# EVALUATION OF PRIVACY POLICIES IN MOBILE APPLICATIONS

**BY**

**EZIOKWU MICHAEL CHIBUIKE**

**PSC1813820**

**DEPARTMENT OF COMPUTER SCIENCE**

**FACULTY OF PHYSICAL SCIENCES**

**UNIVERSITY OF BENIN**

**BENIN CITY**

**EDO STATE, NIGERIA.**

**JANUARY, 2026**

# EVALUATION OF PRIVACY POLICIES IN MOBILE APPLICATIONS

**BY**

**EZIOKWU MICHAEL CHIBUIKE**

**PSC1813820**

**A PROJECT REPORT SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE, FACULTY OF PHYSICAL SCIENCE, UNIVERSITY OF BENIN, BENIN CITY**

**IN PARTIAL FULFILMENT OF THE REQUIREMENT FOR THE AWARD OF A BACHELOR OF SCIENCE (B.Sc.) DEGREE IN COMPUTER SCIENCE**

**JANUARY, 2026**

## CERTIFICATION

This is to certify that this project work was carried out by **EZIOKWU MICHAEL CHIBUIKE** with Matriculation Number **PSC1813820** under my supervision. It is adequate and satisfactory, both in scope and content, for the award of Bachelor of Science (B.Sc) Degree in Computer Science of the University of Benin

_____ _____

**Prof. F.O. Chete** **DATE**

Project Supervisor

## APPROVAL

This project work is hereby approved in partial fulfilment of the requirements for the award of Bachelor of Science (B.Sc.) Degree in Computer Science from the University of Benin

_____ _____

**DR. (MRS). USIOBAIFO A. R.** **DATE**

Head of Department

## DEDICATION

I dedicate this work to God, for giving me the strength and guidance to properly carry out and complete the work and also for his protection throughout my time at the University of Benin.

This work is also dedicated to my parents, for making this journey as possibly easy as they could, for encouraging me, and for guiding me

## ACKNOWLEDGEMENT

My utmost acknowledgement goes to God, for seeing me through all my years in school.

I would like to express my profound gratitude to my project supervisor, Prof. F.O. Chete for his consistent guidance throughout the period of this project.

Special gratitude goes to the Head of the Department of Computer Science, Dr. (Mrs). R.A. Usiobaifo and all the other lecturers in the Department of Computer Science, University of Benin, for their support and inspiration in one way or the other: Prof (Mrs.) S.C Chiemeke, Prof (Mrs.) V.V.N Akwukwuma, Prof E.A Onibere (rtd.), Prof. (Mrs.) F. Egbokhare, Prof. (Mrs.) A.O. Egwali, Prof. G.O. Ekuobase, Prof. F.I Amadin, Prof. K.C. Ukaoha, Prof. (Mrs.) S. Konyeha, Prof. A. Imiavan, Prof. (Mrs.). V. Osubor, Engr F.A.U Imouokhome, Late Dr. S.S. Daodu, Dr. Aladeselu (rtd.), F.O. Oliha Ph.D., Dr. E. Nwelih, Dr. (Mrs.) Aziken, Dr. F.O. Chete, Dr. (Mrs.) A.R. Usiobaifo, Dr. (Mrs.) R.O. Osaseri, Dr. J.C. Obi, Mr. P. Imiefoh, Mr. E.E. Obasohan, Mr. S.O.P. Oliomogbe, Mr. Odetayo, Mr. K.O. Otokiti, Mr. N.E.O. Agbonlahor, Mrs. M. Iyawe, Mr. O.O Igene, Mrs. E.P. Ebietomere, Mr. E.C. Igodan, Mr. I.E. Obayagbona, Mrs. L. Usiosofe, Mrs. T. Agenmomen, Mr. Idehen, Mrs. Izebizuwa. You have all worked hard to set me on the proper road in my professional pursuit and to implant in me sound knowledge about important parts of life.

Special thanks goes to my lovely parents Mr. and Mrs. Eziokwu Aselem for thier love and encouragement throughout my stay in the University of Benin. And also to my lovely siblings, Eziokwu Peter, Eziokwu Victory and Eziokwu Anastasia for always being there for me.

I want to also appreciate my friends Imeh John, Robert Albert, Sefa Michael, Shegun Michael. I want to say God bless you all and I love you.

## TABLE OF CONTENTS

## ABSTRACT

This project is to evaluate privacy policies in mobile applications. It revealed Assess the clarity, readability, and structure of privacy policies used in selected mobile applications, evaluate the consistency between the stated privacy policies and the actual data handling practices of the mobile applications, identify specific areas within mobile privacy policies where vague or misleading terms are commonly used, examine the extent to which user input data is collected, processed, and shared without clear disclosure in the privacy statements, use Python-based tools to automate the detection and analysis of discrepancies between privacy policies and app behaviours, recommend practical improvements for making mobile application privacy policies more transparent, accurate, and user-friendly.

This study designed and evaluation approach to examine how mobile applications present and apply their privacy policies. Selected mobile apps were reviewed based on their popularity and

relevance to everyday users. Their privacy policies were extracted and assessed for clarity, length, and language. Python scripts were then used to carry out static and dynamic analysis on these apps. The static part inspected permissions and data access points declared within the app files, while the dynamic part monitored how the app behaves when in use, especially in handling user data. Any mismatch between what is written in the privacy policy and what the app does will be recorded and analysed. Focus was also be placed on how user input data is managed, as this is often not clearly addressed in policy statements. Results were compared across apps from different categories to detect patterns or risks that repeat across multiple apps

# CHAPTER ONE

# INTRODUCTION

## 1.1 Background of the Study

Mobile applications have become indispensable tools in daily life, offering services that range from communication and navigation to finance and health. Their widespread adoption has introduced a growing need to address how these apps manage user privacy. The field of digital privacy, particularly within mobile applications, continues to draw critical attention due to ongoing concerns about data misuse, lack of transparency, and inadequate regulation. With millions of apps available and billions of downloads globally, understanding how personal data is handled is now more important than ever. Individuals frequently share sensitive information on mobile platforms without clearly understanding what is collected, how it is processed, or who it is shared with (Kandil et al., 2018).

Privacy policies are intended to serve as the primary tool for informing users about data practices. Yet many mobile app privacy policies are either too vague, overly technical, or not even accessible to users. It has been observed that a considerable portion of mobile applications either fail to provide a functioning privacy policy or deliver one that lacks relevant information about data collection and security measures (Graves, 2015). In several cases, apps use generic statements that do not reflect actual data practices, thereby creating a gap between stated policy and real-time behaviour (Sun, 2018). Such inconsistencies limit the user's ability to make informed decisions and may result in unauthorised data sharing (Hashmi et al., 2021).

The complexity of language used in privacy policies has been identified as a critical barrier. Studies have shown that users struggle to read and understand these documents because they are often written in legal or technical jargon not suitable for the average literacy level (Singh, 2018).

This results in users bypassing these policies entirely and unknowingly consenting to data practices that they may otherwise reject. The issue becomes even more severe in mobile environments, where small screen sizes and navigation limitations further restrict readability and comprehension (Singh et al., 2011). Tools like AppAware have been developed to visualise privacy policy content and simplify permissions for better user comprehension, yet they are still not widely adopted by developers (Paspatis et al., 2018).

Another concern is the readability versus legal compliance trade-off. While developers may craft privacy policies to fulfil legal requirements, they often neglect how users interact with or interpret them. About 60% of Android app policies have been found to inaccurately describe data collection practices, with some relying on pre-written templates that do not align with the specific operations of the app (Sun, 2018). These templates frequently fail to include essential aspects such as third-party data sharing or retention policies, leaving users uninformed about critical privacy risks.

Longitudinal studies have also identified a rising trend in non-compliance over time, as newer versions of apps increasingly diverge from their stated privacy commitments. There is evidence that many mobile apps fail to update their policies to reflect changes in data handling behaviours, leading to undisclosed data practices and weakened user trust (Hashmi et al., 2021). In some cases, even the act of accessing a privacy policy page has been associated with third-party data sharing, contradicting the very premise of informed consent (Kollnig, 2021).

Attempts to address these issues include proposals for automatically generated privacy policies that use static code analysis and natural language processing to create accurate and readable policy statements (Yu et al., 2017). Such systems are designed to reduce discrepancies

between declared and actual data practices. Still, they require broad developer adoption and regulatory support to become standard.

