This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2019.2908939, IEEE Transactions on Dependable and Secure Computing

Dating with Scambots: Understanding the Ecosystem of Fraudulent Dating Applications

Yangyu Hu, Haoyu Wang, *Member, IEEE,* Yajin Zhou, *Member, IEEE,* Yao Guo, *Member, IEEE,* Li Li, *Member, IEEE,* Bingxuan Luo and Fangren Xu

Abstract—In this work, we are focusing on a new and yet uncovered way for malicious apps to gain profit. They claim to be dating apps. However, their sole purpose is to lure users into purchasing premium/VIP services to start conversations with other (likely fake female) accounts in the app. We call these apps as *fraudulent dating apps*.

This paper performs a systematic study to understand the whole ecosystem of fraudulent dating apps. Specifically, we have proposed a three-phase method to detect them and subsequently comprehend their characteristics via analyzing the existing account profiles. Our observation reveals that most of the accounts are not managed by real persons, but by chatbots based on predefined conversation templates. We also analyze the business model of these apps and reveal that multiple parties are actually involved in the ecosystem, including *producers* who develop apps, *publishers* who publish apps to gain profit, and *the distribution network* that is responsible for distributing apps to end users. Finally, we analyze the impact of them to users (i.e., victims) and estimate the overall revenue. Our work is the first systematic study on fraudulent dating apps, and the results demonstrate the urge for a solution to protect users.

Index Terms—Fraud, Mobile App, Dating App, Malware, Android.

1 Introduction

Mobile malware is rapidly becoming a serious threat in recent years. The main incentive for attackers to develop malware is that they could gain illegal profit. For example, previous research showed that malware authors could gain a profit by injecting advertisements in benign applications (or apps in short) [1], or by sending SMS messages to premium-rate numbers [2]. With the deployment of new defenses in the latest Android versions, these methods become less effective. However, we have observed a trend that malware authors have invented new ways to make a profit.

In this paper, we focus on a new and yet uncovered way for malicious apps to make a profit. Users are lured into installing a particular kind of dating apps and paying subscription fees for the right to chatting with existing users. However, the sole purpose of these apps is to cheat new users into paying, as the existing accounts in these apps are usually fake identities managed by chatbots. This kind of apps is therefore referenced as *fraudulent dating apps* (or FD apps in short).

Properties of FD Apps FD apps are usually distributed through online advertising networks, enticing users to install them with attractive pictures or fake claims. Once it is installed, users need to register an account to use the app. Surprisingly, this process is much simpler than what we have expected. Indeed, during our analysis of some apps, we find that the user only needs to click a few buttons to register a new account, without providing any personal details such as email address or phone number. This is different from the traditional malware that aims to steal user's private information.

After registration and logging into the app, many (female) accounts will initialize conversation requests to the user within a few minutes and likely with seductive words or pictures. For instance, during the manual analysis of one of the apps in our study, seven users sent conversation requests after logging into the app for 5 minutes (Figure 1(a)). All of them were female with attractive profile avatars. Two different users (the last two) were sending the same message ("Where are you from?") at the same time (8:29).

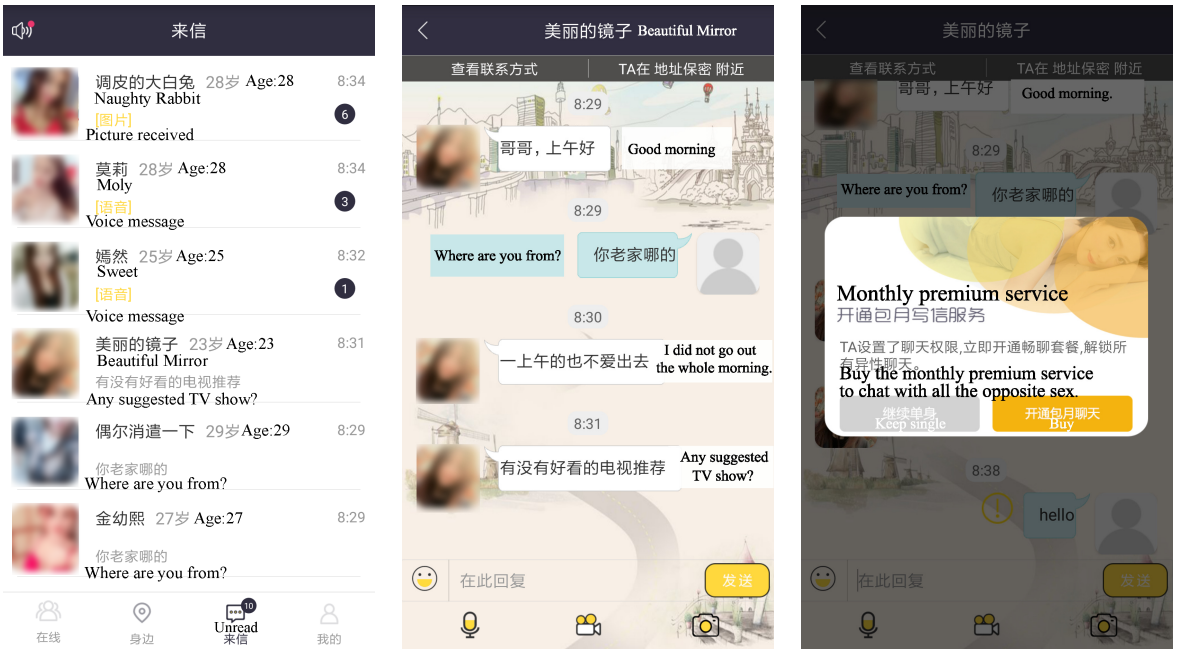
Due to this abnormal user behavior, we suspect that the existing accounts in the app are not real people but chatbots. To confirm our speculation, we sent messages to a randomly picked user (nickname: Beautiful Mirror), and the replied messages were irrelevant to the topic of the conversation (Figure 1(b)). Interestingly, after sending one message, we cannot send messages for free anymore.

