The Cost of Convenience: Identifying, Analyzing, and Mitigating Predatory Loan Applications on Android

Olawale Amos Akanji
Boston University
Boston, MA, USA
olawalea@bu.edu

Manuel Egele
Boston University
Boston, MA, USA
megele@bu.edu

Gianluca Stringhini
Boston University
Boston, MA, USA
gian@bu.edu

Abstract

Digital lending applications, commonly referred to as *loan apps*, have become a primary channel for microcredit in emerging markets. However, many of these apps demand excessive permissions and misuse sensitive user data for coercive debt-recovery practices, including harassment, blackmail, and public shaming that affect both borrowers and their contacts.

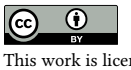
This paper presents the first cross-country measurement of loan app compliance against both national regulations and Google's Financial Services Policy. We analyze 434 apps drawn from official registries and app markets from Indonesia, Kenya, Nigeria, Pakistan, and the Philippines. To operationalize policy requirements at scale, we translate policy text into testable permission checks using LLM-assisted policy-to-permission mapping and combine this with static and dynamic analyses of loan apps' code and runtime behavior.

Our findings reveals pervasive non-compliance among *approved* apps: **141** violate national regulatory policy and **147** violate Google policy. Dynamic analysis further shows that several apps transmit sensitive data (contacts, SMS, location, media) *before* user signup or registration, undermining informed consent and enabling downstream harassment of borrowers and third parties. Following our disclosures, Google removed 93 flagged apps from Google Play, representing over 300M cumulative installs.

We advocate for adopting our methodology as a proactive compliance-monitoring tool and offer targeted recommendations for regulators, platforms, and developers to strengthen privacy protections. Overall, our results highlight the need for coordinated enforcement and robust technical safeguards to ensure that digital lending supports financial inclusion without compromising user privacy or safety.

CCS Concepts

• **Security and privacy** → *Mobile and wireless security*; **Privacy protections**; **Economics of security and privacy**.



ASIA CCS '26, Bangalore, India

ACM ISBN 979-8-4007-2356-8/2026/06
https://doi.org/10.1145/3779208.3785263

Keywords

Loan App, Digital Lending, Privacy Violation, Regulatory Compliance Analysis, Android Apps, Android Permissions

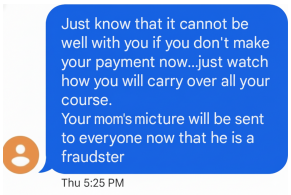
ACM Reference Format:
Olawale Amos Akanji, Manuel Egele, and Gianluca Stringhini. 2026. The Cost of Convenience: Identifying, Analyzing, and Mitigating Predatory Loan Applications on Android. In *ACM Asia Conference on Computer and Communications Security (ASIA CCS '26), June 01–05, 2026, Bangalore, India.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3779208.3785263

1 Introduction

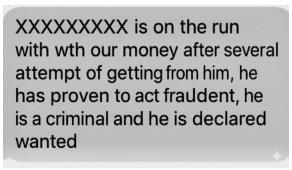
The rapid advancement of mobile technology has revolutionized the financial services industry, fueling the expansion of Fintech products, notably digital money lending applications (*loan apps*). These apps have seen explosive adoption across emerging markets like India, Indonesia, Kenya, Nigeria, Pakistan, and the Philippines. Historically, these economies relied on informal lending systems that lacked the scalability and speed needed to meet the growing credit needs of large populations, particularly those below the poverty line [30, 58, 75].

Loan apps have emerged as accessible, fast alternatives to formal and informal credit systems, promising convenience and ease-of-use to millions seeking quick credit [2, 52]. However, this convenience often comes at a steep cost: many apps obscure predatory financial terms (e.g., excessive interest rates, short repayment windows), trapping borrowers in debt cycles, and increasingly adopt unethical, digitized debt-collection tactics (harassment, blackmail) previously associated with informal lenders [1, 52].

Loan apps have been repeatedly linked to coercive debt collection, threats, harassment, and reputational harm that extend beyond borrowers to their families and contacts. Figures 1a and 1b illustrate examples of the threat messages and defamatory notices sent to borrowers and their contacts, highlighting the severe psychological and social impacts of these invasive practices. In response to widespread abuse, governments have introduced formal registration and privacy constraints for digital lenders, and Google has introduced a Financial Services Policy (FSP) that prohibits specific high-risk Android permissions for lending apps [38]. Yet abuse reports persist, leading to severe psychological distress and suicides of users in India, Pakistan, and the Philippines [19, 45, 59, 61, 73, 76]. Prior work on loan apps has primarily relied on qualitative methods, such as user interviews and case studies to document psychological distress and perceived data misuse [1, 52, 55, 66, 72],



(a) Example of a threat message received by users.



(b) Example of a message sent to the contacts of a defaulting user.

Figure 1: Coercive Debt Recovery Tactics

but does not quantitatively assess technical compliance with national regulations or platform policies. The persistence of abusive practices despite both national rules and Google's FSP therefore suggests significant gaps between policy intent and enforcement reality, motivating our systematic, cross-country technical compliance measurement.

Specifically, this study addresses the following questions:

(1) **To what extent do loan apps targeting *Indonesia, Kenya, Nigeria, Pakistan, and the Philippines* comply with national regulations and Google's Financial Services Policy, and what misalignments or exploitable gaps exist between these policy regimes?**
(2) **How do non-compliant loan apps access, collect, and transmit sensitive user data, and what evidence confirms these data exfiltrations that may facilitate malicious activities such as threats and harassment?**
(3) **What regulatory measures, technical controls, and industry-wide best practices could help mitigate non-compliance, strengthen oversight, and safeguard user privacy and security in digital lending services?**

We answer these research questions with a cross-country measurement of 435 Android loan apps drawn from regulator registries and app markets in Indonesia, Kenya, Nigeria, Pakistan, and the Philippines. Android is the focus of our analysis due to its dominant market share in our study regions (over 84% in Africa and 83% in Asia [70, 71]), making it the primary platform for digital lending and a critical locus for regulatory enforcement. Our methodology follows the subcategory-specific auditing paradigm (e.g., Cardpliance and VioDroid-Finder [23, 46]), but adapts it to loan apps and augments it with LLM-assisted policy-to-permission mapping and cross-country policy analysis.

Our methodology combines three components. (1) *LLM-assisted policy-to-permission mapping* translates regulatory text into Android permissions, enabling automated checks against policy requirements. (2) *Static analysis* inspects app manifests and bytecode to detect prohibited permissions and sensitive API calls, capturing each app's technical capabilities. (3) *Dynamic instrumentation and analysis* monitor runtime behavior to capture the timing and context of actual data access and transmission, distinguishing between legitimate loan or KYC workflows and launch-time collection that occurs before users can meaningfully consent. By combining these components into an end-to-end compliance pipeline, we expose systematic misalignment between national regulations and platform policies and link these gaps to concrete technical pathways for data misuse.

In summary, this paper makes the following contributions:

- We present the first data-driven measurement of loan app compliance with both national regulations (Indonesia, Kenya, Nigeria, Pakistan, and the Philippines) and Google's Financial Services Policy.
- We apply our methodology to 435 loan apps and uncover widespread non-compliance: 141 approved apps violate national regulatory policies, and 147 breach Google's Financial Services Policy, with combined downloads exceeding 300 million. Following our disclosure, Google removed 93 of the flagged apps. Notably, we identify 37 apps that access and transmit sensitive user data immediately upon launch, before any user interaction or registration, thereby undermining the principle of informed consent.
- We provide actionable recommendations to protect user privacy in the digital lending ecosystem, highlighting policy misalignments and asymmetric violations between national rules and Google's Financial Services Policy. We recommend that Google expand its prohibited-permissions list and that regulators deploy automated pre-approval and periodic audits. We have shared our findings with Google and the relevant national regulators, and we will publicly release our regulatory text-to-permission mappings, LLM prompts, and analysis scripts to support further research, enforcement, and policy reform.

2 Background

This section outlines the two-tiered regulatory landscape governing loan applications: Google's Financial Services Policy (FSP) and country-level frameworks. We detail key requirements across both regimes, including licensing mandates, specific data access restrictions, and the role of public registries in enhancing transparency. Finally, we describe the loan procurement process to identify critical stages where sensitive data collection introduces privacy risks.

2.1 Industry-Specific Regulation

Google, the primary distributor of Android apps through its Play Store, regulates digital lending through its Financial Services Policy (FSP) [38], which establishes explicit restrictions on data collection for personal loan apps distributed through its Play Store.

Google's FSP defines personal loan apps as those offering non-recurring consumer loans, excluding mortgages and revolving credit products [38]. It requires developers to categorize such apps under the "Finance" section and provide verifiable licensing and regulatory documentation when targeting

users in seven specified countries: India, Indonesia, Kenya, Nigeria, Pakistan, the Philippines, and Thailand.

Beyond requiring license documentation, Google's policy strictly prohibits loan apps from requesting a defined set of eight high-risk permissions: `READ_EXTERNAL_STORAGE`, `WRITE_EXTERNAL_STORAGE`, `READ_MEDIA_IMAGES`, `READ_MEDIA_VIDEOS`, `READ_CONTACTS`, `READ_PHONE_NUMBERS`, `ACCESS_FINE_LOCATION`, and `QUERY_ALL_PACKAGES` [38]. The intent of these prohibitions is to protect users from harassment and cyber-bullying.

2.2 Country-Specific Regulations

The seven countries referenced by Google's FSP enforce distinct national regulations designed to curb predatory lending and enhance user privacy. These frameworks generally mandate that all digital loan providers obtain proper licensing, undergo regular compliance audits, and adhere to strict limitations on data collection. Furthermore, many jurisdictions maintain public registries listing licensed and delisted providers to aid enforcement and improve user transparency.

These safeguards are especially important in contexts with weak or absent central credit-reporting systems, where lenders may otherwise rely on invasive data practices to assess creditworthiness. We summarize the key regulatory provisions and prohibited data-access practices in Table 1.

2.3 Loan Process and Privacy Risks

Loan apps typically follow a standardized process designed to deliver fast credit. This process involves installation, registration, permission granting, loan application, credit evaluation, approval, and repayment [52]. While this convenience is appealing, it comes at a cost as it introduces significant privacy risks at multiple stages of the loan process.

Users commonly install these apps via the Play Store, but many apps are also distributed through third-party APK sites, lender websites, or social media channels. During the initial stages, apps request access to sensitive data through two primary approaches: some demand permissions to contacts, call logs, and SMS immediately upon installation or launch, coercing users into granting access before any meaningful interaction, while others delay these requests until after registration.

The privacy risks become particularly pronounced during the data collection phase. Legitimate lending only requires basic identifiers (e.g., phone number, government ID), yet many apps request access to contacts, SMS logs, call records, and device storage, often describing this in privacy policies as "creditworthiness assessment" even where regulations explicitly prohibit such collection. In cases of default, this surplus data is frequently weaponized for harassment, blackmail, and public shaming of both borrowers and their contacts, turning initial privacy violations into severe harm and underscoring the need for stronger, enforceable safeguards across platforms and regulatory systems.

