ATTention Please! An Investigation of the App Tracking Transparency Permission

Reham Mohamed and Arjun Arunasalam, *Purdue University;*
Habiba Farrukh, *University of California, Irvine;* Jason Tong,
Antonio Bianchi, and Z. Berkay Celik, *Purdue University*

https://www.usenix.org/conference/usenixsecurity24/presentation/mohamed

This paper is included in the Proceedings of the
33rd USENIX Security Symposium.

August 14–16, 2024 • Philadelphia, PA, USA

978-1-939133-44-1

ATTention Please!
An Investigation of the App Tracking Transparency Permission

Reham Mohamed[†], Arjun Arunasalam[†], Habiba Farrukh[‡], Jason Tong[†],
Antonio Bianchi[†], and Z. Berkay Celik[†]

[†] *Purdue University, {raburas, aarunasa, tong72, antoniob, zcelik}@purdue.edu*
[‡] *University of California, Irvine, habibaf@uci.edu*

Abstract

Apple introduced the App Tracking Transparency (ATT) framework in iOS 14.5. The goal of this framework is to mitigate user concerns about how their privacy-sensitive data is used for targeted advertising. Through this framework, the OS generates an ATT alert to request user permission for tracking. While this alert includes developer-controlled alert text, Apple mandates this text adheres to specific guidelines to prevent users from being coerced into unwillingly granting the ATT permission for tracking. However, to improve apps' monetization, developers may incorporate dark patterns in the ATT alerts to deceive users into granting the permission.

To understand the prevalence and characteristics of such dark patterns, we first study Apple's alert guidelines and identify four patterns that violate standards. We then develop ATTCLS, an ATT alert classification framework that combines contrastive learning for language modeling with a fully connected neural network for multi-label alert pattern classification. Finally, by applying ATTCLS to 4,000 iOS apps, we reveal that 59% of the alerts use four dark patterns that either mislead users, incentivize tracking, include confusing terms, or omit the purpose of the ATT permission.

We then conduct a user study with 114 participants to examine users' understanding of ATT and how different alert patterns can influence their perception. This study reveals that ATT alerts used by current apps often deceive or confuse users. For instance, users can be misled into believing that granting the ATT permission guarantees better app features or that denying it protects all of their sensitive data. We envision that our developed tools and empirical results will aid mobile platforms to refine guidelines, introduce a strict vetting process, and better design privacy-related prompts for users.

1 Introduction

In-app advertising is a primary source of revenue for mobile app developers. To optimize ad revenue, developers use targeted advertising strategies to increase user engagement (e.g., ad clicks and views). This is done by presenting users with ads that align with their preferences and behaviors.

Developers can achieve this by leveraging user identifiers, e.g., device IDs, IP addresses, which link users to detailed user profiles on ad networks. Ad networks collect data about users' browsing habits and interests, which is then used to target users with ads more likely to be of interest to them. iOS facilitates this process by sharing a unique identifier with app developers, the Identifier for Advertisers (IDFA) [1]. The IDFA is a 128-bit number assigned to each device that runs iOS. App developers can use the IDFA to track users' activities in a single app or across apps.

While IDFA enables the delivery of relevant ads to users, it can also be used to track users' online activities and build profiles of their personal lives. This raises privacy concerns for many users, who express concerns about the lack of transparency on data collection and usage practices [2,5,23,55,72].

To address such privacy concerns, Apple has recently introduced the App Tracking Transparency (ATT) framework in iOS 14.5 [8]. App developers must now explicitly ask users for permission before tracking their activity. To request permission for ATT, Apple requires developers to specify their purpose for tracking through an OS-provided permission alert. Developers can also optionally display a custom pre-alert screen to provide additional information to users to help them make an informed decision. If the user grants the ATT permission, iOS shares the IDFA with the app.

The unavailability of IDFA makes it more difficult for developers to track users and deliver personalized ads. To convince users to grant permission, developers customize the purpose string in the ATT permission alert and the pre-alert screen. In this regard, Apple provides guidelines for developers to create clear and straightforward *ATT alerts* (purpose strings and pre-alert screens) that describe the purpose of tracking. Yet, no system-level enforcement mechanism prevents developers from violating these guidelines. As a result, as we expose in this paper, the developers use (dark) "patterns" in ATT alerts.

Previous research has defined dark patterns as deceptive design techniques or practices used by developers or designers to trick users into taking actions unwittingly, often to benefit the developers [24]. In this paper, we focus on those dark patterns used by app developers to violate Apple's design guidelines regarding the ATT permission.

Prior work has explored various dark patterns that mobile and IoT app developers can use to deceive users into taking unintended actions [16,26,34,35,49,54]. In the context of ATT permissions, recent work has focused on how developers track users when ATT permission is denied [48] and the general perception and decision-making process of users on ATT permission [39]. However, less is known about the patterns that real-world iOS apps adopt in ATT alerts to deceive users into granting permission, and it is unclear how these patterns impact users' perception of ATT permission.

In this paper, therefore, we focus on characterizing how developers embed dark patterns in ATT alerts and how this impacts users by answering the following research questions:

RQ1: Do app developers use ATT alerts in a way that conforms to Apple's guidelines? If not, what are the different dark patterns they use?

RQ2: How do different ATT alert patterns impact the perception and understanding of users on ATT permission, and what are the implications for user privacy?

To answer these questions, we first conduct a large-scale analysis of ATT alerts by analyzing 4K iOS apps on the App Store. To achieve this, we develop dedicated static and dynamic analysis tools to automatically collect ATT alert texts for each app that contain the purpose strings in the ATT permission alert and pre-alert texts in the pre-alert screen.

Thereafter, we use Apple's app store review [7] and human interface design [38] guidelines to create an annotation guide for ATT alert texts. We then annotate the texts with five labels that characterize the different alert patterns (e.g., misleading, ambiguous) identified as violating the guidelines. We discover that only 41% of the apps follow Apple's guidelines, while 53% violate Apple's guidelines, and 6% do not clearly describe the purpose of the ATT permission.

To automate the classification of ATT alert texts, we develop a classification framework that extends a contrastive learning loss function [31] to transform alert texts into sentence embeddings. We then employ a fully-connected neural network multi-label classifier to predict the labels for each ATT alert text. Our framework achieves 90% accuracy on the annotated ATT alerts, and yields an average 8.6% increase in accuracy compared to the three baseline approaches.

Given the large number of apps displaying alert patterns that do not conform to ATT guidelines, we conducted an online user study with 114 iOS users to gauge their understanding and perception of ATT permission through different patterns used in ATT alerts. We find that users have a fundamental misconception about ATT, believing that denying ATT permission protects all of their sensitive data, such as their email address and location, when in reality, only IDFA is strictly protected by the OS. Additionally, depending on the dark patterns, (*a*) users are *more likely* to believe they will receive rewards/discounts or better app features, and they are *more likely* to misunderstand what the permission does by (*b*) being unfamiliar with common terms used in alerts, such as IDFA and identifiers, and (*c*) finding the alert texts confusing.

Overall, we expose that most developers do not follow best practices for explaining ATT permission; instead, they use deceptive or manipulative patterns to coerce users into granting permission. The use of such patterns exacerbates user confusion on ATT permissions. This leads to users having unrealistic expectations about their privacy and tracking transparency, such as earning rewards if they allow tracking. Our findings highlight the need to improve the current iOS ecosystem and future efforts to better align user understanding of iOS privacy and tracking transparency with the actual reality.

In summary, we make the following contributions:

- We analyze 4K iOS apps to study the different patterns used by app developers in ATT alerts and categorize them based on Apple's design guidelines.
- We develop a multi-label classification framework that leverages contrastive learning techniques to create a language representation model for ATT alert texts and automatically classify them into five different alert patterns.
- We conduct a 114-participant user study to gauge users' understanding and perceptions of the ATT permission given the different alert patterns. We summarize our findings and provide recommendations to improve Apple's developer guidelines for ATT alerts.

Responsible Disclosure. Given our study exposes dark patterns in ATT alerts, we have provided Apple with a comprehensive report of our findings. Apple has responded by asking for the app IDs from our dataset, and they are currently in the process of investigating the reported issue.

2 Background and Motivation

2.1 Advertising in iOS

In-App and Targeted Advertising. In-app advertising, where ads for other apps or websites are displayed, remains a prominent method for mobile app developers to profit [71]. Developers generate revenue when a user either passively watches the ad or actively clicks on it. This revenue is maximized with more user engagement, such as when a user visits the app store to download an app shortly after exposure. App developers, therefore, aim for ads that target users' interests.

Targeted ads encourage user engagement with ads, yielding higher ad profit. However, for targeted advertising, apps rely on linking users with detailed user profiles [66]. These profiles are built by tracking user activities across apps and maintained by ad networks. In iOS, this linking occurs using an identifier known as Identifier For Advertisers (IDFA) [1], which is unique to a user's device. Additionally, app developers use IDFA to link users to the ads from which they downloaded

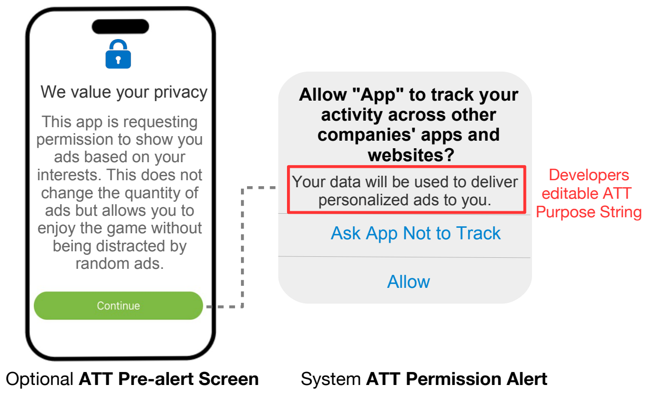

We value your privacy
This app is requesting permission to show you ads based on your interests. This does not change the quantity of ads but allows you to enjoy the game without being distracted by random ads.
Continue
Allow "App" to track your activity across other companies' apps and websites?
Your data will be used to deliver personalized ads to you.
Ask App Not to Track
Allow
Developers editable ATT Purpose String
Optional **ATT Pre-alert Screen**
System **ATT Permission Alert**

Figure 1: An illustration of ATT alerts.

their apps; a process known as attribution [10]. This helps developers evaluate their investments in ad campaigns and tailor their campaigns to reach more users.

Tracking Transparency Efforts. Despite the potential benefits of targeted advertising (e.g., exposure to ads that solely pertain to one's interest), users have raised many concerns about their privacy-sensitive data being collected by tracking and the lack of control and transparency over their tracking data [23, 56, 72]. In an attempt to address these concerns, Apple introduced a system-wide limited ad tracking (LAT) privacy setting on iOS devices. The LAT setting allows users to opt out of sharing their IDFA with advertisers and third parties [6]. This setting was disabled by default, and when enabled by the user, the OS shares a string of zeros instead of the IDFA to prevent apps from tracking the user. However, prior work has demonstrated that most users were unaware of this system-wide LAT setting [30, 61].

To provide users with better transparency and control over individual apps, Apple has recently introduced the App Tracking Transparency (ATT) permission in iOS 14.5 [22]. With the ATT permission, Apple introduces an opt-in policy where it requires all iOS apps to explicitly request users' permission to use their data for tracking. Apps prompt users at runtime by the ATT request with the option to allow tracking or ask the app not to track. If the users deny the permission, apps are not allowed to access their IDFA. This gives the users more control over how app developers track and collect their data.

2.2 Apple's Design Guidelines for ATT Alerts

To request ATT permission, Apple shows a system-provided *permission alert* with a customizable string to show the permission purpose, known as *purpose string*. Apple requires app developers to provide a clear, straightforward string to describe the purpose of requesting ATT permission and the use of tracking data. Moreover, due to the sensitivity of app tracking, Apple's guidelines encourage developers to display an optional custom screen preceding the system ATT permission alert [38], defined as a *pre-alert screen*. The pre-alert screen serves as an extension of the purpose string, as developers

Table 1: Apple's guidelines for ATT permission requests, with relevant portions of each guideline highlighted.

#	Description
G1	Apps should not require users to enable tracking in order to receive monetary or other compensation, including but not limited to gift cards and codes. Don't offer incentives for granting the request. You can't offer people compensation for granting their permission.
G2	Apps should not require users to enable tracking in order to access functionality, content, or use the app.
G3	Aim for a brief, complete sentence that's straightforward, specific, and easy to understand.
G4	Ensure your purpose strings clearly and completely describe your use of the data.

have the flexibility to describe the tracking permission purpose textually as well as visually. An example of the pre-alert screen and the ATT permission alert is shown in Figure 1.

