CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Automatic Privacy Analysis of TCF-based Android Applications

Master's thesis in Computer science and engineering

Joel Ahlinder & Pontus Carlsson

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2025

Master's thesis 2025

Automatic Privacy Analysis of TCF-based Android Applications

Joel Ahlinder & Pontus Carlsson

UNIVERSITY OF GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2025

Automatic Privacy Analysis of TCF-based Android Applications
Joel Ahlinder & Pontus Carlsson

Supervisor: Victor Morel, Computer Science and Engineering
Examiner: Romaric Duvignau, Computer Science and Engineering

Master's Thesis 2025
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2025

Automatic Privacy Analysis of TCF-based Android Applications
Joel Ahlinder & Pontus Carlsson
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

Being greeted by a banner or consent dialog asking: “manage cookies” or “accept all” has become the norm for Europeans when browsing websites or using mobile applications in recent years due to regulations, such as the ePD and the GDPR. To help data controllers conform to these standards, the IAB created the TCF in April 2018. This framework has previously been found to cause several privacy violations when used on websites, and has therefore been updated regularly since. Previous research on the TCF has only been conducted in web contexts, therefore the aim for this thesis is to research the framework’s usage in Android applications. Our goals for this thesis are to determine the prevalence of the TCF in the Google Play Store, to confirm if popular Android apps that implement the framework respect users’ consent dialog choices, and to quantify the presence of `cookie paywalls`. To reach our goals we develop solutions to: 1) scrape and download 4 482 of the most popular Google Play Store apps on an emulated Android device, 2) automatically determine which apps use the TCF, 3) automatically interact with applications’ consent dialogs while simultaneously determining the presence of `cookie paywalls`, and lastly, 4) analyze applications’ traffic in two different stages. We find that 842 applications in our dataset implement the TCF, and that it is possible to interact with consent dialogs of 576 apps, with 15 apps only storing users’ dialog choices if the users provide full consent. In the 576 apps we find four `cookie paywalls`, proving their existence in Android applications. From analyzing apps’ traffic, we find that 66.5% of apps transmit personal data when provided with no consent and no legitimate interest, and 55.4% of apps transmit personal data during interactions with apps’ consent dialogs. These results imply that TCF-based apps potentially violate the GDPR.

Keywords: Android, tracking, consent dialog, privacy, TCF, GDPR, legitimate interest

Acknowledgements

We want to thank our supervisor Victor Morel for proposing the idea for this thesis, assisting us when we had questions, and always being available. We would also like to thank Romaric Duvignau for examining this thesis and providing valuable feedback throughout the project.

Joel Ahlinder & Pontus Carlsson, Gothenburg, 2025-06-30

List of Acronyms

Below are the acronyms used in the report, listed in alphabetical order.

AAID	Android Advertising ID
adb	Android Debug Bridge
CMP	Consent Management Platform
EEA	European Economic Area
ePD	ePrivacy Directive
EU	European Union
GDPR	General Data Protection Regulation
IAB	Interactive Advertising Bureau
LI	Legitimate interest
OCR	Optical Character Recognition
SSL	Secure Sockets Layer
TCF	Transparency and Consent Framework
TLS	Transport Layer Security

Contents

List of Figures

List of Tables

1

Introduction

Being greeted by a banner or consent dialog asking: “manage cookies” or “accept all” has become the norm for Europeans when browsing websites or using mobile applications in recent years. An example of this can be seen in Figure 1.1. This trend is largely due to data protection laws, such as the ePrivacy Directive (ePD) combined with the General Data Protection Regulation (GDPR). Since the introduction of the ePD and the GDPR, data controllers like mobile applications and website owners are required to specify a legal basis for data processing. Two legal bases in particular, namely legitimate interest (“processing is necessary for the purposes of the legitimate interests pursued by the controller or by a third party”, according to Article 6.1(f) of the GDPR [1]) or users’ consent (freely given affirmative action, according to Article 4.11 of the GDPR [2]), are widely used options when collecting any non-necessary cookies [3].

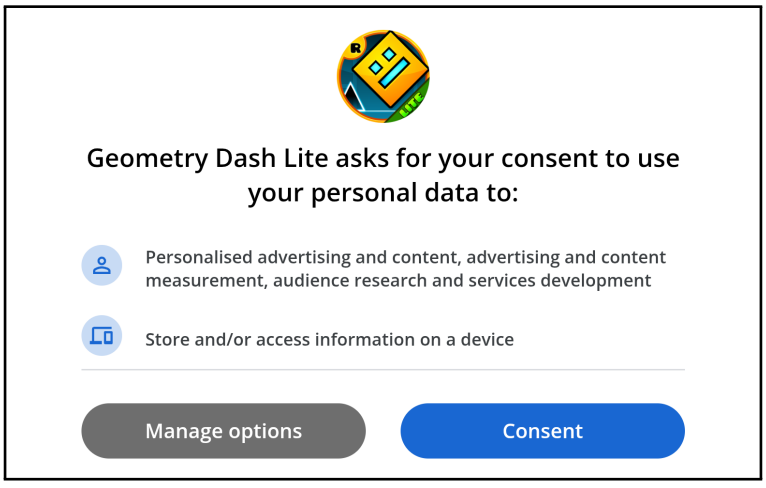

Figure 1.1: Example of a Google LLC (CMP 300) consent dialog from application Geometry Dash Lite (com.robtopx.geometryjumplite).

To help data controllers collect data according to the standards set by the ePD and the GDPR, the Interactive Advertising Bureau Europe (IAB) created the Transparency and Consent Framework (TCF) in April 2018 [4]. This framework is voluntary and aims to make it easier for data controllers to comply with the data collection requirements set by the GDPR.

However, the goal of the TCF to facilitate legal compliance does not ensure it.

Websites using old versions of the TCF (v.1.1) have previously been found to cause several potential privacy violations despite using the framework [5]. The framework has regularly received updates since, and in the middle of May 2023, the TCF was updated to version 2.2, with TCF participants having to update their implementations by late November 2023. One outcome of the update was that legitimate interest was removed as a legal basis when processing data for the purpose of creating personalized advertising and content [6]. Previous research has shown that a majority of the legitimate interest purposes found on websites were for third party vendors, such as advertising partners, with all found websites using the TCF [7]. Therefore, it is of interest to examine whether similar legitimate interest purposes are present in Android applications, as that is no longer allowed.

Recent academic literature has also observed the proliferation of new techniques and business models, in order to collect personal data for advertising purposes while attempting to conform to legal requirements based on the TCF. One such technique is `cookie paywalls` [8], which only allow visitors to access content after paying a fee or accepting tracking. `Cookie paywalls` have so far only been studied in web contexts.

A majority of web traffic comes from mobile devices [9], and for the last five years, Android has been the most widely used mobile operating system [10]. Previous research on the GDPR in Android applications has found that 28% of 86 000 apps potentially violated the GDPR by sending personal user data to ad providers before receiving consent [11]. Furthermore, it has been prevalent for TCF-based websites to contain potential privacy violations [5], which could extend to TCF-based Android applications. Despite this, almost no research has been conducted on TCF-based Android applications, and none at all on their use of legitimate interest, we therefore aim to fill this gap.

1.1 Aim

Considering the previously discussed context, regarding the lack of research on TCF-usage in Android applications, previous privacy violations on TCF-based websites, and the changes to legitimate interest in the updated version of the TCF, we address the following research questions:

RQ1 How prevalent are TCF-based applications in the Google Play Store?

RQ2 Do TCF-based applications respect users' dialog choices and the TCF's restrictions of legitimate interest?

RQ3 Do TCF-based Android applications respect users' choices regarding consent and legitimate interest at runtime?

RQ4 How prevalent are cookie paywalls in TCF-based Android applications?

1.2 Scope

This thesis only targets TCF-based Android applications and no applications on other operating systems, such as iOS.

We crawl the top apps of the Google Play Store until we have at least 5 000 applications, creating our own dataset ensures that it is up-to-date and created in an ethically correct manner. All our results are based on data gathered from applications in this dataset, which may not correctly represent all implementations of the TCF, but it does represent the implementation on popular and commonly used apps in the Google Play Store between 2025-01-29 and 2025-03-10.

Previous papers have studied the legality of `cookie paywalls` and if their implementation adheres to the GDPR and the TCF. Our paper only confirms the potential existence of `cookie paywalls` found in TCF-based Android applications and report the prevalence of these, within our created dataset.

1.3 Outline

The remainder of this thesis is structured in the following way. Chapter 2 covers background information necessary for the understanding of this thesis by describing relevant technical and legal information used during the development of this thesis. We mention in Chapter 3 various previous research papers that relate to the topics of this thesis and the foundation we build upon. Chapter 4 describes the methodology used to answer our research questions, and Chapter 5 present the results gathered when answering the research questions. In Chapter 6 we discuss our results from the previous chapter, we also cover ethical aspects and limitations encountered during the thesis. Lastly, we conclude in Chapter 7 our work and present some potential future work.

In this thesis we make the following contributions:

- We quantify the usage of the TCF in popular Android applications, reporting a significant increase compared to the related work (from 6.6% [12] to 18.8%).
- We present a methodology to automatically interact with TCF-based applications' consent dialogs in three different approaches, using tools previously mentioned by Koch et al. [12].
- We perform the first systematic study of TCF-based Android applications' processing of users' personal data, using traffic analysis. Our results indicate that several applications do not store users' dialog choices (15), and a large majority of the studied apps even forward personal data to adtech servers against users' choices (66.5%), suggesting that **the TCF fails to facilitate legal compliance**.
- We identify the first instances of `cookie paywalls` in Android applications.

2

Background

This chapter introduces legal background related to the TCF, such as the GDPR and the ePD, but also tracking. The chapter also introduces the necessary technical background to understand our analysis of Android applications.

2.1 The GDPR and the ePD

The GDPR was put into effect on May 25th, 2018, and is considered the most rigorous privacy regulation in the world [13]. The regulation was created and passed by the European Union (EU) but still affects organizations anywhere in the world that attempt to process personal data (e.g., collecting, transmitting, or using any data, which can directly or indirectly identify an individual [2]) from people in the European Economic Area (EEA) [14]. To protect personal data and privacy rights in the EEA, the GDPR places requirements on data processing through obligations on data controllers, and by providing rights for data subjects [15]. One example of an obligation on data controllers is the requirement to use legal bases – such as legitimate interest and consent –, when processing personal data [1].

A typical example of personal data that is regulated by the GDPR are cookies, which are small text files that may be stored on a computer or a mobile device and which contain data related to a website you visit. They are however not only regulated by the GDPR, as the ePD also restrict their processing. Cookies are referenced once in the GDPR, where it is stated that they can be used to identify users, and are therefore subject to the regulation [3].

The ePD is a set of rules which were introduced in 2002, regulating cookie usage for websites and mobile applications, and other aspects of data privacy in the EU. The ePD requires data controllers to obtain users' consent before storing non-necessary cookies. This is the reason consent dialogs appear on many websites and mobile applications, and why the ePD is called the "cookie law" [16].

To follow the GDPR and the ePD governing cookies, data controllers must comply with the following [3]:

1. Receive users' consent before using any non-necessary cookies.

2. Provide accurate and specific information about what data each cookie tracks and the purpose of the cookies, before consent is received.

3. Users should be allowed to access services even if they decline cookies.

2.2 The Transparency and Consent Framework

The TCF was created in April 2018 by the IAB, with the goal of helping data controllers process data according to the standards set by the ePD and the GDPR. It was developed for use by three different stakeholders. 1) Publishers of online content where data can be collected, 2) data vendors which process user data gathered by the publishers, 3) and Consent Management Platforms (CMPs) which develop consent dialogs. CMPs provide the consent dialogs that users interact with. They present the different vendors that can access user data, the purposes these vendors process data for, and the legal bases utilized for these purposes [4].

The TCF defines eleven standard purposes, each requiring either consent or legitimate interest as a lawful basis to process data, and these are explained in Table 2.1. All the purposes can be opted into when interacting with applications' consent dialogs. In versions prior to TCF 2.1, legitimate interest could be used as a lawful basis for all the purposes. As of version 2.1, purpose 1 was changed to disallow legitimate interest as a lawful basis, and as of version 2.2, purposes related to personalization (purposes 3,4,5,6) were also changed to disallow legitimate interest as a lawful basis [17].