Another area of concern is that users often lack control over their privacy preferences. Traditional privacy policies offer limited options for users to opt in or out of specific data practices. Research indicates that users favour visual and structured policy formats that allow for greater transparency and control (Kununka, 2019). These user-centric models provide a more interactive and understandable way to convey privacy terms and are shown to improve trust and engagement with privacy content.

The demand for more effective privacy policies in mobile apps is not merely a theoretical concern but a reflection of practical issues affecting user autonomy and data protection. Despite existing legal frameworks, many apps fall short of full transparency and compliance. The current state of privacy policy development lacks standardisation, and users are often left navigating unclear, inconsistent, and overly complex documents that offer little actionable insight. With digital interactions continuing to grow, the need for privacy policy evaluation frameworks that prioritise accuracy, transparency, and user comprehension becomes even more urgent.

## 1.2 Statement of the Problem

A major concern in the use of mobile applications is the lack of alignment between what privacy policies claim and how user data is actually handled. Many apps either fail to provide a privacy policy or offer vague, incomplete documents that do not describe real data practices in any detail (Pan et al., 2024). There is a clear gap in transparency, especially where sensitive data is involved, as many apps request access to personal information without adequately explaining what is collected, why it is needed, or who it is shared with (Wang et al., 2018). Privacy policies also often omit information about embedded third-party libraries, which can access and transmit user

data without the knowledge or consent of users (Zhao et al., 2023). In many cases, policies generated using automated tools contain inaccurate or overly generalised statements that do not reflect app behaviour, making enforcement and user trust difficult (Pan et al., 2024). This problem is not only technical but affects real users who unknowingly allow access to their personal data without informed consent. Educational institutions, financial platforms, healthcare services, and government mobile portals face serious risk when privacy policies are misleading or non-existent, especially when users depend on these platforms to manage sensitive records or official documentation.

### 1.3 Aim and Objectives of the Study

The aim of the study is to carry out an evaluation of privacy policies in mobile applications.

The objectives are;

1. Assess the clarity, readability, and structure of privacy policies used in selected mobile applications.
2. Evaluate the consistency between the stated privacy policies and the actual data handling practices of the mobile applications.
3. Identify specific areas within mobile privacy policies where vague or misleading terms are commonly used.
4. Examine the extent to which user input data is collected, processed, and shared without clear disclosure in the privacy statements.
5. Use Python-based tools to automate the detection and analysis of discrepancies between privacy policies and app behaviours.
6. Recommend practical improvements for making mobile application privacy policies more transparent, accurate, and user-friendly.

## 1.4 Methodology

This study designed and evaluation approach to examine how mobile applications present and apply their privacy policies. Selected mobile apps will be reviewed based on their popularity and relevance to everyday users. Their privacy policies were extracted and assessed for clarity, length, and language. Python scripts were then used to carry out static and dynamic analysis on these apps. The static part inspected permissions and data access points declared within the app files, while the dynamic part monitored how the app behaves when in use, especially in handling user data. Any mismatch between what is written in the privacy policy and what the app does will be recorded and analysed. Focus was also be placed on how user input data is managed, as this is often not clearly addressed in policy statements (Wang et al., 2018). Results were compared across apps from different categories to detect patterns or risks that repeat across multiple apps (Hashmi et al., 2021; Carlsson et al., 2022).

## 1.5 Scope of the Study

This study covers the evaluation of privacy policies used in selected mobile applications, with a focus on how clearly they present the use, collection, and sharing of user data. It examines the level of agreement between what the policies state and what the apps actually do during use. The work is limited to publicly available mobile apps and their visible behaviours, using Python to detect gaps and inconsistencies. It does not involve legal interpretation or encrypted data not accessible through open testing. The study focuses only on mobile applications and does not include web-based platforms or desktop software.

## 1.6 Significance of the Study

This study holds value for mobile application users, developers, policy makers, academic researchers, digital rights organisations, app-based service and data protection regulators. Each of

these groups plays a role in shaping or experiencing the reality of mobile data practices, and will gain from a clearer understanding of how privacy policies affect trust, transparency and usage behaviour. The findings of this work will help reveal where user protection is strong or weak and guide better approaches to digital privacy.

**Mobile application users:** They will benefit from improved awareness about the risks and limitations of the privacy policies they often accept without proper understanding. The study will show how much users actually know about what data is being collected and where their rights may be overlooked. This will encourage more informed decisions and help users become more active in protecting their data.

**Developers:** These will find insight on how policy content is perceived and where improvements are needed. Many policies fail to reflect what the application truly does with user data, which can affect user trust and legal safety (Zhao et al., 2023). Developers who use this study can learn how to write more accurate and readable policies that match the actual functions of their apps (Pan et al., 2023).

**Policy makers and regulators:** These will be able to use the outcomes of the research to strengthen rules that guide digital platforms. When app policies are misleading or incomplete, users are exposed to unfair data practices (Wang et al., 2018). This study supports better policy enforcement by showing where gaps exist and how they affect users.

**Academic researchers:** They can use the findings as a base to expand work in digital privacy, mobile communication and user behaviour studies. It offers a real setting for understanding how students interact with technology in everyday life.

**App-based service providers:** App-based service providers in health, education, commerce and finance will gain from knowing how privacy terms influence user choices. This will support better service delivery, improved data management and clearer communication with users.

**Digital rights organisations:** These will benefit from evidence that helps in campaigns for stronger user protection, especially for younger or less-informed digital users. This research gives data to support calls for fairer digital practices.

## 1.7 Limitations of the Study

This study is limited to analysing the privacy policies of selected mobile applications and comparing them with their actual data handling behaviour. It does not cover every category of mobile app or include all regional legal frameworks. The evaluation will focus only on publicly available policies and accessible data flows, which may exclude hidden processes or encrypted transmissions not detectable through testing. Use of Python tools may also face limitations in interpreting some technical formats or embedded scripts. These constraints may affect how complete the assessment is and may not capture every instance of non-compliance.

## 1.8 Definition of Terms

**Privacy Policy:** A written statement that explains how a mobile application collects, uses, stores and shares user data.

**Mobile Application:** A software program designed to run on smartphones or tablets to provide specific services to users.

**User Consent:** Permission given by a user, usually through agreement, to allow an app to access their personal data.

**Data Collection:** The process through which an app gathers personal or usage information from a user.

**Third Party:** An external group or service that may receive or process data collected by a mobile application.

**User Awareness:** The level of understanding a user has about the data practices and privacy terms of an application.

**Policy Compliance:** The extent to which a mobile app's actual data practices align with the claims made in its privacy policy.

# CHAPTER TWO

# LITERATURE REVIEW

## 2.0 Introduction

This chapter is broken down into three sections. They are as follows: conceptual literature, theoretical literature, and empirical literature. The conceptual literature examines some concepts related to the subject matter, the theoretical literature examines some relevant theories related to the topic under study, and the empirical literature examines some previous studies that are closely related to this current study with their findings.

## 2.1 Privacy Policy

Besides the standard permissions for API access documented in manifest files, applications' privacy policies are a source for identifying what information is collected and used by apps. A privacy policy serves as the primary means to communicate with users regarding which and how sensitive personal information (SPI) has been accessed, collected, stored, shared (app to app, and to third party), used/processed, and the purpose of the SPI collection and processing. Privacy policies generally consist of multiple paragraphs of natural language such as the following excerpt from the Indeed Job Search app's privacy policy2 listed on Google Play:

Indeed may create and assign to your device an identifier that is similar to an account number. We may collect the name you have associated with your device, device type, telephone number, country, and any other information you choose to provide, such as user name, geo-location or e-mail address. We may also access your contacts to enable you to invite friends to join you in the Website. Privacy policies are particularly important in the United States due to the "notice and choice" approach used to address privacy online (Hatia, 2019). Under this framework, app companies post their privacy policies and users read the policies to make informed decisions

on accepting the privacy terms before installing the apps (Hatia, 2019). However, most privacy policies prepared by policy authors are difficult to understand due to their verbose and ambiguous nature, and this can lead to users to skip reading policies even if they have concerns about information collection practices. More significantly, the app developers might not be able to comply with privacy policies effectively. To address this issue, this work aims to provide a framework to achieve alignment between apps' privacy policies and implementation code, and better communication among software developers and policy writers.

A major hindrance in the understanding and analysis of privacy policies is that there is no canonical format for presenting the information. The language, organization, and detail of policies can vary from app to app.

