Want to Earn a Few Extra Bucks? A First Look at Money-Making Apps

Yangyu Hu[1], Haoyu Wang[1*], Li Li[2], Yao Guo[3], Guoai Xu[1*], Ren He[1]
[1] Beijing University of Posts and Telecommunications, Beijing, China
[2] Monash University, Melbourne, Australia
[3] Peking University, Beijing, China

***Abstract*—Have you ever thought of earning profits from the apps that you are using on your mobile device? It is actually achievable thanks to many so-called money-making apps, which pay app users to complete tasks such as installing another app or clicking an advertisement. To the best of our knowledge, no existing studies have investigated the characteristics of money-making apps. To this end, we conduct the first exploratory study to understand the features and implications of money-making apps. We first propose a semi-automated approach aiming to harvest money-making apps from Google Play and alternative app markets. Then we create a taxonomy to classify them into five categories and perform an empirical study from different aspects. Our study reveals several interesting observations: (1) money-making apps have become the target of malicious developers, as we found many of them expose mobile users to serious privacy and security risks. Roughly 26% of the studied apps are potentially malicious. (2) these apps have attracted millions of users, however, many users complain that they are cheated by these apps. We also revealed that ranking fraud techniques are widely used in these apps to promote the ranking of apps inside app markets. (3) these apps usually spread inappropriate and malicious contents, while unsuspicious users could get infected. Our study demonstrates the emergency for detecting and regulating this kind of apps and protect mobile users.**

***Index Terms*—Money-making apps, Mobile Security, Android, Malware, Mobile Ads, Pay-Per-Install**

I. Introduction

Making money online has become some kind of part-time job or even specialized profession in recent years [8], which sometimes becomes even extremely lucrative for full-time freelancers. People can make money anywhere by completing short online tasks (e.g., downloading apps, filling in online surveys), which are issued by merchants on promotion platforms [12], such as Swagbucks [15] and Amazon Mechanical Turk [3]. Taking Mechanical Turk as an example, it is an online platform that leverages crowd-sourced human intelligence to perform tasks that cannot be automatically done by computers. As a result, many academic studies have performed crowdsourcing studies on Mechanical Turk. With the prevalence of smartphones, we have observed a trend that the promotion platforms start to also emerge in mobile apps.

In this work, we refer to such apps that provide promotion platforms for app users to earn profits as "**money-making apps**". App users can install such apps from various resources including app markets and dedicated websites, which are specifically designed to share money-making apps. Once money-making apps are installed, app users usually need to register an account and fill in some personal information such as email address, phone number, bank account, etc., in order to use the app to earn profits.

Fig. 1 illustrates a typical usage example of a money-making app called *MiZhuan* (me.mizhuan). When logging into the app, a list of tasks (including the meta-data of the task such as task names and rewards) is presented to the user (cf. Fig. 1(a)). App users can pick up a task and review more details about it. For example, as demonstrated in Fig. 1(b), the task asks users to install a given app and register an active account for the app. Towards verifying if the task is properly completed (cf. Fig. 1(c)), money-making apps may also adopt some monitoring techniques to verify the process (i.e., the app is indeed downloaded, installed and an account is indeed registered). If a task is successfully completed, the money-making app will reward the users by paying rewards to the user's account (cf. Fig. 1(d)). Finally, as shown in Fig. 1(e), users can cash out the earned profits (i.e., exchanging the virtual currency with real money).

As demonstrated in the aforementioned example, in order to pay app users, money-making apps have legitimate reasons to ask for users' personal information such as phone numbers and bank accounts. Such information is very sensitive to users, and if leaked, can introduce serious troubles to users, e.g., the phone number can be used to conduct phone fraud. Besides, these apps usually request many sensitive permissions, in order to verify whether the task is completed. Hence, attackers have incentives to deliver malicious apps, pretending itself as money-making apps, to harvest users' sensitive information.

Moreover, not only the implementation of the app code, but also the content disseminated during the execution of this kind of apps might also be problematic. Indeed, the core business model of money-making apps is to promote "things" (e.g., advertisements or apps), which could be very diverse and cover different fields, while are also needed to be continuously updated. Unfortunately, there is no guarantee that the content disseminated by these money-making apps is decent or even legal. Devious people may leverage money-making apps to deliver undesired or illegal messages.

To the best of our knowledge, despite a lot of research efforts on the Android ecosystem, none of the existing studies have studied money-making apps, either from the code

*Co-corresponding authors

978-1-7281-0591-8/19/$31.00 © 2019 IEEE

(a) (b) (c) (d) (e)

Fig. 1. An example of a money-making app.

implementation perspective or from the content disseminated point of view. To fill the gap, in this paper, we conduct the first exploratory study of money-making apps, aiming at observing actionable insights that could lead us in regulating a better ecosystem of money-making apps. Based on a semi-automated approach, we have identified 1,377 money-making apps from a population of 2.5 million apps crawled from Google Play and Chinese alternative markets (Section II). We first create a taxonomy of these apps to understand their monetizing schemes (Section III). Then, we take a user-centric study to understand how these apps are adopted by mobile users, and how users feel (or complain) about them (Section IV). To illuminate the security behaviors of these apps, we investigate them from requested permissions, embedded third-party tracking services, and the presence of malware (Section V). Finally, we investigate the contents disseminated by these apps to identify inappropriate and malicious contents (Section VI).

The main research contributions are summarized as follows:

- **Money-Making app identification and taxonomy.** We propose a semi-automated approach to first learn the characteristics of money-making apps, and then perform identification from a corpus of 2.5 million Android apps. Eventually, we have identified 1,377 money-making apps that are further classified into five main categories based on their monetizing schemes.
- **Ranking fraud and user complaints.** Normally, money-making apps are quite popular across different markets. However, our further investigation discloses that ranking fraud techniques are widely used in them to create fake user reviews in order to promote their app rankings. Besides, many real users complain that they are cheated by them or report security concerns.
- **Security and privacy behaviors.** The number of sensitive permissions requested by money-making apps is significantly more than those by other apps. Third-party tracking libraries are prevalent in these apps, while more than 25% of them are identified as potential malware.
- **Disseminated contents.** The contents promoted on money-making apps are not always trustworthy. Our empirical investigation results reveal various bad contents such as malware, inconsistent content and content with aggressive ads.

II. Experimental Setup

In this section, we seek to focus on our investigation on four research questions. We then describe our approach for identifying and characterizing money-making apps.

A. Research Questions

Our empirical study is mainly focused on the following four research questions:

- ***RQ1: What are the typical monetizing schemes in these money-making apps?*** There might exist different types of money-making apps that support mobile users to earn profits, while each type can adopt different business models to entice users to download and use the apps. Towards understanding the working process of money-making apps, there is a need to summarize their possible monetizing schemes.
- ***RQ2: How do money-making apps attract app users?*** The main rationale behind money-making apps is to entice app users (by paying money to them) to download and use the apps in order to achieve some purposes such as displaying advertisements to users. Therefore, we need to understand if these apps can indeed generate interests for

app users. If so, are app users happy about their experience of using these apps?

- ***RQ3: How secure are the money-making apps?*** Because money-making apps are involved with money, which may attract the interests of attackers to exploit these apps and their app users who are interested in earning profits. Besides, these apps have legitimate reasons to ask for users' personal information, it is important to understand whether they respect users' privacy. Therefore, there is also a need to characterize the security and privacy aspects of money-making apps.
- ***RQ4: What are the contents disseminated over money-making apps?*** Finally, we are interested in the contents disseminated over the money-making apps. More specifically, since we do not know if the contents promoted on these apps are trustworthy, it is also worthwhile to characterize their disseminated contents.