Nr	Description
1	Store or access data used to recognize devices
2	Present advertising to the user based on e.g., non-precise location
3	Process data about a user's activity in the app combined with data from outside the app to create personalized advertising profiles
4	Present personalized advertisements based on the user's advertising profiles
5	Process data about a user's activity in the app combined with data from outside the app to create personalized content profiles
6	Present personalized content based on the user's advertising profiles
7	Gather data regarding the advertisements shown to the user, e.g., if the user saw the ad, and if the user interacted with it
8	Gather data regarding the content shown to the user, e.g., if the user interacted with the content, and for how long the user interacted with it
9	Generate reports based on the user and other users' data, regarding their interactions with advertisments or content
10	Use data from the user's activity on the app to improve the app and the content it presents
11	Present content based on e.g., non-precise location and content the user has interacted with in the app

Table 2.1: TCF purposes, where purposes 1, and 3-6 require consent [17].

When a user interacts with websites or mobile applications which uses the TCF, a TC string will be generated on the user's device containing the user's dialog choices.

When interacting with websites the TC strings are stored as HTTP cookies in the user's browser, and on mobile devices the strings are stored locally, which will be further explained in Section 2.4. Data vendors have access to the string and should therefore only process data according to the user's choices [18], [19], [20].

2.3 Tracking

Tracking data involves selecting metrics and events to track, and then analyzing the data to gather valuable insight into consumer behavior. These insights can then be used to change the website or application to better suit customer preferences, but it can also be used to tailor ads to users [21].

Libert [22] explains how users rarely know when websites are tracking them. The paper explains how websites are built using HTML, which makes it simple to add elements from third-parties, such as images and JavaScript code. When a user loads a website using elements from third-parties, the user will send HTTP requests both to the website and to the third-parties hosting the elements. The request to the third party will include data about the user, such as IP-address, browser, operating system, and the address of the website which initiated the request. After collecting enough such data, behavioral patterns of users can be created. Companies can then use these patterns to target users with specific products and services to increase monetary gain.

Previous research by Kyi et al. [7] and Altpeter [23] has found that mobile applications and websites implement deceptive designs to simplify collection of more data about users. Examples of deceptive designs are consent dialogs not including reject options on the first page, or using contrast ratios and coloring to highlight the accept all button compared to other buttons [24].

2.4 Android Studio and Android Debug Bridge

Android Studio is the official IDE made by Android for development of Android applications. To test applications, Android Studio allows users to emulate Android devices on their computer. The emulated devices are similar to regular Android devices and can include the Google Play Store if specified [25], [26], [27].

Android Debug Bridge (`adb`) is a command-line tool that lets users communicate with emulated and real Android devices through a Unix shell. The tool allows users to interact with their device both through Unix- and `adb`-commands, which enables users to programmatically run specified apps. `Adb` also makes it possible to monitor currently running activities and processes on the device. `Activities` on Android are windows in apps which often fills the screen, but sometimes floats on top of other `activities`, such as pop-ups [28]. The ability to monitor currently running `activities` and processes is very important for the dynamic and traffic analysis of applications in this project [29].

Since the `adb` uses the command-line, it can be used to automate interactions with

the connected Android device. `Adb` can be given superuser privileges with device rooting applications like Magisk [30], such privileges allow users to access the full file system of the Android device through `adb` commands. This particular functionality is important for this project as it allows for access to the `SharedPreferences`[1] directory of applications, which contains information about applications' usage of the TCF, including applications' TC strings [29], [31].

2.5 Selenium and Appium

Selenium is a toolset for automation of web browsing which makes it possible to emulate a user's interactions with the browser. Through the Selenium WebDriver library it is possible to automatically identify and interact with website elements, perform mouse movements, and execute JavaScript code, among other features [32].

The most important Selenium features for this project are the possibilities to scroll and to interact with website elements. These features allow for simple extraction of all links on a webpage.

Appium is an open-source project designed to simplify UI automation of applications, which also allows for usage of the Selenium WebDriver library on platforms other than browsers. The initial targets for Appium were mobile applications (on iOS and Android) though the project has since expanded to handle more unique app platforms, such as TV platforms [33]. Appium is central for this project, as it can identify and interact with elements visible on screen in applications and therefore allows us to programmatically interact with consent dialogs.

2.6 HTTPS and mitmproxy

HTTPS is the secure version of HTTP which is the main protocol used for communication over the internet. HTTPS uses encryption protocols to increase security, either Transport Layer Security (TLS) or Secure Sockets Layer (SSL), the latter being deprecated [34]. To further increase security and prevent the possibility that adversaries place themselves in the middle between clients and the server with a proxy to intercept traffic, applications can also implement SSL pinning. SSL pinning is a technique which – in the app –, stores the SSL or TLS certificate's public key of the server that the app is expected to communicate with. Once the app communicates with a server, it will compare the stored public key and the server's certificate's public key, if the keys match, the communication will be considered secure [35].

Mitmproxy is a suite of tools that provides an interactive HTTPS-capable interceptor proxy, which is an intermediary between the client and the server [36], allowing monitoring of network traffic [37]. Mitmproxy also contains an important feature that allows users to provide Python scripts to customize mitmproxy's behavior called addons [38]. This feature is important as addons can hook to given events – e.g., HTTP responses –, and then execute when the events are triggered.

[1] Directory with files containing collections of key-value pairs for an Android application

2.7 Optical Character Recognition and Sentence Similarity

Optical Character Recognition (OCR) allows for conversion of images and documents containing text to a format which is readable for machines. OCR works by first turning the image into a fully black and white format, the image is then ran through an algorithm using pattern recognition or feature extraction. Pattern recognition looks for patterns in characters in the image, based on what has previously been seen in a database with possible characters. Feature extraction splits the characters into features – such as lines –, to then match those features against potential characters in a database [39]. Depending on the software used, OCR can work in many different languages, such as the Python package EasyOCR [40] which supports more than 80 languages.

Sentence similarity makes it possible to determine how similar different texts are. By converting input texts into vectors which contain semantic information about the texts, the distance between the vectors can then be calculated to determine their similarity, with a shorter distance implying a stronger similarity [41]. There are multiple ways of computing the distance between the vectors, one such method is cosine similarity. Cosine similarity works by calculating the cosine of the angle between the vectors, where a smaller difference in angle means a smaller distance between the vectors [42].

3

Related Work

This chapter provides an overview of previous research which has shown earlier versions of the TCF producing potential GDPR violations in web contexts, legitimate interest being used on websites to extract more user data, and potential GDPR violations found in Android and iOS mobile applications. We plan to build upon this existing work in our thesis.

3.1 Potential GDPR Violations and Previous TCF Research

Bollinger et al. [43] analyzed cookie banners on 30 000 websites, 94.7% of these websites had at least one potential GDPR violation [43]. This paper shows that a majority of investigated websites were potentially violating the GDPR in 2022.

Matte et al. [5] built a crawler to specifically find websites using the TCF for their cookie banners. 1 426 of the crawled 22 949 websites were using the TCF. Extensive tests were performed on 560 of these websites and 54.0% of them were potentially violating both the GDPR and the ePD. The violations included: websites sharing user consent between one another through "shared cookies", and websites storing positive consent before giving users a choice by having predetermined non-empty TC strings, etc. The paper showed that the TCF was flawed in 2020 (v.1.1), and even though there have been updates since, these previous findings indicate that the framework could still contain flaws allowing privacy violations.

Smith et al. [44] studied websites using TCF version 2.1 in 2023, and is the most recent study we could find of GDPR compliance on websites using the TCF. The authors denied consent in consent dialogs of 2 230 websites containing the TCF through a crawl where the websites were selected from Deepsee.io [45]. From interacting with the websites, the paper found that a majority of the websites were using legitimate interest for purposes 3-6, which are used for personalization (this usage of legitimate interest is not allowed as of the current TCF version 2.2). The results of analyzing the TC strings after the crawl were that 1.3% of the researched websites had potential GDPR violations, such as: legitimate interest being used for purpose 1 or TC strings stating that consent was given.

In 2023, Morel et al. [8] studied the usage of `cookie paywalls` on websites, which offers users the options to either 1) pay a fee or 2) accept all tracking to access a

website. The authors found that usage of `cookie paywalls` always involved the TCF. Morel et al. also found that some webpages used legitimate interest to track users until manual objection, even if they chose to pay the fee, thereby breaking trust with users. This shows that legitimate interest has previously been used to violate privacy.

3.2 Legitimate Interest and User Perception of Tracking and Consent

Kyi et al. [7] crawled websites in 2022, to research usage of legitimate interest. The authors crawled 10 000 websites, where 474 of them were presented in English and used legitimate interest. All 474 websites were using the TCF and were therefore using the legitimate interest purposes provided by the TCF. A majority of the 474 websites were using legitimate interest to collect data for third party vendors – such as advertising partners –, which at the time was allowed according to the then-active TCF version. Lastly, the paper issued a survey related to legitimate interest with 400 attendees. The results of the survey were that the attendees disapproved of how legitimate interest was being used to access and use their data.

In 2014, Libert [22] analyzed tracking on the top one million websites, the results showed that nine out of ten websites tracked users unknowingly, with Google tracking users on eight out of ten websites. To understand what the American people thought about tracking, Turow et al. [46] issued a survey with 1 500 American participants in 2015. One of the statements the participants were presented with in the survey, is the following: “If companies give me a discount, it is a fair exchange for them to collect information about me without my knowing.”. In simpler terms, is it fair for a company to collect a user’s personal data without the user’s consent, if the company provides the user with product discounts. 91.0% of the survey participants disagreed with this statement.

Both the surveys from Kyi et al. [7] and Turow et al. [46] show that users disapprove of their data being accessed and used, without having explicitly agreed to give up that data, even when provided with benefits.

3.3 Potential Privacy Violations in Mobile Applications

Between 2021 and 2022, Nguyen et al. [11] conducted the first study of potential GDPR violations of explicit consent in Android mobile applications. The authors initially crawled one million applications over five months, of these apps, 86 000 apps were selected to be analyzed. To analyze the outgoing traffic of applications, mitmproxy and Objection [47] were used. By using these tools the authors could confirm when applications sent personal data – such as advertising IDs and location data –, without having granted explicit consent. The results of the paper were that 28% of the studied applications potentially violated the GDPR.

Koch et al. [12] researched consent dialogs between 2022 and 2023 in Android and iOS applications. The studied Android applications were scraped from the top lists of all categories in the Play Store, where the authors used the top 100 apps of each category, making their Android app dataset contain 3 006 apps. Out of these apps, the paper found that 6.6% of the applications used the TCF. To download the scraped applications, the authors utilized the open-source tool `PlaystoreDownloader` [48], which is no longer maintained.

After downloading the applications, they performed a static analysis to assess whether a majority of the applications used the same CMPs. Other than the TCF-compliant CMPs, most applications used their own solutions to present their data practices to the user. Therefore, the authors decided to interact with applications using Appium. With Appium, they were able to classify how the apps presented their consent dialog when launched, where the apps were classified as using links, notices, or proper dialogs. Of the three, only the proper dialogs required users to interact by consenting or denying data collection before using the app, while the others stated that data will be collected if the user continues to use the app.

Lastly, the authors analyzed the apps' traffic, similarly to the methods presented by Nguyen et al. [11]. The analysis was performed by comparing applications' transmitted data with what the authors had consented to in the applications. If the data could be tied to the authors, without given consent, then there was a potential breach of the GDPR. The results of the paper were that 43.1% of total studied apps on Android and iOS potentially violated the GDPR.

The paper by Koch et al. is based upon the work of Altpeter [23], which performed a similar study in 2022. In the paper, a dataset of 3 421 Android apps were analyzed, with 2.6% of apps using the TCF. The paper also analyzed iOS applications, and the results were that 77.3% of total studied apps on Android and iOS potentially violated the GDPR.

4

Methodology

In this chapter, we describe the methodology used to answer our research questions regarding TCF-based Android applications. We begin by explaining how we scraped the Play Store and downloaded apps, addressing **RQ1**. Furthermore, we describe how we analyzed TCF-based applications through dynamic and traffic analysis – to establish their respect of users' dialog choices –, to answer **RQ2** and **RQ3**. Finally, we detail how we determined the prevalence of `cookie paywalls` in our dataset, addressing **RQ4**.

