공학석사학위논문

Consistency Analysis between Privacy Policy and Mobile Application

개인정보 보호 정책과 모바일 어플리케이션의 일관성 분석

2021 년 2 월

서울대학교 대학원

컴퓨터공학부

정 윤 교

공학석사학위논문

Consistency Analysis between Privacy Policy and Mobile Application

개인정보 보호 정책과 모바일 어플리케이션의 일관성 분석

2021 년 2 월

서울대학교 대학원

컴퓨터공학부

정 윤 교

Consistency Analysis between Privacy Policy and Mobile Application

개인정보 보호 정책과 모바일 어플리케이션의 일관성 분석

지도교수 권 태 경

이 논문을 공학석사 학위논문으로 제출함

2020 년 12 월

서울대학교 대학원

컴퓨터공학부

정 윤 교

정 윤 교의 석사 학위논문을 인준함

2020 년 12 월

위 원 장	김종권	
부위원장	권태경	
위 원	전화숙	

Abstract

Consistency Analysis between Privacy Policy and Mobile Application

Yoonkyo Jung
Dept. of Computer Science and Engineering
The Graduate School
Seoul National University

With the increase of mobile users, there have been many privacy issues that sensitive and personal data is leaked while using mobile applications. To deal with this problem, Google App Store has required developers to disclose how the app uses data in privacy protection policies. App developers describe all of the app practices in the privacy policy to meet these legal requirements. However, there is no technical solution to verify the consistency between privacy policies and app activities. Users must rely on privacy protection policies to see how the app uses data. In this paper, we select personal data categories that can be easily exposed through the app and analyze privacy policies and apps to find keywords and APIs related to personal data. We collected the APK files and metadata of 13,223 apps registered in Google Play Store as datasets and select the apps for analysis by preprocessing them by four conditions. According to the results, many apps can access more personal data than they disclosed in the privacy policy.

Keywords: Privacy Policy, Mobile Privacy, App Analysis

Student Number: 2019-27143

Contents

List of Figures

Chapter 1

Introduction

A mobile device is an essential part of our daily lives. As smartphones become popular, most people use the internet on their smartphones and mobile applications for their convenience. However, with the increase in mobile users, there have been many privacy problems.

Like the case of location leakage [1], mobile apps can leak personal and sensitive user data to third parties. So, mobile users and regulators who enforce privacy laws get attention on privacy issues in the mobile environment.

They are interested in privacy issues because mobile devices have more sensitive and personal user information. The data collected by mobile apps, such as contacts and device location information, is more sensitive than the data collected by the web. This information can be used to distinguish an individual's context.

Thus, Google has established a user data policy [2] to ensure data privacy. It limits the use of data for public use and forces app developers to explain the app's practices to the privacy policies when registering apps on the Google Play Store.

Under the policy, developers should disclose what data they can access, how their apps collect and share the user data. Many strict regulations have also been enacted for data protection, such as California Consumer Privacy Act (CCPA) [3], General Data Protection Regulation (GDPR) [4], Children's Online Privacy Protection Act (COPPA) [5] and Personal Information Protection and Electronic Documents Act (PIPEDA) [6]. App developers describe all of the app's activities in the privacy policy to meet these legal requirements.

However, there is no technical solution to ensure that developers specify all of the app's activities in the privacy policy. The privacy policy is the only way to know how the app collects user data. Even if privacy policies and actual app practices are not consistent, users have to rely on the privacy policy to check the app's activities.

To address this issue, previous studies focused on preventing excessive permission requests, and the latest android apps request only the necessary permission related to the data they access [7, 8, 9]. Android apps also request permissions at runtime, allowing users to allow or deny permissions in person. Despite these works, it is easy for app developers to create apps and ask users for permission. They can quickly develop malicious apps and request additional permissions to access personal information. To provide users with clear information on personal data, we need to check the consistency between the privacy policy and the app practice.

In this study, we present a system to ensure privacy consistency while using the app. First, We collect Android Application Package (APK) files and privacy policies of apps in the Google Play Store, originally known as the Android Market. We create a privacy table listing keywords and API lists related to personal data

to perform policy and app analysis. The system conducts policy analysis to check the app practice described by the developers and app analysis to find API methods that access personal data. It compares the privacy table with a policy text and API methods in both analyses. Based on the analysis results, we determine an inconsistency between privacy policies and app practices.

We collected about 13K apps' APK files and privacy policies for an experiment by crawling app stores. After preprocessing the dataset according to four conditions, we choose the android apps for analysis. We analyzed privacy policies and APK files of selected apps and compared the results to check the consistency.

Our paper makes the following contributions:

- We collect dataset from Google Play Store, the official app store for android apps. The dataset contains 13K apps' privacy policies and APK files.
- We create a privacy table that includes keywords and API lists related to personal data. Even if API methods are changed or added, the system can conduct analysis only by adding it to the privacy table.
- We conduct policy and app analysis using the dataset. By comparing both results, we demonstrate that mobile apps can access more personal data than expected.

The rest of this paper is as follows. In Chapter 2, we review the previous works related to policy analysis and app analysis. The system design is described in Chapter 3, and the results are shown in Chapter 4. We conclude our study in Chapter 5.

Chapter 2

Related work

In this section, we summarize prior works on analyzing privacy policies and android apps.

2.1 Privacy Policy Analysis

A privacy policy is an important resource for internet users to check the web or mobile app's privacy practices. However, it is time-consuming to read all of the policy text in the privacy policy [10]. As a way to respond to this challenge, several studies have focused on analyzing privacy policies automatically. Ramanath et al. [11] apply unsupervised hidden markov models, and Liu et al. [12] analyzed the structure of privacy policy by identifying policy sections relating to different topics. Sathyendra et al. [13] provide labels on opt-out choices in privacy policies. Libert [14] extracts policy text from a web page and finds that it is not explicitly disclosed which the third party collects the user data in privacy policies. Various studies focus on creating a policy corpus. Wilson et al. [15] and Zimmeck et al. [16] manually

label privacy policies and create policy corpus.