In cases of default, the extensive data collection enables aggressive debt recovery tactics. Some apps misuse the collected contact information, call logs, and personal data for harassment, blackmail, and public shaming of borrowers and their contacts. These behaviors transform initial privacy violations into severe harm, highlighting the urgent need for stronger, enforceable safeguards across platforms and regulatory systems.

3 Methodology

To measure loan app compliance with data-collection regulations, we develop LoanWatch, a reproducible three-phase compliance-audit methodology (Figure 2). LoanWatch comprises: (1) policy extraction to map regulatory text to Android permissions, (2) static analysis to identify prohibited permissions, APIs, and potential data flows, and (3) dynamic analysis to validate runtime access and transmission of sensitive data. We consider an app *violating* if it requests any permission prohibited by national regulations, Google's Financial Services Policy (FSP), or our harmonized LoanWatch set, which combines Google's prohibited permissions with additional high-risk permissions banned by at least one country.

For policy-to-permission mapping, we use three off-the-shelf LLMs (GPT-4o Mini, Grok3, and Claude Sonnet 4) to extract prohibited data types from regulatory text, distinguish unconditional from conditional bans, and map them to Android Open Source Project (AOSP). Our static analysis inspects each app's manifest to detect declared permissions, analyzes bytecode to identify sensitive API usage, and employs FlowDroid taint analysis to trace possible flows from sensitive sources to network sinks. Finally, semi-automated dynamic testing with Frida exercises flagged apps under guided, pre-registration interactions (launching the app and responding to permission dialogs) to confirm whether high-risk APIs actually access and transmit sensitive information.

In the rest of this section, we first describe our loan app collection process, then detail each phase of LoanWatch.

3.1 Loan App Collection

Google's Financial Services Policy explicitly references seven countries—India, Indonesia, Kenya, Nigeria, Pakistan, the Philippines, and Thailand—where developers must provide proof of compliance with local regulations before distributing loan apps through the Play Store. However, we limit our study to five countries, Indonesia, Kenya, Nigeria, Pakistan, and the Philippines as these jurisdictions maintain publicly accessible registries. We exclude India and Thailand due to the absence of publicly available registries that list compliant or non-compliant digital lending apps at the time of data collection (July 2025).

To build our dataset, we manually collect both approved and delisted apps. Approved apps are sourced directly from the Play Store to reflect their current availability and regulatory status. Delisted apps—removed for violating policy or regulatory conditions—are manually retrieved from the AndroZoo dataset and third-party APK repositories, such as APKCombo, APKMonk, and APKPure [9–11]. Including both categories

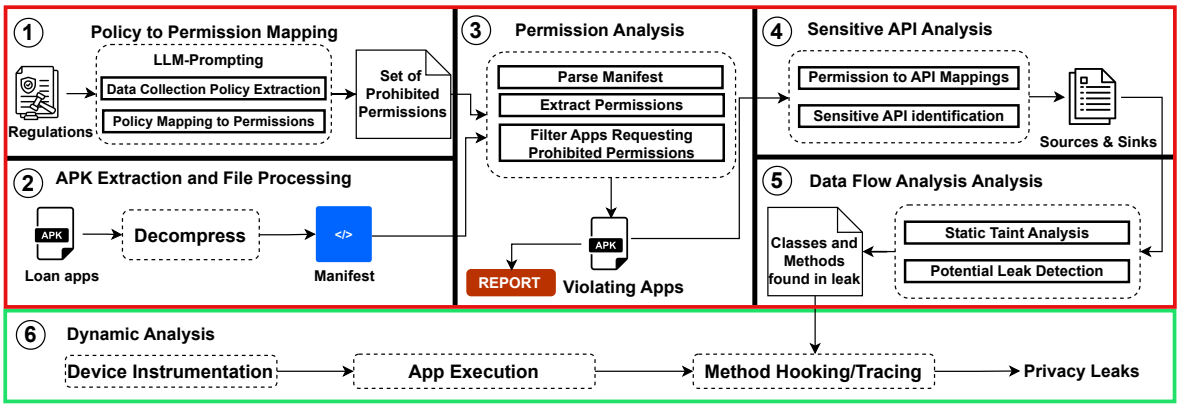


Figure 2: Overview of LoanWatch Methodology.

enables us to compare the operational behavior of compliant versus non-compliant apps, highlighting potential differences in permission usage and privacy practices.

The structure and detail of public registries varies across countries. Nigeria, Pakistan, and the Philippines maintain comprehensive registries that include both approved and delisted apps. These records typically list both company and app names, simplifying the identification of target apps. For these countries, we collect APKs by searching the Play Store, AndroZoo, and third-party repositories. In Nigeria's case, the Federal Competition and Consumer Protection Commission (FCCPC) further categorizes apps into full approval, conditional approval, watchlist, and delisted. Following consultation with the FCCPC, we collapse these into two categories for analytical clarity: "Approved" (full and conditional) and "Delisted" (watchlist and delisted). In contrast, Indonesia and Kenya only list approved apps; they do not maintain public records of delisted or banned apps. In Kenya, the registry lists only company names without associated apps, so we conduct manual searches to identify apps published by the listed entities. We exclude apps that are no longer operational or available on any app marketplace, as well as companies that operate exclusively via websites, since these do not request Android permissions and therefore fall outside the scope of our analysis. Our final dataset comprises 435 unique apps—342 approved and 93 delisted—distributed across the five selected countries.

3.2 Policy-to-Permission Mapping

Policy-to-permission mapping is essential for evaluating whether loan apps comply with regulatory data-access restrictions. Unlike Google's Financial Services Policy, which explicitly lists banned Android permissions, national regulations often describe prohibited practices in natural language (e.g., "shall not access contacts") without direct technical references, making automated checks non-trivial. Early privacy-policy research relied on manual expert annotation to assess compliance and policy evolution, which is accurate but cognitively costly and difficult to scale [4, 50]. NLP-based systems later scaled analysis through keyword detection and ML classifiers, but they require large labeled datasets and still struggle with legal nuance and domain transferability [12, 28, 80]. More recent work shows that LLMs can effectively support privacy tasks such as automated data-practice identification, interactive policy assessment, and GDPR compliance evaluation [36, 51, 74], motivating our LLM-based approach.

To establish ground truth, two Android security experts initially read each country's regulations and manually mapped prohibited data types (contacts, call logs, SMS, media, etc.) to Android permissions. While accurate, this process required several hours per country and does not scale as regulations evolve or new jurisdictions are added. To enable reproducible and scalable audits, we therefore adopt an LLM-assisted approach that uses off-the-shelf models to generalize from legal to technical language without training task-specific classifiers.

We use three LLMs—GPT-4o Mini, Grok3, and Claude Sonnet 4—to automatically map regulatory clauses to Android permissions. A structured prompting workflow (Appendix A) defines the model persona as an Android security and data-privacy expert and provides each policy document with instructions to: (i) identify prohibited data types, (ii) distinguish unconditional from conditional prohibitions, and (iii) propose corresponding AOSP permissions. We apply this prompt to regulations from India, Indonesia, Kenya, Nigeria, the Philippines, Thailand, and Pakistan [18, 22, 32, 35, 54, 65, 68]. For each suggested permission, we verify that it exists in AOSP and that its documented capabilities match the described data access. In head-to-head comparisons against our expert-derived mappings, GPT-4o Mini and Claude Sonnet 4 cover all prohibited data types, whereas Grok3 omits several entries. We therefore construct each country's prohibited-permission set as the union of GPT-4o Mini and Claude Sonnet 4 outputs, validated by expert review and produced in under two minutes per regulation.

For our static compliance analysis, we focus exclusively on *unconditional* prohibitions—permissions that regulators ban irrespective of user consent. Conditional prohibitions depend on timing and user interactions (e.g., only during loan or KYC flows) and are therefore better evaluated, at least in part, via dynamic analysis. Restricting the static check to unconditional bans provides a clear, objective baseline for assessing compliance. This yields three prohibited-permission sets

(1) **Country-specific prohibited permissions sets:** Derived directly from LLM analysis of national regulatory texts, distinguishing clearly between unconditional and conditional prohibitions (Table 1). Countries like Nigeria, Pakistan, and the Philippines specify explicit unconditional prohibitions, while Indonesia emphasize user consent without outright bans.
(2) **Google's prohibited permissions set:** As detailed in Section 2.1, Google's Financial Services Policy explicitly bans eight permissions for loan apps.
(3) **LoanWatch set of prohibited permissions set:** The LoanWatch set represents the union of Google's prohibited permissions and additional high-risk permissions prohibited by individual countries but not by Google. This addresses a critical gap where inconsistencies between national and platform-level prohibitions create exploitable loopholes. For example, while Google prohibits `READ_CONTACTS`, loan apps use `READ_CALL_LOG` to achieve the same goal, accessing recent contact information through call history, which provides more current data about user communications for debt collection purposes. By consolidating all permissions that provide access to sensitive data types prohibited by either framework, the set prevents such circumvention tactics and ensures adequate privacy protection.

3.3 Static Analysis

Static analysis is the second step in our methodology, aimed at identifying non-compliant behavior in loan apps without executing them. This phase examines each app's code and configuration to address both of our research questions: it identifies the extent of regulatory violations by analyzing declared permissions against guidelines (RQ1), and reveals the technical mechanisms through which apps can access and collect sensitive data by examining API calls and data flow paths (RQ2). By uncovering permission-based violations and mapping potential data access pathways, static analysis provides foundational evidence for understanding both compliance failures and data exploitation capabilities, setting the stage for dynamic validation. The static analysis pipeline consists of four core components: (i) APK Extraction and File Processing, (ii) Permission Analysis, (iii) Sensitive API Analysis, and (iv) Data Flow Analysis.

3.3.1 ***APK Extraction and File Processing.*** In phase 2 of our methodology, we process each app with `Androguard` [6], a static-analysis framework for Android apps. Androguard unpacks the APK and exposes key artefacts for later stages: the `AndroidManifest.xml` (declared permissions and components), the `classes.dex` file (Dalvik bytecode), and related resources. We store these artefacts and use them as the input to our permission analysis, sensitive-API analysis, and data-flow analysis.

3.3.2 ***Permission Analysis.*** Android applications must explicitly declare all permissions they intend to use in their `AndroidManifest.xml`, covering both installation-time and runtime permissions [7]. Using the manifests obtained in the previous step, LoanWatch parses and aggregates declared permissions for all apps in our dataset. It then compares these permissions against three reference sets: (i) the country-specific prohibited-permission set (applied only to apps targeting that country), (ii) Google's prohibited-permission set, and (iii) the LoanWatch prohibited-permission set.

We classify any app that declares at least one permission from these sets as a violating app (phase 3 of Figure 2), regardless of whether the app ultimately invokes the guarded APIs. While declaration does not prove misuse, it indicates intent and technical capacity to access sensitive data—sufficient grounds for regulatory concern when permissions are explicitly banned to prevent coercive or invasive practices. For every flagged app, we identify the target jurisdiction and compile structured reports that can be shared with the corresponding national regulator and Google.