B. Dataset Collection

In order to answer the aforementioned research questions, we need to harvest a set of money-making apps. However, to the best of our knowledge, money-making apps have not yet been explored in existing studies, so there is no existing dataset we can utilize or compare with. Furthermore, the characteristics of money-making apps have been generally unknown as well, which could be potentially useful for harvesting money-making apps from the wild. To this end, we propose a heuristic-based approach to harvest money-making apps, in order to fill the research gap in this kind of study.

Fig. 2 presents the working process of our approach for identifying money-making apps. First, we collect money-making apps manually using specific keywords on online search engines including Google and Baidu. Based on the collected apps, we leverage various means (e.g., static analysis and text analysis) to manually understand the collected apps aiming to summarize exclusive characteristics related to money-making apps. Then, we check the summarized characteristics against large-scale market apps, in order to find more money-making apps. We now give more details about these two processes.

1) Learning the Characteristics of Money-Making Apps: To identify money-making apps from millions of apps in the app markets, we resort to search engines (including Google and Baidu) to search keywords such as "making money" and "Android app" (in both English and Chinese). We have collected 41 websites/webpages that publish money-making apps [2]. Then, we write a crawler to collect all the apps (.apk files), as well as the corresponding metadata. In total, we have obtained a set of 708 unique money-making apps.

We then manually inspect the metadata and the decompiled code of these crawled money-making apps. We have observed three main characteristics shared in these apps, which could be helpful to identify more money-making apps from a large number of apps hosted on different app markets. (1) ***Attractive keywords.*** To attract mobile users to identify and download them, these apps usually include attractive common keywords in app names and app descriptions. Thus, we have summarized a list of 10 keywords that represent the most popularly used ones in money-making apps[1]. (2) ***In-app payment services.*** All of these apps have embedded in-app payment services, which could be used by mobile users to exchange virtual currency that they earned in the app into real money. (3) ***Specific user comments.*** We also found that this kind of apps usually has representative user comments including keywords. Thus, we also summarized top 5 representative keywords in user comments for these apps[2].

2) Identifying Money-making Apps in Large-scale: Based on the three characteristics we have summarized, we propose to identify more money-making apps from a large number of apps hosted in app markets. We have collected more than 2.5 million apps from 10 Android app markets, including the official Google Play market and 9 popular Chinese alternative app markets. We crawled the APK files and the metadata of these apps, including app name, publisher name, app version, rating, the number of downloads, etc. All of these apps were downloaded between January and April 2018.

Then, we use a *fast keyword matching method* on the app metadata (e.g., app description and app name) to identify potential money-making app candidates. Eventually, we are able to identify 1,896 apps (out of 2.5 million apps) that contain at least two keywords we summarized. We further perform static analysis on the selected candidate apps to filter apps without in-app payment services. We take advantage of LibRadar [43], an open source obfuscation-resilient tool to *identify third-party libraries used in Android apps*. We have manually labelled a list of 18 different in-app payment services that are widely used in both China and worldwide[3]. Among them, 1,720 apps have embedded in-app payment services, which will be used in the next stage.

At last, we verify these apps by analyzing their user comments. For the apps that have at least one comment that contain any of the five keywords we summarized, we will flag it as a money-making app. In the end, we are able to eventually collect 1,377 money-making apps. It is worth mentioning that all the 708 apps we crawled from the websites in the first step can all be found in app markets.

III. Monetizing Schemes of Money-Making Apps

Given that money-making apps provide different monetizing schemes, we aim at categorizing them according to the monetization-related functionalities provided to mobile users. Thus, we first investigate the distribution of categories in which the money-making apps are displayed in markets. The result is shown in Fig. 3(a). Unfortunately, categories in app markets (e.g., tools and video) are too broad to capture the actual purpose of the apps. The categories of these money-making

[1] 10 keywords (in both English and Chinese): make money online, easy money, leisure time, fragmented time, complete task, share article, watch ads, app tryout, win cash, bonus

[2] 5 keywords (in both English and Chinese): make money, leisure time, bonus, withdraw money, exchange money

[3] 18 in-app payment service: AliPay, WechatPay, BaiduPay, Paypal, YeePay, Ping++, JieshenPay, IPayNow, LianlianPay, UnionPay, MengPay, PayEco, SwiftPass, Juhe, JubaoPay, 99Bill, IAppPay, BBNPay

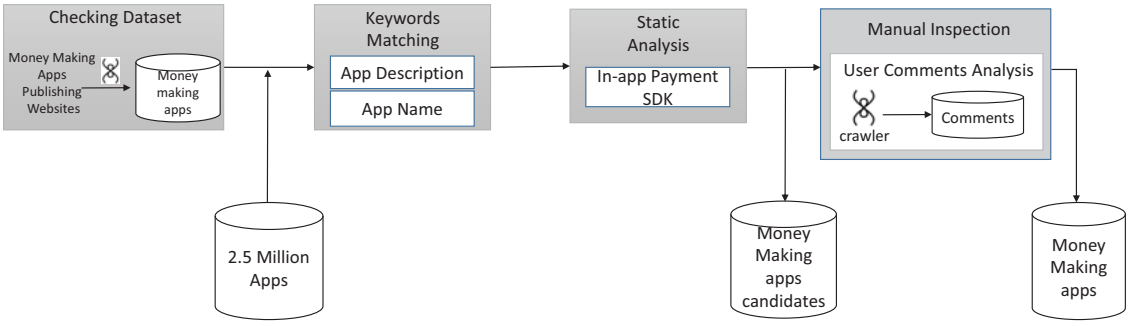
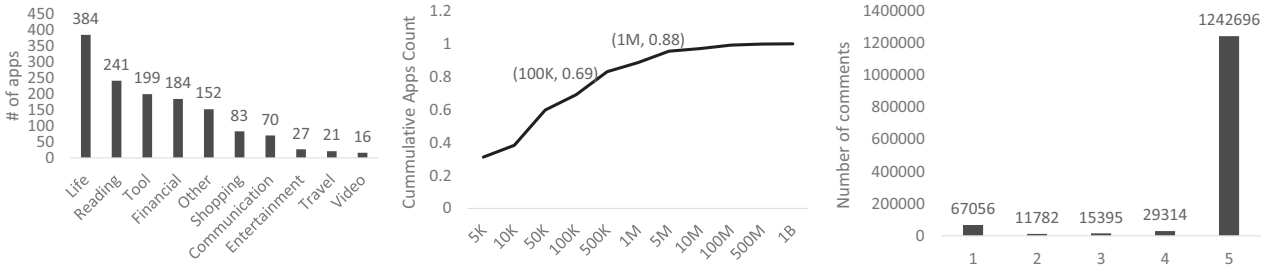


Fig. 2. Our approach to identify money-making apps.



(a) Distribution of App Categories in the Market (b) Distribution of App Installs (c) Distribution of App Rating

Fig. 3. Distribution of app categories, app installs and app ratings for the money-making apps.

TABLE I
A TAXONOMY OF MONEY-MAKING APPS.