2.2 Mobile App Analysis

There are two main approaches to analyze mobile apps: static analysis and dynamic analysis. Static analysis is a method of evaluating mobile apps by checking the apps' source code. This technique does not run the app directly but suggests that a violation is possible if the code is actually executed.

Prior studies conduct static analysis to examine app permission and related system calls [17, 18]. Yu et al. [19] develop a system that conducts static code analysis on its dex file to determine collected and retained information. Many potential privacy leaks of Android apps are discovered by static analysis on information flows [20, 21].

In contrast, dynamic analysis is performed at runtime, examining app methods and network traffic to track the use of personal and sensitive data. Enck et al. [22] suggest TaintDroid, a dynamic taint tracking and analysis system to find potential misuse of the user information. Abbas et al. [23] develop Haystack (now Lumen) to monitor traffic from apps. Ren et al. [24] study privacy leak using Lumen and UI monkey generator. Reyes et al. [25] reveal that many Android apps collect persistent device identifiers to track users, which is not allowed for advertising purposes.

Chapter 3

System Design

To analyze privacy policy and mobile apps for a consistency check, we must address several issues. It is challenging to collect datasets for analysis because there are various policy URLs and no official tool for downloading APK files. We need to consider how to compare the analysis results to examine the consistency between policy and apps. In this section, we introduce the overall structure of the system. First, we present an overview of the system and explain how our system overcomes these problems to create datasets. We also describe how we analyze privacy policy and apps to identify keywords and API methods related to personal data.

3.1 Overview

Figure 3.1 illustrates the system design which includes three major components, playstore crawler, policy analyzer and app analyzer.

We made a privacy table for listing personal data in advance. The privacy table contains policy keywords and API methods corresponding to each personal data.

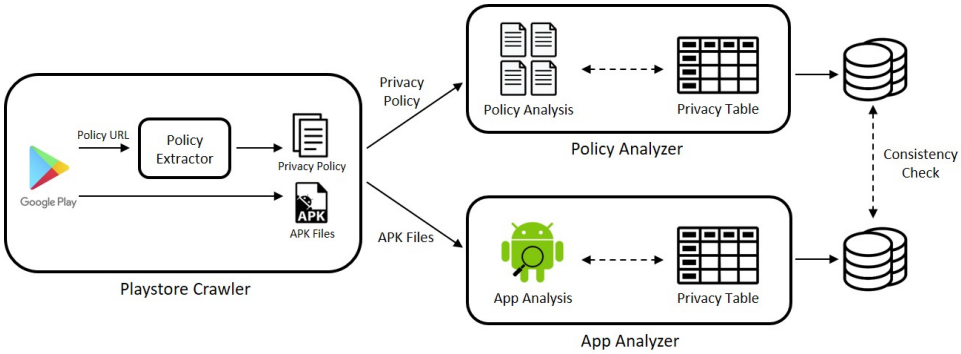


Figure 3.1: System Architecture

We selected categories of personal data based on Google's user data policy [2] and other studies [26], [24]. For policy analysis, we found policy keywords by checking the policy text of 50 popular apps. For app analysis, we searched the API lists that collect personal data based on android documentation.

In playstore crawler(§ 3.2), we collect APK files and privacy policies by rendering a policy URL of metadata. Policy analyzer (§ 3.3) evaluates whether the privacy policy has keywords related to personal data and returns the result. App analyzer (§ 3.4) decompiles the APK files to get the apps' API methods that access personal data and returns the result. We compare both results to check the app's consistency.

3.2 Playstore Crawler

Playstore Crawler collects privacy policies and APK files from the app store for analysis. The Google Play Store, originally known as the Android Market, is an online store that contains mobile applications, books, music, movies and magazines. These contents are downloadable from devices running the Android operating sys-

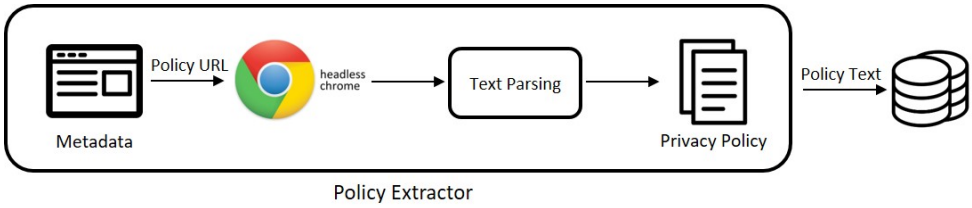


Figure 3.2: Policy Extracting Phase

tem.

However, the app store does not provide an official program to download the APK files. The app's privacy policy is also indicated on different pages, making it difficult to collect them. We collect metadata and apps' APK files using open source programs[1] and extract privacy policy by utilizing the metadata for collecting datasets at once. The program randomly crawls for popular apps listed in specific app genres. It downloads apps' APK files and stores metadata such as app title, package name, policy URL, app genre in JSON form.

The app store page contains only the privacy policy URL, and each app has a different URL format. It is challenging to extract policy text from each URL. The system creates a policy extractor for extracting the privacy policy using the policy URL. If an app's page contains a privacy policy URL, the policy extractor collects and stores privacy policies in the database. Figure 3.2 illustrates policy extracting phase.The chrome headless browser renders each privacy policy page and extracts only policy text using Readability, Node.js library used for Firefox reader view. We check some exceptional cases, such as Readability not working or the URL is invalid. For each case, we specify an error code and extract policy text in other ways.

[1] `https://github.com/alessandrodd/playstore_crawler`

- **Error 0 (Success):** If the privacy URL is a file URL link, we download the file and extract each file's policy text. If readability is not working, we save the entire page as a PDF file and parse the main content.

