Catch Me If You Scan: A Longitudinal Analysis of Stalkerware Evasion Tactics

Anahitha Vijay
Computer Laboratory
University of Cambridge
Cambridge, United Kingdom
av697@cam.ac.uk

Luis A. Saavedra
Computer Laboratory
University of Cambridge
Cambridge, United Kingdom
luis.saavedra@cl.cam.ac.uk

Alice Hutchings
Computer Laboratory
University of Cambridge
Cambridge, United Kingdom
alice.hutchings@cl.cam.ac.uk

***Abstract*—Stalkerware—mobile software that enables covert surveillance, especially in intimate partner relationships—persists as a significant threat on the Android ecosystem despite platform-level policy and security enhancements. We present the first multi-application longitudinal analysis of the stalkerware ecosystem. We analyse 82 APKs from four prominent stalkerware brands sourced from official, third-party, and modded marketplaces, mapping their technical evolution against key policy and OS updates from 2012 to 2025. We find a strategic dichotomy in developer behaviour based on distribution channels. Applications distributed on third-party channels, away from Google Play, consistently target older, less-secure APIs to preserve invasive functionality, effectively ignoring platform policies. In contrast, developers on the Google Play platform respond reluctantly, often employing malicious compliance (e.g., obfuscated notifications) or strategic re-architecting (e.g., 'split-app' models) to circumvent rules while maintaining a market presence. Our findings suggest that platform policies displace rather than eliminate abusive functionality. By systematically documenting how stalkerware developers navigate and subvert platform governance, we provide a nuanced understanding of their adaptive capabilities, offering critical insights for developing more robust, future-proof detection and mitigation strategies.**

***Index Terms*—Android, mobile apps, stalkerware, policy**

I. Introduction

'Stalkerware' remains a persistent vector for technology-facilitated abuse, particularly in the context of intimate partner violence [1]. Mobile phones can provide perpetrators with access to location data, communications, and live camera feeds, allowing for invasive coercion and control over an intimate partner's physical and digital lifestyle. Stalkerware mainly targets the Android ecosystem, due to the ability to 'sideload' apps from third-party sources as opposed to solely downloading apps from a proprietary marketplace like iOS [2]. In response to growing public awareness of the existence of stalkerware and high-profile data breaches of stalkerware vendors [3]–[5], Google has implemented countermeasures on the Google Play platform. In 2018, Google restricted access to sensitive SMS and Call Log permissions within on-platform apps.[1] In 2020, stalkerware was explicitly banned from the Google Play platform.[2]

[1]https://support.google.com/googleplay/android-developer/answer/10208820

[2]https://support.google.com/googleplay/android-developer/answer/10065487

Despite these interventions, stalkerware remains widely distributed and operational. Avast report a 239% global increase in mobile stalkerware encounters from 2020 to 2023, despite ongoing removals of samples from Google Play [6]. While academic research on stalkerware classification and detection has advanced [7], researchers typically analyse contemporary samples in isolation. As a result, little is known about the evolution of stalkerware—particularly how developers adapt their products in response to policy changes or permission restrictions.

To address this gap, we present the first longitudinal analysis of multiple stalkerware families within the Android stalkerware ecosystem. We analyse how stalkerware products evolve across versions and distribution platforms, and how they respond to policies and technical changes in Google Play and the Android ecosystem. By investigating how features are removed, obfuscated, or reintroduced as apps shift between Google Play and less-regulated marketplaces, we generate insights into shared adaptation strategies amongst developers. We address the following research questions:

RQ1: How have stalkerware apps evolved over time in response to changes in Android security features and Google Play policies?

RQ2: How do technical features of stalkerware apps diverge across distribution platforms, namely Google Play, developer websites, and modded markets?

RQ3: What common technical and distribution strategies are shared across different stalkerware families to maintain functionality and evade detection?

We analyse 82 app versions from four prominent stalkerware families sourced from online repositories and two academic datasets [8], [9]. We compare these versions across each app family, mapping technical adaptations and contextual evidence from website archives and developer communications to corroborate developer responses to platform enforcement, and adaptations to key policy and platform milestones. We also compare on-platform (Google Play) and off-platform (sideloaded) variants to chart developers' adaptive strategies and investigate technical convergence across competing products. We identify features that are stripped, hidden, or enhanced, illustrating how developers adapt to platform constraints. Lastly, we explore whether recurring technical behaviours

979-8-3315-8969-1/25/31.00

and evasion techniques exist across families by checking code reuse, parallel feature development between families, and the existence of shared backend infrastructure to identify overlaps that may indicate common development practices, coordinated strategies, or competitive mimicry. We find that developers operating off-platform, away from Google Play's oversight, consistently engage in strategic stagnation, targeting older, less-secure APIs to preserve invasive functionality and effectively ignore platform policies. In contrast, developers maintaining a presence on Google Play employ tactics of 'malicious compliance', such as exploiting developer policy loopholes to circumvent platform rules while maintaining a market presence. Despite operating as commercial competitors with no evidence of shared backend infrastructure, we identified a significant cross-family convergence on core technical features for persistence and stealth. Most notably, multiple families independently weaponise Android's Accessibility Services to bypass modern data access restrictions, enabling keylogging and screen-reading capabilities. Our findings indicate that platform policies primarily displace rather than eliminate abusive functionality, and future work is needed on the Google Play platform to behaviourally analyse stalkerware samples to effectively guard users.

In subsequent sections, the user acting as the monitoring party installing the stalkerware is referred to as the "perpetrator". The target being monitored is referred to as the "survivor".

II. Background and Related Work

A. Stalkerware and Intimate Partner Abuse

While technology's role in facilitating abuse has been recognised for over a decade, research focusing specifically on stalkerware is more recent [10]–[12]. Chatterjee et al. provide one of the first examinations of mobile spyware in the context of intimate partner violence, categorising apps as either overt spyware or 'dual-use' tools (e.g., parental monitoring) that can be repurposed for surveillance [13]. Parsons et al. document how vendors promote their tools for intimate partner surveillance [14].

Detecting stalkerware is challenging. Chatterjee et al. found that while antivirus platforms often flag stalkerware, dedicated mobile anti-spyware tools detect fewer than 7% of samples [13]. To address this, Han et al. introduce a semi-supervised learning system that achieved high accuracy in detecting stalkerware from app descriptions and metadata [15]. However, the focus on non-technical features limits detection of code-level adaptations. Liu et al. analyse 14 Android spyware apps, documenting how stalkerware abuses core Android APIs to covertly activate phone cameras, hide app icons, and persist across device reboots [16]. While these studies provide a valuable foundation for identifying stalkerware's technical behaviours, their analyses are static snapshots, not examining how features evolve over time.

B. App Distribution and Evasion Strategies

A recurring challenge in stalkerware analysis is distinguishing legitimate dual-use apps from tools repurposed for abuse. Chatterjee et al. note this line is blurred by developers who actively market their dual-use apps for intimate partner surveillance [13]. Almansoori et al. conduct the first multilingual study of dual-use apps on Google Play, identifying 3 988 apps across 27 countries [17]. Stalkerware features and Google Play search results vary significantly by region and language, and Google's 2020 policy enforcement against intimate partner surveillance search queries was inconsistent.

To evade official oversight, developers often distribute more invasive versions of their apps via their own websites, a practice known as 'sideloading'. A comparison of sideloaded and Google Play-hosted parental control apps found that sideloaded variants are more likely to lack basic safeguards like encryption and privacy policies [18]. Another evasion vector is 'modded' apps—unauthorised or modified versions of legitimate Android apps, often distributed via third-party marketplaces. We use the ModZoo dataset, introduced by Saavedra et al. as the first large-scale collection of such apps from 13 different markets, to analyse stalkerware families appearing in modded forms and how their capabilities differ from official versions [9]. At the time of publication, ModZoo comprised over 146,000 modded apps, with nearly 9% flagged as malicious by VirusTotal, highlighting their potential danger compared to official app versions.

C. Platform and Policy Countermeasures

Google has implemented two major policy updates to prevent stalkerware's publication on Google Play, under the Google Play Developer Programme Policy. In October 2018, Google restricted access to the `READ_CALL_LOG` and `READ_SMS` permissions, limiting these to apps demonstrating a core functional need. Developers were required to justify their use of these sensitive permissions, with non-compliance by January 9th, 2019 resulting in removal. This move received backlash from stalkerware developers [19], suggesting the policy had some material effect on the surveillance features they could publish.

In October 2020, Google explicitly banned stalkerware, requiring legitimate monitoring apps to provide persistent, visible notifications, include disclosures as to data collection in the app description, obtain verifiable user consent, and ensure the app cannot be hidden. Whilst these requirements should have hindered stalkerware developers, Google Play listings continued to host apps with covert surveillance capabilities well after it came into effect [20].

