# A Study of Privacy-Related Data Collected by Android Apps

Mugdha Khedkar [1*], Ambuj Kumar Mondal [2] and Eric Bodden [1,3]

[1]Heinz Nixdorf Institute, Paderborn University, Paderborn, Germany.
[2] Paderborn University, Paderborn, Germany.
[3]Fraunhofer IEM, Paderborn, Germany.

*Corresponding author(s). E-mail(s): mugdha.khedkar@uni-paderborn.de;
Contributing authors: ambuj.mondal@gmail.com; eric.bodden@uni-paderborn.de;

**Abstract**

Many Android apps collect data from users, and the European Union's General Data Protection Regulation (GDPR) mandates clear disclosures of such data collection. However, apps often use third-party code, complicating accurate disclosures. This paper investigates how accurately current Android apps fulfill these requirements.

In this work, we present a multi-layered definition of privacy-related data to correctly report data collection in Android apps. We further create a dataset of privacy-sensitive data classes that may be used as input by an Android app. This dataset takes into account data collected both through the user interface and system APIs. Based on this, we implement a semi-automated prototype that detects and labels privacy-related data collected by a given Android app.

We manually examine the data safety sections of 70 Android apps to observe how data collection is reported, identifying instances of over- and under-reporting. We compare our prototype's results with the data safety sections of 20 apps revealing reporting discrepancies. Using the results from two Messaging and Social Media apps (Signal and Instagram), we discuss how app developers under-report and over-report data collection, respectively, and identify inaccurately reported data categories.

A broader study of 7,500 Android apps reveals that apps most frequently collect data that can *partially identify* users. Although system APIs consistently collect large amounts of privacy-related data, user interfaces exhibit some more diverse data collection patterns. A more focused study on various domains of apps reveals that the largest fraction of apps collecting personal data belong to the domain of *Messaging and Social Media*. Our findings show that location is collected frequently by apps, specially from the *E-commerce and Shopping* domain. However, it is often under-reported in app data safety sections. Our results highlight the need for greater consistency in privacy-aware app development and reporting practices.

**Keywords:** static analysis, data collection, data protection, privacy-aware reporting

## 1 Introduction

We use many Android applications in our daily life. The number of apps available in the Google Play Store was most recently placed at 2.43 million, after surpassing 1 million apps in July 2013 [1]. Among these apps, almost all collect data from their users, and some apps belong to particularly privacy-sensitive categories.

Since 2018, all Android apps that collect data from users residing in the European Union must comply with the General Data Protection Regulation [2]. The GDPR aims to protect users' *personal data*, which is defined as *"any information relating to an identified or identifiable natural person"* where an identifiable natural person is one who can be identified, directly or indirectly, by reference to an identifier such as a name, an identification number, etc. The GDPR imposes several obligations on the collection, storage, and processing of personal data. Article 25 of the GDPR highlights *data protection by default and design*, making data protection an integral part of the application development process.

The growing demand for privacy by design [3], both by end users and by GDPR necessitates that app developers use state-of-the-art technical measures to protect their users' privacy. The legal description of GDPR is very complex and lengthy and hence it can be difficult for app developers to understand. Since they lack legal expertise, app developers may find it difficult to understand how to protect users' data, or even *which* data they should protect [4].

Although many Android applications state a privacy policy, violations of privacy policies can easily happen inadvertently, in the worst case just by using data-hungry third-party libraries [5]. Moreover, several studies [6–9] have consistently shown that privacy policies diverge significantly from the source code. Google Play recently introduced the data safety section, shifting the responsibility of privacy-related reporting to app developers [10]. They must complete the data safety form, detailing how apps collect, share, and secure users' data. A recent study by Mozilla [11] has revealed discrepancies between the information reported in data safety sections and privacy policies of Android apps. Thus, it appears that neither documentation provides a usable ground truth in case one wants to study the data collected by Android apps. There has been limited research comparing the data safety section with the source code of Android apps, and we aim to fill that gap.

**Our work.** GDPR's definition of *personal data* brings about a paradigm shift in data protection: apart from protecting data that can cause harm if it is *leaked*, the app must also protect data that can *identify* an individual, either directly or indirectly. Our work implements this paradigm shift by centering its data classification around GDPR's definition of *personal data* enabling app developers to conduct a precise risk analysis of the collected user data. This is then used to construct a dataset that classifies possible inputs to an Android app using categories similar to those in the data safety section. This dataset considers data collected both through the user interface and system APIs, suitable for further static analyses such as taint analysis.

We also develop a prototype that statically extracts and labels privacy-related data collected via app source code, user interfaces, and permissions. We make all artifacts available at https://zenodo.org/records/14046829 and hope that the results make users aware of the enormous amount of data their favorite Android apps collect from them.

This paper extends our prior work [12] by conducting a large-scale study of Android apps, providing a more comprehensive overview of the data collected by these apps. While our previous work focused on understanding how data is reported in the data safety sections, the current study expands on this by identifying which types of data are frequently collected across different app domains. This approach helps pinpoint areas where reporting efforts need to be strengthened.

To this end, we conduct two experiments: (1) a small-scale analysis to examine which data categories are consistently reported in the data safety sections of Android apps, and (2) a large-scale study to observe which categories of privacy-related data are most frequently collected by Android apps. The first experiment has been published in our previous work [12].

We use our prototype to statically extract and label privacy-related data collected via app source code, user interfaces, and permissions. Comparing the prototype's results with the data safety sections of 20 apps reveals reporting discrepancies. Using the results from two Messaging and Social Media apps (Signal and Instagram), we discuss how app developers under-report and over-report data collection, respectively, and identify inaccurately reported data categories.

Next, we conduct a quantitative study on 7,500 Android apps, and find that Android apps predominantly collect personal and financial information through user interfaces, and device, location, and browsing information through system APIs. This study is a unique contribution of this paper. Our observations highlight that personal data that can *partially identify* a user, like a pin code or age, is the most frequently collected data. Notably, while system APIs consistently collect large amounts of privacy-related data, user interfaces exhibit some interesting data collection patterns. To examine this observation, we conduct a qualitative analysis of the 100 most popular apps each from the domains: *Messaging and Social Media, Banking and Finance, News and Entertainment, Sports and Games, Technology and Education, E-Commerce and Shopping, and Health and Fitness.* We find that *Messaging and Social Media* apps frequently collect personal data and data required for authentication directly from the user. On the other hand, user interfaces of *Sports and Games* apps collect the least amount of personal data. Additionally, this analysis identifies the most frequently collected partial identifier combinations for each domain. It also reveals that *E-Commerce and Shopping* apps collect the highest percentage of partial identifier combinations.

**Contributions.** To summarize, this work makes the following contributions:

- We propose a multi-layered definition of privacy-related data. This is then used to construct a dataset that classifies possible inputs to an Android app. This dataset considers data collected both through the user interface and system APIs and it can be used for further analysis.
- We develop a prototype that statically detects privacy-related data collected by an Android app and reports the results.
- We compare the data collected by the app code with the reported data in the data safety section, and reveal discrepancies.
- We use our prototype to study privacy-related user input collection across several domains on Android apps.

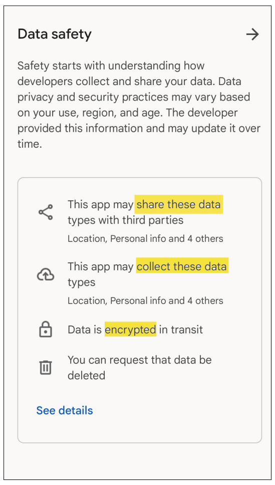
**Roadmap.** The remainder of the paper is organized as follows. Section 2 introduces some background about the GDPR, and data collection in Android apps, and also discusses existing research on the topic. Section 3 presents our definition of privacy-related data. Section 4 discusses how we detect and label the privacy-related data collected by Android apps. Section 5 explains our experiments, and Section 6 discusses our findings in detail. Section 7 explains the threats to validity, and Section 8 concludes the paper.

## 2 Background and Related Work

This section introduces the concepts we will be discussing in this paper, and discusses existing research on the topic.

**GDPR and Reporting of Data**. On 25 May 2018, the European Union enforced the General Data Protection Regulation (GDPR) which has been called the *toughest privacy and security law in the world* [13]. The GDPR introduces a strict definition of *personal data* and imposes obligations on the collection, storage, and processing of personal data.

Article 5 of the GDPR highlights the importance of data minimization, which means collecting only the personal data necessary for specific purposes. Once personal data is collected, the GDPR requires app developers to implement technical and organizational measures to protect it. These measures include effectively disguising personal data using a *pseudonymization* function. Pseudonymization aims to protect personal data by replacing parts of it with unique, non-identifying *pseudonyms*. Personal data can then no longer be attributed to a specific data subject without the use of pseudonyms [14]. Even though pseudonymized data still qualifies as personal data, pseudonymization techniques mitigate data protection risks. By pseudonymizing personal data, organizations may continue to use the data for their purposes while reducing the risk of severe consequences in case of a privacy breach [15].



Data safety

Safety starts with understanding how developers collect and share your data. Data privacy and security practices may vary based on your use, region, and age. The developer provided this information and may update it over time.

This app may share these data types with third parties

Location, Personal info and 4 others

This app may collect these data types

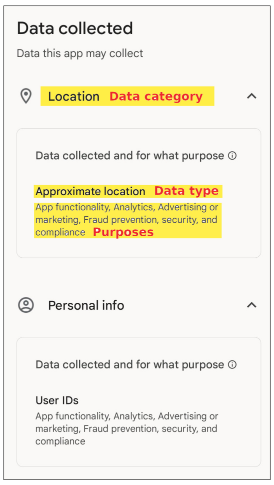
Location, Personal info and 4 others

Data is encrypted in transit

You can request that data be deleted

See details

(a) Data safety section.