- **Error 1 (No policy URL):** If there is no policy URL, we cannot extract the privacy policy text.

- **Error 2 (Not vaild URL):** If a policy URL is expired or the browser gets access denied, timeout error while loading page, the URL is not valid.

As mentioned above, we extract privacy policy text using the policy extractor and store it in a database.

The crawler collects APK files and privacy policies for each app. But it is inefficient to analyze all mobile apps because some apps are less likely to leak information or cannot conduct policy analysis. Therefore, we preprocess the apps with four conditions.

- **Condition 1 (Internet permission):** Without internet permission, no data is leaked from the app. If the app does not have internet permission, we exclude it from a dataset.

- **Condition 2 (Policy URL):** If there is no policy URL, we cannot conduct policy analysis.

- **Condition 3 (Language check):** We exclude non-English text from a dataset. Python langdetect library is used to check the language of privacy policies.

- **Condition 4 (Sanity check):** To check the sanity of data, we select the

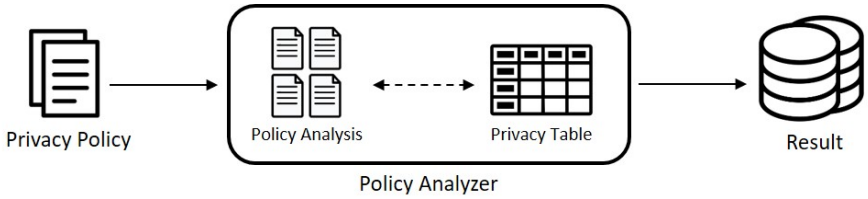


Figure 3.3: Policy Analyzing Phase

required keywords that privacy policies should have and compare them with policy text.

After preprocessing, we analyze privacy policies and APK files of the selected app.

3.3 Policy Analyzer

User privacy policy describes how app developers can comply with privacy requirements. App providers can handle user data for user convenience, but other purposes, such as targeted advertising or marketing. So, developers disclose how their apps access, collect and utilize user data when they register apps in the App Store. In particular, they should state app practices if they deal with personal and sensitive data to identify a user's behavioral characteristics.

Policy analyzer is a module to check the app practice described by the developers. Figure 3.3 illustrates the overall function of this module. This module analyzes the privacy policy extracted from the crawler and compares it with the privacy table. The analyzer store the result in the database.

There are a lot of personal and sensitive user data that can be collected by apps. Specifically, we selected personal data that mainly used to identify users. We found a policy keyword representing the personal data and wrote the keywords in the privacy table. The policy analyzer uses the table for policy analysis. If a

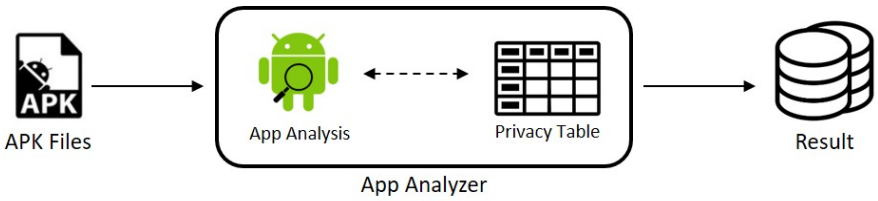


Figure 3.4: App Analyzing Phase

policy keyword exists, there is a description related to personal data in the privacy policy. For each app, the analyzer compares each keyword with policy text stores the analysis results. If a policy text matches policy keywords, we consider that the privacy policy explains app practice related to personal data. If no keywords were present, we decide there is no explanation about the personal data. The analyzer records the results of each app in the database and they are used for consistency check.

3.4 App Analyzer

The official Android documentation describes the Android API's class and public method. We can check what data each API can collect. We investigated the API list calling each personal data and wrote all API lists related to personal data categories.

App analyzer is based on Androguard [27], an open-source static analysis tool written in Python. App analyzer decompiles APK files to check the app's all API list and compares them with the privacy table to see whether the app can collect personal data. If an app's API corresponds to the API list in the privacy table, we consider the app can collect personal data related to the API list. The analyzer conducts this analysis to all APK files and saves the result in the database. We compare these results with the policy analyzer's results to check the consistency

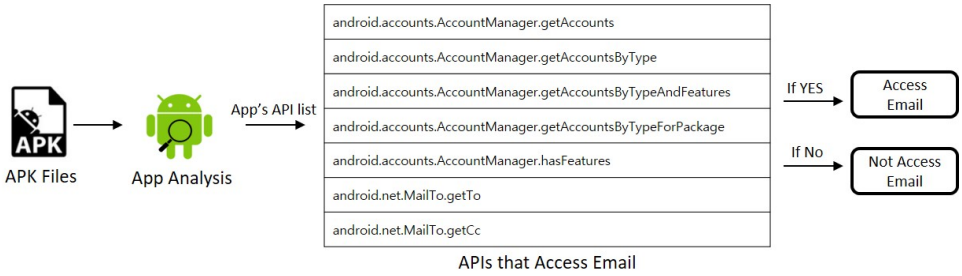


Figure 3.5: Example of App Analysis

between the privacy policy and app practice.

Figure 3.4 illustrates app analyzing phase and figure 3.5 is an example of app analysis. First, the system analyzes apps using Androguard and gets the apps' API. The analyzer compares it with the API list of privacy table to check if the app contains APIs that access personal data and returns the result.

