Dead Domains, Living Data: A Privacy Risk Analysis of Domain Lifecycle in Android Apps

Gabriel Hortea
Universidad Carlos III de Madrid
bhortea@pa.uc3m.es

Aniketh Girish
IMDEA Networks Institute
aniketh.girish@imdea.org

Narseo Vallina-Rodríguez
IMDEA Networks Institute
narseo.vallina@imdea.org

Juan Tapiador
Universidad Carlos III de Madrid
jestevez@inf.uc3m.es

Abstract

Mobile applications transmit sensitive data and user identifiers to domain-based endpoints embedded in SDKs and third-party services. Unlike the web, where operators can update or remove third-party endpoints, apps are static binaries that remain installed for years and continue sending identifiers to endpoints whose domains may expire, change ownership, or become abandoned. This mismatch creates a privacy risk, since identifiers may flow to hijacked, unmaintained, or maliciously re-registered endpoints without users' knowledge or consent. This work presents a privacy risk analysis of domain endpoints in Android apps. We identify endpoints receiving sensitive identifiers through dynamic analysis of 11,131 apps and track the lifecycle of 3,420 associated domains around their expiration events. Our analysis reveals that 25.3% of domains are renewed after expiration, with 78.7% of these belonging to third-party services or SDKs embedded in apps whose install brackets total over 78 billion cumulative downloads, placing a potential user base of billions at risk. While the majority of these lapses are short and fall within registrar grace periods, we identify 218 domains with dangling CNAMEs susceptible to subdomain hijacking (34 of which received 91 valid TLS certificates during their expired period), and show that 17.0% of domains experience TLS issuance gaps where confidentiality is lost. We categorize high-risk endpoints into Advertising and Tracking vs Backend and Utility services, finding that vulnerable endpoints are embedded in over 4,000 apps. By correlating WHOIS, DNS, and TLS lifecycles with dynamically observed identifier flows, we expose how domain mismanagement translates into data flows and user privacy harm. Finally, we derive mitigation strategies to address these supply-chain risks.

Keywords

Privacy, Domains, Android SDKs, DNS, CNAME, TLS

1 Introduction

Mobile applications are a primary entry point to the Internet: they routinely communicate with first-party services operated by the app developer (e.g., for content delivery or account management), and with third-party services that are integrated via Software Development Kits (SDKs) for advertising, analytics, authentication, and other functions. However, these third-party services are often opaque to users and can enable tracking across apps by collecting and correlating sensitive user data, including advertising identifiers, location data, and device fingerprints [30, 76, 79]. In practice, this communication is mediated by domain-based endpoints that are embedded in app code and SDK configurations. A key privacy challenge is that these endpoints are part of software supply chains. Thus, app developers may have limited visibility and control over the domains contacted by bundled SDKs, and maintenance is frequently misaligned across stakeholders. This issue is compounded by dependency staleness, since app developers often ship outdated versions of third-party libraries and updates are adopted slowly even when fixes are available [14]. In this ecosystem, domain lifecycle failures create concrete privacy risk that manifest differently than on the web. While web operators can update or remove third-party endpoints at any time, domains operate under renewal cycles that are independent from app lifecycles. An app version can remain installed for years while its referenced domains expire, are renewed late, change ownership (e.g., due to mergers and acquisitions), or become abandoned. During such events, apps and embedded SDKs may continue sending identifiers and rich user-level data without their knowledge or consent [76, 77, 79], potentially enabling adversaries to capture residual traffic through domain re-registration or DNS-based takeover (e.g., via dangling CNAMEs) and to exploit TLS misconfigurations [31, 75, 90].

In this paper we study the following research question:

> *How does the lifecycle mismanagement of embedded domain endpoints impact the privacy of mobile app users and their data?*

Domain lifecycle failures are a known security issue on the web, where expired domains have been abused for phishing, hijacking, and malware distribution [62, 84, 87]. Registrars provide partial safeguards against these threats, such as grace periods and redemption policies [33, 40], but their enforcement varies and does not eliminate risk. Recent work by So *et al.* [88] shows that domain expirations also affect mobile app DNS footprints at scale, but focuses on ecosystem-wide dependency stability and hijackability without examining the privacy-sensitive data that flows to them. Our work bridges these two lines of research: we correlate domain lifecycle events with the specific identifiers that apps transmit, quantifying how infrastructure mismanagement translates into concrete, observable privacy exposure. The mobile setting amplifies the risk

Proceedings on Privacy Enhancing Technologies 2026(4), 114–130

https://doi.org/10.56553/popets-2026-0112

relative to the web since apps are static, long-lived binaries whose SDK endpoints are opaque to users, where mobile APIs gate access to uniquely sensitive data that has often no direct web analogue.

Contributions. We design a methodology that combines dynamic analysis of Android apps with active DNS measurements. We dynamically execute apps to identify which endpoints transmit sensitive identifiers and track the lifecycle of selected domains around their expiration, combining WHOIS, DNS, TLS, registrar policies, and Google Play install ranges. We use a dataset of 1,008,539 APKs from AndroZoo, filter 11,131 apps, record network activity, extract 9,182 unique second-level domains, and focus our lifecycle analysis on 3,420 domains whose WHOIS expiration dates fall within our observation window. This enables us to characterize domain lifecycle behavior, infrastructure misconfigurations, PII flows, and potential user impact. We make the following key contributions:

- We characterize renewal timing and downtime for 3,420 domains referenced by Android apps and compare our findings with the results by So *et al.* [88]. We find that whereas 25.3% of domains are renewed after expiration, most of these lapses remain within registrar grace periods and only a small number reach the pending deletion phase (§2.1). Among the late-renewed domains, 78.7% are third-party services or SDKs referenced by apps whose Google Play install brackets total over 78 billion cumulative downloads, placing a potential userbase of billions at risk. We further analyze 88 registrars and reveal inconsistent enforcement of grace periods and large disparities in downtime.
- We identify 218 late-renewed domains with dangling CNAMEs during their expired period, which make them vulnerable to subdomain hijacking and silent interception of identifier-bearing traffic. By querying Certificate Transparency (CT) logs, we find that 34 of these domains received 91 unique certificates during their downtime windows. In parallel, we show that 17.0% of domains experience TLS issuance gaps (domains reactivated before valid certificates are issued) and 70.5% have certificates expiring before domain renewal, creating windows where confidentiality guarantees are weakened or lost.
- For expired domains that exhibit downtime and transmit identifiers, we categorize them into two groups: (1) *Advertising and Tracking* that collect a mix of advertising identifiers (e.g., `AAID`), persistent device IDs (e.g., `Android ID`, `bluetooth_address`), and precise geolocation to build cross-app identity graphs; and (2) *Backend and Utility Services* that collect application-specific identifiers such as `Device Fingerprint` and Firebase Installation IDs (`FID`) for operational diagnostics posing over-collection risks. Collectively, the vulnerable commercial tracking endpoints are embedded in 1,127 apps with install brackets totalling 5.9 billion downloads, while the vulnerable utility endpoints are present in 3,870 apps whose install brackets total 50.4 billion downloads.
- Finally, we derive actionable mitigation strategies for app developers, SDK maintainers, OS vendors, app stores, and registrars.

Overall, our results demonstrate that domain lifecycle management is not merely an operational concern, but a potential privacy risk in the mobile ecosystem. While the absolute number of hijackable domains during our 10-month observation window represents a small fraction of the ecosystem, this limitation underscores the severity of the threat. Because modern apps heavily rely on shared infrastructure, a single neglected third-party SDK or diagnostic backend can propagate across thousands of apps, exposing the identifiers and location data of a potentially large userbase. By bridging domain lifecycle security with mobile privacy measurements, and correlating WHOIS/DNS/TLS events with dynamically observed identifier flows, our work provides empirical evidence that infrastructure mismanagement in the domain supply chain translates into personal-data exposure, underscoring the need for privacy-aware governance of endpoint lifecycles in mobile app supply chains.

2 Background and Related Work

We next provide an overview of domain lifecycles and registrar policies (§2.1). We then discuss related work on domain expiration, residual trust, and the security and privacy challenges of the mobile supply chain (§2.2).

2.1 Domain Expiration and Renewal Policies

Once a domain name expires, registrars follow a structured process that determines how long the domain remains recoverable before it is either permanently deleted or made available for re-registration. However, because domains may change ownership over time or remain referenced by legacy SDK code long after products are discontinued, the effective lifetime of an endpoint can deviate from the app developer's expectations [14, 88]. While the specific durations and conditions vary across registrars and sometimes Top-Level Domains (TLDs), most adhere to a process comprising three phases:

(1) **Grace Period (Standard Renewal):** An initial period immediately after expiration during which the original registrant can renew the domain, often without incurring additional fees. Domains may remain operational during this time, though access restrictions or warning messages may be applied.
(2) **Redemption Period:** If the domain is not renewed during the grace period, it may enter a redemption phase where recovery is still possible, but it usually requires paying a penalty redemption fee in addition to the standard renewal cost.
(3) **Pending Deletion:** Domains that remain unrenewed after the redemption period are marked for deletion. At this stage, recovery is no longer possible and the domain is queued for eventual release to the public registry. Some registrars may auction the domain before final deletion.

Although ICANN provides general guidance through the Expired Registration Recovery Policy (ERRP) [33], implementation varies across registrars. While we observed over 80 different registrars overall, we restrict our focus to the most prominent ones (Table 1). Most registrars closely follow ICANN's standard lifecycle: a 30-day grace period, followed by a 30-day redemption phase, and a 5-day pending deletion window. However, exceptions exist: GoDaddy employs overlapping windows, placing domains in auction as early as day 26 while allowing redemption until day 41 [27]. This policy variability directly impacts security and uptime. Short renewal windows cause earlier service disruptions, while longer redemption periods offer structural inconsistencies that attackers can actively exploit to hijack dependent services [2]. We explore these distinctions in §5.2, analyzing renewal delays across registrars to contextualize observed downtime and grace period violations.

Table 1: Post-expiration domain lifecycle policies for 6 prominent registrars

Registrar	Grace Period	Redemption Period	Pending Deletion
GoDaddy [27]	Day 0–18	Day 19–41	Day 42–72
OVH [68]	Day 0–30	Day 31–60	Day 61–66
1API GmbH [1]	Day 0–30	Day 31–60	Day 61–66
Key-Systems GmbH [42]	Day 0–30	Day 31–60	Day 61–66
IONOS [39]	Day 0–30	Day 31–60	Day 61–66
ASCIO [5]	Day 0–30	Day 31–60	Day 61–66

Real-world incidents demonstrate how mismanaged domain lifecycles translate into security and privacy failures. Public reports detail widely used services temporarily losing critical domains, causing outages and exposure to domain parking or takeover [31, 75, 90]. Adversaries re-register expired corporate domains to intercept password-reset emails, hijack SaaS accounts or package repositories, and distribute malware to downstream users [26, 72]. These cases show that expirations do not merely disrupt uptime, but can also enable attackers to silently intercept emails, redirect web traffic, and harvest PII from legacy apps that continue transmitting data to infrastructure whose ownership has changed.

2.2 Related Work

Our work sits at the intersection of domain security, mobile supply chain analysis, and privacy leakage measurement. We next relate our contributions to prior efforts in these areas.

Domain Expiration and Hijacking Risks. The security implications of domain expiration are extensively documented. Lauinger *et al.* [48–50] measured domain lifecycles and showed how re-registered domains can abuse residual trust in the web to mount phishing and man-in-the-middle attacks. Our methodology closely follows their approach, but we extend these principles into the Android ecosystem to monitor app-facing endpoints and capture both PII exfiltration and TLS degradation during lifecycle gaps. Lever *et al.* [52] demonstrated how attackers exploit the residual trust of re-registered domains to deceive users and launch man-in-the-middle attacks. From a forensic perspective, Schlamp *et al.* [83, 84] detailed how adversaries hijack abandoned Internet resources, including domains and IP blocks. Other studies focus on specific exploitation methods, including short-lived registrations [7], automated drop-catching for cybercrime [61, 64], and Vissers *et al.*'s analysis of parked domains used in obscure monetization [98]. Closest to our work, Pariwono *et al.* [70] and Mutchler *et al.* [63] analyzed broken resources in the mobile ecosystem, identifying thousands of Android applications that dangerously continue to reference unreachable or hijackable domains. In contrast, we focus on the full lifecycle of these domains and link such failures to dynamically observed PII flows, TLS validity windows, and registrar policies to quantify privacy risk rather than only reachability.

Proactive Defense and Mitigation. Felegyhazi *et al.* [22] investigated proactive blacklisting approaches, suggesting that registrars and security firms should adopt anticipatory measures to prevent malicious re-registrations. At the policy level, ICANN's ERRP [33] establishes baseline registrar practices, including mandatory grace periods and renewal reminders. Liu *et al.* [54] demonstrated that proactive registrar interventions significantly reduce abuse. However, as Akiwate *et al.* [2] showed, structural inconsistencies in how registrars manage lifecycles can introduce their own security risks.

TLS Lifecycle and Certificate Misconfiguration. Some works have examined TLS behavior across changing infrastructures. Borgolte *et al.* [9] showed that domain-validated certificates can outlive the underlying infrastructure, allowing attackers to secure hijacked cloud domains. Similarly, Friess *et al.* [23] and Zhang *et al.* [104] found that dangling resources often retain valid TLS configurations, obscuring traffic interception. Ma *et al.* [56] further highlighted precarious key reuse and mismanagement among hosting providers. We incorporate this perspective in our work and analyze TLS gaps in mobile app dependencies, correlating them with WHOIS-based downtime and observed PII flows.