Data collected

Data this app may collect

Location **Data category**

Data collected and for what purpose

Approximate location **Data type**

App functionality, Analytics, Advertising or marketing, Fraud prevention, security, and compliance **Purposes**

Personal info

Data collected and for what purpose

User IDs

App functionality, Analytics, Advertising or marketing, Fraud prevention, security, and compliance

(b) Classification of data.

**Fig. 1**: An example of the data safety section of an Android app.

Furthermore, Article 13 of the GDPR mandates that personal data collection and processing be reported to the data subject through documents such as privacy policies. However, long and convoluted documents may be difficult to understand.

In 2009, researchers suggested the use of privacy nutrition labels [16, 17] to encourage transparency. Following this, the Apple App Store introduced their labels in 2020 [18]. In 2022, researchers introduced a tool [19] that helps iOS developers generate privacy labels by identifying data flows through code analysis.

In 2022, Google Play introduced their labels by launching the data safety section [10], requiring app developers to disclose how their apps collect, share, and protect user data. The data safety form provided by Google comprises three sections: *data sharing, data collection, and security practices* (cf. Figure 1a) and is a concise representation of privacy-related information. Within this form, user data is classified into different *data categories*, which are further divided into specific *data types* (cf. Figure 1b). This data can be collected or shared for different *purposes* (chosen from a fixed set defined by Google). App developers are also required to share if the app uses data protection measures such as encryption or on-demand data deletion.

A recent study by Mozilla [11] evaluated popular Android apps by grading them based on the presence or absence of discrepancies between their data safety sections and privacy policies. Khandelwal et al. [20] observed the evolution of the data safety section and highlighted under-reporting, over-reporting and inconsistencies in data practices. They examined how app developers interact with the data safety sections, using a dataset of snapshots without analyzing user interfaces or source code.

In 2022, Google introduced Checks [21], a paid service that assists app developers in completing the data safety section. However, its cost could hinder many app developers from using its services. Recently, open-source alternatives like Matcha [22] and

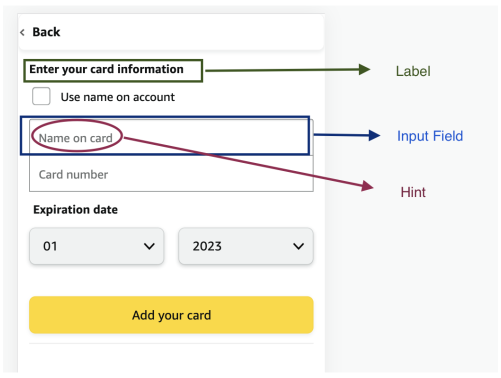


**Fig. 2**: Example: Input fields with labels and Edit Texts.

Privado.ai [23] have been introduced to assist developers in creating accurate Google Play data safety labels.

**Data Collection in Android apps.** An Android app may acquire privacy-related data from the user either directly or indirectly. The most direct way to gain access to data is to ask the user to provide it via various input screens and user interfaces.

Android uses XML (eXtensible Markup Language) to describe the user interface layout of an application. The XML layout files play a crucial role in defining the buttons, text fields, and layouts that constitute the application's user interface. A layout XML file defines the structure and visual appearance of a screen and defines the position, appearance, and characteristics of its components and how they interact with each other. *EditText* is a widely used input field that allows users to enter text-based data. It supports various input types, including but not limited to plain text, numbers, email addresses, and passwords.

Figure 2 illustrates a screen from Amazon's shopping app that collects credit card information from the user. It consists of *editText* fields for capturing the name and number of the credit card, and dropdown fields to select the expiration date. Attributes like *android:inputType* define the allowed input type (e.g., text, number, password), while *android:hint* provides a placeholder or instructional text. These attributes describe the type of data to be captured. In Figure 2, using the label and hint texts, one can guess that the input field is asking the user to input credit card details.

Since system calls like *Element.getText()* return the values assigned to these attributes (IDs, hints, labels), static analysis tools [24, 25] classify them as sensitive sources. Hence, these tools effectively end up tainting all user inputs that are returned by such sources. However, this approach may lack precision, as not all user inputs are equally sensitive, and some may not be sensitive at all. Since there are no specific guidelines in API design to regulate this data, it becomes challenging to monitor and identify whether the data actually is privacy-related. User interfaces are a common means of directly collecting data from the user, emphasizing the need for a more accurate risk analysis of such user input. Several studies analyze the text in user interfaces to detect privacy policy violations and malicious activities [8, 26, 27]. SUPOR [28] and

UIPicker [29] can identify sensitive data collected through user interfaces of Android apps. Unfortunately, these are in-house proprietary tools and do not focus on GDPR's definition of personal data. UiRef [30] analyzes user interfaces to detect sensitive inputs and shows promising results, but its algorithm needs adaptation to the post-GDPR definition of privacy-related data.

An Android app may also acquire data from the operating system through system API calls. Some of this data might be privacy-related, such as the GPS information collected through system API calls like *getLastKnownLocation()*. Many previous studies on taint analysis [25, 31, 32] have used predefined sets of methods which are potential sources of system API privacy-related data. Follow-up work then proposed machine-learning approaches for automatically classifying and categorizing methods into sources [24, 33–35]. In 2023, Kober et al. [36] provided a sound definition of sensitive data derived from the definition of personal data of several legal frameworks (including GDPR). They further publicly provided a list of sensitive sources from the Android framework which we used in our work.

To be able to access certain system APIs in the first place, Android apps may request appropriate permissions from the user at install time or runtime. They are declared in the app's *AndroidManifest.xml* file and relate to various system permissions to read, write or administer the resources. These permissions can be set and reset using the Android settings for the app.

For the remainder of this paper, we adopt a conservative definition of collected data. This includes data collected via user interface screens, via system API calls in the source code, or as permissions that Android developer's guide directly maps to specific collected data types in the data safety section [37].

Although related research exists, we did not find approaches that systematically identify and label all collected data as privacy-related categories, allowing direct comparison with the data safety section.

## 3 Privacy-related data

The terms "private" and "sensitive" are subjective, lacking a precise definition of which data is considered sensitive within the context of an Android app [36]. In legal contexts, GDPR's definition of personal data is directly derived from personal identifiers, which are clearly defined by the Article 29 Working Party [38]. These identifiers consist of information that holds a particularly privileged and close relationship with an individual, allowing for identification [14]. While certain identifiers are sufficient to identify an individual only in some processing contexts, the others may always identify the user. Partial identifiers may only identify users when combined with additional information [14]. However, until Kober et al. [36] provided a sound definition of sensitive data derived from the definition of personal data of several legal frameworks (including GDPR), there was no such definition for a technical framework such as Android. We build on their assessment and expand on their definition of sensitive data, considering the following categories of privacy-related data collected by an Android app:

**Table 1**: Classification of privacy-related data.

| **Privacy-Relevance** | **Data Categories** | **Utility** | **Risk Rank** |
|---|---|---|---|
| Directly identifiable personal data | Personal information, Device or other IDs, Financial information | Identification of user | 1 |
| Partially identifiable personal data | Personal information, Location data, Device data, Audio data, Browsing data, App activity, Photos and videos, Session data, Calendar data | Identification of user possible in combination with other data | 2 |
| Access data | Authentication, Email authentication, Network authentication, Payment authentication | Authentication | 3 |
| Context-dependent data | Message, UI, Audio, Photos and videos, Email | Unknown statically | 4 |

1. *Directly identifiable personal data*: This refers to personal data that can uniquely identify an individual or a device without the use of any additional information. Examples include an email address, a passport number, a phone number, IP address. Such data consists of a single piece of information that is sufficient for identification, and hence it is crucial to pseudonymize it before processing.
2. *Partially identifiable personal data*: This type of personal data can identify an individual or a device when combined with additional information. Browsing histories are one example, as they could be linked to some social network profiles such as Twitter or Facebook accounts [39]. Kurtz et al. [40] show that users of iOS devices could be singled out through their personalized device configurations, despite the absence of any device identifiers. Hence, this data should be pseudonymized before processing. Other examples include a pin code, postal address, age, and gender.
3. *Access data*: This category includes data that can grant access to an individual's device but cannot identify an individual's device. For example, a password, a card verification value (CVV), and a one-time password (OTP). While it is not categorized as personal data, it still needs protection since it can cause harm if leaked.
4. *Context-dependent data*: This type of data may identify an individual or a device depending on its content and the context in which it is used. Protection is necessary, but its exact identifiability cannot be determined statically. For example, the content of an email or a message may contain privacy-related information that is challenging to identify without running the app.

We summarize these data categories in Table 1, assigning a risk ranking to each category. A certain data can get only one risk rank. However, this risk-based classification is too coarse-grained to precisely report data collection in Android apps. To address this issue, we augment data with additional information. To ensure accurate labeling, these augmentations employ a detailed classification and are defined as a pair: Privacy Label → Identifier.

We label potential data sources (API calls or keywords) with a privacy label, which combines privacy relevance and data category (cf. Table 1). We introduced these data categories by carefully examining the categories in Google's data safety section (cf. Figure 1b). We also studied the sensitive Android API sources

**Table 2**: Examples of privacy-related data.

| **Data** | **Privacy label → Identifier** |
|---|---|
| Tax ID | Directly identifiable financial information → Unique ID |
| Family name | Partially identifiable personal information → Name |
| PIN | Access payment authentication data → Password |

labeled by SuSi [24] and then classified these data categories according to their privacy relevance. In some cases, we introduced some new data categories, such as partially identifiable device data. For every piece of data, we then decided its appropriate category.

The second component links the data source to an identifier, manually classified based on Android API documentation and Google's data safety section (cf. Figure 1b). Table 2 gives some concrete examples of privacy-related data labeled according to this classification.