Chapter 4

Result

4.1 Dataset

The playstore crawler collected 13,223 android apps' APK files and metadata in Google Play Store. We preprocessed the collected apps based on their metadata and selected apps for analysis. As shown in Figure 4.1, 56% of apps are selected to be analyzed. The detailed numbers for each condition are as follows:

- **Condition 1: Internet Permission** (13,223 → 13,058)
- **Condition 2: Policy URL** (13,058 → 12,286)
- **Condition 3: English Text** (12,286 → 7,981)
- **Condition 4: Policy Keyword** (7,981 → 7,401)

Before preprocessing, we checked the apps' permission list from the metadata on Google Play. Of the 13,223 apps, 165 apps do not have internet permission. We selected 13,058 apps except for these apps in condition 1. Of 13,058 apps, we chose

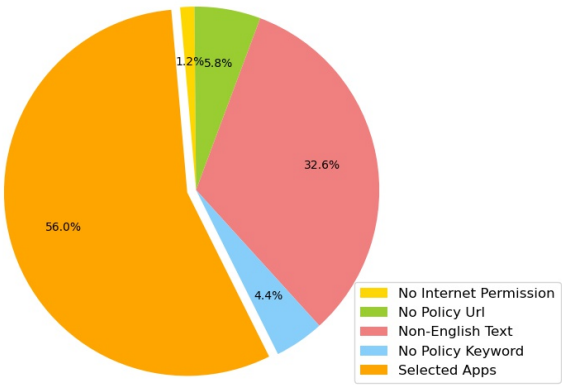


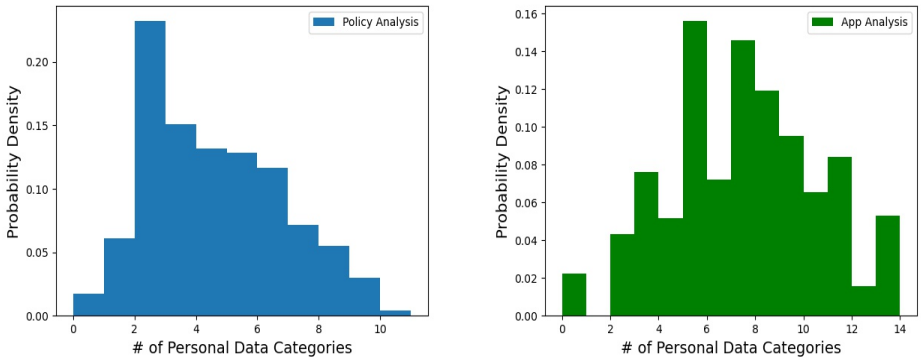
Figure 4.1: The Percent Value of Preprocessed Apps

12,286 apps except for 772 apps without privacy policy URL in condition 2. Of the 12,286 apps, we selected 7,981 apps except for the non-English policy in condition 3. We used a Python Langdetect library to detect the language of privacy policies. Of the 7,981 apps, we adopted 7,401 apps except for those without policy keywords in condition 4. The system conducted policy and app analysis on preprocessed apps.

4.2 Personal Data Category

In this section, we analyze the personal data categories of each results in detail.

We first identified the number of personal data categories in policy analysis and app analysis results. Figure 4.2 shows the number of personal data categories in a histogram. Figure 4.2a shows how many personal data categories are mentioned in the policy, and figure 4.2b presents how many personal data categories are related to the app's API.



(a) Policy Analysis (b) App Analysis

Figure 4.2: Histogram of Personal Data Categories

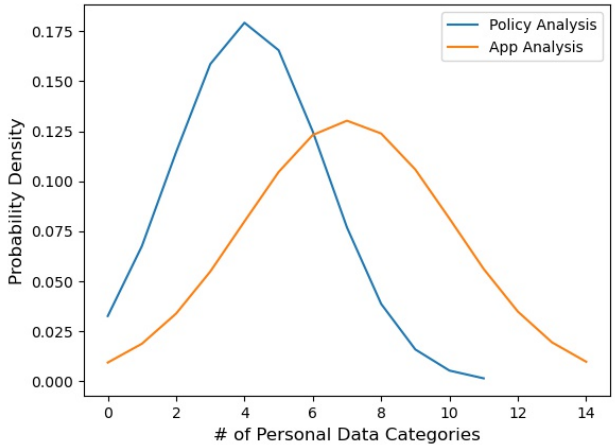
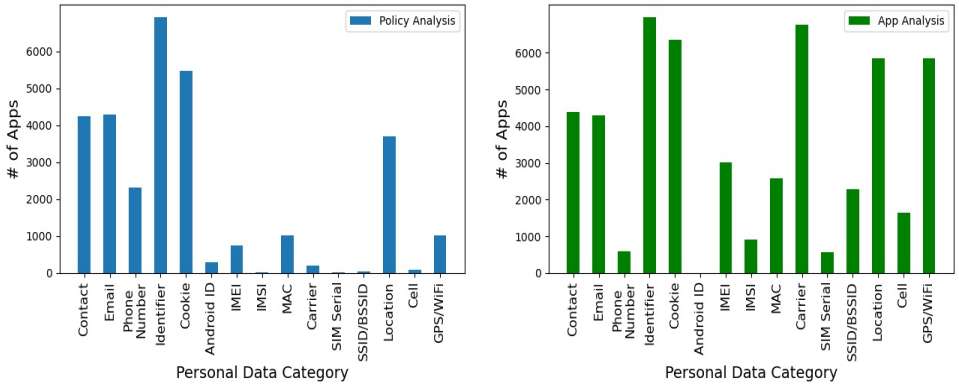


Figure 4.3: PDF of Personal Data Categories
(Policy and App Analysis)

Figure 4.3 illustrates the Probability Density Function (PDF) graphs of both analysis. By looking at both graphs, app analysis results include more personal data categories. We concluded that a mobile app can access more personal data than expected.