Category	Number of Apps	% of Apps
Content Sharing	793	58
Pay-Per-Install	386	28
Shopping and Cash Back	113	8
Cryptocurrency mining	60	4
Crowd-sourcing	101	7
Total	1,377	-

apps diverge greatly. For example, "Life", "Reading" and "Tool" are the top 3 categories that account for the most number of money-making apps. However, by manually inspecting the apps, we observe that most of their core functionalities are not related to the categories labelled in the market. This result suggests that the lack of enforcement and supervision over the metadata provided by app developers in the app markets.

A Taxonomy of Money-making Apps. To create a taxonomy of money-making apps, we rely on manual exploration. We first randomly sample 200 apps and play with them on real Android devices. We manually analyze the descriptions related to the tasks demonstrating the detailed steps that users need to follow in order to earn profits. Based on the characteristics of the observed tasks, we empirically summarize the apps into five categories. We further explore the unique features that could be used to classify the remaining apps. For each app, we apply the TF-IDF technique on app description and app title to generate a word vector. For each category, we manually summarize several keywords related to this category based on the word vectors that belong to this category. At last, we use a keyword matching method to check the descriptions of all the remaining apps to classify them into different categories. Note that an app could belong to several categories, as it may provide more than one type of tasks.

Table I illustrates the taxonomy of the summarized categories. Note that one app could belong to multiple categories because it may embed more than one monetizing schemes. Interestingly, more than half of them belong to the category of "Content Sharing". We then describe each category in detail.

(1) Content Sharing: The task in this kind of money-making apps is to ask mobile users to share specific articles (or videos) to their friends via popular social networking platforms. Mobile users could get more reward if the shared contents are viewed by more people. For the business model of this kind of apps, the merchants can be either the app developers who aim at improving the exposure of their apps using this incentive method, or the mobile advertisers, who want to advertise their contents using this approach (it will be explained in detail in Section VI.B). In general, several reports [9] [10] [11] have disclosed that mobile users could get 0.01-0.03 dollar equivalent reward for each time their shared articles were read.

(2) Pay-Per-Install (PPI): There are millions of apps in each app market. To improve rating and installs, many developers rely on incentive installs, which could help boost app ranking. This kind of money-making apps offers the functionality for app developers to promote their apps. Pay-per-Install [34] is the payment methods to count the price for each install, while the app developers are buying a huge number of installs. Hence, the task is generally asking the users to download, install and use the apps that are promoted in this kind of apps. In general, a mobile user could get 0.3-0.5 dollar equivalent reward once they complete a task in this kind of PPI apps.

(3) Shopping and Cash Back: Such money-making apps claim to help users save money or earn profits while shopping. There are generally two approaches: One is to provide cash-back deals when users are shopping, while the other is to ask users to share the merchandises (e.g. clothes, shoes) links to their friends that are displayed in money-making apps. Once their friends make a purchase on the merchandises they shared, the user will get a specific amount of reward.

(4) Cryptocurrency Mining: Cryptocurrency mining apps generate incremental Bitcoin, Litecoin, Ethereum, and other virtual digital currencies in the background for mobile users by consuming the computing resources of their smartphones. It is worth to mention that Google Play started to ban crypto-mining apps since July 2018 [17]. We have crawled 60 cryptocurrency mining apps, while 6 of them are crawled from Google Play before they are removed.

(5) Crowd-sourcing: These apps provide money-making tasks including a wide range of topics, such as online surveys, taking appointed photos, writing product reviews, and answering questions such as brand awareness, etc. In general, each task is worth roughly $0.5 to $5.

IV. Understanding Money-Making Apps from App Users' Perspective

In this section, we take a user-centric perspective to understand how money-making apps are adopted by mobile users, and how users feel and complain about them.

A. The Number of Potential Users

We first analyze the number of user installs for these apps to understand the scale of potential users. Note that for apps released to multiple markets, we calculate the accumulated installs for each of them.

Fig. 3(b) shows the overall distribution of app installs for all the 1,377 apps. It is obvious that money-making apps are quite popular across markets, and more than 31% of them (426 apps) have accumulated installs higher than 100,000. Around 11.4% of them (157 apps) have the number of downloads over 1 million. The most popular app is "com.coohuaclient", which is a famous Content Sharing app that has aggregated more than 200 million installs. This result indicates that it is a trend that users are willing to use these mobile promotion platforms.

B. User Comment Analysis

We then seek to understand the feelings and complaints (e.g. the privacy and security issues) of users by applying text analysis techniques to user comments.

User Comment Collection. We have crawled the user comments and ratings of these apps in each market as well as the IDs of users who have posted the reviews. For the 1,377 apps, we have crawled 1.36 million user comments in total, each app has 992 user comments on average. We use the user IDs listed on app markets to distinguish mobile users, which leaves us 572,667 unique users in total. The distribution of app ratings is shown in Fig. 3(c). It is surprising to see that most money-making apps receive high user ratings: more than 90% of user comments are 5-star ratings. Thus, we further study user comments to explore the underlying reasons.

Empirical Findings. We first manually analyzed 100 positive comments (with 5-star rating) and 100 negative comments (with 1-star or 2-star rating). We have obtained several interesting results. First, we found that 95% of the positive comments examined have repeated multiple times in the crawled user comment dataset, which indicates that the user reviews may be manipulated, in order to promote the app rankings in markets. Besides, although negative reviews only account for 4.9% of the user comment dataset, we found many users complain about security issues and fraudulent behaviors in these apps. Thus we will further explore user comments in two aspects: *Ranking Fraud Analysis* and *User Complaint Analysis*.

C. Ranking Fraud Analysis

Ranking fraud [24] refers to the behaviors that aim to promote the ranking of apps inside app markets. we distinguish whether the reviews are manipulated by identifying *fake reviews* and *fraudulent reviewers*. Specifically, our analysis is based on the following heuristics: *reviews from different users should be different in most cases*. Though some simple reviews such as "great app" could be posted by different users, other reviews with more meaningful words should not be exactly the same. Based on this heuristic, our analysis works in the following steps and the overall result is shown in Table II.

First, we remove the reviews that have fewer than 5 words from our analysis to avoid potential false positives introduced by simple reviews. The number of reviews, and reviews with the highest rating (five-star) are shown in the second and third column. We also calculate the number of users who have posted the reviews in the eighth column. Second, we compare the similarity of the reviews from different users using exact text matching. If we find that reviews from different users are exactly the same, we classify such reviews as repeated fake reviews (cf. the fourth column in Table II). The users who have posted fake reviews are classified as fraudulent reviewers correspondingly (cf. the ninth column in Table II). Third, we further mark all the reviews from fraudulent reviewers as fake reviews, which is shown in the fifth column. This step is necessary because the criteria used to determine repeated reviews is too strict (exact text matching), and hence may have missed reviews with only little changes, e.g., from the sentence "This is really a good app" to "This is really an excellent app". By adding all the reviews from users who have posted fake reviews, we could cover the reviews that may be otherwise missed in the previous step. At last, we analyze the percentage of fake reviews and fraudulent users, as shown in the sixth and last column in Table II. We also calculate the percentage of five-star ratings of fake reviews (the seventh column).

The percentage of fake reviews are surprisingly high, where over 70% of reviews in 4 categories are fake and more than 95% of the user ratings in the fake reviews are five-star, demonstrating that the ranking system of app markets is actively manipulated by the publishers of those apps.

TABLE II
RANKING FRAUD BASED ON USER REVIEWS.