Web-Mobile Supply Chain and Privacy. In the web ecosystem, Yu *et al.* [103] documented the pervasiveness and flexibility of third-party tracking: trackers are typically loaded as remote scripts that site operators can rotate, remove, or reconfigure without redeploying client code. In contrast, mobile apps embed third-party SDKs into long-lived binaries, so infrastructure decay and domain expirations can persist for years after deployment, making the consequences of lifecycle mismanagement more permanent and harder to remediate. In the mobile domain, the risks of third-party dependencies and privacy leakage have been major research themes. Derr *et al.* [14] showed apps frequently bundle outdated, vulnerable SDKs, while Pariwono *et al.* [70] demonstrated how abandoned resources expose large user bases to hijacking. Zuo and Lin [105] and Alrawi *et al.* [3] further exposed weak or malicious backend dependencies. For privacy, longitudinal analyses by Ren *et al.* [79] and Reardon *et al.* [77] revealed worsening PII leakage and pervasive cross-app tracking [30, 76]. Furthermore, Feal *et al.* [20] and Girish *et al.* [25] detailed how embedded SDKs harvest identifiers and sensor data at scale, with Meng *et al.* [60] highlighting discrepancies between stated privacy policies and actual SDK data exfiltration.

Closest to our study, So *et al.* [88] analyzed Android DNS dependencies using passive DNS, finding that 2.5% of APKs and 4.2% of apps rely on expired domains, and identifying 41 domains that were immediately registrable, some affecting top-10 apps with over 100 million downloads. While their focus is on characterizing dependency stability and prevalence of expired domains in the Android DNS footprint, our work extends this perspective by dynamically detecting which endpoints receive sensitive identifiers and analyzing TLS validity windows around renewal events. This combination allows us to move beyond counting expired dependencies to explicitly quantifying the privacy risks posed by lifecycle gaps.

3 Threat Model

We consider an adversary whose primary objective is to silently harvest sensitive user data and sensor readings by exploiting mobile app lifecycle failures. This adversary targets both first-party domains neglected by developers and third-party endpoints (e.g., analytics, advertising) that have been discontinued or restructured [21, 76]. Mobile apps are appealing targets due to their global reach, deep access to embedded sensors, and heavy reliance on data-driven monetization [8, 17]. We assume the attacker operates without compromising the device, the OS, or the app. Instead, the adversary

operates within standard infrastructure bounds, e.g., legally registering expired domains or claiming dangling cloud resources. This adversary can pose three key threats:

(1) **Continuous Tracking via (Sub)Domain Hijacking.** Mobile apps often transmit a rich set of sensitive identifiers to remote endpoints, including advertising IDs, device fingerprints, and custom user IDs [77]. Attackers can hijack this data by re-registering expired domains [50] or by claiming abandoned cloud resources (e.g., AWS S3 buckets) referenced by dangling CNAME records [53, 104]. Upon taking control of the endpoint, the adversary receives all legacy traffic from installed apps. As demonstrated in previous studies [14], while 85.6% of vulnerable third-party libraries in Android apps can be trivially upgraded without code modifications, developers rarely apply these updates, leaving legacy code active. By hijacking these endpoints, adversaries can aggregate identifiers to build cross-app identity graphs and profile users without their awareness [76].

(2) **Covert Harvesting of Permission-protected Location Data.** Beyond identifiers, apps collect coarse and fine-grained geolocation, Bluetooth telemetry, and Wi-Fi metadata for location-based services and advertising. When an endpoint changes ownership, it silently undermines Android's permission model: location access originally granted to a trusted developer is now routed to an unknown third party. Adversaries intercepting this traffic can reconstruct detailed mobility traces, inferring highly sensitive routines and addresses [25, 55]. Because the app remains unchanged, neither the user nor the OS is alerted to the new registrant harvesting permission-protected sensor data, fundamentally violating user privacy expectations [58, 66].

(3) **Supply Chain Exposure and Passive Interception.** Because third-party SDKs are embedded in thousands of mobile apps, a single SDK whose endpoint is hijacked can expose millions of users worldwide [21]. Moreover, adversaries can act opportunistically: during domain lifecycle events (e.g., late renewals), an endpoint's TLS configuration often lags behind its DNS registration, leaving periods without a valid certificate [9]. Misconfigured apps or legacy SDKs may then fall back to plaintext HTTP or accept invalid certificates, allowing network-level attackers to silently intercept identifier-bearing traffic [77].

Given that app update cycles are slow and legacy versions remain active, these abuses can persist for as long as apps keep sending traffic to expired, re-registered, or hijackable endpoints. The legal and regulatory implications of such unauthorized data flows (e.g., under GDPR or CCPA) are discussed in §6.

4 Methodology

Figure 1 summarizes the methodology we developed to achieve the objectives of this work. We combine dynamic analysis of Android apps with active DNS measurements to detect flows of sensitive user identifiers to endpoints that expire, change ownership, or become hijackable. We note that, due to the inherent limitations of dynamic black-box testing (e.g., code coverage), we do not claim to capture every possible endpoint. However, our methods provide concrete, observable evidence of privacy breaches arising from lifecycle mismanagement.

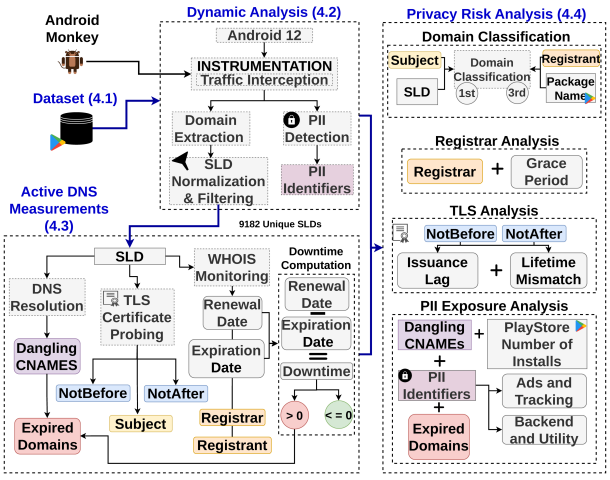

Figure 1: Overview of the methodology pipeline.

4.1 Dataset

To identify endpoints actively used by modern apps, we leveraged the AndroZoo repository [46] to collect 1,008,539 APKs uploaded after January 2023. We then filter this dataset to select 11,131 apps across three categories: (1) smart-home and IoT companion apps, (2) apps embedding wireless-scanning or beacon SDKs, and (3) a baseline random sample of 1,000 apps. For each app, we use a custom scraper to collect metadata, including its developer information and install counts.[1] Our first cohort targets apps that configure devices, proxy traffic between users, devices, and cloud endpoints, and often maintain long-lived backend connections on behalf of resource-constrained devices. Prior work shows that companion apps are an effective vantage point for discovering shared vulnerable backends and misconfigured cloud services across device families [41, 92, 99], making them natural candidates for observing persistent "zombie" traffic. Our second cohort focuses on apps that continuously monitor Bluetooth, Wi-Fi, and GPS to infer fine-grained proximity and mobility, collecting identifiers and scanning results to build long-term profiles [25], making this cohort relevant for studying the privacy impact of infrastructure decay and domain re-registrations. While health or payment apps also process highly sensitive data, prior work has predominantly focused on the semantics of personal information and sector-specific compliance [4, 57]. Our goal, instead, is to analyze how long-lived, autonomously operating dependencies behave as their domains traverse expiration, grace, and re-registration windows, for which IoT companion and wireless-scanning apps represent higher-risk environments. Finally, we include a random sample of 1,000 apps as an ecosystem baseline, ensuring that our findings are not restricted to telemetry-heavy cohorts but are grounded in general-purpose consumer applications observable from an EU vantage point as of July 2024.

[1] Install counts on Google Play represent cumulative historical downloads, not concurrently active users. They therefore constitute an upper bound on the number of users potentially affected, not a lower bound.

4.2 Dynamic Analysis

We execute each APK on a device farm comprising eight Android 9 and eight Android 12 Google Pixel 3a smartphones located in the EU, each running a proprietary instrumented AOSP build [omitted for blind review]. This setup captures two complementary outputs: the set of active network endpoints it contacts and the sensitive data it transmits—both derived from the same execution session.

Instrumentation and Execution. Our proprietary AOSP build logs all network traffic in plaintext by hooking reads and writes directly at the TLS sockets—inherently bypassing certificate pinning—and tracks runtime resource access by apps, including whether permission-protected Android APIs (e.g., location, BLE/Wi-Fi scans) are accessed. To recover transmitted values in cleartext, we decode common encodings like gzip and base64, along with bespoke obfuscation methods used by popular SDKs (e.g., Kochava, Amplitude, and Yandex). We pair this with Android Monkey [29], which generates synthetic UI inputs across 10-minute sessions per app. Our dynamic analysis pipeline handles standard login flows—including Google Single Sign-On (SSO)—to maximize code coverage and bypass registration walls. Each device is pre-loaded with pseudonymous identifiers such as a phone number, email address, and username. We also collect device-specific values (AAID, Android ID, IMEI, MAC address, and geolocation coordinates) at runtime via the respective Android APIs. We scan decoded payloads (JSON, XML, Protobuf, and URL-encoded) for all these values using string matching and regular expressions—including their MD5, SHA-1, and SHA-256 hashes—producing, for each app, a record of which sensitive identifiers each endpoint receives. Our OS-level instrumentation has two important implications for interpreting our results. First, because we intercept traffic below the application layer, an app that correctly implements strict certificate pinning would refuse to connect to a re-registered endpoint with a different certificate, rather than silently sending PII to an attacker. In that sense, the hijacking and interception scenarios should be viewed as an upper bound on what would be observable for such pinned apps. Second, however, prior work shows that certificate pinning remains rare in practice. Pradeep *et al.* find that only between 0.9% and 8% of Android apps in the wild actually deploy pinning [73].

Domain Extraction and SLD Normalization. We extracted the remote hostname from every decoded request and normalized each hostname to its Second-Level Domain (SLD), defined as the eTLD+1 under the Public Suffix List [74]. We operate at SLD granularity for two reasons:

(1) **The SLD is the registrable unit:** all subdomains share the same registrant, registrar, and expiration date as their parent SLD, so lifecycle events such as expiration, ownership transfer, and re-registration occur exclusively at this level [50, 88].
(2) **Dangling CNAME analysis scope**: a CNAME record at any subdomain becomes exploitable when the target SLD expires and is re-registered by an adversary [53, 88]. WHOIS data—our primary source for detecting expiration events—is only available at the SLD level. By anchoring our analysis at SLDs, we can directly cross-reference CNAME targets against WHOIS expiration records to identify cases where the registrable root of a pointed-to domain has lapsed and become hijackable.

With these criteria, we obtained a total of 157,913 unique (APK, SLD) pairs. We then applied a filtering step to ensure data quality: we excluded invalid DNS records, purely IP-based endpoints (2,531), and malformed entries (3,884) resulting from parsing errors or truncated streams. This process resulted in a dataset of 151,498 valid pairs spanning 9,182 unique SLDs.

4.3 Active DNS Measurements

Using the unique SLDs extracted in §4.2 and following established practices by Lauinger *et al.* [48–50] and and Zhang *et al.* [104] we performed a longitudinal measurement campaign to characterize their DNS and TLS posture over time. We focus on the 3,420 SLDs whose WHOIS expiration dates fell within our observation window (August 2024 to May 2025), allowing us to observe complete expiration and renewal events.

WHOIS Monitoring and Downtime Computation. We issued automated WHOIS queries at one-hour intervals for each of the domains throughout the observation window. From each response, we extracted the registrar name, expiration date, and renewal date when observable. The one-hour polling resolution allows us to detect renewal events with a temporal precision sufficient to characterize short-lived expiration windows. Following existing practices [50, 104], we define the *downtime* of a domain as the interval between its expiration timestamp and the first subsequent renewal timestamp observed in WHOIS:

$$\text{Downtime} = T_{\text{renewal}} - T_{\text{expiration}}. \tag{1}$$

A positive value indicates a late renewal (a window during which the domain was expired but eventually recovered). A value of zero or less indicates a proactive renewal before expiration. Domains with positive downtime are the focus of our risk analysis in §4.4.

DNS Resolution and Dangling CNAME Detection. For each SLD, we resolved DNS records and analyzed their delegation structure. We used `danglingcname` [44] to identify CNAME chains that point to deallocated or unclaimed third-party resources, such as removed AWS S3 buckets or Azure App Service endpoints [23, 89]. Such dangling CNAMEs create immediate hijackability risks (§3).

TLS Certificate Measurements. As demonstrated in prior work, the operational lifecycle of a TLS certificate is fundamentally decoupled from the underlying domain registration and DNS delegation [9, 89]. A domain may be DNS-live and WHOIS-renewed while its TLS certificate is expired, not yet issued, or misaligned with the domain's registration period. To capture this operational state, we performed the following measurement. For each SLD in our dataset, we initiated a TLS connection over port 443 with a standard timeout. Upon a successful handshake, we retrieved the server's leaf certificate in DER format and parsed its X.509 structure. From this, we extracted the certificate's `Subject`, `Issuer`, and its cryptographic validity boundaries (specifically the `NotBefore` and `NotAfter` timestamps). Endpoints that timed out or failed to negotiate a TLS session were recorded as unavailable. We use these extracted validity timestamps in §4.4 to evaluate secure-channel coverage gaps around renewal events.

4.4 Privacy Risk Analysis

In the final phase, we focused on the subset of domains with positive downtime, as these are the endpoints for which a concrete window of vulnerability can be established. For these vulnerable endpoints, we combine the lifecycle measurements from §4.3 with the payload observations from §4.2 to evaluate risk along four dimensions.