(a) Policy Analysis (b) App Analysis

Figure 4.4: Number of Apps for Each Category

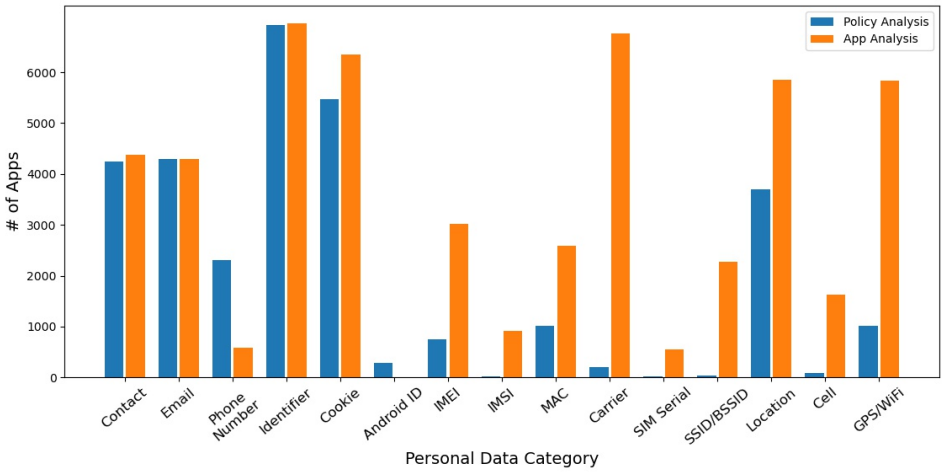


Figure 4.5: Number of Apps for Each Category
(Policy and App Analysis)

Figure 4.4 illustrates how many apps related to each personal data category. We combined both results in figure 4.5. The graph shows that there is a big difference in a particular category.

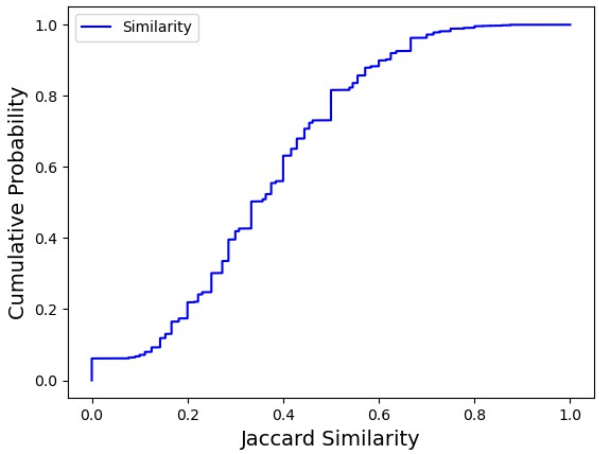


Figure 4.6: Jaccard Similarity between Policy and App Analysis

4.3 Consistency Check

We conducted various analyses to examine the consistency between privacy policy and app practice.

First, we compared Jaccard similarity for consistency check. Jaccard similarity is one of a statistic used to measure the similarity and diversity of sample sets. The Jaccard index has a value between 0 and 1. If both sets are equal, it has a value of 1, and if there is no common element, it has a value of 0. The following expression defines the Jaccard similarity.

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \tag{4.1}$$

The Jaccard similarity indicates the relationship between policy and app practice. Figure 4.6 is a Cumulative Density Function (CDF) graph of Jaccard similarity. We found that similarities were widely distributed from 0.2 to 0.6 and some cases were completely different.

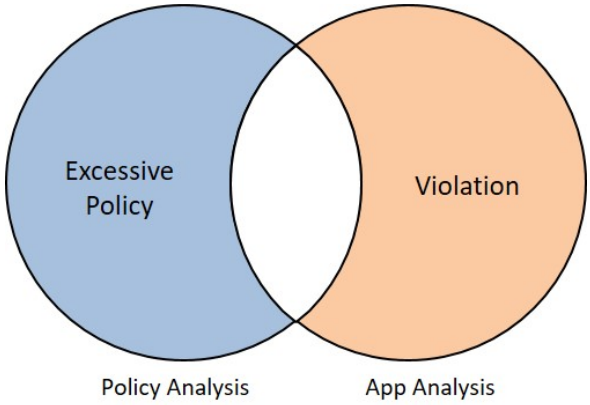


Figure 4.7: Definition of Excessive Policy and Violation

Next, we checked excessive policy and violation. Figure 4.7 shows the excessive policy and violation as defined in our paper. If a personal data category is in a policy analysis result but not in an app analysis result, we define it as excessive policy. If a personal data category is in an app analysis result but not in a policy analysis result, we define it as a violation.

Figure 4.8 is CDF of excessive policy and violation. We can see that the value of violation increases more slowly by comparing both CDF graphs. It means the app has more violations than excessive policies.

Figure 4.9 shows the number of apps by excessive policy and violation count. By analyzing these bar graphs, we concluded that there were more violations than excessive policy overall.

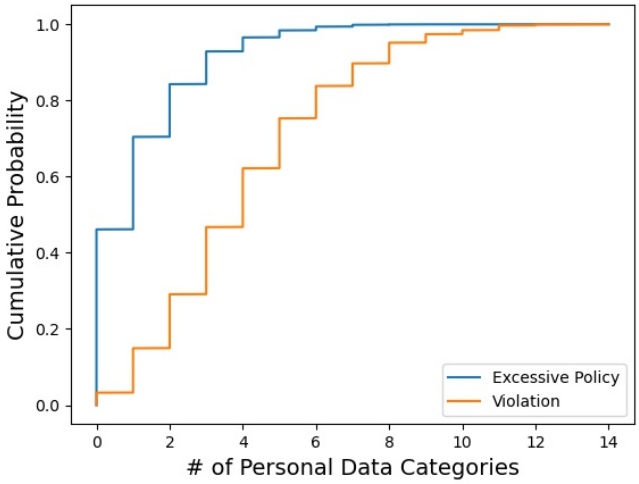
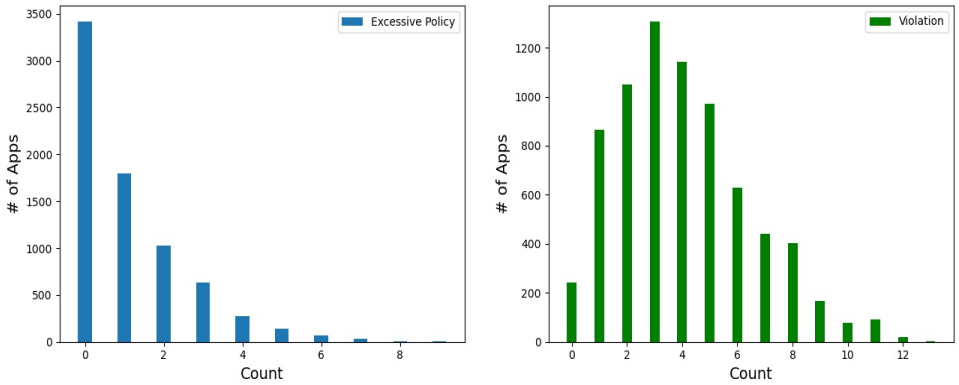