Category	# Reviews	Reviews # with Five-star Rating	# Repeated Reviews	# Fake Reviews	% Fake Reviews	Fake Reviews % with Five-star Rating	# Users	# Fake Users	% Fake Users
Content Sharing	715,771	636,941	443,462	531,465	74.25%	93.73%	437,655	282,619	64.57%
Pay-Per-Install	487,950	447,342	289,216	367,917	75.40%	94.54%	290,608	188,312	64.80%
Shopping and Cash Back	341,598	321,605	275,859	299,353	87.63%	95.99%	104,657	66,247	63.30%
Cryptocurrency Mining	4,311	4,074	3,095	3,570	82.81%	95.57%	3,123	2,547	81.56%
Crowd-sourcing	79,291	71,565	36,985	50,543	63.74%	97.27%	58,343	33,659	57.69%
Total	1,366,243	1,242,696	907,917	1,048,117	76.71%	95.53%	697,093	424,180	60.85%

TABLE III
TOPICS MINED FROM NEGATIVE REVIEWS.

Complaint Category	Most Representative Words
Exchange Service	liar, exchange, fake, customer service, commission, money, cheat
Bugs	register, verification code, update, compatibility, stuck
Crashes	crash, start, failure, login in, flashback
Privacy concerns	steal, account, password, information
Malware reports	malware, virus, cheat

Fig. 4. Top keywords in the 67,056 negative reviews.

D. User Complaint Analysis

We further study the negative reviews (with 1-star rating), because our empirical study suggested that real users may complain about security issues in the negative comments.

Approach. We first classify the 67,056 negative user reviews into different categories via topic modelling (more specifically the Biterm Topic Model (BTM) [56]), which is a widely used clustering algorithm for short text. Our implementation feeds output of NLP pre-processing (including word segmentation, removing stop words). We could freely choose the number of topics to be identified by BTM, and we empirically choose 5 during our experiment. Table III shows the results of topics mined from the 67,056 negative reviews that we analyzed. The "complaint category" is the abstract concept we assigned to that topic. We further generate the word cloud [18] for the negative reviews as shown in Fig. 4.

Surprisingly, around 30% of the user comments (19,534 reviews) complain that they have been cheated by a failure of cash out service. A number of user comments (6,116 reviews) report bugs and crashes, and some comments (966 comments) complain about the security and privacy concerns.

V. SECURITY ANALYSIS OF MONEY-MAKING APPS

In this subsection, we perform a general security analysis on the money-making apps. We first investigate how money-making apps request Android permissions to access sensitive information. Then, we analyze the embedded third-party tracking services in these apps. At last, we rely on VirusTotal [16], a widely used anti-virus service to identify money-making apps that may include malicious behaviors.

A. Permission Analysis

We first investigate how money-making apps request sensitive permissions. For each app, we extract the requested permissions from AndroidManifest.xml. Fig. 5 compares the permissions requested by them with those requested by the remaining 2.5 million free non-money-making Android apps, which we included for reference. We listed the top permissions that have been requested by more than 10% of apps, which includes 23 system permissions.

We observed that money-making apps typically request significantly more sensitive permissions than other apps. For 13 out of the 23 sensitive permissions, the percentage of permission used in money-making apps is more than twice than that of remaining apps. For instance, over 64% of money-making Apps request "ACCESS_COARSE_LOCATION" and "ACCESS_FINE_LOCATION", while the percentages in the 2.5 million apps are only 25% and 26%, respectively.

We further analyzed the rationale behinds this phenomenon and found that several permissions are related to task monitoring in these apps. For example, some tasks in PPI apps require users to download a game app, register an account and play the apps for a certain time (e.g., 10 minutes). To monitor whether the users have fulfilled the tasks, some PPI apps get the top activity name of the activity stack through real-time monitoring, to record the duration of using the specific app. This operation is very sensitive and protected by "GET_TASKS" permission before Android L (version 5.0). As a result, more than 67% of money-making apps request "GET_TASKS" permission, while the percentage for the 2.5 million apps is only 25%.

It is also noticeable that some permissions listed in Fig. 5 are totally unusual/unnecessary for money-making apps. For example, more than 40% of the money-making apps request "READ_LOG" permission (versus 10% in other apps), which is flagged as a very sensitive permission, because app developers

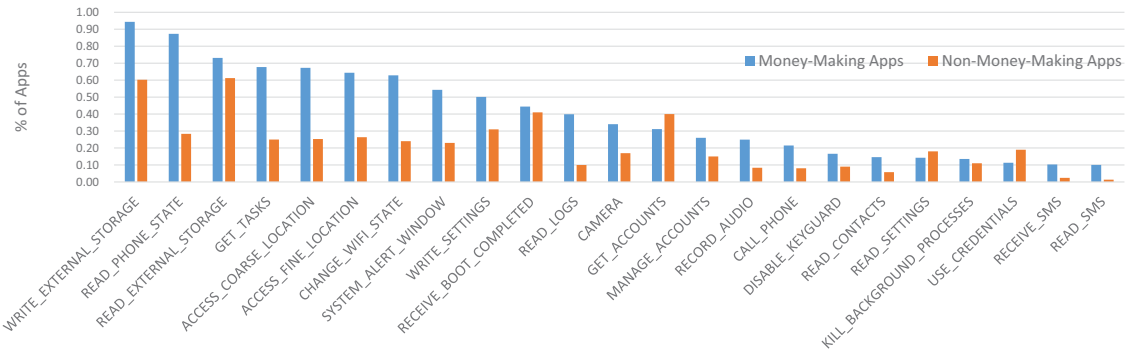


Fig. 5. Detailed comparison of Android permissions (x-axis) requested by money-making apps and the remaining 2.5 million free non-money-making apps.

may carelessly misuse Android's logging capabilities and (unintentionally) expose personal information (e.g., credit card and password) to any other apps requesting it. Besides, more than 10% of money-making Apps request RECEIVE_SMS and READ_SMS permissions (versus 2% and 1% in other apps): one possible explanation to this is because money-making apps may want to send SMS for verification and other purposes and read SMS automatically.

B. Third-party Tracking Services

Third-party services form an integral part of the mobile ecosystem: they ease app development and enable features such as analytics, social network integration, and app monetization through advertisements. However, previous work suggested that these services are largely invisible to users, which may cause potential security and privacy risks [21], [40].

In this work, we take advantage of LibRadar [43], an obfuscation-resilient tool to identify embedded third-party libraries. More than 24% of the money-making apps embed at least 20 third-party libraries in their source code. In particular, two money-making apps (com.zqcall.mobile and com.tonglu.survey) have the highest number (49) of embedded libraries. This number is much higher than that in non-money-making apps suggested by one recent work [54].

We then investigate the most popular third-party tracking services used in these apps. We have collected a list of well-labelled tracking services from PrivacyGrade [14], Seneviratne *et al.* [46] and Li *et al.* [37]. We have identified 120 tracking services used in these apps, while more than 82% of them have embedded tracking services. "Umeng" and "Youmi" are the most popular trackers among our corpus of money-making apps, with more than 31% (429 apps) and 8% (112 apps) of apps embed them, respectively. Besides, we also found that several well-known aggressive offer-wall advertisement services [1], [13] are used, including "BillionMobi" (99 apps), "Dianmoney" (95 apps), "Datouniao" (90 apps) and "WAPS" (74 apps). The offer-wall ads are typically a page in an app with a number of offers and incentives (e.g., downloading apps to earn game props). For the worst case, mobile users cannot close the ad page and they have to download the promoted apps in order to continue using the current app. These aggressive advertisement services would greatly affect user experiences.