### 2.2 Manual Preparation

The goal of this work is to discover information regarding the relationship between terminology used in privacy policies expressed in natural language and API method calls used in the corresponding code. Such a mapping would then provide semantic information regarding the natural language. In turn, an app's source code could more easily be checked for misalignment with its corresponding privacy policy. Before we can perform such an automated detection of privacy policy violations, we must construct initial data sets and a mapping from which the knowledge can be used to detect violations in other apps. The following subsections describe how we leveraged a small subset of Android apps' source code to implement a mapping from API methods to policy phrases. This information is then used to detect violations in a much larger set of Android apps.

In our approach, we created a mapping between API method signatures in the Android SDK and meanings shared between API documents and privacy policies. The shared meanings are described in an ontology that provides support for comparing two technical terms: we say that one term subsumes a second term, when either the first term is more general than the second term, called a hypernym, or when the second term is part of the first term, called a meronym. For example, “mobile device model” and “sensors” are parts of a “mobile device,” whereas “mobile device model” is also a kind of “mobile device information.” In addition, we define two terms as synonyms when the meaning is equivalent for our purposes (e.g., when “IP address” is a synonym for “Internet protocol address”). Because privacy policies tend describe technical information using more generic concepts, the ontology allows us to map from low-level technical terms to high-level technical categories, and vice versa. Once the ontology is constructed, we can use tools to automatically infer which terms should appear in privacy policies based on the API method calls in a mobile application.

We now describe how we created the ontology and mapping by extracting terminology from the privacy policies and API documents respectively, before we classified this terminology using sub-sumption and equivalence relationships. In each step, we employed research methods aimed at improving construct and internal validity and reliability, which we discuss.

### 2.2.1 Extracting the API Terminology

In our approach, a subject matter expert, who would typically be the maintainer of the Application Programmer Interface (API) documentation, annotates an API document. The annotations map key phrases in the API documents to low-level technical terminology in an API lexicon (e.g., “scroll bar width” or “directional bearing” are low-level technical terms). To bootstrap our approach, we chose to annotate the entire collection of API documents in the Android

SDK, which includes 2,988 API documents containing over 6,000 public method signatures (here, the term "public" refers to the Java access modifier). Each API document consists of one or more method signatures, which each consist of the method name, input parameters, the return type, and a natural language description of the method's behavior.

The annotation procedure involves three steps: (a) we extract the method names, input parameters and natural language method descriptions from the API documentation to populate a series of crowd worker tasks; (b) for each crowd worker task, two investigators separately annotate the extracted fields by identifying which phrases correspond to a kind of privacy-related platform information; and (c) the resulting annotations are compiled into a mapping from the fully qualified method name, including API package name, onto each annotated phrase (i.e., each method name can map to one or more platform information phrases). We only compiled mappings where the two investigators both agreed that the phrase was a kind of privacy-related, platform information.

In the first step, the signatures were automatically extracted from the API documents, which were themselves expressed in HTML generated using the Javadoc toolset. The signatures were then segmented into sets of 20 signatures or less, and each set was presented in a separate crowd worker task. Applying the segmentation to the 2,988 API documents yields 310 crowd worker tasks.

The crowd worker task employs a web-based coding toolset developed by Breaux and Schaub (2020) for annotating text documents using coding theory, a qualitative research method for extracting data from text documents. In coding theory, the annotators use a coding frame to decide when to code or not to code a specific item. In our study to annotate the API documents, our coding frame consisted of a single information code defined as information "related to personal privacy and accessed through the platform API."

In the second step, two investigators used this web-based toolset to code the 310 crowd worker tasks, consuming 6.5 and 6.6 hours for each investigator to yield 195 and 196 annotations, respectively. An excerpt from the crowd worker task, where a worker has annotated phrases in the Location package of the An-droid API. The toolset has been validated in a prior case study to extract privacy requirements from privacy policies (Radshaw, 2020). The toolset also includes analytics for extracting overlapping annotations where n or more workers agreed that the phrase should be annotated.

From the two investigator's combined annotations, we produced 219 unique annotations with duplicate annotations removed. The total 219 annotations were next compiled into a mapping between API method signatures and annotated phrases. The phrases in the mapping were normalized by the two investigators by converting the annotated text into simple noun phrases (described further in Section 3.4). This is necessary to reduce the variety of ways that method behaviors are described into a concise, reusable API lexicon. The resulting lexicon contains 162 unique phrases and 169 total mappings between phrases and API method names. A total of 154 methods were annotated based on the criteria that they produce privacy related information.

### 2.2.2 Extracting the Privacy Policy Terminology

Each app page on Google Play includes a link to the app's privacy policy if it is specified by the developer. We created a Python script to download the privacy policies from these links for the top 20 free apps in each app category3. We filtered these policies based on their formatting, language (we only considered policies written in English), and whether or not a "Privacy Policy" section was explicitly stated in the document and randomly selected 50 from this pool for terminology extraction.

For our approach, we determine which kinds of technical information should appear in privacy policies to describe privacy-relevant API method calls. To bootstrap our method, we developed a privacy policy lexicon in which six investigators annotated the 50 mobile app privacy policies using our crowd worker task toolset (Reaux & Schaub, 2019). Unlike the API lexicon, wherein we used only two investigators with programming experience, we used six annotators for extracting terms from privacy policies, because privacy policy terminology includes vague and ambiguous terms that span a broader range of expertise (e.g., "taps" corresponds to user input, whereas "analytics information" includes web pages visited, links clicked, browser information, and so on.) Thus, by increasing the number of annotators, we increased our likely coverage of potentially relevant policy terms.

The crowd worker task employs the same web-based coding toolset developed by (Breaux and Schaub 2019). To prepare the policies for annotation, we first removed the following content: the introduction and table of contents, "contact us", security, U.S. Safe Harbor, policy changes and California citizen rights. This content generally appears in separate sections or paragraphs, which reduces the chance of inconsistency when removing these sections across multiple policies. While these sections do describe privacy-protecting practices, such as complying with the U.S. Safe Harbor, we have never observed descriptions of platform information in our analysis of over 100 privacy policies in our previous research.

Next, we manually split the remaining policy into spans of approximately 120 words. We preserve larger spans which either have an anaphoric reference back to a previous sentence (e.g. when "this information..." depends on a previous statement to understand the context of the information), or when the statement has subparts (e.g., (a), (b) etc.) that depend on the context

provided by earlier sentence fragments. On average, we need 15 minutes per policy to complete the preparation.

The coding frame for the privacy policy terminology extraction consists of two codes: platform information, which we define as "any information that $company or another party accesses through the mobile platform, which is not unique to the app;" and other information, which we define as "any information that $company or another party collects, uses, shares or retains." We replace the $company variable with the name of the company whose policy is being annotated. Next, we compiled the annotations where two or more investigators agreed that the annotation was a kind of platform information; we excluded non-platform information from this data set. We applied an entity extractor (Hen, 2020) to the annotations to itemize the platform information types into unique entities, which were then included in the privacy policy lexicon.

Among the 50 policies, we constructed 5,932 crowd worker tasks with an average word count of 98.6; the average words per policy was 2054.6. These tasks produced a total of 720 annotations across the 50 policies, which yielded a total of 368 unique platform information entities. The total time required to collect these annotations was 19.9 hours across six annotators, all of whom are authors of this research. We now discuss how we created a platform information ontology from this lexicon.

### 2.2.3 Constructing the Ontology

A common phenomena in natural language description is generalization, in which a more general phrase can be used to imply a number of sub-concepts of the phrase. For example, the phrase "technical information" may imply a wide range of technical data, while the phrase "device identifier" is more specific, but its concept is still covered by phrase "technical information". Since

phrase generalization is often used to describe information collected, it is important to be able to distinguish these relationships between phrases in order to identify cases where a concept is represented in another phrase. To handle this, we created an ontology of privacy-related phrases to be used as a cross reference during the identification of methods not represented in privacy policies.

An ontology is a formal description of entities and their properties, relationships, and behaviors (Hinck, 2017), and is described with formal languages such as OWL (based on Description Logic). In the context of phrase mapping, we use an ontology to represent a hierarchical classification of phrases. For example, “IP Address” is a decedent of “Network Information”, indicating that IP Address is a type of network information. The hierarchical nature of an ontology allows for transitive relationships that can be used for mapping API methods to phrases indirectly based on relationships between the phrases themselves.