In order to continue the conversation, we have to subscribe to the monthly premium service (Figure 1(c)) with a cost around seven US dollars. After purchasing the service, the previously communicated users stop responding immediately and there were no more users attempting to communicate with us anymore.

This app is not a single case: there exists an underground ecosystem for these FD apps. In particular, if a dating app shows the following behaviors, it can be categorized as a FD app. 1) *abnormal user behaviors*: after a new user logs into the app, several users will start a conversation in a short period, e.g., a few minutes; 2) *irrelevant messages*: messages are usually irrelevant to the conversation and out of the topic; 3) *premium services*: a new user cannot send or can only send a few messages for free, unless a premium service is

- *Yangyu Hu, Haoyu Wang and Bingxuan Luo are with Beijing University of Posts and Telecommunications, China. E-mail: {huyangyu910731, haoyuwang, unique_girl}@bupt.edu.cn*
- *Yajin Zhou is with Zhejiang University. Email: yajin_zhou@zju.edu.cn*
- *Yao Guo is with Peking University. Email: yaoguo@pku.edu.cn*
- *Li Li is with Monash University. Email: li.li@monash.edu*
- *Fangren Xu is with Rivermont Collegiate. Email:fangrenx@rvmt.org*
- *Haoyu Wang and Yajin Zhou are co-corresponding authors.*

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2019.2908939, IEEE Transactions on Dependable and Secure Computing



(a) Many users want to start conversation within a few minutes

(b) The messages are usually irrelevant to the conversation

(c) Premium service is needed to continue conversation

Fig. 1: An example of a FD app.

purchased. Finally, after purchasing the service, other users suddenly stop responding to any messages. To distinguish with the fake accounts in the app, we call the new user who is purchasing the services as a *victim*.

Study Overview In this paper, we perform a systematic study of FD apps, including the characterization of the user profiles and interaction patterns, the business model and involved parties in the ecosystem, as well as the distribution and the impact of these apps to victims. In particular, our research aims to answer the following questions. First, are the existing user accounts in FD apps are real persons or chatbots? Second, what is the business model of the FD apps, and what parties are involved in the ecosystem? Third, how are the FD apps distributed? Fourth, what is the impact of these FD apps to mobile users? For instance, how much money might be charged to a victim?

To this end, we first propose a method to detect FD apps from 2.5 million apps downloaded from nine third-party Android app markets[1], and the Google Play. In total, we have detected 967 distinct FD apps and classified them into 22 families based on their code similarity. We then perform detailed analysis on the detected apps and observe some interesting findings, listed as following.

- We found that most of the accounts in these apps are chatbots, with fake user profile avatars. For instance, we find that the same user profile avatars are used by multiple different accounts in the same app, and even in different apps.
- There are multiple parties involved in the ecosystem, including app producers, app publishers, and distribution networks. For example, we find that one developer key has been used to sign many FD apps with different package names, and published by different companies with the same legal representative(s).
- These apps are usually distributed through app markets and advertising networks. The fraudulent ranking techniques, e.g., fake user reviews and ratings are used to manipulate the ranking of the apps (i.e., promote the apps).
- We conduct an estimation of the overall revenue for the FD apps we have detected based on several reports. Our estimation concludes that the total market scale is around 200 million US Dollars to 2 Billion US Dollars.