OS-level security enhancements in Android also directly impact stalkerware. Central to Android's security posture are API Levels, integers identifying platform revisions and corresponding to a new version of Android [21]. Developers declare a `targetSdkVersion` in their app's manifest, signalling which API Level the app is designed for. This 'opt-in' model allows Google to introduce breaking changes, as many security enhancements only affect apps targeting the

newer API Level. However, some security enhancements are implemented as system-wide behavioural changes or user-facing controls that apply universally, regardless of an app's `targetSdkVersion`. These are often critical privacy features enforced by the OS itself when running on the respective Android version, representing a more direct and unavoidable challenge for stalkerware. Even if a stalkerware app targets an old API Level, it cannot circumvent system-level protections when running on a newer device. Key OS-level enhancements that could challenge stalkerware functionality include:

- **Android 8 (API 26):** Mandated persistent notifications for foreground services, such as camera or microphone access, making background activity visible.
- **Android 10 (API 29):** Required apps using foreground services for camera or microphone access to declare a specific `foregroundServiceType` in their manifest, preventing developers from obscuring the intent of their used foreground services. Also introduced Scoped Storage, restricting apps to their own media and content directories and preventing broader file system access, potentially hindering data exfiltration.
- **Android 11 (API 30):** Restricted an app's ability to query other installed apps via the `QUERY_ALL_PACKAGES` permission, which is subject to Google Play review before app publication. This could prevent the monitoring of social media activity and the detection of anti-virus software.
- **Android 12 (API 31):** When an app requests the `ACCESS_FINE_LOCATION` permission as well as the `ACCESS_COARSE_LOCATION` permission, the user is now presented with an option to grant either precise or approximate location access within a 10-20 km radius. Stalkerware developers would either need to account for this reduced accuracy or find more sophisticated methods to triangulate a precise location, if a precise location is not granted upon installation.
- **Android 13 (API 33):** Replaced single storage permission `READ_EXTERNAL_STORAGE` that granted access to all media files with separate permissions for image (`READ_MEDIA_IMAGES`), video (`READ_MEDIA_VIDEO`), and audio files (`READ_MEDIA_AUDIO`). While a perpetrator may grant all permissions during installation, this change leaves a clearer trail to a survivor, who could see that the app has specifically been granted access to 'Photos' and 'Videos' from specific sources, rather than a single, ambiguous 'Files and Media' permission.

These policies and features are enforced by (and alongside) Google Play Protect, Google's built-in security service. Following the October 2020 policy, legitimate monitoring apps must now declare an `isMonitoringTool` flag in their manifest, allowing Play Protect to differentiate these tools from stalkerware. However, Play Protect's effectiveness in detecting stalkerware is contested; after building and publishing InstaCam, an experimental spyware app, on the Google Play platform, Hutchinson et al. [22] found the app evaded Play Protect's detection for weeks, remaining downloadable on the official marketplace. The ineffectiveness of such technical enforcement mechanisms allows stalkerware to persist, motivating our analysis of adaptive strategies.

D. AndroZoo and ModZoo Datasets

To gather app versions for analysis, we use two large-scale Android app datasets: AndroZoo [8] and ModZoo [9]. AndroZoo is a continuously maintained dataset that, as of 2024 [23], contains nearly 24 million APKs. AndroZoo contains Android apps from a range of official and alternative marketplaces, including Google Play, Anzhi, AppChina, and F-Droid [8], retaining historical versions of apps spanning over a decade of development. We use AndroZoo to find historical versions of stalkerware apps that are no longer widely available through Google Play, developer websites or other platforms. ModZoo [9] focuses on modded apps, and is a growing dataset currently comprising over 280,000 APKs. ModZoo [9] gathers modded APKs from 13 popular third-party markets, matching them with their closest AndroZoo counterparts using package names and version codes, enabling comparative analysis of modded apps in line with the objectives of this study.

III. Research Methodology

A. Application Selection and Gathering

We began our analysis by grouping apps into 'families' to enable a structured study of their evolution. A family encompasses all temporal and functional variations of a product from a single developer, including platform-specific versions (e.g., a feature-limited Google Play app vs. a fully-featured sideloaded one), 'lite' or trial releases, and rebranded clones designed to evade enforcement.

We gathered a long-list of candidates identified from industry reports (e.g., Kaspersky's The State of Stalkerware [24]), academic literature [16], and the Echap group's Stalkerware Indicators of Compromise repository on GitHub [25]. Echap's repository maintains a list of stalkerware developer websites and associated domains, providing a starting point for identifying active families. We shortened this long-list to four stalkerware families, chosen for their recent activity, data availability, and because each represents a distinct trend within the stalkerware ecosystem:

- **Cerberus:** Selected for its history of leaving and then returning to Google Play [19], marketing a compliant, feature-limited version on the platform while offering more powerful versions on its website. This provides a case study in navigating Google's policies through feature stripping and platform diversification.
- **TrackView:** Chosen for its 'split-app' model, where a compliant 'viewer' app available on Google Play connects to a full-featured, sideloaded surveillance app, highlighting a key evasion strategy of keeping more potent features off-platform.
- **mLite/mSpy:** Active since 2016, mLite is marketed on Google Play as a parental control and monitoring tool.

However, it is developed alongside mSpy, a premium surveillance app associated with poor data security practices and invasive features.
- **TheTruthSpy/PhoneParental:** Included as a prominent family [26] with no Google Play presence, TheTruthSpy serves as a baseline to study adaptation to Android OS changes without direct policy pressure. TheTruthSpy family recently rebranded to PhoneParental, shifting its marketing to appear as a more legitimate parental monitoring tool.

After selecting these families, we located and downloaded their corresponding APKs across multiple versions, distributed across both Google Play and vendor websites to address RQ2. APKs were first searched for in AndroZoo (§II-D), which includes samples from malware repositories such as VirusShare, helping identify older versions removed from official platforms. Public APK archives were also consulted, such as APKMirror and APKPure, as well as developer websites via the Wayback Machine. To address RQ1, gathering versions released immediately before and after the major policy and platform milestones identified in §II-C was prioritised. Minor bug-fix updates were consolidated to retain only significant monthly or quarterly version updates, keeping the dataset rich yet feasible to analyse.

Finally, we use the ModZoo dataset (§II-D), searching for variants of the selected stalkerware families using package names and brand names. Targeted Google search queries were also employed to discover potential variants hosted on platforms not indexed by the ModZoo dataset, such as searching for 'Cerberus premium free' or 'MSpy no ads'. Relevant modded APKs were downloaded directly from the respective marketplace.

B. Static Analysis Methods

Each gathered APK was statically analysed, inspecting its code and resources without execution. The results are used for both temporal analysis within families and horizontal comparison across them. Our analysis combines automated tools and manual inspection. Each APK was decompiled using JADX [27] which converts APKs into human-readable Java source code. This allows inspection of each app's logic and behaviours, making it possible to identify surveillance functions (e.g., background location tracking, data exfiltration) and stealth features (e.g., hidden icons). To complement this, we use the Mobile Security Framework (MobSF), an automated scanner that statically identifies vulnerabilities, dangerous permissions, and insecure coding practices. Combining JADX for deep, contextual code understanding with MobSF for broad, efficient pattern detection ensured a thorough inspection of each app.

1) Permissions and Receivers: Static analysis began by examining the `AndroidManifest.xml` file to identify requested permissions, with special attention paid to those associated with surveillance. Key permissions of interest include:

- `ACCESS_FINE_LOCATION`, `ACCESS_COARSE_LOCATION`: to track the device's physical location.
- `READ_SMS`, `SEND_SMS`: to read and send text messages covertly.
- `RECORD_AUDIO`, `CAMERA`: to enable real-time audio/video surveillance.
- `READ_CALL_LOG`, `READ_CONTACTS`: to access call histories and contacts.
- `SYSTEM_ALERT_WINDOW`, `RECEIVE_BOOT_COMPLETED`: to overlay harmful content or persist after reboots.

This list is not exhaustive, as stalkerware also exploits less obviously dangerous permissions. For instance, `BIND_ACCESSIBILITY_SERVICE` can be used to monitor all UI events, capture keystrokes, and read on-screen messages [16]. Tools like MobSF provide a more context-aware list of potentially harmful permissions and code features based on Android policies and threat intelligence, which we use to identify further features in code. We also examine broadcast receivers, which allow apps to respond to system events like device boots, enabling persistence by silently reactivating surveillance features. We use JADX to trace where and how any identified permissions were invoked in the source code. We manually search for permission-protected API calls, analysing service classes linked to declared receivers, and tracing execution paths from `onReceive()` or `onStartCommand()` methods to understand how the app responds to system events.

2) Network Endpoints, Data Flows and Third-Party Connections: Previous research shows stalkerware often transmits data to remote command-and-control (C2) servers [28]. We extract hardcoded network endpoints using JADX and MobSF to understand external communications and identify any shared infrastructure between families, which could indicate coordinated distribution networks. MobSF provides an initial overview of embedded URLs and IP addresses.

3) Obfuscation, Anti-Analysis Techniques, and Other Points of Interest: We assess APKs for code obfuscation and anti-analysis techniques designed to hinder reverse engineering. We identify obfuscation via non-descriptive class/method names (e.g., `a.a.a.a`), encrypted strings, and reflection; these techniques are frequently concentrated around critical functionality like data exfiltration or C2 communication [16]. Additionally, apps are examined for explicit anti-debugging mechanisms, such as emulator or root detection, which enable more intrusive surveillance features in compromised environments.

C. Dynamic Analysis Methods

To complement static analysis, we dynamically analyse each app's runtime behaviour. All dynamic testing was conducted in a controlled and isolated environment to minimise risk and avoid exposing personal data. Two physical Android devices were used: one rooted and one unrooted, allowing for observation of different runtime behaviours in cases where full functionality may be gated behind root access.

Apps were installed and managed via ADB (Android Debug Bridge), enabling consistent control over the test environment and direct access to app logs and device responses. Network traffic was intercepted and analysed using Burp Suite Community Edition, and the Frida dynamic instrumentation toolkit was used to bypass certificate pinning protections to enable decryption of network communications. For apps requiring paired device interaction, we use BlueStacks emulator instances to simulate a second party.