The ontology is used to formally reason about the meaning of terminology found in the API documents and privacy policies. For an API lexicon $\hat{}A$ and a privacy policy lexicon $\hat{}P$ consisting of unique terms (or concepts), the ontology is a Description Logic (DL) knowledge base KB that consists of axioms C v D, which means concept C is subsumed by concept D, or C ≡ D, which means concept C is equivalent to concept D, for some concepts C, D ∈ (A ∪ P ). Using our API lexicon, our aim is to map a method name m from an API document to a concept A ∈ $\hat{}A$.

Next, we aim to infer (in a forward direction) all policy concepts {P |P ∈ $\hat{}P$ ∧KB |= P v A∨KB |= P ≡ A}. In this respect, we can extract method names from method calls in a mobile app, then infer corresponding policy terms (among which at least one) should appear in the mobile app’s privacy policy. Similarly, we can reason in the backward direction to check which policy

terms mentioned in the app's policy map to which method names corresponding to method calls in the app.

We constructed the ontology following a method developed by Wadkar and Breaux (2012). First, we generated a basic ontology consisting of one concept for each term in the privacy policy lexicon; each concept was subsumed by the > concept, and no other relationships among concepts existed. Second, for two copies of the basic ontology KB1 and KB2, two investigators separately performed pairwise comparisons among term pairs C, D in each ontology, respectively: if two terms were near synonyms, the first investigator created an equivalence relation KB1 |= C ≡ D; else, if one term subsumed the other term, the first investigator created a subsumption relationship KB1 |= C v D. Due to the num- ber of pairwise comparisons, it's not unreasonable to expect that a single investigator would produce an incomplete ontology, or an ontology that is inconsistent with another investigator's ontology.

To check for completeness and consistency between two investigators, we compared all relationship pairs between KB1 and KB2, including cases were a relationship did not exist in one of the ontologies. Both investigators met to reconcile any differences, recognizing that classification differences can persist forward into our analysis of mobile app violations. For two investigators, the resulting ontologies KB1 and KB2 consisted of 431 and 407 axioms, respectively. The first comparison yielded 321 differences and was evaluated using Cohen's Kappa to measure the degree of agreement above chance alone (Saldana, 2022), which was 0.233. After the reconciliation process, the investigators were left with 12 differences and a Cohen's Kappa of 0.979.

### 2.2.4 Constructing the Mapping

With the ontology constructed from the privacy policy lexicon, individual API methods could then be mapped to one or more terms in the ontology based on their annotations from the API lexicon as well as their return types. The study shows how intermediate noun phrases were created as a canonical representation of the method's description and then mapped directly to terms in the ontology based on their relationships. This canonicalization process made explicit the domain knowledge about the methods (i.e., canonical terms) and the natural language used to describe the method in privacy policies (i.e., terms in the ontology). As exemplified in the figure, the documentation describes "dynamic information about the current Wi-Fi connection" as the data it produces. In cases such as this, where the description did not explicitly describe the information returned, we analyzed the object returned by the method. Here, the object (of type WifiInfo) provided multiple public fields and methods from which we were able to assign the canonical terms in the figure (as seen in the white circle). From there, the canonical terms were associated with related terms in the ontology based on their relationships. This effectively produces a mapping between each of the API methods and one or more terms in the ontology (assuming the method is privacy-relate) and vice versa. We refer to this many-to-many mapping relation, of which each element is a pair, (policy term, AP I method), as M appings in the following sections.

### 2.3 Automated Violation Detection

To detect potential privacy policy violations, we first identify API method invocations that produce data covered by a known policy term from the privacy policy lexicon. Next, we use information flow analysis to check whether that data flows to a remote server via a subsequent network API method invocation. Data collected by a method is considered a potential privacy policy violation if the method is not represented in the app's privacy policy through Mappings.

#### 2.3.1 Weak and Strong Violations

As discussed in Section 2.3, privacy policies serve to inform users about how their personal information is collected and used. These policies cover a wide range of practices, including in-store, client-side, and server-side practices, and they may describe all of a company's practices, or be limited to only those practices of a single product or service. In this paper, we are only concerned about client-side practices affecting mobile applications. In addition, privacy policies are not complete: they generally describe a subset of the company's practices. Therefore in our approach, we only detect errors of omission, in which the app collects a kind of information that is not described in the policy. Errors of omission are potential policy violations, because the collection may be unintended by the app developer. Moreover, because privacy includes notifying users about how their information is collected and used, errors of omission represent potential privacy violations. We detect two kinds of violations resulting from errors of omission: strong violations that occur when the policy does not describe an app's data collection practice, and weak violations that occur when the policy describes the data practice using vague terminology. Other kinds of policy errors, such as direct conflicts, in which a conflict occurs because the policy states that an app does not collect a kind of information and the app does indeed collect that kind of information, are out of scope of this research.

### 2.3.2 Evaluations of iOS and Android Labels

Zhang (2010) conducted an interview study with 24 iPhone users to explore people's understanding and perceptions of iOS App Privacy Labels. They found that most users were unaware of iOS privacy labels. After looking at example labels, participants found them useful, although some considered them vague and many did not trust their accuracy. This work identified common misunderstandings of terms used in iOS privacy labels, including confusion about the definitions of data collection groups (e.g., "Data Used to Track You" and "Data Not Linked to

You") and various data type categories (e.g., "user content" and "identifiers"). Our work replicates many of the results found by Zhang et al. and extends their work by evaluating Android Data Safety Labels and comparing the two label designs.

Prior work has also evaluated iOS App Privacy Labels from the developer's perspective. Li et al. conducted an interview study with 12 iOS app developers to understand their challenges in creating labels. They identified common developer errors that led to inaccurate labels, including misunderstanding "Data Linked to You," underestimating data use by third-party libraries, forgetting about collected data, incorrectly handling optional data practices, and incorrectly reporting locally-stored data. They recommended revised definitions, expanded documentation, and automated validity-checking tools to help developers create accurate privacy labels (Kassab, 2020).

Several research teams have performed quantitative analyses of large datasets of iOS App Privacy Labels. (Li, 2020). analyzed a longitudinal dataset of 1.4 million apps on the U.S. App Store from April to November, 2021. They found that developers were slow to add privacy labels, with most apps only adding or updating their privacy label when they update the app. They also reported statistics about reported data practices. Kollnig et al. performed a comparative analysis of 1,759 iOS apps before and after Apple introduced app privacy labels. They found that app privacy labels are inconsistent with actual data practices. For example, 80.2% of apps that claimed they did not use data to track the user in fact contained ad libraries. They also concluded that the addition of app privacy labels had not changed apps' data use practices. Koch et al. (2020) performed a statistical analysis of 11,074 iOS apps and their privacy labels (or lack thereof); they found that most apps report collecting some data, and that game apps in particular collect more

data and use more data for tracking. Many of the labels in their dataset include inconsistencies, for example, 13% erroneously claim personal information as “Data Not Linked to You”.

Prior work has also analyzed applications to evaluate the accuracy of iOS App Privacy Labels. Koch et al. dynamically analyzed the traffic of 1,687 iOS apps and found that 16% of apps transmit data without declaration. Xiao et al. used a combination of static and dynamic analysis to systematically evaluate consistency between 5,102 iOS labels and actual app behavior; they found that 67% of iOS labels are inconsistent with actual app behavior, with most apps failing to fully disclose all data collection and purposes.

Jain et al. (2019) used NLP techniques to compare iOS App Privacy Labels to app privacy policies; they analyzed 354,725 iOS apps and found that only 29.6% of apps provided both an App Privacy label and a privacy policy and that 88.0% of those apps exhibited possible discrepancies between the label and the privacy policy. The first work to look at Android Data Safety labels was a recent study by the Mozilla Foundation that looked at 40 top Android apps and found that almost 80% had Data Safety Labels that were inconsistent with the app’s privacy policy. (Khandelwal, 2020) conducted a large-scale analysis of 165K apps listed on both iOS and Android app stores; they found that many apps were missing labels and that most apps with labels had inconsistencies between the two labels. In other work, (Khandelwal, 2020) conducted a large-scale longitudinal analysis of 1.14 million Android apps and found that missing labels and internally inconsistent labels are common, and that many apps are updating and refining their labels over time. They also identified challenges based on survey responses from 889 Android developers.