Apple provides several guidelines for the tracking permission and the design of the ATT alerts. Generally, Apple requires that developers design clear and informative alerts that clearly inform the users about how the app uses their data for tracking [38]. We aggregate tracking permission alert and pre-alert design standards from Apple's App Store review guidelines [7] and human interface design guidelines [38] Through this, we extract four different guidelines (**G1-G4**) for ATT alerts, as shown in Table 1.

The guidelines restrict app developers from providing monetary compensation (**G1**) or access to extra functionality or content (**G2**) based on users' decision for the tracking permission. It also explicitly mentions that alerts should be straightforward and easily understood by standard users (**G3**). Alerts should also clearly mention the purpose of the permission and how tracking data will be used (**G4**).

2.3 Problem Statement

Apple's guidelines require developers to craft their own purpose strings to provide the purpose for requesting permission. Developers also have the freedom to design a custom pre-alert screen where they have more room to provide additional information and describe the benefits of tracking to users. This flexibility encourages developers to be more transparent to users about how and why they are tracking them. However, it also brings into question whether developers comply with Apple's guidelines when designing these alerts.

Due to the sensitivity of the tracking permission and the necessity of obtaining the IDFA for app developers (as discussed in Section 2.1), we anticipate that some developers may violate Apple's guidelines to influence users' decisions. Developers may include dark patterns [24] that violate Apple's guidelines in their ATT alerts to deceive users into granting permission to track. Dark patterns refer to user interface design techniques or practices that are intentionally deceptive or manipulative. These design patterns are used to trick users

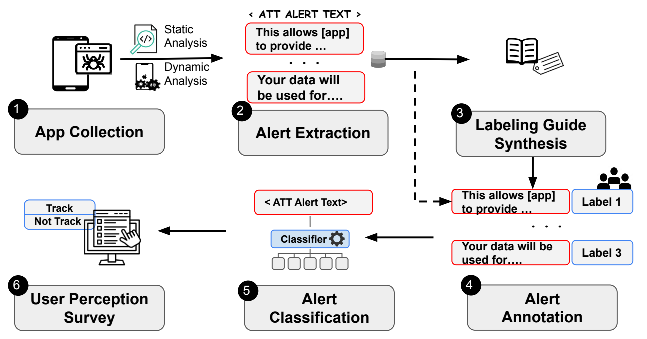


Figure 2: Overview of our approach to understanding ATT patterns and their influence on users.

into taking actions unwittingly, often to benefit developers. Additionally, such use of dark patterns may result in users having misconceptions about the tracking permission, e.g., what it is for, and what happens if permission is granted.

Therefore, our goal is to study the dark patterns that app developers use in ATT alerts and how these patterns do not conform to Apple's guidelines. From understanding the characteristics and prevalence of such patterns, we further study how alert patterns affect the perception of users of the ATT permission. Understanding the prevalence of dark patterns and their influence on users is pivotal to implementing countermeasures and designing ATT permissions that effectively communicate and instill trust among users.

3 Approach Overview

Figure 2 presents an overview of our study methodology. We build an app collection pipeline (❶) which automatically extracts the ATT purpose strings and pre-alert texts from apps on the App Store. To achieve this goal, we first design a static analysis tool, ATTSExtr, to parse the app bundles and extract the ATT purpose string (❷). Since pre-alert screens cannot be extracted directly from app code, we build PREAExtr tool that dynamically executes app bundles on an iPhone, simulates automated touch events, and takes periodic screenshots to capture the pre-alert screens.

We then study Apple's design guidelines for alerts to develop an annotation guide and identify alert patterns that violate these guidelines (❸-❹). Based on these guidelines, we develop ATTCls, an ATT alert classification framework, which leverages contrastive learning for language modeling and a fully connected neural network for multi-label classification of alert texts into one or more patterns (❺).

Guided by our findings from our ATT alert analysis, we conduct a 114-participant online user study to understand how the different alert patterns affect users' perception of the ATT permission (❻). To demonstrate to participants how ATT permissions work in a real scenario, we embedded interactive iOS app prototypes built using Marvel app interface prototyping tool [53] in an online survey.

4 ATT Alert Extraction and Analysis

To understand the different patterns used by app developers in ATT alerts and their conformity with Apple's guidelines (**RQ1**), we conduct a large-scale analysis of the ATT alerts collected from 4K apps available on the Apple App Store. Conducting this analysis involves several challenges that include (1) collecting purpose strings and pre-alerts of ATT alerts automatically, (2) identifying and annotating the alert patterns used by developers in these apps, and (3) automated classification of the ATT alerts.

4.1 Extraction of Alerts from Apps

We develop a static analysis tool ATTSExtr, which analyzes the property file of each app to extract the static purpose strings from ATT alerts. However, since pre-alerts are custom screens shown to the users before the ATT permission alert at run-time, they cannot be obtained through static analysis. Therefore, we complement our static analysis tool with a dynamic analysis tool PREAExtr.

ATT Purpose Strings. ATTSExtr starts analysis by executing the `IPA-tool` [40] to download the app packages. The `IPA-tool` is a command line tool that allows ATTSExtr to search for iOS apps on the App Store and downloads a copy of the app bundle, known as the IPA file. We analyzed the "`Info.plist`" file, which is a structured text file with a collection of key-value pairs of configuration data and included by default in the app bundle. We statically parse the file and extract the ATT purpose string stored under the key `<NSUserTrackingUsageDescription>`.

ATT Pre-alerts. The ATT purpose strings are statically accessible from the app bundles; however, collecting pre-alerts is more challenging. Pre-alerts are custom screens arbitrarily designed by developers; thus, the app bundles must be executed to capture screenshots of the pre-alert screens in real-time. To automate this process, we developed PREAExtr, which dynamically executes the app bundles on an iPhone, simulates random touch events, and periodically captures screenshots while running the app. The injection of touch events requires a rooted device (i.e., jailbreak); thus, PREAExtr runs on a jailbroken iPhone 6 (iOS 14.8).

To detail, PREAExtr leverages `CFGUTIL` [19], Apple's configurator command-line tool, to install the collected IPA files on the phone. We use SSH to control the phone from a server and send system-wide touch events using `ZXTouch` [75], an open-source touch simulation tool. We run each app for 30 secs. This interval is likely to reach ATT since apps usually request permissions when they start. To maximize the chance of clicking the correct buttons, PREAExtr divides the screen into a grid of 45 cells in the center of the screen. During the 30 secs interval, it iteratively picks a cell and simulates a touch event in its center. This approach increases the likelihood of clicking the buttons of other prompts that appear before the ATT permission, such as other permissions or privacy policy

Table 2: Summary of ATT purpose strings collection.

	All collected apps	Apps w. ATT	Non-english / Placeholders filtered	Unique ATT strings
Popular apps	3,156	1,633	4,000	1,605
Random apps	19,579	3,235		
Total apps	**22,735**	**4,868**	**4,000**	**1,605**

Table 3: Summary of ATT pre-alerts collection.

	All apps w. ATT	Apps tested	Reached ATT	Pre-alerts	Unique pre-alerts
Total apps	**4,868**	**4,680**	**2,836**	**273**	**153**

consent popups. PREAEXTR then captures screenshots every second to ensure it obtains ATT pre-alerts and transfers them to the server for further analysis.

After screenshots are recorded, PREAEXTR leverages Google's Tesseract tool [59] to extract the pre-alert text from the screenshot. We use a simple keyword search to identify tracking-related screenshots, such as pre-alert screens and privacy popups. The keywords include *"tracking", "personalized", "ads", "advertisements", "promotions"*. We then manually select the pre-alert screens for further analysis. We note that the pre-alerts may appear as on-screen popups. In such cases, PREAEXTR first crops the popup by detecting the image contours and masking the image using the contour with the maximum area. This process reduces the clutter from the popup's background. An example is shown in Appendix A.

Alert Text Collection Results. To analyze the different patterns used by app developers in ATT purpose strings and pre-alert texts, we start by crawling app IDs from the App Store. We collected two sets of apps, `popular` and `random`.

For `popular` set, we queried the top 100 popular apps in 32 different genres. This search results in a total of 3,156 app IDs. To capture a random set of apps representative of the Apple App Store, we collect `random` set using "`robots.txt`" [62] file, which is designed to assist the navigation of search bots on the store website. We randomly sample the robots file to crawl a large collection of 19,579 apps. Our combined set of apps contained a dataset of 22K apps.

We note that we abide by ethical considerations when we collect data from the Apple App Store. To collect 22K apps, we make at most 1 query per minute. Additionally, we did not circumvent rate limits or use proxies.

We run ATTSEXTR on our dataset. Among the `popular` set, we found 1,633 apps that have ATT permission strings in their property list files, and, for the `random` set, 3,235 of the apps included ATT strings. We detail the distribution of app genres for each set in Appendix A. In our analysis of ATTSEXTR results, we found that a few of our collected apps use a placeholder in the property list instead of the actual purpose string or are not in English. We removed such apps from our dataset, resulting in a total of 4K apps with ATT strings. Additionally, we found that some of the purpose strings used in the Apps are repetitive. We further filtered the non-unique strings, which resulted in a total of 1,605 unique purpose strings.

We summarize our data collection for purpose strings and pre-alerts in Tables 2 and 3. For pre-alerts, we run PREAEXTR on a total of 4,680 apps. We note that, in total, we collected 4,868 IPA files. Yet, 188 of them required an updated version of iOS, which did not have a jailbreak at the time of our analysis. PREAEXTR is able to reach the ATT permission alert for 2,836 apps (60.6%) at run-time. Among the 2,836 apps, it recorded 273 apps that have pre-alerts. Lastly, we filter the repetitive pre-alerts, yielding 153 unique pre-alerts.

To evaluate the effectiveness of PREAEXTR in extracting the pre-alerts, we assess it on a random sample of 480 apps. Out of 480 apps, 63.3% of the apps show the ATT permission alerts. For the rest of the apps that did not obtain the ATT permission alert, we manually reinstalled them on the phone and used each app for approximately one minute. We found that 14.4% of the apps do not show the ATT permission, and 5.2% of the apps require users to sign in. Additionally, PREAEXTR does not click the correct buttons to display the ATT permission alert for 17.7% apps. We analyzed these cases and found that the ATT alert appears after other popups or ads, which are displayed in full-screen mode with a small exit/skip button that is hard to capture due to its small size or unconventional placement. The ATT alert is only displayed after exiting these screens. We also evaluated the pre-alert screenshot detection by our keyword search tool on over 50 apps. We found that the tool has a negligible false-negative rate (FN=0.02). The tool results in a high false-positive rate since it detects other popups (such as privacy policy consent popups), which we manually filtered in our analysis. Overall, out of the 81% apps that show an ATT permission request, PREAEXTR triggers the permission for 77.8% of them.

4.2 Identifying ATT Alert Patterns

We use Apple's App Store review [7] and human interface design [38] guidelines to produce a labeling guide. This guide contains questions that allow an annotator to label an ATT alert text into five different labels.

4.2.1 Annotation of ATT Alerts

Developing ATT Alert Annotation Guide. To annotate alert texts with an appropriate label, two authors performed an initial study on alert texts of 250 random apps and created a labeling guide. We create this guide by checking the patterns in ATT alerts based on how these alerts do not conform to Apple's official guidelines (**G1-G4** in Table 1). In Table 4, we present example alert texts with their associated labels and detail our annotation process below.

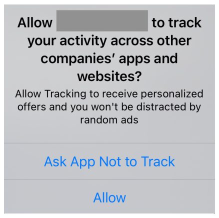
We found apps that *directly* incentivize users by promising rewards, coupons, and special offers ((a) and (b)). Additionally, some apps *indirectly* incentivize users by mentioning that granting ATT permission will enable them to use apps for free ((c)). We label such alerts `Incentive`.

Some apps indicate that users will get better content or user experience if they allow tracking ((d)). Another set of apps

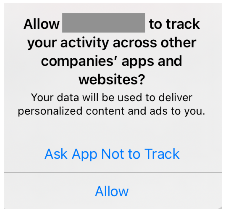
Table 4: Examples of ATT alerts of real-world apps assigned to five different labels with our data annotation guide.