Domain Classification. We classify each domain as either first-party (owned by the app developer) or third-party (external service not controlled by the developer). Third-party domains are typically introduced into apps via external SDKs, advertising networks, and tracking services, which often operate opaquely to the user [20, 43, 76]. We employed a two-step approach: an automated heuristic baseline followed by manual verification. For the automated pass, we extracted the core SLD of each domain and cross-referenced it against the string components of the contacting apps' package names (excluding common roots like `com`, `org`, or `net`). Since package names are the standard indexing identifier for Android apps [24], a string match between the domain and the app's package name indicates first-party ownership. If no match was found, the domain was provisionally flagged as third-party. Because this automated heuristic can misclassify first-party backend domains that do not share the app's primary branding, we manually verified ownership by inspecting WHOIS registrant data, TLS certificate `Subject` and `Subject Alternative Name` (SAN) fields, and developer websites. While WHOIS data is often obscured by privacy proxies, the TLS and package name signals typically resolve ambiguities. Finally, when a domain is a CNAME pointing to a shared cloud infrastructure provider (e.g., AWS, Azure) or a Content Delivery Network (CDN), we classify it as third-party. While developers are responsible for configuring their cloud resources under a shared responsibility model, the underlying namespace and infrastructure lifecycle are controlled by the external provider, making the endpoint distinct from a first-party domain.

Registrar Analysis. We assessed whether observed downtimes were the result of registrar policy violations. We gathered and analyzed the grace periods and redemption window policies for over 80 registrars in our dataset from official documentation. We then compared the observed downtime of each domain against its registrar's specific policy to identify cases where domains remained recoverable (or vulnerable) longer than stated standards.

TLS Analysis. Using the certificate validity metadata (`NotBefore` and `NotAfter`) collected in §4.3, we evaluate the operational and security risks introduced by delayed domain renewals. By comparing these timestamps against the domain's WHOIS lifecycle, we compute two complementary risk windows:

- **Issuance Lag:** Whether the TLS certificate was valid at the time the domain was renewed. We anchor this measurement at renewal rather than the exact moment of domain expiration to capture the full exposure window. While the worst-case scenario occurs when a certificate expires before the domain (leaving the endpoint immediately vulnerable), certificates that lapse during the domain's downtime also render the endpoint unauthenticated upon renewal. In both cases, the downtime window can be actively used for hijackability since an attacker would not face a conflicting valid certificate.
- **Lifetime Mismatch:** Whether the TLS certificate remains valid for the full duration of the renewed domain's lifecycle. Because short-lived certificates can be renewed multiple times within a single registration period, we use this metric as a signal of potential coverage gaps rather than a vulnerability on its own.

PII Exposure Analysis. Finally, we correlate the per-(APK, SLD) payload observations from §4.2 with active DNS measurements from §4.3 to identify domains that both transmitted sensitive identifiers during normal operation and experienced a positive downtime window. This correlation establishes that these domains handled sensitive data in the past and subsequently entered an expired state. By cross-referencing this subset with our DNS resolution data, we isolate the highest-risk domains: those that not only collected PII but also exhibited dangling CNAMEs during their downtime. While trackers often collect richer payloads in addition to identifiers [76, 77, 79], the presence of IDs is a signal of potential tracking activity. This combination creates an active hijackability window during which residual app traffic could be redirected to an adversary who claims the abandoned cloud resource [53, 89, 104]. We group these domains into two categories based on their purpose:

(1) **Advertising and Tracking Services (ATS):** Domains whose primary business purpose relies on collecting user data for profiling, monetization, or analytics [76]. This group includes third-party SDKs that collect advertising identifiers (e.g., AAID), device fingerprints, and geolocation data (both coarse and fine) to build movement traces and behavioral inferences.
(2) **Backend and Utility Services:** Domains facilitating core app functionality, diagnostics, or cloud storage. These endpoints typically collect app-specific or device metadata (e.g., Firebase IDs, OS version, device model) required for the app to operate.

This classification allows us to distinguish between risks driven by commercial third-party tracking infrastructure and risks arising from the mismanagement of an app's own operational backend.

5 Results

This section presents our findings on the privacy impact of domain lifecycle mismanagement in mobile apps. We first characterize the domain lifecycle outcomes observed during our longitudinal study (§5.1). We then examine the post-expiration risk factors affecting late-renewed domains (§5.2). Finally, we quantify the privacy exposure by mapping these vulnerability windows to the dissemination of sensitive user identifiers (§5.3) and discuss case studies (§5.4).

5.1 Domain Lifecycle Outcomes

We first characterize how domains behave around their expiration dates. Figure 2 shows the Cumulative Distribution Function (CDF) of downtime (§4.3) on a symmetric-log x-axis, which allows us to simultaneously visualize early and late renewals. For domains renewed before expiration, the median anticipation was -718.8 hours (mean -672.9 ± 567.1 hours), with the extreme best-case domain, `sleepio.com`, being renewed $7,288.1$ hours (≈ 10 months) in advance. Conversely, for domains renewed late, the median delay was 16.5 hours (mean 41.5 ± 323.6 hours), with the extreme worst case remaining expired for exactly one year before recovery (Case Study 6 in §5.4). The red curve highlights the behavior of

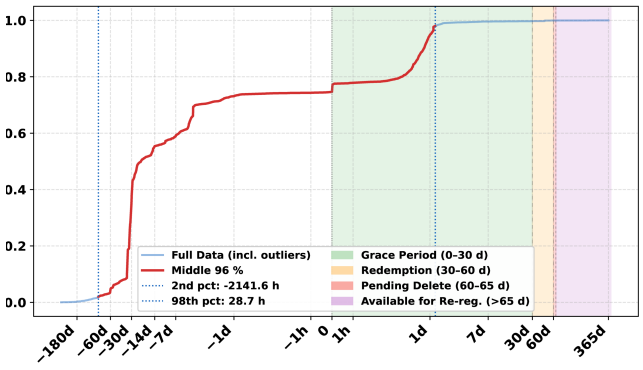

Figure 2: CDF of domain downtime on a symmetric-log x-axis labeled in days.

the middle 96% of the dataset, bounded between the 2nd and 98th percentiles (−2, 141.6 and 28.7 hours, respectively). Between these bounds, the CDF shows that the vast majority of domains are proactively renewed well before expiration, consistent with registrar reminder-driven billing cycles. A secondary cluster of renewals occurs around zero, corresponding to organizations that rely on reactive, last-minute automation that triggers upon or just after expiration [9, 53]. For positive values, the shaded bands map each late renewal to its corresponding ICANN phase [33]. Overall, 25.3% of the monitored domains failed to renew in time. Among these non-renewed domains, the majority were eventually renewed late, and only 3 of them remained inactive as of our last observation (June 2025). Another 3 did not have a subsequent renewal event, in the sense that their WHOIS records were removed and DNS resolution consistently failed.

From an operational security perspective, any positive downtime is a risk indicator, since the domain's behavior during this period is governed by the registrar's grace-period policies (§5.2). Best practices dictate that critical infrastructure domains should be proactively renewed well before expiration to avoid entering registrar-controlled grace periods, during which DNS resolution may fail or the domain may become eligible for third-party re-registration [48, 49]. The prevalence of these failures is consistent with, and complementary to, prior ecosystem-wide measurements by So *et al.* [88]. Their study, which analyzes DNS footprints over many years using passive DNS, reports that at any given point in time, on average ≈ 2.0%–2.5% of APKs and ≈ 3.3%–4.2% of apps rely on at least one expired domain in their dependency footprint. They further show that these expired dependencies can affect thousands of APKs and hundreds of apps concurrently, and that app updates do not systematically remove them. From a domain-centric perspective, we find that a quarter of all observed domains experience at least one late renewal event within a 10-month window, and that many of these events last long enough (hours to days) to plausibly disrupt service or degrade security. Together, these results indicate that expired domains are not isolated anomalies but a systemic property of the mobile app supply chain.

Table 2: Renewals by Days After Expiration.

Days After Expiration	Count	First Party	Third Party
Same Day	686	139	547
1–7 Days	161	42	119
7–30 Days	3	0	3
More than 30 Days	11	2	9

To better understand the severity of late renewals, Table 2 groups the late-renewed domains by the number of days elapsed since expiration. Most late renewals (79.7%) occurred on the same day as expiration, often within a few hours. These cases likely correspond to automation or billing issues that are resolved quickly. However, the behavior of a domain during even a same-day expiration window is entirely governed by the registrar's implementation of the ICANN ERRP policy and is far from uniform.[2] We find that 18.7% of them were renewed within the first week after expiration, extending these windows to multiple days, while a small fraction (1.6%) remained expired for more than a week, with 11 exceeding 30 days.

Having established the prevalence and duration of expiration events, we now turn to the behavior of domains that avoid these failures. The left side of Figure 2 captures the 74.4% domains that were proactively renewed before their expiration date. These proactive renewals cluster around three windows that align with typical registrar reminder schedules:

- **30–60 days before expiration**: The largest group, with 1,077 domains (42.3%), aligns with the initial billing cycle triggered by registrar reminder emails. It is dominated by high-availability infrastructure such as `github.com`, `googletagmanager.com`, and `firebaseio.com`, consistent with enterprise operations that treat domain names as critical digital assets [34].
- **14–30 days before expiration**: A secondary peak of 661 domains (26.0%) seems to reflect coordinated management by large organizations, synchronizing renewals across related domains on the same billing cycle (e.g., Adobe and Samsung sub-brand portfolios). Consumer-facing brands and SDK providers such as `adidas.com` and `apple.com` also fall within this range.
- **1–7 days before expiration**: A "last-minute" cluster of 481 domains (18.9%), including 50 renewed on the exact day of expiration, suggesting reactive, billing-triggered automation. This batch is dominated by ad-tech and analytics domains renewed with minimal margin. From a security standpoint, any billing failure (an expired credit card, a missing invoice approval, or a registrar outage) would have caused these domains to enter expiration. While registrar grace and redemption periods (§5.2) delay third-party re-registration, the domain may become unresolvable during this interval, disrupting service for all mobile apps that depend on it, opening a concrete hijacking window [33, 35, 49].

[2] Depending on the registrar, an expired domain may resolve normally, serve a registrar-controlled parking page, or return an NXDOMAIN entirely, all while remaining formally outside the registrant's control [33, 49, 50]. In all three cases, any mobile app still sending traffic to that endpoint during this window is either communicating with an unintended third party, experiencing a hard failure, or exposing its requests to interception.

At the other extreme, domains renewed 60–180 days before expiration (e.g., `alibaba.com`, `zara.com`, `plex.tv`) reflects deliberate, multi-year brand-protection strategies managed by enterprise-grade registrars such as MarkMonitor and CSC [34]. The SSAC has repeatedly highlighted that registrants who do not adopt such proactive lifecycle management expose themselves to both service disruption and domain hijacking [35]. These patterns show that when renewal workflows function as intended, domains are typically renewed in a narrow window driven by registrar notifications and automation. The 25.3% of domains that miss this window (and, in some cases, remain expired for days or weeks) represent a significant deviation from expected behavior.

5.2 Post-Expiration Risk Factors in Late-Renewed Domains

In this section, we examine how late-renewed domains differ by ownership (first- vs. third-party), how they are exposed through DNS misconfigurations, and how registrar policies shape the duration of their expired states. Among the 861 domains that were renewed after expiration, late renewals are heavily skewed toward third-party infrastructure: 678 (78.7%) were classified as third-party while 183 (21.3%) as first-party. This imbalance persists across all delay ranges in Table 2: even for the most extreme delays (more than 30 days), third-party domains accounted for 9 out of 11 cases. These domains are far from marginal; together, the 861 late-renewed domains are referenced by apps with Google Play install brackets totaling over 78 billion cumulative downloads. In other words, the bulk of lifecycle mismanagement falls on shared services and SDK backends that sit at the core of the mobile supply chain, rather than on isolated app-specific endpoints.

Popularity and User Impact. To estimate the potential user impact of these expiration events, we correlate domain downtime with their global popularity using the Tranco top-1M ranking [51]. We relied on Tranco over the Chrome User Experience Report (CrUX) because CrUX is web-centric and blind to the REST APIs, GraphQL endpoints, and background telemetry hosts used by mobile SDKs, whereas Tranco aggregates passive DNS traffic from recursive resolvers (e.g., Cisco Umbrella), capturing mobile infrastructure footprints. Figure 3 presents a scatter plot of the late-renewed domains, with Tranco rank on the x-axis (log scale, lower rank indicating higher popularity) and post-expiration downtime in hours on the y-axis (log scale).[3] Domains not appearing in the Tranco top-1M are grouped in the rightmost "Unranked" bin (251 domains). Remarkably, 610 (70.8%) of the expired domains appear in the Tranco top-1M, and 122 (14.2%) sit within the elite top-10K—109 of which are third-party services, including core ad-tech endpoints ranked in the top-200. Ten domains in the top-10k remained formally expired for more than 24 hours, demonstrating that expiration is not limited to low-visibility or abandoned domains.

[3]Because top-site rankings fluctuate naturally over time [80], tying longitudinal measurements to a specific list snapshot is standard practice for reproducibility [51]. We anchor our analysis to the Tranco list generated on January 29, 2025 (the median expiration date in our dataset) as this statistically minimizes the average temporal drift between the actual expiration events and our popularity snapshot.

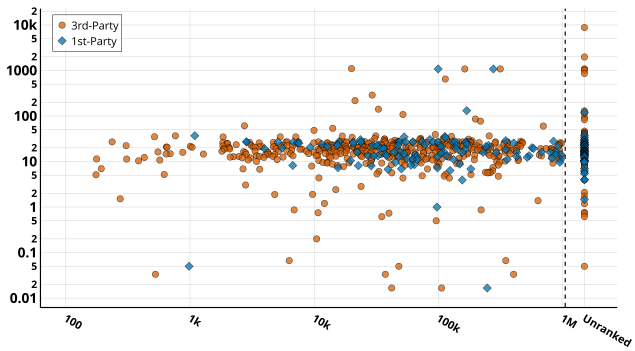

Figure 3: Domain popularity ("Unranked" indicates not present in the Tranco list) versus post-expiration downtime.