### 2.3.3 Evaluations of Privacy Communications

Although mobile app privacy labels have only existed for a few years, the idea of privacy labels has been around for over two decades, motivated by concerns about the length of privacy policies and their ability to communicate to users. As early as 2001, Commissioners of the U.S. Federal Trade Commission advocated for standardized privacy "nutrition labels" to help consumers understand and compare website privacy practices. Since then, privacy labels have been proposed and evaluated in a variety of domains including websites, mobile apps, and IoT devices.

(Kelley, 2020) first proposed designs for "privacy nutrition labels" in 2009 in the context of website privacy policies. They designed and evaluated standardized labels through a series of focus groups, lab studies, and online surveys. Their final design which centered a matrix depicting practices for data type-purpose pairs allowed users to more quickly and accurately identify a website's data practices compared to natural-language privacy policies. In 2009, eight US federal agencies collaborated on a model privacy notice for US financial institutions that included a table of bank data practices in a standard format, similar to a nutrition label.

(Cranor, 2020) collected over 6000 bank privacy notices in this format and conducted a large-scale analysis. They also identified issues with the model notice design. Later work extended privacy labels to other domains. Kelley et al. (2020) proposed "Privacy Facts" for mobile apps that swayed users towards selecting more private apps in contexts where the user was choosing between otherwise similar apps. More recently, Emami-Naeini et al. (2020) proposed a privacy label for IoT devices; they found that their proposed privacy label effectively communicated information about privacy risks to users and that it influence hypothetical user purchasing decisions.

Cookie banners are another type of privacy-related communication that has been evaluated by researchers. Researchers have found that consumers largely do not understand the privacy

decisions facilitated by these banners and that the banners often nudge users to make choices that do not align with their preferences. Habib and Cranor synthesized approaches used to evaluate privacy choice interfaces and proposed an evaluation framework with seven factors. Although app privacy labels do not directly offer choice interfaces, they may be used to inform users' app download decisions. Thus, these factors are relevant to our evaluation, especially two of the factors: awareness and comprehension.

### 2.3.4 Scenarios and User Stories

Scenarios describe a concrete invocation of a system through a sequence of steps, often from a user's perspective, and are widely used in software engineering, as well as human-computer interaction, and organizational process design. Scenarios have been used to elicit, analyze, and validate requirements, including quality requirements for security, resiliency and safety. Scenarios can surface requirements unforeseen by business analysts and improve requirements alignment with users, and can be used to create more robust domain models. Scenarios describe a system at different levels of abstraction, can be used in tracing requirements to software architecture and code, and in software testing. When considering different stakeholder perspectives, scenarios can illustrate areas where value-conflicts arise. Scenarios can be combined with personas and goal modeling to identify conflicting requirements.

Techniques exist to validate the scenario syntax and grammar using templates and rule-based verification, to identify missing steps, and to compute scenario similarity. Scenarios can be used to validate formal models by challenging model assumptions, to semi-automatically derive use-cases, and to identify functional requirements from scenario steps.

Whereas scenarios describe multiple steps, user stories are more concise and expressed using various templates, among which the Connextra format is most popular, i.e., as a <role>, I want <action>, so that <benefit>. User stories can be mapped to scenarios by elaborating on the action from the user's perspective. The general level of detail, which hides the underlying interactions with software, has made user stories popular in agile software development whereby the story summarizes one or more units of work in an iteration. When eliciting requirements from stakeholders, however, there are multiple what, how and why questions to investigate, which is why we chose to use scenarios instead of user stories in our method.

### 2.3.5 Privacy Attitudes and Risk

Privacy researchers have sought to understand why individuals share sensitive data with organizations that might misuse that data. Alan Westin introduced the Privacy Segmentation Index through a series of surveys to segment individuals based on their relationship to privacy: fundamentalists are generally distrustful of organizations, pragmatists weigh the costs and benefits of trust, and the unconcerned are generally trustful of organizations. Acquisti and Grossklags studied the privacy paradox, in which user behaviors indicate a low value placed on privacy despite what they self-report. (Dupree, 2020) clustered users into five privacy behavior categories: fundamentalists, lazy experts, technicians, amateurs, and the marginally concerned. These categories may explain why some users are more or less concerned about their privacy.

Kang, (2021) found that Amazon Mechanical Turk (AMT) workers value anonymity and hiding information and had more privacy concerns than the general U.S. public. Risk has been studied in marketing, psychology, and economics, with popular definitions focusing on a function of the likelihood and magnitude of an adverse event. Marketing risk is a choice among multiple options based on the likelihood and desirability of the choice's consequences, whereas

pyschological risk is an individual's willingness to participate in an activity. (Kaplan and Garrick 2020) define economic risk as a function of probability and consequence, where the consequence is the measure of damage or harm.

While Cronk adapts economic risk to privacy, Bhatia and Breaux (2012) adapt psychological risk to an individual's willingness to share personal data, which they have studied in the context of vague and ambiguous data practices. While users are known to rarely read privacy policies describing data practices, evidence shows users can estimate their privacy risk using data-specific prompts.

### 2.3.6 Natural Language Processing

In requirements engineering, natural language processing techniques have been used to extract important entites from text-based artifacts. (Pudlitz, 2016) apply LSTMs and convolutional neural networks (CNNs) to extract system state de- scriptions from requirements specifications, and Siahaan, 2017) used named-entity recognition (NER) to extract hard and soft-goals from online news sources. In privacy and security, NER and transformer-based deep neural networks have been used to generate access control policies from user stories, and extract data-flow diagram elements from user stories, frequently focusing on the data subject, data type and data action. (Casillo, 2023) apply CNNs to classify words in user stories as disclosure-related (e.g., access, share). Others have used part-of-speech-based rules to identify information types in privacy policies, and RNNs to extract penality clauses from regulations. (Sannier, 2022) have used regular expressions and constituency and dependency parsing to extract legal primitives from laws, which can then be used to query a legal knowledge base.

Social media posts and mobile app reviews, that describe user in-app experiences have been proposed as a source of requirements, including user opinions. These approaches employ sentiment analysis and typed dependencies, for example. (Hatamian, 2020) analyze 812,899 app reviews from the top 10 apps in each of 20 categories on Google Play (200 apps total), and found less than 2,500 reviews (or 0.31%) of all reviews raise privacy concerns. Topics raised in privacy concerns include tracking and spyware, phishing, unintended disclosures, targeted ads, and spam. However, others note that these approaches can be noisy and hard to replicate, with measured recall as low as 0.34 and 0.44 for two popular app review mining approaches. Moreover, app reviews and social media rely on users reaching a level of undesirable frustration before they raise such concerns publicly, which developers should want to avoid. Thus, we propose a method that invites users to directly comment on specific app screens, which developers can deploy with little manual intervention.

# CHAPTER THREE

# SYSTEM ANALYSIS DESIGN

## 3.0 METHODOLOGY

The method for the software design for this research work is the structure analysis and design methodology (SSADM) which encapsulate in the following step.

1. Problem identification
2. Feasibility study
3. Analysis
4. Design
5. Post implementation

## 3.1 PROBLEM IDENTIFICATION

In solving a problem, it is very necessary to first identify the problem. In order for the problem to be identified then the existing system must be invest.

This project seek to investigate into the existing method of data recovery which are:

i. Logical recovery
ii. Physical recovery

Logical Recovery: this technique involved the recovery of logical damage file which are primary cause by power outage that prevent file structure to be strength and weakness of zero knowledge analysis.

Strength

-it rebuild the system from scratch

Weakness

- It does not repair the underlying file system by merely allow for data to be extracted from it to another storage device

- It is very slow, which can take several days, weeks and at times end up without recovering any data.

- It is very technical.

Physical Recovery: These involved the recovery of data that are caused by a physical damage of the hardware.

There are two types of physical recovery these are:

-Hardware repair

-Disk imaging

Hardware Repair: These involved the replacement of physical faulty parts with a matching unfaulty one.

Disk Imaging: These raw images are used to reconstruct usable data after any logical damage has been repaired.

Strength and weakness of physical data recovery

Hardware Repair:

Strength

- It can recover data in a hardware that has fallen into water, electric hazard etc.
- Its 75% accurate in the recovery of data.

- It recovers almost all data in hardware that are physically damaged.

Weakness

- It cannot recover data that has been broken from the magnetic surface
- It is very expensive
- It is technical; it can only be done by a well-trained data recovery expert which is carried out in a control environment.

Disk imaging:

Strength

- It is used to report/select bad sector of the hardware

Weakness

- In some cases, it reports an error/ bad sector resulting in the loss of information which is actually available.