Label	**ATT Alert Text**[†]
Incentive	(a) *This allows [app] to determine if you are eligible to receive a reward after creating your account.*
	(b) *This helps us serve relevant coupons and ads that are tailored to you.*
	(c) *We can offer our service for free to you only by delivering personalized ads.*
Misleading	(d) *This will allow us to offer more relevant content and provide you with a better experience throughout the app.*
	(e) *It will help us deliver a more personalized experience, and fewer ads.*
	(f) *This will help [app] to recruit only professional delivery partners like you. Bonus: the app will send fewer irrelevant messages.*
Ambiguous	(g) *App would like to access IDFA for tracking purpose.*
	(h) *Identifier will be used for attributing installs (from Facebook, Google) and performance measurement in marketing companies.*
Complete	(i) *This helps [app] and our ad partners provide you with a more personalized ad experience.*
	(j) *By clicking "Allow Tracking", your consent to your data being used for analytics purposes and personalized advertisements.*
Other	(k) *Your data allows for the optimization of the in-app ad experience for you.*
	(l) *We personally do not track any user information.*

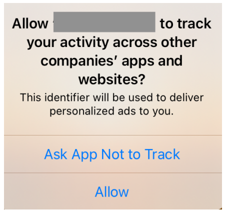
[†] We redact app names in this table.


Allow [redacted] to track your activity across other companies' apps and websites?
Allow Tracking to receive personalized offers and you won't be distracted by random ads
Ask App Not to Track
Allow

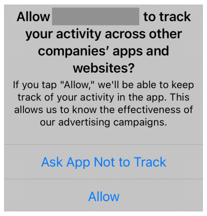
(a) `Incentive`


Allow [redacted] to track your activity across other companies' apps and websites?
Your data will be used to deliver personalized content and ads to you.
Ask App Not to Track
Allow

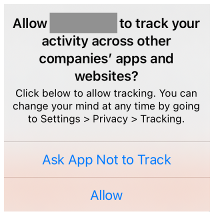
(b) `Misleading`


Allow [redacted] to track your activity across other companies' apps and websites?
This identifier will be used to deliver personalized ads to you.
Ask App Not to Track
Allow

(c) `Ambiguous`

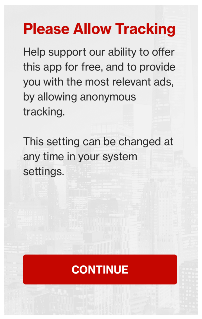

Allow [redacted] to track your activity across other companies' apps and websites?
If you tap "Allow," we'll be able to keep track of your activity in the app. This allows us to know the effectiveness of our advertising campaigns.
Ask App Not to Track
Allow

(d) `Complete`

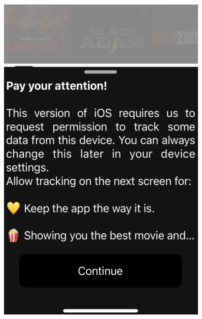

Allow [redacted] to track your activity across other companies' apps and websites?
Click below to allow tracking. You can change your mind at any time by going to Settings > Privacy > Tracking.
Ask App Not to Track
Allow

(e) `Other`

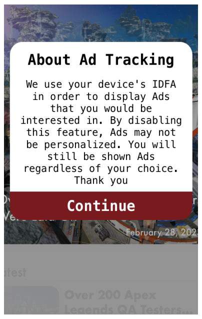
Figure 3: Examples of ATT permission alerts with purpose strings of different patterns.


Please Allow Tracking
Help support our ability to offer this app for free, and to provide you with the most relevant ads, by allowing anonymous tracking.
This setting can be changed at any time in your system settings.
CONTINUE

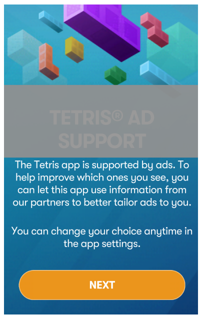
(a) `Incentive`


Pay your attention!
This version of iOS requires us to request permission to track some data from this device. You can always change this later in your device settings.
Allow tracking on the next screen for:
💛 Keep the app the way it is.
🍿 Showing you the best movie and...
Continue

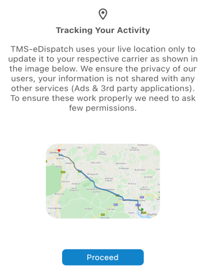
(b) `Misleading`


About Ad Tracking
We use your device's IDFA in order to display Ads that you would be interested in. By disabling this feature, Ads may not be personalized. You will still be shown Ads regardless of your choice. Thank you
Continue

(c) `Ambiguous`


TETRIS® AD SUPPORT
The Tetris app is supported by ads. To help improve which ones you see, you can let this app use information from our partners to better tailor ads to you.
You can change your choice anytime in the app settings.
NEXT

(d) `Complete`


Tracking Your Activity
TMS-eDispatch uses your live location only to update it to your respective carrier as shown in the image below. We ensure the privacy of our users, your information is not shared with any other services (Ads & 3rd party applications). To ensure these work properly we need to ask few permissions.
Proceed

(e) `Other`

Figure 4: Examples of pre-alert screens with different patterns.

promises that granting permission will yield fewer ads ((e)) and allow them to receive a bonus of an app-related feature ((f)). We label such alerts `Misleading`.

We also discover that some developers use technical terms or descriptions vague or difficult to understand for a standard user. For instance, some alert texts contain *IDFA* or *device identifiers* ((g)), and others mention attribution of marketing campaigns ((h)). We label such apps `Ambiguous`.

If an app is not labeled with `Misleading`, `Ambiguous`, or `Incentive`, we check whether the alert informs the user about the purpose of the ATT permission request and how their data will be used. Apps passing this check are labeled `Complete` ((i) and (j)). If no label matches the alert text, we label the alert as `Other`. With `Other`, we refer to alerts that do not have misleading, ambiguous, or incentive patterns, yet still do not explicitly express the ATT permission purpose ((k) and (l)).

During our annotation process, we found some alert texts that fail to conform to multiple ATT guidelines. For instance, an alert includes both `Incentive` and `Misleading` statements, e.g., *"We cannot send you gifts and offers without your permission, we will try to send you only the most important and attractive content!"*. Thus, our annotation guide allows associating a single alert with multiple labels. We present our labeling guide in our project repository [9].

To further illustrate the disclosures of ATT alerts, we provide examples for the different patterns of ATT purpose strings and pre-alert screens in Figures 3 and 4, respectively. For `Incentive` patterns, Figure 3a presents an example mentioning personalized offers, while the pre-alert in Figure 4a indirectly incentivizes the user by mentioning offering the app for free. Similarly, the examples in Figure 3b and Figure 4b mention patterns as "*personalized content*" and "*showing the*

Table 5: Distribution of annotated ATT alerts.

Label	Unique Alerts Purpose Strings	Unique Alerts Pre-alerts	App Alerts Purpose Strings	App Alerts Pre-alerts	Avg. Rate[†] Counts (Std)	Unique Vendors
`Incentive (Inc)`	109 (6.8%)	38 (24.8%)	168 (4.2%)	96 (35.2%)	571,278 (1.78M)	92
`Misleading (Misl)`	591 (36.8%)	38 (24.8%)	810 (20.2%)	76 (27.8%)	169,518 (1.24M)	527
`Ambiguous (Ambg)`	138 (8.6%)	5 (3.3%)	989 (24.7%)	5 (1.8%)	34,210 (164K)	857
`Inc, Misl`	53 (3.3%)	11 (7.2%)	66 (1.7%)	22 (8.1%)	206,700 (422K)	31
`Inc, Ambg`	8 (0.5%)	0 (0%)	8 (0.2%)	0 (0%)	81626(159K)	5
`Misl, Ambg`	68 (4.2%)	3 (2%)	91 (2.3%)	4 (1.5%)	119,862 (319K)	67
`Inc, Misl, Ambg`	3 (0.2%)	0 (0%)	3 (0.08%)	0 (0%)	27,153	1
`Complete`	437 (27.2%)	51 (33.3%)	1631 (40.7%)	62 (22.7%)	115,857 (594K)	1,351
`Other`	198 (12.3%)	7 (4.6%)	234 (5.9%)	8 (2.9%)	471,607 (2.39M)	157
Total	**1,605**	**153**	**4,000**	**273**	—	—

[†] Represent the number of ratings for the app. Standard deviation is presented in parentheses.

best movies". Both examples fall under the `Misleading` pattern since they refer to better functionality. The `Ambiguous` examples (Figures 3c and 4c) mention vague terms e.g., "*identifier*" and "*IDFA*". Moreover, the examples labeled with `Other` (Figures 3e and 4e) do not fit any of the aforementioned patterns, yet they fail to explicitly state the permission purpose.

Guide Validation and Annotating Alerts. Before labeling alert texts from our dataset of unique 1,605 purpose strings and 153 pre-alerts, we first validate our labeling guide. To do so, two authors independently label a random sample of 105 purpose strings and all 153 pre-alerts.

To measure agreement between annotators, we use Krippendorff alpha [50], which supports multi-label annotations. Authors achieved high agreement ($\alpha > 0.8$), reconciled differences and make necessary modifications to the annotation guide, e.g., providing examples and additional clarifications for each labeling guide question. After guide validation, to label the remaining 1,500 alert texts, we sought volunteer annotators through internal posts in the Slack channel of our institution. Through this method, we recruited four security and privacy researchers. We first met with volunteers to explain our labeling methodology and guideline. After clarifying doubts, annotators expressed confidence in the labeling guide.

These four volunteers and two authors were divided into three groups comprising two annotators each. We leverage the open-source tool Doccano [27] for collaborative annotation. Each group labeled 500 alert texts over three sessions (100 in the first session and 200 in the second and third sessions). Annotators within each group independently labeled alerts. The annotators met after each session to resolve the annotation conflicts. We note that in the first round, each group achieved sufficient agreement ($\alpha > 0.667$) [50], and, by the second and third rounds, each group achieved high agreement ($\alpha > 0.8$).

4.2.2 Findings from Annotated Data

Table 5 shows the distribution of labels for the annotated purpose strings and pre-alerts. For the unique purpose strings, we find that 27.2% are `Complete` while 36.8%, 6.8%, and 8.6% alerts are `Misleading`, `Incentive`, and `Ambiguous`, respectively. To better understand and characterize alert patterns used by app developers in the wild, we reflect the annotations to the entire dataset of 4K apps, including both popular and random apps. We, thus, mark all the non-unique alert texts with the same labels as their corresponding alert text labels in the annotated dataset. We summarize our data collection and labeling results for the full apps dataset in Figure 5.

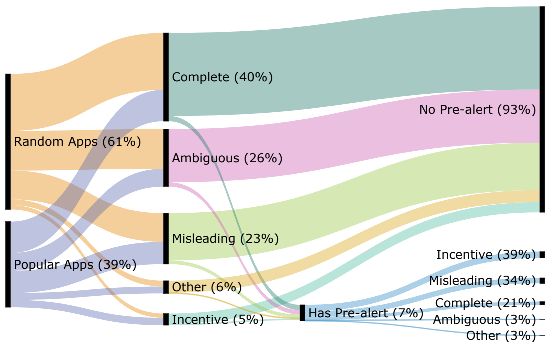


Figure 5: Summary of apps collection and labeling.

To detail, 60% of the apps use repeating alert texts, with the `Ambiguous` pattern repeating most often (86% of `Ambiguous` patterns are repeating). In Table 5, we show the distribution of apps after mapping the annotations to the complete dataset. We also show some app statistics, including the average rating counts (and std), which reflects the popularity of apps. We note that the std values are high for all patterns since we use diverse data sets with popular and random apps. We also report the number of unique vendors per app pattern. When examining label distribution across app categories (e.g., games, business, social networking), we find no correlation between app category and pattern labels. We show the mapping between labels and app categories in Appendix A (Figure 3).

Observing the app patterns, we find that 40.7% of the ATT purpose strings are `Complete`, 25% are `Ambiguous`, 20% are `Misleading`, 4% are `Incentive`, and 6% are annotated as `Other`. As anticipated, more than 60% of app developers are using patterns that violate Apple's guidelines, which can influence users' decisions to grant ATT permission.

Similarly, for the pre-alerts, we show the analysis results for the unique and repeated pre-alert texts. The unique pre-alerts show 33.3% `Complete`, 3.3% `Ambiguous`, 24.8% `Misleading`, 24.8% `Incentive`, and 4.6% as `Other`. Interestingly, the pre-alerts of the full apps' set show that only 22.7% are `Complete`, while the remaining 77.3% are distributed among the rest of the patterns, with 35.2% `Incentive`, 27.8% `Misleading` and 8.1% both `Incentive` and `Misleading`. We also notice that while the `Incentive` pattern is the least common among purpose strings, it is the most frequent pattern in pre-alerts. This highlights that most apps that show customizable pre-alerts leverage them to persuade users into granting the ATT permission by using these patterns.