We use dynamic analysis to detect runtime behaviours associated with surveillance, data exfiltration, and stealth. Key areas of investigation include covert background service initiation, icon hiding, real-time network activity, and the abuse of Accessibility Services or overlays to covertly interact with the device under a legitimate guise. We also identify persistence and stealth mechanisms with dynamic analysis methods. For instance, if an app was designed to relaunch itself after being force-closed or after a device reboot, this behaviour was observed by restarting the virtual device and monitoring which services were reinitialised. Similarly, UI suppression techniques are assessed by checking whether the app removed its icon from the app drawer, disguised itself using a system-like name, or overlaid decoy screens, among other deceptive visual tactics.

D. Contextualising and Evaluating Stalkerware Evolution

To meaningfully evaluate the adaption of stalkerware over time, we consider the broader ecosystem in which these apps operate. This includes, but is not limited to:

- Investigating the legal or organisational structure of each developer or distributor (e.g., registered business names).
- Reviewing timelines of Google Play presence and removals, using the Wayback Machine and app metadata.
- Analysing how apps are marketed over time, including the use of euphemisms such as 'parental control', 'employee monitoring' or 'anti-theft', to understand how developers attempted to comply with or evade policy scrutiny.
- Identifying any documented incidents, takedowns, or controversies involving these apps, as reported in media coverage or public abuse reports.

These sources give us a more comprehensive understanding of stalkerware's development lifecycle.

E. Ethics

We avoid financially supporting or legitimising stalkerware developers by ensuring no paid apps or subscriptions were purchased. This mitigates risks from their often unreliable and exploitative subscription models [2], where users frequently report difficulties cancelling payments. Any app exclusively available behind a paywall is therefore out of scope.

Apps are only collected for analysis, and only used in a controlled, isolated environment. Physical mitigations, such as covering device cameras or spoofing device location, were employed to prevent accidental capture of compromising data.

Furthermore, our research is strictly observational. No penetration testing or vulnerability disclosure is performed. There is no attempt to exploit weaknesses or access backend systems, particularly given the stalkerware industry's history of poor security practices.

IV. Results

We present our findings from the static, dynamic, and contextual analysis of the four stalkerware app families.

A. Overview of Selected Application Families

Across the four families, a total of 82 APKs are analysed—75 official releases and seven modded variants. Full details of all analysed apps are provided in the Appendix (Table I). To quantify the perceived risk of each app family prior to analysis, every official APK is scanned using VirusTotal, with average scores also listed in Table I. With one exception, at least one security vendor flags every family as malicious, suspicious, or Potentially Unwanted Application (PUA). TheTruthSpy is the most widely detected, with an average of 32 out of 66 vendors flagging each sample. Google Play-exclusive mLite was recognised at an average of two out of 66 detections. The only family to entirely evade detection was the modern Cerberus Google Play variant, whose versions were not flagged by any security vendors.

The Cerberus family, marketed as an anti-theft tool, includes features allowing for remote device wiping, location tracking, and covert picture-taking. We analyse 21 versions of Cerberus spanning from initial release in March 2012 to its latest versions in mid-2025. Cerberus was removed from Google Play in 2019 for violating Call and SMS Log policies, transitioning to a sideload-only distribution method via its website. In October 2023, a new, compliant version (starting with `v1.1.2_play`) was re-released onto Google Play, marketed as Cerberus: Anti-theft Alarm. However, the developer maintained a dual-distribution strategy, continuing to offer Cerberus Standard and Cerberus Disguised, which share a version code of 3.8.0 and are positioned on the developer's website as fully-featured alternatives to the `v1.2.6_play` release, providing a direct point of comparison between on-platform and off-platform feature sets. Two modded versions of Cerberus were discovered on third-party marketplaces; their version codes suggest they are based on older releases of the app, prior to its 2023 return to the Google Play.

TheTruthSpy is openly marketed for intimate partner surveillance and designed to covertly exfiltrate a wide range of personal data, including social media messages, location data and call and text logs. Ten versions from the TheTruthSpy family are analysed, from October 2016 to its February 2024 rebrand as PhoneParental. The PhoneParental app is functionally and visually identical to its predecessor, effectively a successive version. The APKs' digital certificates are signed by 'iSpyoo Teams', developers behind a parallel suite of nearly identical but less well-known stalkerware products. A single modded version of the original TheTruthSpy was also found, reporting a version code later than any official release, highlighting the brand's potential circulation on unregulated marketplaces.

The TrackView family is represented by 22 versions spanning July 2018 to June 2025. Unlike other families we analyse, TrackView's core functionality centres on live remote surveillance. Initially, the full, all-in-one TrackView app was available on Google Play; 12 versions are analysed from this period (July 2018–December 2021). Following its removal from Google Play in late 2021, the developer adopted a 'split-app' strategy. This model consists of two components; the limited TrackView Viewer app is available on Google Play and acts as a remote control to monitor other devices. The full-featured TrackView HomeSafe app must be sideloaded from the developer's website, functioning as a surveillance tool when installed on a target device, but also having viewer capabilities, allowing for a direct connection without needing the Google Play Viewer app.[3] In addition, two modded TrackView APKs are discovered on third-party marketplaces.

Lastly, the dataset analysed includes 22 versions from the **mLite/mSpy ecosystem**. This includes 21 versions of mLite, sourced from Google Play and spanning July 2016 to May 2024, and a single version of the full mSpy suite. mSpy has a similar feature set to TheTruthSpy; mLite, however, has a long-standing Google Play presence, also featuring real-time location tracking and app usage monitoring under the context of 'child safety'. The mSpy APK was opportunistically gathered via a publicly accessible QR code linked on the official mSpy website. Additionally, two modded mSpy versions are discovered on third-party markets: one advertised with unlocked premium features for free, and another claiming to track WhatsApp messages.

We categorise the advertised capabilities of each family into four main groups, with a full summary available in Table II (Appendix). These include: **Communication Monitoring** (e.g., call/SMS logging, social media tracking); **Live Surveillance** (e.g., GPS tracking, remote camera/microphone access, keylogging); **Data Exfiltration** (e.g., accessing stored photos, videos, and contacts); and **Stealth/Remote Commands** (e.g., hiding the app icon, remote device wipe/lock). A distinction emerges between the claimed feature sets of on-platform and off-platform apps. Off-platform families such as TheTruthSpy and mSpy advertise the most extensive surveillance capabilities, claiming to monitor nearly every aspect of device activity. In contrast, apps available on Google Play, such as the Cerberus variant, offer a more limited feature set, focusing on location tracking and remote commands framed as anti-theft measures. However, a core set of features is still consistently present across almost all variants, centred on live surveillance and location tracking. This suggests these capabilities form the foundational offering for modern stalkerware. Cerberus is the only family that advertises remote device wiping, aligning with its 'anti-theft' branding. Conversely, comprehensive communication monitoring and data exfiltration are almost exclusively the domain of the commercial stalkerware families mSpy and TheTruthSpy, which are designed for in-depth interpersonal surveillance rather than device recovery. RQ2 (§IV-D1) explores the technical implementation strategies that account for this divergence further.

[3]The developer also offers a third app, TrackView Sender, which must also be sideloaded. However, Sender is a stripped-down version of HomeSafe that only contains camera-related features. As HomeSafe already includes all of the family's off-platform surveillance capabilities, it was selected as the representative sideloaded app for this analysis.

B. Commercial and Operational Models

All four families rely on a paid subscription model, locking their most invasive features behind a paywall. Offerings vary, from limited free tiers (TrackView) to time-bound trials (mLite, Cerberus, TheTruthSpy). While ethical constraints prevent the purchase of these apps for live dynamic testing, our static analysis of the app code allows for a complete examination of all potential capabilities, including paywalled ones. Beyond their payment structures, all families require the perpetrator to gain temporary physical access to the survivor's device for manual installation and configuration. Once installed, the families employ one of two operational models. The most common is the web portal model, used by Cerberus, mSpy, and TheTruthSpy, where the survivor's device transmits data to a remote server for the perpetrator to view via a web dashboard. In contrast, TrackView and mLite use a peer-to-peer model, requiring an app on both the survivor's and perpetrator's phones, which connect directly to each other. Both TrackView apps need to be logged into the same Google account, allowing the Viewer app to remotely access and control the features of the linked HomeSafe app. Conversely, mLite uses a single app installed on both phones; during setup, one device is designated as the 'child' (target) and the other as the 'parent' (monitor).

C. Longitudinal Evolution in Response to Platform and Policy Changes (RQ1)

To address **Research Question 1**, we analyse the technical and contextual changes within each stalkerware family, correlating them to two distinct forms of pressure: explicit policy updates from Google Play and mandatory security enhancements in the Android OS itself.

1) Response to 2018 Call/SMS Log Restrictions: The response to Google's 2018 policy restricting Call Log and SMS permissions was dictated by an app's distribution model. Families operating exclusively off-platform—including TheTruthSpy, the premium mSpy suite, and sideloaded Cerberus versions—ignored the policy entirely. The restrictions directly prompted Cerberus' initial exit from the Google Play to maintain its surveillance capabilities for its off-market users, a strategy evidenced by the fact that these permissions persist in its sideloaded versions to this day.

In contrast, on-platform families adapted, albeit slowly. mLite retained the restricted permissions for months after the January 2019 enforcement date, only removing them in December 2019. Cerberus exemplifies compliance through re-architecture, returning to Google Play as a compliant app. TrackView, which never required the targeted permissions, was unaffected by this policy.