DNS Risks and Application Exposure. Late-renewed domains are also frequently correlated with fragile DNS configurations. During their expired periods, 218 of the 861 domains (25.3%) exhibited at least one dangling CNAME record, i.e., a CNAME pointing to deallocated or non-existent cloud resources, creating an opportunity for subdomain hijacking [53]. To assess whether these windows were actively leveraged, we queried CT logs via crt.sh [85] for all 218 domains, checking whether any TLS certificate was issued within the downtime interval. Of the 218 domains, 34 (15.6%) received at least one certificate during their lapse (91 unique certificates in total after deduplicating precertificate leaf pairs). The prevalence of dangling CNAMEs among late-renewed domains indicates that expiration events often coincide with incomplete teardown of dependent infrastructure. Matching our dataset against the Tranco top-1M list that we used in Figure 3 reveals that 196 of the 218 domains with dangling CNAMEs rank in the top 1M, and 50 are in the top 10K. Table 3 lists 15 popular domains that exposed a high number of dangling CNAMEs during their downtime, ordered by Tranco rank, where the most striking example is `t.co` (Case Study 2 in §5.4). Beyond consumer platforms, the issue also affects the supply chain of mobile ad-tech (`fyber.com`, a major mobile advertising monetization SDK acquired by Digital Turbine [15]), digital publishing (`pagesuite.com` used by newspapers to distribute e-editions to mobile apps [69]), and enterprise management (`smartenspaces.com`, an AI-driven hybrid workplace and desk-booking platform used by some Fortune 500 companies [86]). Major first-party brands are also similarly vulnerable: `porsche.com` (rank 2,852) exposed 39 dangling CNAMEs, while `mediaset.net` (rank 27,049) exposed 17 over only 13.4 hours. The CT logs further corroborate these risks: the majority of the 91 certificates were issued by automated DV providers (Let's Encrypt 50%, Google Trust Services 22%, Amazon ACM 14%). Two particular cases provide the strongest observed individual signals: `fibaro.com` (Case Study 4, §5.4); and `californiapsychics.com`, which received a Let's Encrypt certificate whose CN is `blackbears.polo-development.com`, an entirely unrelated domain, while the SAN field was found pointing to `cpeducation.californiapsychics.com`, a cross-domain bundling pattern indicative of opportunistic certificate harvesting. High-profile domains were also affected: `teladoc.com` (rank 2,753) accumulated 13 unique certificates, including regional wildcards,

Table 3: Top domains exhibiting positive downtime and dangling CNAME records, illustrating subdomain takeover risk.

Domain	Downtime	Rank	Installs	CNAMEs	Party	Registrar
t.co	5.1h	175	46B+	4	3rd	CSC Corporate
bbb.org	25.1h	1,991	1k+	70	3rd	CSC Corporate
fyber.com	12.5h	2,106	42M+	37	3rd	INWX GmbH
porsche.com	27.0h	2,852	10k+	39	1st	PSI-USA
takeaway.com	26.2h	3,023	13M+	15	3rd	Metaregistrar
sumup.com	13.3h	6,748	1k+	15	3rd	United-Domains
stjude.org	24.1h	11,493	5k+	64	1st	CSC Corporate
informa.com	19.9h	12,006	30k+	21	3rd	SafeNames
exlibrisgroup.com	12.3h	13,601	219k+	16	3rd	GoDaddy
audi.com	27.0h	16,075	600k+	15	3rd	1API GmbH
tiaa.org	25.2h	20,571	100k+	18	1st	CSC Corporate
mediaset.net	13.4h	27,049	50k+	17	1st	Register SPA
auction.com	8.0h	28,779	210k+	21	1st	GoDaddy
pagesuite.com	31.9h	30,670	63M+	15	3rd	GoDaddy
smartenspaces.com	25.9h	958,585	15k+	105	3rd	GoDaddy

during its 4.5-day downtime, while `takeaway.com` and `sumup.com` (Table 3) had certificates issued for production API and authentication endpoints. Whether these certificates represent active adversarial exploitation or automated cloud provisioning, they demonstrate that domain-validation control was not held by the original registrant, yet valid TLS certificates were successfully obtained.

Registrar Behavior and Renewal Patterns. To understand how registrar policies contribute to these risks, we analyze the 88 registrars responsible for the late-renewed domains. Table 4 lists a subset of 17 registrars that account for the highest volume of late renewals or whose domains exhibit the longest observed downtimes. For each registrar, we report the number of expired domains by ownership, the average and maximum domain downtime observed across those domains, and the officially documented grace period (where available). Across these registrars, 6 exhibited at least one case where the observed downtime of a domain exceeded the registrar's published grace period: GoDaddy, Key-Systems GmbH, Domain Robot, NameCheap, Register SPA, and Hostinger. GoDaddy emerges as the dominant registrar, managing 453 domains renewed after expiration, including 350 third-party (77.3%) and 103 first-party (22.7%). Despite this scale, GoDaddy's average downtime remains relatively low at 0.8 days, though we observe highly popular domains falling through the cracks: textttdisqus.com (Tranco rank 518), a ubiquitous commenting SDK acquired by Zeta Global in 2017 [47]; or `jeepselectedforyou.in` (Case Study 3, §5.4). NameCheap and Hostinger also stand out for individual cases in Table 5: NameCheap allowed `weathertrackertv.live` to remain expired for a full year (Case Study 6, §5.4), while Hostinger permitted `tigerkreemz.online` to linger for 82.5 days [32]. Table 5 also includes `net.in` and `co.in`, foundational shared infrastructure domains under the Indian ccTLD managed jointly by NIC and Neustar's registry business [28]; `net.in` alone supports 15 APKs with over 69 million cumulative installs and remained expired for 45 days. Similarly, `metro.taipei` (the Taipei Metro's official domain [91]) and `ncdhhs.gov` (North Carolina's Department of Health and Human Services [67], managed by CISA) experienced 27 and 12 days of downtime, respectively. While public documentation confirms the existence of grace periods for these entities [36, 97], their exact durations are often opaque, and the combination of long-lived expiration windows with critical public-facing services creates trust and impersonation risks.

Table 4: Registrars with late renewals. Domain Downtime (Avg./Max) and Grace Period are reported in days.

Registrar	1st P.	3rd P.	Avg.	Max.	Grace Period
GoDaddy	103	350	0.8	42.0	18.0 [27]
OVH	10	39	0.6	1.7	30.0 [68]
1API GmbH	4	34	0.6	9.1	30.0 [1]
IONOS	6	23	0.6	1.1	30.0 [39]
Key-Systems GmbH	6	22	2.4	46.0	30.0 [42]
ASCIO	7	18	0.8	1.1	30.0 [5]
SafeNames	4	15	0.7	2.5	30.0 [81]
Domain Robot	5	14	3.0	45.0	45.0 [38]
NameCheap	0	12	33.8	365.0	30.0 [65]
Register SPA	6	6	8.2	45.0	20.0 [78]
Wild-West Domains	0	9	1.0	2.5	18.0 [101]
Hostinger	0	1	82.5	82.5	30.0 [32]
Ubilibet	1	0	45.0	45.0	Unknown[95]
Neustar 1	0	1	45.0	45.0	Unknown
NIC	0	1	45.0	45.0	Unknown[36]
Taipei CG	0	1	27.0	27.0	Unknown
CISA	0	1	12.0	12.0	Unknown[97]

Table 5: Domains with extended registration downtime (≥30 days), representing domain re-registration takeover risk.

Domain	Downtime	Tranco Rank	Installs	CNAMEs	Party	Registrar
weathertrackertv.live	365.0d	–	100	0	3rd	NameCheap
tigerkreemz.online	82.4d	–	50k+	0	3rd	Hostinger
solocoo.tv	45.9d	19,958	500k+	1	3rd	Key-Systems
net.in	45.0d	–	69M+	1	3rd	NIC
ynap.biz	45.0d	315,093	1M+	4	3rd	Register SPA
bonpreuesclat.cat	45.0d	277,247	5k+	0	1st	UBILIBET
yoox.biz	45.0d	99,558	100k+	0	1st	Register SPA
co.in	45.0d	–	3M+	0	3rd	Neustar
komm.one	45.0d	163,048	100	0	3rd	Domain-Robot
jeepselectedforyou.in	42.0d	–	1k+	0	3rd	GoDaddy
udownloader.vip	35.6d	–	100	0	3rd	NameCheap

TLS Certificate Validity. Beyond DNS and registrar behavior, TLS certificate management introduces a critical dimension of risk during and after renewal gaps. To successfully and silently intercept HTTPS traffic, which the vast majority of mobile apps use, an attacker needs not only to re-register an expired domain but must also be able to present a valid TLS certificate. If a hijacked domain lacks a valid certificate, modern clients will drop the connection. Therefore, understanding the TLS lifecycle is essential to determine whether an expired domain can be practically exploited for silent traffic interception. For the 861 late-renewed domains, TLS metadata could be retrieved for 648 domains (75.3%). The remaining 213 (24.7%) either did not support TLS or could not be resolved over HTTPS at measurement time. We assess TLS validity along two axes defined in §4.3: issuance lag and lifetime mismatch. Among the TLS-enabled domains, 146 (17.0%) exhibited a post-renewal issuance delay, meaning that the domain was reachable again but lacked a valid certificate for a non-trivial period. These issuance lags ranged from several hours to over 95 days, creating concrete windows in which clients attempting to reconnect could encounter certificate errors or, in misconfigured apps, fall back to unencrypted or weakly validated connections [9, 19, 77]. Lifetime mismatches were even more common: In 607 cases (70.5%), the certificate's expiration date preceded the renewed domain's expiration, setting up future points where the secure channel might break even though the domain remains valid. Only 23 domains (2.7%) showed no TLS

Table 6: Top 10 domains affected by both Issuance Lag and Lifetime Mismatch, sorted by issuance delay.

Domain	Issuance Lag	Lifetime Mismatch	Issuer
jeepselectedforyou.in	95 days	215 days	Google Trust Services
tigerkreemz.online	55 days	209 days	Let's Encrypt
yoox.biz	39 days	190 days	Let's Encrypt
openevse.com	25 days	249 days	Let's Encrypt
bluecode.com	24 days	249 days	Let's Encrypt
auryc.com	23 days	250 days	Google Trust Services
lightningfighter2.net	23 days	250 days	Google Trust Services
electricnow.tv	23 days	251 days	Let's Encrypt
eprintinguk.com	21 days	252 days	Let's Encrypt
jswipeapp.com	20 days	252 days	Google Trust Services

Table 7: Top App / Domain Offender Crosswalk: most-downloaded apps and the late-renewed domains they reference.

Application	Developer	Installs	#Dom.	#Dang.	#LT$_{\neq}$	Max DT
Google News	Google	1 B+	13	5	10	27 h
Google Search Lite	Google	1 B+	3	2	3	22 h
MS Outlook	Microsoft	1 B+	1	1	1	5 h
Beat Hopper	Amanotes	500 M+	4	2	3	21 h
HP Print Service	HP Inc.	500 M+	2	1	2	15 h
Duolingo	Duolingo	500 M+	1	1	1	5 h
My Verizon	Verizon	100 M+	7	4	2	31 h
uTorrent	Rainberry	100 M+	5	3	3	21 h
Wattpad	Wattpad.com	100 M+	4	1	4	19 h
innisfree	innisfree	50 M+	7	2	3	22 h

validity issues along either axis, suggesting that coordinated lifecycle management of domains and certificates is rare in practice. Crucially, 128 domains (14.9%) suffered from both vulnerabilities simultaneously. Across our dataset, these dual-fault endpoints are actively referenced by 503 APKs whose aggregate install brackets total over 1.59 billion downloads. Table 6 illustrates the intersection of these problems by listing 10 domains affected by both issuance lag and lifetime mismatch, in decreasing order of issuance lag. Several of these domains also appear in our previous analyses, showing that long issuance lags and misaligned certificate lifetimes often coincide with extended domain downtime and non-trivial user exposure. Moreover, our dataset includes cases such as `elpais.com` (a major international news outlet [71] affecting 100k+ installs), `flaticon.com` (a widely used vector icon resource [13] with 50k+ installs), and `urbanoutfitters.com` (a global retail brand [96] with 100k+ installs). While their issuance lags are shorter than the most extreme cases in Table 6, they represent gaps on production services. Table 7 lists the ten most-downloaded apps in our dataset alongside the number of distinct late-renewed domains they reference (**#Dom.**), the subset exhibiting dangling CNAMEs (**#Dang.**) and TLS lifetime mismatches (**#LT$_{\neq}$**), and the longest observed downtime (**Max DT**). Google News stands out: with over one billion installs, it contacts 13 distinct domains that experienced late renewals, five of which exposed dangling CNAME records and ten of which have TLS lifetime mismatches. Even single-domain exposure can be significant at scale: the 500M-install Duolingo app and the 1B-install Microsoft Outlook both depend on `t.co` (case study 2 in §5.4), confirming that rare domain-lifecycle failures are amplified by the third-party SDK supply chain.

Table 8: Top 15 domains with the longest downtime, including PII collected, TLS support (✓/×), and party ownership.

