Understanding Mobile App Reviews to Guide Misuse Audits

Vaibhav Garg*, Hui Guo†, Nirav Ajmeri‡, Saikath Bhattacharya§, Munindar P. Singh¶
*Virginia Tech, Alexandria, USA
Email: vaibhavg@vt.edu
†Quora Inc., Mountain View, California, USA
Email: ncsuguo@gmail.com
‡University of Bristol, Bristol, UK
Email: nirav.ajmeri@bristol.ac.uk
§Milwaukee School of Engineering, Milwaukee, Wisconsin, USA
Email: bhattacharya@msoe.edu
¶North Carolina State University, Raleigh, North Carolina, USA
Email: mpsingh@ncsu.edu

***Abstract*—Problem: We address the challenge in responsible computing where an *exploitable* mobile app is misused by one app user (an *abuser*) against another user or bystander (*victim*). We introduce the idea of a *misuse audit* of apps as a way of determining if they are exploitable without access to their implementation. Method: We leverage app reviews to identify exploitable apps and their functionalities that enable misuse. First, we build a computational model to identify alarming reviews (which report misuse). Second, using the model, we identify exploitable apps and their functionalities. Third, we validate them through manual inspection of reviews. Findings: Stories by abusers and victims mostly focus on past misuses, whereas stories by third parties mostly identify stories indicating the potential for misuse. Surprisingly, positive reviews by abusers, which exhibit language with high dominance, also reveal misuses. In total, we confirmed 156 exploitable apps facilitating the misuse. Based on our qualitative analysis, we found exploitable apps exhibiting four types of exploitable functionalities. Implications: Our method can help identify exploitable apps and their functionalities, facilitating misuse audits of a large pool of apps.**

I. Introduction

Traditional audits of mobile apps conduct a review of their source code [14, 52]. However, interpersonal misuse arising from app users (instead of app developers) goes unnoticed by such processes. We introduce *misuse audit*, a process for auditing mobile apps through the reported misuse cases.

A misuse occurs when an app user (*abuser*) exploits the app to access information of other users or third parties (*victims*). In particular, a misuse audit can identify cases when the victim (1) doesn't know about the information access (spying) or (2) may know about the access but is uncomfortable with it. The latter case (often missed by traditional audits) includes incidents of forced consent or when public information (such as on dating apps) is accessed beyond the victim's level of comfort. We use the term *exploitable behavior* to denote these two types of information access and use *exploitable apps* to refer to apps that enable exploitable behavior.

Research [6, 13, 19] shows that exploitable apps may cause discomfort, fear, and potential harm to the victim. Possible ways to prevent this risk include highlighting these apps and their exploitable functionalities to users, app distribution platforms, and app developers. However, identifying these apps is nontrivial since they have a legitimate purpose but are misused by abusers. We propose an approach, MissAuditor, for misuse audits that identify exploitable apps and uncover their exploitable functionalities.

We find that the reviews of an app often describe exploitable functionalities, its misuse (potential and actual), and users' expectations. Such reviews are evidence of exploitable behavior and can guide audits of such apps. Moreover, app reviews are more valuable than app metadata (such as app descriptions) because metadata indicates only their legitimate purpose and not the actual misuse. In particular, we propose the following research questions:

RQ_{info}. What information regarding misuse audit is contained in reviews?

$RQ_{identify}$. How can we conduct a misuse audit through app reviews?

$RQ_{functionality}$. What exploitable functionalities are present in audited apps?

Example 1 shows three reviews (edited for grammar), taken from Apple's App Store [37] that are relevant to exploitable behaviors. Although our study is based on Apple App Store's reviews, MissAuditor can be applied to reviews from other sources, including Google's Play Store [47].

In Example 1, the first review, for AirBeam Video [36], addresses the scenario where the app assists a user in accessing a victim's information without the victim's knowledge. AirBeam Video is a surveillance app to be installed on the abuser's device. Hence, the victim may not be an app user but a bystander. The second review, for Life360 [41], complains about the problem of inappropriate access to the user's location by the user's mother. Due to the unequal power dynamics between the victim (reviewer in this case) and the abuser (mother in this case), the victim is forced to install apps that violate

Example 1: Cases relevant to exploitable behavior

Fly on the wall!
(for the AirBeam Video Surveillance app [36])
"with this app, i can spy on my family without them knowing it! it's such an awesome app!"

This app basically ruined my family to an extent
(for the Life360 app [41])
"My mother made everyone in the family get this app. She freaks out when the app doesn't do its job because of random obstacles that mess with the location accuracy. Drains the battery and makes my parents paranoid to know where I am at all times. I don't even do any bad stuff, yet years of trust building are being swept away by the ability to spy on the children of a household. If you're a parent I highly recommend you don't get this app because it is extremely uncomfortable to have and it makes parents trust their children less."

Honest
(for the 3Fun: Threesome & Swingers app [45])
"...A lot of the local people I've talked to (Male half of a couple) have been guys who are saying they're part of a couple, and in all reality are single guys just looking to collect pictures. There is no way to report that that is why you are reporting them. It's just a boilerplate report feature. I feel there should be a way for the 3Fun community to point out people for bad behavior like this."

their privacy. The third review, from 3Fun [45], describes a story of improper access to profile pictures. Even though the profile pictures are public, the victim is uncomfortable with the access. It is common for users to upload such information (pictures in this case) on an app with expectations of how other users would access it. As shown in these three cases, information access may lead to discomfort, fear, or potential harm [13, 19]. Thus, such cases of information access indicate misuse.

A. Key Findings

We find that reviews contain rich information about apps, which is crucial in auditing them for their misuse potential. For example, not only negative but positive reviews also reveal an app's exploitable functionalities. Relevant reviews are written by abusers, victims, and third person, each type linguistically distinct, which makes mining reviews challenging (Section II). Moreover, while reporting exploitability, reviews have varying degrees of *convincingness* and *severity*, which can be leveraged in misuse audit (Section III). We found that exploitable apps exhibit a variety of exploitable functionalities, ranging from tracking location to monitoring phone activities such as accessing chats, phone contacts, call history, and so on (Section IV).

B. Contributions and Novelty

Our work's novelty lies in introducing the problem of misuse audits. We leverage app reviews, which are a large unexplored resource for misuse audits. We contribute to responsible computing in the following manner.

First, we show that app reviews are a viable source of information on its exploitable functionalities. Second, MISSAUDITOR, an approach based on app reviews for MISUSE AUDITS of apps. Third, a list of exploitable apps along with their alarming reviews revealing exploitable functionalities.

C. Organization

The rest of this paper is organized as follows. Section II describes our preliminary investigation that shows that app reviews contain evidence useful for audits. Section III describes how to identify exploitable apps using our misuse audit approach, MISSAUDITOR. Section IV shows the procedure to uncover exploitable functionalities of audited apps. Section V lists related work on app reviews and mobile apps. Section VI concludes the paper and Section VII discusses the reproducibility of our findings.

II. RQ_{INFO}: App Reviews for Misuse Audits

We describe how we collected and investigated reviews for misuse audit, which led to several qualitative findings.

A. Seed Dataset

Chatterjee et al. [6] identified 2,707 iOS apps as candidates for Intimate Partner Surveillance (IPS), i.e., apps used by a person to spy on their intimate partner. IPS apps are specific to intimate partners and, hence, a subclass of our concept of exploitable apps. We started our analysis from Chatterjee et al.'s IPS candidate list.

During our data collection, 1,687 of these 2,707 apps received at least one review on the Apple App Store, leading to a *seed dataset* containing 11.57 million reviews. Out of 1,687 apps in the seed dataset, only 210 were confirmed IPS by Chatterjee et al..

B. Investigating Reviews

Since the seed dataset contains 11.57 million reviews, it is impractical to manually check each review. To find the reviews revealing exploitable behavior, we sampled them using three keywords (*spy*, *stalk*, and *stealth*). From the 5,287 reviews that contain at least one of our keywords, we randomly sampled 1,000 reviews for manual scrutiny. We obtained a simple random sample to investigate if arbitrary reviews reveal any misuse cases. Moreover, we analyzed what information regarding misuse they contain. This sample is diverse: it involves 179 apps with between 1 and 237 reviews each.

Out of 1,000 reviews, we found 403 reviews reporting exploitable behavior and 597 others. We found that reviews reporting exploitable behavior show variation across two dimensions: *story* and *reviewer*. Based on the story, we found reviews of the following two types:

***exploitable act*:** Reviews describe someone performing an exploitable behavior. In such reviews, the reviewer is sure about the app's exploitable functionality. For example, *"This app is useless and it just helps overprotected parents spy on their sad kids"*

***potential*:** Reviews express the possibility of exploitable behavior. The reviewer may not be sure of the exploitable functionality but identifies risks with the app that can be exploitable in the future. For example, *"...Hate to be a hater. May work well to spy on the kids by 'accidentally' leaving iPhone in secret place."*

The functionalities actually or could be misused by abusers (in exploitable act or potential) are called *exploitable functionalities*. The above two reviews are negative. However, we also found positive reviews indicating an exploitable app. Example 2 shows positive reviews by *abusers*, who brag about the exploitable functionalities (history tracking in this case) and sometimes express their delight in the misuse. Other reviews are written by *victims*, who state their concerns and grievances (frustration at the loss of privacy), and some by *third persons* reporting the misuse of the app.

Example 2: Types of reviewers reporting exploitable behavior

Abuser

"So much better than ¡other app¿*. I love the history feature. My kids say it's creepy and I'm being a stalker. I don't disagree but it is really nice and convenient to be able to keep track of my kids."*

Victim

"I hate this app so much! My mother is always questioning me and if I delete it she will ground me ...No one wants their parents to stalk them!!"