Due to the subjectivity and ambiguity of data identifiers, this classification was done manually. Automating it will be an interesting but non-trivial task, and is beyond the scope of this paper.

## 4 Approach

After establishing a clear definition of privacy-related data, the next task is to detect if Android apps actually collect such data. To accomplish this, we first use the data definition presented in Section 3 to construct datasets that classify possible inputs to an Android app (cf. Section 4.1). We then implement a prototype to statically detect and label the collected data (cf. Section 4.2).

### 4.1 Dataset Construction

To label sources as potential sources of privacy-related data, we require a dataset that can be used as a classification criterion. We constructed two different datasets: an *identifier keywords dataset* for labeling UI data elements, and an *identifier API dataset* for labeling system API calls within the app code.

**Initial corpus.** We referred to Google Play Store and AppBrain statistics [41] to select the 50 most downloaded apps each from these categories: *Messaging and Social Media, Finance and Banking, News and Entertainment, Sports and Games, Technology and Education, E-Commerce and Shopping*, and *Health and Fitness*. We focused on these categories because they represent the major top-level categories of mobile apps and encompass a diverse range of data-collection behaviors. They include both high-risk categories (e.g., banking, health) and general-purpose categories (e.g., social media, entertainment), allowing us to capture representative patterns of identifiability across different types of applications. In July 2023, we used ApkMirror [42] to access these 350 apps crawled from the Google Play Store, and manually downloaded these 350 APKs.

We automatically extracted all the keywords from user input fields along with their metadata (editText id, input type, label, and hint) from the UI layout files of

**Table 3**: Third-Party Libraries in the Identifier API Dataset. Not an exhaustive list.

| **Category** | **Library** |
|---|---|
| Analytics | Google, Firebase |
| Authentication | Google, Firebase, Apache |
| Advertisements | Adjust, Applovin, AppsFlyer, Adcolony, Mopub |
| Image Processing | Google ZXing |
| Network | okHttp, Apache, gRPC, Volley |
| Email | Apache James |
| Location | Baidu |
| Phone number validation | Google i18n |

the selected 350 apps. To generate the initial dataset of system API methods, we extended SootFX [43] to detect potential data sources (getter methods) from the source code (APK). We first focused on commonly used Android and Java APIs, like *android.location, android.accounts* and *javax.net*. We then used the AppBrain statistics [41] to select the most installed third-party libraries across categories such as analytics and advertisements. Table 3 gives some examples of these libraries.

To distinguish first-party from third-party data collection, we analyze the package namespaces of all methods that access privacy-related data: methods belonging to the app's own package are labeled as first-party collection, while calls into known third-party SDK packages (e.g., those listed in Table 3 and sourced from AppBrain) are labeled as third-party collection. This separation follows the disclosure boundaries expected in Google's data safety section. We conservatively treat any collection operation performed within an SDK as third-party, regardless of DSS-specific exemptions, since developers remain responsible for declaring third-party data practices.

Finally, we combined the data sources extracted by SootFX with sources generated by SuSi [24] and the predefined sources used by FlowDroid [25] to produce the initial dataset of potential data-collection methods.

**Data annotation.** After constructing the initial corpus for our datasets, we filtered the metadata keywords and manually annotated them according to the data categories described in Section 3. The first and second authors independently performed an initial round of annotations. We then included feedback from a broader group of domain-adjacent researchers.

To do this, we selected a set of 50 keywords which the authors identified as potentially ambiguous and subjective to human judgement, and conducted a session with 22 research scholars from the Computer Science department at our university. We first introduced the data classification scheme and then asked participants to complete a survey evaluating these keywords along with their assigned categories. For each keyword, participants indicated whether they agreed with the proposed classification. Participants agreed unanimously on 45 of the 50 keywords. Two keywords (first name, family name) were reclassified from directly identifiable personal information to partially identifiable personal information based on the feedback. In addition, responses indicated the need for an additional category to capture data items that did not fit well into existing directly identifiable categories; we therefore introduced

**Table 4**: Part of the Identifier Keywords Dataset.

| **Keyword(s)** | **Privacy label → Identifier** |
|---|---|
| IBAN, Account number | Directly identifiable financial information → Account |
| First name, Family name | Partially identifiable personal information → Name |
| PIN, TAN | Access payment authentication data → Password |
| Chat | Context-dependent data → Message |

**Table 5**: Part of the Identifier API Dataset.

| **Method signature** | **Privacy label → Identifier** |
|---|---|
| *android.net.IpPrefix: java.net.InetAddress getAddress()* | Directly identifiable device or other IDs → IP Address |
| *android.location.Location: double getLatitude()* | Partially identifiable location data → Approximate location |
| *com.google.android.gms.auth.api.identity. SignInPassword: java.lang.String getPassword()* | Access email authentication data → Password |
| *android.app.Activity: android.view.View findViewById(int)* | Context-dependent UI data → Text field |

directly identifiable other data, which includes items such as signatures, biometric data, and images.

The final identifier keywords dataset was constructed by combining the updated annotations from this survey process. A part of the *identifier keywords dataset* is illustrated in Table 4.

The authors manually examined the official API documentation to annotate system API calls from the initial corpus, classifying them into the data categories described in Section 3. In cases where documentation was not available, we studied the labels SuSi [24] associated with these calls. This annotated information formed the basis for the *identifier API dataset*, as illustrated in Table 5.

We decided to annotate the datasets manually since it was non-trivial to automate this, especially for cases that required extensive discussion among the authors. For instance, we categorized IP addresses under directly identifiable device or other IDs rather than combining it with other device information and classifying it as partially identifiable device data. We carefully discussed ambiguous cases, which could have been classified under multiple categories. We classified postal address as partially identifiable personal information and the more general keywords such as zip code, city, or country as partially identifiable location data.

## 4.2 Detection of Sources

Our prototype (cf. Figure 3) accepts an APK as input and generates labeled privacy-related data sources found in the source code. We now discuss the modules needed to detect privacy-related data in the source code.

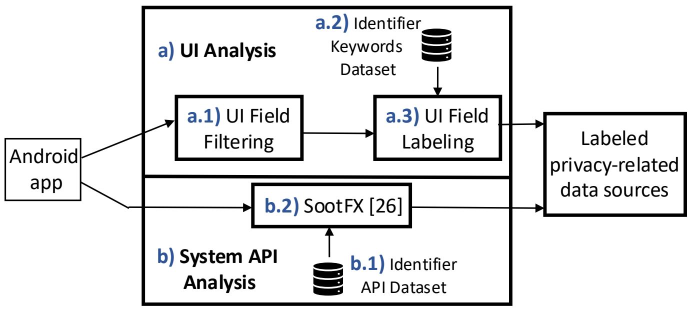


**Fig. 3**: Our prototype.

a) **UI Analysis.** Analyzing UI screens statically is challenging because the behavior of fields depends on human interaction. However, by exploring the attributes associated with these fields, which are defined in the layout XML file for that screen, we can deduce the fields and the data they capture to a certain extent.

We first decompile the APK file using *apktool* [44] and extract all XML files that present user screens. Next, we filter all user input fields and their associated attributes such as identifier, label, hint, and input type (a.1). Attributes like *android:inputType* define the allowed input type (e.g., text, number, password), while *android:hint* is an instructional text and suggests the format of data. Attributes like label give additional information about the data to be inserted. All *editText* fields have a unique identifier that is used to fetch the value and trigger events. It is reasonable to assume that this identifier contains text defining the data it collects (e.g., an identifier like txt_name suggests a field for collecting a name). Combining the identifier with the input type and hint provides a good starting point for heuristics to determine information sensitivity [28, 29].

Our field labeling analysis (a.3) starts by examining the type of the user input field. Input fields with types like *Password* are immediately classified as privacy-related information. If the input type does not reveal the nature of the information, we check the unique ID of the editText field and compare it with the identifier keywords dataset (a.2). In many cases, this unique ID contains important information about the data. In case we are still unable to decide the sensitivity, we examine the hint attribute and the text associated with the field. Combining these attributes gives a good idea about the input type. We then use the identifier dataset to assign a privacy label to the input field. Additionally, we disclose all the permissions declared in the app's *AndroidManifest.xml* file. The analysis is currently limited to textual data collected by an Android app. Identifying and classifying more complex data (such as images, audio, video, and biometrics) is out of the scope of the analysis.

b) **System API Analysis.** To analyze the privacy relevance of system API methods in an Android app, we extend SootFX [43], a static code-feature extraction tool for Java and Android. SootFX is a standalone tool that uses the Soot framework [45] to extract features like methods, classes, and the call graph of the target APK. SootFX allows the extraction of whole-program features such as the number of method calls or the number of reachable statements in a target program.

For our prototype, we introduce a new whole-program feature that uses our *identifier API dataset* (**b.1**) as input and checks the APK for privacy-sensitive system API calls. To this end, SootFX (**b.2**) uses the SPARK call graph construction algorithm [46] to extract the static call graph of the APK, and matches every method call with the methods described in the *identifier API dataset*.

If a match is found, it looks for the privacy label corresponding to the method call and labels it as a privacy-related data source.

## 5 Evaluation

We next use our prototype to automatically investigate how privacy-related data is collected and reported by Android apps. To this end, we conduct two experiments: (1) a small-scale analysis to examine which data categories are *most consistently reported* in the data safety sections of Android apps, and (2) a large-scale study to observe which categories of privacy-related data are *most frequently collected* by Android apps.

**Evaluation Setup.** To evaluate our prototype, we conducted experiments on a Linux virtual machine with an Intel(R) Xeon(R) Platinum 8462Y+ 8-core processor with 128 GB memory.