3.3.3 ***Sensitive API Analysis.*** Permission analysis alone identifies apps requesting prohibited permissions but does not confirm whether these permissions are actively used. Our sensitive API analysis extends this by systematically examining each app's codebase to identify the presence of sensitive API calls that correspond to prohibited permissions. This step is essential as it verifies that apps have the technical capability to programmatically invoke restricted APIs beyond mere permission declarations.

Permission-to-API Mapping Construction: Android's permission system changes across API levels, so accurately linking sensitive APIs to prohibited permissions requires coverage across all versions present in our dataset (API 16–36). No single source provides this, so we combine multiple mappings: Axplorer for API 16–25 [16], NatiDroid for API 26–29 [42], and Barzolevskaia et al. for API 30–33 [20]. For API 34–36, we perform our own static analysis using Soot, generating call graphs and control-flow analyses over AOSP framework JARs to identify permission checks in newer system APIs.

API Usage Detection: With comprehensive permission-to-API mappings established, we next identify which sensitive APIs are actually invoked within each loan app's code. We apply Androguard [6] to analyze each app's `classes.dex` file, examining the Dalvik bytecode for calls to the sensitive APIs identified in our mapping phase. We record each matched API call along with its corresponding prohibited permission, noting the exact method and class context. Since we analyze legitimate loan applications rather than malicious software, we expect minimal sophisticated obfuscation techniques that

Table 1: LLM-Generated Country-Specific Permission Mappings

Country	Category	Data Types	Mapped Android Permissions
India	Unconditional	File & media, Contact list, Call logs, Telephony, Biometric	READ_EXTERNAL_STORAGE, WRITE_EXTERNAL_STORAGE, READ_MEDIA_IMAGE, READ_MEDIA_VIDEO, READ_MEDIA_AUDIO, MANAGE_EXTERNAL_STORAGE, READ_CONTACTS, WRITE_CONTACTS, GET_ACCOUNTS, READ_CALL_LOG, WRITE_CALL_LOG, PROCESS_OUTGOING_CALLS, READ_PHONE_STATE, CALL_PHONE, ANSWER_PHONE_CALLS, ADD_VOICEMAIL, USE_SIP, USE_FINGERPRINT, USE_BIOMETRIC
	Conditional	Camera, Microphone, Location	CAMERA, RECORD_AUDIO, ACCESS_FINE_LOCATION, ACCESS_COARSE_LOCATION, ACCESS_BACKGROUND_LOCATION
Indonesia	Conditional	All data types (with consent)	All permissions (with explicit user consent)
Kenya	Unconditional	Contact access for debt collection	READ_CONTACTS, WRITE_CONTACTS, GET_ACCOUNTS, READ_CALL_LOG, WRITE_CALL_LOG
Nigeria	Unconditional	File & media, Contact list, Call logs	READ_EXTERNAL_STORAGE, WRITE_EXTERNAL_STORAGE, READ_MEDIA_IMAGE, READ_MEDIA_VIDEO, READ_MEDIA_AUDIO, MANAGE_EXTERNAL_STORAGE, READ_CONTACTS, WRITE_CONTACTS, GET_ACCOUNTS, READ_CALL_LOG, WRITE_CALL_LOG, PROCESS_OUTGOING_CALLS
Pakistan	Unconditional	Contact list, Photo gallery, SMS	READ_CONTACTS, WRITE_CONTACTS, GET_ACCOUNTS, READ_EXTERNAL_STORAGE, WRITE_EXTERNAL_STORAGE, READ_MEDIA_IMAGE, READ_MEDIA_VIDEO, READ_MEDIA_AUDIO, MANAGE_EXTERNAL_STORAGE, READ_SMS, SEND_SMS
	Conditional	Camera	CAMERA
Philippines	Unconditional	Contact list, Email, Social media	READ_CONTACTS, WRITE_CONTACTS, GET_ACCOUNTS
	Conditional	Camera, Photo gallery	CAMERA, READ_EXTERNAL_STORAGE, WRITE_EXTERNAL_STORAGE, READ_MEDIA_IMAGE, READ_MEDIA_VIDEO, READ_MEDIA_AUDIO, MANAGE_EXTERNAL_STORAGE
Thailand	Conditional	All personal data (with consent)	All permissions (with explicit user consent)

would deliberately hide API usage patterns. This process correlates sensitive API calls with declared permissions to build an evidence-based profile of each app's technical capability to access sensitive user data.

3.3.4 ***Data Flow Analysis.*** We perform static data flow analysis to assess whether loan apps can exfiltrate sensitive user data, such as contacts, call logs, or media. While permission and API analysis confirms technical capability, it does not establish actual data transfer paths. To bridge this gap, we employ FlowDroid [14], a static taint-analysis tool widely used for static data flow detection [41, 67, 69].

We configure FlowDroid to detect data flows from sensitive APIs (sources) identified during permission analysis, such as `ContactsContract.Contacts`, `CallLog.Calls`, and `MediaStore`, to network-related methods (sinks) typically used for data transmission. Our sink definitions include FlowDroid's default set (`HttpURLConnection.getInputStream`, `java.net.URLConnection`, `org.apache.http.HttpResponse.getEntity`) [41], and popular third-party libraries (`OkHttp`, `Retrofit`, `OkGo`). This configuration allows us to identify apps with plausible static data flows from sensitive data access to network endpoints, highlighting potential data exfiltration (phase 5 in Figure 2).

While FlowDroid may produce false positives due to over-approximation or struggle with obfuscation [37, 41], we mitigate these limitations by focusing our analysis on APIs associated with declared permissions. This targeted approach reduces noise and flags apps demonstrating realistic technical pathways for sensitive data exfiltration, thereby highlighting genuine operational privacy risks.

3.4 Dynamic Analysis

Dynamic analysis validates the runtime behavior of loan apps by directly observing how they access, collect, and transmit sensitive user data during execution [29, 37, 43, 64]. While static analysis identifies declared permissions and potential data-flow paths, dynamic analysis provides concrete evidence of whether sensitive APIs are invoked in practice, under which conditions, and at what *time* in the user journey. This temporal dimension is critical for regulations that permit certain data accesses only in specific contexts (e.g., during loan applications or Know Your Customer (KYC) verification). In this work, we apply dynamic analysis to (i) apps that static analysis flags as high risk (e.g., due to prohibited permissions or suspicious data-flow paths), and (ii) all apps from consent-focused jurisdictions such as Indonesia, where regulations emphasize timing and consent rather than outright bans. Our structured workflow comprises two primary stages: (i) device setup and instrumentation, establishing controlled runtime monitoring conditions; and (ii) app execution and method tracing, systematically capturing detailed evidence of sensitive data interactions, transmission paths, and remote storage endpoints.

Core lending workflows often require verifiable personal information such as government-issued IDs, locally registered phone numbers, selfies, or bank-account details tied to domestic infrastructure. Within the scope of this study, it is

neither feasible nor appropriate to supply such information or fabricate borrower identities at scale, and many apps additionally enforce geo-blocking, SIM-based checks, or emulator-detection defenses. As a result, we cannot systematically drive apps through full loan and KYC workflows.

Consequently, we explicitly focus our dynamic analysis on *pre-registration* behaviors. Several loan apps request sensitive permissions immediately upon launch, forcing users to grant access before they can explore the app or understand its services. This coercive approach enables apps to collect sensitive user data even when users subsequently abandon the loan application process. Our analysis examines this pattern, focusing on whether apps access contacts, call logs, and other sensitive data immediately after initial permission grants, regardless of whether users complete the loan workflow.

3.4.1 ***Device Setup and Instrumentation.*** To simulate typical user environments, we conduct dynamic analysis on a rooted Huawei Nexus 6P, an affordable Android device representative of those commonly used by low-income individuals in our study regions. Root access allows us to bypass OS-level restrictions and enables low-level instrumentation for behavioral monitoring. Using Android Debug Bridge (ADB) [8], we install each loan app and prepare it for controlled execution.

For runtime inspection, we use Frida [60], a dynamic instrumentation framework that injects lightweight stubs into the app's runtime. These stubs enable external hooks to monitor method calls, allowing us to inspect arguments, return values, and object states without modifying the original app logic. We generate a custom Frida script for each app based on the sensitive data-flow paths identified during static analysis. These scripts target methods associated with sensitive sources and sinks—such as `ContactsContract.Contacts.CONTENT_URI` and `OkHttpClient.newCall()`—and are designed to log relevant contextual information, including accessed data and transmission endpoints. This automation ensures consistency across apps, with only minimal manual verification required to confirm proper hook attachment and logging functionality[1].

3.4.2 ***App Execution and Method Tracing.*** Following setup, we execute each instrumented app and monitor whether the sensitive data-flow paths flagged during static taint analysis are activated at runtime under pre-registration conditions. Our objective is to validate whether sensitive permissions declared in the manifest are actually exercised by confirming that associated APIs are invoked and used to transmit user data. Frida's instrumentation logs both input (e.g., contact records, SMS content) and output (e.g., HTTP requests, endpoints) in real time, offering direct evidence of how the app handles sensitive data after permissions are granted.

A key limitation arises from Android's on-demand class loading. Unlike systems that load the full codebase at startup, Android loads classes only when invoked. This means that some sensitive API methods may not be loaded during our pre-registration analysis phase, as they are triggered only by specific user actions or authentication flows that occur later in the app lifecycle. To address this limitation, we explicitly interpret our findings as characterizing sensitive data access during the initial app execution phase—specifically, APIs that are invoked immediately when apps request and obtain sensitive permissions upon launch.

While this approach may miss apps that call sensitive APIs only in later-loaded classes, it effectively captures those with coercive permission-requesting behavior that immediately access and transmit sensitive data upon permission grant. For these apps, we obtain concrete runtime evidence of exploitation during the pre-registration phase, showing that users' data can be exfiltrated even if they never complete a loan application.

[1]Our regulatory text-to-permission mappings, prompting templates, and analysis scripts are available at https://github.com/marshalwahlexyz1/-The-Cost-of-Convenience-Identifying-Analyzing-and-Mitigating-Predatory-Loan-Applications-on-Android. [48]

4 Results

In this section, we present the results of our cross-country measurement of loan-app compliance with regulatory guidelines across Indonesia, Kenya, Nigeria, Pakistan, and the Philippines. We apply LoanWatch to a dataset of 435 loan apps, including both approved and delisted apps from public registries, to identify privacy violations and validate runtime behaviors. Together, the static and dynamic analyses address our research questions on the extent of regulatory violations and the mechanisms of unauthorized data access, exposing enforcement gaps and informing stricter oversight.

4.1 Static Analysis Results

We first present the results of our static analysis phase: permission analysis, sensitive-API analysis, and data-flow analysis. Together, these components use manifests, bytecode, and source-to-sink paths to quantify compliance violations, characterize policy mismatches, assess the impact of our harmonized prohibited-permission set, and identify apps technically capable of exfiltrating sensitive data without informed consent.