Third Person

"...I don't feel like parents should track their kids AT ALL. everyone needs a little something called trust and if you don't have it then your kids will act out and have to become sneaky ...I do have this app but only with my friends and we don't stalk each other ..."

Our linguistic analysis reveals that reviews by abusers not only are positive (higher valence than the other two categories) but also illustrate the abuser's dominance over other parties. Figure 1 shows this difference in the valence and dominance scores between all three reviewers. In general, these relevant reviews show linguistic variations, complicating the task.

Figure 2 shows the relative distribution of 403 relevant reviews across the type of story and reviewer. The third person mostly writes potential stories (73%), whereas abusers and victims mainly describe exploitable acts (∼99% and 100%, respectively).

The above examples indicate that reviews contain rich information about an app's exploitable behavior. We found two types of stories that are relevant for our purposes. Moreover, such stories are written by three types of reviewers, whose language varies, making the problem of mining these reviews

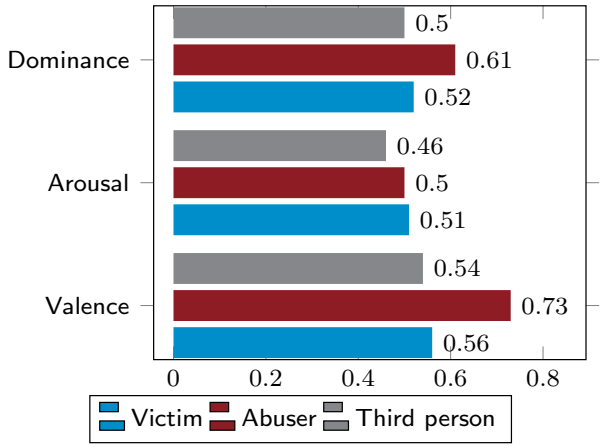


Fig. 1. Affective analysis showing valence, arousal, and dominance scores for each type of reviewer. Abusers' language shows higher dominance and valence than the language used by victims and third persons. Mining app reviews is challenging due to such linguistic variations.

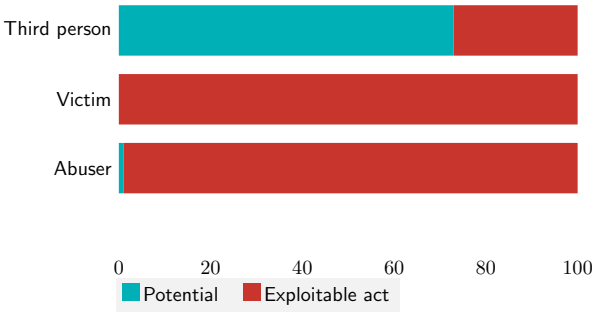


Fig. 2. Out of 403 stories manually identified as relevant, abusers and victims mostly write stories indicating exploitable acts (∼99% and 100% respectively), whereas third persons write stories indicating potential cases (∼73%).

nontrivial. No new types of stories and reviewers were found during our extensive verification of reviews (discussed in Section VI). This indicates that our random sample (1,000 reviews used for this analysis) was a representative set.

III. RQ$_{\text{IDENTIFY}}$: Identifying Exploitable Apps through Misuse Audit

We propose MissAuditor, a review-based approach for misuse audits. Figure 3 shows an overview of the MissAuditor approach. First, we collect reviews from the Apple App Store, as shown for the seed dataset in Section II-A. Second, we label a subset of the collected reviews for *alarmingness* (defined below) and train a computational model (Section III-A). Third, we apply our model to all reviews to identify exploitable apps (Section III-B). Fourth, we manually examine reviews of some of the identified exploitable apps to find their exploitable functionalities (Section IV).

We envision MissAuditor to be incrementally updated by adding reviews of newly found exploitable apps. Moreover, apps with no reviews can be audited as soon as their reviews arrive.

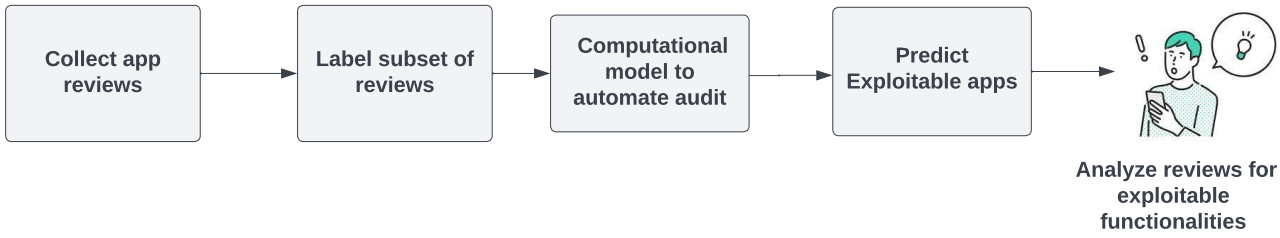


Fig. 3. Overview of MISSAUDITOR.

A. Alarmingness of Reviews

We find that reviews exhibit two characteristics: (i) *convincingness* their claims about misuse and (ii) *severity* or the effect of the misuse. We describe each of these characteristics below.

Convincingness depends on the review's claim about the app's exploitability. Some reviews report detailed exploitable behavior and, therefore, are convincing, whereas others are merely suspicions. In Example 3, the first review is unrelated to exploitable behavior and, hence, is not convincing. The second review describes the reviewer's suspicion of the app, which may or may not be true (slightly convincing). The third review (by an abuser) confirms the exploitable behavior but lacks details of the exploitable functionalities or victims. In contrast, the other reviews in Example 3 are convincing because they confirm exploitable behavior and mention the location feature, how to set up devices, or the victims being stalked. Extremely convincing reviews also include cases when the app is used for positive purposes, such as tracking family members or pets for safety, but shows the potential to be misused in the future. The reviews that are slightly, moderately, or extremely convincing are relevant to misuse audits. Assigning a convincingness score helps rank all reviews according to the strength of their claims.

Severity measures the effect of exploitable behavior on the victim. Example 4 shows the range of reviews varying in severity. The first review is unrelated to exploitable behavior. Thus, it is not severe. The second review shows that the exploitable act is performed with consent, making this review a slightly severe case. The third review is written by the abuser and lacks the victim's perspective to analyze the exploitable effect. We assume such acts are performed without consent and consider them moderately severe. The fourth review describes the victim's misery. The victim even says, "This app has truthfully ruined my teenage years" in the review, which gives solid evidence to be an extremely severe case. Moreover, in the fifth review, the victim complains that others can see when he was last active (also known as last-seen information). This is public information on each profile, but still, the victim is uncomfortable with the access. App developers should be aware of such misuse.

To capture the above two characteristics, we define the *alarmingness* of a review as the geometric mean of its convincingness and severity. An app can receive a large number of reviews. Unlike binary classification of reviews, the alarmingness score not only identifies relevant reviews for audit but also ranks them based on the likelihood and danger of the misuse. For an app, the higher the alarmingness of the review, the more useful it is in auditing and identifying misuse.

Example 3: Varying degree of convincingness

1: Not convincing
"It is such a great game, love it so much!"

2: Slightly convincing
"Setup was a breeze. Quicktime 7 pro found it easily. Unfortunately, resolution seems much, much, lower than hoped. Video size can not be adjusted live. Hate to be a hater. May work well to spy on the kids by 'accidentally' leaving iPhone in secret place."

3: Moderately convincing
"This app is perfect for stalking people. . . "

4: Extremely convincing
"This app is awesome for our family to keep track of where everyone is at all times! (You can turn the location off too in case you want to be in stealth mode when buying Christmas presents too.) . . . Even our dog knows that the alert sound when a family member arrives home means . . . "

". . . I use it to spy on my dogs while I'm at work; so I use it for fun, nothing fancy. My iPad is my camera, and my iPhone is my viewer. . . . "

"bro this app is high key creepy. when i'm with my dad on his days my mom even mentions how she knew everything i was doing and it even made my dad creeped out. if you need this app then ngl yo wack. i don't want my mom stalking me."

We created a training set of reviews as follows. From the seed dataset, we randomly selected two sets of about 1,000 reviews each: those that match at least one of our keywords and those that do not. After removing duplicate reviews, we were left with 952 and 932 reviews, respectively, combined into our training corpus of 1,884 reviews. Including both matching and nonmatching reviews in about equal numbers makes our training corpus unbiased toward the keywords.

Example 4: Varying degree of severity

1: Not severe
"Love the graphics so far it is a great game"

2: Slightly severe
"I love this app, just great because you can time your day accordingly, I like my girlfriend knowing where I am and I love stalking her, we have fun with it.. . . "

3: Moderately severe
"This app is perfect for stalking people. . . "

4: Extremely severe
"honestly if you want your kid to rebel against you even more, this is the app for you! This app has truthfully ruined my teenage years all because my mother now has a way of tracking me down 24/7. I couldn't do the normal teenage things because I was being stalked all day . . . "

". . . i want to share my last seen just to my family and my girlfriend not others. please add new feature in privacy that i can share my last seen to no body except my family and girl friend thanks soo much !"

From these 1,884 reviews, we excluded the reviewers' identifiers, such as usernames. The task was to rate these reviews for convincingness and severity on a four-point Likert scale (1: not, 2: slightly, 3: moderately, 4: extremely). Two authors were annotators. For reviews rated by both, we computed the average convincingness and severity scores. This annotation study possessed minimal risk and was exempted by the Institutional Review Board (IRB) of our university. We used these annotated reviews as our training data.