2) Response to 2020 Explicit Stalkerware Policy: Google's 2020 stalkerware policy had a negligible effect on off-platform developers and was inconsistently enforced on Google Play. Off-platform apps like TheTruthSpy, sideloaded Cerberus versions, and the premium mSpy suite universally ignored the policy, consistently failing to declare the mandatory `isMonitoringTool` flag in all analysed versions. This non-compliance extended to Google Play, where every analysed version of the TrackView Viewer and modern Cerberus app was published without the flag, years after the policy took effect. The only family to eventually comply was mLite, which added the flag in a June 2023 release, nearly three years late. Beyond the flag requirement, on-platform apps demonstrated more subtle policy violations. For instance, post-2020 versions of the TrackView Viewer app contained hardcoded links to the non-compliant, full-featured sideloaded app, directly violating the rule in the 2020 policy against linking to non-compliant APKs hosted outside Google Play.

We analyse how all four stalkerware families respond to key OS updates between Android 8 and Android 13, finding the willingness and speed to adapt is dictated by their reliance on the Google Play platform. Off-market apps consistently target older, less secure API levels to preserve their invasive capabilities. This evolutionary divergence is illustrated in Figure 1, which plots each family's `targetSdkVersion` over time along with key features released at each API Level. On-platform families, such as mLite, show a reluctant but steady upward trend to meet Google's API requirements. In contrast, off-market families such as TheTruthSpy exhibit long periods of stagnation, targeting older, less-secure APIs to preserve invasive functionality and avoid modern privacy features. This strategy allows them to maximise their data collection capabilities on less secure devices that permit it. The following subsections provide a detailed technical analysis of the specific adaptations driving these trends.

Android 8 (API 26) — Persistent Notifications: By 2019, most major stalkerware families were targeting API Level 26 or higher and incorporating the required notification code. For example, legacy versions of Cerberus before its Google Play ban made the transition to API level 26 between v3.5.7 (August 2018) and 3.5.9 (December 2018), and mLite followed suit between v2.1.4 (December 2018) and v2.1.5 (February 2019). However, the implementation of these mandatory notifications indicates an intent to obscure rather than inform. Analysis of mSpy (v8.6.0.1, May 2025) shows its persistent notification title is set to the generic and uninformative string, 'Foreground Service', disguising the app as a generic system process and ensuring the notification fails to act as a meaningful warning for most survivors.

Android 10 (API 29) — Scoped Storage: Numerous families, including the TrackView ecosystem and the premium mSpy suite, explicitly added the `requestLegacyExternalStorage = "true"` flag to their manifests after adopting API level 29. This opts out of scoped storage and retains broad, unfettered access to the device's file system. TrackView's older Google Play variant made this transition between v3.5.28-tv (May 2020) and v3.6.29 (September 2020), immediately including the opt-out flag. Conversely, the majority of the off-market families, such as TheTruthSpy and the various sideloaded Cerberus versions, simply never adopted API Level 29 at all.

Android 11 (API 30) — Package Visibility: Responses to restrictions were inconsistent, with the primary workaround, the `QUERY_ALL_PACKAGES` permission seeing sparse adoption. mLite eventually added this in v3.0.8 (August 2022), and due to this feature requiring Google's vetting, this suggests their justification as a child safety and tracking app was a compelling enough case to include it. The only off-market app to utilise the permission was the premium mSpy suite.

Android 12 & 13 (API 31 & 33) — Granular Permissions: Every family targeting API 31 and above correctly requests both `ACCESS_FINE_LOCATION` and `ACCESS_COARSE_LOCATION`, which is the technical prerequisite to trigger the new location granularity-choice dialogue. However, dynamic analysis reveals the apps are not designed to function with the less-permissive choice. If coarse location is selected during setup, the apps either fail to operate entirely (mLite) or enter a loop directing the perpetrator back to the system settings window until the fine location permission is granted (TheTruthSpy). The adoption of the new `READ_MEDIA_IMAGES` and `READ_MEDIA_VIDEO` permissions in Android 13 was only observed in the premium mSpy suite. The remaining families that target API 33 and above do not have functionality that requires them to read or access media already stored on the phone.

Contextual Changes Over Time: A trend observed across the stalkerware ecosystem, particularly following the 2020 policy update, is the pivot towards the more socially and regulatorily acceptable frame of 'parental control'. For example, TheTruthSpy operation has a documented history of security failures and data breaches, including a still-unpatched vulnerability in 2022 (CVE-2022-0732) [29]. By rebranding to PhoneParental, the operators can distance the product from this negative history, attracting a new user-base searching for legitimate parental control apps who would otherwise be deterred by the record of the original brand. The mSpy ecosystem employs a similar strategy; while mSpy is repeatedly characterised as stalkerware in media reports and suffered multiple data breaches [30], mLite, is marketed towards 'family safety' and includes emergency features such as a 'Panic Button', distancing itself from mSpy's negative branding.

D. Platform Adaptations and Feature Divergence (RQ2)

To address **Research Question 2** we compare the feature sets between on and off-platform apps (including modded apps) with their actual implementations, requested permissions, and implementation techniques.

1) Google Play vs. Developer Website Versions: A direct comparison of the latest on-platform (Google Play) and off-platform (sideloaded) versions from the Cerberus, TrackView, and mSpy families indicates a strategy of feature differen-

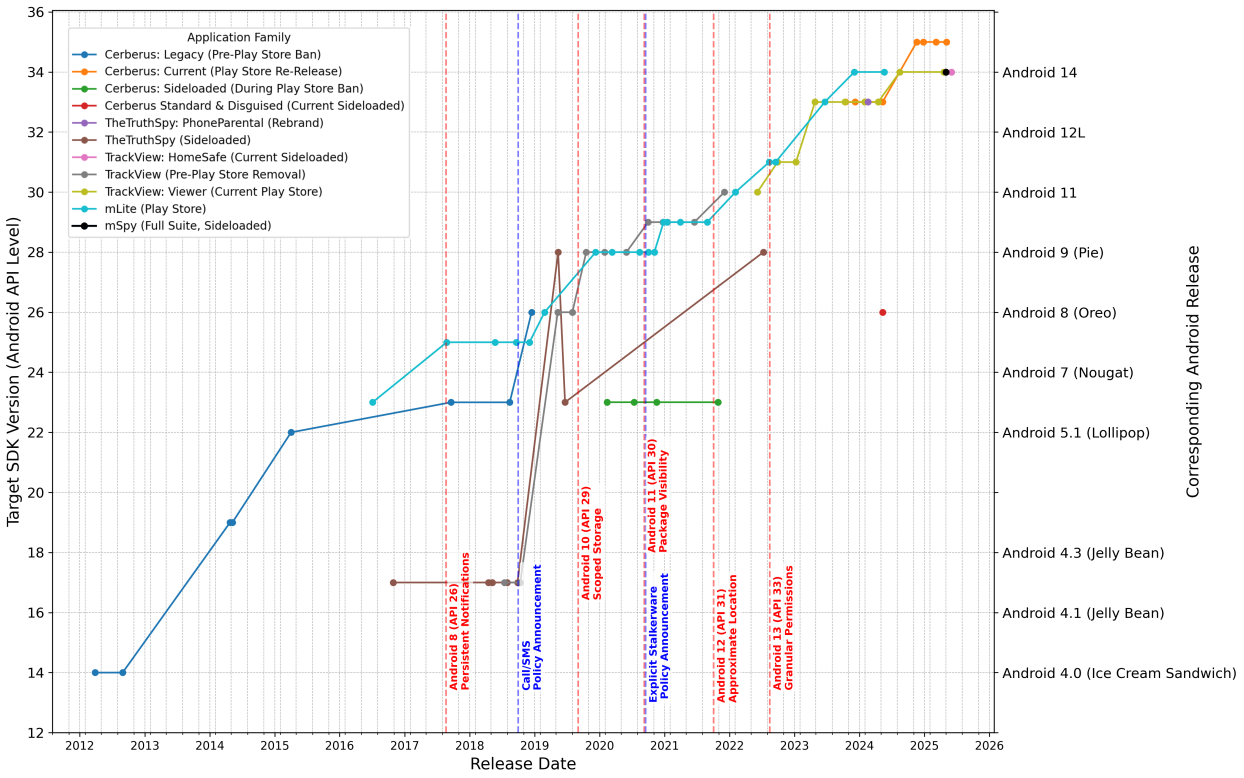

Fig. 1. Timeline of `targetSdkVersion` evolution for each version in each stalkerware family. The plot tracks the targeted API level (left Y-axis) against the release date of a particular app version, with the corresponding Android release name to the API level shown on the right Y-axis. Vertical lines indicate release dates of these key Android OS updates (red) and Google Play policy changes (blue).

tiation tailored to regulatory environment. A full list of the compared APKs is available in Table III (Appendix).

Feature Stripping: Off-platform variants consistently request more permissions than their Google Play counterparts (see Figure 2). Capabilities stripped from on-platform apps are typically those used for direct communication surveillance (e.g., `READ_CALL_LOG`, `READ_SMS`). Permissions used to covertly monitor general activity, such as the `SYSTEM_ALERT_WINDOW` permission, which enables various malicious capabilities such as capturing screenshots without triggering notifications [16], were also generally stripped.