Figure 4.8: CDF of Excessive Policy and Violation



(a) Excessive Policy (b) Violation

Figure 4.9: Number of Apps by Excessive Policy and Violation Count

We also looked at the number of violations for each category to identify which personal data can easily overused. According to the Figure 4.10, carrier and GPS/WiFi information had the largest number of violations.

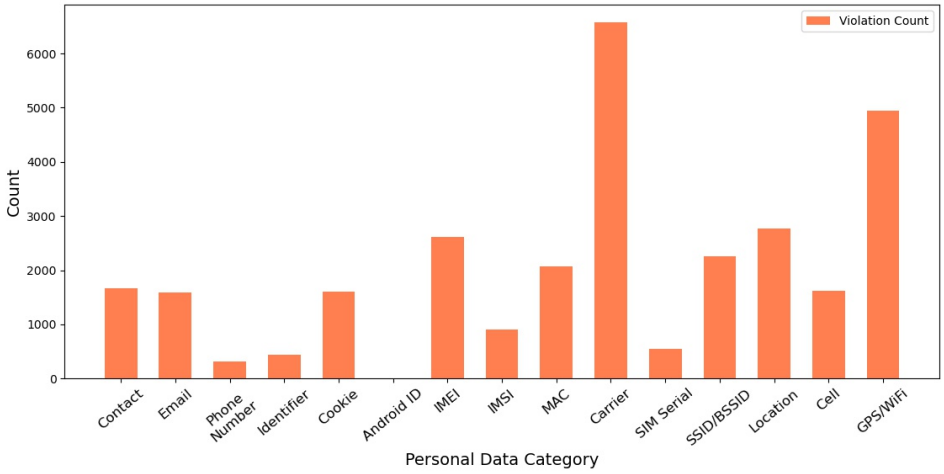


Figure 4.10: Violation Count by Personal Data Category

4.4 App Genre

In this section, we study the analysis results by app genre. To compare datasets, we analyzed the number of apps by genre. Figure 4.11 shows the number of apps by genre. We selected genres with more than 100 apps by genre and used them for analysis.

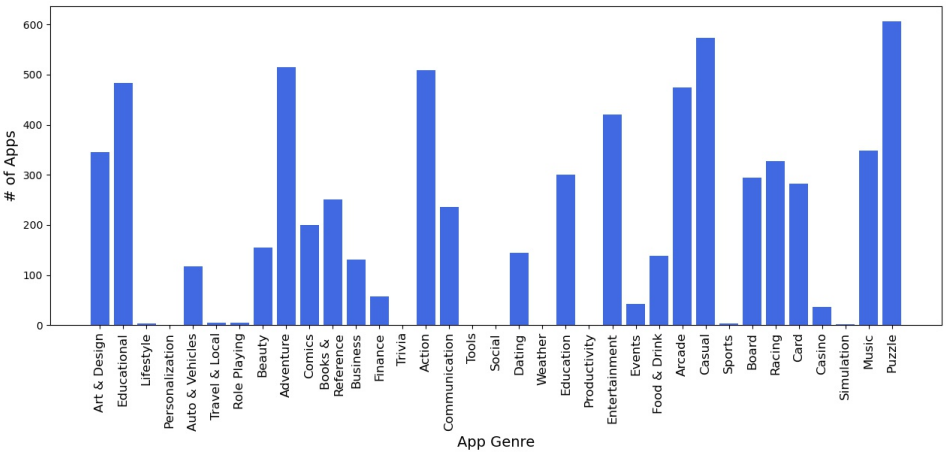
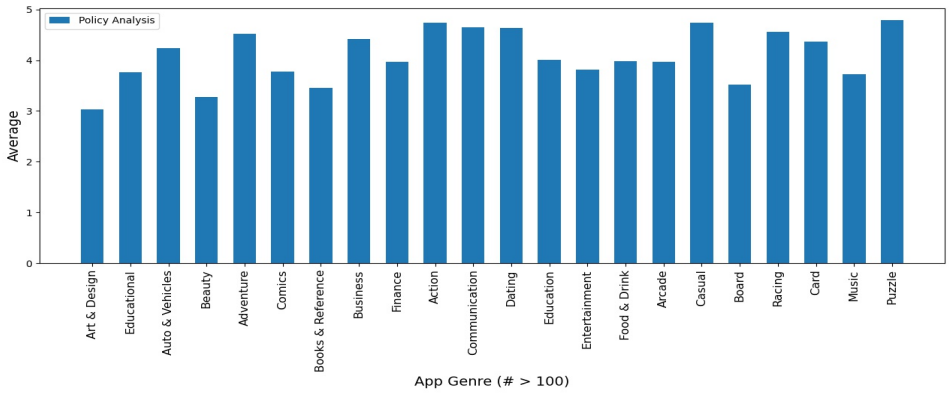
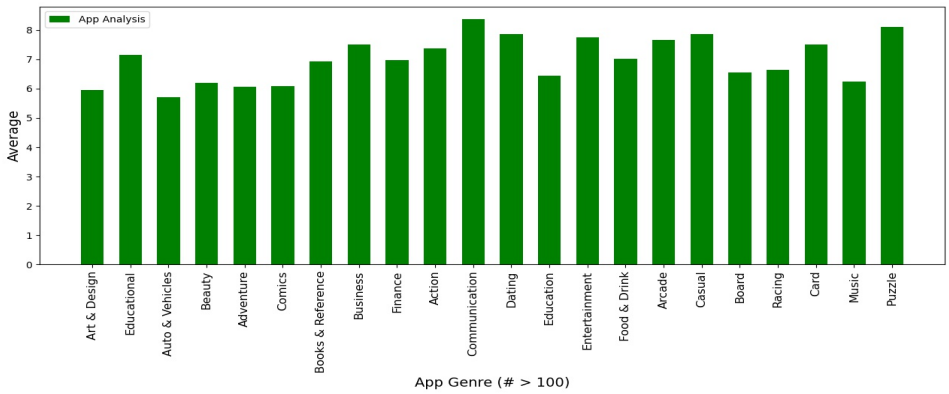