TABLE IV
DISTRIBUTION OF POTENTIAL MALWARE IN EACH CATEGORY.

App category	AV-rank (# apps) (% apps) ≥ 1	≥ 10	≥ 20
Content Sharing	477(0.60)	**136(0.17)**	38(0.05)
Pay-Per-Install	316(0.82)	**210(0.54)**	62(0.16)
Shopping and Cash Back	69(0.61)	24(0.21)	8(0.07)
Cryptocurrency Mining	52(0.87)	8(0.13)	2(0.03)
Crowd-sourcing	68(0.67)	18(0.18)	6(0.06)
Total	917(0.67)	356(0.26)	100(0.07)

C. Malware Presence

At last, we explore the presence of malware in the collected money-making apps. We uploaded all the apps to VirusTotal [16], an online analysis service that aggregates more than 60 anti-virus engines, which is widely adopted by the research community. Previous studies [19], [55] have suggested that some anti-virus engines may not always report reliable results. In order to deal with such potential false positives, we analyzed the results grouped by how many engines (AV-rank) flag an app as malware. Previous work [32], [60] have suggested that a threshold of 10 engines can be regarded as a robust choice.

Overall Results. Table IV shows the overall detection results. Remarkably, around 67% (917 apps) of the collected apps are flagged by at least one anti-virus engines. When using the threshold of "AV-Rank>=10", around 26% of the apps (356 apps) are labelled as malware. Table V lists the top 5 malware according to their AV-Rank. For example, the app "com.media" was flagged by 32 anti-virus engines ("gappusin" family), which is actually a trojan that steals sensitive information [4].

We also noticed that some malware can even attract millions of installs. For example, app "com.change.unlock" has over 54 million installs. The total number of installs for these 356 potentially malware achieves 1.7 billion across 10 app markets. This is a large number, cause the total number of installs for all the apps in Google Play is around 193 billion according to a recent study [54]. Thus, the number of installs for these malicious apps roughly equals to 1% of total installs of Google Play. The result suggests that many unsuspicious users may indeed be exposed to the threats introduced by these malicious "money-making apps".

TABLE V
TOP 5 MALICIOUS APPS ACCORDING TO AV-RANK.

Package_name	MD5	app catogory	AV-Rank	malware family
com.media	9eeae7883a930731dbd454dfbd3aef27	pay-per-install	32	gappusin
com.kuaizhuan.omgorgtwob	23cc77cba667a1ed5ca1b1c4c3888c96	pay-per-install	29	youmi
com.haojiao.liuliang	abe86306450d7baf2b7ab7585b04ef34	sharing;pay-per-install	28	smsthief
com.dou.zhuandou2	02ba039548a017690ec173bcb68754d4	shopping	27	youmi
com.terry.makemoney	9a9074cc54c23f8ab247dca06eac101c	pay-per-install	27	youmi

Fig. 6. An example of Article Sharing (app: com.tiantiankandian.mm).

Malware Category and Malware Family. We then analyze the distribution of malware categories and families. The malware signatures for these apps mainly correspond to 5 different type of malware: Trojan (38.4%), Adware (36.2%), Malware (22.1%), Riskware (11.4%) and Spyware (1.2%). We then use AVClass [45] to obtain the family name (label) of each identified malware. The "youmi" and "gappusin" families are most popular, more than 42% (148 apps) and 16% (56 apps) of flagged malicious apps belong to them.

Malware Distribution across Different Types of Money-Making Apps. As shown in Table IV, most of the flagged malicious apps (86%) belong to "Content Sharing" and "Pay-Per-Install" categories, which suggests that malicious developers prefer to use this way to attract unsuspicious users.

VI. ANALYSIS OF DISSEMINATED CONTENTS

Considering that malicious behaviors and negative comments are mostly focused on the apps of the first two categories ("Content Sharing", "Pay-Per-Install"), as the last research question, we hence study the contents disseminated by the apps of these two categories. More specifically, we would like to answer the following two research questions:

- RQ4.1: Do *Content Sharing Apps* keep their promise to share consistent contents that are observed by app users?
- RQ4.2: Can we trust the apps installed via *PPI* apps?

A. Consistency in Browsed and Shared Contents

The mission of "content sharing" is to ask users to share the in-app articles to their friends and then reward users with virtual currency according to the reading times of the shared articles (i.e., impressions in mobile advertising).

Ideally, the contents that are actually shared via *"content sharing" apps* should be the same as what the users intend to share. While using social networking platforms such as Wechat, the authors have frequently observed that many shared articles are very annoying, which may contain a lot of ads and other undesirable contents. In the worst situation, the shared content contains all ads without any useful texts. It is kind of weird that users are willing to share such articles with their friends. Hence, we study if such annoying articles are shared via *"content sharing" apps*.

Field Study. We randomly choose 40 apps out of the 793 "content sharing" apps, and then perform a field study on the shared contents in them by installing them on real smartphones (Nexus 5 and Huawei P9). For each app, we first register an account to log in, randomly select 5 articles, and then share them to friends on different social platforms. Furthermore, for each article, we obtain the two links (before and after sharing), and we crawled content and then further compare them.

Consistency Analysis. For each article (400 articles in total), we obtain their content by leveraging the tool "charles" [5], which is a web debugging proxy. We crawl the text and download all the pictures shown on the page. To identify whether the contents are consistent, we further compare the similarity between the crawled contents. We compute the similarity of texts by calculating the ratio of longest Common sub-string (LCS) using a dynamic programming algorithm, and we use Dup Detector [7], which is a pixel-level comparison technique, to identify duplicate images. We found that, for the 200 article pairs, 98% of them have inconsistent contents, while more than 40% of them have only less than 20% text similarity, and more than 98% of them have included new pictures in the shared links. Interestingly, for one app "app com.xiangzi.wcz", all the shared links are reported to be malicious by both Wechat and VirusTotal.

What has been modified? We observe the differences between two articles in detail and found that all of the articles in the 39 apps add additional ads after sharing. We consider the purpose of this behavior is to mislead users of money-making apps that the articles are ad-free so that they are more willing to share the articles. For 30 apps among them, they add additional ads into various positions of the articles, including top (before the title), middle, as well as the bottom of articles (after the content). As shown in Fig. 6, the article add ads in the three different positions after sharing. Meanwhile, we found that another 9 apps add additional ads by URL redirecting. They guide users to the pages that are completely unrelated to the actually article content. The shared links of

articles in 6 apps are redirected to webpages of the domain `https://cpu.baidu.com`, which are full of ads through our manual confirmation. The shared links of articles in another app are redirected to a promotion page of WeChat subscription (https://m.baozoukanshu.com) and the others are redirected to app downloading pages.

B. Malware Analysis of the Promoted Pay-Per-Install Apps

The mission of PPI apps usually asks users to download or register an app, more advanced apps may require users to use the app for a certain time or complete particular in-app tasks (e.g. get five wins in a game app). Since the recommended way to install mobile apps is via app markets, we are interested in the reason why developers want to promote their apps via *Pay-Per-Install* platforms. Is it because their apps are malicious ones that are not allowed to enter app markets?