Feature Enhancement and Deception: Beyond stripping features entirely, developers also modify the implementation of shared capabilities to be less overtly malicious. We compare the code snippets of features that exist in both on-platform and off-platform variants, observing a pattern of enhancement within sideloaded versions and restraint within Google Play versions. In the sideloaded 2024 version of Cerberus Disguised, the `com.surebrec.AccService` abuses accessibility service methods to read the content of other apps on screen; the off-platform mSpy app shows similar behaviour. Whilst the on-platform Cerberus version also uses Accessibility APIs, the findings are confined to benign functions like reading the text of its own UI elements (e.g., `chip.getText()`) instead of the content of other apps. The sideloaded Cerberus Disguised version is also designed

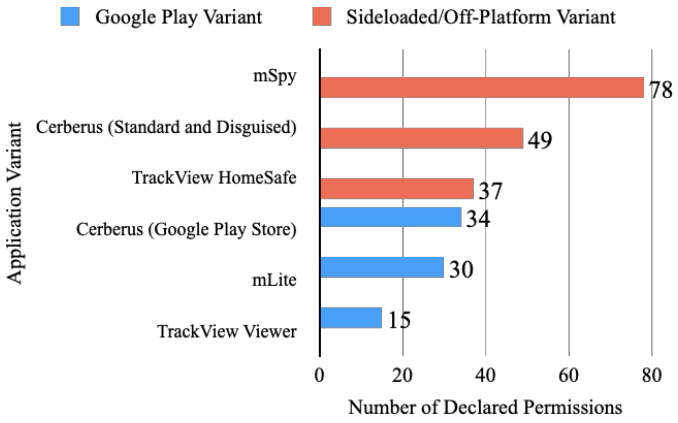

Fig. 2. Number of total permissions requested by application variants, on and off-platform.

to respond to remote commands from its online control panel to call a `TAKEPICTURE` method covertly. The on-platform Cerberus app retains the ability to take a picture, but reframes it as a safety feature that can only be triggered by a failed phone unlock attempt. Whilst the underlying capability is the same, the trigger mechanism is modified to appear legitimate. On-platform apps also tend to use the modern and battery-efficient `FusedLocationProviderClient` API in location services, which is Google's recommended

best practice. Conversely, off-platform apps use the older `LocationManager` API, which allows for more direct and aggressive control over location polling.

2) Analysis of Modded APK Variants: The seven modded variants analysed present a varied risk profile compared to their official counterparts. Four of the seven—Cerberus 3.6.4, Cerberus AntiTheft App (v3.7.8), TheTruthSpy (v9.41), and TrackviewApk (v3.7.06)—appeared to be functionally identical to official releases re-hosted on unregulated platforms. This conclusion is supported by matching package names, digital certificates, network behaviour in relation to known domains used by standard versions, and average VirusTotal scores, although a direct byte-for-byte comparison was only possible for TrackviewApk v3.7.06 due to version availability.

The remaining three APKs were significantly altered. The modded TrackView (v22.23) variant, though not flagged by VirusTotal, resembled adware. Its original surveillance functionality was removed entirely and its UI triggered intrusive Google Ads dialogues upon any user interaction.

The two mSpy mods, mSpy MOD Premium Unlocked and MSPY WhatsApp, were flagged as malicious but were revealed to be incomplete 'dropper' apps. Upon launch, they present a simple interface with marketing 'tips' that ultimately instructs the user to 'go to site official and download apk'. Despite requesting permissions, dynamic analysis confirms they make no network calls, contain no surveillance or downloader code and are thus classified as PUAs.

E. Shared Technical and Distribution Strategies Across Families (RQ3)

To address **Research Question 3** we analyse the 75 non-modded apps both technically and contextually according to release date, examining whether families independently converged on a shared set of features or implementations.

UI Suppression and Stealth Features: 'Stealth features' hide the app's presence and obfuscate its notifications from a survivor. A common implementation for programmatically hiding an app's icon from the device's launcher uses the `PackageManager.setComponentEnabledSetting()` method with the `COMPONENT_ENABLED_STATE_DISABLED` flag. This removes the app's icon from the app drawer, and is adopted by initial and sideloaded Cerberus versions (v2.2, March 2012 through to all sideloaded v3.8.0 variants in May 2024) as well as all TheTruthSpy versions analysed from October 2016 onwards.

Other implementations within this category are more app specific. mLite versions from v2.1.2 (May 2018) consistently use `android.R.color.transparent` in their notification iconography, effectively making icons invisible in the status bar to stop serving as a clear visual warning that a background service is active. Early versions of TheTruthSpy from March 2016 to September 2018 use the `disableKeyguard` method, a now-deprecated function that bypasses the device's lock screen, potentially allowing the app to perform actions while the screen was off.

Persistence Mechanisms and Boot-Time Behaviour: Another core shared strategy across all families is ensuring the stalkerware's persistence across various events. The most fundamental persistence mechanism across nearly all apps is the ability to automatically start when the device is turned on. From the earliest Cerberus versions (v2.2, March 2012) to PhoneParental (v1.0, February 2024) and mSpy (v8.6.0.1, May 2025), all register a `BootReceiver` to automatically launch on device startup by listening for the `android.intent.action.BOOT_COMPLETED` action. Upon receiving this signal, the receiver's code is executed, typically with the sole purpose of starting the main surveillance service. All versions of TheTruthSpy and mSpy also listen for the `QUICKBOOT_POWERON` action, which is non-standard but commonly used by device manufacturers for faster boot cycles [31].

Once running, we find all families universally employ Android's `AlarmManager` to schedule recurring surveillance tasks, such as restarting services or uploading data. `AlarmManager` sets alarms that wake the app at various intervals, ensuring continued operation even when the device is idle, as seen in Cerberus v3.8.0 (May 2024) using `setAndAllowWhileIdle()` to guarantee execution, even when the device is in sleep mode. TheTruthSpy (from October 2016) employs `AlarmManager` to trigger the periodic synchronisation of data, such as live location information via the `SYNC_LOCATION_ACTION`.

Obfuscation, Anti-Analysis Techniques and Root Detection: To evade detection and hinder reverse engineering, developers consistently employ several shared techniques. Reflection is a pervasive method used to access hidden Android APIs and obscure functionality from static analysis tools, present in sideloaded apps from early Cerberus versions (v2.2, March 2012) to its v3.8.0 release in May 2024. Older Cerberus releases (until v3.2.1, April 2015) and TheTruthSpy (October 2016) use reflection to toggle system settings such as mobile data usage and to dismiss system notifications, while modern Cerberus versions use it within the `com.surebrec.SuCommands` class to interact with a wide range of powerful system services, including the `IPackageManager` and `IDevicePolicyManager` to grant permissions and block app uninstallation.

Debugger and emulator detection techniques appear consistently across all the Cerberus families and mLite, particularly from 2018 onwards. These checks, which look for signs of an analysis environment (e.g., `Debug.isDebuggerConnected()` or hardware names like `generic_x86`), serve to prevent dynamic analysis by security researchers. We confirm that these checks are almost exclusively located within the proprietary code of the app (e.g., `l2.AbstractC1230h` in Cerberus) rather than being inherited from third-party libraries, indicating a deliberate, in-house effort by these developers to build anti-analysis capabilities into their core product.

Older apps also used root detection to acquire elevated privileges. In proprietary packages in Cerberus' 2012 versions,

the app searches for the `su` binary in `/system/bin/su` or `/system/xbin/su`, whilst TheTruthSpy versions from October 2016 to September 2018 contain a `com.ispyoo.common.root.RootPermission` class that, on a rooted device, executes commands such as `am force-stop com.bbm`. This command forcibly terminates the BlackBerry Messenger (BBM) app, likely to unlock its database files from an active process and allow TheTruthSpy to exfiltrate a survivor's private chat logs. However, from July 2018 onward, root check practices shifted, almost exclusively appearing as an incidental feature inherited from third-party SDKs (e.g., `io.fabric.sdk.android`) rather than being part of the stalkerware's proprietary code. This suggests that developers may no longer see significant benefits in building dedicated features for rooted devices compared to their other surveillance capabilities and their continued need for debugger and emulator detection.

Use of Accessibility Services: Our analysis reveals that off-platform stalkerware families—Cerberus, mSpy, and TheTruthSpy—have converged on the use of Android's Accessibility Services as a primary mechanism for data exfiltration and surveillance. The implementation follows a consistent, shared blueprint across the otherwise distinct families, and has persisted in versions from 2018 onwards in all three families. First, the app requests the `BIND_ACCESSIBILITY_SERVICE` permission and declares a custom service extending Android's `AccessibilityService`. Once enabled by the perpetrator, this service monitors all UI interactions by implementing the `onAccessibilityEvent` callback method, listening for events like `TYPE_VIEW_TEXT_CHANGED` to detect when new text appears on screen. The stalkerware can then use the `AccessibilityNodeInfo` API to traverse the screen's view hierarchy and read the content of other apps. For example, a 2018 version of TheTruthSpy explicitly uses `source.findAccessibilityNodeInfosByViewId("com.whatsapp:id/message_text")` to locate and extract the text from WhatsApp messages. Similarly, the 2024 versions of the Cerberus Standard and Disguised apps implement a class, `com.surebrec.AccService`, which recursively traverses the `AccessibilityNodeInfo` tree to find and log text content, effectively creating a keylogger, which is not one of its advertised capabilities.

F. Shared Ecosystem Patterns

Beyond analysing individual families, we investigate the extent seemingly separate developers operate within a shared ecosystem below, finding that these stalkerware families operate without collaborative commonalities but are posited as distinct competitors.