We extracted linguistic features of 1,884 reviews through the Universal Sentence Encoder (USE) [5], a widely used approach that has proved effective for app reviews [16, 9]. Using USE features, we trained various multitarget (the targets here being convincingness and severity scores) regression models [4] and found that the Support Vector Regressor (SVR) outperforms others. We considered using the reviews' metadata, such as ratings and titles, but found the metadata to be unhelpful for identifying misuse and left it out. Not only negative ratings and titles but also reviews with positive ratings and titles indicate misuse (partly discussed in Section II-B). For example, an abuser writes a review with the title "Great app" and provides five-star ratings as the app facilitates misuse. As these nondiscriminatory attributes may confuse the model, we decided not to include them while training. We used only the text of reviews for training.

We leveraged trained SVR to predict convincingness and severity scores of all 11.57 million reviews in the seed dataset. The alarmingness of each review is calculated by taking the geometric mean of its predicted convincingness and severity.

B. Identifying Exploitable Apps

Using statistical methods, we aggregated the alarmingness of rexviews and ranked all apps according to their reported misuse. From the seed dataset, our model predicted a total of 100 exploitable apps (including false positives). Moreover, our approach is not dependent on the choice of candidate apps and could be applied to any set of apps. To audit additional apps, we applied our model on (i) a dataset of similar apps and (ii) a dataset of 100 popular apps in the utility category. We found not only IPS but also many general-purpose exploitable apps exhibiting multiple exploitable functionalities, which are described below.

1) Similar Apps: For each app determined to be exploitable from the seed dataset, we obtained recommendations for similar apps from the Apple App Store's "You May Also Like" feature. Through this process, we obtained 788 similar apps. Our motivation in using Apple-recommended apps is that these apps should offer functionalities similar to the apps classified as exploitable. Further, we collected reviews (over the period August 2008 to August 2022) of these 788 apps—we term this the *snowball dataset*.

Our model predicts 90 apps as exploitable (including false positives) from the snowball dataset. Our evaluation using the snowball dataset reveals that the model yields a recall of 71.60%, which is much higher than the other baseline approaches. In this scenario, a false negative can lead to harm through misuse, whereas a false positive just wastes effort in an unnecessary audit. Hence, high recall is more valuable than high precision.

2) 100 Popular Utility Apps: Surveillance apps that can be misused for spying fall under the "Utilities" category, making it an important category to audit. We considered 100 popular utility apps (mentioned on the Apple's App Store page [46]) and collected their reviews. Since a popular app can have a lot of reviews (over the years), collecting them can be computationally expensive.

From these 100 apps, our approach, MISSAUDITOR, classifies only one app as exploitable. Upon examining its reviews, we found that app to be nonexploitable. We also scrutinized some apps classified as nonexploitable and found their predictions true (based on manual verification of reviews).

Some app categories, such as payment apps, calculators, and so on, are unlikely to be misused. Hence, they don't form good candidates for identification. Iteratively auditing apps (through MISSAUDITOR) similar to the already identified exploitable apps (as shown in Section III-B1) is a feasible solution to uncover a large exploitable landscape.

3) Relevance of Findings: Some reviews in our datasets are old. For example, the seed dataset was collected over the period of July 2008 to January 2020. To check if our findings are still relevant, we randomly sampled 50 confirmed exploitable apps from the union of the seed and the snowball datasets. We collected new reviews (from January 2020 to April 2024) for these apps and applied our MISSAUDITOR approach to these reviews. Then, we scrutinized their descriptions (to know their basic functionalities) and the alarming reviews (to know misuse cases if any), following the same process as before. We found that the new reviews of three of these 50 apps

don't report misuse. On the other hand, the other 47 apps still possess exploitable functionalities causing potential or actual misuse. A high success rate (94%; 47 of 50) on a random sample implies that most of the identified apps still facilitate misuse.

IV. RQ$_{\text{FUNCTIONALITY}}$: Uncovering exploitable Functionalities

We illustrate the idea of uncovering exploitable functionalities using the exploitable apps found from the seed dataset. First, we analyzed an app's description to understand its functionalities. Second, we analyzed the top alarming reviews (discovered by MissAuditor) for identifying existing misuse and exploitable functionalities. Through this process, we discovered the following types of exploitable functionalities.

Monitoring phone activities. Some apps monitor a victim's phone activities, such as browsing history and text messages. Such apps are installed on the victim's device and activities can be monitored on another synced device. For example, SaferKid Text Monitoring App [44] allows synced devices to monitor call history, web history, texts, and so on.

Audio or video surveillance. Some apps enable audio or video surveillance without the victim's knowledge. These apps listen, view, or record a victim's voice or actions and some of them need not be installed on the victim's phone. For example, Find My Kids: Parental control [38] is misused to record private conversations between people without them knowing.

Tracking location. Some Global Positioning System (GPS) apps enable tracking a victim's phone, making them uncomfortable with access. For example, Find My iPhone [39] is a legitimate app but can be misused to spy on the location of connected devices.

Profile stalking. Some apps are misused for stalking user profiles or user-generated content (such as text and images). For example, Kik Messaging and Chat App [40] can be misused for stalking images and victims' other information on the app.

Table I shows these four types of exploitable functionalities and some alarming reviews reporting them. Some of these reviews are old (2014 or 2012), but we confirmed that similar concerns are being raised in the recent reviews of the same apps. For example, the Find My iPhone [39] app lets its users see the location of the connected devices. Due to unequal power dynamics between the abuser and the victim (say in a family setting) [6], the victim is sometimes forced to connect to such apps and allow their device to be located. In other words, legitimate apps designed for locating loved ones can be misused for exploitable behavior when the requirements for consent [34] are violated. Hence, they are dual-use [6]. Through a qualitative study, Freed et al. [12] found that many find-my-phone applications are intended for anti-theft and safety purposes but are heavily misused against privacy. Chatterjee et al. [6] categorize such apps as spyware, especially in the case of intimate partner surveillance. Just because an app has some legitimate uses doesn't justify its possible misuse. Hence, we consider such apps to be exploitable. Relying on app reviews highlights both types of exploitable functionalities (always malicious and possibly with legitimate uses) against which future and current app users should be warned. Moreover, app distribution platforms and developers should mitigate privacy risks.

We illustrate the exploitable behavior of the SaferKid Text Monitoring App [44] by installing it on two devices: a parent's device (iOS version 14.4.1) and a child's device (Android version 11.0). Figure 5 shows the exploitable features present in this app. Activities on the child's device can be monitored on the synced parent's device. Figure 5(a) shows SaferKid exploitable functionalities such as monitoring text messages, web history, and call history. We verified each of these functionalities. Figure 5(b) shows the screen displaying all chats of the victim. Apps such as SaferKid are advertised as safety apps for children but can be secretly or forcefully installed on another device to monitor the user's activity. Not only parents but anyone can misuse such apps by installing them on a victim's phone.

V. Related Work

We describe previous works focusing on auditing apps, app reviews, and spying through mobile apps.

A. Auditing and App Reviews

Most research on software audits analyzes the flow and dependencies in source code [14, 52, 29, 27]. Only a few audit studies consider data such as system traces [51]. However, all these studies fail to identify exploitable apps because they focus on the technical aspects of building or running them. Thus, their analysis doesn't address cases of forced consent and inappropriate access to public information. Our work leverages app reviews, unexplored by prior studies, for misuse audits.

Prior studies show that app reviews are valuable in other ways. Dhinakaran et al. [10] mine requirement-related information from app reviews and propose an active-learning framework to fetch relevant ones in a semi-supervised manner. Haque et al. [18] leverage reviews to draw implicit and explicit comparisons between competing apps. Guo and Singh [16] extract ⟨action, app problem⟩ pairs from the stories expressed in reviews, which can help direct developers to common problems. Chen et al. [7] showcase the most relevant reviews to developers by grouping informative ones and applying unsupervised techniques. For developers to learn from feature suggestions and bug reports, Kurtanović and Maalej [20] filter rationale-backed reviews through classification. Some studies [26, 2] mine reviews to understand users' perceptions about apps selling their data. However, app reviews remain largely unexplored for understanding misuse by an app's users.

B. Spying through Mobile Apps

Chatterjee et al. [6] apply search queries and manual verification based on information such as app descriptions and permissions to identify Intimate Partner Surveillance apps.

TABLE I
TYPES OF EXPLOITABLE FUNCTIONALITIES.

Exploitable Functionality	App Example	Alarming Review
Monitoring phone activities	SaferKid Text Monitoring App [44]	*. . . Tracking things like social media, texts, and search history is just a complete disregard of privacy. You have to have trust in your kids . . . Apps like these shouldn't be allowed. IF YOU TRUST YOUR KID, DONT DOWNLOAD.* (Date: 2019-12-07)
Audio or video surveillance	Find My Kids: Parental control [38]	*This app proves to have a invasion of privacy. Due to the fact if your kids was at a friends house and talking to his friends parents, this app records what is going on and is a invasion of privacy. If your child left their phone downstairs or anywhere and they are playing it can record private conversation between adult and is a unsafe . . .* (Date: 2019-01-16)
Tracking location	Find My iPhone [39]	*. . . It's supposed to be used to recover a lost phone, not to religiously stalk your children.. . . The fact that a mom actually installed this app onto her son's phone without his knowledge is flat out wrong. . . . If you're constantly monitoring your child 24/7, just imagine what your child will do when they go off to college. . . .* (Date: 2014-02-13)
Profile stalking	Kik Messaging and Chat App [40]	*MAKE THIS SAFE PEOPLE WANT TO USE IT BUT DON'T WANT TO BE STALKED OR ABUSED IN THE APP PROTECT THE APP OR PEOPLE WON'T USE IT MY FRIEND HAD A MAN SEND HER A BAD PICTURE IF YOU KNOW WHAT I MEAN. OVERALL THIS APP IS GREAT!!!!!"* (Date: 2015-06-24)

However, in general, the actual usage deviates from the legitimate purpose shown in app descriptions. To identify such misuse, we focus on the evidence provided in app reviews.