4.1.1 ***Widespread Non-Compliance.*** Table **??** summarizes violations for apps listed as Approved (available on Google Play at analysis time) and Delisted in each country's registry. Across all 435 loan apps analyzed, **188 (43.2%)** violate at least one *country* policy, and **191 (43.9%)** violate Google's Financial Services Policy (FSP). Among these, 141 of the 188 apps violating country policy are approved apps, and 147 of the 191 apps violating Google's FSP are approved apps.

Violations among delisted apps are unsurprising, as they may have been removed due to non-compliance. The high rate of violations among *approved* apps is more concerning. Despite a dual-layered regulatory framework requiring compliance with both Google's FSP and country-specific regulations, many approved apps still request sensitive permissions explicitly prohibited under these policies. This widespread non-compliance reveals critical enforcement gaps, allowing non-compliant apps to remain on Google Play and exposing

borrowers to persistent privacy risks and predatory practices such as harassment and blackmail.

4.1.2 ***Policy Mismatches and Implications.*** To characterize the gaps between policy frameworks and their real-world impact, we focus on *mismatch* apps—those that violate only one of the two policy frameworks—and identify the specific permissions that create these gaps. We refer to these mismatch cases as *asymmetric* violations: apps that violate either a country's rules or Google's FSP, but not both. Conversely, apps that either violate both frameworks or comply with both form *symmetric* cases, where national rules and Google's policy are effectively aligned. We focus here on *approved* apps to show how asymmetric violations allow developers to exploit inconsistencies between national regulations and Google's FSP to obtain prohibited data while remaining compliant under one framework.

Google's FSP prohibits a fixed set of eight high-risk permissions targeting channels linked to social harm, while several national regulations in our study instead enumerate and prohibit specific "social-harm" data types (contacts, call logs, SMS). This fundamental difference in approach creates two operational forms of asymmetric violations:

(1) **Country-only violations:** Apps request permissions such as `READ_CALL_LOG` or `READ_SMS` that violate national regulations but not Google's FSP, enabling access to social graphs and communication content for predatory practices.
(2) **Google-only violations:** Apps request permissions such as `ACCESS_FINE_LOCATION` or `READ_EXTERNAL_STORAGE` that violate Google's FSP but not national regulations, enabling location-based threats or unauthorized media access.

Below, we detail these mismatches for each country, focusing on approved apps.

Indonesia. Indonesia's regulations do not explicitly prohibit specific permissions, so no approved apps violate country-specific policies in our static analysis. However, **26 of 52 approved apps (50.0%)** violate Google's FSP by requesting permissions banned by Google but not addressed by Indonesian regulations. The most commonly requested prohibited permissions across these 26 apps include `ACCESS_FINE_LOCATION` (21), `WRITE_EXTERNAL_STORAGE` (14), `READ_EXTERNAL_STORAGE` (12), `READ_CONTACTS` (4), `QUERY_ALL_PACKAGES` (1), and `READ_PHONE_NUMBERS` (1). All Indonesian violators are thus *asymmetric* Google-only cases; the remaining 26 apps are symmetric (compliant under both frameworks).

Kenya. Among 33 approved apps, **13 (39.4%)** violate Kenya's policy and **15 (45.5%)** violate Google's FSP. There are 4 country-only violators (`READ_CALL_LOG` in 2, `GET_ACCOUNTS` in 2) and 6 Google-only violators (`READ_EXTERNAL_STORAGE` in 6, `WRITE_EXTERNAL_STORAGE` in 4, `ACCESS_FINE_LOCATION` in 3, `READ_PHONE_NUMBERS` in 1). The remaining 9 violators breach both frameworks, and 14 apps comply with both, so Kenya exhibits 10 asymmetric apps (4 country-only, 6 Google-only) and 23 symmetric apps (9 violating both, 14 violating neither).

Nigeria. Among 191 approved apps, **92 (48.2%)** violate Nigeria's policy and **69 (36.1%)** violate Google's FSP. There are 26 country-only violators—mostly requesting `READ_CALL_LOG` (23, with 2 also requesting `GET_ACCOUNTS`) and `GET_ACCOUNTS` alone (3)—and 3 Google-only violators that do not request any Nigeria-prohibited permissions but do request permissions banned by Google's FSP (all three request `ACCESS_FINE_LOCATION`). The remaining 66 violators breach both frameworks, and 96 apps comply with both, yielding 29 asymmetric apps (26 country-only, 3 Google-only) and 162 symmetric apps (66 violating both, 96 violating neither).

Pakistan. Among 11 approved apps, **9 (81.8%)** violate both Pakistan's policy and Google's FSP, with no country-only or Google-only violators. The remaining 2 apps comply with both frameworks. Pakistan therefore exhibits exclusively symmetric behavior: 9 symmetric violators and 2 symmetric compliant apps, and no asymmetric cases.

Philippines. Among 56 approved apps, **27 (48.2%)** violate the Philippines' policy and **28 (50.0%)** violate Google's FSP. There is 1 country-only violator (`GET_ACCOUNTS` in 1 app) and 2 Google-only violators—apps that do not request any Philippines-prohibited permissions but request one or more permissions banned by Google's FSP. Across these 2 apps, the flagged permissions include `ACCESS_FINE_LOCATION` (1) and both `READ_EXTERNAL_STORAGE` and `WRITE_EXTERNAL_STORAGE` (1). The remaining 26 violators breach both frameworks, and 27 apps comply with both, so the Philippines exhibits 3 asymmetric apps (1 country-only, 2 Google-only) and 53 symmetric apps (26 violating both, 27 violating neither).

Implications of policy mismatches. Asymmetric violations are common across countries, showing that mismatches between national regulations and Google's FSP are systemic and allow high-risk apps to remain on Google Play. Symmetric violators mark points of agreement on unacceptable data access, while country-only and Google-only violators reveal each framework's blind spots: Kenya, Nigeria, and Pakistan often ban "side-channel" and abuse-linked permissions such as `READ_CALL_LOG` and `READ_SMS`, which Google permits, whereas Google bans permissions such as `ACCESS_FINE_LOCATION` and `READ_EXTERNAL_STORAGE`, which no country restricts. These gaps create compliance "back doors," allowing developers to exploit unregulated vectors without losing abusive functionality. In some cases, such as Indonesia, policy gaps are predominantly one-sided, rendering Google the sole potential line of defense, yet our results show its enforcement is weak.

4.1.3 ***Importance of the LoanWatch Harmonized Permission Set.*** Our LoanWatch harmonized set consolidates all permissions prohibited by either national regulations or Google's FSP, revealing the full scope of non-compliance and preventing apps from bypassing restrictions via alternative permissions that enable the same data access.

Applying this harmonized policy shows that **251/342 (73.4%)** of approved apps violate the combined prohibition set, compared to 141 (41.2%) under country rules alone and 147 (43.0%) under Google's FSP alone. This jump reflects

Table 2: Summary of Loan App Violations by Country

	Regulator Registry		Violate - Country Policy		Violate - Google Policy		Violate - LoanWatch Harmonized	
Country	**Approved**	**Delisted**	**Approved**	**Delisted**	**Approved**	**Delisted**	**Approved**	**Delisted**
Indonesia	52	0	0/52 (0%)	0/0 (N/A)	26/52 (50.0%)	0/0 (N/A)	27/52 (51.9%)	0/0 (N/A)
Kenya	32	0	13/32 (40.6%)	0/0 (N/A)	15/32 (46.9%)	0/0 (N/A)	24/32 (75.0%)	0/0 (N/A)
Nigeria	191	54	92/189 (48.7%)	35/54 (64.8%)	69/189 (36.5%)	30/54 (55.6%)	145/189 (76.7%)	49/54 (90.7%)
Pakistan	11	28	9/11 (81.8%)	9/28 (32.1%)	9/11 (81.8%)	9/28 (32.1%)	9/11 (81.8%)	9/28 (32.1%)
Philippines	56	10	27/55 (49.1%)	3/11 (27.3%)	28/55 (50.9%)	5/11 (45.5%)	46/55 (83.6%)	8/11 (72.7%)
Total	**342**	**92**	141/339 (41.6%)	47/92 (51.1%)	147/339 (43.4%)	44/92 (47.8%)	251/339 (74.0%)	66/92 (71.7%)

Note: Violations are presented as a fraction (Violating Apps / Total Analyzed Apps) followed by the percentage of the total pool.

two key gaps: apps (i) exploit side-channel permissions (e.g., `READ_CALL_LOG` reconstructs contact graphs when `READ_CONTACTS` is blocked), and (ii) leverage omissions where one framework permits what the other prohibits (e.g., Google bans `ACCESS_FINE_LOCATION` while some national policies do not). These findings show that harmonized, policy-driven permission mapping is essential to capture the full space of sensitive data access beyond what any single framework can see.

Enforcement outcomes. We disclosed our findings to Google's Android Security team and country regulators in two categories: (1) 147 apps already violating existing policies (Google's FSP or national regulations), and (2) an additional 104 apps that comply with individual policies but violate the harmonized LoanWatch set. Alongside the app lists, we provided Google with the harmonized set and the rationale for including additional permissions, such as the fact that `READ_CALL_LOG` can be used as a functional substitute for `READ_CONTACTS`. Google formally acknowledged receipt, confirmed removal of 93 flagged apps from the Play Store, and continues to investigate remaining cases. As of the time of writing, Nigeria's FCCPC has also acknowledged our submissions and reported ongoing coordination with Google.

4.1.4 ***Apps Extracting and Exfiltrating Sensitive User Data.*** Beyond permission requests, our static analysis examines how non-compliant loan apps can exploit sensitive permissions through API usage and data-flow paths. We analyze 317 apps flagged by the LoanWatch benchmark—spanning both approved and delisted apps—and confirm that each invokes at least one sensitive API (e.g., `ContactsContract`, `CallLog`, `MediaStore`, `LocationManager`) and holds the `INTERNET` permission alongside networking libraries such as `HttpURLConnection` and `OkHttp`. Static taint analysis with FlowDroid reveals that 166 apps contain feasible source-to-sink paths linking sensitive data sources to network transmission methods. Specifically, 59 apps have flows that can leak location data, 46 leak storage contents, 16 exfiltrate media files, 15 expose device identifiers (e.g., SIM serial numbers), 14 transmit contact information, 12 send SMS content, and 8 enumerate installed packages. Several apps exhibit multiple leakage channels, indicating deeply embedded exfiltration logic.

These technical findings align with documented real-world abuses of borrower harassment, blackmail, and public shaming [1, 39, 52, 53] and confirm that many loan apps do more than *declare* prohibited permissions—they embed working code paths to harvest and transmit private user data. By moving from simple permission-violation counts to concrete evidence of exploit-ready data-flow paths from access to exfiltration, our results underscore the need for stronger, technically grounded enforcement mechanisms.