4.1 Quantifying the Presence of the TCF

We initially created a dataset of Android applications by scraping the Play Store, with the dataset consisting of application package names, application names, application category and the date when we scraped the application. To create the dataset we used Selenium with Python [49]. The Play Store has top charts for each category: `https://play.google.com/store/apps/category/CATEGORY` which contain between 60–900 applications, depending on the category. We scraped all categories daily between 2025-01-29 and 2025-03-10, until we had 5 067 unique applications in our dataset.

Once we had a working scraper, we created an emulated Android device through Android Studio and allocated it with 300 GB of storage. The device was selected to be a Google Pixel 9 Pro using API 35. Google Pixel 9 Pro was the newest available device in Android Studio and API 35 was a recent stable release that worked well based on some preliminary testing. To download Android applications, we initially tried open source tools, such as `PlaystoreDownloader` [48]. All the tools we found were however deprecated, and we could not get them to function. Since we were unable to find an existing solution for downloading applications, we created an ad-hoc solution using `adb`. The Play Store application worked on the emulated device, and we could therefore automatically download applications by booting the Play Store with apps from our dataset, through a Python script using Appium. From the Play Store page we could find an app's download button through Appium and click it, the script then waits until the application is fully downloaded – by inspecting the file system of the emulated device –, before downloading the next application.

When applications were downloaded, we manually set our device's GPS coordinates to Sweden and had a Swedish IP, to confirm that the GDPR applies to our device. These coordinates could then be seen in the device's fused location which is described

by Google as a fuse of underlying location technologies, such as GPS and WI-FI [50]. We accessed the device's coordinates by using the `adb`-command `dumpsys location`.

According to the IAB CMP API, files in the `SharedPreferences` directory of applications should contain strings, such as `IABTCF_TCString` if the applications are using the TCF [19]. To generate the `SharedPreferences` directory of applications, we launched the apps and closed them by using the `adb` commands, `monkey` and `am force-stop` through a Python script. To confirm if applications were using the TCF, the script then executed the Unix-command `grep`[1] on all files in the apps' generated `SharedPreferences` directory, searching for mentions of `IABTCF`.

4.2 Dynamic Analysis

To dynamically analyze the TCF-based applications, we used Appium and `adb`. Appium makes it possible to launch applications and automatically interact with their consent dialogs. The design of consent dialogs are based on what CMP the application uses, we therefore initially researched the most common CMPs in our dataset. In a majority of applications, the CMP could be found in the app's `preferences.xml` file in its `SharedPreferences` directory by looking at the string `IABTCF_CmpSdkID` if not, we had to manually launch the applications and inspect their consent dialogs to look for patterns. Some applications had their CMP mentioned in other files in their `SharedPreferences` directory and some applications had their CMP included in their application package name, for example `com.outfit7.mytalkingtom2`, where the CMP is Outfit7. Once we knew what CMPs the applications were using, we manually inspected the consent dialog of some applications using each CMP and created cases for how the different CMPs were to be interacted with through Appium with Python. A majority of cases consisted of waiting for certain elements to show up on the mobile screen and then scrolling through the consent dialog to interact with buttons and togglebuttons based on the buttons' content-description- or text-elements. To select which elements we interacted with, we read all of the text on elements currently present on screen and used their content-description- or text-attributes as input into a Python function using the multilingual sentence similarity model `paraphrase-multilingual-MiniLM-L12-v2` [51], [52]. The function also takes a list of expected texts and then outputs the element which is most likely to be the one we want to interact with, if producing a similarity score above a threshold.

To boot applications with Appium, the package name of the application and its launch `activity` needs to be specified. We had data of all the application package names from scraping and downloading, but we had no data on the application launch activities. We therefore initially created a Python script which ran all applications, confirmed their launch activities through `adb` with the command `dumpsys activity activities` and then stored their launch `activities` in a CSV file, also containing the application package names and their CMP.

[1] Command which searches for patterns in a file.

The most commonly used CMP in our dataset is Google LLC (CMP 300), and is shown in Figure 4.1. This CMP allowed us to interact with consent and legitimate interest in a standardized manner after clicking “Manage option”. The figure shows the purpose “Measure advertising performance” and how it includes both legitimate interest and consent, where consent always follows an opt-in basis and legitimate interest follows an opt-out basis.

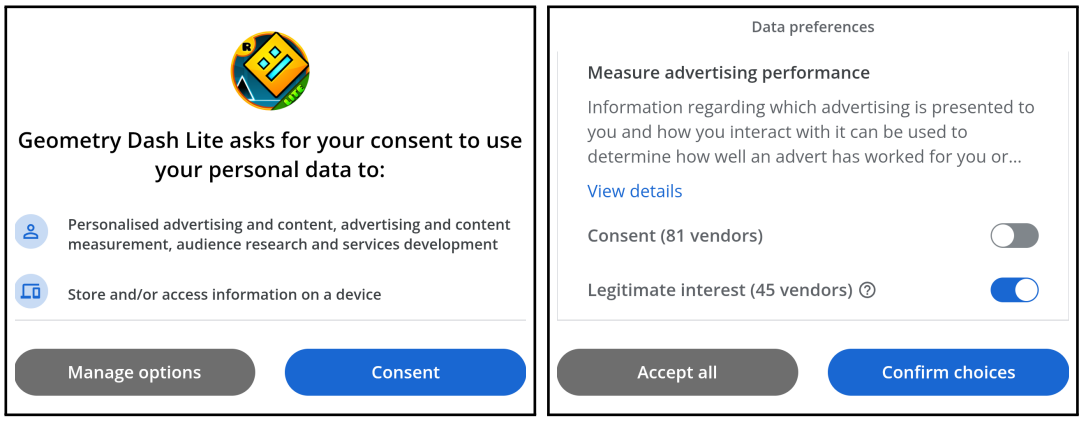

Figure 4.1: Consent dialog settings for Google LLC (CMP 300) consent dialog from application Geometry Dash Lite (com.robtopx.geometryjumplite).

Other CMPs were less standardized or had various extra steps to reach the options containing consent and legitimate interest, an example of this is shown in Figure 4.2. This CMP requires the user to input their age before any consent dialog becomes available, and the consent dialog is only available if the user is at least 16. If the user is younger than 16, the application automatically stores no consent and no legitimate interest. In Figure 4.3 another CMP is shown where the only visible button is “Accept”, which consents to all data purposes. To reach settings related to legitimate interest we had to click the “legitimate interest”-hyperlink.

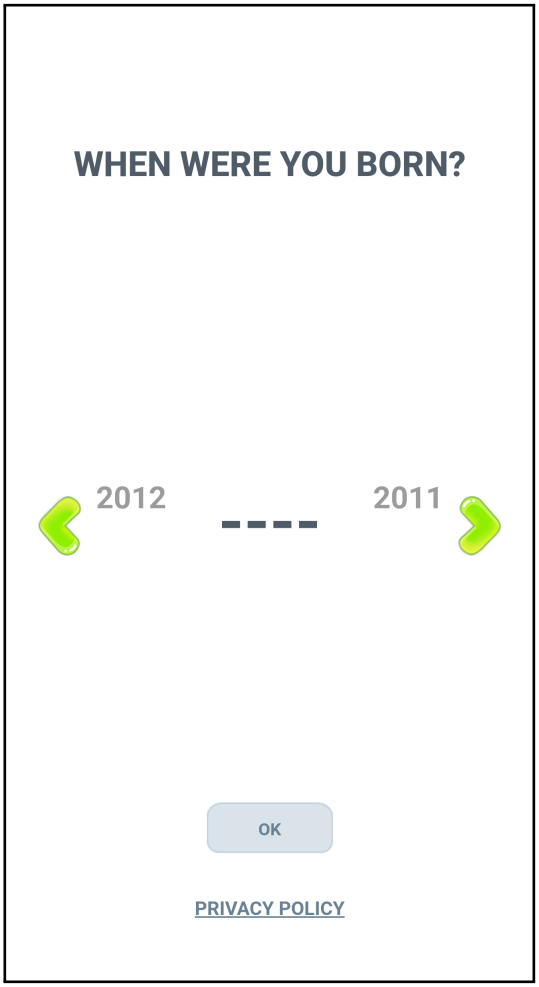

Figure 4.2: Outfit7 (CMP 348) dialog from application My Talking Tom 2 (com.outfit7.mytalkingtom2).

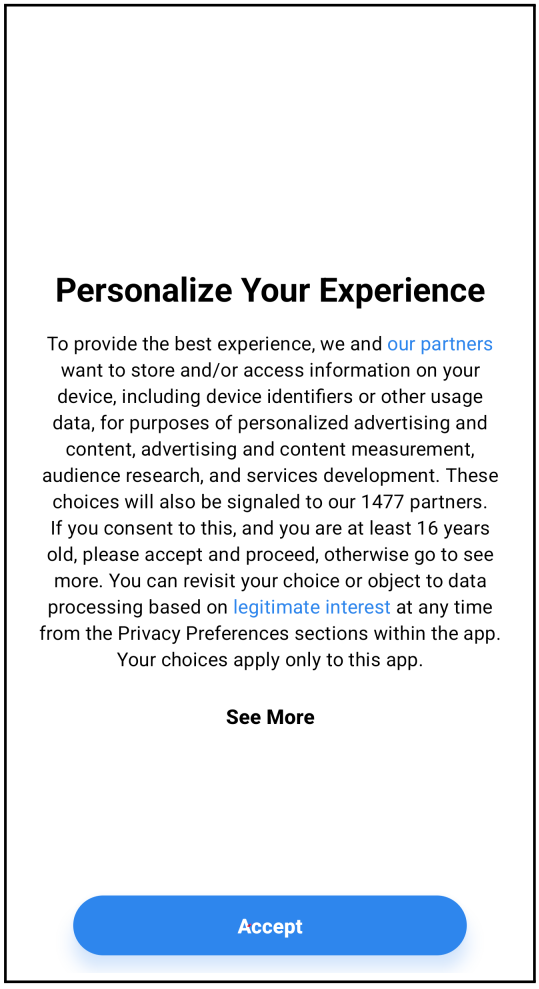

Figure 4.3: Easybrain (CMP 350) dialog from app Sudoku.com - Classic Sudoku (com.easybrain.sudoku.android).

We decided to interact with consent dialogs through three different approaches: 1) agree to everything (including legitimate interest), denoted **C+LI**; 2) only accept legitimate interest, denoted **LI**; 3) consent to nothing and accept no legitimate interest, denoted **Ø**. With the first approach we would be able to analyze the traffic of applications and determine what personal data is being transmitted to different servers when permission is given by the user. With the **LI**- and **Ø**-approaches we wanted to analyze if any disallowed legitimate interest purposes were in use or any purposes other than the purposes we had accepted. We also wanted to use the **Ø**-approach to analyze the traffic of applications and compare the data being transmitted with no consent given, versus the data being transmitted with full permissions granted, as in the **C+LI**-approach. We decided that an approach where we provide consent without legitimate interest would be meaningless, as this has no functional difference from the **C+LI**-approach where we agree to everything. Another explanation of the three different approaches can be seen in Table 4.1.

Approach	Consent	Legitimate Interest
C+LI	✓	✓
LI	**X**	✓
Ø	**X**	**X**

Table 4.1: Visualization of the legal bases accepted in consent dialogs during each interaction approach.

The three different approaches required interacting with each CMP in three different ways. For the **C+LI** approach it would often suffice to instantly press an "accept all"-button. For the **LI**-approach we would find the button which allows us to manually change our dialog choices and then confirm the choices which were automatically given to us. This method worked as legitimate interest is on an opt-out basis and consent is on an opt-in basis, meaning that the automatically given choices will most often only contain legitimate interest. For the **Ø**-approach we would have to find the button for manually changing our choices and then opt-out of every purpose related to legitimate interest. In most cases, this involved scrolling through a consent dialog to find and click six or more different buttons belonging to different purposes.

Once we had interacted with consent dialogs, we confirmed what choices (regarding legitimate interest and consent) the applications stored in their `preferences.xml` file. This was done through a Python script utilizing the Unix-command `grep`, looking for the strings, `IABTCF_PurposeConsents` and `IABTCF_PurposeLegitimateInterests` of applications. These strings contain information on which of the 11 purposes the application is using, and should be based on how the user interacted with the consent dialog, see Figure 4.4. We stored results from our interactions with consent dialogs in a CSV file and could confirm if applications were storing incorrect amounts of purposes or not, while also confirming if applications were using disallowed legitimate interest purposes according to version 2.2 of the TCF.

```
<string name="IABTCF_PurposeConsents">00000000000</string>
<string name="IABTCF_PurposeLegitimateInterests">00000000000</string>
```

Figure 4.4: Example of app having no active purposes in its preferences xml-file.