Domain	Downtime	Identifiers	TLS	APKs	Dangling	Party
net.in	45.0d	A	×	15	1	3rd
co.in	45.0d	A	×	9	0	3rd
metro.taipei	27.0d	F	×	1	4	3rd
cuebiq.com	27.1h	A, RM, RSM, RSS, RS	✓	19	0	3rd
sponsorpay.com	23.6h	A	×	1	6	3rd
playags-games.com	23.5h	A	×	2	0	3rd
kazandirio.io	22.9h	A	✓	1	0	1st
tivoli-envr.com	22.5h	F	×	1	0	3rd
playlnk.io	22.4h	A	×	4	0	3rd
akinator.com	22.3h	A, F	✓	1	0	1st
singlespot.com	21.9h	A	×	4	0	3rd
startappservice.com	21.9h	A	✓	112	11	3rd
crwdcntrl.net	20.9h	A	×	51	0	3rd
mobincube.com	19.6h	A, CG, FG	✓	87	2	3rd
kochava.com	15.7h	A, ID, BT, RM, RS	✓	261	3	3rd

Identifiers: A: AAID; F: FID; ID: Android ID; CG/FG: Coarse/Fine Geolocation; BT: Bluetooth; RM: Router MAC; RS: Router SSID; RSM: Router Scan MAC; RSS: Router Scan SSID.

5.3 Privacy Exposure Risks

In this section we focus on late-renewed or expired domains that both experienced measurable downtime and were observed transmitting at least one type of personally identifiable information (PII) during our dynamic analysis.[4] If they enter expired or misconfigured states while apps continue to send data, attackers who hijack or re-register them may inherit ongoing PII flows from legacy app versions. Table 8 lists the top 15 such domains, along with the types of identifiers observed, their TLS support status, the number of contacting APKs, the count of dangling subdomains, and their ownership party. The Advertising ID (AAID) is the most common identifier, appearing in 13 out of 15 cases. Location data (coarse or fine geolocation) is present in 5 domains, often combined with persistent identifiers, and more sensitive hardware- or network-level identifiers such as IMEI, Bluetooth MAC address, or router MAC/SSID. Even though many of the downtime windows are on the order of tens of hours, the presence of persistent identifiers and location data means that any interception or hijacking during these windows could reveal stable device identities and mobility patterns, rather than transient session information. We classify the domains based on their primary business purpose and operational role within the app ecosystem, rather than solely on the type of identifiers they collect, as persistent identifiers (like the Android ID or the AAID) are often dual-use: They can be harvested for cross-app profiling, but are also frequently collected by infrastructural services for rate-limiting, attribution, or fraud prevention. Based on this classification, our analysis reveals two main groups:

(1) **Advertising and Tracking Services.** This group consists of domains whose core business model relies on cross-app profiling, data monetization, and targeted advertising. Collectively, these domains are embedded in 1,127 applications whose aggregate install brackets total over 5.9 billion downloads. A significant portion of this exposure is driven by major ad-exchange and data management platforms. `Startappservice.com` (Start.io)

[4] In line with established works on mobile privacy and tracking [17, 76, 77], we define PII as any data capable of uniquely identifying a user or device, tracking their physical movements, or correlating their activities across applications.

and crwdcntrl.net (Lotame) act as dedicated advertising backends, collecting AAID payloads across 112 and 51 apps with 74.7 and 39.4 million installs, respectively. The risks associated with these endpoints are not just theoretical. For example, Start.io SDKs have previously faced enforcement actions and blocklists from Google Play for secretly collecting and selling device location data without programmatic consent checks [12]. Another case that stands out is Kochava.com (Case Study 1, §5.4). Mobility analytics domains like cuebiq.com (Case Study 5, §5.4) also collected both coarse and precise geolocation data.

(2) **Backend and Utility Services.** Domains in this category provide core infrastructure functionality such as crash reporting, bot protection, cloud storage, or link routing. While their primary purpose is operational rather than ad-targeting, they still aggregate massive volumes of persistent identifiers from 3,870 apps whose install brackets total over 50.4 billion downloads. For instance, t.co experienced over 5 hours of downtime while being contacted by 3,052 apps in our dataset, leaking Android IDs (Case Study 2, §5.4). Similarly, newrelic.com, a performance monitoring and diagnostics service, was observed in 562 apps accounting for 2.0 billion installs. Despite being a diagnostic tool, it suffered a 5-hour downtime window during which it collected coarse and fine geolocation data, AAIDs, and Android IDs. This highlights a classic over-collection gray area: utility backends harvesting sensitive location and tracking data far beyond their core operational requirements. Other notable examples include perimeterx.net (bot protection, 63 apps, 154.7 million installs) leaking Android IDs, and wzrkt.com (the official telemetry endpoint for CleverTap CRM, formerly WizRocket, 21 apps, 62.9 million installs) collecting AAIDs and Firebase Installation IDs (FIDs). The exposure of CRM and marketing backends like CleverTap carries significant real-world weight. CleverTap was at the center of a massive 2018 privacy scandal when security researchers revealed that the official app of the Indian Prime Minister was silently exfiltrating citizens' personal data, device information, and location metrics to CleverTap's third-party servers without consent [45].

An important attack vector across these groups is the presence of dangling subdomains, as discussed in §5.2. Several domains in Table 8, such as sponsorpay.com, mobincube.com, and metro.taipei, have multiple subdomains that point to deallocated cloud resources while still being referenced in APKs. If an attacker claims the underlying cloud resource (e.g., by creating a storage bucket or application endpoint at the target hostname), they can receive traffic originally intended for the legitimate service, including AAIDs, FIDs, and location data. Because these subdomains often belong to SDKs or shared backends, a single successful hijack can aggregate traffic from many apps at once. TLS support further modulates the privacy risk. Roughly half of the domains in Table 8 lacked TLS during their operational window, meaning that any residual communication with these endpoints (whether or not the domain was fully expired) would be transmitted in plaintext and subject to passive interception on the network path. For domains that did support TLS, the lifecycle misalignments observed in §5.2 create additional opportunities for misconfigured clients to fall back to insecure behavior or to accept invalid certificates during downtime and recovery. Overall, these results indicate that a subset of expired or late-renewed domains not only underpin critical third-party and backend services, but also collect rich combinations of identifiers and context data. While the number of such domains in our observation window is limited compared to the entire ecosystem, their concentration of PII and their role as shared dependencies suggest that they form a plausible and concerning attack surface, especially in the presence of DNS and TLS misconfigurations.

5.4 Case Studies

In this section, we consolidate and describe six case studies that illustrate distinct failure modes.

Case 1: Kochava. Kochava is a mobile attribution and analytics platform embedded in 261 apps whose install brackets total 650 million downloads, including Amazon Kindle, PlayIt, and several 100M-tier titles. Its domain kochava.com expired on April 7, 2025 and was renewed 15.7 hours later, during which three dangling CNAMEs pointed to deallocated cloud endpoints (analytics.kochava.com, looker.kochava.com, collective.kochava.com). During our dynamic analysis, Kochava collected five distinct identifier types simultaneously: AAID, Android ID, Bluetooth MAC, Router MAC, and Router SSID. Kochava is the subject of a landmark FTC lawsuit for selling precise geolocation data that tracked users to reproductive health clinics and domestic violence shelters [11]. An attacker who claimed the dangling cloud resource during the 15.7-hour window could have inherited this same multi-identifier data flow from hundreds of apps.

Case 2: t.co. The domain t.co (Tranco rank 175) is X's (formerly Twitter) mandatory URL shortener, systematically embedded in the platform's mobile clients and third-party integrations [102]. It expired on April 25, 2025 and was renewed 5.1 hours later. During this window, four dangling CNAMEs pointed to deallocated CloudFront distributions (albtls.t.co, albtls-dc.t.co, albtls-na.t.co, albtls1.t.co). In our dataset, t.co is referenced by 3,052 apps spanning install brackets totaling 46.8 billion downloads—including Google Search (10B+), Microsoft OneDrive (1B+), Outlook (1B+), Duolingo (500M+), and Google Earth (500M+). Our dynamic analysis observed Android ID transmission to this endpoint. Because CloudFront distributions can be claimed by any AWS account if the original owner releases them, a successful hijack of even one of these four CNAMEs during the 5.1-hour window could have allowed an attacker to intercept or redirect user clicks across thousands of apps globally.

Case 3: Jeep India. The domain jeepselectedforyou.in is nmthe official online platform for Jeep India's certified pre-owned vehicle business [10], managed by GoDaddy. It expired on September 17, 2024 and was not renewed until October 29—a 42-day lapse which is more than twice of GoDaddy's documented 18-day grace period [27]. During this period, no TLS certificate was in place. After eventual renewal, the first new certificate was not issued until February 1, 2025—a 95-day issuance lag, the longest in our entire dataset (Table 6). The certificate also exhibits a 215-day lifetime mismatch, meaning it will expire well before the domain's next renewal date, creating TLS coverage gaps if renewal fails. This domain appears in both Tables 5 and 6, occupying the extreme end of every lifecycle axis we measure.

Case 4: Fibaro. Fibaro is a Polish IoT home-automation platform whose mobile app (`com.fibaro.homecenter`, 100k+ installs) manages smart-home devices via the subdomain `*.cloud.fibaro.com`. The parent domain `fibaro.com` expired on August 20, 2024 and was renewed 23.9 hours later. During this window, the wildcard CNAME `*.cloud.fibaro.com` dangled, pointing to a deallocated hosting endpoint at `v66.c3.dhosting.pl`. Our CT log query revealed that a DV certificate was issued by the Polish CA Certum for `*.cloud.fibaro.com` and `cloud.fibaro.com` at 10:50 UTC on August 20—just 3.7 hours after expiration—with the SAN exactly matching the dangling record. This is the strongest signal in our dataset: a certificate whose subject corresponds precisely to the dangling subdomain, issued during the exact expiry window, by a CA completing automated domain validation against an endpoint no longer controlled by Fibaro.

Case 5: Cuebiq. Cuebiq is a mobility analytics platform that officially deprecated its public SDK in 2021 in favor of privacy-preserving alternatives [82]. Our dynamic analysis detected 19 apps (20.8 million installs) actively transmitting AAID, Router MAC, Router SSID, and Router Scan MAC/SSID to `cuebiq.com`—the richest combination of location-relevant identifiers in Table 8. The domain expired on January 4, 2025 and was renewed 27.1 hours later. Following renewal, its new TLS certificate was not issued until January 7 (1.8-day issuance lag), and that certificate expires on April 7, 2025—637 days before the domain's own expiration, increasing the risk of extended periods without a valid certificate if automated renewal fails. The observed downtime is consistent with a ticket-based resolution taking 1–2 working days, especially when accounting for time zone differences. However, if notification emails are missed or the relevant team is deprecated (as implied by Cuebiq's SDK shutdown), the domain may be lost entirely to hijackers.

Case 6: Weather Tracker. The domain `weathertrackertv.live` belongs to Weather Tracker TV, a 24/7 live streaming weather channel serving the Dallas-Fort Worth metropolitan area via mobile apps, Roku, Apple TV, and Amazon Fire TV [100]. Managed by NameCheap, the domain expired on November 21, 2023 and was not renewed until exactly one year later (365 days), the longest downtime period in our entire dataset. No TLS certificate was available during this period. While the associated Android application (`com.app.appd7ebd6096de8`, 100+ installs) is a small-scale deployment, this case is significant because it demonstrates the absolute failure of registrar renewal safeguards: the domain remained in a formally expired state for 12× the registrar's published grace period without being released for public re-registration or reclaimed. During this entire year, any user of the Weather Tracker TV app would have been sending requests to an endpoint whose ownership status was ambiguous, creating a sustained window for phishing or traffic interception.

6 Discussion

Our findings demonstrate that domain renewal lapses are a persistent systemic failure in the mobile ecosystem, particularly among third-party infrastructure. Across 3,420 Android apps domains, we observed delayed renewals, TLS lifecycle gaps, and residual PII collection risks. While prior work has quantified the volume of expired domains [88], our work advances the state of the art by analyzing the specific, privacy-related payloads transmitted during these vulnerability windows. Despite registrar grace periods, a significant number of domains renew late or face abandonment, risking hijacking, residual trust abuse, and data leakage. The core of the problem lies in the decoupling of lifecycles: mobile apps are long-lived static binaries, whereas their backend cloud infrastructure is dynamic and ephemeral [14, 23, 70, 88]. When these timelines diverge, privacy guarantees are at risk.

6.1 Long-tail Risks: Zombie Apps and Abandoned Infrastructure

Beyond late renewals, a subset of dependencies experienced permanent or highly prolonged failures. Between our initial measurement window ending in May 2025 and a follow-up manual analysis in February 2026, we observed two distinct long-tail failure modes: permanent corporate abandonment and delayed recovery. Four domains vanished entirely from the WHOIS database following prolonged expiration. `qoo10.in`, a regional endpoint associated with an app boasting over 50 million installs, was permanently abandoned following the financial liquidation of its parent company in late 2024 [6, 93]. The remaining three (`reshbagonline.com`, `lamouai.com`, `goorb.in`) appear to be malicious or low-quality domains that were seized or "burned."[5] While DNS failures prevent active PII exposure during these states, embedded SDKs in legacy "zombie apps" attempt connections to the dead backends. Although residual traffic no longer placed these domains in the February 2026 Tranco rankings, this passive DNS noise publicly advertises a hijackable data stream to potential drop-catchers. Conversely, we also observed cases of extreme, yet temporary, administrative negligence. `moorparkca.gov` (the official portal for Moorpark, California, managed by CISA and present in an app with 5,000+ installs) and `mdoc.one` (a German digital health provider acquired by CompuGroup Medical [59], managed via INWX GmbH and present in an app with 10 million+ installs) remained unrenewed during our June 2025 measurements. By February 2026, both had been restored after 193 and 30 days of downtime, respectively, regaining Tranco rankings of 976,215 and 3,019,602. These delayed recovery cases are arguably more dangerous than permanent failures: relying apps fail for months, potentially retrying connections, before the backend silently resumes accepting traffic without re-authenticating the infrastructure's integrity.