4.2 Dynamic Analysis Results

To validate whether the static taint paths we identified translate into real-world data exfiltration, we performed dynamic analysis on the 166 apps flagged by FlowDroid. Of these, 148 launched successfully on a rooted test device; the remaining 18 crashed, typically due to emulator- or root-detection defenses. Using Frida-based instrumentation targeted at sensitive APIs and network libraries, we monitored runtime invocations and outbound data flows without progressing into full registration workflows, which would require authentic personal data and raise ethical concerns. Our tests therefore focus on the *pre-registration* phase, from first launch through initial permission dialogs and basic navigation.

We observe a coercive permission-requesting pattern in many apps: they request high-risk permissions such as `READ_CONTACTS`, `READ_SMS`, `ACCESS_FINE_LOCATION`, or `QUERY_ALL_PACKAGES` at startup and terminate or block further use if permissions are denied—effectively forcing users to consent before any meaningful interaction. When these permissions are granted, **37 of the 148** functioning apps immediately begin transmitting sensitive data upon launch, before any user registration or loan application process.

A particularly aggressive subset of six apps not only enforce these permission gates but also trigger exfiltration of contacts, SMS messages, image metadata, and installed-package lists within the launcher activity itself (declared via `action.MAIN` and `category.LAUNCHER`), directly undermining Android's runtime-permission model and the informed-consent expectations embedded in both national regulations and Google's Financial Services Policy. Beyond these six immediately exfiltrating apps, we observe additional unauthorized behaviors: 29 apps transmit location data on startup, and 2 apps send

complete package lists without user interaction. For the remaining apps, we do not observe pre-registration exfiltration; however, the exploit-ready code paths uncovered by static analysis indicate that sensitive data can still be harvested later in the loan or KYC workflow.

Dynamic analysis also clarifies how consent-focused regulations play out in practice. For Indonesia, whose guidelines emphasize user consent rather than enumerating hard bans, we ran dynamic analysis on all 26 approved apps that violate only Google's FSP in our static checks. Under our pre-registration workflows, none of these apps accessed contacts, call logs, SMS, or storage immediately upon launch; sensitive access was only triggered deeper in the loan or KYC flows. Thus, while these apps clearly violate Google's FSP, our analysis did not observe launch-time violations of Indonesia's timing-based restrictions under the tested conditions.

In summary, our analysis of 434 loan apps across five countries reveals a regulatory landscape that systematically fails to protect users. While 73.4% of approved apps (251/342) violate our harmonized prohibition set, only 41.2% violate country-specific rules and 43.0% violate Google's policy alone, showing how apps exploit gaps between frameworks to access prohibited data while remaining technically compliant under at least one of them.

Static analysis shows that all 317 apps flagged by LoanWatch invoke sensitive APIs together with network-capable libraries, creating the technical infrastructure for exfiltration. Dynamic analysis confirms that this infrastructure is actively abused: 37 apps begin transmitting sensitive data immediately upon launch, and 6 exfiltrate multiple categories of data (contacts, SMS, media, installed apps) from the launcher activity itself, before users can meaningfully engage with the app. This systematic, early-stage collection of highly sensitive information enables the harassment, shaming, and blackmail practices documented in prior work, and highlights how reactive, complaint-driven enforcement leaves borrowers exposed even in the presence of formal regulations and platform policies.

5 Discussion

This section consolidates our findings to highlight core challenges in regulating digital lending ecosystems. We discuss how failures in a dual-layered oversight model and fragmented policy design enable systematic non-compliance, link these technical violations to documented societal harms, and outline recommendations for regulators, platforms, developers, and users.practices in financial

Failures in a Dual-Layered Oversight Model. Despite the dual-layered framework involving national regulators and Google's Financial Services Policy, our analysis reveals persistent enforcement gaps. On the regulatory side, loan apps that request explicitly prohibited permissions still appear in "approved" registries. Since such violations are immediately visible from the `AndroidManifest.xml`, this strongly suggests that technical checks (e.g., automated manifest inspection) are not being systematically applied during licensing or renewal. The absence of version-specific identifiers (e.g., APK hashes) in registries further prevents external verification that the app reviewed by regulators matches the one distributed to users.

On the platform side, Google's FSP is intended to provide an additional protection layer, yet we find numerous non-compliant apps available on Play even when they request permissions that Google itself bans. These violations could be detected via straightforward static analysis, but current practice appears to rely heavily on developer self-disclosure and complaint-driven enforcement. Together, these gaps in both national and platform oversight allow the same harmful apps to pass through multiple checkpoints and remain available to borrowers despite clear technical violations.

Policy Fragmentation and the Case for a Harmonized Set. Our policy-to-permission mapping (Table 1) shows that loan app regulation is highly fragmented. Some countries (Nigeria, Pakistan, Philippines) unconditionally ban permissions such as `READ_CALL_LOG` and `READ_SMS` to prevent contact harvesting and harassment, while others (e.g., Indonesia, Thailand) emphasize consent and context rather than enumerating hard bans. Google's FSP, in contrast, proscribes a fixed set of eight permissions based on its own assessment of harassment and abuse risk.

This misalignment creates exploitable loopholes. Apps can reconstruct social graphs via call logs when `READ_CONTACTS` is blocked, or leverage location and storage permissions to facilitate threats or reputational harm even when local regulations do not explicitly address these channels. Our LoanWatch harmonized set, which merges Google's prohibited permissions with additional high-risk permissions banned by at least one national regulator, shows that 73.4% of approved apps would be flagged under a consolidated policy, compared to about 40–43% under either framework alone.

These results motivate a harmonized, conservative baseline: Google should extend its FSP for loan apps to cover all permissions that national authorities have already identified as harassment- or abuse-linked. Under such a policy, any loan app requesting a permission in the consolidated set would fail pre-approval checks, closing common "back doors" where developers swap one sensitive permission for another functionally equivalent one. Because these checks can be implemented via standard static analysis over the manifest, they are technically simple yet materially strengthen protection.

Societal Impact of Technical Violations. These violations harm not only borrowers but also people who never interact with loan apps. Prior work documents coercive debt-collection practices in which lenders use borrowers' private information to send threats, shaming messages, and defamatory broadcasts to their social networks [1, 19, 26, 52]. Our static and dynamic analyses show the technical basis for this behavior: harvesting contact graphs, call histories, SMS content, location traces, media, and app inventories, and exfiltrating this data to remote servers soon after permission grant.

For borrowers, launch-time permission coercion and early data exfiltration create an immediate power imbalance: apps request high-risk permissions, block use if users refuse, and

begin transmitting sensitive data before users can review terms or understand downstream uses. For third parties, the harms are entirely non-consensual: contacts appearing in a borrower's address book or call log may receive abusive or defamatory messages despite never installing the app or agreeing to its terms. These "secondary victims" face reputational risks and relationship strain, while at a community level such practices erode trust in digital financial services and can push vulnerable users back toward informal credit channels. By linking permission use, API-level behavior, and observed exfiltration patterns to these direct and third-party harms—including documented cases of self-harm and suicide [45, 59, 61, 76]—our results show that privacy violations in digital lending are not merely procedural non-compliance but a core enabler of coercive practices.

Recommendations for Stakeholder Groups. To address our third research question about what regulatory, technical, and design interventions can mitigate non-compliance, we offer recommendations for regulators, app platforms, developers, and users.

For regulators: Documentation-based review for licensing purposes is not enough. Regulators should adopt scalable auditing tools—such as static permission checks and code inspection—during both initial app approval and license renewal. Public registries should include version-specific identifiers like APK hashes to support external verification. In addition, regulators should establish clear channels for user complaints and whistleblowing.

For app stores and platforms: Google Play's reliance on self-disclosure leaves compliance gaps. We recommend automatic permission checks at submission time, especially for permissions prohibited under Google's own Financial Services Policy. The platform should also expand its policy to include omitted high risk permissions, which are banned by several national regulators.

For developers: Our analysis shows that several compliant loan apps deliver core lending functionality without requesting sensitive permissions. This suggests that it is technically feasible to provide digital credit services without invasive data access. Developers targeting regulated markets may benefit from minimizing permission use and reviewing their app's data access practices particularly for permissions flagged by regulators or platform policies.

For users: While imperfect, public registries can help users avoid unlicensed or delisted apps. Users should report apps that request unnecessary sensitive permissions either via app store feedback or national consumer protection channels. Nigeria's FCCPC, for example, maintains a reporting email for users to flag non-compliant apps [33]. Still, privacy protection should not rely solely on user vigilance; systemic enforcement is essential.

Limitations and Future Directions. LoanWatch relies on public registries to identify approved and delisted loan apps. As such, our dataset is bounded by countries that maintain accessible registries—namely Nigeria, Indonesia, Kenya, Pakistan, and the Philippines. Consequently, our coverage excludes countries like India or Thailand and misses unlicensed or gray-market apps, which may operate entirely outside regulatory frameworks. While this constraint narrows scope, it aligns with our objective of evaluating apps that are officially recognized by regulators and intended for lawful distribution. Future work could expand beyond licensed apps to capture unauthorized or unlisted apps via behavioral or linguistic heuristics.

Ethical and operational constraints also limit our dynamic analysis. Simulating the full loan lifecycle (registration, approval, repayment, default) requires valid and verifiable identity credentials—such as phone numbers, national IDs, or bank verification numbers—to trigger backend authentication and OTP verification. Dummy or anonymized data cannot replicate these processes, restricting our visibility into post-authentication behaviors. Future work could explore partnerships with regulators or establish controlled testing environments that ethically enable full lifecycle evaluation. Despite these limitations, our methodoogy offers a conservative but reliable lower-bound assessment of compliance and risk, flagging apps violating regulations and users privacy.

6 Related Work

Android Permissions and Privacy Violation Detection. The Android permission model has been widely studied for over-permissioning, covert data access, and the gap between declared permissions and actual behavior. Felt et al. [34] showed that many apps request more permissions than needed, and tools such as Stowaway [34] detect over-privileged apps by comparing requested permissions with API use. Khatoon et al. [40] further found that free apps often exploit permissions to harvest sensitive data under ostensibly legitimate use, while later work documents systematic collection and unauthorized sharing of contacts, call logs, and SMS, raising consent and transparency concerns [17, 24, 25]. To uncover such practices, prior work developed static, dynamic, and hybrid analysis frameworks: AndroidLeaks [37], PVDetector [69], and FlowDroid [13] perform taint-based static analysis; Reardon et al. [62] and Sarwar et al. [15] reveal runtime circumvention via covert channels; and Hush [41] combines static filtering with runtime confirmation. Our work directly builds on these techniques—especially static taint analysis and dynamic instrumentation—but applies them within a regulatory compliance lens tailored to loan apps' prohibited data flows rather than general over-permissioning.