**Accuracy and Sanity Checks.** Before conducting our experiments, we performed a sanity check to assess the reliability of our prototype, since all subsequent analyses depend on its ability to identify privacy-related data. For user-interface analysis, our tool processes each app in an average of 12.18 seconds. We manually examined 25 randomly selected apps spanning multiple domains, producing 536 UI-based data records out of approximately 24,000 extracted records. In this sample, we identified 379 true positives, 30 false positives, 98 true negatives, and 29 false negatives, resulting in a precision of **0.926** and a recall of **0.928** (F1-score: **0.927**).

For system API analysis, we rely on SootFX [43], an established static analysis framework. SootFX detects system API sources in approximately 71.42 seconds per app. As SootFX is widely validated and used in prior work [43], we treat its identification of API-level data flows as ground-truth for the purpose of our analyses.

### 5.1 Experiment 1: Data Reporting

In the first study, we focus on popular Android apps, analyzing how they disclose collected data through their data safety sections. Our empirical evaluation addresses the following research questions:

**RQ1.1.** How do data safety sections report data collection and sharing?

**RQ1.2.** How does the reported data collection compare to the data actually collected by the source code?

For our experiment, we initially selected 70 Android apps. In November 2023, we referred to AppBrain statistics [41] to select 10 popular apps from each of the following (7) domains: *Messaging and Social Media, Banking and Finance, News and Entertainment, Sports and Games, Technology and Education, E-Commerce and Shopping, and Health and Fitness.* Based on the findings from RQ1.1, we narrowed down our focus to 20 apps in RQ1.2. Since the experiment was conducted in November 2023, some data safety sections may have been updated since then.

***RQ1.1. How do data safety sections report data collection and sharing?*** To answer RQ1.1, we manually examine the data safety sections of 70 apps to understand how data collection and sharing is reported. The data safety section classifies all user data into different *data categories*, which are further divided into specific *data types* (cf. Figure 1b). This data can be collected or shared for 7 different *purposes* predefined by Google. For RQ1.1, we use the classification proposed by Khandelwal et al. [20] to categorize data safety sections that report the collection of all data types or attribute most data types to 6 or 7 purposes as *over-reporting* practices. Conversely, sections that claim no data collection or report only one data type are categorized as *under-reporting* practices.

**Data collection.** Among all the apps that claim to collect data, only two apps (Signal[1] and Telegram[2]) claim not to collect *device or other IDs*. The most frequently reported data categories are *device or other IDs, personal information*, and *location. Sports and Games* apps claim to collect the highest number of data types and categories. Surprisingly, 8 data safety forms (11%) we analyze *over-report* the data collection, reporting the collection of all data types (Instagram, details below) or attributing most data types to all (7) or nearly all (6) purposes. In contrast, 9 forms (13%) *under-report* the data collection, either claiming no data collection or reporting the collection of just one data type (Signal, details below).

**Data sharing and security practices.** 26 data safety sections (37%) claim that the app does not share any data with third parties, even though their privacy policies indicate otherwise. Western Union[3] claims that data is not collected but is shared. SkyMap[4] claims that no data is collected or shared, but it can be encrypted. However, the same app claims that users cannot request data to be deleted. We categorize these cases as instances that *inconsistently report* data collection and sharing.

***Finding 1.1.*** Many apps appear to inconsistently report data collection and sharing via their data safety sections. This is consistent with a large-scale study conducted by Khandelwal et al. [20].

***RQ1.2. How does the reported data collection compare to the data actually collected by the source code?***

To answer RQ1.2, we analyze the app source code and compare its data collection with the data safety section. For this experiment, we compare the results of our prototype (cf. Section 4) with the data safety sections of 20 apps across all domains. This study focuses on the 20 apps we previously identified as over-reporting and under-reporting data collection (**RQ1.1**). Focusing on these ambiguous cases enables us to examine whether the apps that *seem to misreport* data are *actually doing so*, and to identify the specific factors, such as ambiguous category definitions, UI-based data not

[1] https://signal.org/
[2] https://telegram.org/
[3] https://www.westernunion.com/
[4] https://play.google.com/store/apps/details?id=com.google.android.stardroid/

being recognized as user inputs, or developers interpreting categories differently, that contribute to these discrepancies. These examples are therefore intended to illustrate the types of challenges developers may face, rather than to make claims of statistical generalizability.

In addition to detecting privacy-related data in the source code and user interfaces, we statically extract the permissions declared in the app's *AndroidManifest.xml* file. Given that the Android developer documentation references app permissions as a means to complete the data safety form [37], we use permissions as another means to examine how effectively these apps report data collection.

Since the data labels detected by our prototype are similar to the data categories in the data safety form, our experiment allows us to examine 10 distinct data categories out of the 14 present in the data safety section (cf. Table 6). Our results validate that 14 apps (**70%**) reporting *device or other IDs* do indeed collect such data, making it the most consistently reported category, closely followed by *personal information* (13 apps correctly report it: **65%**). This is encouraging, since these two categories comprise personal data (risks 1 and 2). In contrast, 10 apps (**50%**) reporting *health and fitness data* and *calendar data* do not use the correct manifest permissions, making these the most inconsistently reported data categories. We observe that all apps which under-report data collection miss reporting *location* through the data safety section. This is concerning since this category encompasses risk 2 personal data. We now discuss two apps in detail:

We observe that Signal only reports the collection of phone number (personal information) in their data safety section. In contrast, our experiment reveals that the app's user interfaces also collect location data (country name) and access data such as username, password, PINs. Furthermore, the app's source code uses system APIs that collect device IDs such as IP address and MAC address, personal information borrowed from Google such as client ID, email address, user ID, name, profile photo. It also collects audio, location, and browsing data. Our findings indicate that Signal collects device data such as SIM card information, but the data safety form does not have a separate category to report such data.

We observe that Signal declares 70 permissions in its manifest file. A careful examination of these permissions reveals that their data safety section should include the collection of audio, contacts, location, messages, and calendar data. This confirms our hypothesis that *Signal under-reports data collection in their data safety form.*

In contrast to Signal, Instagram[5] reports the collection of **all** data categories in their data safety form. However, our results reveal that Instagram's user interfaces only collect financial information, specifically card number and CVV. The source code uses system APIs which collect device IDs such as IP address and MAC address, location, and personal information. Instagram also collects device data such as mobile network information, but the data safety form does not have a separate category to report such data.

We extract the 40 permissions that Instagram declares in its manifest file. A careful examination of these permissions reveals that their data safety section should report collecting personal information, audio, contacts, location, and photos and

[5]https://www.instagram.com/

**Table 6**: Comparison between our results and the data safety section of 20 popular apps. Highlighted apps are discussed in detail.

| | **Categories in the Data Safety Section** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| ⋆: collected, ◯: reported, ⊛: collected and reported | **Device or other IDs** | **Personal info** | **Audio** | **Contacts** | **Location** | **Photos and videos** | **Financial info** | **Messages** | **Health and fitness** | **Calendar** |
| Signal | ⋆ | ⊛ | ⋆ | ⋆ | ⋆ | ⋆ | | ⋆ | | ⋆ |
| SkyMap | ⋆ | ⋆ | | | ⋆ | | | | | |
| Western Union | ⋆ | ⋆ | ⋆ | ⋆ | ⋆ | ⋆ | | | | |
| Yazio | ⊛ | ⊛ | | | ⋆ | ⊛ | | | ⋆ | |
| Temu | ⊛ | ⊛ | | | ⋆ | ⊛ | ⊛ | | | |
| DuckDuckGo | ⋆ | ⋆ | ⋆ | | ⋆ | ⋆ | | | | |
| Covpass | ⋆ | ⋆ | | | ⋆ | | | | | |
| Trust | ⊛ | ⊛ | | | ⋆ | | ⋆ | ◯ | | |
| Paytm | ⋆ | ⊛ | ⋆ | ⊛ | ⋆ | ⋆ | ⋆ | | | ⋆ |
| Instagram | ⊛ | ⊛ | ⊛ | ⊛ | ⊛ | ⊛ | ⊛ | ◯ | ◯ | ◯ |
| Gmail | ⊛ | ⊛ | ⊛ | ⊛ | ⊛ | ⊛ | ◯ | ⊛ | | ⊛ |
| Discord | ⊛ | ◯ | ⊛ | ⊛ | ⊛ | ⊛ | | ◯ | | |
| Youtube | ⊛ | ⊛ | ⊛ | ◯ | ⊛ | ⊛ | ⊛ | ⊛ | ⋆ | |
| Flo | ⊛ | ⊛ | | | ⊛ | ◯ | ⊛ | ◯ | ◯ | |
| Candy Crush Saga | ⊛ | ⊛ | | | ⊛ | ⋆ | ◯ | ◯ | | |
| Babbel | ⊛ | ◯ | ⊛ | | ⊛ | ⋆ | ◯ | | | |
| Google Maps | ⊛ | ⊛ | ⊛ | ⊛ | ⊛ | ⊛ | | ⊛ | ⋆ | |
| Netflix | ⊛ | ◯ | ⊛ | | ⊛ | ⋆ | | | | |
| Fitbit | ⊛ | ⊛ | ⋆ | ⊛ | ⊛ | ⋆ | ⊛ | ⊛ | ⊛ | ⋆ |
| Amazon | ⊛ | ⊛ | ⊛ | ⋆ | ⊛ | ⊛ | ◯ | ◯ | ⊛ | |

videos. Consequently, the inclusion of other data categories in their existing form such as calendar, health data appears redundant, thus confirming our hypothesis that *Instagram over-reports data collection in their data safety section.*

***Finding 1.2.*** Our experiment reveals evidence of over- and under-reporting by Android app developers. We also identify the most inconsistently reported data categories (location, health and fitness, and calendar).