Roundy et al. [33] identify apps used for phone number spoofing and message bombing, which lie outside our scope of exploitable apps. Conversely, exploitable apps include those that enable stalking public information, which lie outside their scope. Roundy et al. use installation data, to uncover spying apps that are installed on infected devices. However, we focus on evidence of exploitable behavior present in app reviews to uncover exploitable apps. Roundy et al. rely upon Norton's security app [42] to determine which devices are infected. Thus, their approach would miss apps that a general user can leverage to spy.

All these studies, along with a few others [48, 54, 49], investigate how technology is abused for spying. However, they do not support the broader set of exploitable apps. In particular, they do not consider cases when the victim is uncomfortable with the access (e.g., of public information) even if aware of it, along with cases involving family, not only strangers, as abusers.

Some studies address user privacy in the context of user information shared with software developers [32, 3, 22] . They do not consider cases where an app user can exploit the app to access the information of a victim (another app user or bystander).

VI. DISCUSSION

We proposed MISSAUDITOR, an approach to automatically analyze app reviews for misuse audits. Specifically, MISSAUDITOR identifies exploitable apps along with their exploitable functionalities from reviews. Doing so reduces the burden of manually installing all apps to perform a misuse audit. MISSAUDITOR predicts exploitable apps from multiple sources. After verification of reviews and apps' descriptions, we confirmed 156 apps facilitating the misuse. To confirm their misuse, we manually investigated their reviews in two steps. First, we scrutinized their top 50 alarming reviews to confirm any exploitable functionalities. Second, we scrutinized their reviews containing our keywords. Solid misuse cases (in at least one of these two steps) give credence to the exploitability of these 156 apps (full list in the appendix ordered alphabetically).

Below, we describe our process of reporting exploitable apps, threats to validity, and promising future directions.

A. Reporting to Apple App Store and App Developers

We informed the Apple App Store about the potential misuse of all exploitable apps that we confirmed above. We found that 17 of them have already been deleted from the Apple Store, possibly because of raised privacy concerns or other unknown reasons.

Only 90 developers (of apps that we confirmed as exploitable) provided their email or chat support on the Apple App Store. We contacted these developers and informed them about the misuse cases (template in appendix). We heard back from 14 of them. Despite asking specific questions, we received generic replies from them. Six of them acknowledged the misuse or informed that their privacy teams take necessary steps in whatever way possible; Five developers assumed that app users (including abusers) would never misuse their apps (an unrealistic assumption); responses from two developers suggested they were not receptive to our feedback; remaining one developer did not answer back after we asked more specific questions.

B. Threats to Validity

We now discuss the threats we identify in our work. The identified threats are of two types: (i) the threats that we mitigate; and (ii) the threats that remain.

1) Threats Mitigated: We mitigated the following threats to validity. To address a potential bias because of our keywords, we constructed a training set that was balanced between reviews that matched and didn't match our keywords. Using this training set helped our model learn from the context and not from specific keywords. Second, instead of crowd workers, who might not understand the problem well, two authors

of this paper annotated the whole training data. Third, the ground truth (of exploitable apps) could be biased if it was formed only using top alarming reviews. We mitigated this bias by scrutinizing reviews from seed and snowball datasets (explained in appendix) containing our keywords, which can have evidence missed by alarming reviews.

2) Threats Remaining: Now, we describe the threats that remain in our work. First, we investigated only a few thousand apps, which may not represent all apps on the Apple App Store [37]. The performance of MISSAUDITOR may be reduced while testing it on all apps of the Apple App Store. Second, we targeted apps and their reviews only on Apple's App Store. Upon deployment on other app stores, the performance of our approach can differ. Third, suppose an app distribution platform does not offer recommendations for similar apps. In that case, finding good candidates for exploitable apps will be a challenge. An alternative is to prioritize applying MISSAUDITOR for the apps flagged by app users (victims). Fourth, some reviews in our datasets are old. Due to frequent app updates, the types of misuse may change over time. Although we confirm that our findings are still relevant in Section III-B3, there is a possibility that a few apps may not be exploitable anymore. Fifth, some findings are based on a random review sample because manually reading all reviews is not feasible. The results may vary in the latter case.

C. Limitations and Future Directions

We identify the following limitations of this approach, each of which suggests topics for future research. First, MISSAUDITOR may miss some exploitable apps if they do not have alarming reviews at the time of analysis, possibly because they are new apps. However, this limitation can be overcome if MISSAUDITOR is used along with app descriptions. Users could be warned against apps whose descriptions are similar to already identified exploitable apps (or against similar apps found through a recommendation system). Further, MISSAUDITOR can flag these new apps as soon as their alarming reviews arrive. In this way, our computational model can be updated by including newly found exploitable cases in the training set and iteratively identifying a large landscape of misuse.

Second, uncovering exploitable functionalities involves the manual effort of inspecting top alarming reviews. A future direction is to automate this process. Moreover, each functionality is analyzed in isolation from other functionalities. Future audit studies can work on the interdependence of the functionalities facilitating misuse.

Third, app reviews describe perceptions of an app and its misuse, which can be affected by their personal preferences, e.g., how much information access is too much to share. Moreover, cases involving family power dynamics (parent-child and intimate partners), especially regarding forced consent, are nontrivial to analyze user interactions and perceptions. Manual inspection of such reviews by developers can mitigate but not eliminate this limitation. Future studies can explore recommending apps based on a user's personal preferences.

Fourth, some reviews may falsely claim the exploitable behavior of the app. This includes reviews written by the app's competitors or by people assuming themselves victims. Identifying such misleading cases is challenging and out of the scope of our study.

Fifth, app reviews can be ambiguous in differentiating between inherently malicious and benign but exploitable functionalities. For example, the Safer Kid Text Monitoring App [44] is designed to keep loved ones safe (benign app) but is misused according to reviews. Future research can distinguish between these benign apps and other malicious ones.

VII. REPRODUCIBILITY

We have released our entire training data along with the computational model on Zenodo [15]. Our appendix provides details of (1) dataset collection and annotation, (2) training and evaluation, and (3) linguistic analysis.

ACKNOWLEDGEMENT

Thanks to the anonymous reviewers for their helpful comments. Thanks to the Department of Defense for partially supporting this research under the Science of Security Lablet. NA acknowledges partial support from EPSRC (EP/W025361/1).

REFERENCES

[1] Debasish Basak, Srimanta Pal, and Dipak Patranabis. Support vector regression. *Neural Information Processing – Letters and Reviews*, 11, November 2007.

[2] Andrew R. Besmer, Jason Watson, and M. Shane Banks. Investigating user perceptions of mobile app privacy: An analysis of user-submitted app reviews. *International Journal of Information Security and Privacy (IJISP)*, 14 (4):74–91, October 2020.

[3] Kenneth Block, Sashank Narain, and G. Noubir. An Autonomic and Permissionless Android Covert Channel. *Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, pages 184–194, 2017.

[4] Hanen Borchani, Gherardo Varando, Concha Bielza, and Pedro Larrañaga. A survey on multi-output regression. *Wiley Interdisciplinary Reviews Data Mining and Knowledge Discovery*, 5(5):216–233, September 2015.

[5] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. Universal sentence encoder. *CoRR*, abs/1803.11175:1–7, 2018.

[6] Rahul Chatterjee, Periwinkle Doerfler, Hadas Orgad, Sam Havron, Jackeline Palmer, Diana Freed, Karen Levy, Nicola Dell, Damon McCoy, and Thomas Ristenpart. The spyware used in intimate partner violence. In *Proceedings of the 39th IEEE Symposium on Security and Privacy (SP)*, pages 441–458, San Francisco, CA, USA, May 2018. IEEE Press.

[7] Ning Chen, Jialiu Lin, Steven C. H. Hoi, Xiaokui Xiao, and Boshen Zhang. Ar-miner: Mining informative reviews for developers from mobile app marketplace. In *Proceedings of the 36th International Conference on*

Software Engineering, pages 767–778, New York, May 2014. Association for Computing Machinery.

[8] Jacob Cohen. *Statistical Power Analysis for the Behavioral Sciences*. Lawrence Erlbaum Associates, Hillsdale, New Jersey, 2nd edition, 1988.

[9] Peter Devine, Yun Sing Koh, and Kelly Blincoe. Evaluating Unsupervised Text Embeddings on Software User Feedback. In *Proceedings of the IEEE 29th International Requirements Engineering Conference Workshops (REW)*, pages 87–95, Indiana, 2021. IEEE. doi: 10.1109/REW53955.2021.00020.

[10] Venkatesh T. Dhinakaran, Raseshwari Pulle, Nirav Ajmeri, and Pradeep K. Murukannaiah. App Review Analysis via Active Learning: Reducing Supervision Effort without Compromising Classification Accuracy. In *Proceedings of 26th IEEE International Requirements Engineering Conference (RE)*, pages 170–181, Banff, Canada, 2018. IEEE.

[11] Tensor Flow. Tensorflow Hub, 2023. URL https://www.tensorflow.org/hub. Accessed 2023-08-05.

[12] Diana Freed, Jackeline Palmer, Diana Minchala, Karen Levy, Thomas Ristenpart, and Nicola Dell. "A Stalker's Paradise": How intimate partner abusers exploit technology. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, pages 1–13, New York, April 2018. Association for Computing Machinery.