General Privacy Compliance Tools and Methods. Previous work on regulatory compliance in mobile apps has primarily focused on assessing the alignment between stated privacy practices and actual data handling behaviors. Tools like MAPS [79], POLICHECK [5], PTPDroid [69], and VioDroid-Finder [23] use natural language processing and static analysis to detect discrepancies between app's policy claims and code behavior, particularly regarding sensitive data collection (e.g.,

location, contacts, identifiers). VioDroid-Finder, for instance, analyzes policy structures to identify declared data access and cross-checks this against app permissions and API calls. Beyond policy-code consistency, several studies assess compliance with external regulations like GDPR and COPPA. Nguyen et al. [57] identify unauthorized third-party data sharing without user consent, violating GDPR, while Fan et al. [31], Xiang et al. [77], McConkey et al. [49], and Reyes et al. [44] evaluate violations related to consent, policy completeness, and lawful basis. These approaches provide valuable methodological foundations for our work, particularly policy-to-code mapping concept. However, existing tools focus primarily on established Western regulations (GDPR, COPPA). Our work extends this paradigm by introducing LLM-assisted policy to permission mapping to handle diverse emerging market regulations and by focusing on a specific high-risk app category rather than general compliance.

App-Category-Specific Privacy and Compliance Audits. Recent work has increasingly recognized that different app categories face distinct regulatory landscapes and risk profiles that generic compliance audits may overlook. Prior work highlights the critical need for focused audits of specific Android app subcategories due to their unique risks, access to highly sensitive data, and stringent regulatory requirements. For instance, financial apps have been evaluated for PCI DSS compliance using static analysis and UI inference [46], while female mHealth apps were scrutinized for sensitive data handling post-Roe v. Wade through privacy policy reviews and usability inspections [47]. IoT companion apps have been analyzed for excessive permissions via static and network analysis [56], and VR applications have been examined for GDPR and PIPL compliance using static analysis and NLP to identify policy inconsistencies [78].

Within the financial app category specifically, considerable research has documented pervasive privacy breaches and aggressive debt-collection practices. Bowers et al. [63] and Cranor et al. [27] highlight how many finance apps omit clear disclosures about their data usage, while Butcher and James [21] document microfinance exploitation in Latin America. In Kenya, Munyendo et al. [52] rely on semi-structured user interviews to explore privacy concerns about mobile loan apps' data-collection practices and discover that users often consent to invasive permissions despite misgivings, owing to financial necessity. A similar interview-based approach by Aggarwal et al. [1] in India identifies predatory digital lending tactics involving psychological harassment and intimidation of borrowers. Akgul et al. [3] similarly find recurring complaints of data theft by loan apps in Nigeria and Indonesia. Meanwhile, Ndruru et al. [55] and Sutedja et al. [72] analyze Indonesian online lending cases, uncovering personal-data leaks, unauthorized access, and systemic harassment facilitated by gaps in enforcement.

Our methodology bridges these research streams by combining category-specific focus with technical compliance measurement and established privacy violation detection techniques. We focus specifically on loan apps because they represent a uniquely critical subcategory with distinct regulatory and societal implications. Unlike other app categories, loan apps operate under dual-layered regulatory constraints—both Google's Financial Services Policy (FSP) and country-specific lending regulations—while mandatorily collecting highly sensitive, non-anonymizable user data (government IDs, bank verification numbers, contacts, SMS, media files). Most critically, permission misuse in this domain directly enables coercive debt collection practices that harm borrowers and their social networks.

To address the compliance challenges specific to this domain, we present a reproducible compliance-audit methodology that combines LLM-assisted policy-to-permission mapping to automatically translate regulatory text into Android permissions for precise compliance checks. This provides the first scalable, cross-country measurement of loan apps privacy violations, confirming through code-level inspection that the privacy risks reported qualitatively in prior work are both widespread and technically traceable across multiple regulatory jurisdictions.

7 Conclusion

This study exposes systematic regulatory failure in digital lending oversight. Through LoanWatch, our reproducible audit methodology, we demonstrate that most approved loan apps across five countries violate data protection standards by exploiting enforcement and policy gaps to harvest sensitive user information for harassment and coercion. Our findings reveal widespread non-compliance, reactive enforcement, and misaligned policies that enable loan apps to systematically collect contact lists, location data, and communication records, information they weaponize against borrowers and their families. We call for immediate reform through harmonized permission standards, proactive technical enforcement, and strengthened coordination between platform providers and regulators to protect users.

Acknowledgments

We thank the anonymous reviewers for their comments. This work was supported by the National Science Foundation under Grants CNS-2419829 and CNS-2211576. We are grateful to the Google Android Security Team for their engagement during disclosure and mitigation efforts. We also thank the various National Regulatory bodies for their essential assistance in validating the loan app registries, regulatory policy intent, and their efforts to protect consumer privacy.

References

[1] Vaibhav Aggarwal, Neha Aggarwal, Barkha Dhingra, Shallu Batra, and Mahender Yadav. 2024. Predatory Loan Mobile Apps in India: A New Form of Cyber Psychological Manipulation. In *ASU International Conference in Emerging Technologies for Sustainability and Intelligent Systems*

(ICETSIS). IEEE, Manama, Bahrain, 1918–1922. doi:10.1109/ICETSIS61505.2024.10459589

[2] Aghogho Udi. 2023. New study shows 27% of Nigerians now use loan apps to keep up with their expenses. https://nairametrics.com/2023/10/22/new-study-shows-27-of-nigerians-now-use-loan-apps-to-keep-up-with-their-expenses.

[3] Omer Akgul, Sai Teja Peddinti, Nina Taft, Michelle L. Mazurek, Hamza Harkous, Animesh Srivastava, and Benoit Seguin. 2024. A decade of privacy-relevant android app reviews: large scale trends. In *Proceedings of the 33rd USENIX Conference on Security Symposium* (Philadelphia, PA, USA) *(SEC '24)*. USENIX Association, USA, Article 285, 18 pages.

[4] Ryan Amos, Gunes Acar, Eli Lucherini, Mihir Kshirsagar, Arvind Narayanan, and Jonathan Mayer. 2021. Privacy Policies over Time: Curation and Analysis of a Million-Document Dataset. In *Proceedings of the Web Conference 2021* (Ljubljana, Slovenia) *(WWW '21)*. Association for Computing Machinery, New York, NY, USA, 2165–2176. doi:10.1145/3442381.3450048

[5] Benjamin Andow, Samin Yaseer Mahmud, Justin Whitaker, William Enck, Bradley Reaves, Kapil Singh, and Serge Egelman. 2020. Actions speak louder than words: entity-sensitive privacy policy and data flow analysis with POLICHECK. In *Proceedings of the 29th USENIX Conference on Security Symposium (SEC'20)*. USENIX Association, USA, Article 56, 18 pages.

[6] Androguard. 2014. Reverse engineering and pentesting for Android applications. https://github.com/androguard/androguard.

[7] Android Documentation. 2020. Permission - Android Developers. https://developer.android.com/guide/topics/manifest/permission.

[8] Android Documentation. 2024. Android Debug Bridge (adb). https://developer.android.com/tools/adb.

[9] APKCombo. 2024. APKCombo - APK Downloader and Installer. https://apkcombo.com.

[10] APKMonk. 2024. Download Android App Apks. https://www.apkmonk.com.

[11] APKPure. 2024. APKPure: Download APK on Android with Free Online APK. https://www.apkpure.com.

[12] Siddhant Arora, Henry Hosseini, Christine Utz, Vinayshekhar Bannihatti Kumar, Tristan Dhellemmes, Abhilasha Ravichander, Peter Story, Jasmine Mangat, Rex Chen, Martin Degeling, Thomas Norton, Thomas Hupperich, Shomir Wilson, and Norman Sadeh. 2022. A Tale of Two Regulatory Regimes: Creation and Analysis of a Bilingual Privacy Policy Corpus. In *2022 Language Resources and Evaluation Conference, LREC 2022*. European Language Resources Association (ELRA), Marseille, France, 5460–5472. https://aclanthology.org/2022.lrec-1.585/

[13] Steven Arzt, Siegfried Rasthofer, and Eric Bodden. 2017. The Soot-Based Toolchain for Analyzing Android Apps. In *IEEE/ACM 4th International Conference on Mobile Software Engineering and Systems*. IEEE, Buenos Aires, Argentina, 13–24. doi:10.1109/MOBILESoft.2017.2

[14] Steven Arzt, Siegfried Rasthofer, Christian Fritz, Eric Bodden, Alexandre Bartel, Jacques Klein, Yves Le Traon, Damien Octeau, and Patrick McDaniel. 2014. FlowDroid: precise context, flow, field, object-sensitive and lifecycle-aware taint analysis for Android apps. In *Proceedings of the 35th ACM SIGPLAN Conference on Programming Language Design and Implementation* (Edinburgh, United Kingdom) *(PLDI '14)*. Association for Computing Machinery, New York, NY, USA, 259–269. doi:10.1145/2594291.2594299

[15] Golam Sarwar Babil, Olivier Mehani, Roksana Boreli, and Mohamed-Ali Kaafar. 2013. On the effectiveness of dynamic taint analysis for protecting against private information leaks on Android-based devices. *International Conference on Security and Cryptography* (2013), 1–8. https://api.semanticscholar.org/CorpusID:16434324

[16] Michael Backes, Sven Bugiel, Erik Derr, Patrick McDaniel, Damien Octeau, and Sebastian Weisgerber. 2016. On Demystifying the Android Application Framework: Re-Visiting Android Permission Specification Analysis. In *25th USENIX Security Symposium (USENIX Security 16)*. USENIX Association, Austin, TX, 1101–1118. https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/backes_android

[17] Rebecca Balebako, Jaeyeon Jung, Wei Lu, Lorrie Faith Cranor, and Carolyn Nguyen. 2013. Little brothers watching you: raising awareness of data leaks on smartphones. In *Symposium On Usable Privacy and Security* (Newcastle, United Kingdom) *(SOUPS '13)*. Association for Computing Machinery, New York, NY, USA, Article 12, 11 pages. doi:10.1145/2501604.2501616

[18] Bank of Thailand. 2021. *Re: Regulations, Procedures and Conditions for Undertaking Business of Personal Loan under Supervision for Non-Bank Operator*. Bank of Thailand. https://www.bot.or.th/content/dam/bot/fipcs/documents/FPG/2560/EngPDF/25600206.pdf.

[19] Bankole Abe. 2023. Nigerians continue to patronise loan apps despite crude ways of fund recovery. https://www.icirnigeria.org/nigerians-continue-to-patronise-loan-apps-despite-crude-ways-of-fund-recovery/.

[20] Anna Barzolevskaia, Enrico Branca, and Natalia Stakhanova. 2024. Measuring and Characterizing (Mis)compliance of the Android Permission System. *IEEE Trans. Softw. Eng.* 50, 4 (April 2024), 742–764. doi:10.1109/TSE.2024.3362921

[21] Will Butcher and James Galbraith. 2019. Microfinance Control Fraud in Latin America. *Forum for Social Economics* 48, 1 (2019), 98–120. https://doi.org/10.1080/07360932.2015.1056203

[22] Central Bank of Kenya. 2021. *Central Bank of Kenya (Digital Credit Providers) Regulations*. Central Bank of Kenya. https://www.centralbank.go.ke/uploads/banking_circulars.