6.2 Privacy and Data Sovereignty Risks

Under GDPR Article 6, processing personal data is lawful only if a legal basis applies [18]. Unlike standard data sovereignty violations, where a developer actively (although illegally) routes data to an unauthorized region or recipient, domain lifecycle failures are uniquely dangerous because the unauthorized transfer is *invisible* and *involuntary*. The app binary remains identical: no code changes, no policy update is triggered, and neither the developer nor the user is notified that the data receiver has changed. When a domain changes ownership through acquisition or re-registration, the new recipient inherits live identifier streams without ever establishing

[5] `reshbagonline.com`, for instance, is a typo-squatting domain impersonating the luxury resale site *Rebag* [94], likely hardcoded into fraudulent apps before being suspended by its registrar.

an independent lawful basis for processing them. The practical consequence is that expired or hijacked tracking domains can be absorbed by data brokers or malicious actors to construct illicit cross-app identity graphs [76, 77]. Consequently, privacy compromise in this context can be seen as the exposure of identifiers to parties not authorized by the original consent framework, which includes: persistent exposure to uncontrolled infrastructure, the silent loss of confidentiality via passive network interception (e.g., during TLS issuance lags), and the fundamental loss of lawful processing when data streams are captured by unvetted third parties without renewed legal basis [37].

6.3 Mitigation Strategies and Best Practices

The root cause of the risks we expose is a lifecycle mismatch: mobile apps and embedded SDKs behave as long-lived binaries, while the DNS and TLS infrastructure they depend on is short-lived and frequently reconfigured [14, 23, 70, 88]. Effective mitigation requires coordinated changes by all affected parties.

Developers and SDK Maintainers. For first-party apps and SDK providers, the most direct defense is TLS certificate pinning for endpoints that receive identifiers or permission-protected sensor data. Correctly deployed, it prevents silent TLS connections to endpoints with altered trust relationships, even if an attacker provisions a new certificate during a downtime window [16, 73]. In contrast, our measurement setup hooks TLS at the OS level to inspect plaintext, which intentionally bypasses application-layer pinning; in a real deployment, pinning would cause many of the hijacking scenarios we describe to fail. However, because only a small minority of Android apps correctly deploy pinning [73], the ecosystem remains exposed. Beyond pinning, developers can reduce risk by adopting explicit fail-closed behaviors (such as disabling telemetry and surfacing warnings) when DNS or TLS handshakes fail, rather than silently retrying over downgraded channels. Finally, while upgrading third-party SDKs to fix endpoints is non-trivial, often requiring SDK level bumps and re-testing [14], our results suggest that dependency hygiene should be treated as a routine maintenance task to eliminate late-renewed or abandoned infrastructure.

OS Vendors and Networking Stacks. Mobile operating systems and their networking stacks can further reduce exposure by enforcing HTTPS by default for app traffic and adopting HSTS-like policies at the platform level. Once a domain has been contacted over authenticated HTTPS, subsequent connections from sandboxed apps should refuse to downgrade to plaintext or accept invalid certificates, even if the endpoint enters a TLS issuance gap following a renewal event [9, 16]. Such policies do not eliminate the risk of domain re-registration, but they narrow the attack surface by ensuring that identifier-bearing traffic is never transmitted over unauthenticated channels during the lifecycle gaps we quantify in § 5.2 and § 5.3. Additionally, some Android/iOS versions remove permissions or disable apps that have not been used in a while by the user, making it a partial defense that can reduce residual data flows from dormant installation.

App Stores and Marketplaces. App stores are in a unique position to detect infrastructure decay and guide developers towards safer practices at scale. Major app stores such as Google Play, could introduce upload and update-time checks that resolve an app's embedded domains, query WHOIS and DNS records, and flag cases where dependencies have already expired, exhibit prolonged downtime, or rely on dangling CNAMEs to deallocated cloud resources [70, 88]. Rather than immediately blocking submissions, stores could surface authenticated warnings in developer dashboards when an app depends on domains with repeated late renewals or recent expiration events, and require developers to acknowledge and document mitigation steps.

DNS Providers, Registrars, and Regulators. Finally, our registrar analysis from §5.2 highlights that inconsistent grace and redemption periods, when combined with aggressive auctioning policies, can substantially prolong the windows during which expired domains remain recoverable or attractive to drop-catchers (Table 4) [2, 33]. Registrars and regulators could reduce this risk by standardizing minimum grace-period behavior, shortening or hardening auction phases for domains that host sensitive APIs, and introducing additional verification or temporary holds before re-registering domains known to be referenced by widely deployed mobile applications. One practical step would be for registrars and large app-store operators to share dependency intelligence: domains observed in app dependency graphs could be added to a "protected" list whose re-registration triggers automatic notification to the original registrant. Such policies would not completely eliminate lifecycle mismanagement, but they would materially raise the bar for adversaries seeking to exploit expired mobile endpoints as a source of residual PII.

7 Conclusions

In this paper, we presented a privacy risk analysis of domain endpoints in the Android ecosystem by combining dynamic analysis of 11,131 apps with longitudinal WHOIS, DNS, and TLS measurements of 3,420 domains around their expiration events. Our results show that 25.3% of these domains are renewed after expiration, with 78.7% of the late-renewed domains belonging to third-party services and SDKs embedded in apps whose install brackets total over 78 billion cumulative Google Play downloads. We identified 218 late-renewed domains with dangling CNAME records susceptible to subdomain hijacking, and queried the Certificate Transparency logs to confirm that 34 of these domains received 91 valid TLS certificates during their expired periods. Our TLS lifecycle analysis further revealed that 17.0% of domains experience post-renewal issuance gaps and 70.5% have certificates that expire before domain renewal, creating windows where confidentiality guarantees can be weakened or even lost. By correlating these infrastructure and lifecycle failures with dynamically observed identifier flows, we showed that vulnerable advertising and tracking endpoints are embedded in over 1,100 apps spanning 5.9 billion cumulative downloads, while vulnerable utility and backend endpoints affect over 3,800 apps spanning 50.4 billion cumulative downloads, collectively transmitting identifiers to endpoints whose ownership may no longer be controlled by the original operator. Our findings demonstrate that domain lifecycle mismanagement is not merely an operational concern but a concrete and measurable privacy risk in the mobile supply chain. We argue that the mobile ecosystem requires coordinated governance mechanisms to ensure that users' personal data does not outlive the infrastructure meant to protect it.

Acknowledgments

We thank the reviewers and the shepherd for their valuable feedback and suggestions for improving the paper. We used generative AI-based tools (specifically Perplexity, powered by Gemini 3.1 Pro and Deep Research) to revise the text, improve structural flow, and correct typographical and grammatical errors. This research was supported by the Spanish MCIN/AEI/10.13039/50-1100011033/ through grants PID2022- 140126OB-I00 (CYCAD) and RED2024-154240-T (EMACS). Narseo Vallina-Rodriguez has been appointed as 2019 Ramon y Cajal fellow (RYC2020-030316-I) funded by MICIU/AEI/10.13039/501100011033 and the ESF Investing in your future. The opinions, findings, and conclusions or recommendations expressed are those of the authors and do not necessarily reflect those of any of the funding agencies.

References

[1] 1API GmbH. 2023. Registrant Agreement. https://www.1api.net/legal/registrant-agreement

[2] Gautam Akiwate, Mattijs Jonker, Raffaele Sommese, Ian Foster, Geoffrey M. Voelker, Stefan Savage, K. C. Claffy, and Alessio Santanna. 2021. Risks of Registrar Name Management. In *Proceedings of the ACM Internet Measurement Conference (IMC)*. ACM, 406–420.

[3] Omar Alrawi, Chaoshun Zuo, Ruian Duan, Ranjita Pai Kasturi, Zhiqiang Lin, and Brendan Saltaformaggio. 2019. The Betrayal at Cloud City: An Empirical Analysis of Cloud-Based Mobile Backends. In *28th USENIX Security Symposium (USENIX Security 19)*. USENIX Association, 551–566.

[4] Alireza Ardalani, Joseph Antonucci, and Iulian Neamtiu. 2022. A Study of Personal Information Leaks in Mobile Medical, Health, and Fitness Apps. *IADIS International Journal on WWW/Internet* 22, 2 (2022), 93–110. https://www.iadisportal.org/ijwi/papers/2024220207.pdf

[5] Ascio. 2023. What Do Domain Status Codes Mean? https://www.ascio.com/blog/what-do-domain-status-codes-mean/

[6] Channel News Asia. 2024. E-commerce platform Qoo10 to be wound up; liquidators appointed. https://www.channelnewsasia.com/singapore/ecommerce-platform-qoo10-winding-up-debt-insolvency-liquidators-appointed-4741261.

[7] Timothy Barron, Najmeh Miramirkhani, and Nick Nikiforakis. 2019. Now You See It, Now You Don't: A Large-Scale Analysis of Early Domain Deletions. In *22nd International Symposium on Research in Attacks, Intrusions and Defenses (RAID)*. USENIX Association, 383–397.

[8] Reuben Binns, Ulrik Lyngs, Max Van Kleek, Jun Zhao, Timothy Libert, and Nigel Shadbolt. 2018. Third Party Tracking in the Mobile Ecosystem. In *Proceedings of the 10th ACM Conference on Web Science (WebSci)*. ACM, 23–31.

[9] Kevin Borgolte, Tobias Fiebig, Shuang Hao, Christopher Kruegel, and Giovanni Vigna. 2018. Cloud Strife: Mitigating the Security Risks of Domain-Validated Certificates. In *Network and Distributed System Security Symposium (NDSS)*. Internet Society.

[10] CarWale. 2020. Jeep India launches 'SELECTEDforYOU', its pre-owned vehicle business. https://www.carwale.com/news/jeep-india-launches-selectedforyou-its-preowned-vehicle-division/

[11] Federal Trade Commission. 2022. FTC Sues Kochava for Selling Data that Tracks People at Reproductive Health Clinics, Places of Worship, and Other Sensitive Locations. https://www.ftc.gov/news-events/news/press-releases/2022/08/ftc-sues-kochava-selling-data-tracks-people-reproductive-health-clinics-places-worship-other

[12] Kodular Community. 2022. Start.io SDK version - Your app includes non compliant SDK version. https://community.kodular.io/t/start-io-sdk-version-your-app-includes-non-compliant-sdk-version/198451

[13] Freepik Company. 2026. The Start of Flaticon. https://www.flaticon.com/blog/the-start-of-flaticon/

[14] Erik Derr, Sven Bugiel, Sascha Fahl, Yasemin Acar, and Michael Backes. 2017. An Empirical Study of Third-Party Library Updatability on Android. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS)*. Association for Computing Machinery, Dallas, TX, USA. https://doi.org/10.1145/3133956.3134059

[15] Digital Turbine. 2021. Digital Turbine Completes Acquisition of Fyber. https://ir.digitalturbine.com/news-events/press-releases/detail/610/digital-turbine-announces-completion-of-acquisition-of

[16] Zakir Durumeric, James Kasten, Michael Bailey, and J Alex Halderman. 2013. Analysis of the HTTPS certificate ecosystem. In *Proceedings of the 2013 Internet Measurement Conference (IMC)*. 291–304.

[17] William Enck, Peter Gilbert, Seungyeop Han, Vasant Tendulkar, Byung-Gon Chun, Landon P. Cox, Jaeyeon Jung, Patrick McDaniel, and Anmol N. Sheth. 2014. TaintDroid: An Information-Flow Tracking System for Realtime Privacy Monitoring on Smartphones. *ACM Transactions on Computer Systems (TOCS)* 32, 2 (2014), 1–29.

[18] European Union. 2016. *Article 6 GDPR – Lawfulness of Processing*. European Union. https://gdpr-info.eu/art-6-gdpr/

[19] Sascha Fahl, Marian Harbach, Thomas Muders, Lars Baumgärtner, Bernd Freisleben, and Matthew Smith. 2012. Why Eve and Mallory love Android: An analysis of Android SSL (in)security. In *Proceedings of the 2012 ACM Conference on Computer and Communications Security (CCS)*. ACM, 50–61. https://doi.org/10.1145/2382196.2382205

[20] Alvaro Feal, Paolo Calciati, Narseo Vallina-Rodriguez, Carmela Troncoso, and Alessandra Gorla. 2020. Angel or Devil? A Privacy Study of Mobile Parental Control Apps. *Proceedings on Privacy Enhancing Technologies (PoPETs)* 2020, 2 (2020), 314–335.

[21] Álvaro Feal, Julien Gamba, Juan Tapiador, Primal Wijesekera, Joel Reardon, Serge Egelman, and Narseo Vallina-Rodriguez. 2021. Don't Accept Candy from Strangers: An Analysis of Third-Party Mobile SDKs. In *Data Protection and Privacy: Data Protection and Artificial Intelligence*. Bloomsbury Publishing.

[22] Mark Felegyhazi, Christian Kreibich, and Vern Paxson. 2010. On the Potential of Proactive Domain Blacklisting. In *Proceedings of the 3rd USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET '10)*. USENIX Association, Berkeley, CA, USA, 1–7.

[23] Kevin Frieß, Haya Schulmann, Michael Waidko, and Michael Waidner. 2024. Cloudy with a Chance of Cyberattacks: Dangling Resources Abuse on Cloud Platforms. In *21st USENIX Symposium on Networked Systems Design and Implementation (NSDI 24)*. USENIX Association, 1265–1280.