Figure 4.11: Number of Apps by genre



(a) Policy Analysis



(b) App Analysis

Figure 4.12: Average of Personal Data Categories by Genre

Figure 4.12 illustrates the average number of personal data categories by app genre. It shows that the average value is different by genres in both analysis results. We combined these graphs to analyze precisely in figure 4.13. According to this result, we can prove that each app genre has a different average value, and there is a big difference in personalization, tools, and card genre.

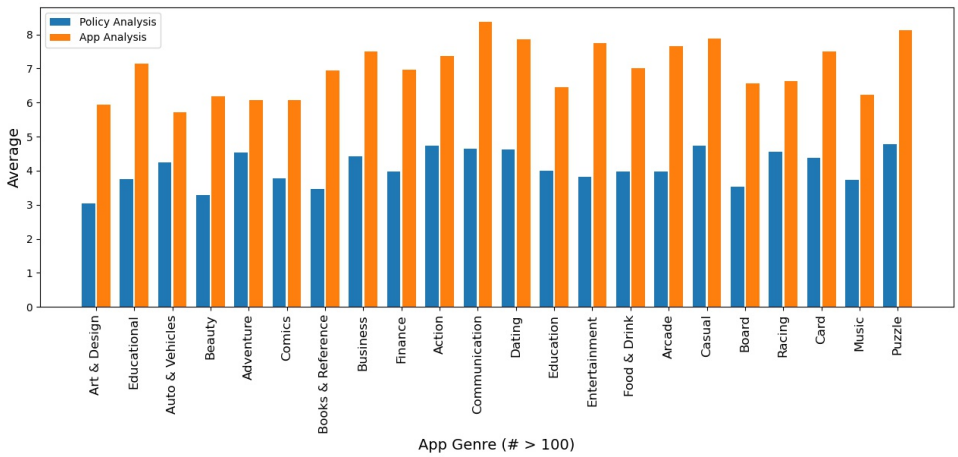


Figure 4.13: Average of Personal Data Categories by Genre (Policy and App Analysis)

Chapter 5

Conclusion

We presented a comprehensive system that analyzes the privacy policy and app practices to check the consistency. We created privacy table to analyze the privacy policies and android methods regarding sensitive data. To test our analysis module, the system crawled play store to collect APK files and policy text. We analyzed privacy policy text to identify which data is informed to be collected. As an app analysis, we analyzed app source code to identify whether the app has API that return personal data.

Policy regulators require developers to mention app activities in privacy policy. While privacy policies are intended to disclose applicable privacy practices, they are often incongruous with the actual practices performed.

This system conduct analysis of policy text and app source code. It pairs the policy analysis with app analysis of mobile apps to identify potential inconsistency between privacy policy and app practice. The system has the potential to improve privacy transparency and enhance privacy levels overall.

Bibliography

[1] "Secret-sharing app whisper left users' locations, fetishes exposed on the web." https://www.washingtonpost.com/technology/2020/03/10/secret-sharing-app-whisper-left-users-locations-fetishes-exposed-web/. Accessed: 2020-12-15.

[2] "User data policy." https://support.google.com/googleplay/android-developer/answer/10144311. Accessed: 2020-12-15.

[3] "California consumer privacy act (ccpa)." https://oag.ca.gov/privacy/ccpa. Accessed: 2020-12-15.

[4] "General data protection regulation (gdpr)." https://gdpr-info.eu/. Accessed: 2020-12-15.

[5] "Children's online privacy protection act (coppa)." https://www.ftc.gov/enforcement/rules/rulemaking-regulatory-reform-proceedings/childrens-online-privacy-protection-rule. Accessed: 2020-12-15.

[6] "Protection and electronic documents act (pipeda)." https://www.priv.gc.ca/en/privacy-topics/privacy-laws-in-canada/the-personal-

information-protection-and-electronic-documents-act-pipeda/. Accessed: 2020-12-15.

[7] A. P. Felt, E. Chin, S. Hanna, D. Song, and D. Wagner, "Android permissions demystified," in *Proceedings of the 18th ACM conference on Computer and communications security*, pp. 627–638, 2011.

[8] K. W. Y. Au, Y. F. Zhou, Z. Huang, and D. Lie, "Pscout: analyzing the android permission specification," in *Proceedings of the 2012 ACM conference on Computer and communications security*, pp. 217–228, 2012.

[9] D. Arp, M. Spreitzenbarth, M. Hubner, H. Gascon, K. Rieck, and C. Siemens, "Drebin: Effective and explainable detection of android malware in your pocket.," in *Ndss*, vol. 14, pp. 23–26, 2014.

[10] A. M. McDonald and L. F. Cranor, "The cost of reading privacy policies," *Isjlp*, vol. 4, p. 543, 2008.