[13] Diana Freed, Sam Havron, Emily Tseng, Andrea Gallardo, Rahul Chatterjee, Thomas Ristenpart, and Nicola Dell. Is my phone hacked? analyzing clinical computer security interventions with survivors of intimate partner violence. *Proceedings of the 17th ACM Conference on Human-Computer Interaction*, 3:202:1–202:24, 2019.

[14] César García, Alejandro Guerrero, Joshua Zeitsoff, Srujay Korlakunta, Pablo Fernandez, Armando Fox, and Antonio Ruiz-Cortés. Bluejay: a cross-tooling audit framework for agile software teams. In *Proceedings of 43rd IEEE International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)*, pages 283–288, Online, 2021. IEEE.

[15] Vaibhav Garg, Hui Guo, Nirav Ajmeri, Saikath Bhattacharya, and Munindar P. Singh. Missauditor Dataset and Software, 2024. Zenodo Public Repository. Accessed 2024-07-13.

[16] Hui Guo and Munindar P. Singh. Caspar: Extracting and Synthesizing User Stories of Problems from App Reviews. In *Proceedings of the IEEE 42nd International Conference on Software Engineering*, pages 628—-640, New York, NY, USA, July 2020. Association for Computing Machinery. ISBN 9781450371216. doi: 10.1145/3377811.3380924.

[17] Kevin A. Hallgren. Computing inter-rater reliability for observational data: An overview and tutorial. *Tutorials in Quantitative Methods for Psychology*, 8(1):23–34, 2012.

[18] Amanul Haque, Vaibhav Garg, Hui Guo, and Munindar P Singh. Pixie: Preference in Implicit and Explicit Comparisons. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, volume 2, pages 106–112, Dublin, Ireland, 2022. Association for Computational Linguistics.

[19] Sam Havron, Diana Freed, Rahul Chatterjee, Damon McCoy, Nicola Dell, and Thomas Ristenpart. Clinical computer security for victims of intimate partner violence. In *Proceedings of the 28th USENIX Security Symposium*, pages 105–122, Santa Clara, January 2019. USENIX Association.

[20] Zijad Kurtanović and Walid Maalej. Mining user rationale from software reviews. In *Proceedings of the 25th IEEE International Requirements Engineering Conference (RE)*, pages 61–70, Lisbon, Portugal, September 2017. IEEE Press.

[21] Christopher D. Manning. Part-of-speech tagging from 97% to 100%: Is it time for some linguistics? In *Proceedings of the Computational Linguistics and Intelligent Text Processing*, pages 171–189, Berlin, Heidelberg, February 2011. Springer.

[22] Claudio Marforio, Hubert Ritzdorf, Aurélien Francillon, and Srdjan Capkun. Analysis of the Communication between Colluding Applications on Modern Smartphones. In *Proceedings of the 28th Annual Computer Security Applications Conference*, pages 51–60, New York, December 2012. Association for Computing Machinery.

[23] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS, pages 3111–3119, Lake Tahoe, Nevada, December 2013. Neural Information Processing Systems Foundation.

[24] George A. Miller. Wordnet: A lexical database for english. *Communications of the ACM*, 38(11):39–41, November 1995. doi: 10.1145/219717.219748.

[25] Saif M. Mohammad. Obtaining Reliable Human Ratings of Valence, Arousal, and Dominance for 20,000 English Words. In *Proceedings of The Annual Conference of the Association for Computational Linguistics (ACL)*, Melbourne, Australia, 2018.

[26] Duc Cuong Nguyen, Erik Derr, Michael Backes, and Sven Bugiel. Short text, large effect: Measuring the impact of user reviews on android app security & privacy. In *Proceedings of the 40th IEEE Symposium on Security and Privacy (SP)*, pages 555–569, San Francisco, May 2019. IEEE Computer Society.

[27] Ivan Pashchenko, Henrik Plate, Serena Elisa Ponta, Antonino Sabetta, and Fabio Massacci. Vuln4real: A methodology for counting actually vulnerable dependencies. *IEEE Transactions on Software Engineering*, 48(5): 1592–1609, 2020.

[28] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics.

[29] Henning Perl, Sergej Dechand, Matthew Smith, Daniel Arp, Fabian Yamaguchi, Konrad Rieck, Sascha Fahl, and Yasemin Acar. VCCFinder: Finding Potential Vulnera-

bilities in Open-Source Projects to Assist Code Audits. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 426–437, New York, NY, USA, 2015. Association for Computing Machinery.

[30] PyDictionary. Dictionary in Python, 2023. URL https://pypi.org/project/PyDictionary/. Accessed 2023-08-05.

[31] Pythesaurus. Thesaurus in Python, 2023. URL https://pypi.org/project/py-thesaurus/. Accessed 2023-08-05.

[32] Joel Reardon, Álvaro Feal, Primal Wijesekera, Amit Elazari Bar On, Narseo Vallina-Rodriguez, and Serge Egelman. 50 Ways to Leak Your Data: An Exploration of Apps' Circumvention of the Android Permissions System. In *Proceedings of the 28th USENIX Security Symposium*, pages 603–620, Santa Clara, August 2019. USENIX Association.

[33] Kevin A. Roundy, Paula Barmaimon Mendelberg, Nicola Dell, Damon McCoy, Daniel Nissani, Thomas Ristenpart, and Acar Tamersoy. The Many Kinds of Creepware Used for Interpersonal Attacks. In *Proceedings of the 41th IEEE Symposium on Security and Privacy (SP)*, pages 753–770, Los Alamitos, May 2020. IEEE Computer Society.

[34] Munindar P. Singh. Consent as a foundation for responsible autonomy. *Proceedings of the 36th AAAI Conference on Artificial Intelligence (AAAI)*, 36(11):12301–12306, February 2022. doi: 10.1609/aaai.v36i11.21494. Blue Sky Track.

[35] Paul F. Smith, Siva Ganesh, and Ping Liu. A comparison of random forest regression and multiple linear regression for prediction in neuroscience. *Journal of Neuroscience Methods*, 220:85–91, October 2013.

[36] Apple App Store. AirBeam Video Surveillance App, 2023. URL https://apps.apple.com/us/app/airbeam-video-surveillance/id428767956. Accessed 2023-10-08.

[37] Apple App Store. Apple App Store, 2023. URL https://www.apple.com/app-store/. Accessed 2023-11-11.

[38] Apple App Store. Find My Kids: Parental Control, 2023. URL https://apps.apple.com/us/app/id994098803. Accessed 2023-10-08.

[39] Apple App Store. Find My iPhone App, 2023. URL https://apps.apple.com/us/app/find-my-iphone/id376101648. Accessed 2023-10-08.

[40] Apple App Store. Kik App, 2023. URL https://apps.apple.com/us/app/kik/id357218860. Accessed 2023-10-08.

[41] Apple App Store. Life360 App, 2023. URL https://apps.apple.com/us/app/life360-find-family-friends/id384830320. Accessed 2023-10-08.

[42] Apple App Store. Nortan Mobile Security App, 2023. URL https://buy-static.norton.com/norton/ps/bb/ushard/4up_mnav05w_us_en_fl_tw_branded_mix-n360.html. Accessed 2023-10-08.

[43] Apple App Store. Ourpact Jr. Child App, 2023. URL https://apps.apple.com/us/app/id1127917970. Accessed 2023-10-08.

[44] Apple App Store. Saferkid Text Monitoring App, 2023. URL https://apps.apple.com/us/app/saferkid-text-monitoring-app/id1143802529. Accessed 2023-10-08.

[45] Apple App Store. 3Fun: Threesome & Swingers App, 2023. URL https://apps.apple.com/app/id1164067996. Accessed 2023-10-08.

[46] Apple App Store. Popular Utilities Apps, 2023. URL https://apps.apple.com/us/genre/ios-utilities/id6002. Accessed 2023-11-11.

[47] Google Play Store. Google Play Store, 2023. URL https://play.google.com/store/apps. Accessed 2023-02-15.

[48] Emily Tseng, Rosanna Bellini, Nora McDonald, Matan Danos, Rachel Greenstadt, Damon McCoy, Nicola Dell, and Thomas Ristenpart. The Tools and Tactics Used in Intimate Partner Surveillance: An Analysis of Online Infidelity Forums. In *Proceedings of the 29th USENIX Security Symposium*, pages 1893–1909, Virtual, August 2020. USENIX Association.

[49] Emily Tseng, Diana Freed, Kristen Engel, Thomas Ristenpart, and Nicola Dell. A digital safety dilemma: Analysis of computer-mediated computer security interventions for intimate partner violence during covid-19. In *Proceedings of the 2021 Conference on Human Factors in Computing Systems*, pages 1–17, Virtual, 2021. Association for Computing Machinery.

[50] Alper Kursat Uysal and Serkan Gunal. The impact of preprocessing on text classification. *Information Processing & Management*, 50(1):104–112, 2014.

[51] Ting-Yu Wang, Haiming Jin, and Klara Nahrstedt. mauditor: Mobile auditing framework for mhealth applications. In *Proceedings of the 2015 Workshop on Pervasive Wireless Healthcare*, pages 7–12, New York, NY, USA, 2015. Association for Computing Machinery.

[52] Mingyuan Xia, Lu Gong, Yuanhao Lyu, Zhengwei Qi, and Xue Liu. Effective real-time android application auditing. In *Proceedings of the 36th IEEE Symposium on Security and Privacy*, pages 899–914, San Jose, May 2015. IEEE.

[53] Min Xu, Pakorn Watanachaturaporn, Pramod K. Varshney, and Manoj K. Arora. Decision tree regression for soft classification of remote sensing data. *Remote Sensing of Environment*, 97(3):322–336, 2005. doi: https://doi.org/10.1016/j.rse.2005.05.008.