Finally, we study the combinations of pattern labels among purpose strings and pre-alerts. We detail the distribution of labels for both purpose strings and pre-alerts in Appendix A (Table 1). We found that only 36.6% of the apps use similar patterns in the purpose strings and pre-alerts, with the rest using different patterns. We also found that among the apps with `Complete` purpose strings that show pre-alerts, about

73.5% have `Incentive` or `Misleading` pre-alerts, while only 20.7% show `Complete` pre-alerts. This indicates that even apps with `Complete` purpose strings still try to manipulate users by showing dark patterns in their pre-alerts.

4.3 Automated Classification of ATT Alerts

To automate the classification of ATT alerts, we develop a multi-class classification framework, ATTCLS, which predicts the labels for each ATT alert text.

ATT Alert Embedding Representation. To accurately classify the ATT alerts, we first extract semantic features from the alert text. To achieve this goal, we extend a contrastive learning-based sentence embedding framework, SIMCSE [31], to transform each alert text into an embedding space representation. This framework uses a contrastive objective on top of pre-trained language models such as BERT [44] or ROBERTA [52] to produce sentence embeddings from limited labeled data. We leverage this contrastive objective to learn an effective sentence representation for each alert text, even with a small training dataset, such that alerts with the same class label are closer to each other while alerts with different class labels are farther apart in the embedding space.

Our sentence embedding framework learns from pairs of ATT alerts that are semantically related. To generate positive pairs, we consider combinations of alerts with the same class labels as they are contextually similar. Therefore, assuming a set of m paired examples $\mathcal{D} = \{(x_i, {x_i}^+)\}$ for $i \in \{1, m\}$, where x_i and ${x_i}^+$ is considered as the positive pair, we first extract language representations em_i and ${em_i}^+$ for x_i and ${x_i}^+$, using a pre-trained language model. We then fine-tune these representations using a cross-entropy training objective computed for a mini-batch of N pairs:

$$l_i = -\log \frac{e^{\mathrm{sim}(em_i, {em_i}^+)/\tau}}{\sum_{j=1}^{N} e^{\mathrm{sim}(em_i, {em_j}^+)/\tau}} \tag{1}$$

Here τ is a temperature scaling hyperparameter and $\mathrm{sim}(a,b)$ is the cosine similarity between vectors of a and b ($a^T b/(\| a \| \| b \|)$). The output of the framework is a sentence embedding for the ATT alert that captures the semantic and contextual meaning of the alert text.

ATT Multi-Class Pattern Prediction. Given that ATT alerts may fail to conform to multiple guidelines, we consider the problem of inferring the ATT pattern as a multi-class prediction task. We design a fully-connected neural network as part of ATTCLS that takes an ATT alert embedding extracted through SIMCSE and predicts its class labels.

The model leverages a sigmoid objective function for label prediction. Specifically, we add a sigmoid function for each of the five labels (identified in Section 4.2) as $\mathrm{sigmoid}(h_i) = \frac{1}{1+e^{-h_i}}$ which outputs a score in the range $[0,1]$.

In this way, the network outputs a vector $\hat{y}_i$ with a length equal to the number of class labels. An alert is assigned the label j if its sigmoid function score $\geq T$. This vector is then fed into a logistic cross-entropy loss function, which outputs the final class predictions for the alert.

At inference time, if the sigmoid score is the highest for the `Complete` or `Other` class, we only assign a single class label corresponding to the highest sigmoid function. This is because the alerts containing these patterns do not display the other three patterns, `Misleading`, `Incentive`, or `Ambiguous`.

Implementation. We implement ATTCLS in Python 3.10 using Simple Transformers library [65] and SimCSE model implementation [31]. We leverage our full labeled ATT alert dataset consisting of 4K ATT purpose strings and 273 pre-alerts to train and test our framework. We use stratified sampling to split the ATT strings into 90% for training and 10% for testing. To train the contrastive embedding framework, we generate 150K positive pairs from our training data set. We leverage ROBERTA [52] as the pre-trained language model to obtain the intermediate language representations. We use the fine-tuned hyperparameters provided in [31] for training the supervised SIMCSE on our ATT alerts dataset. We do not fine-tune the model on the test set or pre-alerts. We set the sigmoid function score T in our multi-class classifier to 0.5 as it yields the best accuracy in our validation dataset.

Table 6: Classification results for [purpose strings / pre-alerts].

Model	Precision	Recall	F-measure
ATTCLS	**0.90 / 0.79**	**0.90 / 0.82**	**0.90/ 0.80**
$\mathrm{ATTCLS}_{\mathrm{SIMCSE}_{\mathrm{BERT}}}$	0.88 / 0.75	0.87 / 0.77	0.88 / 0.76
$\mathrm{ATTCLS}_{\mathrm{BERT}}$	0.88 / 0.76	0.89 / 0.77	0.89 / 0.77
Random Forest	0.92 / 0.86	0.68 / 0.68	0.78 / 0.76

Table 7: F-measure results per alert label.

Alert Label	Purpose strings	Pre-alerts
`Incentive`	1.00	0.87
`Misleading`	0.88	0.81
`Ambiguous`	0.97	0.57
`Complete`	0.90	0.66
`Other`	0.78	0.60

4.3.1 Effectiveness of ATTCLS

We measure the framework's effectiveness through precision, recall, and F-measure. We compute each metric as the average of the corresponding metric per class label. Table 6 (row 1) shows the average precision, recall, and F-measure for the ATT purpose strings and pre-alert texts. For the purpose strings, ATTCLS achieves a 90% accuracy. We show the F-measure results per alert label in Table 7. The accuracy is slightly lower for the `Other` label since alerts labeled as `Other` do not cover a specific pattern and lack semantic structure.

For the pre-alerts, ATTCLS achieves an accuracy of 78%. The accuracy of pre-alerts decreases compared to purpose strings as the app developers have the flexibility to explain the purpose of ATT permission with more words and phrases. This results in pre-alert texts that contain patterns with lexical and semantic ambiguity, yielding lower accuracy.

Comparison with Baseline Approaches. We compare the effectiveness of ATTCLS with three baseline approaches. These approaches are widely used for similar text classification tasks, such as mobile privacy settings and policies classification [36, 45] and phishing detection [21]. The first baseline is a Random Forest classifier that leverages vectorization to obtain a text representation of the alerts. Second, we train a classifier that uses the BERT model to extract the embedding representation of alerts and then perform multi-label classification through a fully connected neural network. Lastly, we replace the RoBERTa model with the BERT model in SimCSE model of ATTCLS as the pre-trained language model.

Table 6 (row 2-4) shows the results for these baseline models compared to our classification framework for both purpose strings and pre-alerts, respectively. ATTCLS achieves a higher accuracy of 90% compared to all three baseline models.

5 User Attitudes Towards ATT Alerts

In this section, we pivot to studying user understanding of the ATT permission and how different alert patterns influence user perception of ATT permission alerts. To achieve this goal, we designed an online survey and divided our second research question **RQ2** (*How do different ATT alert patterns impact the perception and understanding of users on ATT permission, and what are the implications for user privacy?*) into the following four sub-research questions:

SQ1: How do users perceive the purpose of the ATT permission for different alert patterns?

SQ2: How do users' tracking preferences change for different alert patterns?

SQ3: What is the influence of `Misleading` and `Incentive` alerts on users' understanding of ATT?

SQ4: Do users find different alert patterns vague or difficult to comprehend?

5.1 Study Design

Our study's objective is to measure users' perception of the ATT permission given different patterns used in alerts. Therefore, we perform a between-subjects study with five groups corresponding to the five alert patterns. We conducted an online user survey using Qualtrics [60]. To simulate a realistic scenario, we embedded an interactive iOS app prototype in our survey implemented using the Marvel app tool [53]. We implemented interface prototypes for 20 real-world apps, four apps per alert pattern. Selected apps represent common textual patterns exhibited in each alert pattern.

We randomly selected one app prototype for each participant (all alert texts displayed to different participants are in Appendix B.3). We then requested participants to interact with the prototype. We displayed the initial screens in the app, including the ATT alert. After making a selection on the alert, we displayed the message *"You finished using this app!"* and a random code of 8 digits and letters. This code is used as an attention check; we asked the users to use the app until they see this message and type the displayed code in a text box.

We divide our survey into two main blocks. First, we display the Marvel app prototype and ask questions related to the alert displayed in the app (`Alert-Questions`). Questions comprised Likert scales (to measure user preference/familiarity/confidence) and open-text questions (where users were able to justify their responses or provide further detail). Second, we ask general questions about prior knowledge of users on app tracking and ATT (`Tracking-Questions`). We also collect demographic data and users' technical experience. `Tracking-Questions` were asked at the conclusion of the survey to prevent influencing participants' responses.

Ethics. We obtained IRB approval prior to our study. Our study was deemed as "IRB exempt" by our institution's IRB office. IRB defines exemption as not requiring the study to be reviewed by IRB board members in a convened meeting. To minimize harm, our study was reviewed by a primary reviewer and analyst from the IRB office, who worked with the authors to ensure ethical protocol design. Before starting the survey, we ask the participants to accept an online consent form where we explain the study details and participants' rights. We avoid collecting personally identifiable data other than demographic data. We remove any other identifiers and store the responses in an encrypted online storage.

5.2 Recruitment

After conducting pilot studies to carefully revise our survey (detailed in Appendix B.1), we recruited participants on Prolific [58], specifying three criteria: (1) Participants were required to use iOS as their primary phone operating system, (2) reside in the US, and (3) be above 18 years of age. Our study was advertised as a perception study of permissions in iPhones. We recruited a total of 121 participants. We rejected responses from 7 participants who failed attention checks (in accordance with Prolific guidelines). The accepted 114 participants received a $3 reward upon survey completion.

Participants are 46.5% males and 50% females. 21% of the participants have technical experience, either by receiving an education in computer science, software development, or other technical fields. Complete demographic breakdown of the survey participants are presented in Appendix B.2 (Table 3).

5.3 Data Analysis

Qualitative Analysis. We thematically analyze participants' responses to open-text questions. Two authors conducted inductive coding independently. They generated iteratively refined codes before grouping them under themes. Authors met to reconcile differences. We note that authors achieve high inter-rater agreement (Cohen's Kappa [37], $\kappa > 0.80$).

Quantitative Analysis. We employed three statistical significance tests to measure significant differences and associations (between categorical data). We use a one-way ANOVA test [29] to test for significant differences across alert patterns (between subjects). We use the Wilcoxon signed-rank test [29] to test for significant differences within subjects (e.g., how

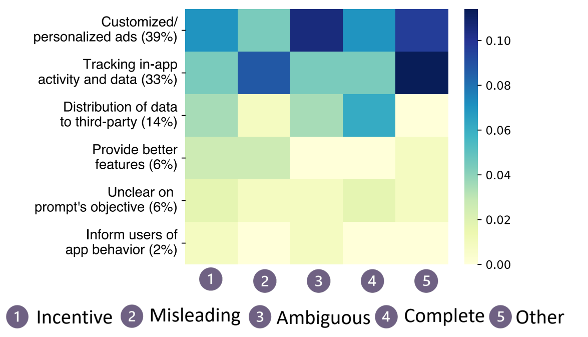


Figure 6: Users perception of what permissions are for - percentages with respect to total participants.

scores change in *"Allow"* vs. *"Not track"* scenarios). To test for association between alert patterns and themes extracted from inductive coding, we use the Fisher-Freeman-Halton (FFH) test [11]. We note that the Fisher's Exact test only considers association in 2-by-2 tables, yet our study has five alert patterns and more than two themes for inductively coded responses, which the FFH test supports.

For all tests, we assume a null hypothesis (H_0) of no difference/association and an alternative hypothesis of significant difference/association (H_1). We reject H_0 and accept H_1 when the p value is less than a threshold, and we report p values when there is a significant difference/association. For most tests, we assign a threshold of 0.05. We do note in relevant subsections when we leverage Bonferroni correction [63], which we apply to a family of tests to adjust the corresponding p value. When results are significant, we report the effect size using the Common Language Effect Size (CLES) [57].

5.4 Results and Findings

General Understanding of ATT. We asked participants to score familiarity with app tracking and ATT (1 and 5, with 5 being very familiar). They report, on average, moderate familiarity with app tracking (mean=3.1, std=1.2) but a lack of familiarity with ATT (mean=1.7, std=1.1).