UI Exploration based Approach. In order to answer the aforementioned question, we design an approach to automatically play with *Pay-Per-Install* platforms so as to harvest as much as possible apps promoted via these platforms. To this end, we propose an automated testing approach to achieve this purpose. As money-making apps will help users download the promoted apps automatically when starting a task, we are able to obtain the apps via simulating the process of completing tasks. However, existing automated testing methods is limited to low UI coverage so that it can hardly extract all the apps [62] [20], while it is also time-consuming to write test scripts for each app manually. Therefore, we propose an improved semi-automated script generation method to improve the efficiency and accuracy of app extraction.

Specifically, our method is based on the following findings: the operating process of downloading apps in each PPI app are similar (click the widget of a task in the task list, click the install button, go back and continue to choose another task). To convert the process of downloading app into the corresponding testing script, for each money-making app we just need to obtain two values: *the coordinate of the first task in the task list*, and *the height of task widget*. As the height of each task widget is the same, we can calculate a coordinate in each task widget in the UI directly by linear superposition. Furthermore, to ensure that all apps will be downloaded, we predefined a starting coordinates and ending coordinates to help sliding.

Crawling the Promotion Apps. For each of the Pay-Per-Install apps in our dataset (386 apps), we generate the testing script, and then convert the script into the "input" operation command [6], which is provided by "Android command-line" and is used to implement simulating interaction with UI. Eventually, we extract more than 9,000 apps disseminated over 386 PPI apps, however, they include only 476 unique apps (with the same MD5 value). It is interesting to see that different PPI apps promote the same apps, thus we further explore the underlying reason. By manually inspecting several cases, we found that for the PPI apps that promote the same apps, the app promotion UI pages are always similar. By further analyzing the decompiled code, we found that they always use the same Offer-Wall advertising library to promote apps.

TABLE VI
THE DISTRIBUTION OF PPI APPS THAT EMBEDDED OFFER-WALL ADS.

Platform	#Apps	Platform	#Apps
Domob	98	WQMobile	1
Youmi	7	AppDriver	3
Limei	36	Mobsmar	25
Guomob	37	Juzilm	1
Yijifen	13	TapJoy	3
WAPS	74	Dianmoney	95
Miidi	20	Datouniao	90
BillionMobi	99	-	-

We then analyzed the libraries used in the 386 PPI apps, and found 15 different Offer-Wall advertising libraries embedded in 362 apps, as shown in Table VI. The remaining 24 apps have implemented app promotion mechanisms themselves. Note that some advertising libraries provided different forms of ads besides offer-wall (e.g., banner ads and interstitial ads), thus for each of the 15 libraries, we have summarized the unique characteristics (e.g., APIs and Strings) based on the library document to identify whether the apps use offer-wall ads to promote apps. For example, as we mentioned in Section V, 112 apps embedded Youmi ad libraries, while only 7 of them use offer-wall ads, as listed in Table VI.

Malware Presence We upload all the 476 apps that are disseminated over money-making apps to VirusTotal to examine how many of them are flagged by existing anti-virus engines. Experiment result suggests that 56% of the downloaded apps are labelled as malware by at least 1 anti-virus engine. When using the threshold of "AV-Rank $>=$ 10", around 12% of the downloaded apps are labelled as malware by more than 10 anti-virus engines. Remarkably, 2 of them ("hqa.sj183d.djin" and "cpqi.j002sil.jstu") are labelled by more than 35 engines. More seriously, we found some malicious apps that are downloaded multiple times during our study.

Malware Family Distribution. We then analyze the distribution of malware families labelled by VirusTotal. The malware signatures for these apps mainly correspond to 5 different type of malware: PUA/PUP (36.3), Trojan (26.5%), Adware (21.6%), Riskware (13.7%) and Smsreg (10.9%). We then use AVClass [45] to obtain the family name (label) of each identified malware. "singleton" and "smspay" are the most popular malware families, more than 34% (164 apps) and 8% (37 apps) of flagged malicious apps belong to them.

Existence in App Markets. In addition, we check whether these apps are published in Google Play and 9 popular Chinese app markets (as shown in Table 1). We use package name matching and found that 103 apps (21.6%) cannot be matched, which indicates that these app are not released to these markets or they were removed by these markets. We further manually analyze these apps, and found that 50 of them are labelled as malware with "AV-Rank $>=$ 10", 8 of them are porn apps, and 49 of them are gambling apps. This result suggests that a considerable number of apps distributed in the money-making apps may contain inappropriate content, which also indicates

that most developers of money-making apps may lack necessary security auditing of their in-app contents.

VII. Discussions

In this paper, we present the first explorative study of the ecosystem of money-making apps, and have uncovered various security and privacy issues in these apps. Nonetheless, our work still faces several limitations that could be further improved. First, the method used to detect money-making apps is straightforward and conservative, which may miss some of them. The summarized keywords and the list of third-party payment libraries may not be complete. Second, the taxonomy we created may not be complete, it is quite possible that there are other kinds of money-making apps that are not included in this paper. Third, for the disseminated contents analysis, we only focus on the contents distributed in "Content Sharing" apps and "Pay-Per-Install" apps, because these two kinds of apps have received the most number of user complaints and most potentially malicious apps belong to these two categories. However, it is quite possible that other kinds of money-making apps (e.g., cryptocurrency mining) may also host malicious or unwanted contents. We leave it for future studies.

Our empirical investigation also set forth several implications that should be conducted towards providing a better mobile ecosystem. First, app markets should pay special attention to money-making apps and need to propose regulations to define the boundary of money-making apps so as to keep illegitimate money-making apps from entering the markets in the first place. Second, app markets should also closely monitor user comments of their apps and develop automated approaches to flag manipulated comments and subsequently to purge unethical money-making apps (could apply to other apps as well) from the markets. Furthermore, to improve trustworthiness in the ecosystem, as well as to mitigate the possibility of cheating users, money-making apps should resort to certified third-party platforms to implement their rewarding mechanism. Finally, we also argue that, when detecting malware, anti-virus products need to not only scan the app code, but also investigate the contents manipulated by Android apps.

VIII. Related Work

To the best of our knowledge, the ecosystem of money-making apps has not yet been investigated. Nevertheless, various studies have explored the security and privacy aspects of mobile apps, as well as the general mobile app ecosystem.

A. Security and Privacy Analysis of Mobile Apps

A large mount of studies have analyzed mobile apps from security and privacy aspects [35], [38], including malware detection [19], [28], [57], permission and privacy analysis [41], [42], [50]–[52], repackaging detection [22], [29], [39], [49], [61], privacy leakage identification [26], [27], [36], [59], and identifying and analyzing third-party libraries [21], [25], [40], [43], [52], etc. The most related studies to this paper is analyzing financial apps and payment services. Taylor *et al.* [47] analyzed more than 10K financial apps to understand how they have evolved in terms of sensitive permission usage and security vulnerabilities. AUSERA [23] was proposed to identify security weaknesses of banking apps. Yang *et al.* [58] investigated current third-party mobile payment ecosystem and propose to detect the violations of security rules in Android apps.

B. Understanding the Mobile App Ecosystem

1) Market-level Measurement: PlayDrone [48] performed a large-scale characterization of 1.1 million apps published in Google Play, including app evolution analysis and authentication scheme risks. Wang *et al.* [53] analyzed the mobile app ecosystem from the perspective of app developers. Ishii *et al.* [33] investigated 4.7 million Android apps covering 27 app markets to understand the security management of global third-party markets. Wang *et al.* [54] performed a large-scale comparative study that covers more than 6 million Android apps downloaded from 16 alternative markets and Google Play, aim to understand the catalog similarity across app stores and various misbehaviors.