[54] Yixin Zou, Allison McDonald, Julia Narakornpichit, Nicola Dell, Thomas Ristenpart, Kevin Roundy, Florian Schaub, and Acar Tamersoy. The role of computer security customer support in helping survivors of intimate partner violence. In *Proceedings of the 30th USENIX Security Symposium*, pages 429–446, Virtual, 2021. USENIX Association.

APPENDIX

A. Seed Dataset

Chatterjee et al. [6] identified 2,707 iOS apps as potential Intimate Partner Surveillance (IPS) apps, i.e., apps used by a person to spy on their intimate partner. Out of these 2,707

apps, Chatterjee et al. confirmed 414 apps to be IPS, using semi-supervised pruning.

When we collected our data (from July 2008 to January 2020), 1,687 (out of 2,707) apps existed with at least one review on the Apple App Store. This yielded our *seed dataset* containing 11.57 million reviews. Out of 1687 apps in the seed dataset, only 210 were confirmed IPS by Chatterjee et al..

B. Our Keywords

For the formation of a list of keywords, we initialized a set with the words: *spy*, *stalk*, and *stealth* and queried WordNet [24] for their synonyms. We performed the query operation until we didn't find any new word in the set. The resulting set contained keywords: *spy*, *stalk*, *stealth*, *descry*, *chaff*, and *haunt*. However, *chaff* and *haunt* weren't relevant for describing exploitable behavior in reviews. Also, *descry* was present in only two reviews, both of which were irrelevant to exploitable behavior. To expand the set of keywords, we explored other corpora such as PyDictionary [30] and Thesaurus [31] but did not find synonyms that are widely used in app reviews. Moreover, keywords used in the previous study [6], such as *track* and *control* bring many false positive reviews. For example, "I like tracking my distance when I walk with my dog." and "...you can also control the audio of your mac through the app ...I can control music tracks without having to touch the computer." are not relevant. Thus, our relevant set of keywords reverts to *spy*, *stalk*, and *stealth*.

C. Linguistic Analysis

To compute valence, arousal, and dominance scores, we leveraged lexicons provided by Mohammad [25], which contains scores for about 20,000 English words. First, from each review, we extracted relevant sentences using our keywords. Second, using parts of speech tagging, we extracted adjectives mentioned in the relevant sentences. Since sentences describe the exploitable behavior, adjectives in them represent the reviewer's sentiment. For example, *"This horrible (adj) app spy on me"*. Third, for each type of reviewer, we computed average scores of adjectives that are present in the lexicon.

D. Annotating Reviews and Model Training

The annotation was conducted in three steps. First, two authors of this paper rated 599 of 1,884 selected reviews according to the initial set of annotation instructions. The initial instructions included definitions (of convincingness and severity scores) and examples corresponding to each point on the Likert scale. In this step, for each annotator, we calculated the alarmingness scores of reviews using the convincingness and severity ratings. If the alarmingness scores computed for both the annotators were not at least three (median value on Likert scale), or both are not less than three, annotators resolved such cases via discussions. After discussing, the annotators produced the final set of annotation instructions. In the second step, the annotators followed the final instructions and rated 900 reviews. In this step, we computed an Intraclass Correlation Coefficient (ICC) [17] to measure inter-rater agreement. We found ICC(3,k) to be 0.9195 for convincingness and 0.9190 for severity, which accounts for excellent agreement. ICC is suitable for Likert scale ratings. Unlike other measures such as Cohen's [8] kappa, which is based on (all or nothing) agreement, ICC takes into account the magnitude of agreement (or disagreement) to compute inter-rater reliability. In the third step, the remaining reviews were divided among the two annotators so that only one annotator rates each review.

1) Extracting Deep Learning Features from App Review: We obtained the feature vector of each app review as follows:

Combine Sentences: Remove periods in each app review and combine all its sentences to form single sentence.

Text Preprocessing: Remove all punctuation marks, stop words [50], and our keywords, the latter because those keywords may correlate with reviews with higher scores and could create bias in the model.

Sentence Embedding: We leverage the Universal Sentence Encoder (USE) [5] to extract embeddings for each app review. USE uses Deep Averaging Network (DAN) to provide a 512-dimension embedding for a long text. USE is trained on a large variety of natural language tasks with the aim of capturing the context. In our case, USE directly provides sentence level embeddings of an app review, by keeping the context intact. However, alternatives such as GLoVe [28] and Word2Vec [23] lose such context. We leverage the pretrained USE network by using Tensorflow Hub [11].

We evaluated the performance of three regression models: support vector regression [1], random forest [35], and decision tree [53], by ten-fold cross-validation on our dataset. To mitigate bias of our keywords, we remove such keywords in the preprocessing step, so that the regression model learns from the the context of the review and not from specific keywords. As shown in Table II, the Support Vector Regressor (SVR) yields the smallest MSE of 0.625 with 0.458 standard deviation, so we choose it for the subsequent phases of our approach.

TABLE II
PERFORMANCE OF THREE REGRESSION MODELS ON TEN-FOLD CROSS VALIDATION.

Regression Model	Average MSE	Standard Deviation
Decision Tree	1.344	0.402
Random Forest	0.712	0.417
Support Vector	**0.625**	0.458

For training, we also explored OpenAI's two advanced embeddings: text-embedding-3-small and text-embedding-3-large. However, while using these embeddings, MSE over 10 folds do not significantly differ (p-value is 0.0582 (¿0.05) for small embedding and 0.0533 (¿0.05) for large embedding) from MSE values of USE based model. Moreover, training OpenAI's embedding-based model is more expensive than training on USE embeddings. OpenAI's embeddings occupy 3 times (1536-dimensional vector for small embedding) and 6 times (3072-dimensional for large embedding) larger space than USE embedding. Hence, we stick to the USE embedding-based SVR model for predictions.

E. Statistical Methods for Identifying and Ranking exploitable Apps

Weighted Mean of Alarmingness: In general, for a exploitable app, a small proportion of reviews report exploitable behavior. Thus, we need to catch exploitable apps using their few reviews that have high values on the alarmingness scale. Thus, we assign weights to reviews based on their alarmigness, as follows:

Defining score buckets: While annotating reviews, we defined levels of convincing reviews (not convincing to extremely convincing) and severe reviews (not severe to extremely severe) on a Likert scale. We also follow same levels on the alarmingness scale (1: not alarming to 4: extremely alarming). We define a score bucket between every consecutive level of alarmingness (not alarming to slightly, slightly alarming to moderately alarming, moderately alarming to extremely alarming). Table III shows how score buckets are formed using levels of alarmingness.

Assigning weights to score buckets: We have 11.57 million reviews in the seed dataset. Based on the alarmingness computation (discussed in main document), we calculated the probability of a review falling in a score bucket. Since, the reviews reporting exploitable behavior are less, probabilities in buckets 2 and 3 are less than that in bucket 1. We take inverse of these probabilities to get the weights for each score bucket. As a result, we assign higher weights to buckets 2 and 3 than to the bucket 1. Table III also shows the weight assigned to each score bucket.

TABLE III
SCORE BUCKETS FOR ALARMINGNESS.

Alarmingness Score Range	Alarmingness Level Range	Bucket	Bucket Weight
$[1, 2)$	Not alarming to Slightly	1	$2.29 \cdot 10^{-3}$
$[2, 3)$	Slightly to Moderately	2	$6.08 \cdot 10^{-2}$
$[3, 4]$	Moderately to Extremely	3	$9.36 \cdot 10^{-1}$

If a_1, a_2, …, a_n are the alarmingness scores of an app's reviews, and w_1, w_2, …, w_n are their respective weights (according to Table III), then, $W_{\text{alarmingness}}$, the weighted mean of alarmingness is given by:

$$W_{\text{alarmingness}} = \frac{a_1 * w_1 + a_2 * w_2 + \ldots a_n * w_n}{w_1 + w_2 + \ldots w_n}$$

The weighted mean of alarmingness ranges from 1 to 4.

Normalized Count: The weighted mean of alarmingness does not account for the count of reviews that report exploitable behavior against an app. Suppose, *app A* has 15 reviews reporting exploitable behavior and *app B* has 25 reviews reporting exploitable behavior. If all reviews reporting exploitable behavior have the same alarmingness score, the weighted means of the two apps would be the same. But, *app B* shows more evidence of exploitable behavior and should have a higher exploitable score than *app A*. Thus, we also consider the count of reviews. For each app, we calculate the number of reviews in bucket 3. We tried incorporating counts of other buckets, but it led to worse performance of the approach.

The minimum possible value of the count is zero. However, in some cases, counts can be high, leading to no definite upper limit. Thus, we normalize the counts of all the apps between one and four.

We want to assign high exploitable score to apps that have high scores in both (i) weighted mean of alarmingness and (ii) normalized count. Thus, exploitable score is computed as the geometric mean of these two values.

1) Selecting Threshold for Prediction: For each app in the seed dataset, MISSAUDITOR computes the exploitable score. All apps are ranked in decreasing order of exploitable scores. The apps with a score greater than a threshold are predicted exploitable. To decide the correct threshold, we follow two steps: (i) label the ground truth of exploitable apps and (ii) vary a threshold between certain values and choose the threshold which gives the best performance of MISSAUDITOR.

We create our ground truth by manually scrutinizing reviews. However, scrutinizing reviews of all 1,687 apps (seed dataset) is not feasible. Thus, first, we scrutinize the most 50 alarming reviews (with a minimum score of two—at least slightly alarming) for apps with the highest 100 exploitable scores. Second, we scrutinize reviews containing our keywords for these 100 apps. The first step is aligned with our approach since it checks the top alarming reviews. However, the second step is neutral because it searches for evidence for the apps that MISSAUDITOR failed to identify through top alarming reviews. This way we mitigate the threat of bias while curating ground truth for apps. On average, top 50 alarming reviews cover $\sim$75% distribution of these alarmingness scores, hence we decided to scrutinize the most 50 alarming reviews for each of these apps. We also adopt this step of review scrutiny for uncovering exploitable functionalities (Section 4 of the main manuscript) and investigating the relevance of our findings (Section 3.2.3 of the main manuscript).