Feasibility Study: Here the economic and financial benefit is always considered.

The financial aspect can be measured on the aspect of cresting a data.

In most business organization data is the most priceless Asset, in other words data makes the business organization to be in existence, losing some aspect of data can be a threat to the business organization.

Once there is a suspected loss of data what the organization will first do is as mentioned below:

- The cost of continuing with the data
- The cost of recreating the data

- The cost of notified user in the event of a company.

It is now clear that the cost of losing a data is far more expensive than recovering the data itself.

In consisting the short pace of time allocated to this project work designing a complex data recovery software will not be feasible, but in order to ensure this project is completed within the limited time frame, we hope to design a rather simple data recovery software that will make use of the backup technique.

In view of this fact, we also want to recommend that future research work in this line of research, can take this project work to a greater height by improving on what has been done.

## 3.2 ANALYSIS

### 3.2.1 Data Gathering Techniques Used

In gathering information, and data for this project work, the following were used:

i. Information was gathered from textbook on data processing, data recovery technique/schemes
ii. Articles and books was also downloaded from the internet

### 3.2.2 Analysis of the Present System

Since smartphone applications deal with consumers' private information, they must abide by a variety of security and privacy regulations, such as the GDPR (EU General Data Protection Regulation, 2016) and the NDPR (Nigeria Data Protection Regulation, 2019). By law, app developers must provide users with a written privacy policy that describes how they gather and process data. As a result, privacy policies are the major source of information for users interested in knowing how an application uses their private details. The privacy texts of the appset were

investigated using keyword and semantic-based search approaches to establish the extent to which privacy policy texts are relevant to the developer's request in our analysis (in manifest). As a result, the app privacy policies were examined to assess how focused they are on app data collection tactics; such as if the purpose definition of information gathering premised on dangerous permissions is expressed explicitly inside the policy text. As the NDPR is relatively new and not fully enforced, this research was based on the GDPR.

### 3.3 Weakness of the present System

Some of the weakness of the present privacy policies in mobile applications is

**Vague language**: policies often use complex jargon, making it hard for users to understand what's happening with their data.

**Lack of transparency:** Apps don't always clearly disclose data collection, usage or sharing practices.

**Overly broad permissions**: Some apps request excessive permissions, accessing more data than necessary.

**Data sharing with third parties**: Policies often allow sharing with advertisers, analytics firms, or other companies.

### 3.4 The proposed System

In view of the above weakness or problems identified existing in present privacy policies in mobile application, a new understandable framework will be established. Which include a good data summary. This allows users have a brief overview of data practices at app installation.

**Permission manager:** this allow users to adjust permission and data sharing

**Data dashboard:** this provide access to collected data and insight

**Opt-out options:** make is easy for users to delete data or leave the service

**Physical Security is Limited**

Smartphone applications are usually created by a sole person or a group of developers with scarce funds and poor knowledge of privacy and security. As a result, implementing the best data privacy-related practical solutions and strategies for smartphone application developers is challenging.

**User Interfaces are Limited**

Tiny User Interfaces are common on smartphones (UI). This has a huge effect on privacy, openness, and security. For example, it was discovered that passwords made on smartphones are weaker. Privacy notices on a smartphone are more difficult to comprehend and require extra attention. Consequently, privacy rules ought to be developed using a 'tiered' approach in which the most relevant features are stated first, with further information available if the consumer wishes to learn more.

**Third-party Software is used**

Many smartphone apps are made up of a number of features produced by companies other than the app's creator. The 3rd party libraries help app creators perform analytics, such as tracking user activity, integrating them with social networking, and generating cash by displaying advertisements. Libraries may, however, gather sensitive private information for their own use in addition to the services they provide. The library's authors may be able to use this data to develop precise digital buyer personas by combining information from multiple mobile apps. For instance, a customer may offer one app access to only obtain their geolocation data while granting access to their contacts to another app. When both apps are using the same 3$^{rd}$ party library, the creator of

the library might be able to link the two datasets. Furthermore, since the libraries sometimes are not open-source, analysis is challenging. As a result, it's possible for an app developer to be oblivious of the data collected by these functionalities.

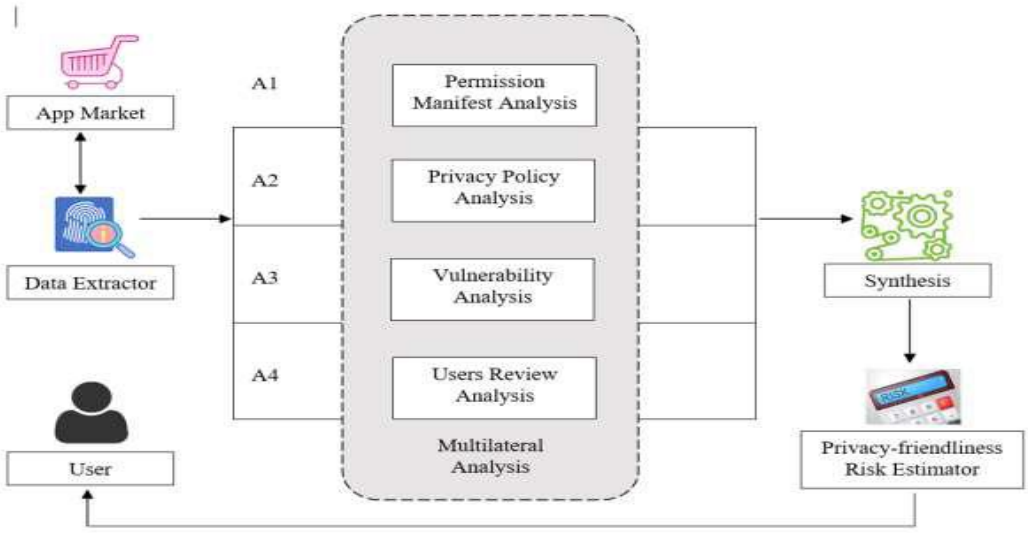


**Figure 3.1:** Privacy Policies analysis in Mobile Applications

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.myfirstapplication">

    <uses-permission android:name="android.permission.READ_CONTACTS"/>
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <meta-data
            android:name="preloaded_fonts"
            android:resource="@array/preloaded_fonts" />
    </application>

</manifest>
```

**Figure 3.2**: Sample Android manifest.xml file

```
apktool d path/to/app.apk
```

**Figure 3.3:** A Decompiling APK file from a Linux terminal

```
sudo apt-get install git

sudo apt-get install python3.8

sudo apt-get install openjdk-8-jdk

sudo python3-dev python3-venv python3-pip build-essential libffi-dev libssl-dev libxml2-dev libxsltl-dev libjpeg8-dev xliblg-dev
wkhtmltopdf
```

**Figure 3.4:** Commands to install MobSF dependencies on Linux terminal