During execution of the dynamic analysis we encountered apps that needed to be updated in the Play Store to be analyzed properly. These necessary updates often covered the app's consent dialog with a pop-up, or crashed the app instantly, halting us from executing the dynamic analysis on the app. To solve this issue, each app that fails during dynamic analysis is rerun once. When we rerun an app, we check for updates by opening the application in the Play Store and looking for an update button by using Appium. If we manage to update the app, we open the app again, in search for a consent dialog to interact with. Other than updates hindering our interactions with consent dialogs, we were also presented with different pop-ups – often ads –, covering an app's consent dialog. These pop-ups

were handled by recognizing the name of the `activity` related to the pop-up and closing it, by tapping at the correct coordinates, examples of such pop-ups can be seen in Figure 4.5 and Figure 4.6. If the pop-up's `activity` was unknown, we instead tried to put our expected `activity` containing the consent dialog at the top of all activities. To change the order of `activities` we used the `adb`-command: `am start -n com.our.package/.our.activity`.

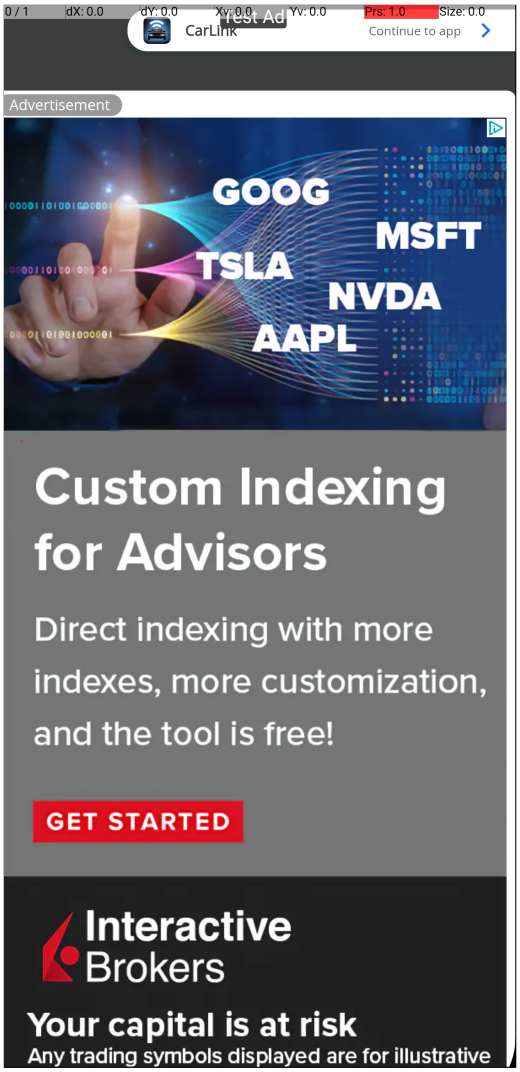

Figure 4.5: Ad pop-up from application CarLink: MirrorLink & Car Sync (com.cooldev.carplay.castcar).

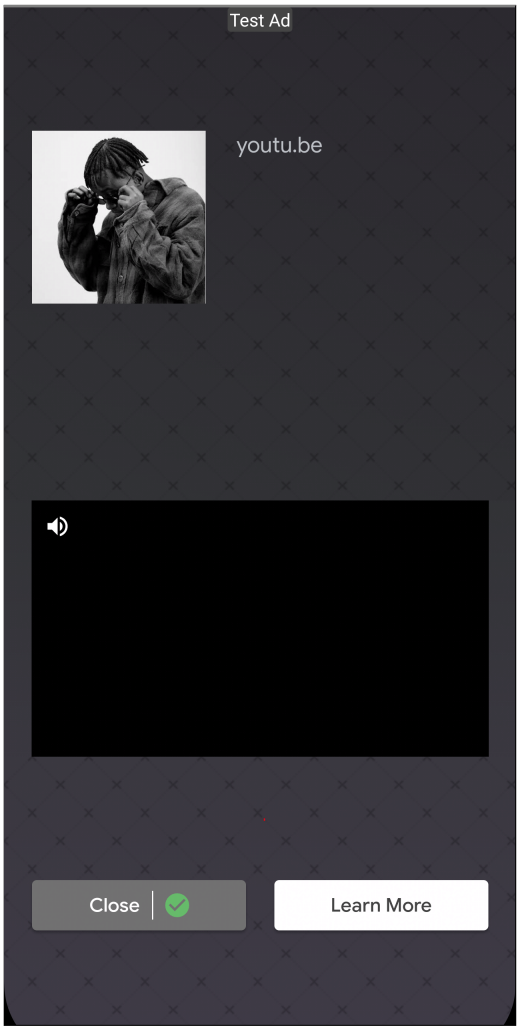

Figure 4.6: Ad pop-up from application Easy Piano Keyboard - Piano88 (com.banix.piano.learnpiano).

4.3 Traffic Analysis

Once we had interacted with consent dialogs of applications we had to analyze their traffic to confirm the apps' respect of our dialog choices. To perform the traffic analysis we utilized the mitmproxy tool suite to capture traffic, and Objection, to disable SSL pinning of applications.

We initially had to setup our device to function with mitmproxy and Objection. It was required by mitmproxy that our device installed and trusted mitmproxy's certificate authority [53], this allowed mitmproxy to decrypt encrypted traffic. The certificate authority was downloaded on our device through a Magisk module [54]. Objection is a toolkit powered by Frida [55], which therefore required us to install a Frida server on our device.

To analyze traffic we initially made sure the mitmproxy command-line functionality `mitmdump` is launched with a Python script of our choice as an addon. The addon running alongside `mitmdump` searched through the payload and URL of each HTTP-request on our emulated device for personal data. To identify personal data in requests we used hard-coded strings, strings based on running `adb` commands to extract present data on the device – such as Android Advertising ID (`AAID`) and location data –, and regular expressions for data, such as the device's phone number and MAC address, due to the different formats this data could be presented in. If any personal data was found in a request, we stored the following in our Postgres-database: the application package name, the personal data being transmitted and the request method, the request-url, the type of personal data being transmitted, and the date of the request. The types of personal data we tried to identify in requests can be seen in Table 4.2.

Personal Data	**Description**
AAID	Android Advertising ID
Email	Email address used on device
Phone number	Phone number used on device
IMEI	Hardware ID of mobile device
MAC	Device Wi-Fi MAC address
GPS	Latitude and longitude location of the device
IMSI	SIM-card identification
ICCID	SIM-card serial number
Phone serial number	Unique device identifier
Public IP	Can be used as an identifier

Table 4.2: Personal data considered during the traffic analysis.

When analyzing traffic we also had to set the emulated device's `http_proxy` global setting, which ensures that all traffic passes through the mitmproxy server. After confirming that `mitmdump` and the emulator was running, we then launched each application through Objection with the startup-command `"android sslpinning disable"` – which allowed for traffic collection in apps –, and let the app remain idle for 40 seconds, before launching the next one. Certain applications would detect SSL pinning being disabled and would sometimes crash, we worked around this by continuously confirming that the application was running by monitoring running processes on the device. If the process related to the current application was terminated preemptively, we re-executed the app up to two times, before labeling the app as non-analyzable and moving on to the next application.

Occasionally when apps crashed, our device would also crash or reboot. These reboots would often take a lot of time. To work around this we used snapshots [56], a funtionality of the emulated device in the form of a preservation of the device's state at the time the snapshot was saved. Using snapshots was much faster than rebooting the phone and was a necessity as the crashes were frequent during our traffic analysis.

We created a snapshot for each dynamic analysis approach, capturing the application state at the point when our dialog choices had been stored for that specific approach.

We analyzed apps' traffic in two stages. In the first stage we analyzed apps' traffic for 40 seconds while the apps remained idle, with our dialog choices already stored. For the second stage we reset all dialog choices to then analyze traffic during interactions with apps' consent dialogs and also while idling for 20 seconds after. Analyzing traffic during interactions with consent dialogs also allowed us to assess if apps were transmitting personal data before any dialog choices were stored.

4.4 Determining the Prevalence of Cookie Paywalls

From interactions with consent dialogs during the dynamic analysis of apps, we found that apps occasionally present `cookie paywalls`, before or after the user interacts with the app's consent dialog. Two examples of `cookie paywalls` from our dataset can be seen in Figure 4.7 and Figure 4.8.

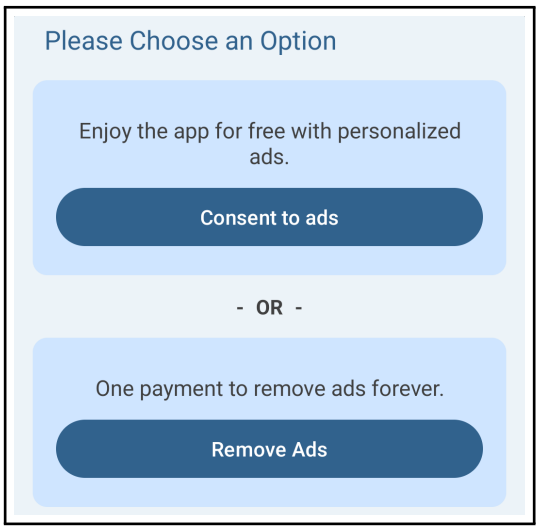

Figure 4.7: Cookie paywall in app Chess Online & Offline (com.splendapps.checkmate).

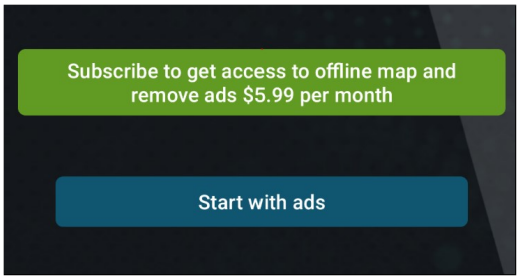

Figure 4.8: Cookie paywall in app Compass 22G (GPS Camera) (com.milu.compass22g).

To determine the prevalence of `cookie paywalls` in our dataset we decided to extend our interaction with consent dialogs when using the **Ø**- or **LI**-approach, by also using `OCR` and sentence similarity. In our interactions with Android applications, `cookie paywalls` show up before consent dialogs or right after having interacted with consent dialogs – if not providing full consent –, if we were unable to detect a consent dialog we therefore search for a `cookie paywall` in the app. We also search for `cookie paywalls` right after having interacted with a consent dialog. To find `cookie paywalls` we automatically capture a screenshot of the emulated device's screen which is then used as input into the Python `OCR` model, EasyOCR. The

OCR model then outputs all sentences found in our screenshot and we can use this output as candidate inputs – which are embedded –, using the multilingual sentence transformers model paraphrase-multilingual-MiniLM-L12-v2. The similarity between the embedded candidate sentences and embedded target sentences – which have been found in previous cookie paywalls –, are then calculated using cosine similarity. If the best contender surpasses a threshold we save the screenshot used as input, since the app it belongs to potentially presents a cookie paywall. If the threshold was not surpassed the screenshot is removed. The content of the stored screenshots are lastly manually investigated, to remove potential false positives, such as regular paywalls and subscription based services.

5

Results

This chapter describes what results were achieved based on the initial research questions. We present in Section 5.1 the amount of applications that were scraped and what percentage of them were using the TCF, answering **RQ1**. We further describe potential GDPR violations found in TCF-based Android applications during our dynamic and traffic analysis in Section 5.2 and Section 5.3, covering **RQ2** and **RQ3**. Lastly, we present our results on the prevalence of `cookie paywalls` in TCF-based Android applications in Section 5.4, answering **RQ4**. The data used to produce the results can be found in our public repo [57].