When prompted to divulge how they had heard of these terms, 31% and 24% had heard of app tracking from social media and the app store, respectively. Lesser common sources comprised other blogs or websites, e.g., tech-related blogs, forums, (22%) and Apple's Website (8%), additional sources such as news or online classes (8%). When asked how they had heard of ATT, 12% mentioned social media and 13% mentioned other online blogs or websites. Participants also mentioned Apple's app store (10%), Apple's website (1%), and other sources (e.g., Facebook/news, 3%).

When comparing participants with and without claimed technical experience, a one-way ANOVA test reveals no significant differences in their familiarity with app tracking ($p = 0.54$) or ATT ($p = 0.18$). This suggests that technical experience is a poor indicator of familiarity with ATT.

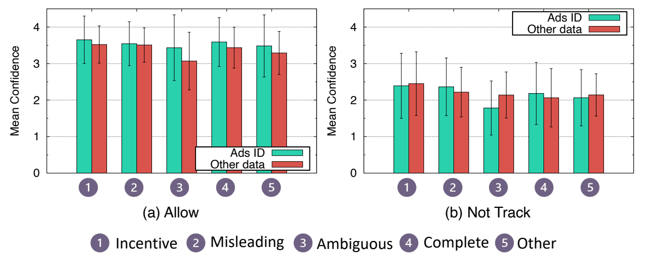


Figure 7: Participants' mean confidence about whether ad IDs vs other data types could be used for tracking.

5.4.1 SQ1: *Users' Perceptions of ATT Alert Purpose*

User Interpretations of the ATT Permissions. We asked participants what they believed the permission was used for in an open-text response. Figure 6 presents the six themes of how they interpret ATT permissions. We found no association between alert patterns and perception of the permission ($p > 0.05$). Broadly, they believe permissions are either for customized/personalized ads (39%), tracking in-app activity/data (33%), or distribution of data to third-party (14%). A minority of them believed it was intended to inform users of app behavior (2%) or provide better app features (6%) through tracking. Participants also occasionally note the uncertainty in what permissions are used for (unclear on alert's objective, 6%). Although customized advertising is the most prevalent theme, none of the participants who associate ATT permissions with ad support mention IDFA or identifiers.

Data Sharing. We asked participants to report how confident they were that certain data types could be used for tracking in both possible scenarios; when the permission response is either *"Allow"* and *"Not track"*. They were asked to score their confidence level as one of four options ("I am sure it can be used" encoded as 4, and "I am confident it cannot be used" as 1), for the advertising ID and other common data types (IP address, email, location, phone name, and phone model).

Figure 7 presents the mean confidence level for advertising ID and aggregate of all other data types in the *"Allow"* and the *"Not track"* scenarios. Interestingly, participants report high confidence that **both** advertising ID and other data types could be used to track them, if *"Allow"* is selected, irrespective of the pattern (no significant difference, $p > 0.05$, between patterns in both advertising ID and other data types). This confidence drops in the *"Not track"* scenario. Table 8 presents confidence levels in finer granularity across patterns and for each data type. We apply a multiple-test correction (Bonferroni correction) to reduce family-wise error as we consider different tests across data as a family. After correction, we independently tested each pattern and each data (e.g., *Is there a difference in confidence level between "Allow" and "Not track" for advertising ID in* `Ambiguous` *alert?*), we observe a significant confidence level decrease ($p < 0.0083$), for all data types across each pattern except for two data types in the `ambiguous` label: (1) phone model and (2) phone name.

Table 8: Participants' confidence level of whether different data items could be accessed for tracking, assuming the user selected *"Allow"* or *"Not track"* in the ATT permission prompt. 4: "I am sure it can be used", 3: "I think so", 2: "I am not sure", 1: "I am confident it can't be used". We aggregate values and present the mean and standard deviation (Mean [±Stdev])

	Assuming the user selected *"Allow"*					**Assuming the user selected *"Not track"***				
Data type	**Incentive**	**Misleading**	**Ambiguous**	**Complete**	**Other**	**Incentive**	**Misleading**	**Ambiguous**	**Complete**	**Other**
Advertising ID	3.7 [± 0.7]	3.6 [± 0.6]	3.4 [± 0.9]	3.6 [± 0.7]	3.5 [± 0.9]	2.4 [± 0.9]	2.3 [± 0.8]	1.8 [± 0.7]	2.2 [± 0.9]	2.0 [± 0.8]
IP Address	3.6 [± 0.6]	3.6 [± 0.7]	2.9 [± 1.0]	3.5 [± 0.6]	3.4 [± 0.9]	2.7 [± 1.1]	2.2 [± 0.9]	2.2[± 0.9]	2.1 [± 0.9]	2.3 [± 0.9]
Email	3.3 [± 0.7]	3.3 [± 0.8]	3.1 [± 0.9]	3.3 [± 1.0]	3.2 [± 0.8]	2.4 [± 0.9]	2.2 [± 0.7]	2.1 [± 0.9]	2.0 [± 0.9]	2.2 [± 0.9]
Location	3.8 [± 0.5]	3.8 [± 0.5]	3.6 [± 0.8]	3.7 [± 0.7]	3.7 [± 0.6]	2.5 [± 1.0]	2.2 [± 0.9]	2.0 [± 0.8]	2.0 [± 1.1]	2.2 [± 0.9]
Phone Namel	3.6 [± 0.6]	3.7 [± 0.5]	3.0 [± 1.0]	3.4 [± 0.8]	3.3 [± 0.7]	2.3 [± 0.9]	2.2 [± 0.9]	2.3 [± 0.8]	2.1 [± 0.9]	2.2 [± 0.8]
Phone Model	3.3 [± 0.9]	3.4 [± 0.8]	2.8 [± 1.0]	3.3 [± 0.8]	3.2 [± 0.8]	2.3 [± 0.9]	2.2 [± 0.8]	2.1 [± 0.9]	2.1 [± 0.9]	2.0 [± 0.7]

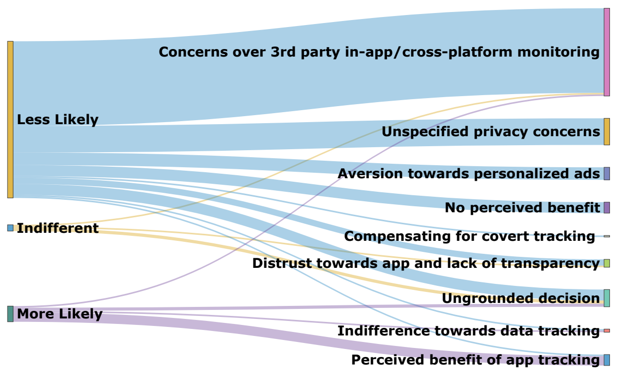


Figure 8: Participant justification for preference towards/against tracking.

Effect sizes for significant results are moderate (CLES ~0.6).

We posit that this general decrease is due to users holding an (incorrect) belief that denying the ATT permission protects **all** of their sensitive data from being used for tracking instead of solely their advertising ID. On the contrary, ATT permission only automatically protects the advertising ID (as explained in Section 3). Other data types are only protected by Apple's privacy policy and the operating system does not prevent apps from accessing them. Hence, apps may still use other data to track users even without users granting explicit permission. In fact, this behavior is common, as pointed out by related research on app tracking in iOS [48].

`Finding-1:` Participants predominantly attribute ATT permissions to (1) customized advertising (2) tracking app activity/behavior or (3) distributing data to third parties. Irrespective of alert pattern, participants incorrectly believe that selecting *"Not track"* protects various data types beyond Advertising ID.

5.4.2 SQ2: *Users' Preferences of Tracking*

We measure participants' preference towards selecting *"Allow"* or *"Not track"* and whether their preference changes between alert patterns. Since they are not using their personal device, we opt for a Likert scale with 1 referring to *more likely to select "Allow"* and 5 referring to *more likely to select "Not track"* instead of binary options (*"Allow"*, *"Not track"*).

We find that participants express a general disinterest towards tracking regardless of the alert, with most leaning towards *"Not track"*. Mean scores range between 4.4 to 4.7 (no significant difference between alert patterns, $p = 0.7240$).

We also asked participants to provide reasoning for their scores, as presented in Figure 8. We map the reasons they provide to their tracking preference scores, which we grouped into three categories: (1) *less likely* to allow tracking, scores > 3, (2) *indifferent*, score $= 3$, and (3) *more likely* to allow tracking, score < 3. We observed an association between the reasons and scores ($p = 0.0001$).

A lesser likelihood is associated with a negative perception of data tracking. As seen in Figure 8, the negative perception is expressed among six themes. 49% have concerns over third-party in-app/cross-platform monitoring (e.g., *"I don't want this app tracking me or giving my info to 3rd parties"*, *"Whatever I do outside of that app is none of that app's business"*). Other reasons include aversion towards personalized ads (7%), no perceived benefit of tracking (6%), and distrust towards the app and lack of transparency (4%). Interestingly, one participant (1%) avoids data tracking on apps to compensate for covert tracking that occurs in other online spaces (*"There's enough stuff online tracking my activity that I don't know about. The ones I do know about, [I say] please do not."*) We also note that 15% attribute their scores to unspecified privacy concerns, since they note being generally cautious about privacy but avoid providing explicit reasons. Participants who are indifferent or more likely to opt for tracking provide a variety of other reasons, including the perceived benefit of app tracking (6%) and indifference towards data tracking (2%). 10% of them admit not having a concrete reason for their score (e.g., *"I don't care which I pick"*).

`Finding-2:` Participants are *more inclined* to select *"Not track"*, regardless of alert pattern, with most attributing this choice to a strong preference against data tracking.

5.4.3 SQ3: *Influence of Misleading and Incentive Alerts*

For *"Allow"* and *"Not track"* scenarios, we asked participants to score likelihood (1-5, with 5 being extremely likely) of (1) getting a reward or extra discount and (2) getting access to better app features. Results are shown in Table 9. Participants provide low likelihood for rewards/discounts across all patterns in *"Not track"*. Those exposed to `Incentive` alerts provided

Table 9: Participants' self-reported likelihood for receiving better rewards or discounts (Mean [±Stdev]).

Scenario	Better Rewards/Discount				
	Incentive	Misleading	Ambiguous	Complete	Other
"Allow"	2.1 [± 1.5]	1.8 [± 1.0]	1.3 [± 0.7]	1.8 [± 0.9]	1.5 [± 0.7]
"Not track"	1.2 [± 0.4]	1.4 [± 0.8]	1.1 [± 0.3]	1.4 [± 1.0]	1.2 [± 0.9]
Scenario	**Better App Features**				
	Incentive	**Misleading**	**Ambiguous**	**Complete**	**Other**
"Allow"	1.7 [± 1.0]	2.1 [± 1.3]	1.4 [± 0.7]	1.6 [± 1.0]	1.8 [± 1.0]
"Not track"	1.2 [± 0.5]	1.3 [± 0.7]	1.1 [± 0.3]	1.4 [± 1.0]	1.1 [± 0.4]

the highest likelihood in the *"Allow"* scenario. Similarly, for app features, participants provide low scores (mean=1.2). `Misleading` alerts elicit the highest (still modest, mean=2.1) likelihood for app features with *"Allow"* scenario.

To better understand if users' opinion of tracking changes from *"Not track"* to *"Allow"*, we measure the difference in likelihood between these two scenarios. We note that we conducted separate Wilcoxon tests instead of using a unified model as we found no correlation between (1) participants' belief towards the impact of extra features/rewards and (2) the likelihood score of allow-vs-not-track. For better reward/discount likelihood, a Wilcoxon signed-rank test reveals a significant difference (p = 0.0069 , CLES= 0.64) in likelihood for `Incentive` alerts. This shows that users are more likely to believe they will get better rewards/discounts if they change their decision from *"Not track"* to *"Allow"*.

Expectedly, participants exposed to `Misleading` patterns display a significant difference (p = 0.0065, CLES= 0.67) in the likelihood of receiving app features between *"Not track"* and *"Allow"*. These users were more likely to believe they would get better app features if they provided consent to tracking. Surprisingly, we also find a significant difference in the likelihood of better app feature between *"Allow"* and *"Not track"* for both `Other` (p = 0.001, CLES= 0.69) and `Incentive` (p = 0.02, CLES= 0.62) alerts. We note that for significant results, effect sizes (CLES) are moderate (~0.60).

We posit that, in the `Incentive`/`Other` alerts, confusion caused by the alert text may lead to misconceptions surrounding app features. For instance, one participant exposed to `Incentive` alerts believed *"allowing [app] to know your interests across other platforms [...] ties in to possibly [...] better app features."* Similarly, another participant exposed to an `Other` alert stated *"If you do not allow [tracking], then they have no reason to give you anything in return."* These findings suggest that, when the purpose of tracking is not mentioned (`Other` pattern) or when rewards are promised (`Incentive` pattern), participants may incorrectly infer better app features are given when consent to tracking is granted, although this is not explicitly mentioned.