We label an app as exploitable provided any of the scrutinized reviews report a exploitable behavior. Table IV shows the types of reviews we consider indicative or otherwise of a exploitable evidence. If the reviews of an app describe information access performed without the victim's knowledge or when the victim shows discomfort (first three reviews in Table IV), we consider the app as exploitable. Also, some exploitable apps are used for positive purposes such tracking family members for safety (fourth review in Table IV) but still possess a potential for future misuse. Similarly, apps used for tracking pets or other objects are considered exploitable (fifth review). Reviews of some apps don't possess any misuse in present or in future, leading to their final label of not exploitable (sixth review). Through manual inspection, we determine that of the 100 apps, 73 are exploitable and 27 are not.

For the 100 apps, the exploitable score varies between 1.74 and 3.60. We vary the threshold from 1.73 to 3.59 in steps of 0.01. Apps with a exploitable score above the threshold are predicted exploitable but not exploitable otherwise. At each value of threshold, we report the recall, precision, and F1 score.

TABLE IV
INSTRUCTIONS FOLLOWED FOR LABELING EXPLOITABLE APPS.

Type of case	Subtype of case	Example	Exploitable?
Tracking people's information	Without the victim's knowledge	*"Now that I can spy on my wife I will always know when she is cheating"*	Yes
Tracking people's information	With the victim's knowledge but with discomfort	*"Ok my mom got this for me and . . . it's kinda creepy that this app was made so parent could basically stalk their kids."*	Yes
Tracking people's information	Public information but the victim is uncomfortable	*"I had someone cyberstalking and harassing me. Multiple attempts in every way shape and form were made to contact app-name to block and ban the stalker's account due to a concern for my well- being."*	Yes
Tracking people's information	Positive purpose	*"I love finding my family members. Wife was in bad car wreak and I was able to find her location using this app. Thank you!"*	Yes
Tracking pets or other objects		*"Wow! Day one and I'm stalking my puppet like a soccer mom that ran out of adderall! I'm very excited to use this to interact with my puppet while I'm at work and to check in on the dog walker!"*	Yes
Not related to information accessing		*"an absolutely amazing and very helpful app. i don't know how i would keep track of prayer times without it. love the app. thank u!!!"*	No

Table V shows the performance achieved at specific thresholds. As we increase the threshold, the precision increases at the cost of recall. For exploitable apps, a false negative costs more than a false positive because a false negative leaves a exploitable app undetected, which can harm many victims, whereas a false positive causes only wasted effort in manual scrutiny. Hence, achieving high recall is more important than achieving high precision. Thus, from Table V, we choose 1.73 threshold that gives the best recall of 100% at 73% precision. Since we fine-tune the threshold on the same seed dataset, we also check MISSAUDITOR's performance (using the chosen threshold) on the other dataset (Section F).

TABLE V
CHOOSING AN APPROPRIATE THRESHOLD ACCORDING TO THE RECALL SCORES.

Threshold	Precision (%)	Recall (%)	F1 Score (%)
1.73	**73.00**	**100.00**	**84.39**
1.74	73.40	94.52	82.63
1.75	76.13	91.78	83.22
1.76	76.82	86.30	81.29
1.77	76.92	82.19	79.47
1.78	78.37	79.45	78.91
1.79	79.71	75.34	77.46
1.80	80.95	69.86	75.00
1.81	80.95	69.86	75.00
1.82	81.96	68.49	74.62
1.83	81.03	64.38	71.75
1.84	80.35	61.64	69.76
1.85	82.69	58.90	68.80
1.86	83.33	54.79	66.11
1.87	82.97	53.42	65.00

2) Performance of Baseline Methods: On the seed dataset, we also check the performance of baseline methods described below.

Our Keywords on App Description. We search for the presence of one of our keywords (*spy*, *stalk*, and *stealth*) in app descriptions. Apps whose descriptions contain any of these keywords are predicted exploitable, whereas other apps are predicted as not exploitable.

Extended Keywords on App Description. We identify additional relevant keywords by extracting verbs through Part-Of-Speech (POS) tagging [21]. POS tagging marks every word in a sentence to an appropriate part of speech (verb, noun, adjective, and so on). Applying this process on descriptions of 73 exploitable apps (from the ground truth) produced 145 verbs, out of which six (*track*, *monitor*, *locate*, *control*, *stolen*, *lost*) we selected as relevant to exploitable behavior. The verbs: *stolen* and *lost* are relevant because they describe the apps that are used to find a misplaced phone, which indicates an ability to track another device. We extend our keywords by adding these six verbs. Apps whose description contain these keywords are predicted exploitable.

T% Keyword Reviews. For each app, we compute the percentage of reviews containing our keywords. We set a threshold, T, on this percentage, above which apps are predicted exploitable. In our evaluation, T takes the values of 0.3, 0.2, and 0.1, respectively.

Example 5: Exploitable app from seed dataset

App: Find My Family & Friends [41]
Exploitable Score: 3.60/4.00

Alarming Review 1
Alarmingness: 4.00/4.00
Date of Review: 2019-11-28
"...Such a terrible thing for unaware parents to use. Most parents think teens don't need privacy and they constantly need to know where they are and what they're doing and who they're with at all times. This may make the parent feel at peace but what about the child? It's selfish of parents to not take into consideration of how the teen may feel about always having this app and the parent giving them a very stalkish feeling, it's very uncomfortable."

Table VI summarizes the precision, recall, and F1 scores of all baselines and our approach. Our keywords on the description predict only one exploitable app, leading to 100% precision (highest among all). However, our keywords miss 72 exploitable apps, which leads to the worst recall of 1.36%. Among all the baselines, keyword search on reviews with 0.1%

Example 6: Exploitable app from seed dataset

App: OurPact Jr. Child App [43]
Exploitable Score: 2.47/4

Alarming Review 1
Alarmingness: 4.00/4.00
Date of Review: 2018-06-19
"...however this app shuts down almost everything and can see every text and website you've visited. now, i haven't done anything bad online (recently), but i find that a little creepy and honestly an invasion of privacy. no wonder this app has such crappy reviews. also, i used to have way more apps than i do now. because my parents now have the ability to restrict apps that may be "inappropriate". i already have to ask permission to download apps, so if they were inappropriate my parents wouldn't let me download them. there's too many apps like this and i think kids need a break from all this crap on their devices."

Alarming Review 2
Alarmingness: 4.00/4.00
Date of Review: 2018-08-16
"This is a useless app that no parent need to install I pray for every child who has this app installed on their electronics some parents don't understand the modern society but that's okay (but not really) I'm only given 2 hours and writing this review is using up time WHICH IS NOT FRIKEN OK!!! I hate this hate this app and I hope every child that has had their device attacked by this installment hates this app as much as me. This app should never be okay to use its inappropriate and everybody's children who have this app installed are making there children ANTI-SOCIAL AND VERY NOT COOL. I have many reasons why this app is SOOOOOOO scaring and dreadful so if your reading and thinking about installing this on ur child's device DONT INSTALL IT because that will ruin their future."

threshold achieves the highest recall of 65.07%, which is much lower than MISSAUDITOR's recall value. MISSAUDITOR's better performance may be due to fine tuning MISSAUDITOR's threshold on the same seed dataset. Thus, we also compare MISSAUDITOR's performance with these baselines on the other dataset (Section F).

TABLE VI
PERFORMANCE (IN %) OF BASELINE METHODS AND MISSAUDITOR ON THE SEED DATASET. BOLD VALUE FOR A METRIC INDICATES THE HIGHEST SCORE AMONG ALL APPROACHES.

Method	Recall	Precision	F1
Our keywords on app descriptions	01.36	**100.00**	02.68
Extended keywords on app descriptions	61.64	80.35	69.76
0.3% keyword reviews	46.03	96.66	62.36
0.2% keyword reviews	50.79	96.96	66.66
0.1% keyword reviews	65.07	95.34	77.35
MISSAUDITOR	**100.00**	73.00	**84.39**

Examples 5 and 6 show alarming reviews of Find My Family & Friends App [41] and OurPact Jr. Child App [43], which MISSAUDITOR correctly identifies as exploitable. Both of them are dual-use apps. Find My Family & Friends is a safety app, but alarming reviews report parents misusing the tracking functionality on children, to which children are uncomfortable. Moreover, the alarming reviews of the OurPact Jr. Child App report that parents can monitor children's texts and visited websites by installing the app on the child's device.

F. Performance on Snowball Dataset

For a fair evaluation of MISSAUDITOR on the snowball dataset, we manually scrutinized significantly more apps than what MISSAUDITOR predicted. Hence, using the same labeling process as described in Section E1, we curated the ground truth, for the apps with the highest 140 exploitable scores. Out of these 140 apps, we label 81 apps as exploitable.

Table VII shows the performance of all baseline methods and MISSAUDITOR on the snowball dataset. Our keywords when used on descriptions predict no app as exploitable, leading to the lowest recall. This is because, on Apple App Store [37], dual-use apps are not advertised using keywords: *spy*, *stalk*, and *stealth*.

On app descriptions, extended keywords perform better (61.72% recall at 79.36% precision) than our keywords due to commonly used words (such as *track*, *locate*) in app descriptions. Our keywords when applied on reviews yields 100% precision, however at a low recall value of 30.86%. Having high recall is desirable in the context of exploitable apps. Our model yields the best recall of 71.6% as compared to all other baselines.