Infrastructure and Common Libraries: To investigate potential technical ecosystem overlaps, a two-pronged analysis across the 75 non-modded APKs is performed. First, a script iterates over each decompiled app, searching for shared infrastructure indicators, including hardcoded IP addresses, C2 domains, developer emails, and unique non-standard code libraries via regular expressions. This was complemented by dynamic analysis, where captured network traffic was cross-referenced to identify any shared C2 communications or other endpoints that emerge only at runtime. Neither approach found significant evidence of shared infrastructure across the four distinct ecosystems (Cerberus, mSpy, TheTruthSpy, and TrackView). Beyond standard Android development libraries, analysis identified no common domains, IP addresses, or custom libraries that would suggest a shared technical backend. This null finding suggests that despite operating in the same market and adopting similar high-level strategies, these major stalkerware developers maintain distinct and separate technical infrastructures. The lack of overlap indicates they likely develop their core backend systems and surveillance features in-house, positioning them as direct commercial competitors rather than collaborators.

Contextual Evidence: Contextual evidence from privacy policies and marketing content corroborates that these developers operate as competitors, revealing their strategies for managing legal risk and market positioning. This competitive posturing is evidenced by marketing content. The mSpy and PhoneParental websites, for example, host blogs that rank their products against rivals (Figure 3). These articles, often styled as 'Top 10' lists, position their own product as superior by highlighting the perceived shortcomings of competitors. This strategy implies a direct commercial rivalry, reinforcing the idea that these are distinct, competing entities, not collaborators.

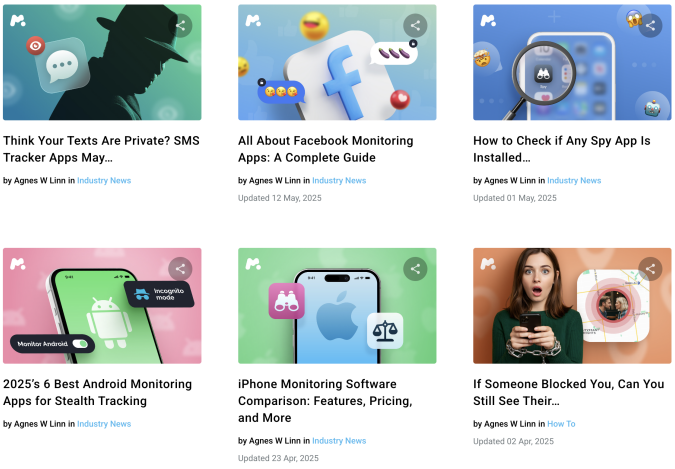

Fig. 3. mSpy's most recent blog posts, featuring how-to articles on general surveillance topics and product comparisons against rival products.

Lastly, a comparative analysis of privacy policies reveals divergent approaches to transparency and legal posturing. The mLite/mSpy policy uses dense, GDPR-focused language to project robust compliance, claiming survivors are not 'data subjects' because their data is encrypted—a questionable

interpretation of the regulation.[4] Cerberus and TrackView offer some specifics on data collection but include broad liability disclaimers, with TrackView's policy being particularly generic about its surveillance purpose. Both disclaim any security warranties. PhoneParental's policy is the most opaque, omitting any details on data collection, security, or legal adherence, while asserting that all data is gathered with the subject's explicit consent.

V. Discussion

Our study, motivated by the continual growth in stalkerware's user base, which persists despite recent platform countermeasures, analyses the longitudinal and technical evolution of Android stalkerware. We investigate how developers adapt their apps in response to platform policies, OS security enhancements and differing distribution channels. Through a multi-faceted analysis of four distinct app families—Cerberus, TheTruthSpy/PhoneParental, TrackView, and mLite/mSpy—across 82 APKs from various sources, we explore the key technical, strategic, and contextual patterns that define the modern stalkerware ecosystem.

1) RQ1: Longitudinal Evolution in Response to Platform and Policy Changes: Analysis of RQ1 reveals a dichotomy in evolutionary strategies based on an app's distribution model. A primary takeaway is that the effectiveness of Google's policies is largely confined to its own marketplace. For developers operating exclusively off-platform, such as TheTruthSpy, both the 2018 permission restrictions and the 2020 explicit stalkerware policy were met with non-response. These families consistently chose to target older, less secure Android API levels for years, a strategic stagnation that likely allows them to bypass modern privacy controls like Scoped Storage and granular permissions, thereby maximising their surveillance capabilities for longer periods of time.

Conversely, apps with a presence on Google Play, such as mLite, TrackView Viewer and the modern Cerberus Google Play variant, demonstrate a pattern of slow, and often reluctant, compliance. For example, the delayed adoption of the `isMonitoringTool` flag by mLite, nearly three years post-policy, suggests that enforcement may be inconsistent or developers wait until direct pressure is applied. This tactic, much like the prohibited practice of funnelling users from a Google Play app to a non-compliant sideloaded version, highlights a persistent failure in Google's policy enforcement. It also indicates that stalkerware developers perceive platform policies not as ethical guidelines, but as technical obstacles to be navigated with minimal functional compromise.

Overall, analysis of RQ1 demonstrates that current policy pressure often redirects rather than eliminates covert-surveillance functionality, a problem compounded by inconsistent platform governance. Google's granular permission bans and `isMonitoringTool` flag create visible compliance incentives for developers who value Google Play reach, but off-platform actors can ignore these rules entirely, or create compliant variants that can sidestep intent by funnelling users toward sideloaded builds. Our findings suggest that store-level policy must be coupled with robust device-level guardrails. To account for perpetrators bypassing initial installation warnings by design, Android should implement periodic, non-dismissible system alerts if an app exhibits high-risk behaviours post-installation (e.g., hiding its icon while accessing location data). This provides a persistent warning to the device's primary user, complementing stronger install-time checks based on permission bundles or known stalkerware hashes. Enforcement lags, such as mLite's three-year delay in adding `isMonitoringTool`, indicate that Google Play Protect and Google's app review systems for stalkerware-like products remains insufficient. These processes should be reviewed and examined in further detail to evaluate their true efficacy in combatting stalkerware proliferation.

2) RQ2: Platform Adaptations and Feature Divergence: We find a deliberate strategy of feature differentiation across platforms in our analysis of RQ2. Direct comparison of on-platform and off-platform variants from the same families (Cerberus, TrackView, mSpy) reveals Google Play versions are consistently stripped of their most invasive functionalities. Static analysis reveals invasive capabilities exclusive to sideloaded versions; this 'feature stripping' is an act of strategic re-architecting. TrackView's 'split-app' model, where the compliant Google Play Viewer app acts as a remote control for the fully-featured sideloaded HomeSafe app, is an example of developers leveraging Google's platform for distribution and legitimacy while keeping prohibited functionality easily available. Similarly, the enhancement of features in off-platform versions, such as Cerberus Disguised version's use of Accessibility Services to implement screen-reading and key-logging (capabilities absent from its on-platform counterpart) underscores that developers maintain a separate, more potent development track for their unregulated products.

The analysis of modded APKs reveals that third-party modifications often introduce new risks, such as adware or deceptive downloaders, rather than enhancing core surveillance features. This suggests that the threat from the official developers, who tailor their products to evade detection, may be more potent than that from opportunistic modders. Overall, findings from RQ2 underline how platform governance displaces, rather than removes, abusive functionality.

3) RQ3: Shared Technical and Distribution Strategies: Cross-family analysis confirms the existence of shared technical blueprints for core stalkerware functionality. Despite operating as commercial competitors, developers have independently converged on a set of common and highly effective techniques for achieving persistence, stealth, and surveillance. These features also emerge and persist at the same time across developers — the universal use of `BootReceiver` to ensure operation after a reboot, `AlarmManager` to schedule recurring tasks, and the abuse of Accessibility Services, points to a set of established 'best practices' within the stalkerware

[4]Under GDPR (Recital 26), encrypted data is typically considered pseudonymised, not anonymised. As long as a key exists to decrypt the information and re-identify the individual, the data remains personal data and is subject to the regulation's protections.

industry.

This technical convergence is contrasted by a lack of shared infrastructure. We find no evidence of common C2 servers, code libraries, or developer certificates across the four main families. This, combined with contextual evidence such as comparative marketing blogs hosted by mSpy and PhoneParental, indicates these developers operate as distinct, competitive entities rather than collaborators within a unified malware economy. They may learn from one another's public techniques, but their commercial operations remain siloed.

While static identifiers, including package names or file hashes, are easily altered between releases, underlying malicious functionalities are core behaviours within stalkerware products and remain consistent across all OS versions. These persistent indicators, such as the abuse of high-privilege APIs or similar implementations of stealth techniques, should be weighed more heavily than static cues by malware scanners, detection engines and Google Play Protect to determine if an app could be stalkerware. Additionally, having observed that families operate distinct infrastructures, IP or domain block-lists need to be vendor-specific; generic 'stalkerware C2' lists risk high false-negatives, presenting another issue in using static indicators over active malicious behaviour. Effective scanners must examine runtime behaviour—what an app actually does with its permissions—to correctly flag abuse.

Our work provides evidence that, while platform policies can force cosmetic or functional changes, they do not eradicate the threat. Developers are adept at navigating these restrictions, often by leveraging the very platforms that seek to regulate them. The rebranding of The TruthSpy to the benign-sounding 'PhoneParental', and TrackView's use of Google Play to distribute the benign part of its system, are testaments to this strategic resilience. Mapping these adaptive strategies is therefore critical for developing stalkerware countermeasures, and this work provides a foundational longitudinal analysis that underscores the need for continuous observation of the stalkerware ecosystem, which will only continue to evolve as Android does.