`Finding-3:` Participants believe they will *more likely* receive rewards when alerts have `Incentive` patterns.

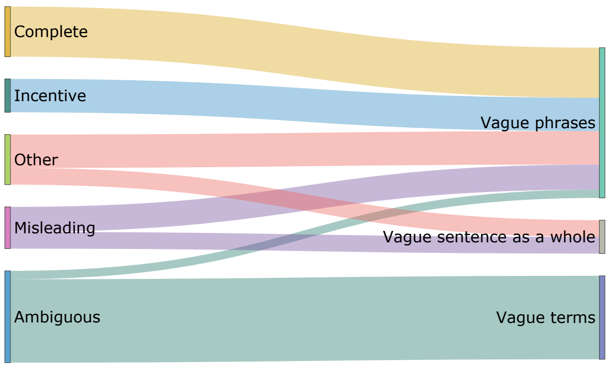


Figure 9: Participants reported confusion about the whole sentence, phrases within alerts, or specific terms.

`Finding-4:` Participants are *more likely* to believe they will receive better app features when alerts have `Misleading`, `Incentive`, or `Other` patterns in alerts.

5.4.4 SQ4: *Comprehension of Alerts and Vague Terms*

We asked participants to disclose if they found any portion of the alert vague or unclear. 28% of the participants respond to this optional open-text question to express confusion. We found that 47.8% of those who saw `Ambiguous` alerts note vagueness. Percentages of other alerts are 25% for `Misleading`, 27% for `Complete`, 17.4% for `Incentive`, and 23% for `Other`. After inductive coding, we discovered this confusion occurs at one of three levels, which we mapped to alert patterns as shown in Figure 9. We find an association between alert pattern and confusion type (p = 0.0008). Among those who noted vague alerts, 31.25% associated vagueness to specific words. Words noted surrounded advertising IDs (e.g., *"IDFA"*, *"Identifier"*). Unsurprisingly, all who noted vague terms were exposed to `Ambiguous` alerts.

For other alert patterns, we find participants expressed confusion in one of two ways. First, 56.25% expressed confusion due to vague phrases (e.g., *"Third party partners are vague as it could be anyone at all."*, *"what is partner information?"*). Second, 12.5% expressed confusion around the entire sentence (e.g., *"The whole sentence is vague"*).

In `Tracking-Related Questions`, we also explicitly asked participants about their knowledge of "IDFA" and "Identifiers" on a Likert scale of 1 to 5 (5 being very knowledgeable). We selected these terms after our analysis in Section 4 revealed them to be common in `Ambiguous` alerts. Participants expressed low familiarity with both "IDFA" (mean = 1.2, ±0.6) and "Identifiers" (mean = 1.68, ±1.1).

`Finding-5:` 28% of the participants find ATT alerts confusing, with the `Ambiguous` group noting vague alerts the most (47.8%). Participants are also largely unfamiliar with common terms in `Ambiguous` alerts. For other alert patterns (e.g., `Misleading`, `Incentive`), participants may be confused by vague phrases or the sentence as a whole.

6 Discussion and Limitations

6.1 Takeaways and Recommendations

Clarity in Guidelines. Instructions and recommendations for tracking, permission purpose strings, and pre-alerts are dispersed in multiple documents and webpages via Apple's Developers Guidelines. For instance, to view violations, developers would have to access the "Patterns" section of the "Human Interface Guidelines"; however, appropriate tracking uses are listed in the "App Privacy Details" section. This makes guideline adherence difficult, as developers do not have a single information source they can reference. To rectify this, we first recommend that Apple provide a single source of documentation for ATT permissions that addresses tracking and design guidelines (for alerts and pre-alerts).

Additionally, current guidelines only have (1) limited positive examples (appropriate ATT alert texts) and (2) no negative examples (alert texts that violate guidelines). Guidelines should be expanded to include **more detailed** positive examples beyond the default alert text provided, e.g., alert texts should specify what type of ad tracking is conducted if permission is granted. Similarly, guidelines should be periodically updated to include real-world negative examples that contain dark patterns, e.g., sample alert texts that include incentives.

Proactive Vetting of Alerts. The extent to which they are ATT guidelines are enforced remains in question. The prevalence of dark patterns violating Apple's guidelines suggests that, if enforcement exists, it cannot detect discovered dark patterns. Therefore, we strongly suggest proactive enforcement of guidelines through a strict vetting process.

We envision that ATTCLS can be integrated into this vetting process, given its reliable performance. Similarly, PREAEXTR can detect apps that include ATT purpose strings in the "`Info.plist`" file yet never display an ATT permission alert. Leveraging these automated tools can ensure that violating apps can be flagged before they are publicly available.

User-Centered Design of ATT Permissions. ATT purpose strings should effectively express the purpose of the prompt. Given possibility of multiple purposes (e.g., cross-platform tracking, sharing IDFA with 3rd party), further study is required to carefully design texts effective in informing users of each purpose. However, clear purpose strings may not be sufficient. As our study reveals, users do not necessarily trust developers to adhere to what is listed in the prompt. To resolve this, Apple could enforce a predetermined list of texts that developers can select from, and that can only be minimally modified. If the selected text aligns with how the app uses the IDFA, this should be communicated to the user to instill trust.

Similarly, enabling system-generated pre-alerts would improve user trust in ATT. This would incorporate a developer-defined pre-alert text but also display data types used for tracking in both *"Allow"* and *"Not track"* scenarios, to better inform the users of the permission implications.

6.2 Limitations

Pre-alert UI Violations. Apple guidelines include specific UI guidelines for the design of the pre-alert screens. These guidelines restrict developers from using visual cues in the pre-alert screens that can trick users into clicking *"Allow"* unintentionally. The guidelines label these practices as: (1) *Imitation Request*, (2) *Alert Image* and (3) *Alert Annotation*. (details and examples in Appendix C).

We manually analyzed the 273 apps that show a pre-alert screen and found that 22 apps violate UI guidelines. Specifically, 9 pre-alerts violate the *Imitation Request*, 9 violate *Alert Image* guideline, and 6 apps show pre-alerts with *Alert Annotation* violations. To automate the detection of UI guidelines violations, future work will explore computer vision techniques to identify visual cues that simulate permission alerts or annotations in the pre-alert screen.

Automated Alerts Classification. We introduce a multi-label classification framework to automatically classify the ATT alerts into five different patterns. We manually labeled the entire dataset of unique ATT alerts to test our model's effectiveness. This framework will help the research community to conduct further analysis on ATT alerts. It can also be integrated into the app-vetting process implemented by Apple's App Store for continuous monitoring of apps' guideline violations. However, to increase the effectiveness of this model, several methods can be integrated to increase its accuracy. We notice that the model's accuracy slightly decreases for `Other` label (78%), since this label does not contain a specific textual pattern as compared to the rest of the labels. Our model also is less accurate for the pre-alerts classification as compared to the purpose strings. This occurs because developers have the flexibility to explain the permission purpose with more words and phrases in pre-alert texts, resulting in patterns with lexical and semantic ambiguity. By collecting a larger dataset, the model can better learn to differentiate the `Other` label from the rest of the labels. Moreover, data augmentation techniques can be applied to increase the dataset size for pre-alerts, allowing training and testing a model on pre-alert texts. Finally, more advanced text filtering techniques can be added to filter the irrelevant words and phrases in the pre-alert text.

Evasion of Violations Detection. Our patterns detection model can be integrated into the vetting process of apps on Apple's store. However, developers can use different techniques to evade this tool. For instance, developers can adopt adversarial attacks for text data [3, 32]. These techniques use perturbed versions of the original text, which are not recognizable by humans but misclassified by NLP models (as BERT). Developers can also hide text in pre-alert screens using adversarial watermark attacks [20, 67], which are undetectable by human eyes while evading OCR detection techniques. To defend against these attacks, the language model can be trained using adversarial training methods [73] or modified to detect adversarial examples [12, 68, 74].

7 Related Work

Web and Mobile Tracking. Data tracking has long been explored in mobile/web contexts. Several works attempted to analyze apps' data usage purposes including tracking [17,18,41]. Others studied tracking data transparency and how it affects targeted ads [13,42,46]. A recent study [48] performed a dynamic network analysis of 1,759 iOS apps before/after ATT and showed that the number of tracking libraries has roughly stayed the same for both versions, where many apps still collect device IDs after ATT for cohort tracking or fingerprinting. Our work contrasts prior research in that we focus on developer non-conformance to tracking prompt (ATT) guidelines instead of determining data flows with ATT.

Studies on Privacy and Permissions. Early studies [14,28] evaluated users' understanding of installation-time and runtime permissions given the permission dialogs [4,15]. Shen et al. identified users' misunderstanding of runtime permission models and highlighted the additional information required to make informed decisions [64]. A comparison study [69] showed confusion about permissions reasoning from the perspectives of developers and users. Few studies were conducted on the developers' permission prompts. In [43], it was found that providing clear and real-time permission information assists users in making more informed decisions. In [70], a study was performed on the developer-specified purpose strings of iOS permissions, including location, contacts, photos/videos, calendars, reminders, and Bluetooth, showing that users are more likely to approve clear purpose strings. A similar analysis studied permission rationale messages in Android [51], showing that purposes stated in a majority of rationales are incorrect. Our work focuses on the ATT permission alerts, studying how dark patterns in alerts can affect user perception.

Several works have studied maliciously crafted dark patterns in UIs that deceive users/trick them into giving consent [16, 26, 34, 35, 49, 54]. Dark patterns in mobile applications and their impact on users have been studied in [26], showing that most apps have dark patterns that are not recognized by users. In [47], a large-scale analysis on mobile apps' privacy consent dialogues showed that many apps employ dark patterns to coerce the user to consent. Moreover, dark patterns were also acknowledged by the European Data Protection Board, which recently released guidelines in the GDPR [33] for regulating dark patterns on social media.

Recent studies investigated ATT permission in iOS and how it affects users. Hutton et al. [39] measured which apps users grant ATT permission and how it correlates with users' privacy concerns/motives. However, their work does not consider the permission alerts and their effects on users' perceptions. Another work studied the design patterns used for tracking permission in 200 popular iOS apps [25]. This work measures the effect of users' likelihood of tracking with different design patterns (e.g., permission purpose, framing, priming). In our work, however, we identify the ATT alert patterns based on Apple's guidelines and perform a large-scale analysis of apps to measure how they comply with the guidelines. We, hence, focus our user study on the perceptions of users and how their perceptions are affected by different alert types.

8 Conclusions

In this paper, we perform a large-scale analysis of the ATT permission alerts in iOS. We leverage static and dynamic analysis to extract ATT alert texts and pre-alerts. We then synthesize a labeling guide to identify ATT alert patterns and discover that over 59% of apps contain dark patterns used by app developers. We subsequently use advanced NLP techniques to detect dark patterns in alert texts automatically. Finally, via a user study, we discovered that regardless of alert patterns, users have misperceptions about ATT, and they often find alert texts confusing. Moreover, certain patterns (e.g., `Incentive`, `Misleading`) can influence the users' perception of receiving benefits (e.g., rewards, app features). Our findings motivate the need for improved developer guidelines, app vetting process, and designing user-centered ATT alerts.

Acknowledgments

We thank our shepherd and the anonymous reviewers for their comments and suggestions. This work has been partially supported by the National Science Foundation (NSF) under grant CNS-2144645. Any findings, conclusions, and recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the NSF.

References

[1] Advertising Identifier. `https://developer.apple.com/documentation/adsupport/asidentifiermanager/1614151-advertisingidentifier`, 2023. [Accessed: 2023-05-21].

[2] Lalit Agarwal, Nisheeth Shrivastava, Sharad Jaiswal, and Saurabh Panjwani. Do not embarrass: Re-examining user concerns for online tracking and advertising. In *Symposium on Usable Privacy and Security (SOUPS)*, 2013.

[3] Moustafa Alzantot, Yash Sharma Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. Generating Natural Language Adversarial Examples. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

[4] Panagiotis Andriotis, Martina Angela Sasse, and Gianluca Stringhini. Permissions snapshots: Assessing users' adaptation to the Android runtime permission model. In *IEEE International Workshop on Information Forensics and Security (WIFS)*, 2016.

[5] Annie I Antón, Julia B Earp, and Jessica D Young. How internet users' privacy concerns have evolved since 2002. In *IEEE Security & Privacy*, 2010.