TABLE VII
PERFORMANCE (IN %) OF BASELINE METHODS AND MISSAUDITOR ON THE SNOWBALL DATASET. BOLD VALUE FOR A METRIC INDICATES THE HIGHEST SCORE AMONG ALL APPROACHES.

Method	Recall	Precision	F1
Our keywords on app descriptions	0	–	–
Extended keywords on descriptions	61.72	79.36	**69.44**
All keyword reviews	30.86	**100**	47.16
MISSAUDITOR	**71.60**	64.44	67.84

G. Scrutinizing Utility Apps

We consider 100 popular utility apps that are mentioned on the Apple App Store page [46]. Out of 100 apps, nine are already scrutinized either in the seed or snowball dataset. For the rest 91 apps, we retrieved 392,928 reviews, over the duration of October 2008 to August 2022.

MISSAUDITOR predicts only one app as exploitable, which after reviews' scrutiny by us, comes out to be non-exploitable. We also scrutinize 10 apps with the highest exploitable scores, by reading their top 50 alarming reviews and reviews containing our keywords. But, none of them are actually exploitable.

H. Contacting App Developers

We used the following template to reach out to app developers.

Example 7: Reaching out to developers.

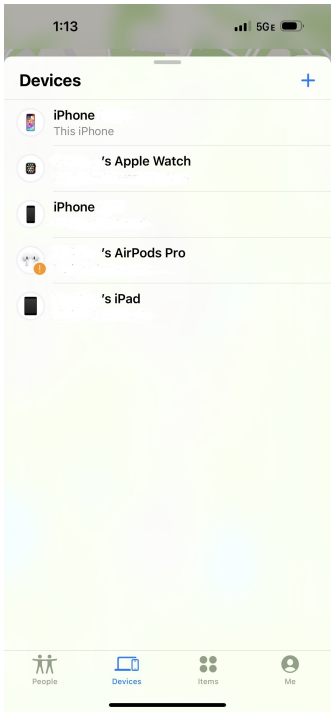
We are security and privacy researchers from the NC State University. We found that your app is designed for legitimate usage but may still be misused against the privacy of some users. We got to know such potential cases through your app reviews shown below.

<Sample Alarming Reviews>

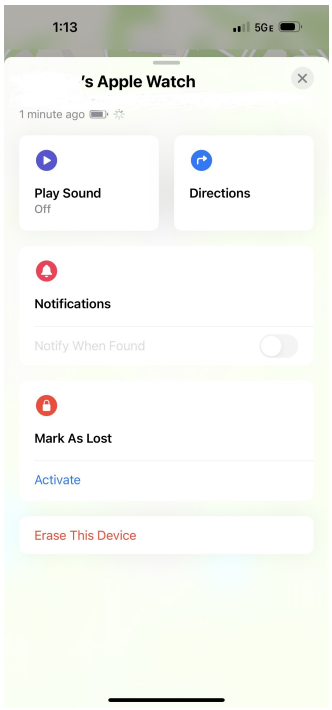
We thought this issue is worth bringing to your attention so that you can take appropriate measures. Please let us know what you think about the same by answering the following questions:
1. Were you aware that users may misuse your apps for the wrong purposes?
2. Do you consider redesigning the app to prevent the users from misuse?
3. Any other steps you think can be taken to ensure victims' privacy?

I. List of exploitable Apps Identified

• 5-0 Radio Police Scanner • Ahgoo baby monitor - audio and video monitoring • Alfred Home Security Camera • AngelSense Guardian • Annie Baby Monitor: Nanny Cam • AT&T FamilyMap® • AT&T Secure Family Companion • Baby Monitor 3G • Baby Monitor 3G/4G/5G/Wi-Fi • Baby Monitor: Video Nanny Cam • Baby Tracker by Sprout • Bark - Parental Controls • BeenVerified: People Search • Bloomz: For Teachers & Schools • Bond - Personal Security • Boomerang Parental Control • bSafe - Never Walk Alone • bthere • BuddyTag • Canopy - Parental Control App • Carpin - Find Family & Friends • Circle 1st Generation • Circle Parental Controls App • Citizen: Local Safety Alerts • Cloud Baby Monitor • Covenant Eyes: Quit Porn Now • Covert Alert • Ear Spy: Super Hearing • FamiGo: Parental Control App • Familo: Find My Phone Locator • Family GPS Locator GeoZilla • Family GPS Tracker Kid Control • Family Locator and GPS Tracker • Family Locator by Fameelee • Family Locator by Fameelee • Family locator My Family • Family Orbit: Parental Control • FamilyTime - Parental Controls • FamilyTime App For Kid's iPhone & iPad • FamilyWall: Family Organizer • FamiSafe Jr - Blocksite • FamiSafe-Parental Control App • Find My Family Friends & Phone • Find My Family, Friends, Phone • Find My Friends - Phone Locator • Find my Friends & Family Track • Find my Friends, Family Phone • Find My iPhone • Find My Kids Footprints • Find my kids: child GPS tracker • Find my Phone - Family Locator • Find my Phone, Friends - iMapp • Find My Phone, Friends Tracker • Findup: Phone Location Tracker • FollowMee GPS Location Tracker • Funa - Family Locator • GeoLoc - GPS Location Tracker • GizmoHub • Glympse-Share your location • GoGuardian Parent App • Google Family Link • GPS Phone Tracker for Smartphones • GPS TRACKER (Phone location tracking) • GPS Tracker — GPS tracking • GPS Tracker and Locator Chirp • GPS Tracker by FollowMee • GPSme Friends & Family Tracker • GroupMe • Here Comes the Bus • HeyTell • HiddenApp, Find My Device App • Hum: GPS Family Locator • iHeartCam Home Security Camera • iMap - Find My Phone & Friends • iSharing: Share Live Location • Kaspersky Safe Kids with GPS • Kidgy - Parental control app • Kidgy: Find My Family • Kidslox - Parental Control App • Kik Messaging & Chat App • LMK: Make New Friends • Locate & Track Phone By Number • Location Tracker - find GPS • Locator for Family & Friends • MamaBear Family Safety • Meet24 - Flirt, Chat, Singles • MeetMe - Meet, Chat & Go Live • Microsoft Family Safety • MMGuardian Parent App • MMGuardian Parental Control • Mobicip Parental Controls • MobiLinc Cam Viewer • MovieStarPlanet • mSpy Lite - Phone tracker app • MyHeritage: Family Tree & DNA • Net Nanny Parental Control App • Norton Family Parental Control • Off Remote • Offender Locator • Offender Locator Lite • Online Walkie Talkie Pro • OurFamilyWizard Co-Parent App • OurPact Jr. Child App • Parental Control & Screen Time • Parental Control App - unGlue • Parental Control Kid's App • Parental Control Parent's App • ParentKit - Parental Controls for iOS • People Tracker Pro - Cell Phone Tracker App! • Periscope Live Video Streaming • Phone Tracker By Number • Phone Tracker for iPhones • Phone Tracker for iPhones (Track people with GPS) • Pingo by Findmykids • Placeter • PocketFinder 2 • Presence: Video Security • Qustodio Parental Control • Qustodio Parental Control App • React Mobile – Safety App • Safe Family: Screen Time App • Safe SMS • SaferKid Text Monitoring App • Safety App for Silent Beacon • Screen Time Parental Control • ScreenGuide Parental Control • SecureTeen Parental Control • Securly Home • SeTracker2 • Share Location: Phone Tracker • Skout — Meet New People • Smart Family Companion • SoSecure by ADT: Safety App • Spoten Phone Location Tracker • StudentVUE • T-Mobile FamilyMode • Tango - Video Call & Chat • TeenOrbit Parent Control Panel • Text Free: Texting App + SMS • Text Me - Phone Call + Texting • TextFree: Call + Texting Line • TextNow • Two Way : Walkie Talkie • unGlue Kids • Verizon FamilyBase • Viber: Free Calling & Texting • Voxer Walkie Talkie Messenger • WebWatcher Parent App • WeSeeYou Safety App • WhereAreYou App Locate friends • Whisper - Share, Express, Meet • Wink - Dating & Friends • Wizz App - chat now • XFINITY xFi • Yik Yak • Zenly - your social map

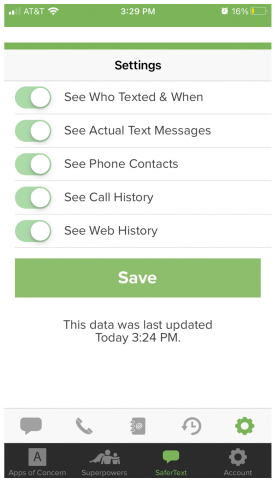


(a) Find My iPhone [39] app enables the abuser to track multiple devices at a time.

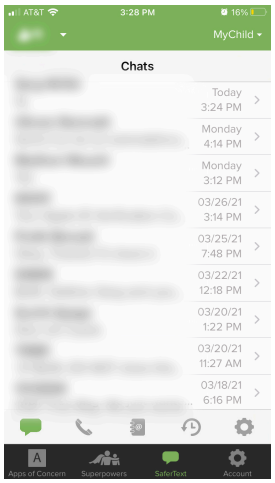


(b) After selecting a device, abuser can see the directions to victim's location.

Fig. 4. Exploitable functionalities in Find My iPhone [39]. The names and locations of devices are hidden for anonymity.



(a) The SaferKid [44] app provides multiple ways to monitor victim's phone activities.



(b) Using the SaferKid [44] app, victim's chats can be seen on the abuser's phone.

Fig. 5. Exploitable functionalities in SaferKid Text Monitoring [44]. The actual chats and device's name are hidden for anonymity.