VI. Conclusion

Android stalkerware remains a persistent threat, as stalkerware developers consistently find new methods to evade platform-level countermeasures. Given the ongoing risk to survivors, it is important for platform owners, such as Google, and security vendors to re-evaluate and innovate their approach to this challenge. Merely implementing store-level policies is not a sustainable solution, as our study demonstrates these strategies are systematically circumvented through shared tactics such as malicious compliance and feature stripping. Our research also clearly showcases changes in developer behaviour across platforms and a technical convergence on shared strategies for stealth and surveillance across multiple stalkerware products by differing developers.

We are confident that there is potential for improvements in user protection when these adaptive patterns are understood and anticipated across the Android security ecosystem, and actively prioritised by Google in new Android OS releases. We underscore the necessity of embracing a more behaviour-centric perspective to counter these evolving threats, creating future-proof detection engines to discern malicious behaviours such as icon hiding or Accessibility Service abuse regardless of an app's name. In essence, by harnessing these insights into developer strategy, platform owners and security tools can move from a reactive posture to one that can proactively safeguard users from technology-facilitated abuse.

The primary limitation of this study, imposed by ethical constraints, is the inability to purchase and analyse the most recent, premium versions of the stalkerware families. The analysis therefore relies on publicly available and freely accessible APKs. While this provides a rich historical dataset, the most cutting-edge, paywalled features may not be fully observed in a dynamic context.

Future work could investigate iOS-focused surveillance tools, which may differ from their Android counterparts in both technical approach and distribution models. Apple's App Store stalkerware presence and review processes are an unexplored area within this study, so a systematic comparison between iOS and Android stalkerware implementations could explore whether the adaptation patterns documented in Android stalkerware emerge similarly under Apple's ecosystem governance. Such insights could guide platform-agnostic detection strategies and inform coordinated policy responses. Future work could also investigate stalkerware apps in non-English-speaking countries. Previous work [17] and initial app gathering suggest that certain stalkerware apps may be advertised under different language marketplaces entirely. Further study could clarify whether observed adaptation patterns hold globally, how powerful Google's enforcement patterns are worldwide, and whether novel evasion techniques unseen in English-language apps and forums exist.

Acknowledgment

This work is supported by Nokia Bell Labs (for LAS) and the European Research Council under the Horizon 2020 programme (grant agreement No 949127) (for AH).

References

[1] M. M. Rogers, C. Fisher, P. Ali, P. Allmark, and L. Fontes, "Technology-facilitated abuse in intimate relationships: A scoping review," *Trauma, Violence, & Abuse*, vol. 24, no. 4, p. 152483802210902, 5 2022.

[2] C. Gibson, V. Frost, K. Platt, W. Garcia, L. Vargas, S. Rampazzi, V. Bindschaedler, P. Traynor, and K. Butler, "Analyzing the monetization ecosystem of stalkerware," *Proceedings on Privacy Enhancing Technologies*, vol. 2022, no. 4, p. 105–119, 10 2022.

[3] A. Shahani, "Smartphones are used to stalk, control domestic abuse victims," *NPR.org*, 9 2014. [Online]. Available: https://www.npr.org/sections/alltechconsidered/2014/09/15/346149979/smartphones-are-used-to-stalk-control-domestic-abuse-victims

[4] A. Greenberg, "Hacker Eva Galperin has a plan to eradicate stalkerware," 4 2019. [Online]. Available: https://www.wired.com/story/eva-galperin-stalkerware-kaspersky-antivirus/

[5] I. Thomson, "After blitzing FlexiSpy, hackers declare war on all stalkerware makers: "we're coming for you"," 4 2017. [Online]. Available: https://www.theregister.com/2017/04/25/hackers_attack_stalkerware_flexispy/

[6] Avast PR, "Stalkerware grows 239% worldwide over the past three years," 3 2023. [Online]. Available: https://press.avast.com/stalkerware-grows-239-worldwide-over-the-past-three-years

[7] P. Mangeard, B. Tejaswi, M. Mannan, and A. Youssef, "WARNE: A stalkerware evidence collection tool," *Forensic Science International: Digital Investigation*, vol. 48, p. 301677, Mar 2024.

[8] K. Allix, T. F. Bissyandé, J. Klein, and Y. L. Traon, "AndroZoo: Collecting millions of Android apps for the research community," in *2016 IEEE/ACM 13th Working Conference on Mining Software Repositories (MSR)*, 2016, pp. 468–471.

[9] L. A. Saavedra, H. S. Dutta, A. R. Beresford, and A. Hutchings, " ModZoo: A Large-Scale Study of Modded Android Apps and Their Markets ," in *2024 APWG Symposium on Electronic Crime Research (eCrime)*. Los Alamitos, CA, USA: IEEE Computer Society, Sep. 2024, pp. 162–174. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/eCrime66200.2024.00018

[10] C. Fraser, E. Olsen, K. Lee, C. Southworth, and S. Tucker, "The new age of stalking: Technological implications for stalking," *Juvenile and Family Court Journal*, vol. 61, no. 4, pp. 39–55, 2010. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1755-6988.2010.01051.x

[11] D. Freed, J. Palmer, D. E. Minchala, K. Levy, T. Ristenpart, and N. Dell, "Digital technologies and intimate partner violence: A qualitative analysis with multiple stakeholders," *Proc. ACM Hum.-Comput. Interact.*, vol. 1, no. CSCW, Dec. 2017. [Online]. Available: https://doi.org/10.1145/3134681

[12] E. Tseng, R. Bellini, N. McDonald, M. Danos, R. Greenstadt, D. McCoy, N. Dell, and T. Ristenpart, "The tools and tactics used in intimate partner surveillance: an analysis of online infidelity forums," in *Proceedings of the 29th USENIX Conference on Security Symposium*, ser. SEC'20. USA: USENIX Association, 2020.

[13] R. Chatterjee, P. Doerfler, H. Orgad, S. Havron, J. Palmer, D. Freed, K. Levy, N. Dell, D. McCoy, and T. Ristenpart, "The spyware used in intimate partner violence," in *2018 IEEE Symposium on Security and Privacy (SP)*, 2018, pp. 441–458.

[14] C. Parsons, A. Molnar, J. Dalek, J. Knockel, M. Kenyon, B. Haselton, C. Khoo, and R. Deibert, "The predator in your pocket: A multidisciplinary assessment of the stalkerware application industry," Jun 2019. [Online]. Available: http://hdl.handle.net/1807/96320

[15] Y. Han, K. A. Roundy, and A. Tamersoy, "Towards stalkerware detection with precise warnings," in *Proceedings of the 37th Annual Computer Security Applications Conference*, ser. ACSAC '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 957–969. [Online]. Available: https://doi.org/10.1145/3485832.3485901

[16] E. Liu, S. Rao, S. Havron, G. Ho, S. Savage, G. M. Voelker, and D. McCoy, "No privacy among spies: Assessing the functionality and insecurity of consumer Android spyware apps," *Proceedings on Privacy Enhancing Technologies*, vol. 2023, no. 1, p. 207–224, 1 2023.

[17] M. Almansoori, A. Gallardo, J. Poveda, A. Ahmed, and R. Chatterjee, "A global survey of Android dual-use applications used in intimate partner surveillance," *Proceedings on Privacy Enhancing Technologies*, vol. 2022, no. 4, p. 120–139, Oct 2022.

[18] E.-M. Maier, L. M. Tanczer, and L. D. Klausner, "Surveillance disguised as protection: A comparative analysis of sideloaded and in-store parental control apps," *Proceedings on Privacy Enhancing Technologies*, vol. 2025, no. 2, p. 107–124, Apr 2025. [Online]. Available: https://petsymposium.org/popets/2025/popets-2025-0052.pdf

[19] R. Hager, "Google's new SMS and call permission policy is crippling apps used by millions," Jan 2019. [Online]. Available: https://www.androidpolice.com/2019/01/05/googles-new-sms-and-call-permission-policy-is-crippling-apps-used-by-millions/

[20] A. Stevanović, "Why is there so much spyware hidden in the Play Store?" Mar 2025. [Online]. Available: https://www.techradar.com/vpn/vpn-privacy-security/why-is-there-so-much-spyware-hidden-in-the-play-store

[21] Google, "Codenames, tags, and build numbers," 2024. [Online]. Available: https://source.android.com/docs/setup/reference/build-numbers

[22] S. Hutchinson, B. Zhou, and U. Karabiyik, "Are we really protected? an investigation into the Play Protect service," in *2019 IEEE International Conference on Big Data (Big Data)*, 2019, pp. 4997–5004.

[23] M. Alecci, P. J. R. Jiménez, K. Allix, T. F. Bissyandé, and J. Klein, "AndroZoo: A retrospective with a glimpse into the future," in *Proceedings of the 21st International Conference on Mining Software Repositories*, ser. MSR '24. New York, NY, USA: Association for Computing Machinery, 2024, p. 389–393. [Online]. Available: https://doi.org/10.1145/3643991.3644863

[24] Kaspersky, *The State of Stalkerware in 2023 - A Kaspersky Report*, Feb 2024. [Online]. Available: https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2024/03/07160820/The-State-of-Stalkerware-in-2023.pdf

[25] AssoEchap, "Stalkerware indicators of compromise," Sep 2022. [Online]. Available: https://github.com/AssoEchap/stalkerware-indicators

[26] Z. Whittaker, "TheTruthSpy spyware found on 50,000 Android devices - business & human rights resource centre," Feb 2024. [Online]. Available: https://www.business-humanrights.org/en/latest-news/thetruthspy-spyware-found-on-50000-android-devices/

[27] skylot, "jadx," Nov 2019. [Online]. Available: https://github.com/skylot/jadx

[28] K. Graf, J. Lerga, and B. Dobraš, "Data collection and hiding capabilities of Android stalkerware," *2023 IEEE 21st Jubilee International Symposium on Intelligent Systems and Informatics (SISY)*, p. 000389–000394, 9 2023.

