather of multiple operational variants or service configurations adapted to different deployment scenarios.

A particularly notable finding of this study was the presence of technical overlap between distinct campaigns. In particular, S1 and S2 reused the same backend communication path and identical response semantics despite operating through different delivery lures and backend IP addresses. Combined with similarities in behavioral capabilities and staged payload execution, these findings strongly suggest reuse of protocol logic, malware

components, or shared development practices across multiple campaigns. Such overlap is consistent with operational characteristics commonly associated with MaaS-oriented ecosystems.

The investigation additionally showed that some campaigns employed staged infection workflows extending beyond the initial APK installation stage. In S1, the first-stage application downloaded a secondary payload after presenting a fake update flow to the victim. Similar update-oriented workflows were also identified in related reporting involving S3. These findings suggest that phishing infrastructures may operate not only as initial delivery vectors, but also as part of a broader social-engineering pipeline designed to gradually escalate malicious capabilities after user installation.

Another important observation concerns the role of anti-analysis and execution-evasion mechanisms. Multiple samples impaired automated inspection workflows through obfuscation and archive structures consistent with zip bomb-like behavior, requiring manual DEX extraction and fallback reverse-engineering procedures. This demonstrates that even campaigns targeting broad consumer audiences may incorporate defensive mechanisms intended to hinder automated malware triage and static inspection pipelines.

Overall, the analyzed campaigns show how some Brazilian-targeting Android fraud operations can combine phishing infrastructures, localized deception, staged payload delivery, reusable malware components, and remote backend communication into coordinated campaign workflows.

7. Limitations and Threats to Validity

This study has several limitations. First, the analysis was based on a limited set of manually collected Android samples associated with Brazilian-targeting phishing campaigns. Although the samples exhibited recurring technical and operational patterns, they do not comprehensively represent the broader Android fraud ecosystem or all variants distributed through related infrastructures. Moreover, because data collection was restricted to the period between March and May 2026, the study does not support conclusions about seasonal variation, long-term campaign evolution, or the persistence of the identified operational patterns.

Second, dynamic analysis coverage varied across samples. While confirmed command-and-control behavior was directly observed in multiple cases, some samples showed weaker network indicators, and certain behaviors may depend on remote triggers, delayed activation, device state, or backend-side conditions that were not reproducible in the laboratory. Anti-analysis mechanisms, including obfuscation, staged payload delivery, malformed archives, native components, and split-package deployment, also limited static inspection and may have left some capabilities under-characterized.

Third, although FCM-mediated C2 behavior was confirmed in S3, the complete backend-side orchestration described in prior external reporting could not be fully reproduced during the observation period. These findings were therefore interpreted conservatively while supporting the classification of S3 as exhibiting FCM-mediated command-and-control behavior.

Finally, this study focused on campaign-level operational correlation rather than

definitive actor attribution. Shared delivery patterns, protocol reuse, and overlapping malware capabilities indicate technical and operational relatedness, but are insufficient to establish that all campaigns were controlled by the same operator. The conclusions should therefore be interpreted as evidence of ecosystem-level convergence and MaaS-oriented operational practices rather than definitive attribution.

8. Conclusion

This study investigated a set of Android malware campaigns targeting Brazilian users through localized phishing infrastructures and deceptive application-delivery workflows. By combining OSINT-driven infrastructure discovery, static inspection, dynamic execution, and cross-sample comparison, the analysis showed that the investigated campaigns were not limited to isolated phishing applications, but instead exhibited recurring operational patterns consistent with MaaS-oriented operational practices within the analyzed Brazilian-targeting mobile fraud campaigns.

The results demonstrated repeated use of fake Google Play-style delivery flows, APK sideloading, PT-BR social engineering, staged payload deployment, Accessibility Services abuse, WebView-supported interaction, overlay-capable behavior, and remote backend communication. A particularly notable finding was the technical overlap observed between S1 and S2, which reused the same backend path and identical response semantics despite relying on different lures and backend IP addresses. Additional overlap in staged delivery mechanisms and anti-analysis techniques further indicated reuse of development practices and operational templates across distinct campaigns.

These findings highlight the value of campaign-oriented malware analysis. Examining delivery infrastructure, runtime behavior, and protocol-level artifacts together made it possible to identify relationships that would have been less visible through isolated APK inspection alone. For defenders, security vendors, SOC teams, and mobile threat-intelligence analysts, this suggests that detection strategies should incorporate recurring workflow characteristics, infrastructure reuse, staged delivery patterns, backend communication semantics, and monitoring of sideloading-oriented phishing infrastructures rather than relying only on single-sample indicators.

Future work should investigate larger longitudinal datasets in order to evaluate whether the operational patterns identified in this study remain stable across broader campaign populations and extended observation periods. Additional investigation of backend infrastructure clustering, secondary payload evolution, and campaign lifecycle correlation may also help determine whether the proposed operational taxonomy generalizes across other regional Android fraud ecosystems.

Acknowledgments

Carlos Adriel Sousa Bastos acknowledges the financial support provided by Fundação Cearense de Apoio ao Desenvolvimento Científico e Tecnológico (FUNCAP) through a master's scholarship, Process No. BMD-0008-10070.01.02/26.

A. Analyzed Sample Hashes

The analyzed Android malware samples are identified throughout the paper using anonymized identifiers (S1–S6). Table 4 maps the sample identifiers (S1–S6) to their SHA-256 hashes.

Table 4. Analyzed Android malware samples and associated lure themes.