```
git clone https://github.com/MobSF/Mobile-Security-Framework-MobSF.git

cd Mobile-Security-Framework-MobSF

./setup.sh
```

**Figure 3.5:** Commands to download and install MobSF on a Linux terminal

# CHAPTER FOUR

# SYSTEM IMPLEMENTATION

## 4.1 PRIVACY POLICY ANALYSIS

In this section we present our automated large-scale ML analysis of privacy policies. We discuss the law on privacy notice and choice, our evaluation of how many apps have a privacy policy and the analysis of policy content.

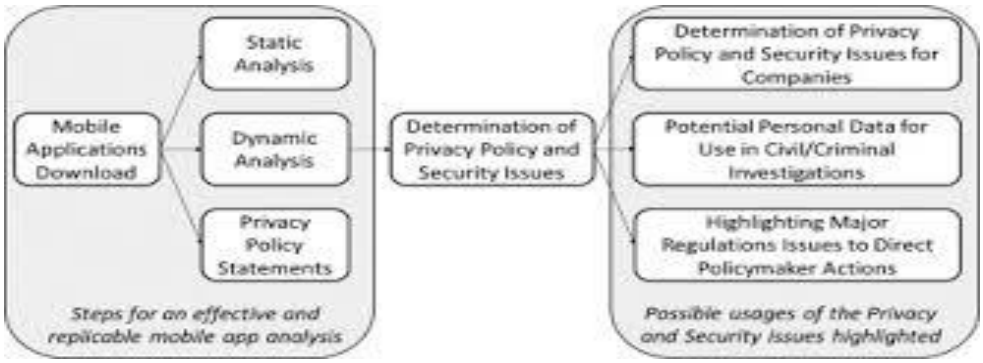


**Figure 4.1: Mobile App Privacy Preferences**

Given the large variances identified above, a unified default setting evidently cannot satisfy all the users' privacy preferences. Therefore, chose to investigate methods for segmenting the entire user population into a number of subgroups that have similar preferences within the subgroups. Then by identifying the suitable default settings for each of these groups and the group each user belongs to, we can suggest individual users with more accurate default settings.


**Figure 4.2: Create a privacy policy for phone**

In order to compare the policy analysis results to what apps actually do according to their code we now turn to our app analysis approach. The system design will be discussed.

**App Analysis System Design**

The app analysis system is based on Androguard (2012), an open source static analysis tool written in Python that provides extensible analytical functionality. Apart from the manual intervention in the construction and testing phase our system's analysis is fully automated. A brief example for sharing of device IDs will convey the basic program flow of our data-driven static analysis. For each app our system builds an API invocation map, which is utilized as a partial call graph. To illustrate, for sharing of device IDs all calls to the TelephonyManager.getDevice. Id API are included in the call graph because the caller can use it to request a device ID. All calls to this and other APIs that can be used to request a device ID are included in the call graph and passed to the identification routine, which checks the package names of the callers against the package names of selected third party libraries that we want to analyze. In order to make use of the get DeviceId API a library needs the READ_PHONE_STATE permission. Only if the analysis detects that the library has the required permission, the app is classified as sharing device IDs with third parties. We identified relevant Android API calls for the types of information we are interested in and the permission.

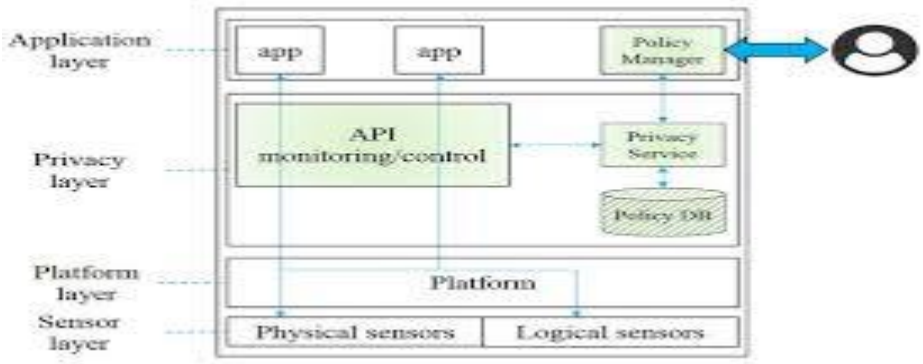


**Figure 4.3: Safeguard mechanism for mobile device**

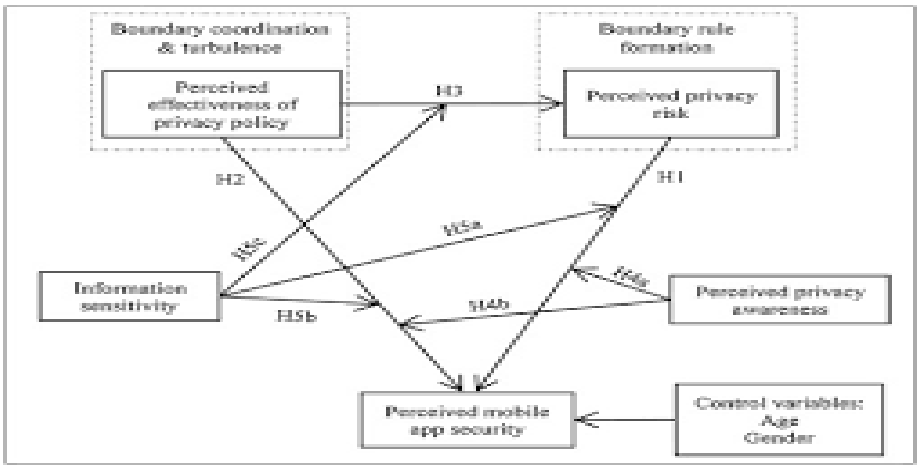


**Figure 4.4: Mobile application security**

It ensure sensitive data in encrypted in transit (e.g., HTTPs) and at rest. It can be used to implement robust authentication mechanisms, like OAuth or JWT. It also validate user input to prevent SQL injection and cross-site scripting (XSS).

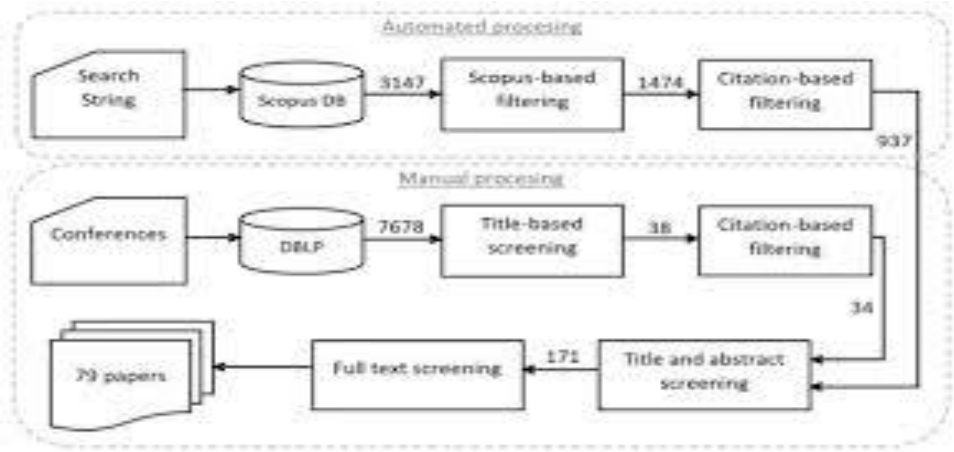


**Figure 4.5: Privacy assessment in android app**

This application ensure where and how data is stored, is it encrypted? It also shared data with third party. It also check if permissions are justified and properly handled.

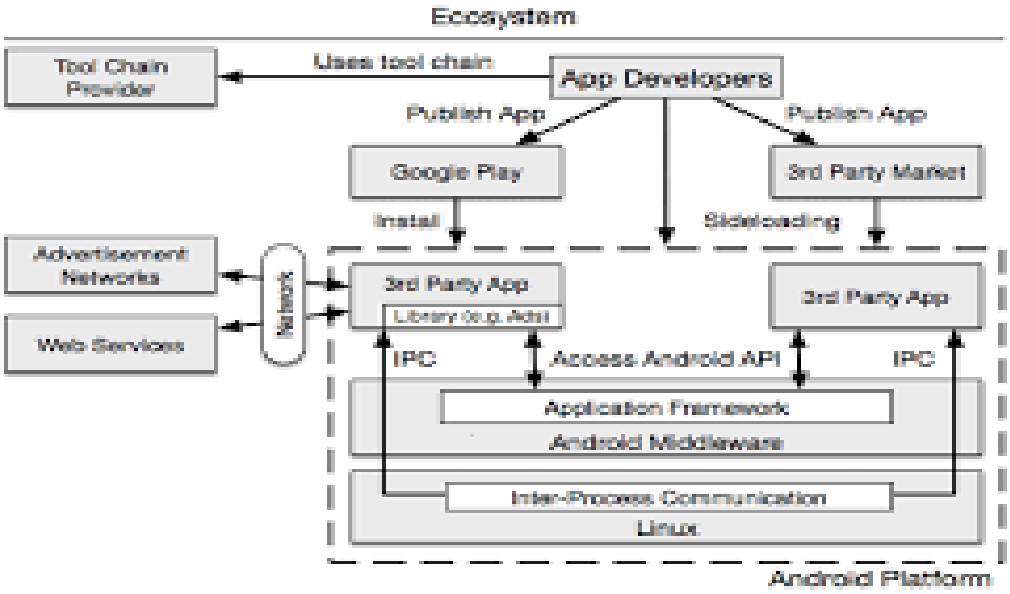


**Figure 4.6: android Ecosystem**

It safeguard user data, implement these measures. Use robust authentication mechanism, limit data access to authorized parties, collect only necessary data, use secure storage mechanism (e.g., Keychain, Android, Keystone, inform users about data collection and usage.

# CHAPTER FIVE

# SUMMARY, CONCLUSION AND RECOMMENDATION

## 5.1 Summary

The law of notice and choice is intended to enable enforcement of data practices in mobile apps and other online services. However, verifying whether an app actually behaves according to the law and its privacy policy is decisively hard. To alleviate this problem we propose the use of an automated analysis system based on machine learning and static analysis to identify potential privacy requirement inconsistencies. Our system advances app privacy in three main areas: it increases transparency for otherwise largely opaque data practices, offers the scalability necessary for potentially making an impact on the app eco-system as a whole, and provides a first step towards automating mobile app privacy compliance analysis.

Our results suggest the occurrence of potential privacy requirement inconsistencies on a large scale. However, the possibilities of the techniques introduced here have yet to be fully explored. For example, the privacy policy analysis can be further developed to capture nuances in policy wording possibly by leveraging the structure of policies (e.g., by identifying definitions of PII and where those are referenced in a policy). Similarly, the accuracy of the app analysis could be enhanced by integrating data flow analysis techniques. However, for performance reasons resources should be used sparingly. A practical system for the purpose of large-scale app analysis necessarily remains relatively lightweight.

### 5.2 Conclusion

In conclusion, our large-scale measurements of Privacy Labels have provided valuable insights into the privacy practices of apps. By analyzing Data Safety Sections for android apps and Apple Privacy Labels, we provided a comprehensive picture of the privacy practices of the applications. On the one hand, privacy labels provide users with more specific information about the data practices of apps than traditional privacy policies. However, our comparison of Privacy Labels for cross-listed apps in the Play Store and Apple Store showed differences in the practices disclosed, indicating that developers are not consistently disclosing the same information on different platforms. This can confuse users and make it difficult to make informed decisions about which apps to use based on their privacy concerns. Overall, these findings highlight the importance of carefully reviewing Privacy Labels and other sources of information when evaluating the privacy practices of apps. They also suggest that there is a need for improved transparency and accountability in the app industry, as developers may not always be accurately disclosing their data collection and use practices. A more transparent system will allow the consumers to be aware

of the data collection and use practices of the apps and make informed decisions about their privacy.