[29] NIST, "NVD - CVE-2022-0732," 2022. [Online]. Available: https://nvd.nist.gov/vuln/detail/CVE-2022-0732

[30] C. Pallardy, "Sensitive data of millions stolen in MSpy breach," 2024. [Online]. Available: https://www.informationweek.com/cyber-resilience/sensitive-data-of-millions-stolen-in-mspy-breach

[31] HTC, "How to use the bootloader," 2025. [Online]. Available: https://www.htc.com/lk/contact/productissue/htc/5bf5ade5-d551-94c4-c09f-57d25d90cbcc/

VII. Appendix

This appendix provides supplementary data supporting our main analysis. Table I details the provenance, distribution, and VirusTotal risk score for every APK analysed in this study. Table II catalogues the advertised surveillance capabilities for each app family, comparing their on-platform and off-platform offerings. Finally, Table III outlines the specific APK pairs selected for the direct on-platform versus off-platform comparison in line with RQ2, and the rationale for their selection.

TABLE I: Provenance and Distribution of Analysed APKs.

APK Details	Family	Original Distribution Source			APK(s) Sourced From	Average VirusTotal Score
		Google Play	Dev. Site	Modded Market		
Cerberus (Pre-Google Play Ban) (Eight versions) v2.2, v2.3, v2.5, v2.5.1, v3.2.1, v3.5.1, v3.5.7, v3.5.9 (Mar 2012 – Dec 2018)	Cerberus	✓	✓		AndroZoo	17/66
Cerberus (During Google Play Ban) (Four versions) v3.6.5, v3.6.6, v3.6.7, v3.6.9 (Feb 2020 – Oct 2021)	Cerberus		✓		AndroZoo	25/66
Cerberus (Google Play Re-Release, Current) (Seven versions) v1.1.2_play, v1.1.7_play, v1.2.6_play, v1.3.8_play, v1.4.0_play, v1.4.3_play, v1.4.5_play (Oct 2023 – May 2025)	Cerberus	✓			Androzoo, APKMirror	0/66
Cerberus (Standard & Disguised Editions) (2 versions) v3.8.0, v3.8.0 (May 2024)	Cerberus		✓		Cerberus Website	18/66
Cerberus 3.6.4 (Modded)[a] v3.6.4 (Jul 2023)	Cerberus			✓	Malavida	2/65
Cerberus AntiTheft App (Modded)[a] v3.7.8 (Oct 2023)	Cerberus			✓	OfflineModAPK	19/64
TheTruthSpy (Nine versions) v1.0 (Eight versions)[b], v8.80 (Oct 2016 – Jul 2022)	TheTruthSpy		✓		AndroZoo	32/66
PhoneParental (TheTruthSpy Rebrand) (One version) v1.0 (Feb 2024)	TheTruthSpy		✓		PhoneParental Website	29/66
The Truth Spy (Modded)[a] v9.41 (Jul 2022 – Dec 2022)	TheTruthSpy			✓	APKAdmin	34/66
TrackView HomeSafe (One version) v3.8.65 (Jun 2025 – Jun 2025)	TrackView		✓		TrackView Website	4/67
TrackView (Pre-Removal from Google Play) (12 versions) v3.3.17-tv, v3.4.11-tv, v3.5.03-tv, v3.5.11-tv, v3.5.19-tv, v3.5.24-tv, v3.5.28-tv, v3.6.29, v3.6.45, v3.6.48, v3.6.77, v3.7.06 (Jul 2018 – Dec 2021)	TrackView	✓			APKPure	10/61

TABLE I – continued from previous page

APK Details	Family	Original Distribution Source			APK(s) Sourced From	Average VirusTotal Score
		Google Play	Dev. Site	Modded Market		
TrackView Viewer (Google Play) (Nine versions) v3.7.49, v3.8.02, v3.8.06, v3.8.12, v3.8.27, v3.8.30, v3.8.35, v3.8.39, v3.8.42 (Jun 2022 – Apr 2025)	TrackView	✓	✓		APKPure	22/65
TrackView (Modded)[a] v22.23 (Apr 2022)	TrackView			✓	APKPure	0/66
TrackviewApk (Modded)[a] v3.7.06 (Sep 2023)	TrackView			✓	LatestModAPKs	13/67
mLite (Google Play) (21 versions) v1.0.1, v1.4.2, v2.1.2, v2.1.3, v2.1.4, v2.1.5, v2.3.11, v2.3.14, v2.3.16, v2.3.2, v2.3.21, v2.3.22, v2.3.26, v2.3.31, v2.3.35, v3.0.8, v3.0.9, v3.2.16, v3.2.23, v3.2.7 (Jul 2016 – May 2024)	mSpy	✓			AndroZoo	7/62
mSpy (One version) v8.6.0.1 (May 2025)	mSpy		✓		mSpy Website	3/64
mSpy MOD Premium Unlocked (Modded)[a] v2.01.54.08 (Oct 2024)	mSpy			✓	ApkResult	4/66
MSPY WhatsApp (Modded)[a] v6.2 (Oct 2024)	mSpy			✓	APKRabi	3/66

[a] Modded versions are designated in **red**. Unlike official releases, they are treated as distinct, non-sequential snapshots; their version details and release dates reflect their listings on third-party marketplaces, not a continuous development timeline.
[b] Most TheTruthSpy APKs list a static '1.0' version in their manifests. As file comparisons confirm they are distinct updates, they are identified in this study by sequential labels ordered by their AndroZoo collection.
Some release dates are estimated due to a lack of official changelogs or AndroZoo data. To ensure a sound timeline, these dates were approximated by correlating several datapoints, including the target SDK version (which corresponds to a specific Android release period), the APK's digital certificate validity dates, and its first appearance in data repositories.

TABLE II
ADVERTISED SURVEILLANCE CAPABILITIES BY APPLICATION FAMILY AND VARIANT.

Category	Capabilities	Cerberus (Google Play)	Cerberus (Sideloaded)	TheTruthSpy/PhoneParental	mLite	mSpy	TrackView[a]
Communication & Social Media Monitoring	Call Logging & Recording	—	—	✓	—	✓	—
	SMS & MMS Monitoring	—	—	✓	✓	✓	—
	Social Media & Instant Messaging Monitoring (e.g. WhatsApp, Facebook)	—	—	✓	✓	✓	—
	Email Monitoring	—	—	✓	—	✓	—
Location, Live Surveillance & Input Capture	Real-Time GPS Tracking	✓	✓	✓	✓	✓	✓
	Location History	✓	✓	✓	✓	✓	✓
	Geofencing	✓	✓	✓	✓	✓	—
	Ambient Listening (Record Surroundings)	✓	✓	✓	✓	✓	✓
	Remote Camera Access (Live Photo/Video)	✓	✓	✓	✓	✓	✓
	Screen Recording	—	✓	✓	—	✓	—
	Keylogging	—	—	✓	—	✓	—
Stored Data Exfiltration	View Saved Photos & Videos	—	—	✓	—	✓	—
	Browser History Tracking	—	—	✓	✓	✓	—
	View Contacts	—	—	✓	✓	✓	—
	View Calendar	—	—	✓	—	✓	—
	View Installed Applications & App Usage Statistics	—	—	✓	✓	✓	—
Stealth & Remote Commands	Stealth Mode/Hidden Icon	—	✓	✓	—	✓	✓
	Remote Device Wipe/Lock	✓	✓	—	—	—	—
	Remote Alarm	✓	✓	—	✓	—	✓

✓: Feature explicitly claimed on website, marketing materials, or within app.
[a] TrackView's capabilities are consolidated into a single column, as all surveillance features are implemented in the sideloaded TrackView HomeSafe application. The separate TrackView Viewer app available on Google Play is a client that only accesses and controls the main app and has no surveillance features of its own. This is explained further in Section IV-B.

TABLE III: Selected APK Pairs for On-Platform vs. Off-Platform Comparison.

Family	On-Platform Version (Google Play)[a]	Off-Platform Version (Sideloaded)	Rationale for Comparison
Cerberus	**Cerberus (Google Play Re-Release)** v1.2.6_play (May 2024)	**Cerberus (Standard & Disguised)** v3.8.0 (May 2024)	The developer's website explicitly positions these versions as direct, fully-featured alternatives to the feature-limited Google Play release.
TrackView	**TrackView Viewer (Current Google Play)** v3.8.42 (Apr 2025)	**TrackView HomeSafe (Current Sideloaded)** v3.8.65 (Jun 2025)	Represents the 'split-app' model. Unlike other families, the Viewer app (Google Play) is designed to work with the full HomeSafe application, which must be sideloaded. HomeSafe, however, can also work as a standalone product without Viewer.
mSpy / mLite	**mLite (Google Play)** Latest Analysed Version (May 2024)	**mSpy (Full Suite, Sideloaded)** v8.6.0.1 (May 2025)	mLite and mSpy are developed by the same company; however, the Google Play app lacks invasive features compared to the premium mSpy product.

[a] No on-platform versions of TheTruthSpy were available for comparison, as this family has historically operated entirely outside of Google Play.