[6] Limit ad targeting on iPhone iOS 13. `https://support.apple.com/guide/iphone/limit-ad-targeting-iphf60a6a256/13.0/ios/13.0`, 2021. [Accessed: 2023-05-21].

[7] Apple's data collection and storage guidelines. `https://developer.apple.com/app-store/review/guidelines/`, 2023. [Accessed: 2023-05-01].

[8] App Tracking Transparency. `https://developer.apple.com/documentation/apptrackingtransparency`. [Accessed: 2023-06-04].

[9] ATT Analysis Repository. https://github.com/purseclab/ATT_Analysis, 2021. [Accessed: 2023-10-09].

[10] Attributing ads. https://developer.apple.com/app-store/ad-attribution/, 2023. [Accessed: 2023-05-21].

[11] R. M. Van Auken and S. A. Kebschull. Type I error convergence of three hypothesis tests for small RxC contingency tables. In *RMS: Research in Mathematics & Statistics*, 2021.

[12] Rongzhou Bao, Jiayi Wang, and Hai Zhao. Defending Pre-trained Language Models from Adversarial Word Substitution Without Performance Sacrifice. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP*, 2021.

[13] Natã M Barbosa, Gang Wang, Blase Ur, and Yang Wang. Who Am I? A Design Probe Exploring Real-Time Transparency about Online and Offline User Profiling Underlying Targeted Ads. In *ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2021.

[14] Kevin Benton, L Jean Camp, and Vaibhav Garg. Studying the effectiveness of Android application permissions requests. In *IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*, 2013.

[15] Bram Bonné, Sai Teja Peddinti, Igor Bilogrevic, and Nina Taft. Exploring decision making with Android's runtime permission dialogs using in-context surveys. In *Symposium on Usable Privacy and Security (SOUPS)*, 2017.

[16] Christoph Bösch, Benjamin Erb, Frank Kargl, Henning Kopp, and Stefan Pfattheicher. Tales from the dark side: privacy dark strategies and privacy dark patterns. In *Privacy Enhancing Technologies (PoPETs)*, 2016.

[17] Duc Bui, Kang G Shin, Jong-Min Choi, and Junbum Shin. Automated Extraction and Presentation of Data Practices in Privacy Policies. In *Privacy Enhancing Technologies (PoPETs)*, 2021.

[18] Duc Bui, Yuan Yao, Kang G Shin, Jong-Min Choi, and Junbum Shin. Consistency analysis of data-usage purposes in mobile apps. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2021.

[19] CFGUTIL Commands Manual Page. https://configautomation.com/cfgutil-man-page.html, 2019. [Accessed: 2023-05-21].

[20] Lu Chen, Jiao Sun, and Wei Xu. FAWA: Fast adversarial watermark attack on optical character recognition (OCR) systems. In *Machine Learning and Knowledge Discovery in Databases*, 2021.

[21] Asaf Cidon, Lior Gavish, Itay Bleier, Nadia Korshun, Marco Schweighauser, and Alexey Tsitkin. High precision detection of business email compromise. In *USENIX Security Symposium*, 2019.

[22] Control is yours. https://www.apple.com/privacy/control/, 2023. [Accessed: 2023-05-01].

[23] Kovila PL Coopamootoo, Maryam Mehrnezhad, and Ehsan Toreini. " I feel invaded, annoyed, anxious and I may protect myself": Individuals' Feelings about Online Tracking and their Protective Behaviour across Gender and Country. In *USENIX Security Symposium*, 2022.

[24] Deceptive patterns. https://www.deceptive.design/types, 2023. [Accessed: 2023-05-21].

[25] Anzo DeGiulio, Hanoom Lee, and Eleanor Birrell. "Ask App Not to Track": The Effect of Opt-In Tracking Authorization on Mobile Privacy. In *International Workshop on Emerging Technologies for Authorization and Authentication*. Springer, 2022.

[26] Linda Di Geronimo, Larissa Braz, Enrico Fregnan, Fabio Palomba, and Alberto Bacchelli. UI dark patterns and where to find them: a study on mobile applications and user perception. In *ACM Conference on Human Factors in Computing Systems (CHI)*, 2020.

[27] Doccano. https://doccano.herokuapp.com/. [Accessed: 2023-05-01].

[28] Adrienne Porter Felt, Elizabeth Ha, Serge Egelman, Ariel Haney, Erika Chin, and David Wagner. Android permissions: User attention, comprehension, and behavior. In *Symposium on Usable Privacy and Security (SOUPS)*, 2012.

[29] Rudolf J Freund and William J Wilson. *Statistical methods*. Elsevier, 2003.

[30] Alisa Frik, Juliann Kim, Joshua Rafael Sanchez, and Joanne Ma. Users' expectations about and use of smartphone privacy and security settings. In *ACM Conference on Human Factors in Computing Systems (CHI)*, 2022.

[31] Tianyu Gao, Xingcheng Yao, and Danqi Chen. SimCSE: Simple Contrastive Learning of Sentence Embeddings. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2021.

[32] Siddhant Garg and Goutham Ramakrishnan. BAE: BERT-based Adversarial Examples for Text Classification. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

[33] Guidelines on Dark patterns in social media platform interfaces: How to recognise and avoid them. https://edpb.europa.eu/system/files/2022-03/edpb_03-2022_guidelines_on_dark_patterns_in_social_media_platform_interfaces_en.pdf. [Accessed: 2023-09-26].

[34] Colin M Gray, Yubo Kou, Bryan Battles, Joseph Hoggatt, and Austin L Toombs. The dark (patterns) side of ux design. In *ACM Conference on Human Factors in Computing Systems (CHI)*, 2018.

[35] Johanna Gunawan, Amogh Pradeep, David Choffnes, Woodrow Hartzog, and Christo Wilson. A Comparative Study of Dark Patterns Across Web and Mobile Modalities. In *ACM on Human-Computer Interaction*, 2021.

[36] Hamza Harkous, Kassem Fawaz, Rémi Lebret, Florian Schaub, Kang G Shin, and Karl Aberer. Polisis: Automated analysis and presentation of privacy policies using deep learning. In *USENIX Security Symposium*, 2018.

[37] Louis M Hsu and Ronald Field. Interrater agreement measures: Comments on Kappan, Cohen's Kappa, Scott's π, and Aickin's α. *Understanding Statistics*, 2003.

[38] Human interface guidelines. https://developer.apple.com/design/human-interface-guidelines/patterns/accessing-private-data/, 2023. [Accessed: 2023-05-01].

[39] Hannah J Hutton and David A Ellis. Exploring User Motivations Behind iOS App Tracking Transparency Decisions. In *ACM Conference on Human Factors in Computing Systems (CHI)*, 2023.

[40] IPA tool. https://github.com/majd/ipatool. [Accessed: 2023-04-27].

[41] Haojian Jin, Minyi Liu, Kevan Dodhia, Yuanchun Li, Gaurav Srivastava, Matthew Fredrikson, Yuvraj Agarwal, and Jason I Hong. Why are they collecting my data? inferring the purposes of network traffic in mobile apps. In *ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2018.

[42] Yucheng Jin, Karsten Seipp, Erik Duval, and Katrien Verbert. Go with the flow: effects of transparency and user control on targeted advertising using flow charts. In *International Working Conference on Advanced Visual Interfaces*, 2016.

[43] Patrick Gage Kelley, Lorrie Faith Cranor, and Norman Sadeh. Privacy as part of the app decision-making process. In *Proceedings of the SIGCHI conference on human factors in computing systems*, 2013.

[44] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2019.

[45] Rishabh Khandelwal, Thomas Linden, Hamza Harkous, and Kassem Fawaz. Prisec: A privacy settings enforcement controller. In *USENIX Security Symposium*, 2021.

[46] Tami Kim, Kate Barasz, and Leslie K John. Why am I seeing this ad? The effect of ad transparency on ad effectiveness. In *Journal of Consumer Research*, 2019.

[47] Simon Koch, Benjamin Altpeter, and Martin Johns. The OK Is Not Enough: A Large Scale Study of Consent Dialogs in Smartphone Applications. In *USENIX Security Symposium*, 2023.

[48] Konrad Kollnig, Anastasia Shuba, Max Van Kleek, Reuben Binns, and Nigel Shadbolt. Goodbye tracking? Impact of iOS app tracking transparency and privacy labels. In *ACM Conference on Fairness, Accountability, and Transparency*, 2022.

[49] Monica Kowalczyk, Johanna T Gunawan, David Choffnes, Daniel J Dubois, Woodrow Hartzog, and Christo Wilson. Understanding Dark Patterns in Home IoT Devices. In *ACM Conference on Human Factors in Computing Systems (CHI)*, 2023.

[50] Klaus Krippendorff. Reliability in content analysis: Some common misconceptions and recommendations. In *Human communication research*, 2004.

[51] Xueqing Liu, Yue Leng, Wei Yang, Wenyu Wang, Chengxiang Zhai, and Tao Xie. A large-scale empirical study on android runtime-permission rationale messages. In *IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, 2018.

[52] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[53] Marvel app. `https://marvelapp.com`. [Accessed: 2023-04-15].

[54] Arunesh Mathur, Gunes Acar, Michael J Friedman, Eli Lucherini, Jonathan Mayer, Marshini Chetty, and Arvind Narayanan. Dark patterns at scale: Findings from a crawl of 11k shopping websites. In *ACM on Human-Computer Interaction*, 2019.

[55] Aleecia McDonald and Lorrie Faith Cranor. Beliefs and behaviors: Internet users' understanding of behavioral advertising. In *TPRC*, 2010.

[56] Aleecia M McDonald and Lorrie Faith Cranor. Americans' attitudes about internet behavioral advertising practices. In *ACM Workshop on Privacy in the Electronic Society (WPES)*, 2010.

[57] Kenneth O McGraw and Seok P Wong. A common language effect size statistic. *Psychological bulletin*, 1992.

[58] Prolific. `https://www.prolific.co`. [Accessed: 2023-05-26].

[59] Python Tesseract Library. `https://pypi.org/project/pytesseract/`. [Accessed: 2023-05-21].

[60] Qualtrics. `https://www.qualtrics.com`. [Accessed: 2023-05-26].

[61] Kopo M Ramokapane, Anthony C Mazeli, and Awais Rashid. Skip, skip, skip, accept!!!: A study on the usability of smartphone manufacturer provided default features and user privacy. In *Privacy Enhancing Technologies (PoPETs)*, 2019.

[62] Robots.txt for Apple. `https://www.apple.com/robots.txt`. [Accessed: 2023-06-04].

[63] Philip Sedgwick. Multiple significance tests: the Bonferroni correction. *Bmj*, 2012.

[64] Bingyu Shen, Lili Wei, Chengcheng Xiang, Yudong Wu, Mingyao Shen, Yuanyuan Zhou, and Xinxin Jin. Can Systems Explain Permissions Better? Understanding Users' Misperceptions under Smartphone Runtime Permission Model. In *USENIX Security Symposium*, 2021.

[65] Simple Transformers Library. `https://simpletransformers.ai/`. [Accessed: 2023-06-04].

[66] Sooel Son, Daehyeok Kim, and Vitaly Shmatikov. What Mobile Ads Know About Mobile Users. In *Network and Distributed System Security Symposium (NDSS)*, 2016.

[67] Congzheng Song and Vitaly Shmatikov. Fooling OCR systems with adversarial text images. *arXiv preprint arXiv:1802.05385*, 2018.

[68] Abigail Swenor. Using Random Perturbations to Mitigate Adversarial Attacks on NLP Models. In *AAAI Conference on Artificial Intelligence*, 2022.

[69] Mohammad Tahaei, Ruba Abu-Salma, and Awais Rashid. Stuck in the Permissions With You: Developer & End-User Perspectives on App Permissions & Their Privacy Ramifications. In *ACM Conference on Human Factors in Computing Systems (CHI)*, 2023.

[70] Joshua Tan, Khanh Nguyen, Michael Theodorides, Heidi Negrón-Arroyo, Christopher Thompson, Serge Egelman, and David Wagner. The effect of developer-specified explanations for permission requests on smartphone user behavior. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2014.

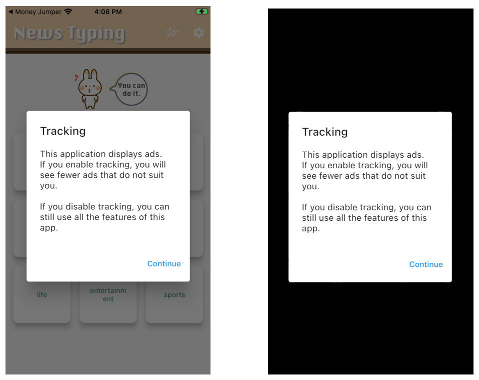
[71] The digital world is built on advertising. `https://www.cnn.com/2021/05/18/health/kids-digital-advertising-wellness/index.html`, 2021. [Accessed: 2023-05-21].