The study suggests that it is necessary to develop the proposed privacy requirement analysis in tandem with public policy and law. Regulators are currently pushing for app store standardization and early enforcement of potentially non-compliant privacy practices. Approaches like the one proposed here can relieve regulators' workloads through automation allowing them to focus their limited resources to move from a purely reactionary approach towards systematic oversight. As we also think that many software publishers do not intend non-compliance with privacy requirements, but rather lose track of their obligations or are unaware of them, we also see potential for implementing privacy requirement analyses in software development tools and integrating them into the app vetting process in app stores.

### 5.3 Recommendation

Based on the conclusion, the following recommendations were made;

1. Detailed sections on data collection, usage and sharing should be looked for.
2. Ensure they align with app functionality
3. Verify how data is protected and stored
4. Understand if and how data is shared with others
5. Check options for managing data and opting out
6. See how the app informs users of policy changes.

## Reference


Alexander, I.F. & Maiden, N. (2014). Scenarios, stories, use cases: through the systems development life-cycle. John Wiley & Sons, 2014.

Alexandron, G. & Armoni, M, Gordon, M, Harel, D. (2023) “Scenario-based programming: Reducing the cognitive load, fostering abstract thinking.” 36th International Conference on Software Engineering, pp. 311-320, 2023.

Allenby, K & Kelly, T. (2021) “Deriving safety requirements using scenarios,” Fifth IEEE International Symposium on Requirements Engineering, pp. 228-235, 2021.

Alspaugh, T. A, Anton A. I, Barnes, T, & Mott, B. W. (1999) “An integrated scenario management strategy,” International Symposium on Requirements Engineering, pp. 142-149, 1999.

Anish, P. R, (2021), “Automated Identification and Deconstruction of Penalty Clauses in Regulation,” IEEE 29th International Requirements Engineering Conference Workshops (REW), pp. 96-105, 2021.

Ant´on, C. (1998) “A representational framework for scenarios of system use.” Requirements Engineering 3 (1998): 219-241.

Aoyama, Y. (2020) “Persona-Scenario-Goal Methodology for User-Centered Requirements Engineering,” 15th IEEE International Requirements Engineering Conference, pp. 185-194, 2020.

Balebako, L. (2020) “Improving app privacy: Nudging app developers to protect user privacy.” IEEE Security & Privacy 12(4): 55-58, 2014.

Barth, A. (2020), “Privacy and Contextual Integrity: Framework and Applications,” IEEE Symp. on Sec. & Priv., 2006, pp. 184-198.

Bauer, R. A (2021) “Consumer behavior as risk taking.” 43rd National Conference of the American Marketing Assocation, 2021.

Bhatia, T & Breaux, T. D. (2015) “Towards an information type lexicon for privacy policies,” IEEE Eighth International Workshop on Requirements Engineering and Law (RELAW), pp. 19-24, 2015.

Bhatia, T & Breaux, T. D. (2018) “Empirical Measurement of Perceived Privacy Risk.” ACM Transactions on Computer Human Interaction, 25(6): Article 34 (December 2018), 47 pages.

Bhatia, T & Breaux, T.D, Reidenberg, J.R, & Norton T.B. (2016). “A theory of vagueness and privacy risk perception.” 24th International Requirements Engineering Conference, pp. 26-35, 2016.

Carlsson, R., Heino, T., Koivunen, L., Rauti, S., & Leppänen, V. (2022, April). Where does your data go? comparing network traffic and privacy policies of public sector mobile applications. In *World Conference on Information Systems and Technologies* (pp. 214-225). Cham: Springer International Publishing.

Casillo, F, Deufemia, V, Gravino, C. (2022) “Detecting privacy requirements from User Stories with NLP transfer learning models,” Information and Software Technology, v. 146, 2022.

Graves, J. (2015). An exploratory study of mobile application privacy policies. *Technology Science*. Retrieved from https://techscience.org/a/2015103002/. Accessed 16th September, 2025

Harkous, H., Fawaz, K., Lebret, R., Schaub, F., Shin, K. G., & Aberer, K. (2018). Polisis: Automated analysis and presentation of privacy policies using deep learning. In *27th USENIX Security Symposium (USENIX Security 18)* (pp. 531-548).

Hashmi, S. S., Waheed, N., Tangari, G., Ikram, M., & Smith, S. (2021, November). Longitudinal compliance analysis of android applications with privacy policies. In *International Conference on Mobile and Ubiquitous Systems: Computing, Networking, and Services* (pp. 280-305). Cham: Springer International Publishing.

Kandil, S. A., van den Akker, M., van Baarsen, K., Jansen, S., & van Vulpen, P. (2018, June). Benchmarking privacy policies in the mobile application ecosystem. In *International Conference of Software Business* (pp. 43-55). Cham: Springer International Publishing.

Kollnig, K. (2021). Tracking in apps' privacy policies. *arXiv preprint arXiv:2111.07860*.

Kununka, S. (2019). *User centric privacy policy modelling*. The University of Manchester (United Kingdom).

Pan, S., Zhang, D., Staples, M., Xing, Z., Chen, J., Xu, X., & Hoang, T. (2024). Is it a trap? a large-scale empirical study and comprehensive assessment of online automated privacy policy generators for mobile apps. In *33rd USENIX Security Symposium (USENIX Security 24)* (pp. 5681-5698).

Paspatis, I., Tsohou, A., & Kokolakis, S. (2018). *AppAware: A Model for Privacy Policy Visualization for Mobile Applications*

Singh, R. I., Sumeeth, M., & Miller, J. (2011). Evaluating the readability of privacy policies in mobile environments. *International Journal of Mobile Human Computer Interaction (IJMHCI)*, *3*(1), 55-78.

Sun, Y. P. (2018). *Investigating the effectiveness of android privacy policies*. University of Toronto (Canada).

Wang, X., Qin, X., Hosseini, M. B., Slavin, R., Breaux, T. D., & Niu, J. (2018, May). Guileak: Tracing privacy policy claims on user input data for android applications. In *Proceedings of the 40th International Conference on Software Engineering* (pp. 37-47).

Yu, L., Zhang, T., Luo, X., Xue, L., & Chang, H. (2016). Toward automatically generating privacy policy for android apps. *IEEE Transactions on Information Forensics and Security*, *12*(4), 865-880.

Zhao, K., Zhan, X., Yu, L., Zhou, S., Zhou, H., Luo, X. & Liu, Y. (2023, May). Demystifying privacy policy of third-party libraries in mobile apps. In *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)* (pp. 1583-1595). IEEE.