[24] Julien Gamba, Mohammed Rashed, Abbas Macak, Juan Aragon, Carmela Carmona, Narseo Vallina-Rodriguez, and Juan Tapiador. 2020. An Analysis of Pre-installed Android Software. In *41st IEEE Symposium on Security and Privacy (S&P)*. IEEE, 1039–1052.

[25] Aniketh Girish, Juan Tapiador, and Narseo Vallina-Rodriguez. 2025. Your Signal, Their Data: An Empirical Privacy Analysis of Wireless-Scanning SDKs in Android. *arXiv preprint arXiv:2503.15238* (2025).

[26] GO2 IT Group. 2025. *How Expired Domains Can Lead to Hijacking and Data Breaches*. GO2 IT Group. https://go2itgroup.com/expired-domains-and-data-breaches/

[27] GoDaddy. 2023. What Happens When My Domain Expires? https://www.godaddy.com/en-in/help/what-happens-when-my-domain-expires-609

[28] GoDaddy Inc. 2020. GoDaddy Acquires Neustar's Registry Business. https://aboutus.godaddy.net/newsroom/news-releases/press-release-details/2020/GoDaddy-Acquires-Neustars-Registry-Business/

[29] Google. 2024. UI/Application Exerciser Monkey. https://developer.android.com/studio/test/other-testing-tools/monkey.

[30] Yue He, Xiaofeng Pan, Rui Wang, Yue Cao, Xiao Liu, Zhou Li, and Xuejun Tang. 2019. Dynamic Privacy Leakage Analysis of Android Third-Party Libraries. *IEEE Access* 7 (2019), 92587–92598.

[31] HostGator. 2024. Brands That Forgot to Renew Their Domain Name. https://www.hostgator.com/blog/brands-forgot-renew-domain-name/

[32] Hostinger. 2023. Expired Registration Recovery Policy. https://www.hostinger.in/legal/expired-registration-recovery-policy

[33] ICANN. 2013. Expired Registration Recovery Policy (ERRP). https://www.icann.org/resources/pages/errp-2013-02-28-en

[34] ICANN Security and Stability Advisory Committee (SSAC). 2010. *SAC044: A Registrant's Guide to Protecting Domain Name Registration Accounts*. Technical Report. Internet Corporation for Assigned Names and Numbers. https://www.icann.org/en/system/files/files/sac-044-en.pdf

[35] ICANN Security and Stability Advisory Committee (SSAC). 2024. *SAC125: SSAC Report on Registrar Nameserver Management*. Technical Report. Internet Corporation for Assigned Names and Numbers. https://itp.cdn.icann.org/en/files/security-and-stability-advisory-committee-ssac-reports/sac-125-09-05-2024-en.pdf

[36] IN Registry. 2025. .IN Registry Policies. https://www.registry.in/policies

[37] Information Commissioner's Office. 2024. *A Guide to Lawful Basis*. Information Commissioner's Office. https://ico.org.uk/for-organisations/uk-gdpr-guidance-and-resources/lawful-basis/a-guide-to-lawful-basis/

[38] InterNetX. 2023. Registration Agreement. https://www.internetx.com/fileadmin/files/internetx/pdf/tld-terms/InterNetX_registration_agreement

[39] IONOS. 2023. Reactivating an Expired Domain. https://www.ionos.com/help/domains/terms-and-cancellations/reactivating-an-expired-domain/

[40] IONOS. 2024. Reactivating an Expired Domain. https://www.ionos.com/help/domains/terms-and-cancellations/reactivating-an-expired-domain/

[41] Davino Mauro Junior, Luis Melo, Harvey Lu, Marcelo d'Amorim, and Atul Prakash. 2019. Beware of the App! On the Vulnerability Surface of Smart Devices through their Companion Apps. arXiv:1901.10062 [cs.CR] https://arxiv.org/abs/1901.10062

[42] Key-Systems GmbH. 2023. Registration Agreement. https://www.key-systems.net/en/legal/registration-agreement/

[43] Konrad Kollnig, Reuben Binns, Max Van Kleek, Ulrik Lyngs, Jun Zhao, Claudine Tinsman, and Nigel Shadbolt. 2021. Before and after GDPR: tracking in mobile apps. *Internet Policy Review* 10, 4 (2021).

[44] Konrads Smelkovs. 2023. danglingcname: Detecting and Reporting Dangling CNAMEs. https://github.com/truekonrads/danglingcname

[45] Tripti Lahiri. 2018. The Indian-owned tech firm in the eye of the Narendra Modi app storm. https://qz.com/india/1237311/narendra-modi-app-and-clevertap-the-indian-owned-tech-firm-in-the-eye-of-the-storm

[46] Pierre Laperdrix, Alexandre Biryukov, Marc Dacier, Jean-Luc Falliere, Axel Legay, Jacques Klein, Yves Le Traon, Martin Monperrus, Li Li, and Guillaume Suarez-Tangil. 2016. AndroZoo: Collecting Millions of Android Apps for the Research Community. https://androzoo.uni.lu/

[47] Frederic Lardinois. 2017. Zeta Global acquires commenting service Disqus. TechCrunch. https://techcrunch.com/2017/12/05/zeta-global-acquires-commenting-service-disqus/

[48] Tobias Lauinger, Ahmet S. Buyukkayhan, Abdelberi Chaabane, William Robertson, and Engin Kirda. 2018. From Deletion to Re-Registration in Zero Seconds: Domain Registrar Behaviour During the Drop. In *Proceedings of the Internet Measurement Conference 2018 (IMC '18)*. Association for Computing Machinery, New York, NY, USA, 322–328. https://doi.org/10.1145/3278532.3278560

[49] Tobias Lauinger, Abdelberi Chaabane, Ahmet Salih Buyukkayhan, Kaan Onarlioglu, and William Robertson. 2017. Game of Registrars: An Empirical Analysis of Post-Expiration Domain Name Takeovers. In *Proceedings of the 26th USENIX Security Symposium (USENIX Security '17)*. USENIX Association, Berkeley, CA, USA, 865–880.

[50] Tobias Lauinger, Kaan Onarlioglu, Abdelberi Chaabane, William K. Robertson, and Engin Kirda. 2016. WHOIS Lost in Translation: (Mis)Understanding Domain Name Expiration and Re-Registration. In *Proceedings of the 2016 Internet Measurement Conference*. ACM, New York, NY, USA, 1–14.

[51] Victor Le Pochat, Tom Van Goethem, Samaneh Tajalizadehkhoob, Maciej Korczyński, and Wouter Joosen. 2019. Tranco: A Research-Oriented Top Sites Ranking Hardened Against Manipulation. In *Proceedings of the 26th Annual Network and Distributed System Security Symposium (NDSS)*. Internet Society.

[52] Chaz Lever, Robert Walls, Yacin Nadji, David Dagon, Patrick McDaniel, and Manos Antonakakis. 2016. Domain-Z: 28 Registrations Later Measuring the Exploitation of Residual Trust in Domains. In *Proceedings of the 2016 IEEE Symposium on Security and Privacy (SP '16)*. IEEE Computer Society, Los Alamitos, CA, USA, 691–706. https://doi.org/10.1109/SP.2016.47

[53] Daiping Liu, Shuai Hao, and Haining Wang. 2016. All Your DNS Records Point to Us: Understanding the Security Threats of Dangling DNS Records. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 1414–1425.

[54] He Liu, Kirill Levchenko, Mark Felegyhazi, Christian Kreibich, Gregor Maier, and Geoffrey M. Voelker. 2011. On the Effects of Registrar-Level Intervention. In *Proceedings of the 4th USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET '11)*. USENIX Association, Berkeley, CA, USA, 1–7.

[55] Allan Lyons, Primal Wijesekera, and Joel Reardon. 2023. Log: It's Big, It's Heavy, It's Filled with Personal Data! Measuring the Logging of Sensitive Information in the Android Ecosystem. In *32nd USENIX Security Symposium (USENIX Security 23)*. USENIX Association.

[56] Zane Ma, Aaron Faulkenberry, Thomas Papastergiou, Zakir Durumeric, Michael D. Bailey, Angelos D. Keromytis, Fabian Monrose, and Manos Antonakakis. 2023. Stale TLS Certificates: Investigating Precarious Third-Party Access to Valid TLS Keys. In *Proceedings of the 2023 ACM on Internet Measurement Conference* (Montreal QC, Canada) *(IMC '23)*. Association for Computing Machinery, New York, NY, USA, 222–235. https://doi.org/10.1145/3618257.3624802

[57] Samin Yaseer Mahmud, Akhil Acharya, Benjamin Andow, William Enck, and Bradley Reaves. 2020. Cardpliance: PCI DSS Compliance of Android Applications. In *Proceedings of the 29th USENIX Security Symposium*. USENIX Association. https://www.usenix.org/system/files/sec20fall_mahmud_prepub.pdf

[58] René Mayrhofer, Jeffrey Vander Stoep, Chad Brubaker, and Dianne Hackborn. 2021. The Android Platform Security Model. *ACM Transactions on Privacy and Security (TOPS)* 24, 3 (2021), 1–35.

[59] m.Doc GmbH. 2025. About Us - m.Doc Smart Health Platform. https://mdoc.one/en/about-us/. Part of CompuGroup Medical.

[60] Mark Huasong Meng, Chuan Yan, Qing Zhang, Zeyu Wang, Kailong Wang, Sin Gee Teo, Guangdong Bai, and Jin Song Dong. 2024. Assessing Privacy Compliance of Android Third-Party SDKs. *arXiv preprint arXiv:2409.10411* (2024).

[61] Najmeh Miramirkhani, Timothy Barron, Michael Ferdman, and Nick Nikiforakis. 2018. Panning for gold.com: Understanding the Dynamics of Domain Dropcatching. In *Proceedings of the 2018 World Wide Web Conference*. ACM, New York, NY, USA, 257–266. https://doi.org/10.1145/3178876.3186092

[62] Giovane C. M. Moura, Thomas Daniels, Maarten Bosteels, Sebastian Castro, Moritz Müller, Thymen Wabeke, Thijs van den Hout, MacIej Korczyński, and George Smaragdakis. 2024. *Characterizing and Mitigating Phishing Attacks at ccTLD Scale (Extended)*. Technical Report EWI-TR-2024-1. Delft University of Technology, Delft, The Netherlands.

[63] Patrick Mutchler, Adam Doupé, John Mitchell, Christopher Kruegel, and Giovanni Vigna. 2015. A Large-Scale Study of Mobile Web App Security. In *Proceedings of the Mobile Security Technologies Workshop (MoST)*.

[64] Muhammad Muzammil, Zhengyu Wu, Aruna Balasubramanian, and Nick Nikiforakis. 2024. Panning for gold.eth: Understanding and Analyzing ENS Domain Dropcatching. In *Proceedings of the 2024 ACM on Internet Measurement Conference (IMC '24)*. Association for Computing Machinery, New York, NY, USA, 731–738. https://doi.org/10.1145/3646547.3689009

[65] NameCheap. 2023. TLDs Grace Periods. https://www.namecheap.com/support/knowledgebase/article.aspx/9916/2207/tlds-grace-periods/

[66] Helen Nissenbaum. 2004. Privacy as Contextual Integrity. *Washington Law Review* 79 (2004), 119–158.

[67] North Carolina Department of Health and Human Services. 2024. About NCDHHS. https://www.ncdhhs.gov/about

[68] OVH Cloud. 2023. Domain Name Registration Gone Wrong. https://blog.ovhcloud.com/domain-name-registration-gone-wrong/

[69] PageSuite. 2024. PageSuite: Digital Publishing Platform for eEditions & News Apps. https://www.pagesuite.com/

[70] Elkana Pariwono, Daiki Chiba, Mitsuaki Akiyama, and Tatsuya Mori. 2018. Don't Throw Me Away: Threats Caused by the Abandoned Internet Resources Used by Android Apps. In *Proceedings of the 2018 on Asia Conference on Computer and Communications Security (ASIACCS '18)*. Association for Computing Machinery, New York, NY, USA, 147–158. https://doi.org/10.1145/3196494.3196554

[71] Ediciones El País. 2026. About EL PAÍS. https://english.elpais.com/usa/2021-06-09/about-the-el-pais-english-edition.html

[72] Pop Lens. 2025. *PyPI Unverifies 1,800 Emails with Expired Domains to Prevent Account Takeovers*. Pop Lens. https://www.youtube.com/watch?v=vSiVM0TvMcM

[73] Amogh Pradeep, Muhammad Talha Paracha, Protick Bhowmick, Ali Davanian, Abbas Razaghpanah, Taejoong Chung, Martina Lindorfer, Narseo Vallina-Rodriguez, Dave Levin, and David Choffnes. 2022. A Comparative Analysis of Certificate Pinning in Android & iOS. In *Proceedings of the 22nd ACM Internet Measurement Conference (IMC)*. ACM, 1–14.

[74] Public Suffix List contributors. 2026. The Public Suffix List. https://publicsuffix.org/list/. Accessed: February 2026.

[75] Quora User. 2024. I accidentally forget to renew my domain and now someone else has registered for the domain. What can I do? https://www.quora.com/I-accidentally-forget-to-renew-my-domain-and-now-someone-else-has-registered-for-the-domain-What-can-I-do

[76] Abbas Razaghpanah, Rishab Nithyanand, Narseo Vallina-Rodríguez, Srikanth Sundaresan, Mark Allman, Christian Kreibich, and Phillipa Gill. 2018. Apps, Trackers, Privacy, and Regulators: A Global Study of the Mobile Tracking Ecosystem. In *Proceedings of the 25th Network and Distributed System Security Symposium (NDSS)*. The Internet Society, San Diego, CA, USA, 1–15.