Sample	Lure Theme	SHA-256
S1	Payment-terminal financial lure	1bd7047740a40e48aff1df37b06ad2c16 12e6c50aaaf395a16aed180cf4c296d
S2	Prize-themed financial lure	1f333d910158316c6211f7b24d118aadc a778abb3964f3a4b7d3914e49278b7d
S3	Government-benefit reimbursement fraud	bb9c6a6c84f26f5d98332089f90a4bfa7 35cbcc984a3f49e2ca8124db9d1600f
S4	NFC payment-card operation	173192ec576111adbc0c228f75244cf62 7513c95ca1709036e9850f0cdb7c90b
S5	Betting / casino phishing	ac34dfc7b0ccafec63080336150d7cc3b c832e5925210728b6a0dfabf6b3455c
S6	Betting-themed phishing	48073427089d9377d50fc3ba69da9ef46 7d34c27fa8674291129a880fa64b9dc

Table 5. SHA-256 hashes of secondary payload artifacts dynamically delivered by S1.

Artifact	SHA-256 Hash
base.apk	b7ebb627e3531f074fb85553046dd93d522 fd1b3c8d996ee91b71084cc79d051
split_x.apk	e1cfecf5e4fb0894b294447c1ce646c597a 5cf8506286d3ddb9977a9c7bd7ffc
split_xx.abc.apk	ee0e6670ef2d70e740a8edee161bdce78dc 859f68c32b9a9613be73c8ea15da6

B. Observed Infrastructure Artifacts

Table 6 summarizes the sanitized infrastructure artifacts.

Table 6. Representative sanitized infrastructure artifacts.

Sample	Representative Infrastructure Artifacts
S1	playstoregooglestone-*.edgeone.app
S2	theaphotostudio[.]com
S3	cupomgratisfood[.]shop
S4	seguroprotegido[.]org, missani19[.]shop
S5	pgsoft777dasorte[.]com, ec2-18-228-43-105.sa-east-1.compute.amazonaws[.]com
S6	bossgoogle[.]com, ec2-15-229-145-153.sa-east-1.compute.amazonaws[.]com

References

Black, P., Gondal, I., and Layton, R. (2018). A survey of similarities in banking malware behaviours. *Computers & Security*, 77:756–772.

Black, P. and Opacki, J. (2016). Anti-analysis trends in banking malware. In *2016 11th International Conference on Malicious and Unwanted Software (MALWARE)*, pages 1–7.

Botacin, M., Geus, P. d., and Grégio, A. (2020). An empirical study on the blocking of http and dns requests at providers level to counter in-the-wild malware infections. In *Anais do Simpósio Brasileiro de Cibersegurança (SBSeg)*, pages 188–200. SBC.

Bragança, H., Rocha, V., Barcellos, L. V., Souto, E., Kreutz, D., and Feitosa, E. (2023). Capturing the behavior of android malware with mh-100k: A novel and multidimensional dataset. In *Anais do Simpósio Brasileiro de Cibersegurança (SBSeg)*, pages 510–515. SBC.

Damasceno, J. R. F., Bastos, C. A. S., Pereira, F. L. F., Rodrigues, E. B., and Rego, P. A. L. (2026). Replication Package for Characterizing Malware-as-a-Service-Oriented Android Fraud Campaigns Targeting Brazil. Zenodo. Version 1.0.0. Available at: `https://doi.org/10.5281/zenodo.21767014`.

Etyang, F., Pavithran, P., Mandela, N., and Mwendwa, G. (2025). The evolution and impact of malware-as-a-service (maas) in the dark web: Systematic review. In *Proceedings of the 2025 12th International Conference on Computing for Sustainable Global Development (INDIACom)*, pages 1–7, Delhi, India. IEEE.

Gezer, A., Warner, G., Wilson, C., and Shrestha, P. (2019). A flow-based approach for trickbot banking trojan detection. *Computers & Security*, 84:179–192.

Karo-Karo, G. F. M., Harumnanda, M. S. A., and Lim, C. (2023). Investigating multiple malware as a service (maas): Analysis and prevention techniques. In *2023 IEEE International Conference on Cryptography, Informatics, and Cybersecurity (ICoCICs)*, pages 270–274. IEEE.

Kaspersky Securelist (2026). Beatbanker: A dual-mode android trojan. `https://securelist.com/beatbanker-miner-and-banker/119121/`. Accessed: 2026-05-11.

Malik, S., Singh, N., and Tripathy, S. (2026). Semantic characterization of android malware through runtime system call analysis. *Journal of Information Security and Applications*, 98:104406.

McElroy, S. (2024). Identifying android banking malware through measurement of user interface complexity. In *2024 IEEE International Conference on Cyber Security and Resilience (CSR)*, pages 348–353. IEEE.

Paganini, P. (2022). Cybercrime-as-a-service: Eu perspectives. In Nestoras, A., Martino, L., and Gamal, N., editors, *European Cybersecurity in Context: A Policy-Oriented Comparative Analysis*, pages 67–75. European Liberal Forum.

Patsakis, C., Arroyo, D., and Casino, F. (2025). The malware as a service ecosystem. In Gritzalis, D., Choo, K.-K. R., and Patsakis, C., editors, *Malware: Handbook of Prevention and Detection*, pages 371–394. Springer, Cham.

Schmutz, D., Rapp, R., and Fehrensen, B. (2024). Forensic analysis of hook android malware. *Forensic Science International: Digital Investigation*, 49:301769.

Silva, C. M. R. d., Teixeira, L. C., Barros, J. C. G. d., Feitosa, E. L., and Garcia, V. C. (2019). Suscetibilidade através da forja de fidedignidade: uma abordagem sobre ataques de phishing. In *Anais do Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais (SBSeg)*, pages 139–154. SBC.