5.1 Quantifying the TCF and the Achieved Dataset

The Play Store was scraped daily between 2025-01-29 and 2025-03-10, resulting in a dataset of 5 067 apps. When downloading applications, 585 apps presented issues. Some apps were premium and required a payment to install, a few apps were pre-registrations and did not provide an installable option at the time, and some applications were removed from the Play Store between the creation of our dataset and our attempted download. Most of the apps which could not be downloaded, replaced the installation button with the message: "Your device isn't compatible with this version.", as seen in Figure A.3 in Appendix A. This is most likely due to our device being emulated. The apps which could be downloaded, were downloaded between 2025-02-04 and 2025-03-12. Full details about the dataset results can be seen in Table 5.1.

Of the downloaded applications, 842 apps were identified as implementing the TCF, representing 18.8% of the downloaded apps.

App Status	App Amount (N = 5 067)
Not Downloadable	585 (11.5%)
Not Using the TCF	3640 (71.8%)
Using the TCF	842 (16.6%)

Table 5.1: Dataset results from scraping and downloading apps.

5.2 Dynamic Analysis

Of the 842 applications using the TCF in our dataset, 266 apps could not be dynamically analyzed due to various issues. The most common issues were apps not presenting a consent dialog, apps crashing on launch, and apps using unique consent dialogs. Indeed, some apps would also use words with unique semantics in their consent dialogs, which would result in us being unable to interact with them correctly, these apps are considered as not being analyzable. Of the 576 apps we could interact with, no analyzed apps used any disallowed legitimate interest purposes, regardless of the interaction method, however, 15 apps stored our dialog choices incorrectly. These apps only stored our choices if provided with full consent, the user would otherwise be prompted with a consent dialog to provide consent again on the next launch of the app. Compiled results from the dynamic analysis can be seen in Table 5.2.

15 applications stored our dialog choices incorrectly, suggesting that there are apps using the TCF without respecting users' dialog choices.

Analysis Results	**App Amount (N = 842)**
Not Analyzable	266 (31.6%)
Apps Storing Dialog Choices Correctly	561 (66.6%)
Apps Storing Dialog Choices Incorrectly	15 (1.8%)

Table 5.2: Dynamic analysis results from automatic interactions with consent dialogs.

5.3 Traffic Analysis

After finishing the dynamic analysis, we performed two stages of traffic analysis on the apps which we had interacted with. The following subsections describe the results produced from the two stages of analysis.

5.3.1 Stage One

Of the 561 apps that we could interact with in the dynamic analysis, 550 apps had their traffic analyzed during our first stage between 2025-03-23 and 2025-05-17. 11 apps could not have their traffic analyzed, as they would instantly crash or deny SSL pinning being disabled. The following paragraphs describe the results from the first stage of traffic analysis.

Using the **LI**-approach, 353 applications were found to transmit personal data. Of the 353 apps, 351 apps transmitted `AAID` and 12 apps transmitted public IP.

Using the **Ø**-approach, we found 366 applications that transmit personal data, with 364 apps transmitting `AAID` and 15 apps transmitting the public IP connected to the device.

Using the **C+LI**-approach, we found 367 apps that transmit `AAID`, and 21 apps transmitting public IP, resulting in 372 apps transmitting personal data, an increase

by 19 apps compared to the results when apps were provided with no consent and no legitimate interest. The compiled results can be seen in Table 5.3.

Consent Dialog Interaction	**Apps Transmitting Personal Data (N = 550)**	**Apps Transmitting AAID**	**Apps Transmitting Public IP**
LI-Approach	353 (64.2%)	351	12
Ø-Approach	366 (66.5%)	364	15
C+**LI**-Approach	372 (67.6%)	367	21

Table 5.3: Traffic analysis results from the first stage.

5.3.2 Stage Two

We got different results for the second stage of traffic analysis, where we analyzed the traffic before and while interacting with apps' consent dialogs. This analysis was conducted between 2025-05-15 and 2025-05-21, where 513 apps had their traffic analyzed. 37 additional apps could not have their traffic analyzed in this stage as they had been removed from the Play Store, no longer presented a consent dialog or instantly crashed on launch. During the second stage we found 284 applications that transmitted personal data before or during our dialog interaction, which means they have not yet received our consent to gather any personal data. All of those apps transmitted our `AAID` and nine of them transmitted our public IP. The results from the second stage of traffic analysis can be seen in Table 5.4.

Using the **LI**-approach, we found that 192 apps transmit our `AAID` and 7 apps transmit our public IP.

Using the **Ø**-approach, we found that 145 apps transmit our `AAID` while 5 apps transmit our public IP.

Using the **C**+**LI**-approach, we found that 218 apps transmit our `AAID` and 7 apps transmit our public IP.

Consent Dialog Interaction	**Apps Transmitting Personal Data (N = 513)**	**Apps Transmitting AAID**	**Apps Transmitting Public IP**
LI-Approach	193 (37.6%)	192	7
Ø-Approach	147 (28.7%)	145	5
C+**LI**-Approach	218 (42.5%)	217	7
During Dialog Interaction	284 (55.4%)	284	9

Table 5.4: Traffic analysis results from the second stage.

In the first stage of traffic analysis, **66.5%** of applications transmit personal data when using our Ø-approach. During the second stage of analysis, **55.4%** of apps also transmit personal data before or while we were interacting with the apps' dialogs. Based on these observations, **a majority of TCF-based Android apps do not respect users' dialog choices during runtime.**

5.4 Prevalence of Cookie Paywalls

In our dataset of 842 applications containing the TCF, four cookie paywalls were found. These apps belong to three different developers who have made other applications with similar `cookie paywalls`.

6

Discussion

This chapter discusses the results achieved from our research questions and the limitations and ethical issues encountered.

6.1 Results Concerning TCF-based Android Applications

In this section, we discuss results from **RQ1** in Subsection 6.1.1, regarding the increase in usage of the TCF in Android applications compared to previous research. Subsection 6.1.2 discusses results achieved based on **RQ2**, covering TCF-based apps' storage of users' dialog choices. How TCF-based apps transmit personal data based on results from **RQ3** is discussed in Subsection 6.1.3. Lastly, results from **RQ4**, regarding the prevalence of `cookie paywalls` in our dataset, is discussed in Subsection 6.1.4.

6.1.1 Presence of the TCF

Our results showed that 18.8% of downloaded applications were using the TCF. The most recent research mentioning TCF-based Android applications from Koch et al. [12] in 2023 found that 6.6% of researched applications were using the TCF. Our findings of 18.8% of applications using the TCF, show **an increase of TCF-usage in Android applications by a factor of 2.8 in three years**, compared to the previous research. Such an increase in a short amount of time makes it even more important that TCF-based applications fulfill the requirements of legislations, such as the GDPR. The increase in usage of the TCF in Android apps can be seen in Table 6.1. It is however worth mentioning that previous papers have not used a similar methodology to confirm apps' usage of the TCF, potentially attributing to the increase.

We encountered twelve different CMPs in our dataset. The most common CMP in the dataset was Google LLC with a large majority (82.9%). Statistics on all CMPs found in the dataset can be seen in Table 6.2.

Author(s)	Year(s) Studied	Apps Using the TCF	Dataset Size
Altpeter [23]	2022	64 (1.9%)	3 421
Koch et al. [12]	2022-2023	199 (6.6%)	3 006
Us	2025	842 (18.8%)	4 482

Table 6.1: Presence of the TCF in Android apps, compared to previous research.

CMP ID	CMP Name	Apps Using CMP (N = 842)
300	Google LLC	698 (82.9%)
5	Usercentrics GmbH	35 (4.2%)
28	OneTrust LLC	27 (3.2%)
7	Didomi	26 (3.1%)
No ID	-	17 (2.0%)
350	Easybrain Ltd	15 (1.8%)
348	Outfit7 Limited	11 (1.3%)
6	Sourcepoint Technologies, Inc	8 (1.0%)
90	Commanders Act	1 (0.1%)
371	Removed	1 (0.1%)
280	Removed	1 (0.1%)
258	Removed	1 (0.1%)
14	Removed	1 (0.1%)

Table 6.2: CMP statistics, presenting Google's position of dominance, names from IAB's list of mobile CMPs for TCF v2.2 [58].

When scraping the Play Store, we stored which categories apps were scraped from. Data regarding app categories which were most common to use the TCF in our dataset can be seen in Table 6.1.1, where Personalization has the largest TCF-usage at 60.4% of applications. All data on TCF-usage and categories from our dataset can be found in Dataset App Categories in Appendix C. Additionally, we also quantify the amount of TCF-based applications developed by each country found in our dataset, in Developer Country Statistics in Appendix D.

Category	TCF-usage
Personalization	61/101 (60.4%)
Music and Audio	35/85 (41.2%)
Libraries and Demo	31/76 (40.8%)
Art and Design	32/79 (40.5%)
Productivity	25/67 (37.3%)
Weather	24/65 (36.9%)
Video Players	23/67 (34.3%)
Tools	30/90 (33.3%)
Maps and Navigation	28/86 (32.6%)
Photography	24/77 (31.2%)
News and Magazines	18/61 (29.5%)

Table 6.3: App categories in the Play Store that most commonly use the TCF in our dataset (more than 25% of apps from the category using the TCF).

6.1.2 Dynamic Analysis

15 apps were found to only store users' dialog choices if provided with full consent. If providing anything other than full consent, the consent dialog will keep showing up on consecutive launches of the app. Such a deceptive design is likely implemented to coerce users into providing full consent.

Another deceptive design encountered during the dynamic analysis was how identical CMP's could work differently in different applications. For example, applications using the same CMP could store a user's dialog choices (regarding legitimate interest) differently, despite the user clicking identical "disagree to all"-buttons. This made our automatic interactions with applications for the **Ø**-approach more time-consuming as it was often required to manually opt out of every individual purpose, since "disagree to all"-buttons regularly functioned as "disagree to all consent"-buttons. Furthermore, this shows what is necessary of a user, if they want to ascertain that the application stores dialog choices according to their expectations.

6.1.3 Traffic Analysis

63.2% of analyzable applications were found to transmit personal data – when using the **Ø**-approach –, during the first stage of traffic analysis. The IAB have previously in 2018 explained that online identifiers, such as IP addresses and `AAID` are explicit examples of personal data under the GDPR [59]. Google's ad service, AdMob, declares that **it is necessary to obtain consent from users to use personal data such as the AAID** in the EEA [60]. Android states that `AAID` is the recommended identifier when building profiles of users and tracking them. It is also stated that if the user explicitly opts out of personalization using `AAID` – in their Android device's settings –, the `AAID` will be removed and made unidentifiable [61]. This further implies that the `AAID` should only be used by developers in instances

for tracking and personalization. Furthermore, we reached out to a legal expert in the field who confirmed that usage of the `AAID` requires consent from users.

We expected a large majority of apps to transmit personal data when using the **C+LI**-approach, but since 32.2% of apps in the first stage and 57.5% of apps in the second stage were found to not transmit any personal data, we decided to further analyze some of these apps. We initially tried increasing the idling time from 40 to 90 seconds on these apps, without observing an increase in HTTP requests and transmission of personal data. We therefore randomly picked apps to extensively investigate their requests using the `mitmweb` browser GUI-tool from mitmproxy. From this investigation we found that these apps did send similar requests as apps which transmitted personal data, but without the personal data. These requests' payloads did however mention newer "SDK-versions" related to the domains the data was being sent to. We further investigated some requests related to the domain `https://fundingchoicesmessages.google.com` – now part of Google AdMob [62] –, and found that in recent "SDK-versions" (since 2024-01-24) usage of `AAID` has been removed [63]. This potentially explains why some apps provided with full consent were not identified to transmit personal data, leaving the following question unanswered: what personal data do these apps transmit and when?

To further analyze applications' traffic, we created a second stage of analysis, analyzing traffic while interacting with apps' consent dialogs. The second stage resulted in us finding that a majority of apps (55.4%) transmitted personal data, before or during interaction with apps' consent dialogs. In the second stage of analysis we found six additional applications that transmitted personal data using the **C+LI**-approach, but this still leaves 172 apps without any traces of transmitting personal data when given full consent. Further analysis of these applications would be necessary to establish the underlying cause.

Comparing the results from the analysis in stage one and two, we noticed that personal data transmitted while idling in the second stage had a lower percentage of apps transmitting data in our dynamic analysis approaches than in the first stage (see Table 5.3 and Table 5.4). By examining the applications which no longer appeared while idling and instead appeared during the consent dialog interaction in the stage two analysis, we found that a majority of those apps now instead transmitted personal data during the consent dialog interaction. This potentially explains why the percentages in personal data transmitted while idling during stage one and two had differing amounts, amended stage two results can be seen in Table 6.4.