[77] Joel Reardon, Álvaro Feal, Primal Wijesekera, Amit Elazari Bar On, Narseo Vallina-Rodriguez, and Serge Egelman. 2019. 50 Ways to Leak Your Data: An Exploration of Apps' Circumvention of the Android Permissions System. In *28th USENIX Security Symposium (USENIX Security 19)*. USENIX Association, 603–620.

[78] Register.IT. 2023. Domain Registration Agreement. https://www.register.it/company/legal/ods-registrazione-nomi-dominio.html

[79] Jingjing Ren, Martina Lindorfer, Daniel J. Dubois, Ashwin Rao, Narseo Vallina-Rodríguez, and Gábor Gün Karsai. 2018. A Longitudinal Study of PII Leaks Across Android App Versions. In *Proceedings of the 25th Network and Distributed System Security Symposium (NDSS)*. The Internet Society, San Diego, CA, 1–15.

[80] Kimberly Ruth, Deepak Kumar, Brandon Wang, Luke Valenta, and Zakir Durumeric. 2022. Toppling top lists: evaluating the accuracy of popular website lists. In *Proceedings of the 22nd ACM Internet Measurement Conference* (Nice, France) *(IMC '22)*. Association for Computing Machinery, New York, NY, USA, 374–387. https://doi.org/10.1145/3517745.3561444

[81] SafeNames. 2023. Terms and Conditions. https://www.safenames.net/resources/terms-conditions

[82] Allison Schiff. 2021. *Location Intel Provider Cuebiq Is Shutting Down Its SDK In The Name Of Privacy*. https://www.adexchanger.com/mobile/location-intel-provider-cuebiq-is-shutting-down-its-sdk-in-the-name-of-privacy/

[83] Johann Schlamp, Georg Carle, and Ernst Biersack. 2013. A Forensic Case Study on AS Hijacking: The Attacker's Perspective. *ACM SIGCOMM Computer Communication Review* 43, 2 (2013), 5–12. https://doi.org/10.1145/2479957.2479959

[84] Johann Schlamp, Josef Gustafsson, Matthias Wählisch, Thomas C. Schmidt, and Georg Carle. 2015. The Abandoned Side of the Internet: Hijacking Internet Resources When Domain Names Expire. In *Proceedings of the 7th Workshop on Traffic Monitoring and Analysis (TMA '15)*. Springer-Verlag, Berlin, Heidelberg, 188–201. https://doi.org/10.1007/978-3-319-17172-2_13

[85] Sectigo. 2025. crt.sh – Certificate Search. https://crt.sh/ Accessed: 2025-05-04.

[86] Smarten Spaces. 2024. About Us: Powering the Future of Workspaces. https://smartenspaces.com/about/index.html

[87] Johnny So, Najmeh Miramirkhani, Michael Ferdman, and Nick Nikiforakis. 2022. Domains Do Change Their Spots: Quantifying Potential Abuse of Residual Trust. In *Proceedings of the 2022 IEEE Symposium on Security and Privacy (SP '22)*. IEEE Computer Society, Los Alamitos, CA, USA, 119–133.
[88] Johnny So, Iskander Sánchez-Rola, and Nick Nikiforakis. 2025. Lost in the Mists of Time: Expirations in DNS Footprints of Mobile Apps. In *Proceedings of the 34th USENIX Security Symposium*. USENIX Association, Seattle, WA, 3297–3314.
[89] Marco Squarcina, Mauro Tempesta, Lorenzo Zago, Stefano Pelosi, and Alessandro Bugliesi. 2021. Can I Take Your Subdomain? Exploring Same-Site Attacks in the Modern Web. In *30th USENIX Security Symposium (USENIX Security 21)*. USENIX Association, 2915–2932.
[90] Stack Exchange User. 2024. Registrar forgot to renew my domain name after I paid the invoice. https://webmasters.stackexchange.com/questions/120109/registrar-forgot-to-renew-my-domain-name-after-i-paid-the-invoice
[91] Taipei Rapid Transit Corporation. 2024. Metro Taipei: Company Profile. https://english.metro.taipei/
[92] Faiza Tazi, Suleiman Saka, Griffin Opp, Shradha Neupane, Sanchari Das, Lorenzo De Carli, and Indrakshi Ray. 2023. Accessibility Evaluation of IoT Android Mobile Companion Apps. In *Extended Abstracts of the 2023 ACM Conference on Human Factors in Computing Systems*. ACM. https://doi.org/10.1145/3544549.3585890
[93] The Straits Times. 2025. Qoo10 Singapore creditors file $198M in claims, only $34,650 recovered. https://www.straitstimes.com/business/qoo10-under-fire-creditors-file-198m-in-claims-but-only-34650-recovered.
[94] Trustpilot. 2025. Rebag Reviews and Scam Alerts. https://www.trustpilot.com/review/rebag.com. Context for impersonation domains like reshbagonline.
[95] Ubilibet. 2023. General Terms and Conditions. https://www.ubilibet.com/wp-content/uploads/2022/07/EN_Ubilibet-Condiciones-generales-v01.2.pdf
[96] Inc. Urban Outfitters. 2026. About Us - URBN. https://www.urbn.com/our-brands/urban-outfitters/about-us
[97] U.S. Government. 2025. .gov Domain Requirements. https://get.gov/domains/requirements/
[98] Thomas Vissers, Wouter Joosen, and Nick Nikiforakis. 2015. Parking Sensors: Analyzing and Detecting Parked Domains. In *Proceedings of the 2015 Network and Distributed System Security Symposium (NDSS '15)*. Internet Society, Reston, VA, USA, 1–15. https://doi.org/10.14722/ndss.2015.23053
[99] Xueqiang Wang, Haining Wang, Kangjie Lu, and Suman Jana. 2019. Looking from the Mirror: Evaluating IoT Device Security through Mobile Companion Apps. In *Proceedings of the 28th USENIX Security Symposium*. USENIX Association. https://www.usenix.org/conference/usenixsecurity19/presentation/wang-xueqiang
[100] Weather Tracker TV. 2024. Weather Tracker TV DFW: About Us. https://www.weathertrackertv.com/aboutUs.php
[101] WildWestDomains. 2023. Registration Service Agreement. https://www.wildwestdomains.com/agreements/REG_SA
[102] X Corp. 2024. X Link Shortener (`t.co`) and How It Works. https://help.x.com/en/using-x/url-shortener
[103] Zhonghao Yu, Sam Macbeth, Konark Modi, and Josep M. Pujol. 2016. Tracking the Trackers. In *Proceedings of the 25th International Conference on World Wide Web (WWW)*. ACM, 121–132. https://doi.org/10.1145/2872427.2883028
[104] Mingming Zhang, Xinrui Li, Baojun Liu, Jianjun Lu, Yiming Zhang, Jianjun Chen, Haixin Duan, Shuang Hao, and Zaifeng Zhang. 2023. Detecting and Measuring Security Risks of Hosting-Based Dangling Domains. *Proceedings of the ACM on Measurement and Analysis of Computing Systems (POMACS)* 7, 1 (2023), 1–28.
[105] Chaoshun Zuo and Zhiqiang Lin. 2017. SmartGen: Exposing Server URLs of Mobile Apps with Selective Symbolic Execution. In *Proceedings of the 26th International Conference on World Wide Web (WWW)*. International World Wide Web Conferences Steering Committee, 867–876.

A Ethical Considerations and Responsible Disclosure

All dynamic analyses were conducted on instrumented test devices under our control, populated with synthetic profiles and dummy location data, where no real user traffic was intercepted. We did not re-register expired domains, claim dangling cloud resources, or exploit any of the endpoints; external interaction was limited to DNS, WHOIS, TLS, and CT queries needed for measurement.

Targeted Developer Outreach. On February 16, 2026, we initiated a manual disclosure process targeting the 25 highest-risk applications in our dataset, associated with 12 domains. This cohort was chosen by jointly considering (i) longest out-of-grace downtime, (ii) transmission of highly sensitive identifiers, and (iii) DNS/TLS issues such as missing certificates or dangling CNAMEs. Among these 25 cases, 21 rely on third-party services or SDK backends and 4 on first-party domains. Using the official Google Play developer contact emails, we notified each developer; as of May 2026, none has responded. The reason for not contacting all the affected developers is that, across all late-renewed domains, the average downtime was 17.59 hours (range 0.0003–288.08 hours); for domains operated by shared SDK or third-party infrastructure, the average was 16.89 hours (range 0.0003–217.95 hours). These windows fall within typical 30-day registrar grace/redemption periods and therefore mostly indicate poor operational practices and short-lived disruptions rather than long-lived, exploitable expirations. Mass-emailing thousands of developers using scraped contacts about such lapses would likely cause high bounce/spam rates.

Platform-Level Escalation. To enable scalable remediation, on May 5, 2026 we submitted a detailed report via the Google Bug Hunters (Google VRP) portal, classifying the issue as sensitive data exposure and referencing the Google Play Console as the affected service. The report included our full dataset of 861 late-renewed domains with downtime, TLS validity, dangling-CNAME status, and corresponding Google Play package names. In parallel, we sent the same report by email to the Android Security Team and offered to share our manuscript confidentially. Our goal is for Google to feed these domains into its App Security Improvement (ASI) and related scanning pipelines so that authenticated warnings can be surfaced directly in Play Console dashboards, reaching affected developers through a coordinated, platform-level disclosure channel. However, on May 7, 2026 the Google VRP rejected the report as "Out of Scope," instructing us instead to use the consumer-facing "Report App" feature to flag individual applications for general policy violations.

B Open Science

This appendix describes the artifacts generated with this paper and how they can be accessed.

Artifact Enumeration. To support full reproducibility of our methodology and findings, we provide the measurement pipeline, analysis scripts, input datasets, and intermediate results. The repository is structured as follows:

- **Input Datasets (`data/`).** The complete set of (APK package name, domain) pairs extracted from our dynamic analysis campaign. Each subdirectory corresponds to a device category in the instrumentation setup.
- **Active DNS Measurement Pipeline (`scripts/4.3_active_dns_measurements/`).** The seven-stage sequential pipeline implementing the longitudinal measurement campaign described in §4.3: WHOIS and DNS resolution (`01_process_domains.py`), expiring domain filtering (`02`), hourly WHOIS polling with resumable state persistence (`03_monitor_whois.py`), downtime calculation (`04`), metadata enrichment with Play Store install counts (`05`), TLS certificate retrieval and X.509 parsing (`06`), and dangling CNAME detection via the included Go binary (`07`) [44].
- **Privacy Analysis (`scripts/4.4_privacy_risk_analysis/`).** The analysis scripts implementing §4.4: first-party vs. third-party domain classification (`01_classify_domains.py`), registrar extraction (`02`), TLS lifecycle and ownership verification (`03a`, `03b`),

dangling CNAME risk scoring against Tranco rankings (`04a`), aggregate user exposure statistics (`04b`), and PII exposure cross-referencing (`05_pii_exposure_analysis/`).
- **PII Observation Data (`scripts/4.4_.../05_pii_exposure_analysis/`).** The Jupyter notebook and five PII observation CSVs from the AOSP instrumentation platform. The notebook cross-references detected PII identifiers with expired domains and generates the LaTeX risk summary tables used in §5.
- **Intermediate and Final Results (`results/`).** All per-month WHOIS monitoring outputs (August 2024–May 2025), enriched JSON files with TLS and dangling CNAME data, Play Store metadata, Certificate Transparency log queries, and the final consolidated dataset (`expired_domains_as_of_JUNE25.json`).
- **Visualization Scripts (`scripts/visualization/`).** The scripts used to generate the downtime CDF with ICANN lifecycle phase overlays (Figure 2), the Tranco rank vs. downtime scatter plot (Figure 3), and Certificate Transparency log analysis.
- **Auxiliary Tools.** The Play Store metadata scraper (`scripts/auxiliary/get_playstore_metadata.py`), the pre-compiled dangling CNAME detector (`danglingcname/`) [44], the Tranco top-1M list (`tranco/`), and documentation (including `README.md` and `requirements.txt`).

Artifact Repository In strict adherence to the double-blind review policy, all artifacts have been uploaded to an anonymous, non-tracking repository. The Program Committee can access the complete artifact package at the following URL:

https://anonymous.4open.science/r/expired_domains_code-8EA5

Reproducing the Results To reproduce the quantitative evaluation without active network access, reviewers should:

(1) Install dependencies via `pip install -r requirements.txt`.
(2) Run the Stage 4.4 analysis scripts (`01_classify_domains.py` through `04b_exposure_analysis.py`) to regenerate all domain classification, TLS risk, CNAME, and exposure statistics.
(3) Open the PII exposure notebook (`05_pii_exposure_analysis.ipynb`) to reproduce the PII cross-reference tables.
(4) Execute the visualization scripts to regenerate all figures.

The pre-computed intermediate results from all ten monthly measurement cycles are included in `results/`. Note that the largest intermediate files (e.g., `domain_list_filtered_output_sorted.zip`, `non_iot_output_sorted.zip` and `from_ACR_dataset_output_sorted.zip`) are provided as ZIP archives to comply with repository limits and must be unzipped to reproduce the full Stage 4.3 pipeline. All tables and figures in §5 can be fully reproduced using the consolidated `expired_domains_as_of_JUNE25.json` file.

Justification for Redacted Artifacts The traffic interception component (§4.2) relies on a custom Android Open Source Project (AOSP) build with proprietary OS-level hooks for transparent TLS decryption and network-layer instrumentation. Because these modifications are tightly coupled to the institution's testing infrastructure and contain components that cannot be redistributed, the instrumented OS image is not included in the artifact. However, this component serves exclusively as a data-collection front-end: its sole output is the set of (APK, SLD) pairs and PII observation records, both of which are released in full.