## 5.2 Experiment 2: Data Collection Trends

We now extend our experiment to a larger number of apps, and aim to observe which categories of privacy-related data are most frequently collected by Android apps. Throughout this experiment, we use the multi-layered definition of privacy-related data we presented in Section 3. Our empirical evaluation addresses the following research questions:

**RQ2.1.** How do Android apps collect privacy-related data?

**Table 7**: Statistics of 7,500 apps. All tables in this section present percentages between 50-100 in red, 20-50 in yellow, and below 20 in green.

| | #Apps | %Apps |
|---|---|---|
| Successful analysis (UI) | 6922 | 92.3% |
| Successful analysis (system API) | 7150 | 95.33% |
| Parsing errors (UI) | 578 | 7.7% |
| Parsing errors (system) | 350 | 4.66% |
| With UI fields | 6600 | 95.32% |
| With privacy-related UI fields | 2182 | 31.52% |
| With system API privacy-related data | 7150 | 100% |
| With both UI and system API privacy-related data | 2182 | 31.52% |
| With permissions | 6378 | 92.14% |
| With UI and system API privacy-related data but no permissions | 57 | 0.82% |

**Table 8**: UI privacy-related data distribution (2182 apps).

| Data | #Apps | %Apps |
|---|---|---|
| Directly identifiable personal data | 694 | 31.81% |
| Partially identifiable personal data | 1094 | 50.14% |
| Access data | 596 | 27.31% |
| Context-dependent data | 987 | 45.23% |

**Table 9**: System API privacy-related data distribution (7150 apps).

| Data | #Apps | %Apps |
|---|---|---|
| Directly identifiable personal data | 7150 | 100% |
| Partially identifiable personal data | 7150 | 100% |
| Access data | 7150 | 100% |
| Context-dependent data | 5821 | 81.41% |

**RQ2.2.** Which data categories are directly collected from users by the most downloaded apps across different domains?

***RQ2.1. How do Android apps collect privacy-related data?***

To answer RQ2.1, we evaluate our prototype on 7,500 randomly chosen Android apps. We downloaded these apps using *AndroZoo* [47], a research database of Android applications created and maintained by University of Luxembourg. Since GDPR was introduced in 2018, we selected apps that were released on Google Play Store after October 2018. We downloaded APKs with a maximum size of 300 MB.

We present the general results in Table 7. We observe that 95.32% of the Android apps that our prototype could successfully parse collect data from user interfaces. Among them, **31.52%** of apps collect privacy-related data through user interfaces. We observe that *all* of the apps that our prototype could successfully parse collect at least *some* privacy-related data through system API calls in their source code, details given below.

**All** apps that collect privacy-related data through user interfaces collect some privacy-related data through system APIs, too (cf. Table 7). We summarize privacy-related data distribution in Table 8 and Table 9.

**Table 10**: UI personal data distribution (2182 apps).

| Data | Category | #Apps | %Apps |
|---|---|---|---|
| Directly identifiable personal data | Personal information | 629 | 28.83% |
| | Device or other IDs | 233 | 10.68% |
| | Financial information | 130 | 5.96% |
| Partially identifiable personal data | Personal information | 764 | 35.01% |
| | Financial information | 757 | 34.69% |
| | Location data | 357 | 16.36% |
| | Health and fitness | 177 | 8.11% |
| | Calendar data | 143 | 6.55% |

**Table 11**: System API personal data distribution (7150 apps).

| Data | Category | #Apps | %Apps |
|---|---|---|---|
| Directly identifiable personal data | Device or other IDs | 7150 | 100% |
| | Personal information | 1551 | 21.69% |
| Partially identifiable personal data | Personal information | 7150 | 100% |
| | Location data | 7150 | 100% |
| | Browsing data | 7150 | 100% |
| | Device data | 3260 | 45.59% |
| | Audio data | 1751 | 24.49% |
| | Session data | 259 | 3.62% |
| | App activity | 253 | 3.54% |
| | Photos and videos | 51 | 0.71% |
| | Calendar data | 15 | 0.21% |

A notable observation is that **50.14%** of the apps that gather privacy-related data through user interfaces collect *partially identifiable personal data* (risk 2), while only **31.81%** of apps collect *directly identifiable personal data* (risk 1). In contrast, all apps that have reachable privacy-related data sources through system APIs collect personal data.

To examine this observation in detail, we provide a detailed breakdown of personal data in Table 10 and Table 11. We observe that user interfaces collect *directly identifiable and partially identifiable personal information* almost to the same extent as *partially identifiable financial information*. By our observation, user interfaces commonly collect personal information such as names, email addresses, and phone numbers, along with access data like passwords.

In contrast, calls to system APIs mostly collect device, location and browsing information through the OS. The identifier API dataset links one category to multiple identifiers, which are then associated with several system API calls. Consequently, the presence of even one dominant system call indicates data collection. We can attribute the 100% data collection in four categories to specific identifiers like IP address and MAC address (device or other IDs), user data from URLs (personal information), approximate location (location data) and HTTP cookie name (browsing data).

Our experiment reveals that only **27.31%** of all Android apps collect access data through user interfaces. The high percentage of system API-based access data collection (cf. Table 9) can be traced back to predominant network-based authentication present in the source code, such as the use of the *java.net.PasswordAuthentication: char[] getPassword()* system call.

Moreover, we observe that **45.23%** of all apps collect context-dependent data through user interfaces, emphasizing the importance of dynamic analysis to effectively assess the risks associated with collection of such data in specific contexts. The high percentage (81.41%) of context-dependent data collection through system API calls can be attributed to the presence of the *findViewById(..)* method call, which connects the source code with the text fields in the layout files.

***Finding 2.1.*** Through user interfaces, Android apps predominantly collect personal and financial information; and device, location and browsing information through system APIs. *Partially identifiable personal data* (risk 2) is the most frequently collected data.

**Table 12**: Statistics of 700 most downloaded apps (100 from each domain).

| | #Apps | %Apps |
|---|---|---|
| Successful analysis (UI) | 683 | 97.57% |
| Successful analysis (system API) | 668 | 95.43% |
| Parsing errors (UI) | 17 | 2.42% |
| Parsing errors (system API) | 32 | 4.57% |
| With UI fields | 677 | 99.12% |
| With privacy-related UI fields | 552 | 80.81% |
| With system API privacy-related data | 668 | 100% |
| With both UI and system API privacy-related data | 552 | 80.81% |
| With permissions | 665 | 97.36% |
| With UI and system API privacy-related data but no permissions | 3 | 0.44% |

**Table 13**: UI privacy-related data distribution (552 most downloaded apps).

| Data | #Apps | %Apps |
|---|---|---|
| Directly identifiable personal data | 344 | 62.32% |
| Partially identifiable personal data | 423 | 76.63% |
| Access data | 259 | 46.92% |
| Context-dependent data | 521 | 94.38% |

***RQ2.2. Which data categories are directly collected from users by the most downloaded apps across different domains?***

To answer RQ2.2, we extend our experiment to analyze in depth 100 apps each from the following domains: *Messaging and Social Media, Banking and Finance, News and Entertainment, Sports and Games, Technology and Education, E-Commerce and Shopping, and Health and Fitness.* In July 2023, we referred to Google Play Store and AppBrain statistics [41] to select 50 most downloaded apps from each domain. We used AppBrain and AndroidRank statistics [48] again in October 2023 to choose 50 additional most downloaded apps from these domains. We manually downloaded these APKs, and we evaluate this research question by running our prototype on these 700 APKs.

We present the general results in Table 12 and the privacy-related data distribution in Table 13. We observe that **76.63%** of the apps that gather privacy-related data through user interfaces collect *partially identifiable personal data*, while **62.32%** of apps collect *directly identifiable personal data.*

The domain-wise privacy-related data distribution is summarized in Table 14. Since the system API privacy-related data distribution is uniformly high due to the constant presence of some specific method calls in the source code, we focus on privacy-related data that is collected directly from the user via user interfaces (cf. Table 14).

We examine the different categories of personal data collected by user interfaces for different domains. The results are illustrated in Table 15, and can be summarized as follows:

- The highest percentage of apps collecting *directly identifiable financial information* (34.44%) and *partially identifiable location data* (61.11%) belong to the domain ***E-commerce and Shopping***.

**Table 14**: Domain-wise UI privacy-related data distribution. Highlighted: highest percentage of apps collecting data.

| Data | Percentage of apps that collect data (per domain) | | | | | | |
|---|---|---|---|---|---|---|---|
| | E-commerce and Shopping | Messaging and Social Media | Banking and Finance | Sports and Games | Technology and Education | Health and Fitness | News and Enter-tainment |
| Directly identifiable personal data | 71.11% | **84.34%** | 57.89% | 39.22% | 52.27% | 67.09% | 56.47% |
| Partially identifiable personal data | 83.33% | **91.57%** | 76.32% | 54.90% | 78.41% | 81.01% | 65.88% |
| Access data | 55.56% | **60.24%** | 55.26% | 29.41% | 42.05% | 48.10% | 32.94% |
| Context-dependent data | **97.78%** | 97.59% | 96.05% | 84.31% | 93.18% | 97.47% | 91.76% |

**Table 15**: Domain-wise UI personal data distribution. Highlighted: highest percentage of apps collecting data. Risk 1: directly identifiable, Risk 2: partially identifiable.

| Data | Category | Percentage of apps that collect data (per domain) | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | E-com and Shopping | Messaging and Social Media | Banking and Finance | Sports and Games | Technology and Education | Health and Fitness | News and Enter-tainment |
| Risk 1 | Personal information | 65.56% | **73.49%** | 46.05% | 33.33% | 42.05% | 54.43% | 45.88% |
| | Financial information | **34.44%** | 25.30% | 26.32% | 3.92% | 15.91% | 17.72% | 8.24% |
| | Device or other IDs | 35.56% | **54.22%** | 27.63% | 15.69% | 25% | 36.71% | 23.53% |
| Risk 2 | Personal information | 65.56% | **77.11%** | 52.63% | 41.18% | 63.64% | 64.56% | 55.29% |
| | Location data | **61.11%** | 48.19% | 34.21% | 17.65% | 21.59% | 27.85% | 21.18% |
| | Financial information | 34.44% | 14.46% | **44.74%** | 0% | 6.82% | 22.78% | 3.53% |
| | Calendar data | 18.89% | 13.25% | 19.74% | 0% | 10.23% | **21.52%** | 5.88% |
| | Health and fitness | 3.33% | 7.23% | 2.63% | 1.96% | 3.41% | **17.72%** | 4.71% |

- The ***Messaging and Social Media*** domain has the highest proportion of apps collecting *directly identifiable personal information* (73.49%), *directly identifiable device or other IDs* (54.22%) and *partially identifiable personal information* (77.11%).
- The highest percentage of apps collecting *partially identifiable financial information* (44.74%) belong to the domain ***Banking and Finance***.
- Among all the various domains, ***Sports and Games*** apps collect the least amount of personal data.
- Unsurprisingly, ***Health and Fitness*** apps collect *partially identifiable calendar data* (21.52%) and *partially identifiable health and fitness information* (17.72%).