1. Since the Google Play is not available in some countries or regions, these third-party app markets are the de facto official markets in these countries or regions.

Contributions In summary, this paper makes the following main contributions:

- We have presented a new way adopted by malware authors to make a profit through luring users into buying premium services in dating apps.
- We have conducted a systematic study of FD apps and answered several key research questions. To the best of our knowledge, our study is the first systematic study of such kind of apps.
- Our investigation has revealed various interesting findings that are previously unknown to the community. We believe our study is the first step towards a better detection and regulation of such apps.

To engage the community, we have released all the FD apps we identified in this study and the experiment results to the research community for further analysis at:

https://github.com/fakedatingapp/fakedatingapp

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2019.2908939, IEEE Transactions on Dependable and Secure Computing

TABLE 1: Overview of our dataset and the distribution of identified FD apps.

Market	#Apps	# FD Apps	# FD Apks
Baidu Market	227,454	673	3,227
360 Market	163,121	177	1,761
Tencent Myapp	636,265	432	2,782
Xiaomi	91,190	91	974
Wandoujia	554,138	285	2,767
Huawei	51,303	308	1,994
Lenovo	37,716	186	1,994
OPPO	426,419	305	1,628
Meizu	80,573	512	2,571
Google Play	287,110	7	123
Total	2,555,199	967	3,697

2 Fraudulent Dating Apps Characterization

In this section, we first present our approach to identify FD apps in Section 2.1. Then, in order to answer the following research question: *Are the existing user accounts in these apps real persons or chatbots?*, we dissect the existing account profiles in different FD app families and then analyze the interaction patterns between users of those dating apps in Section 2.2 and Section 2.3, respectively.

2.1 Identifying Fraudulent Dating Apps

2.1.1 Dataset

Table 1 presents an overview of our raw dataset that contains more than 2.5 million apps collected from ten Android app markets, including the official Google Play store. All of these apps were downloaded between April and August 2017. We have also crawled the metadata of these apps, including app name, publisher company name, app version, rating, the number of downloads, etc.

2.1.2 Methodology

We propose a semi-automated approach to identify FD apps, as shown in Figure 2. We first use a fast keywords matching method on the app metadata (e.g., app description and app name) to filter dating app candidates from the millions of apps we have crawled. Then we perform static code analysis on the selected candidate apps to check whether they have embedded in-app purchase services.

The rationale behind this checking is that those services are essential for the app publisher to gain a profit, i.e., for victims to purchase premium services. we then cluster them based on resource similarity and code similarity. For apps in each cluster, we manually select several apps to inspect whether they have suspicious characteristics such as abnormal user behaviors and irrelevant messages, which make those apps as fraudulent app candidates. We also analyze the user comments to further confirm that there exist victims of these apps in the real world.

Keywords Matching Because the FD apps usually use seductive texts to attract victims, we first manually summarized 11 common words (in both English and Chinese)[2] that frequently occur in the descriptions or app names of those apps. We use a fast keyword matching method to identify potential dating app candidates. Eventually, we are able to identify $61,133$ apps (out of 2.5 million apps) that contain at least two keywords.

2. 11 Keywords in both English and Chinese: secret dating, local single, find girl, find love, date invitation, blind date, lonely, say hello, quick match, nearby date, chat up

TABLE 2: Third-party in-app purchase services.

Alipay	https://www.alipay.com
WeChatPay	https://pay.weixin.qq.com/index.php/core/home/
Paypal	https://github.com/paypal/PayPal-Android-SDK
YeePay	https://www.yeepay.com
Ping++	https://www.pingxx.com
BaiduPay	https://www.baifubao.com
JieShenPay	http://jieshenkj.com
IPayNow	Ipaynow.cn
LianlianPay	http://www.lianlianpay.com/international/
UnionPay	https://merchant.unionpay.com/join/index
MengPay	http://www.cnmengpay.com
PayEco	https://www.payeco.com
SwiftPass	http://www.swiftpass.cn
JuHe	https://www.ijuhepay.cn
JuBaoPay	http://www.jubaopay.com/#/
99Bill	https://www.99bill.com
IAppPay	https://www.iapppay.com
BBNPay	https://www.bbnpay.com

In-app Purchase Analysis One of the most important characteristics of FD apps is that they try to entice users to purchase their premium services. For the selected dating app candidates, we perform static code analysis to detect whether they contain embedded in-app purchase services. If so, they will be identified as candidates for further analysis.

In general, there are two ways for an app to implement in-app purchase functions. The most popular approach is to embed in-app purchase SDKs, and the other one is to implement their own payment functions (e.g., send SMS to premium number). In this paper, we propose to detect them separately.