Consent Dialog Interaction	Stage One Results (N = 550)	Stage Two Results (N = 513)	Stage Two Amended Results (N = 513)
LI-Approach	353 (64.2%)	193 (37.6%)	339 (66.1%)
Ø-Approach	366 (66.5%)	147 (28.7%)	330 (64.3%)
C+LI-Approach	372 (67.6%)	218 (42.5%)	347 (67.6%)

Table 6.4: Stage two traffic analysis results, amended with data from apps transmitting personal data during consent dialog interaction.

Based on the results we found from stage one and two of our traffic analysis, **if TCF-based applications want to transmit users' personal data, they will in a majority of cases do so regardless of the users' dialog choices.** The one workaround for users who does not want to share their data with applications would be to remove their `AAID` in their Android device's settings, as mentioned by Android themselves [61].

The most common domains (more than 100 HTTP requests) which received our personal data during stage one and two of trafic analysis, can be seen in Table 6.5. An extension of this table with all the domains receiving personal data during our analyses can be seen in Domains in Appendix B.

Domain	Request Count	Country of Origin
adjust.com	1555	Germany
facebook.com	803	USA
unity3d.com	678	USA
rayjump.com	490	China
bidmachine.io	439	USA
google.com	351	USA
inmobi.com	231	India
singular.net	213	USA
easybrain.com	170	Cyprus
amazon-adsystem.com	145	USA
vungle.com	125	USA

Table 6.5: Most common domains receiving personal data during stage one and two of traffic analysis, country of origin found through privacy policies and website homepages of domains.

6.1.4 Prevalence of Cookie Paywalls

During the automatic search for `cookie paywalls`, the most prominent issue encountered was that applications with normal premium subscriptions shown on screen after the consent dialog, often contain similar words as `cookie paywalls`. All screenshots of found `cookie paywalls` therefore had to be manually confirmed to then remove false positives (e.g., regular paywalls or subscription based services). We performed no further analysis on apps containing `cookie paywalls`, as it is outside the scope of the project.

6.2 Ethics and Limitations

Ethical considerations. Our research of how TCF-based applications respect user privacy was limited by the amount of applications we could find that use the TCF. One factor was the Play Store `robots.txt` [64] file – which explains what interactions are allowed and disallowed on a website, for non-human users [65] –, as it limited our scraping of the Play Store for potential applications, due to us wanting to scrape ethically. One limitation according to the Play Store `robots.txt` file was that non-human users, such as web scrapers, were disallowed from using the Play Store search bar. It was however allowed to access top apps based on category, as it was not mentioned in the file.

Downloading bottleneck. Downloading applications was our biggest bottleneck, as the downloading speed of the emulated device was locked at a much lower speed than our host computer's downloading speed. We had to download applications through the Google Play Store UI of the emulated device. This was necessary since all open-source software we tested – with the purpose of downloading Android applications to a computer –, had been deprecated. This limited us to a dataset of 4 482 downloadable apps.

Only interacting with consent dialogs. When interacting with applications, we only interacted with consent dialogs. This leaves most of the applications' functionalities untouched, potentially causing us to miss out on transmission of personal data. Furthermore, we study the applications on an emulated Android device through Android Studio, which can be a limiting factor since it does not always properly represent a real Android device.

Dataset of CMPs. There exists 50 different CMPs in the TCF on mobile [58], all these CMPs look different and require different logic to interact with their consent dialogs. We had 12 different CMPs in our dataset, and we interacted with six out of the 12. Each of the six CMPs we did not interact with, either only appeared once in unique apps or had unique consent dialogs in each app. We therefore decided that it would be inefficient to address these CMPs in our analysis. Our results of analyzing the TCF are limited to the CMPs we found in our dataset.

Analyzing other Operating Systems. The methodology used in our thesis is dependent on `adb` which is unique for Android devices, making extensions of our methodology to other operating systems complicated. Another limiting factor is the

requirement of Apple hardware that can run macOS to interact with an iOS device, which we did not have access to. However, as mentioned in section 3.3, Altpeter did their study on both iOS and Android applications [23], proving that it is possible to automatically analyze applications on iOS (using Appium and mitmproxy).

7

Conclusion

In this chapter we initially conclude the results from our research questions, and then present future work based on our limitations.

7.1 Summary

This thesis investigated the usage of the TCF in Android applications. The research was conducted by scraping and downloading apps from the Google Play Store, automatically interacting with consent dialogs while simultaneously determining the presence of `cookie paywalls`, and lastly, analyzing apps' traffic in two different stages. Hereby follows the conclusions derived from our research questions.

RQ1: How prevalent are TCF-based applications in the Google Play Store? To quantify the usage of the TCF in Android applications, we scraped and downloaded 4 482 of the most popular apps in the Google Play Store, as described in Section 5.1. Of the downloaded applications, 842 apps were identified as implementing the TCF, representing 18.8% of the downloaded apps. Indicating an increase in usage of the framework compared to previous research by Altpeter [23] and Koch et al. [12].

RQ2: Do TCF-based applications respect users' dialog choices and the TCF's restrictions of legitimate interest? In Section 5.2, we automatically interacted with consent dialogs of 576 applications and confirmed their stored dialog choices. 15 applications stored our dialog choices incorrectly, suggesting that there are apps using the TCF without respecting users' dialog choices.

RQ3: Do TCF-based Android applications respect users' choices regarding consent and legitimate interest at runtime? We performed two stages of traffic analysis, in the first stage, 66.5% of applications transmitted personal data when using our Ø-approach (providing apps with no consent and no legitimate interest). During the second stage of analysis, 55.4% of apps also transmitted personal data before or while we were interacting with the apps' dialogs. Based on the results in Section 5.3, we conclude that TCF-based Android applications do not respect users' dialog choices during runtime, potentially violating the GDPR.

RQ4: How prevalent are cookie paywalls in TCF-based Android applications? In our dataset of 842 applications containing the TCF, four `cookie paywalls` were found, confirming that they do exist in Android applications.

7.2 Future Work

This thesis presents an overview of TCF usage in Android applications, researching popular apps using the most common CMPs. Future work could extend this research into larger datasets by improving solutions for downloading apps and by proposing methods for further interacting with apps post their consent dialog.

Downloading applications was a bottleneck for our study, it would be beneficial to extend our research by improve the process of downloading apps. We propose to either download apps outside the emulator – similar to methods used by deprecated open-source tools, such as `Playstoredownloader` [48] –, and then moving them onto the emulator, or finding a way to adjust the download speed of the device to bypass its initial limit.

Each downloaded app in our dataset required analysis to confirm its potential usage of the TCF. A large amount of time was spent on downloading apps that did not use the TCF (81.2% of downloaded apps). During interactions with consent dialogs, we noticed that many apps using the TCF were published by the same developers. In the Google Play Store, each application has a link to its developer, meaning that given an app, it is simple to scrape all the apps from the same developer. With this approach, extending the dataset with more apps likely to use the TCF becomes a possibility. We tried this method on our set of 842 TCF-based applications and found 4 741 apps, all published by developers who have been using the TCF in other apps. We have not been able to analyze this extended dataset, but we present this method as an opportunity to further analyze usage of the TCF in Android applications.

In this thesis we only prove that `cookie paywalls` exist in Android applications, with no further analysis. To further analyze `cookie paywalls` in Android applications, a larger dataset would be necessary. We hereby propose a method to create a larger dataset of Android applications containing `cookie paywalls`. Given a dataset of websites using `cookie paywalls`, which can be found in previous research by Thunberg et al. [66], it is possible to extract websites' Android applications using their `assetlinks.json` file [67] with the format:
`https://domain.name/.well-known/assetlinks.json` if they have an associated application. This file exists so that mobile users who visit a website – with a related application –, can be redirected to install or open the website's content in its application instead. The `assetlinks.json` files of websites can easily be scraped using Selenium, this method makes it possible to find apps likely to use `cookie paywalls` and create a larger dataset of apps, to further analyze `cookie paywalls` in Android applications. The dataset could be further extended by using our previously proposed method to download all apps created by developers who previously have published apps using `cookie paywalls`.

Our traffic analysis was only conducted on apps while interacting with consent dialogs and then while idling afterward. Further interacting with apps, post consent dialog interaction, would more closely simulate how a human interacts with applications. However, further interactions with apps in a general manner is challenging, since apps are unique.

Bibliography

[1] Intersoft Consulting, *Lawfulness of processing*, Accessed: 2025-02-06. [Online]. Available: `https://gdpr-info.eu/art-6-gdpr/`.

[2] Proton AG, *Art. 4 GDPR Definitions*, Accessed: 2024-03-31. [Online]. Available: `https://gdpr.eu/article-4-definitions/`.

[3] R.Koch, *Cookies, the GDPR, and the ePrivacy Directive*, Accessed: 2024-11-23, Aug. 2024. [Online]. Available: `https://gdpr.eu/cookies/`.

[4] IAB Europe, *Transparency & Consent Framework*, Accessed: 2024-11-23. [Online]. Available: `https://iabeurope.eu/transparency-consent-framework/`.

[5] C. Matte, N. Bielova, and C. Santos, "Do cookie banners respect my choice?: Measuring legal compliance of banners from iab europe's transparency and consent framework," in *2020 IEEE Symposium on Security and Privacy (SP)*, IEEE, 2020, pp. 791–809.

[6] IAB Europe, *Understanding the Transparency & Consent Framework v2.2*, Accessed: 2024-11-28. [Online]. Available: `https://iabeurope.eu/understanding-the-upcoming-transparency-consent-framework-v2-2/`.

[7] L. Kyi, S. Ammanaghatta Shivakumar, C. T. Santos, F. Roesner, F. Zufall, and A. J. Biega, "Investigating deceptive design in gdpr's legitimate interest," in *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, 2023, pp. 1–16.

[8] V. Morel, C. Santos, V. Fredholm, and A. Thunberg, "Legitimate interest is the new consent-large-scale measurement and legal compliance of iab europe tcf paywalls," in *Proceedings of the 22nd Workshop on Privacy in the Electronic Society*, 2023, pp. 153–158.

[9] similarweb, *Mobile vs. Desktop vs. Tablet Traffic Market Share*, Accessed: 2025-02-11. [Online]. Available: `https://www.similarweb.com/platforms/`.

[10] Statcounter, *Mobile Operating System Market Share Worldwide*, Accessed: 2025-02-11. [Online]. Available: `https://gs.statcounter.com/os-market-share#monthly-202001-202501/`.

[11] T. T. Nguyen, M. Backes, N. Marnau, and B. Stock, "Share first, ask later (or never?) studying violations of {gdpr's} explicit consent in android apps," in *30th USENIX Security Symposium (USENIX Security 21)*, 2021, pp. 3667–3684.

[12] S. Koch, B. Altpeter, and M. Johns, "The {ok} is not enough: A large scale study of consent dialogs in smartphone applications," in *32nd USENIX Security Symposium (USENIX Security 23)*, 2023, pp. 5467–5484.

[13] Proton AG, *General Data Protection Regulation (GDPR)*, Accessed: 2024-01-31. [Online]. Available: https://gdpr.eu/tag/gdpr/.

[14] Directorate-General for Communication, *Legal framework of EU data protection*, Accessed: 2025-06-09. [Online]. Available: https://commission.europa.eu/law/law-topic/data-protection/legal-framework-eu-data-protection_en.

[15] Proton AG, *Faq*. [Online]. Available: https://gdpr.eu/faq/.

[16] Cloudflare, *What is the ePrivacy Directive?* Accessed: 2025-01-31. [Online]. Available: https://www.cloudflare.com/learning/privacy/what-is-eprivacy-directive.

[17] IAB Europe, *IAB Europe Transparency & Consent Framework Policies*, Accessed: 2025-03-17. [Online]. Available: https://iabeurope.eu/iab-europe-transparency-consent-framework-policies/.

[18] IAB Europe, *TCF for Vendors*, Accessed: 2025-01-31. [Online]. Available: https://iabeurope.eu/tcf-for-vendors/.

[19] IAB Europe, *Consent Management Platform API*, Accessed: 2024-11-29. [Online]. Available: https://github.com/InteractiveAdvertisingBureau/GDPR-Transparency-and-Consent-Framework/blob/master/TCFv2/IAB%20Tech%20Lab%20-%20CMP%20API%20v2.md.