[72] Blase Ur, Pedro Giovanni Leon, Lorrie Faith Cranor, Richard Shay, and Yang Wang. Smart, useful, scary, creepy: perceptions of online behavioral advertising. In *Symposium on Usable Privacy and Security (SOUPS)*, 2012.

[73] Xiaosen Wang, Yichen Yang, Yihe Deng, and Kun He. Adversarial training with fast gradient projection method against synonym substitution-based text attacks. In *AAAI Conference on Artificial Intelligence*, 2021.

[74] Xiaosen Wang, Xiong Yifeng, and Kun He. Detecting textual adversarial examples through randomized substitution and vote. In *Uncertainty in Artificial Intelligence*, 2022.

[75] A system-wide touch event simulation library for iOS 11-14. `https://github.com/xuan32546/IOS13-SimulateTouch`, 2020. [Accessed: 2023-05-01].



Tracking

This application displays ads. If you enable tracking, you will see fewer ads that do not suit you.

If you disable tracking, you can still use all the features of this app.

Tracking

This application displays ads. If you enable tracking, you will see fewer ads that do not suit you.

If you disable tracking, you can still use all the features of this app.

(a) Original pre-alert. (b) Cropped pre-alert.

Figure 1: An example of (a) pre-alert screen and (b) after cropping the pre-alert popup.

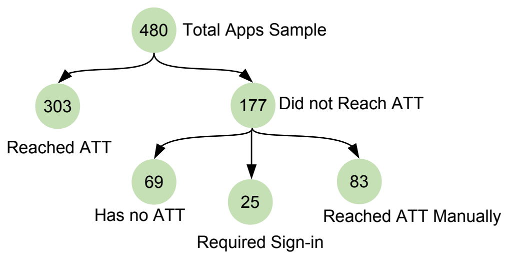
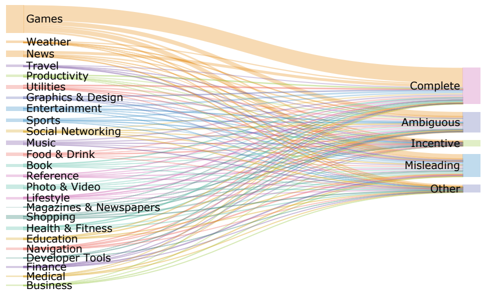


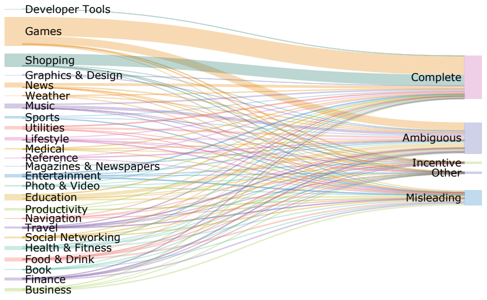
Figure 2: Evaluation of the dynamic data collection tool.

Table 1: Detailed distribution of apps labels for purpose strings and pre-alerts.

	Purpose Strings					Pre-alerts					
	Incentive	Misleading	Ambiguous	Complete	Other	Incentive	Misleading	Ambiguous	Complete	Other	Apps Frequency
No Pre-alerts	✓	—	—	—	—	—	—	—	—	—	150
	—	✓	—	—	—	—	—	—	—	—	753
	—	—	✓	—	—	—	—	—	—	—	908
	✓	✓	—	—	—	—	—	—	—	—	57
	✓	—	✓	—	—	—	—	—	—	—	9
	—	✓	✓	—	—	—	—	—	—	—	90
	✓	✓	✓	—	—	—	—	—	—	—	3
	—	—	—	✓	—	—	—	—	—	—	1534
	—	—	—	—	✓	—	—	—	—	—	222
Has Pre-alerts	—	—	✓	—	—	—	—	✓	—	—	3
	—	✓	—	—	—	—	—	✓	—	—	2
	—	✓	—	—	—	—	✓	✓	—	—	2
	—	—	—	✓	—	—	✓	✓	—	—	1
	—	—	—	—	✓	—	✓	✓	—	—	1
	✓	—	—	—	—	✓	—	—	—	—	9
	—	✓	—	—	—	✓	—	—	—	—	3
	—	—	✓	—	—	✓	—	—	—	—	41
	✓	✓	—	—	—	✓	—	—	—	—	4
	—	✓	✓	—	—	✓	—	—	—	—	1
	—	—	—	✓	—	✓	—	—	—	—	31
	—	—	—	—	✓	✓	—	—	—	—	7
	✓	—	—	—	—	✓	✓	—	—	—	1
	—	✓	—	—	—	✓	✓	—	—	—	4
	—	—	✓	—	—	✓	✓	—	—	—	10
	✓	✓	—	—	—	✓	✓	—	—	—	3
	—	—	—	✓	—	✓	✓	—	—	—	4
	✓	—	—	—	—	—	✓	—	—	—	4
	—	✓	—	—	—	—	✓	—	—	—	39
	—	—	✓	—	—	—	✓	—	—	—	5
	✓	✓	—	—	—	—	✓	—	—	—	1
	—	—	—	✓	—	—	✓	—	—	—	25
	—	—	—	—	✓	—	✓	—	—	—	2
	✓	—	—	—	—	—	—	—	✓	—	4
	—	✓	—	—	—	—	—	—	✓	—	6
	—	—	✓	—	—	—	—	—	✓	—	16
	—	—	—	✓	—	—	—	—	✓	—	36
	—	✓	—	—	—	—	—	—	—	✓	1
	—	—	✓	—	—	—	—	—	—	✓	4
	✓	✓	—	—	—	—	—	—	—	✓	1
	—	—	—	—	✓	—	—	—	—	✓	2



(a) Popular apps



(b) Random apps

Figure 3: Distribution of app genres among pattern labels.

A Apps Analysis

Figure 1 shows an example of a pre-alert screen. We crop the pre-alert by detecting the image contours and masking the image using the contour with maximum area. We extract the text from the cropped image. The detailed distribution of app labels for purpose strings and pre-alerts is shown in Table 1.

Figure 3 shows the distribution of patterns among different app genres for the popular and random app sets. We found no correlation between the app categories and pattern labels.

B User study Details

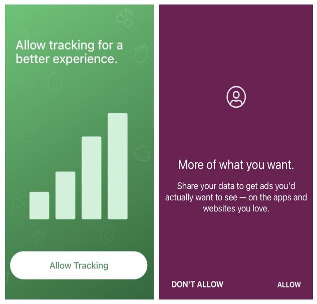
B.1 Pilot Studies

We test our survey design by conducting two rounds of pilot studies. First, we posted a call for participants on a university campus; we conducted the study with 7 volunteers and stepped through the survey to incorporate feedback (e.g., revise unclarities, remove leading questions). We clarified the wording of some questions pointed out as unclear.

In the second round, we recruited 17 participants on Prolific [58] to test the participants' response quality and their level of engagement with the survey. We found that participants would not effectively engage in the open-text questions when asked to justify their selections. To encourage engagement, we added *"In two or more sentences,"* as a prefix to these questions. Additionally, we found that only displaying the alert once before questions on the alert limited the participants' ability to recall the alert text. To correct this issue, we split the survey's `Alert-Questions` into multiple pages and displayed the alert text at the top of each page.

B.2 Survey Demographics

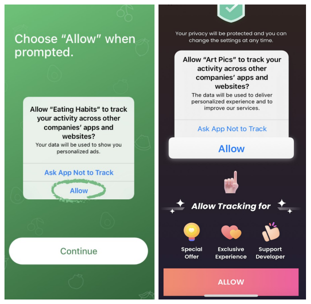
Table 3 presents the demographics of our participants.


Allow tracking for a better experience.

More of what you want.

Share your data to get ads you'd actually want to see — on the apps and websites you love.

(a) Imitation Request.

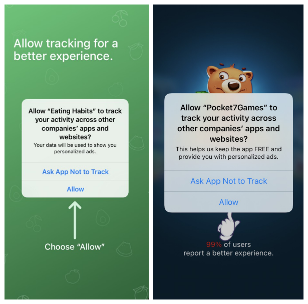

Choose “Allow” when prompted.

Allow “Eating Habits” to track your activity across other companies’ apps and websites?
Your data will be used to show you personalized ads.

Your privacy will be protected and you can change the settings at any time.

Allow “Art Pics” to track your activity across other companies’ apps and websites?
The data will be used to deliver personalized experience and to improve our services.

(b) Alert Image.


Allow tracking for a better experience.

Allow “Eating Habits” to track your activity across other companies’ apps and websites?
Your data will be used to show you personalized ads.

Allow “Pocket7Games” to track your activity across other companies’ apps and websites?
This helps us keep the app FREE and provide you with personalized ads.

99% of users report a better experience.

(c) Alert Annotation.

Figure 4: Apple’s UI guidelines for pre-alerts and examples of discovered apps with violations.

Table 2: ATT alerts displayed in the survey.

ATT Alerts†	Alert Label	N
Partner information is preferred to serve you with relevant promotions.	Incentive	6
This keeps your App free and reduces irrelevant Ads.	Incentive	5
This allows [*app*] to determine if you are eligible to receive a reward after creating your account.	Incentive	5
Discover products and offers tailored to your interests on [*app*] and partner sites.	Incentive	7
This data is anonymized, no personal data, only how you play to deliver the best gaming experience and to show less but relevant ads.	Misleading	4
This allows us to deliver personalized content to you and improves your ordering experience.	Misleading	6
This helps us provide you with a more personalized experience.	Misleading	5
This will allow to deliver better and fewer personalized ads to you.	Misleading	5
Your device’s advertising ID will be used to provide you with an improved personalized ad experience over our standard ad experience.	Ambiguous	6
This identifier is used to deliver personalized ads to you.	Ambiguous	7
We use IDFA for promoting our service on advertising platforms. We don’t pass your personal data on to someone else.	Ambiguous	3
[*app*] uses your IDFA for attributing app installs to advertisements.	Ambiguous	7
Hate random ads? Allow [*app*] to track activity to deliver personalized ads for you.	Complete	4
[*app*] and our third party partners will use your data to deliver personalized ads.	Complete	6
Your data will only be used to deliver personalized ads to you.	Complete	8
Your data will be used to provide you a better and personalized ads to you.	Complete	4
Your data allows for the optimization of the in-app ad experience for you.	Other	5
To ensure the best possible ad experience.	Other	7
This allows [*app*] to provide you with a better ads experience.	Other	7
[*app*] will track you across other applications.	Other	7

† We redact app names in this table. App names are available on request.

Table 3: Survey participants demographics.

Demographic Data	N	%
Age Group		
18-24	27	23.68%
25-34	39	34.21%
35-44	20	17.54%
45 and above	28	24.56%
Race and Ethnicity		
White	83	72.81%
Asian or Asian American	11	9.65%
Black or African American	8	7.02%
American Indian or Alaska Native	2	1.75%
Other	7	6.14%
Prefer not to specify	3	2.63%
Gender		
Female	57	50.0%
Male	53	46.49%
Prefer not to specify	4	3.5%
Annual Income		
less than $10k	22	19.3%
$10-$20k	7	6.14%
$20-$30k	12	10.53%
$30-$50k	21	18.42%
More than $50k	47	41.23%
Prefer not to specify	5	4.39%
Technical Experience		
Computer science	9	7.9%
Software engineering	2	1.76%
App development	2	1.76%
Other technical fields	11	9.65%
None	90	78.93%

B.3 Survey Alerts

Table 2 displays the alert shown to different participants. We implemented prototypes for 20 apps using Marvel app [53], with four apps per alert pattern.

C Apple’s ATT UI Guidelines

Apple’s ATT UI guidelines define three practices to guide developers design of pre-alert screens: (1) *Imitation Request*, (2) *Alert Image* and (3) *Alert Annotation*.

Imitation Request states that developers should not display a custom screen that mirrors the functionality of the OS-generated ATT permission alert. *Alert Image* guideline states that pre-alert should not show an image of the OS-provided permission alert and modify it in any way, such as by circling or highlighting the *“Allow”* button. *Alert Annotation* guideline states that developers should not draw a visual cue on the actual permission alert that attracts users’ attention to the permission alert’s *“Allow”* button.

Figure 4 shows examples of the violation of Apple’s guidelines for the UI design of ATT pre-alert screens.