2) The Ecosystem of Specific App Types: Ikram *et al.* [32] measured 283 Android VPN apps to understand security and privacy issues. They also measured the ecosystem of ad blocking apps [31], with similar findings that these apps have embedded third-party tracking libraries to access sensitive resources on users' mobile devices, and the presence of malware. Martínez-Pérez *et al.* [44] analyzed the mobile health apps from the aspect of security and privacy legislation. Hu *et al.* [30] analyzed the ecosystem of fraudulent dating apps, i.e., the sole purpose of these apps is to is to lure users into purchasing premium/VIP services to start conversations with other (likely fake female) accounts in the app.

IX. Concluding Remarks

The increasing number of mobile money-making apps available on app markets and the growing number of user complaints suggest that there exist serious security, privacy and fraudulent issues among this unexplored ecosystem. In this paper, we present the first study towards understanding and characterizing the ecosystem of money-making apps. We first propose a heuristic-based approach to identify money-making apps and create a taxonomy for them, then we explore these apps from various aspects. Our study has revealed various interesting findings, including the presence of ranking fraud, privacy issues, malware presence, inconsistent and malicious distributed contents. We believe our findings have demonstrated the necessity to better regulate this kind of apps and protect mobile users from potential risks.

Acknowledgement

This work is supported by the National Key Research and Development Program of China (grant No.2017YFB0801903) and the National Natural Science Foundation of China (grants No.61702045, No.61772042 and No.61873069).

REFERENCES

[1] What Are Offer Walls? - MobileFOMO, 2013. https://mobilefomo.com/2013/12/offer-walls/.

[2] A List of 41 Money-making App Publishing Website, 2018. https://github.com/shouzhuan/Money-making-App-Publishing-Website.

[3] Amazon Mechanical Turk, 2018. https://www.mturk.com.

[4] Android/Gappusin.A | ESET Virusradar, 2018. https://www.virusradar.com/en/Android_Gappusin.A/description.

[5] Charles, 2018. https://www.charlesproxy.com.

[6] Command-line Tool, 2018. https://developer.android.com/studio/test/command-line.

[7] Dup Detector, 2018. https://www.keronsoft.com/dupdetector.html.

[8] Make Money Online, 2018. https://wahadventures.com.

[9] Money-making App Introduction, 2018. https://www.kozhu.com/info/186.htm.

[10] Money-making App Introduction, 2018. http://www.mdd4.com/info/39.html.

[11] Money-making App Introduction, 2018. http://blog.sina.com.cn/s/blog17a5e3dc20102x8qa.html.

[12] Money Saving Expert, 2018. https://www.moneysavingexpert.com/family/make-money-online.

[13] Offer Wall Advertising Networks, 2018. www.businessofapps.com/ads/offer-wall/.

[14] PrivacyGrade, 2018. privacygrade.org/.

[15] Swagbucks, 2018. http://www.swagbucks.com.

[16] VirusTotal, 2018. https://www.virustotal.com.

[17] Why Google is Removing All Bitcoin Mining Apps on the Play Store, 2018. https://www.newsbtc.com/2018/07/29/why-google-is-removing-all-bitcoin-mining-apps-on-the-play-store/.

[18] WordCloud, 2018. https://www.jasondavies.com/wordcloud.

[19] Daniel Arp, Michael Spreitzenbarth, Malte Hubner, Hugo Gascon, Konrad Rieck, and CERT Siemens. Drebin: Effective and explainable detection of android malware in your pocket. In *NDSS 2014*.

[20] Neamtiu I Azim T. Targeted and depth-first exploration for systematic testing of android apps. In *ACM SIGPLAN Notices*, pages 641–660. ACM, 2013.

[21] Michael Backes, Sven Bugiel, and Erik Derr. Reliable third-party library detection in android and its security applications. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 356–367. ACM, 2016.

[22] Kai Chen, Peng Liu, and Yingjun Zhang. Achieving accuracy and scalability simultaneously in detecting application clones on android markets. In *Proceedings of the 36th International Conference on Software Engineering*, pages 175–186. ACM, 2014.

[23] Sen Chen, Guozhu Meng, Ting Su, Lingling Fan, Yinxing Xue, Yang Liu, Lihua Xu, Minhui Xue, Bo Li, and Shuang Hao. Ausera: Large-scale automated security risk assessment of global mobile banking apps. *arXiv preprint arXiv:1805.05236*, 2018.

[24] Geumhwan Cho, Junsung Cho, Youngbae Song, and Hyoungshick Kim. An empirical study of click fraud in mobile advertising networks. In *Availability, Reliability and Security (ARES), 2015 10th International Conference on*, pages 382–388. IEEE, 2015.

[25] Feng Dong, Haoyu Wang, Li Li, Yao Guo, Tegawendé F Bissyandé, Tianming Liu, Guoai Xu, and Jacques Klein. Frauddroid: Automated ad fraud detection for android apps. In *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 257–268.

[26] William Enck, Peter Gilbert, Seungyeop Han, Vasant Tendulkar, Byung-Gon Chun, Landon P Cox, Jaeyeon Jung, Patrick McDaniel, and Anmol N Sheth. Taintdroid: an information-flow tracking system for realtime privacy monitoring on smartphones. *ACM Transactions on Computer Systems (TOCS)*, 32(2):5, 2014.

[27] Clint Gibler, Jonathan Crussell, Jeremy Erickson, and Hao Chen. Androidleaks: automatically detecting potential privacy leaks in android applications on a large scale. In *International Conference on Trust and Trustworthy Computing*, pages 291–307. Springer, 2012.

[28] Michael Grace, Yajin Zhou, Qiang Zhang, Shihong Zou, and Xuxian Jiang. Riskranker: scalable and accurate zero-day android malware detection. In *Proceedings of the 10th international conference on Mobile systems, applications, and services*, pages 281–294. ACM, 2012.

[29] Steve Hanna, Ling Huang, Edward Wu, Saung Li, Charles Chen, and Dawn Song. Juxtapp: A scalable system for detecting code reuse among android applications. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, pages 62–81. Springer, 2012.

[30] Yangyu Hu, Haoyu Wang, Yajin Zhou, Yao Guo, Li Li, Bingxuan Luo, and Fangren Xu. Dating with scambots: Understanding the ecosystem of fraudulent dating applications. *arXiv preprint arXiv:1807.04901*, 2018.

[31] Muhammad Ikram and Mohamed Ali Kaafar. A first look at mobile ad-blocking apps. In *Network Computing and Applications (NCA), 2017 IEEE 16th International Symposium on*, pages 1–8. IEEE, 2017.

[32] Vallina-Rodriguez N. Seneviratne S. Kaafar M. A. Paxson V. Ikram, M. An analysis of the privacy and security risks of android vpn permission-enabled apps. In *Proceedings of the 2016 Internet Measurement Conference*, pages 349–364, 2016.

[33] Yuta Ishii, Takuya Watanabe, Fumihiro Kanei, Yuta Takata, Eitaro Shioji, Mitsuaki Akiyama, Takeshi Yagi, Bo Sun, and Tatsuya Mori. Understanding the security management of global third-party android marketplaces. In *Proceedings of the 2nd ACM SIGSOFT International Workshop on App Market Analytics*, pages 12–18. ACM, 2017.