**Table 16**: Identifier combinations collected by most downloaded apps, Highlighted: highest percentage of apps collecting the identifier combination.

| | Percentage of apps that collect the identifier combination | | | | | | |
|---|---|---|---|---|---|---|---|
| **Identifier combination** | **E-commerce and Shopping** | **Messaging and Social Media** | **Banking and Finance** | **Sports and Games** | **Technology and Education** | **Health and Fitness** | **News and Enter-tainment** |
| Name + Approx location | **41.11%** | 33.73% | 19.74% | 11.76% | 18.18% | 22.78% | 12.66% |
| Phone number + Email address | 34.44% | **34.94%** | 25% | 11.76% | 7.95% | 22.78% | 16.46% |
| Photo + Text | 23.33% | **27.71%** | 18.42% | 13.73% | 18.18% | 26.58% | 24.71% |
| Name + Postal address | **21.11%** | 14.46% | 14.47% | 3.92% | 7.95% | 8.86% | 6.33% |
| Approx location + Postal address | **21.11%** | 9.64% | 10.53% | 3.92% | 5.68% | 6.33% | 5.06% |
| Name + Card details | **23.33%** | 6.02% | 22.37% | 0% | 4.55% | 7.59% | 1.27% |
| Email address + Card details | **20%** | 6.02% | 14.47% | 1.96% | 6.82% | 7.59% | 1.27% |
| Card details + Bill amount | 10% | 2.41% | **19.74%** | 0% | 0% | 1.27% | 12.66% |
| Name + Birthday | 4.44% | **4.82%** | 1.32% | 0% | 0% | 2.53% | 1.27% |

- The most collected data category among all the domains is *partially identifiable personal information*, closely followed by *directly identifiable personal information.*

According to our observations (cf. Table 14), 60.24% of ***Messaging and Social Media*** apps collect access data through their user interfaces, which is the highest among all domains. They are followed by ***E-Commerce and Shopping*** apps (55.56%) and ***Banking and Finance*** apps (55.26%). This is surprising, since one would reasonably expect shopping and banking apps to authenticate their users at least as much as messaging and social media apps.

Our dataset contains various identifiers that can *partially identify* a user. This suggests that one can define "dangerous combinations of data" based on these categories and alert app developers about them. For example, a user's first name is insufficient to identify them, but combining it with other information like postal address or family name or age could potentially identify the user. In a broader sense, combining multiple instances of risk 2 data could result in risk 1 data.

To this end, we automatically analyze the report generated by our prototype and examine some combinations of partial identifiers collected by the selected 700 apps. The results, summarized in Table 16, reveal that ***E-Commerce and Shopping*** apps collect the highest percentage of partial identifier combinations. These include *names* along with other partial identifiers related to location (*postal address, approximate location*) and financial information (like *card details*). Unsurprisingly, ***Banking and Finance*** apps collect multiple partial identifiers related to financial information. Our results indicate that ***Messaging and Social Media*** apps collect context-dependent data as by combining photos and texts/SMSs. It may be sensible to assign a higher risk ranking to context-dependent data collected by a health and banking app as compared to context-dependent data collected by other app domains.

***Finding 2.2.*** Through user interfaces, *Messaging and Social Media* apps predominantly collect personal and access data. In contrast, *Sports and Games* apps collect the least amount of privacy-related data. Many apps collect combinations of partial identifiers from users, with those in the *E-Commerce and Shopping* domain collecting the highest percentage of such combinations.

## 6 Discussion

In this section, we discuss the findings of both experiments (Section 5.1 and Section 5.2).

### 6.1 Experiment 1: Data Reporting

In Section 5.1, we examine the data safety sections of the most popular apps across different domains and observe that many Android apps either over-report or under-report data collection. This emphasizes the importance of supporting developers better in accurately reporting data collection. Our findings indicate that app developers struggle to accurately report data collection, despite the existence of tool support [21–23].

One major reason for this issue could be the suboptimal design of Google's data safety form. While our experiment considers data collected via user interface screens, system API calls in the app code, or manifest permissions as collected data, Google's definition of *collected data* is abstract and includes many exemptions [49]. For example, one does not need to disclose the collection of anonymized data, although the effectiveness of common anonymization techniques is disputed in the digital privacy community [50]. A study by Mozilla [11] questioned Google's decision and suggested eliminating the definition of anonymized data from the data safety form. In contrast, pseudonymized data needs to be disclosed, while end-to-end encrypted data does not need reporting. This raises the question of how app developers determine if data has been properly anonymized or pseudonymized. Additionally, data processed locally on the user's device and not sent off the device does not need to be disclosed either, which seems questionable, as substantial information might still be inferred from such data.

Another reason for under-reporting may be that the form lacks a separate category for data collected by third-party libraries used by the app. This could lead developers to believe they do not need to disclose data collected by these libraries [20], potentially giving users a false sense of privacy. This third-party collection information might only be available in the privacy policy of the app (if at all), which is often too long and verbose for users to read. Privacy-conscious users might want to know which data is collected by third-party analytics libraries used by the app, and adding this information is crucial towards enhancing transparency.

After examining the data collected by real-world Android apps, we believe that **all** personal data collected by an Android app should be declared in the *data collected* section of the data safety form, irrespective of whether it is pseudonymized or shared with third parties. The data shared with third parties can then be reported separately in the *data shared* section of the form. Furthermore, the form should be reordered to follow a risk-based reporting of collected data. For example, directly identifiable personal data like *device or other IDs* should be placed on top, followed by the less risky categories. A comparison of our results with the data safety section reveals that the data categories in the data safety form are not expressive enough to report all the data the system APIs collect from the device. Data categories in the form should be updated after conducting a thorough research of the system APIs that Android apps use to collect data. We make the identifier datasets open source[6] and public so that they can be used to enhance transparency and clarity in the data reporting documents. We further suggest the inclusion of the following new data categories:

1. `Device data`: Android APIs and other libraries collect some data from the device. This data can be partially identifiable and hence is different from the identifiable category of *Device or other IDs*. We suggest including a new data type for such data. Examples of such data are SIM card information, bluetooth information and mobile network information.
2. `Session data`: Media players, advertising platforms and analytics libraries collect session data to analyse user behavior, which does not have a distinct data type in the data safety form.
3. Sub-types in the data category `Device or other IDs`: We observe *Device or other IDs* to be a very coarse representation, and recommend distinguishing these IDs by their purposes. For example, IP address may be used for gaining access to location and installation ID may be used for advertising and analytics.

Khandelwal et al.'s study [20] noted a lack of educational resources and demos to aid app developers in completing the data safety section. Our findings indicate that app developers may be unaware of Google's guidelines for Android developers [37]. These guidelines specify manifest permissions and system APIs that should map directly to data safety labels, yet this mapping is often not followed in practice. For example, the privacy-conscious app Signal (discussed in Section 5) declares **70** permissions, many of which should correspond to specific data safety labels. However, as shown in Table 6, Signal exhibits the most under-reported data collection among the 20 apps we analyzed, suggesting that even developers of well-known apps may not fully follow the guidelines.

Furthermore, we observe that Google's guidelines for Android developers [37] and Google's data collection exemptions for the data safety section [49] can be complementary, potentially causing confusion for app developers. According to the Android developers' guide, Signal's usage of permissions should translate to data safety labels. However, if the data collected via those permissions is encrypted in the source code, app developers might refer to Google's data safety exemptions and choose to not report it as part of the data safety section. Our observations highlight the need for standardized guidelines and rules before introducing more tools to automate the

[6]Available at https://zenodo.org/records/14046829

**Table 17**: UI privacy-related data distribution for random apps (RQ2.1) vs. most downloaded apps (RQ2.2).

| Data | %Random Apps | %Most Downloaded Apps |
|---|---|---|
| Directly identifiable personal data | 31.81% | 62.32% |
| Partially identifiable personal data | 50.14% | 76.63% |
| Access data | 27.31% | 46.92% |
| Context-dependent data | 45.23% | 94.38% |

reporting process.

***Takeaway 1.*** App developers struggle to accurately report data collection, either due to the suboptimal design of Google's data safety form, lack of standardized guidelines, or insufficient existing tool support.

## 6.2 Experiment 2: Data Collection Trends

In Section 5.2, our quantitative study on 7,500 Android apps reveals that most apps collect data that can *partially identify* a user (risk 2 data). While system API methods constantly collect privacy-related data, user interfaces demonstrate more controlled patterns of data collection. A more focused study on various domain apps confirms that partially identifiable data is the most commonly collected type, with the user interfaces of highly downloaded apps collecting more privacy-related data than those of randomly selected apps (cf. Table 17). This finding highlights the need for more refined UI analysis in privacy assessments, as existing approaches that focus solely on APIs may overlook significant data flows originating from app screens.