[11] R. Ramanath, F. Liu, N. Sadeh, and N. A. Smith, "Unsupervised alignment of privacy policies using hidden markov models," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 605–610, 2014.

[12] F. Liu, S. Wilson, P. Story, S. Zimmeck, and N. Sadeh, "Towards automatic classification of privacy policy text," *School of Computer Science Carnegie Mellon University, Pittsburgh, PA, Tech. Rep. CMU-ISR-17-118R and CMULTI-17-010*, 2018.

[13] K. M. Sathyendra, S. Wilson, F. Schaub, S. Zimmeck, and N. Sadeh, "Identifying the provision of choices in privacy policy text," in *Proceedings of the 2017*

Conference on Empirical Methods in Natural Language Processing, pp. 2774–2779, 2017.

[14] T. Libert, "An automated approach to auditing disclosure of third-party data collection in website privacy policies," in *Proceedings of the 2018 World Wide Web Conference*, pp. 207–216, 2018.

[15] S. Wilson, F. Schaub, A. A. Dara, F. Liu, S. Cherivirala, P. G. Leon, M. S. Andersen, S. Zimmeck, K. M. Sathyendra, N. C. Russell, *et al.*, "The creation and analysis of a website privacy policy corpus," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1330–1340, 2016.

[16] S. Zimmeck, P. Story, D. Smullen, A. Ravichander, Z. Wang, J. Reidenberg, N. C. Russell, and N. Sadeh, "Maps: Scaling privacy compliance analysis to a million apps," *Proceedings on Privacy Enhancing Technologies*, vol. 2019, no. 3, pp. 66–86, 2019.

[17] M. Backes, S. Bugiel, and E. Derr, "Reliable third-party library detection in android and its security applications," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pp. 356–367, 2016.

[18] I. Leontiadis, C. Efstratiou, M. Picone, and C. Mascolo, "Don't kill my ads! balancing privacy in an ad-supported mobile application market," in *Proceedings of the Twelfth Workshop on Mobile Computing Systems & Applications*, pp. 1–6, 2012.

[19] L. Yu, X. Luo, X. Liu, and T. Zhang, "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, IEEE, 2016.

[20] M. I. Gordon, D. Kim, J. H. Perkins, L. Gilham, N. Nguyen, and M. C. Rinard, "Information flow analysis of android applications in droidsafe.," in *NDSS*, vol. 15, p. 110, 2015.

[21] F. Wei, S. Roy, and X. Ou, "Amandroid: A precise and general inter-component data flow analysis framework for security vetting of android apps," in *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*, pp. 1329–1341, 2014.

[22] W. Enck, P. Gilbert, S. Han, V. Tendulkar, B.-G. Chun, L. P. Cox, J. Jung, P. McDaniel, and A. N. Sheth, "Taintdroid: an information-flow tracking system for realtime privacy monitoring on smartphones," *ACM Transactions on Computer Systems (TOCS)*, vol. 32, no. 2, pp. 1–29, 2014.

[23] A. Razaghpanah, N. Vallina-Rodriguez, S. Sundaresan, C. Kreibich, P. Gill, M. Allman, and V. Paxson, "Haystack: A multi-purpose mobile vantage point in user space," *arXiv preprint arXiv:1510.01419*, 2015.

[24] J. Ren, M. Lindorfer, D. J. Dubois, A. Rao, D. Choffnes, and N. Vallina-Rodriguez, "Bug fixes, improvements,... and privacy leaks: A longitudinal study of pii leaks across android app versions," in *Network and Distributed System Security Symposium*, Internet Society, 2018.

[25] I. Reyes, P. Wijesekera, J. Reardon, A. E. B. On, A. Razaghpanah, N. Vallina-Rodriguez, and S. Egelman, ""won't somebody think of the children?" exam-

ining coppa compliance at scale," *Proceedings on Privacy Enhancing Technologies*, vol. 2018, no. 3, pp. 63–83, 2018.

[26] Y. Song and U. Hengartner, "Privacyguard: A vpn-based platform to detect information leakage on android devices," in *Proceedings of the 5th Annual ACM CCS Workshop on Security and Privacy in Smartphones and Mobile Devices*, pp. 15–26, 2015.

[27] "Androguard documentation." `https://androguard.readthedocs.io/`. Accessed: 2020-12-15.

초 록

모바일 사용자가 증가함에 따라, 모바일 어플리케이션을 사용하는 동안 민감한 개인정보가 유출되는 프라이버시 문제가 많아졌다. 이를 해결하기 위해 구글 앱스토어에서는 개발자들이 앱이 데이터를 어떻게 활용하는지 개인정보 보호 정책에 공개하도록 했다. 앱 제공자들은 법적인 요구사항을 만족하기 위해 개인정보 보호 정책에 앱의 활동을 명시하고 있다. 하지만 개인정보 보호 정책과 앱 활동의 일관성을 확인할 수 있는 기술적인 해결책이 없으며, 사용자는 앱이 데이터를 어떻게 활용하는지 알기 위해 개인정보 보호 정책에 의존해야만 한다. 이 논문에서는 앱을 통해 쉽게 유출될 수 있는 개인정보 목록을 선정하고 개인정보 보호 정책과 앱을 분석해 민감한 정보와 관련있는 키워드 및 API를 찾아 그 결과를 비교한다. 데이터셋으로 구글 플레이 스토어에 등록된 13,223개 앱의 패키지 파일과 부가정보를 수집했고, 이를 전처리하여 실험 대상이 되는 앱을 선정했다. 우리는 실험 결과를 바탕으로 모바일 앱이 프라이버시 보호 정책에 명시된 것보다 더 많은 개인정보에 접근할 수 있음을 입증한다.

주요어: 개인정보 보호 정책, 모바일 프라이버시, 앱 분석

학번: 2019-27143