[20] IAB Europe, *Transparency and Consent String with Global Vendor & CMP List Formats*, Accessed: 2025-02-12. [Online]. Available: https://github.com/InteractiveAdvertisingBureau/GDPR-Transparency-and-Consent-Framework/blob/master/TCFv2/IAB%20Tech%20Lab%20-%20Consent%20string%20and%20vendor%20list%20formats%20v2.md#how-should-a-transparency--consent-string-be-store.

[21] MailChimps, *Data Tracking*, Accessed: 2025-01-31. [Online]. Available: https://mailchimp.com/resources/data-tracking/.

[22] T. Libert, "Exposing the hidden web: An analysis of third-party http requests on 1 million websites," *arXiv preprint arXiv:1511.00619*, 2015.

[23] B. Altpeter, "Informed consent? A study of "consent dialogs" on android and iOS.," Accessed: 2024-11-28, M.S. thesis, Technische Universität Braunschweig, 2022.

[24] Cookie Banner Taskforce, *Report of the work undertaken by the Cookie Banner Taskforce*, Accessed: 2025-02-13, Jan. 2023. [Online]. Available: https://www.edpb.europa.eu/system/files/2023-01/edpb_20230118_report_cookie_banner_taskforce_en.pdf.

[25] Android, *Android Studio*, Accessed: 2024-11-29. [Online]. Available: https://developer.android.com/studio.

[26] Android, *Create and manage virtual devices*, Accessed: 2025-02-03. [Online]. Available: https://developer.android.com/studio/run/managing-avds.

[27] Android, *Run apps on the Android Emulator*, Accessed: 2025-02-03. [Online]. Available: https://developer.android.com/studio/run/emulator.

[28] Android, *Introduction to activities*, Accessed: 2025-03-13. [Online]. Available: https://developer.android.com/guide/components/activities/intro-activities.

[29] Android, *Android Debug Bridge*, Accessed: 2025-03-13. [Online]. Available: https://developer.android.com/tools/adb.

[30] magisk, *Magisk Manager*, Accessed: 2025-03-13. [Online]. Available: https://magiskmanager.com/#What_is_Magisk.

[31] Android, *Save simple data with SharedPreferences*, Accessed: 2025-03-13. [Online]. Available: https://developer.android.com/training/data-storage/shared-preferences.

[32] Selenium, *A deeper look at Selenium*, Accessed: 2025-02-27. [Online]. Available: https://www.selenium.dev/documentation/overview/details/.

[33] Appium, *How Does Appium Work*, Accessed: 2025-02-27. [Online]. Available: https://appium.io/docs/en/latest/intro/appium/.

[34] Cloudflare, *What is HTTPS?* Accessed: 2025-03-06. [Online]. Available: https://www.cloudflare.com/learning/ssl/what-is-https/.

[35] Indusface, *What is SSL Pinning? – A Quick Walk Through*, Accessed: 2025-03-06. [Online]. Available: https://www.indusface.com/learning/what-is-ssl-pinning-a-quick-walk-through/.

[36] Fortinet, *What Is A Proxy Server? How does It Work?* [Online]. Available: https://www.fortinet.com/resources/cyberglossary/proxy-server.

[37] mitmproxy, *Mitmproxy*, Accessed: 2025-03-13. [Online]. Available: https://mitmproxy.org/.

[38] mitmproxy, *Addons*, Accessed: 2025-03-31. [Online]. Available: https://docs.mitmproxy.org/stable/addons-overview/.

[39] C. S. Smith, *What Is OCR (Optical Character Recognition) Technology?* Accessed: 2025-04-23. [Online]. Available: https://www.forbes.com/sites/technology/article/what-is-ocr-technology/.

[40] JaidedAI, *EasyOCR*, Accessed: 2025-04-23. [Online]. Available: https://github.com/JaidedAI/EasyOCR.

[41] Hugging Face, *Sentence Similarity*, Accessed: 2025-04-23. [Online]. Available: https://huggingface.co/tasks/sentence-similarity.

[42] P. Miesle, *What is Cosine Similarity: A Comprehensive Guide*, Accessed: 2025-04-23. [Online]. Available: https://www.datastax.com/guides/what-is-cosine-similarity.

[43] D. Bollinger, K. Kubicek, C. Cotrini, and D. Basin, "Automating cookie consent and {gdpr} violation detection," in *31st USENIX Security Symposium (USENIX Security 22)*, 2022, pp. 2893–2910.

[44] M. Smith, A. Torres-Agüero, R. Grossman, P. Sen, Y. Chen, and C. Borcea, "A study of gdpr compliance under the transparency and consent framework," in *Proceedings of the ACM Web Conference 2024*, 2024, pp. 1227–1236.

[45] DeepSee, *Deepsee.io*, Accessed: 2025-03-13. [Online]. Available: https://deepsee.io/.

[46] J. Turow, M. Hennessy, and N. Draper, "The tradeoff fallacy: How marketers are misrepresenting american consumers and opening them up to exploitation," *Available at SSRN 2820060*, 2015.

[47] sensepost, *objection - Runtime Mobile Exploration*, Accessed: 2025-03-13. [Online]. Available: https://github.com/sensepost/objection.

[48] C. Georgiu, *PlaystoreDownloader*, Accessed: 2025-03-13. [Online]. Available: https://github.com/ClaudiuGeorgiu/PlaystoreDownloader.

[49] Selenium, *Selenium*, Accessed: 2025-05-14. [Online]. Available: https://www.selenium.dev/.

[50] Developers Google, *Fused Location Provider API*, Accessed: 2025-05-22. [Online]. Available: https://developers.google.com/location-context/fused-location-provider.

[51] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Nov. 2019. [Online]. Available: http://arxiv.org/abs/1908.10084.

[52] N. Reimers and I. Gurevych, *sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2*, Accessed: 2025-04-23. [Online]. Available: https://huggingface.co/sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2.

[53] mitmproxy, *About Certificates*, Accessed: 2025-03-14. [Online]. Available: https://docs.mitmproxy.org/stable/concepts-certificates/.

[54] Magisk, *Magisk Module*, Accessed: 2025-03-14. [Online]. Available: https://magiskmodule.gitlab.io/.

[55] Frida, *Dynamic instrumentation toolkit for developers, reverse-engineers, and security researchers.* Accessed: 2025-03-14. [Online]. Available: https://frida.re/.

[56] Android, *Snapshots*, Accessed: 2025-05-13. [Online]. Available: https://developer.android.com/studio/run/emulator-snapshots.

[57] carlssonp, ahlinder, *Android-TCF-Analysis-Data*, Accessed: 2025-05-27. [Online]. Available: https://github.com/carlssonp/Android-TCF-Analysis-Data.

[58] IAB Europe, *CMP List*, Accessed: 2025-04-30. [Online]. Available: https://iabeurope.eu/cmp-list/.

[59] IAB Europe, *THE DEFINITION OF PERSONAL DATA*, Accessed: 2025-05-05. [Online]. Available: https://iabeurope.eu/wp-content/uploads/20180718_IABEU-GIG-Working-Paper02_Personal-Data_v1.1.pdf.

[60] Google AdMob, *GDPR IAB support*, Accessed: 2025-05-05. [Online]. Available: https://developers.google.com/admob/android/privacy/gdpr.

[61] Android, *Best practices for unique identifiers*, Accessed: 2025-05-05. [Online]. Available: https://developer.android.com/identity/user-data-ids#ads.

[62] Google, *Funding Choices has moved*, Accessed: 2025-05-19. [Online]. Available: https://support.google.com/fundingchoices/answer/9010669?hl=en.

[63] Google, *Release notes*, Accessed: 2025-05-19. [Online]. Available: https://developers.google.com/admob/android/privacy/release-notes.

[64] Google Play, *Google Play robot.txt*, Accessed: 2025-03-13. [Online]. Available: https://play.google.com/robots.txt.

[65] Cloudflare, *What is robots.txt? | How a robots.txt file works*, Accessed: 2025-05-12. [Online]. Available: https://www.cloudflare.com/learning/bots/what-is-robots-txt/.

[66] A. Thunberg and O. Wallgren, "Large-scale detection of cookie paywalls," 2023.

[67] Android Developers, *Verify Android App Links*, Accessed: 2025-04-30. [Online]. Available: `https://developer.android.com/training/app-links/verify-android-applinks`.

A

Additional Figures

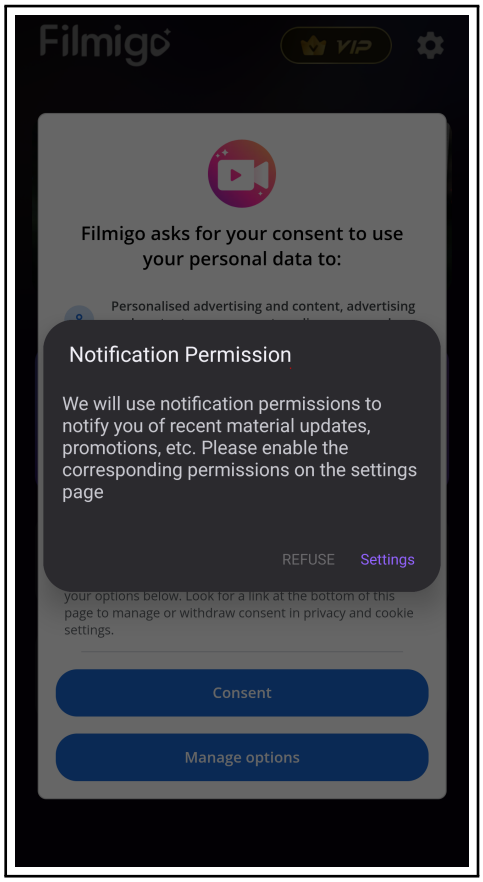

Figure A.1: Permissions popup.

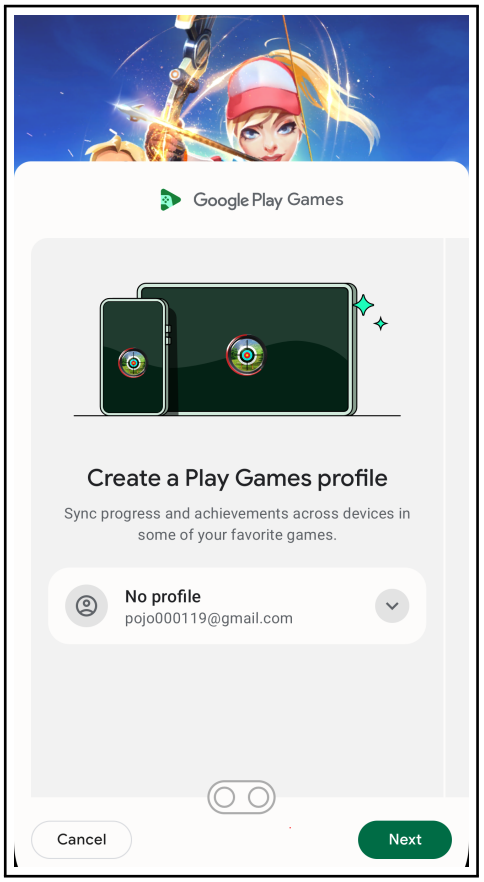

Figure A.2: Google Play Games.

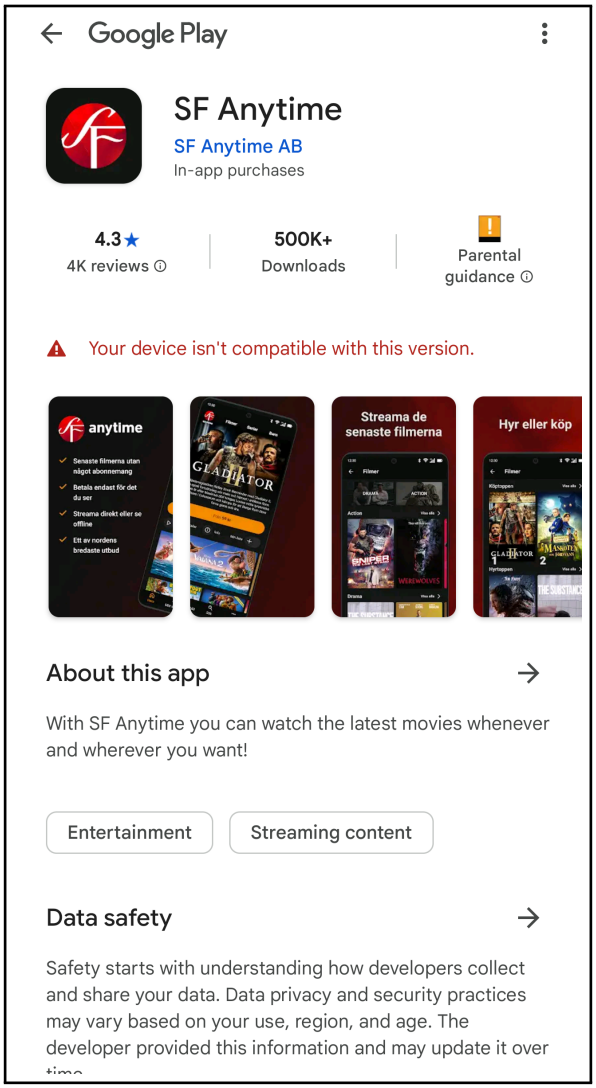

Figure A.3: SF Anytime, non-downloadable application.