Our study also reveals domain-specific risks. *E-Commerce and Shopping* apps collect the highest proportion of partial identifier combinations. This raises important questions about how developers prevent these data elements from being combined in ways that could lead to user identification. For example, if multiple partially identifiable data points are processed together, developers need to ensure appropriate pseudonymization to prevent re-identification. Historical studies, such as Sweeney [51], have shown that combinations of pincode, gender, and date of birth can uniquely identify a large portion of the population in the United States. In our analysis, we identified three apps—Finopaytech (*Banking and Finance*), Aliexpress (*E-Commerce and Shopping*), and Meetme (*Messaging and Social Media*) — that directly collect these three partial identifiers. These findings highlight the urgent need for heuristics or automated checks to assess the risk associated with combinations of partial identifiers when analyzing Android apps via source code.

Our experiments also show inconsistencies in reporting to Google's data safety section. Location, health and fitness, and calendar data are among the most inconsistently reported categories (cf. Section 5.1). Domain-specific analysis (cf. Section 5.2) further indicates that personal information, device or other IDs, and location are frequently collected, yet often under-reported, especially in E-Commerce and Shopping apps, where location data is commonly collected but *rarely disclosed accurately*. These results point to a gap between actual app behavior and DSS reporting, emphasizing the need for greater consistency in privacy-aware app development and disclosure practices.

Finally, our evaluation highlights actionable opportunities for supporting developers through tooling. The prototype reveals nuanced data collection patterns in user interfaces that can inform the design of privacy-aware apps. A potential plugin could leverage these insights to alert developers in real-time about the data collected on specific screens, guiding them toward "privacy by design" practices and helping ensure accurate reporting.

***Takeaway 2.*** The user interfaces of the most downloaded apps collect significant amounts of privacy-related data, with *partially identifiable data* (risk 2) being the most frequently collected. Although *location data* is frequently collected, it is often under-reported in the apps' data safety sections.

## 7 Threats to Validity

We next discuss the threats to the validity of our experiment, and how we sought to mitigate them.

**Datasets.** The accuracy of the results depends on how precisely one constructs the datasets. The degree of identifiability relies on human subjectivity and it may vary with different domains and compliance regimes. The proposed definition of data focuses on GDPR but modifying the datasets is sufficient to adapt the analysis to alternative compliance regimes.

Another threat stems from the annotation process. After independently annotating the initial corpus, the first and second authors identified a subset of 50 keywords on which they disagreed or considered ambiguous. Instead of relying on a formal inter-annotator agreement metric, we sought feedback from a broader group of 22 domain-adjacent researchers. Although this approach helped refine several classifications and led to the introduction of an additional category, it may not fully eliminate subjectivity in interpreting keyword meaning. Moreover, the selection of 50 keywords for external review may limit the generalizability of the feedback.

Since certain keywords in the *identifier keywords dataset* may have different meanings in different contexts, they may produce false positives. In early experiments, we observed that the UI analysis automatically labeled the keywords "body" and "height" as *partially identifiable health and fitness data.* In reality, input fields in messaging and e-commerce apps use the keyword "body" when describing the body of a chat or an email, and many apps use the keyword "height" to describe the dimensions of an image, making these input fields *context-dependent data* instead. The mislabeling initially occurred because we prioritized labeling risk 2 data (partially identifiable data) before moving on to risk 4 data (context-dependent data), causing these keywords to be labeled incorrectly. To address this issue, we adjusted the order of precedence levels of risk 2 health data in different domains, reducing false positives by 0.5%. We consider the remaining false positives to be insignificant. However, in the future, we aim to explore more advanced Natural Language Processing techniques similar to UiRef's disambiguation strategy [30] to further mitigate the risk of false positives.

Despite the process of third-party library selection we outlined in Section 4 and Table 3, the *identifier API dataset* may miss out on some libraries. We will explore whether using library detection tools [52–54] can help us cover a wider range of libraries.

**Prototype.** The UI analysis is currently limited to textual data collected by an Android app, but it can be broadened to include a wider range of UI widgets. It focuses on scanning the layout files of Android apps, and does not associate privacy-related input fields with their corresponding variables in the source code.

For system API analysis, SootFX matches the method signatures in the call graph with the labeled methods present in the *identifier API dataset*. Since it uses keyword matching to detect the presence of data sources including those from some third-party libraries, its analysis will not be able to handle code obfuscation in cases where the obfuscation renames also calls to these third-party APIs.

**Experiments.** While the number of apps in our experiments might appear limited, we ensure a comprehensive evaluation by including the most popular apps from different domains actively used by people. Since we had to manually download the domain-specific apps for experiment 2, we chose 100 apps per domain. However, selecting top apps per domain could threaten external validity, so our findings may not be generalizable to other domains.

## 8 Conclusion

In this paper, we have introduced a multi-layered definition of privacy-related data to correctly report data collection in Android apps. This data definition focuses on GDPR's definition of personal data. We have further described how we created a dataset of privacy-sensitive data classes that researchers may use to classify inputs to Android apps. This dataset takes into account data collected both through the user interface and system APIs.

We have developed a prototype that statically extracts and labels privacy-related data collected via app source code, user interfaces, and permissions. We have conducted two experiments with this prototype: (1) a small-scale analysis to examine which data categories are *most consistently reported* in the data safety sections of Android apps, and (2) a large-scale study to observe which categories of privacy-related data are *most frequently collected* by Android apps.

**Experiment 1:** We have conducted a case study to observe how data collection is reported via the data safety sections of 70 apps, and observed over- and under-reporting of collected data (**RQ1.1**). Using the results from two *Messaging and Social Media* apps (Signal and Instagram), we have discussed how app developers under-report and over-report data collection, respectively, and identified inaccurately reported data categories (**RQ1.2**). Our findings have shown that app developers struggle to accurately report data collection, either due to Google's abstract definition of collected data or insufficient existing tool support.

**Experiment 2:** Next, we have conducted a quantitative study on 7,500 Android apps, focusing on apps released on Google Play Store after the introduction of GDPR. This study has revealed that apps most frequently collect data that can *partially identify* users. While system APIs consistently collect large amounts of location and device privacy-related data, user interfaces exhibit some more diverse data collection patterns (**RQ2.1**). A more focused study on various domains of apps reveals that the largest fraction of apps collecting personal data belong to the domain of *Messaging and Social Media*. In contrast, *Sports and Games* apps collect the least amount of privacy-related data (**RQ2.2**). Our findings have shown that *location* is collected frequently by apps, specially from the *E-commerce and Shopping* domain. However, it is often under-reported in app data safety sections. Our results have highlighted the need for greater consistency in privacy-aware app development and reporting practices.

We next aim to interview Android developers to understand their perceptions of collected data and evaluate whether existing tools assist them effectively in completing the data safety section.

## References

[1] Google Play Statistics (2023). https://www.statista.com/statistics/266210/number-of-available-applications-in-the-google-play-store/

[2] The European parliament and the council of the European union. General Data Protection Regulation (GDPR) (2018). https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679

[3] Cavoukian, A.: Privacy by design: The 7 foundational principles (2009). https://privacy.ucsc.edu/resources/privacy-by-design---foundational-principles.pdf

[4] Horstmann, S.A., Domiks, S., Gutfleisch, M., Tran, M., Acar, Y., Moonsamy, V., Naiakshina, A.: "those things are written by lawyers, and programmers are reading that." mapping the communication gap between software developers and privacy experts. Proc. Priv. Enhancing Technol. **2024**(1), 151–170 (2024) https://doi.org/10.56553/POPETS-2024-0010

[5] Moonsamy, V., Batten, L.: Android applications: Data leaks via advertising libraries. In: 2014 International Symposium on Information Theory and Its Applications, pp. 314–317 (2014)

[6] Yu, L., Luo, X., Liu, X., Zhang, T.: Can we trust the privacy policies of android apps? In: 2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), pp. 538–549 (2016). https://doi.org/10.1109/DSN.2016.55

[7] Zimmeck, S., Wang, Z., Zou, L., Iyengar, R., Liu, B., Schaub, F., Wilson, S., Sadeh, N., Bellovin, S., Reidenberg, J.: Automated Analysis of Privacy Requirements for Mobile Apps. Proceedings 2017 Network and Distributed System

Security Symposium. Korea Society of Internet Information, Korea, Republic of (2017). https://doi.org/10.14722/ndss.2017.23034

[8] Wang, X., Qin, X., Hosseini, M.B., Slavin, R., Breaux, T.D., Niu, J.: Guileak: Tracing privacy policy claims on user input data for android applications. In: Proceedings of the 40th International Conference on Software Engineering. ICSE '18, pp. 37–47. Association for Computing Machinery, New York, NY, USA (2018). https://doi.org/10.1145/3180155.3180196 . https://doi.org/10.1145/3180155.3180196

[9] Tan, Z., Song, W.: Ptpdroid: Detecting violated user privacy disclosures to third-parties of android apps. In: 2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE), pp. 473–485 (2023). https://doi.org/10.1109/ICSE48619.2023.00050

[10] Data Safety Section (2022). https://blog.google/products/google-play/data-safety/

[11] See No Evil: Loopholes in Google's Data Safety Labels Keep Companies in the Clear and Consumers in the Dark (2023). https://foundation.mozilla.org/en/campaigns/googles-data-safety-labels/

[12] Khedkar, M., Mondal, A.K., Bodden, E.: Do android app developers accurately report collection of privacy-related data? In: Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering Workshops. ASEW '24, pp. 176–186. Association for Computing Machinery, New York, NY, USA (2024). https://doi.org/10.1145/3691621.3694949 . https://doi.org/10.1145/3691621.3694949