[23] Junren Chen, Cheng Huang, and Jiaxuan Han. 2024. VioDroid-Finder: automated evaluation of compliance and consistency for Android apps. *Empirical Software Engineering* 29 (2024), 64. https://api.semanticscholar.org/CorpusID:269580057

[24] Terence Chen, Imdad Ullah, Mohamed Ali Kaafar, and Roksana Boreli. 2014. Information leakage through mobile analytics services. In *Proceedings of the 15th Workshop on Mobile Computing Systems and Applications* (Santa Barbara, California) *(HotMobile '14)*. Association for Computing Machinery, New York, NY, USA, Article 15, 6 pages. doi:10.1145/2565585.2565593

[25] Saksham Chitkara, Nishad Gothoskar, Suhas Harish, Jason I. Hong, and Yuvraj Agarwal. 2017. Does this App Really Need My Location?: Context-Aware Privacy Management for Smartphones. *Proceedings of the Association for Computing Machinery on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1, 3, Article 42 (Sept. 2017), 22 pages. doi:10.1145/3132029

[26] Consumer Complaints Court. 2023. Loan Harassment. https://consumercomplaintscourt.com/loan-harassment-302/.

[27] Lorrie Faith Cranor, Pedro Giovanni Leon, and Blase Ur. 2016. A Large-Scale Evaluation of U.S. Financial Institutions' Standardized Privacy Notices. *ACM Trans. Web* 10, 3, Article 17 (Aug. 2016), 33 pages. doi:10.1145/2911988

[28] Hao Cui, Rahmadi Trimananda, Athina Markopoulou, and Scott Jordan. 2023. PoliGraph: Automated Privacy Policy Analysis using Knowledge Graphs. In *32nd USENIX Security Symposium (USENIX Security 23)*. USENIX Association, Anaheim, CA, 1037–1054. https://www.usenix.org/conference/usenixsecurity23/presentation/cui

[29] 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 Real-time Privacy Monitoring on Smartphones. *Association for Computing Machinery Transactions Computer System* 32, 2, Article 5 (June 2014), 29 pages. doi:10.1145/2619091

[30] Toyin Falola. 1993. My Friend the Shylock: Money-Lenders and their Clients in South-Western Nigeria. *The Journal of African History* 34, 3 (1993), i–xiii. https://api.semanticscholar.org/CorpusID:163147969

[31] Ming Fan, Le Yu, Sen Chen, Hao Zhou, Xiapu Luo, Shuyue Li, Yang Liu, Jun Liu, and Ting Liu. 2020. An Empirical Evaluation of GDPR Compliance Violations in Android mHealth Apps. arXiv:2008.05864 [cs.SE] https://arxiv.org/abs/2008.05864

[32] Federal Competition and Consumer Protection Commission. 2023. Guidelines for Digital Money Lenders. https://fccpc.gov.ng/limited-interim-regulatory-registration-framework-and-guidelines-for-digital-lending-2022/.

[33] Federal Competition and Consumer Protection Commission. 2024. Enforcement against Digital Lending violations. https://fccpc.gov.ng/enforcement-against-digital-lending-violations/.

[34] Adrienne Porter Felt, Erika Chin, Steve Hanna, Dawn Song, and David Wagner. 2011. Android permissions demystified. In *Proceedings of the 18th Association for Computing Machinery Conference on Computer and Communications Security* (Chicago, Illinois, USA). Association for Computing Machinery, New York, NY, USA, 627–638. doi:10.1145/2046707.2046779

[35] Financial Services Authority of Indonesia. 2016. *The Regulation of the Financial Services Authority Number 77/POJK.01/2016 regarding Information Technology-Based Borrowing-Lending Services*. Financial Services Authority of Indonesia. https://makna.co/wp-content/uploads/2018/01/OJK-Regulation-No-77-of-2016-Makna-Eng.pdf.

[36] Vincent Freiberger, Arthur Fleig, and Erik Buchmann. 2025. "You Don't Need a University Degree to Comprehend Data Protection This Way": LLM-Powered Interactive Privacy Policy Assessment. In *Proceedings of the Extended Abstracts of the CHI Conference on Human Factors in Computing Systems (CHI EA '25)*. Association for Computing Machinery, New York, NY, USA, Article 36, 12 pages. doi:10.1145/3706599.3719816

[37] Clint Gibler, Jonathan Crussell, Jeremy Erickson, and Hao Chen. 2012. AndroidLeaks: automatically detecting potential privacy leaks in android applications on a large scale. In *Proceedings of the 5th International Conference on Trust and Trustworthy Computing* (Vienna, Austria) *(TRUST'12)*. Springer-Verlag, Berlin, Heidelberg, 291–307. doi:10.1007/978-

3-642-30921-2_17
[38] Google Play Console. 2023. Financial Services Personal loans. https://support.google.com/googleplay/android-developer/answer/9876821?hl=en.
[39] HDB Financial Services Limited. 2023. WhatsApp Loan Scam Alert – Protect Yourself with these Tips. https://www.hdbfs.com/blogs/whatsapp-loan-scam-alert-protect-yourself-with-these-tips.
[40] Asma Khatoon and Peter M. Corcoran. 2017. Android permission system and user privacy — A review of concept and approaches. *IEEE 7th International Conference on Consumer Electronics - Berlin* (2017), 153–158. https://api.semanticscholar.org/CorpusID:3860605
[41] William Koch, Abdelberi Chaabane, Manuel Egele, William Robertson, and Engin Kirda. 2017. Semi-automated discovery of server-based information oversharing vulnerabilities in Android applications. In *Proceedings of the 26th ACM SIGSOFT International Symposium on Software Testing and Analysis* (Santa Barbara, CA, USA) *(ISSTA 2017)*. Association for Computing Machinery, New York, NY, USA, 147–157. https://doi.org/10.1145/3092703.3092708
[42] Chaoran Li, Xiao Chen, Ruoxi Sun, Minhui Xue, Sheng Wen, Muhammad Ejaz Ahmed, Seyit Camtepe, and Yang Xiang. 2022. Cross-language Android permission specification. In *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering* (Singapore, Singapore) *(ESEC/FSE 2022)*. Association for Computing Machinery, New York, NY, USA, 772–783. doi:10.1145/3540250.3549142
[43] Martina Lindorfer, Matthias Neugschwandtner, Lukas Weichselbaum, Yanick Fratantonio, Victor van der Veen, and Christian Platzer. 2014. ANDRUBIS – 1,000,000 Apps Later: A View on Current Android Malware Behaviors. In *Proceedings of the 2014 Third International Workshop on Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS '14)*. IEEE Computer Society, USA, 3–17. doi:10.1109/BADGERS.2014.7
[44] Minxing Liu, Haoyu Wang, Yao Guo, and Jason Hong. 2016. Identifying and Analyzing the Privacy of Apps for Kids. In *Proceedings of the 17th International Workshop on Mobile Computing Systems and Applications* (St. Augustine, Florida, USA) *(HotMobile '16)*. Association for Computing Machinery, New York, NY, USA, 105–110. doi:10.1145/2873587.2873597
[45] Livemint. 2023. Morphed Photos of Wife Circulated by Loan App Agents, Andhra Man Ends Life: Here's What Happened. https://www.livemint.com/news/india/morphed-photos-of-wife-ciruclated-by-loan-app-agents-andhra-man-ends-life-heres-what-happened-11733913896364.html.
[46] Samin Yaseer Mahmud, Akhil Acharya, Benjamin Andow, William Enck, and Bradley Reaves. 2020. Cardpliance: PCI DSS Compliance of Android Applications. In *29th USENIX Security Symposium (USENIX Security 20)*. USENIX Association, Boston, 1517–1533. https://www.usenix.org/conference/usenixsecurity20/presentation/mahmud
[47] Lisa Mekioussa Malki, Ina Kaleva, Dilisha Patel, Mark Warner, and Ruba Abu-Salma. 2024. Exploring Privacy Practices of Female mHealth Apps in a Post-Roe World. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '24)*. Association for Computing Machinery, New York, NY, USA, Article 576, 24 pages. https://doi.org/10.1145/3613904.3642521
[48] marshalwahlexyz1. 2025. The Cost of Convenience: Identifying, Analyzing, and Mitigating Predatory Loan Applications on Android (Code Repository). https://github.com/marshalwahlexyz1/-The-Cost-of-Convenience-Identifying-Analyzing-and-Mitigating-Predatory-Loan-Applications-on-Android. Accessed: 2025-12-10.
[49] Ryan McConkey and Oluwafemi Olukoya. 2024. Runtime and Design Time Completeness Checking of Dangerous Android App Permissions Against GDPR. *IEEE Access* 12 (2024), 1–22. doi:10.1109/ACCESS.2023.3347194
[50] Abraham Mhaidli, Selin Fidan, An Doan, Gina Herakovic, Mukund Srinath, Lee Matheson, Shomir Wilson, and Florian Schaub. 2023. Researchers' Experiences in Analyzing Privacy Policies: Challenges and Opportunities. *Proceedings on Privacy Enhancing Technologies* 2023, 4 (2023), 287–305. doi:10.56553/popets-2023-0111
[51] Keika Mori, Daiki Ito, Takumi Fukunaga, Takuya Watanabe, Yuta Takata, Masaki Kamizono, and Tatsuya Mori. 2025. Evaluating LLMs Towards Automated Assessment of Privacy Policy Understandability. (01 2025). doi:10.14722/usec.2025.23009
[52] Collins W. Munyendo, Yasemin Acar, and Adam J. Aviv. 2022. "Desperate Times Call for Desperate Measures": User Concerns with Mobile Loan Apps in Kenya. (2022), 2304–2319. doi:10.1109/SP46214.2022.9833779
[53] Suraj Naik. 2023. How to Spot and Avoid WhatsApp Loan Scams: Essential Tips To Protect Yourself. https://www.cashe.co.in/our-blog/how-to-spot-and-avoid-whatsapp-loan-scams/.
[54] National Privacy Commission Republic of the Philippines. 2020. Guidelines on the Processing of Personal Data for Loan-Related TransactionS. https://www.privacy.gov.ph/wp-content/uploads/2020/10/NPC-Circular-No.-20-01.pdf.
[55] Linus Ndruru, Carolus Herman, Deny Ttistian, and Sigit Widodo. 2023. Law Enforcement on Misuse of Personal Data by Online Loan Business Actors. *Indonesian Journal of Law and Islamic Law (IJLIL)* 5 (12 2023), 40–49. doi:10.35719/ijlil.v5i2.317
[56] Shradha Neupane, Faiza Tazi, Upakar Paudel, Freddy Veloz Baez, Merzia Adamjee, Lorenzo De Carli, Sanchari Das, and Indrakshi Ray. 2022. On the Data Privacy, Security, and Risk Postures of IoT Mobile Companion Apps. In *Data and Applications Security and Privacy XXXVI: 36th Annual IFIP WG 11.3 Conference, DBSec 2022, Newark, NJ, USA, July 18–20, 2022, Proceedings* (Newark, NJ, USA). Springer-Verlag, Berlin, Heidelberg, 162–182. doi:10.1007/978-3-031-10684-2_10
[57] Trung Tin Nguyen, Michael Backes, Ninja Marnau, and Ben Stock. 2021. Share First, Ask Later (or Never?) Studying Violations of GDPR's Explicit Consent in Android Apps. In *30th USENIX Security Symposium (USENIX Security 21)*. USENIX Association, Vancouver, British Columbia, Canada, 3667–3684. https://www.usenix.org/conference/usenixsecurity21/presentation/nguyen
[58] Nigerian Bureau of Statistics. 2022. Nigeria Launches its Most Extensive National Measure of Multidimensional Poverty. https://nigerianstat.gov.ng.
[59] Olatunji Olaigbe. 2022. Loan apps ruined their reputations. A shady online market offered to repair them. https://restofworld.org/2022/nigeria-loan-apps-reputation/.
[60] Oleavr. 2014. Frida binary instrumentation toolkit. https://frida.re.
[61] Oluwakemi Abimbola. 2022. Loan shark declares defaulters dead, distributes obituaries to relatives, friends. https://punchng.com/loan-shark-declares-defaulters-dead-distributes-obituaries-to-relatives-friends/.
[62] 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 Association, Santa Clara, CA, 603–620. https://www.usenix.org/conference/usenixsecurity19/presentation/reardon
[63] Bradley Reaves, Jasmine Bowers, Nolen Scaife, Adam Bates, Arnav Bhartiya, Patrick Traynor, and Kevin R. B. Butler. 2017. Mo(bile) Money, Mo(bile) Problems: Analysis of Branchless Banking Applications. *ACM Trans. Priv. Secur.* 20, 3, Article 11 (Aug. 2017), 31 pages. doi:10.1145/3092368
[64] Jingjing Ren, Ashwin Rao, Martina Lindorfer, Arnaud Legout, and David Choffnes. 2016. ReCon: Revealing and Controlling PII Leaks in Mobile Network Traffic. In *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services* (Singapore, Singapore) *(MobiSys '16)*. Association for Computing Machinery, New York, NY, USA, 361–374. doi:10.1145/2906388.2906392
[65] Reserve Bank of India. 2022. *Guidelines on Digital Lending*. Reserve Bank of India. https://www.rbi.org.in/Scripts/NotificationUser.aspx?Id=12382&Mode=0.
[66] Mora Saritha. 2023. DEMYSTIFYING THE MISERY BEHIND LOAN APPS IN INDIA. *Indian Journal of Finance and Banking* 13, 1 (Feb. 2023), 104–109. doi:10.46281/ijfb.v13i1.2047
[67] Christian Schindler, Muslum Atas, Thomas Strametz, Johannes Feiner, and Reinhard Hofer. 2022. Privacy Leak Identification in Third-Party Android Libraries. In *2022 Seventh International Conference On Mobile And Secure Services (MobiSecServ)*. IEEE, Gainesville, FL, USA, 1–6. doi:10.1109/MobiSecServ50855.2022.9727217
[68] Securities and Exchange of Pakistan. 2022. *Requirements for NBFCs engaged in digital lending*. Securities and Exchange of Pakistan. https://www.secp.gov.pk/document/circular-no-15-of-2022-requirements-for-nbfcs-engaged-in-digital-lending/?ind=1672222021650&filename=Circular-No.15-of-2022..pdf&wpdmdl=46436&refresh=63ac43d13db561672233937.
[69] Rocky Slavin, Xiaoyin Wang, Mitra Bokaei Hosseini, James Hester, Ram Krishnan, Jaspreet Bhatia, Travis D. Breaux, and Jianwei Niu. 2016. PVDetector: a detector of privacy-policy violations for Android apps. In *Proceedings of the International Conference on Mobile Software Engineering and Systems* (Austin, Texas) *(MOBILESoft '16)*. Association for Computing Machinery, Austin Texas, 299–300. doi:10.1145/2897073.2897720
[70] StatCounter Global Stats. 2024. Mobile Operating System Market Share in Africa. https://gs.statcounter.com/os-market-share/mobile/africa, Data period: March 2024–March 2025.
[71] StatCounter Global Stats. 2024. Mobile Operating System Market Share in Asia. https://gs.statcounter.com/os-market-share/mobile/asia.
[72] Indrajani Sutedja, Muhammad Firdaus Adam, Fauzan Hafizh, and Muhammad Farrel Wahyudi. 2024. An Analysis of the Effect of Using Online Loans