[34] C. Kreibich J. Caballero, C. Grier and V. Paxson. Measuring pay-per-install: the commoditization of malware distribution. In *Usenix security symposium*, 2011.

[35] Pingfan Kong, Li Li, Jun Gao, Kui Liu, Tegawendé F Bissyandé, and Jacques Klein. Automated testing of android apps: A systematic literature review. *IEEE Transactions on Reliability*, 2018.

[36] Li Li, Alexandre Bartel, Tegawendé F Bissyandé, Jacques Klein, Yves Le Traon, Steven Arzt, Siegfried Rasthofer, Eric Bodden, Damien Octeau, and Patrick Mcdaniel. IccTA: Detecting Inter-Component Privacy Leaks in Android Apps. In *Proceedings of the 37th International Conference on Software Engineering (ICSE 2015)*, 2015.

[37] Li Li, Tegawendé F Bissyandé, Jacques Klein, and Yves Le Traon. An investigation into the use of common libraries in android apps. In *Software Analysis, Evolution, and Reengineering (SANER), 2016 IEEE 23rd International Conference on*, volume 1, pages 403–414. IEEE, 2016.

[38] Li Li, Tegawendé F Bissyandé, Mike Papadakis, Siegfried Rasthofer, Alexandre Bartel, Damien Octeau, Jacques Klein, and Yves Le Traon. Static analysis of android apps: A systematic literature review. *Information and Software Technology*, 2017.

[39] Li Li, Daoyuan Li, Tegawendé F Bissyandé, Jacques Klein, Yves Le Traon, David Lo, and Lorenzo Cavallaro. Understanding android app piggybacking: A systematic study of malicious code grafting. *IEEE Transactions on Information Forensics & Security (TIFS)*, 2017.

[40] Menghao Li, Wei Wang, Pei Wang, Shuai Wang, Dinghao Wu, Jian Liu, Rui Xue, and Wei Huo. Libd: scalable and precise third-party library detection in android markets. In *Proceedings of the 2017 International Conference on Software Engineering*, pages 335–346.

[41] Jialiu Lin, Shahriyar Amini, Jason I Hong, Norman Sadeh, Janne Lindqvist, and Joy Zhang. Expectation and purpose: understanding users' mental models of mobile app privacy through crowdsourcing. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, pages 501–510. ACM, 2012.

[42] Minxing Liu, Haoyu Wang, Yao Guo, and Jason Hong. Identifying and analyzing the privacy of apps for kids. In *Proceedings of the 17th International Workshop on Mobile Computing Systems and Applications*, HotMobile '16, pages 105–110, 2016.

[43] Ziang Ma, Haoyu Wang, Yao Guo, and Xiangqun Chen. Libradar: fast and accurate detection of third-party libraries in android apps. In *Proceedings of the 38th International Conference on Software Engineering Companion*, pages 653–656, 2016.

[44] Borja Martínez-Pérez, Isabel De La Torre-Díez, and Miguel López-Coronado. Privacy and security in mobile health apps: a review and recommendations. *Journal of medical systems*, 39(1):181, 2015.

[45] et al Sebastián, Marcos. Avclass: A tool for massive malware labeling. In *International Symposium on Research in Attacks, Intrusions, and Defenses*. Springer, 2016.

[46] Kolamunna H. Seneviratne A Seneviratne, S. A measurement study of tracking in paid mobile applications. In *ACM Conference on Security Privacy in Wireless and Mobile Networks*, 2015.

[47] Vincent F Taylor and Ivan Martinovic. A longitudinal study of financial apps in the google play store. In *2017 International Conference on Financial Cryptography and Data Security*, 2017.

[48] Nicolas Viennot, Edward Garcia, and Jason Nieh. A measurement study of google play. In *ACM SIGMETRICS Performance Evaluation Review*, volume 42, pages 221–233. ACM, 2014.

[49] Haoyu Wang, Yao Guo, Ziang Ma, and Xiangqun Chen. Wukong: a scalable and accurate two-phase approach to android app clone detection. In *Proceedings of the 2015 International Symposium on Software Testing and Analysis*, pages 71–82. ACM, 2015.

[50] Haoyu Wang, Yao Guo, Zihao Tang, Guangdong Bai, and Xiangqun Chen. Reevaluating android permission gaps with static and dynamic analysis. In *Global Communications Conference (GLOBECOM), 2015 IEEE*, pages 1–6. IEEE, 2015.

[51] Haoyu Wang, Jason Hong, and Yao Guo. Using text mining to infer the purpose of permission use in mobile apps. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '15, pages 1107–1118, 2015.

[52] Haoyu Wang, Yuanchun Li, Yao Guo, Yuvraj Agarwal, and Jason I Hong. Understanding the purpose of permission use in mobile apps. *ACM Transactions on Information Systems (TOIS)*, 35(4):43, 2017.

[53] Haoyu Wang, Zhe Liu, Yao Guo, Xiangqun Chen, Miao Zhang, Guoai Xu, and Jason Hong. An explorative study of the mobile app ecosystem from app developers' perspective. In *Proceedings of the 26th International Conference on World Wide Web*, WWW '17, pages 163–172, 2017.

[54] Haoyu Wang, Zhe Liu, Jingyue Liang, Narseo Vallina-Rodriguez, Yao Guo, Li Li, Juan Tapiador, Jingcun Cao, and Guoai Xu. Beyond google play: A large-scale comparative study of chinese android app markets. In *2018 Internet Measurement Conference (IMC '18)*, 2018.

[55] Fengguo Wei, Yuping Li, Sankardas Roy, Xinming Ou, and Wu Zhou. Deep ground truth analysis of current android malware. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, pages 252–276. Springer, 2017.

[56] Yanyan Lan Xueqi Cheng Xiaohui Yan, Jiafeng Guo. A biterm topic model for short text. *WWW2013*, 2013.

[57] Lok-Kwong Yan and Heng Yin. Droidscope: Seamlessly reconstructing the os and dalvik semantic views for dynamic android malware analysis. In *USENIX security symposium*, pages 569–584, 2012.

[58] Wenbo Yang, Yuanyuan Zhang, Juanru Li, Hui Liu, Qing Wang, Yueheng Zhang, and Dawu Gu. Show me the money! finding flawed implementations of third-party in-app payment in android apps. In *Proceedings of the Annual Network & Distributed System Security Symposium (NDSS)*, 2017.

[59] Zhemin Yang, Min Yang, Yuan Zhang, Guofei Gu, Peng Ning, and X Sean Wang. Appintent: Analyzing sensitive data transmission in android for privacy leakage detection. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pages 1043–1054. ACM, 2013.

[60] Min Zheng, Patrick PC Lee, and John CS Lui. Adam: an automatic and extensible platform to stress test android anti-virus systems. In *International conference on detection of intrusions and malware, and vulnerability assessment*, pages 82–101. Springer, 2012.

[61] Wu Zhou, Yajin Zhou, Xuxian Jiang, and Peng Ning. Detecting repackaged smartphone applications in third-party android marketplaces. In *Proceedings of the second ACM conference on Data and Application Security and Privacy*, pages 317–326. ACM, 2012.

[62] Haowen Zhu, Xiaojun Ye, Xiaojun Zhang, and Ke Shen. A context-aware approach for dynamic gui testing of android applications. In *Computer Software and Applications Conference (COMPSAC), 2015 IEEE 39th Annual*, volume 2, pages 248–253. IEEE, 2015.