[13] What is GDPR? (2018). https://gdpr.eu/what-is-gdpr/

[14] Konstantinos Limniotis, Marit Hansen: Recommendations on shaping technology according to gdpr provisions - an overview on data pseudonymisation. Technical report, European Union Agency for Cybersecurity (ENISA) (2019). https://www.enisa.europa.eu/publications/recommendations-on-shaping-technology-according-to-gdpr-provisions

[15] M. Jensen, C. Lauradoux, and K. Limniotis: Pseudonymisation techniques and best practices. Technical report, European Union Agency for Cybersecurity (ENISA) (2019). https://www.enisa.europa.eu/publications/pseudonymisation-techniques-and-best-practices

[16] Kelley, P.G., Bresee, J., Cranor, L.F., Reeder, R.W.: A "nutrition label" for privacy. In: Proceedings of the 5th Symposium on Usable Privacy and Security. SOUPS '09. Association for Computing Machinery, New York, NY, USA (2009). https://doi.org/10.1145/1572532.1572538 . https://doi.org/10.1145/1572532.1572538

[17] Kelley, P.G., Cranor, L.F., Sadeh, N.: Privacy as part of the app decision-making process. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. CHI '13, pp. 3393–3402. Association for Computing Machinery, New York, NY, USA (2013). https://doi.org/10.1145/2470654.2466466 . https://doi.org/10.1145/2470654.2466466

[18] Apple: Transparency is the best policy. (2020). https://www.apple.com/privacy/labels/

[19] Gardner, J., Feng, Y., Reiman, K., Lin, Z., Jain, A., Sadeh, N.: Helping mobile application developers create accurate privacy labels. In: 2022 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), pp. 212–230 (2022). https://doi.org/10.1109/EuroSPW55150.2022.00028

[20] Khandelwal, R., Nayak, A., Chung, P., Fawaz, K.: Unpacking Privacy Labels: A Measurement and Developer Perspective on Google's Data Safety Section (2023)

[21] Castelly, N., Hurley, F.: Introducing Checks: Simplifying Privacy for App Developers - Google: The Keyword. https://blog.google/technology/area-120/checks/

[22] Li, T., Cranor, L.F., Agarwal, Y., Hong, J.I.: Matcha: An ide plugin for creating accurate privacy nutrition labels. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies **8**(1), 1–38 (2024) https://doi.org/10.1145/3643544

[23] Privado.ai (2022). https://www.privado.ai/data-safety-report

[24] Rasthofer, S., Arzt, S., Bodden, E.: A machine-learning approach for classifying and categorizing android sources and sinks. In: 2014 Network and Distributed System Security Symposium (NDSS) (2014). https://www.bodden.de/pubs/rab14classifying.pdf

[25] Arzt, S., Rasthofer, S., Fritz, C., Bodden, E., Bartel, A., Klein, J., Le Traon, Y., Octeau, D., McDaniel, P.: 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. PLDI '14, pp. 259–269. Association for Computing Machinery, New York, NY, USA (2014). https://doi.org/10.1145/2594291.2594299 . https://doi.org/10.1145/2594291.2594299

[26] Huang, J., Zhang, X., Tan, L., Wang, P., Liang, B.: Asdroid: Detecting stealthy behaviors in android applications by user interface and program behavior contradiction. In: Proceedings of the 36th International Conference on Software Engineering. ICSE 2014, pp. 1036–1046. Association for Computing Machinery, New York, NY, USA (2014). https://doi.org/10.1145/2568225.2568301 . https://doi.org/10.1145/2568225.2568301

[27] Pandita, R., Xiao, X., Yang, W., Enck, W., Xie, T.: Whyper: Towards automating risk assessment of mobile applications. In: Proceedings of the 22nd USENIX Conference on Security. SEC'13, pp. 527–542. USENIX Association, USA (2013)

[28] Huang, J., Li, Z., Xiao, X., Wu, Z., Lu, K., Zhang, X., Jiang, G.: SUPOR: Precise and scalable sensitive user input detection for android apps. In: 24th USENIX Security Symposium (USENIX Security 15), pp. 977–992. USENIX Association, Washington, D.C. (2015). https://www.usenix.org/conference/usenixsecurity15/technical-sessions/presentation/huang

[29] Nan, Y., Yang, M., Yang, Z., Zhou, S., Gu, G., Wang, X.: UIPicker: User-Input privacy identification in mobile applications. In: 24th USENIX Security Symposium (USENIX Security 15), pp. 993–1008. USENIX Association, Washington, D.C. (2015). https://www.usenix.org/conference/usenixsecurity15/technical-sessions/presentation/nan

[30] Andow, B., Acharya, A., Li, D., Enck, W., Singh, K., Xie, T.: Uiref: analysis of sensitive user inputs in android applications. In: Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks. WiSec '17, pp. 23–34. Association for Computing Machinery, New York, NY, USA (2017). https://doi.org/10.1145/3098243.3098247 . https://doi.org/10.1145/3098243.3098247

[31] Tang, F., Østvold, B.M.: Assessing software privacy using the privacy flow-graph. In: Proceedings of the 1st International Workshop on Mining Software Repositories Applications for Privacy and Security. MSR4P&S 2022, pp. 7–15. Association for Computing Machinery, New York, NY, USA (2022). https://doi.org/10.1145/3549035.3561185 . https://doi.org/10.1145/3549035.3561185

[32] Li, L., Bartel, A., Bissyandé, T.F., Klein, J., Le Traon, Y., Arzt, S., Rasthofer, S., Bodden, E., Octeau, D., McDaniel, P.: Iccta: Detecting inter-component privacy leaks in android apps. In: 2015 IEEE/ACM 37th IEEE International Conference on Software Engineering, vol. 1, pp. 280–291 (2015). https://doi.org/10.1109/ICSE.2015.48

[33] Piskachev, G., Nguyen Quang Do, L., Bodden, E.: Codebase-adaptive detection of security-relevant methods. In: ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA) (2019)

[34] Piskachev, G., Do, L.N.Q., Johnson, O., Bodden, E.: Swanassist: semi-automated detection of code-specific, security-relevant methods. In: Proceedings of the 34th IEEE/ACM International Conference on Automated Software Engineering. ASE '19, pp. 1094–1097. IEEE Press, Piscataway, NJ, USA (2020). https://doi.org/10.1109/ASE.2019.00110 . https://doi.org/10.1109/ASE.2019.00110

[35] Samhi, J., Kober, M., Kabore, A.K., Arzt, S., Bissyandé, T.F., Klein, J.: Negative

results of fusing code and documentation for learning to accurately identify sensitive source and sink methods. In: 2023 30th Edition of the IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER) (2023)

[36] Kober, M., Samhi, J., Arzt, S., Bissyandé, T.F., Klein, J.: Sensitive and personal data: What exactly are you talking about? In: 2023 10th International Conference on Mobile Software Engineering and Systems 2023 (MobileSoft) (2023)

[37] Declare your app's data use (2022). https://developer.android.com/privacy-and-security/declare-data-use

[38] Opinion 04/2007 on the concept of personal data (2007). https://www.clinicalstudydatarequest.com/Documents/Privacy-European-guidance.pdf

[39] Su, J., Shukla, A., Goel, S., Narayanan, A.: De-anonymizing web browsing data with social networks. In: Proceedings of the 26th International Conference on World Wide Web. WWW '17, pp. 1261–1269. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE (2017). https://doi.org/10.1145/3038912.3052714 . https://doi.org/10.1145/3038912.3052714

[40] Kurtz, A., Gascon, H., Becker, T., Rieck, K., Freiling, F.: Fingerprinting mobile devices using personalized configurations. Proceedings on Privacy Enhancing Technologies **2016** (2016) https://doi.org/10.1515/popets-2015-0027

[41] AppBrain (2014). https://www.appbrain.com/

[42] Android-apkmirror (2014). https://www.apkmirror.com/

[43] Karakaya, K., Bodden, E.: Sootfx: A static code feature extraction tool for java and android. In: 2021 IEEE 21st International Working Conference on Source Code Analysis and Manipulation (SCAM), pp. 181–186 (2021). https://doi.org/10.1109/SCAM52516.2021.00030

[44] Android-apktool (2010). https://ibotpeaches.github.io/Apktool/

[45] Vallée-Rai, R., Co, P., Gagnon, E., Hendren, L., Lam, P., Sundaresan, V.: Soot: a java bytecode optimization framework. In: CASCON First Decade High Impact Papers. CASCON '10, pp. 214–224. IBM Corp., USA (2010). https://doi.org/10.1145/1925805.1925818 . https://doi.org/10.1145/1925805.1925818

[46] Lhoták, O., Hendren, L.: Scaling java points-to analysis using spark. In: Proceedings of the 12th International Conference on Compiler Construction. CC'03, pp. 153–169. Springer, Berlin, Heidelberg (2003)

[47] Allix, K., Bissyandé, T.F., Klein, J., Le Traon, Y.: Androzoo: collecting millions of android apps for the research community. Association for Computing Machinery,

New York, NY, USA (2016). https://doi.org/10.1145/2901739.2903508 . https://doi.org/10.1145/2901739.2903508

[48] AndroidRank (2011). https://androidrank.org/android-most-popular-google-play-apps?price=free

[49] Google's definition of data collection. (2022). https://support.google.com/googleplay/android-developer/answer/10787469?hl=en#zippy=%2Cdata-collection

[50] Anonymization in Data Privacy. (2019). https://www.theguardian.com/technology/2019/jul/23/anonymised-data-never-be-anonymous-enough-study-finds

[51] Sweeney, L.: Simple Demographics Often Identify People Uniquely. http://dataprivacylab.org/projects/identifiability

[52] LibD (2017). https://github.com/IIE-LibD/libd

[53] LibRadar (2016). https://github.com/pkumza/LibRadar

[54] Han, H., Li, R., Tang, J.: Identify and inspect libraries in android applications. Wirel. Pers. Commun. **103**(1), 491–503 (2018)