on User Data Privacy. *International Journal of Advanced Computer Science and Applications* 15, 7 (2024), 6. doi:10.14569/IJACSA.2024.01507118

[73] Tomas Lujambio. 2023. SpyLoan Malware Targets Latin American Mobile Users. https://mexicobusiness.news/cybersecurity/news/spyloan-malware-targets-latin-american-mobile-users?tag=fabio-assolini.

[74] David Rodríguez Torrado, Ian Yang, José M. del Álamo, and Norman Sadeh. 2024. Large language models: a new approach for privacy policy analysis at scale. *Computing* 106 (2024), 3879 – 3903. https://api.semanticscholar.org/CorpusID:270199788

[75] UNICEF. 2023. Nigeria's Population. https://www.unicef.org/media/152961/file/Nigeria-2023-COAR.pdf.

[76] Wara Irfan and Mutaher Khan. 2023. Between digital loan sharks and weak regulation, Pakistanis are stuck between a rock and a hard place. https://www.dawn.com/news/1767080.

[77] Anhao Xiang, Weiping Pei, and Chuan Yue. 2023. PolicyChecker: Analyzing the GDPR Completeness of Mobile Apps' Privacy Policies. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security* (Copenhagen, Denmark) *(CCS '23)*. Association for Computing Machinery, New York, NY, USA, 3373–3387. doi:10.1145/3576915.3623067

[78] Yuxia Zhan, Yan Meng, Lu Zhou, Yichang Xiong, Xiaokuan Zhang, Lichuan Ma, Guoxing Chen, Qingqi Pei, and Haojin Zhu. 2024. VPVet: Vetting Privacy Policies of Virtual Reality Apps. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security* (Salt Lake City, UT, USA) *(CCS '24)*. Association for Computing Machinery, New York, NY, USA, 1746–1760. https://doi.org/10.1145/3658644.3690321

[79] Sebastian Zimmeck, Peter Story, Daniel Smullen, Abhilasha Ravichander, Ziqi Wang, Joel Reidenberg, N. Russell, and Norman Sadeh. 2019. MAPS: Scaling Privacy Compliance Analysis to a Million Apps. *Proceedings on Privacy Enhancing Technologies* 2019 (07 2019), 66–86. doi:10.2478/popets-2019-0037

[80] Sebastian Zimmeck, Ziqi Wang, Lieyong Zou, Roger Iyengar, Bin Liu, Florian Schaub, Shomir Wilson, Norman M. Sadeh, Steven M. Bellovin, and Joel R. Reidenberg. 2016. Automated Analysis of Privacy Requirements for Mobile Apps. In *Network and Distributed System Security Symposium*. NDSS, San Diego, California, USA. https://api.semanticscholar.org/CorpusID:655548

A Prompt Template for Policy-to-Permission Mapping

This appendix outlines the prompts used in our automated policy analysis system to map regulatory policies to Android permissions.

A.1 System Prompt

The system is initialized with the following role definition:

> *You are an Android security and data privacy expert with extensive knowledge of Android permissions and regulatory policies. Your expertise includes interpreting natural language policy statements and mapping them to technical permissions on the Android platform.*

A.2 User Prompt Template

For each policy document, the following structured prompt is used:

Task: Analyze the following policy document and identify data access restrictions for mobile applications.
Policy Document: *[POLICY_TEXT_PLACEHOLDER]*
Instructions:

(1) **Identify prohibited data types:**
- Unconditionally prohibited: Data types explicitly banned regardless of user consent
- Conditionally prohibited: Data types that require explicit user consent

(2) **Map to Android permissions:** For each identified data type, determine the corresponding Android permissions required to access that data.

(3) **Reasoning process:**
   a. Extract all statements relating to data access prohibitions
   b. Classify each as unconditional or conditional based on consent requirements
   c. Map prohibited data types to specific Android permissions
   d. Explain reasoning for uncertain mappings

Output Format: For each identified restriction, provide: Data Type (as described in policy), Prohibition Type (unconditional/-conditional), Android Permissions (specific permission names), and Notes (any ambiguities or uncertainties).

A.3 Example Policy Input

The following excerpt from India's RBI guidelines demonstrates the input format:

> **Chapter IV: Technology and Data Requirement**
> **12. Collection, usage and sharing of data with third parties**
> **i.** RE shall ensure that any collection of data by their DLA and DLA of their LSP is need-based and with prior and explicit consent of the borrower having audit trail. In any case, RE shall also ensure that DLA of RE/LSP desist from accessing mobile phone resources like file and media, contact list, call logs, telephony functions, etc.
> *[Additional clauses omitted for brevity...]*

A.4 Sample Model Response

The GPT-4 model processes the policy through a systematic extraction and classification approach:

Extracted Statements:
- From 12.i: "DLA … desist from accessing mobile phone resources like **file and media, contact list, call logs, telephony functions**, etc."
- From 12.i: "A one-time access can be taken for **camera, microphone, location** … only, with explicit consent."
- From 13.iii: "RE shall ensure that **no biometric data is stored/collected** …"

Classification:
- **Unconditional:** file & media, contacts, call logs, telephony, biometric data
- **Conditional:** camera, microphone, location (one-time with consent)

Table 3 shows the complete mapping from data types to Android permissions generated by the model.

Table 3: Data Type to Android Permission Mapping (GPT-4 Response)

Data Type	Prohibition Type	Android Permission(s)
File & media	Unconditional	READ_EXTERNAL_STORAGE, WRITE_EXTERNAL_STORAGE, READ_MEDIA_IMAGES, READ_MEDIA_VIDEO, READ_MEDIA_AUDIO, MANAGE_EXTERNAL_STORAGE
Contact list	Unconditional	READ_CONTACTS, WRITE_CONTACTS, GET_ACCOUNTS
Call logs	Unconditional	READ_CALL_LOG, WRITE_CALL_LOG, PROCESS_OUTGOING_CALLS
Telephony functions	Unconditional	READ_PHONE_STATE, CALL_PHONE, ANSWER_PHONE_CALLS, ADD_VOICEMAIL, USE_SIP
Biometric data	Unconditional	USE_BIOMETRIC, USE_FINGERPRINT
Camera	Conditional (one-time)	CAMERA
Microphone	Conditional (one-time)	RECORD_AUDIO
Location	Conditional (one-time)	ACCESS_FINE_LOCATION, ACCESS_COARSE_LOCATION, ACCESS_BACKGROUND_LOCATION