B

Domains

Domain	Request Count
app.adjust.com	1555
graph.facebook.com	763
configure.rayjump.com	445
configv2.unityads.unity3d.com	356
fundingchoicesmessages.google.com	351
api.bidmachine.io	322
sdk-api-v1.singular.net	213
ads.inmobi.com	127
auction-load.unityads.unity3d.com	110
api-eu.bidmachine.io	109
cfg.easybrain.com	89
o-sdk.mediation.unity3d.com	83
cross-promo-provider.easybrain.com	81
httpkafka.unityads.unity3d.com	79
api.adapty.io	76
aax-eu.amazon-adsystem.com	76
api.revenuads.com	67
aax.amazon-adsystem.com	66
unif-id.ssp.inmobi.com	65
api.kickoffo.site	57
api.inmense.site	57
api.dollphoin.site	55
adx.ads.vungle.com	46
hb-af-us-central1.outfit7.com	45
api.gameanalytics.com	40
config.inmobi.com	39
app.adjust.net.in	38
www.googleadservices.com	33
rtb.ads.vungle.com	32
ad2.fivecdm.com	32
fid.agkn.com	30
app.adjust.world	30
wv.inner-active.mobi	24
configure.mtgglobals.com	24

www.facebook.com	22
news-af.feednews.com	22
events.ads.vungle.com	22
api.revenuecat.com	22
api.bytebrew.io	22
bidder.dsp.outfit7.com	21
new.ads.vungle.com	20
gateway-2.papermobi.com:8080	20
api.pubnative.net	20
api.monedata.io	20
a.appbaqend.com	20
api.qonversion.io	19
web.facebook.com	18
api2.branch.io	18
track.saygames.io	17
outcome-ssp.supersonicads.com	16
live.saygames.io	16
sdk-api.maticooads.com	15
srv.wego.com	14
ads-bi.zenkube.com	14
infoevent.startappservice.com	13
x.everestop.io	12
sdk-bi.zenkube.com	12
game.bphfive.com	12
evtruck.magnus.ms	12
business-api.tiktok.com	12
analytics.rayjump.com	12
policy.rayjump.com	11
pok.mintegral.net	11
lazy.rayjump.com	11
in1.clevertap-prod.com	11
gw-rv.iads.unity3d.com	11
gateway.unityads.unity3d.com	11
check.rayjump.com	11
adsmetadata.startappservice.com	11
word-connect.zenkube.com	10
sdk.ad.smaato.net	10
rr3—sn-ovgq0oxu-j2ie.googlevideo.com	10
mobile.yandexadexchange.net	10
app.saygames.io	10
analytics.us.tiktok.com	10
slots-api.langlang-game.com	9
publisher-config.unityads.unity3d.com	9
gw-is.iads.unity3d.com	9
d67f2nk1leuwd.cloudfront.net	9
rr2—sn-ovgq0oxu-j2ie.googlevideo.com	8

gum.criteo.com	8
arg.atomex.net	8
api-eu-a.bidmachine.io	8
rpc.tapjoy.com	7
ingest.databuckets.com	7
f.everestop.io	7
event.tradplusad.com	7
control.kochava.com	7
api.tradplusad.com	7
api-a.op-mobile.opera.com	7
startup.mobile.yandex.net	6
smartx.collector.rbwtech.net	6
self-overseas-uskn.cn-hongkong.log.aliyuncs.com	6
rr1—sn-ovgq0oxu-j2ie.googlevideo.com	6
remain.appcpi.net	6
placements.tapjoy.com	6
overseas-hk.cn-hongkong.log.aliyuncs.com	6
news.opera-api.com	6
mobile.apps.estoty.games	6
gateway-3.hierugo.com	6
events-prod.bhaskarapi.com	6
ws.tapjoyads.com	5
trackdownload.startappservice.com	5
regs.tappx.com	5
operanews-sub.osp.opera.software	5
mtag.yieldoptimizer.com	5
mrgs-api.my.games	5
mlokagames.com	5
logs.ads.vungle.com	5
impression.appsflyer.com	5
game-log.boomplaymusic.com	5
cooking-prod.tuanguwen.com	5
connect.tapjoy.com	5
api2.amplitude.com	5
adc3-launch.adcolony.com	5
a.youleapps.com	5
wd.adcolony.com	4
ttplugins.ttpsdk.info	4
top.calulatorsecret.com	4
prod.bhaskarapi.com	4
os.rolicloud.com	4
o.isx.unity3d.com	4
node1-rtb.gravite.net	4
mjp.zimad.com	4
mcp.opera-api.com	4
dlsdk.appsflyersdk.com	4

da.chartboost.com	4
crocword.labsystech.ru	4
ase.clmbtech.com	4
api.getblueshift.com	4
api.amplitude.com	4
vekuql-dlsdk.appsflyersdk.com	3
sdk-exchange.startappservice.com	3
s-odx.oleads.com	3
report.roiquery.com	3
report.appmetrica.yandex.net	3
newsfeed.miniclippt.com	3
m.youtube.com	3
hustlemerge.cleverappssg.com	3
gateway.prod.anzu-us.com	3
gateway-b.offerwall.unity3d.com	3
bidder.tradplusad.com	3
b5207ba9-a5cd-46b5-a353-557cd6125d9f.goliath.atlas.bi .miniclippt.com	3
auction.unityads.unity3d.com	3
useastserv.bigabidserv.com	2
trk.atomex.net	2
stat.haloapps.com	2
ssp-eu.thecas.xyz	2
s.amazon-adsystem.com	2
nl-gcp-ad-track-sdk-europe-west4-c.mtgglobals.com	2
manifest.googlevideo.com	2
events.prod.anzu-us.com	2
c.webengage.com	2
b-eu.adx.opera.com	2
api.olaexbiz.com	2
adslog.apexinnotech.com	2
a-sta.haloapps.com	2
track.tenjin.com	1
ssp.api.tappx.com	1
r4—sn-5goeen7y.c.2mdn.net	1
r.youleapps.com	1
metricreceiver.cellrebel.com	1
ingest.m2catalyst.com	1
hb-failover-stpceyl2ua-uw.a.run.app	1
dl-adx.op-mobile.opera.com	1
d26xw8rp6mlgfg.cloudfront.net	1
co.kvaedit.site	1
bidder.criteo.com	1
api3.4dvertible.com	1
api.sekanderali32.xyz	1
aax-us.amazon-adsystem.com	1

a1w2y5-dlsdk.appsflyersdk.com	1
5yxfxv-register.appsflyersdk.com	1

Table B.1: Domains receiving personal data during stage one and two of traffic analysis.

C

Dataset App Categories

Category	**TCF-usage**
Personalization	61/101 (60.4%)
Music and Audio	35/85 (41.2%)
Libraries and Demo	31/76 (40.8%)
Art and Design	32/79 (40.5%)
Productivity	25/67 (37.3%)
Weather	24/65 (36.9%)
Video Players	23/67 (34.3%)
Tools	30/90 (33.3%)
Maps and Navigation	28/86 (32.6%)
Photography	24/77 (31.2%)
News and Magazines	18/61 (29.5%)
Entertainment	17/75 (22.7%)
Game Word	21/93 (22.6%)
Parenting	13/62 (21.0%)
Sports	21/100 (21.0%)
Game Board	18/92 (19.6%)
Communication	11/57 (19.3%)
Health and Fitness	17/91 (18.7%)
Family	16/86 (18.6%)
Dating	11/60 (18.3%)
Game Arcade	17/93 (18.3%)
Family Education	2/11 (18.2%)
Beauty	9/52 (17.3%)
Game Sports	18/107 (16.8%)
Game Trivia	11/66 (16.7%)
Events	14/86 (16.3%)
Game Puzzle	18/113 (15.9%)
Game Music	13/85 (15.3%)
Game Casino	10/66 (15.2%)
Game Simulation	16/115 (13.9%)
Social	11/80 (13.8%)
Game Casual	18/130 (13.8%)
Game Adventure	14/108 (13.0%)
Game Role Playing	15/119 (12.6%)

House and Home	8/64 (12.5%)
Game Racing	13/105 (12.4%)
Books and Reference	9/77 (11.7%)
Game Card	8/75 (10.7%)
Lifestyle	8/80 (10.0%)
Family Create	4/40 (10.0%)
Education	8/82 (9.8%)
Game Educational	15/156 (9.6%)
Medical	3/32 (9.4%)
Game Action	15/162 (9.3%)
Game Strategy	9/98 (9.2%)
Game	43/484 (8.9%)
Business	7/82 (8.5%)
Comics	4/53 (7.5%)
Family Musicvideo	3/44 (6.8%)
Travel and Local	6/94 (6.4%)
Auto and Vehicles	3/51 (5.9%)
Application	8/144 (5.6%)
Shopping	2/62 (3.2%)
Food and Drink	2/74 (2.7%)
Family Action	1/54 (1.9%)
Finance	1/71 (1.4%)
Family Pretend	0/16 (0.0%)
Family Braingames	0/66 (0.0%)

Table C.1: App categories from our dataset and their usage of the TCF.

D

Developer Country Statistics

Country	Apps using the TCF (N = 842)
Vietnam	111 (13.2%)
Hong Kong	72 (8.6%)
United States	70 (8.3%)
China	55 (6.5%)
India	54 (6.4%)
Cyprus	53 (6.3%)
Israel	41 (4.9%)
United Kingdom	30 (3.6%)
Pakistan	30 (3.6%)
Singapore	25 (3.0%)
United Arab Emirates	23 (2.7%)
Sweden	22 (2.6%)
Spain	21 (2.5%)
Türkiye	17 (2.0%)
Morocco	15 (1.8%)
Germany	14 (1.7%)
South Korea	13 (1.5%)
Poland	13 (1.5%)
Japan	11 (1.3%)
France	11 (1.3%)
Canada	6 (0.7%)
Australia	6 (0.7%)
Andorra	6 (0.7%)
Switzerland	5 (0.6%)
Romania	5 (0.6%)
Argentina	5 (0.6%)
Russia	4 (0.5%)
Portugal	4 (0.5%)
Netherlands	4 (0.5%)
Moldova	4 (0.5%)
Lithuania	4 (0.5%)
Brazil	4 (0.5%)
Ukraine	3 (0.4%)
Serbia	3 (0.4%)

Ireland	3 (0.4%)
Indonesia	3 (0.4%)
Hungary	3 (0.4%)
Czechia	3 (0.4%)
Cayman Islands	3 (0.4%)
Norway	2 (0.2%)
Mexico	2 (0.2%)
Italy	2 (0.2%)
British Virgin Islands	2 (0.2%)
Yemen	1 (0.1%)
Uruguay	1 (0.1%)
Taiwan	1 (0.1%)
Slovakia	1 (0.1%)
Qatar	1 (0.1%)
New Zealand	1 (0.1%)
Marshall Islands	1 (0.1%)
Malta	1 (0.1%)
Malaysia	1 (0.1%)
Latvia	1 (0.1%)
Estonia	1 (0.1%)
Chile	1 (0.1%)
Bulgaria	1 (0.1%)
Belgium	1 (0.1%)
Bangladesh	1 (0.1%)
Azerbaijan	1 (0.1%)
Armenia	1 (0.1%)

Table D.1: Distribution of countries where TCF-based apps in our dataset were developed.