Identifying the in-app purchase SDKs is a non-trivial task due to code obfuscation and numerous of SDK versions, as suggested by many previous work. Note that developers charging for apps and downloads from Google Play must use Google Plays payment system [3] (Google Play Billing API [4]), which could be easily identified by checking the corresponding permission "com.android.vending.BILLING" and the related APIs (e.g., billingClient.launchBillingFlow). Besides, in some countries or regions (e.g., China) where the Google Play service is not available, app developers tend to use third-party in-app purchase services. For instance, the AliPay [5] and WeChatPay [6] are the two most popular third-party in-app payment services in China. In this paper, we take advantage of LibRadar [7], a widely used obfuscation-resilient tool to identify third-party libraries used in Android apps. We consider 18 popular third-party in-app purchases SDKs that are widely used in both China and worldwide, as shown in Table 2. Note that we did not rely on simple text matching, but code-level feature comparison to identify the third-party in-app purchases SDKs. LibRadar has labelled the code-level features (e.g., Android System API calls and Intent, etc.) of these third-party SDKs using clustering-based approach, and we use LibRadar to identify these SDKs, which is obfuscation-resilient and could be used to identify different SDK versions.

Since some apps may implement their own payment

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2019.2908939, IEEE Transactions on Dependable and Secure Computing

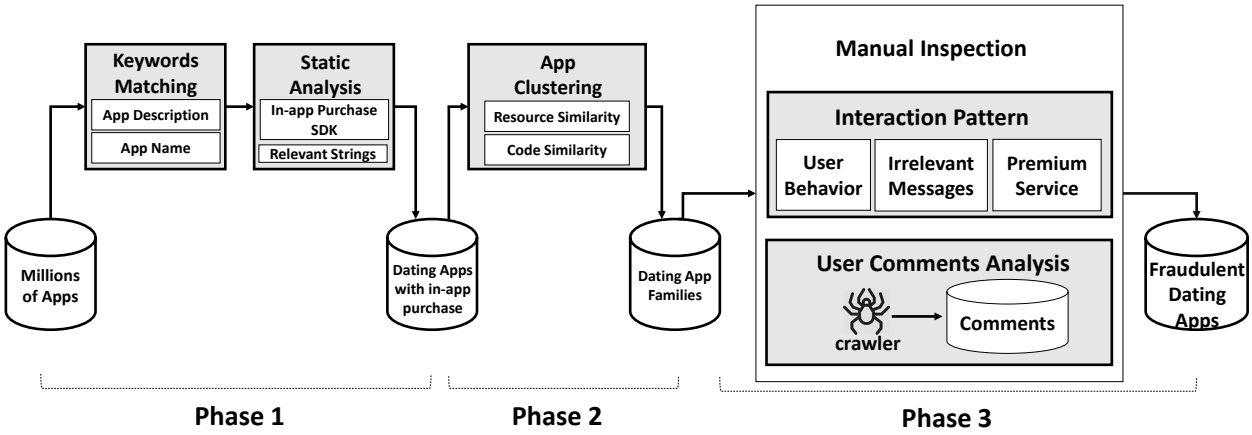


Fig. 2: Our approach to identify FD apps.

TABLE 3: An Overview of 22 FD app families.

Family	# Apks	#Pkg	Family	# Apks	#Pkg
Youyuan	1,104	496	Yuanlai	70	12
Appforwhom	742	70	Qianshoulian	272	15
Youairen	596	140	Erwanshenghuo	25	4
Yueaiapp	37	20	Zlewx	16	5
Tanliani	165	26	Xiangyue	53	25
Wmlover	179	32	99Paoyuan	44	5
Tongchengsupei	50	42	Qiaiapp	27	13
Jiangaijiaoyou	32	13	Meiguihunlian	65	2
Aiaihunlian	31	10	Jucomic	22	8
Sipuhaiwei	55	13	Ailiaoba	7	2
Yuanfenba	51	11	Michun	54	3
Total	3,697	967			

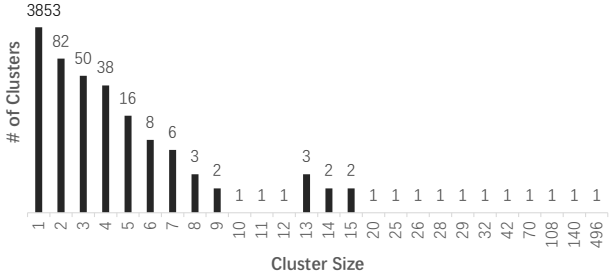


Fig. 3: Distribution of app cluster.

functions (e.g., send SMS to premium number) instead of directly embedding third-party payment services, we further investigate 5 related keywords[3] in the layout configuration files (e.g., strings.xml) of decompiled apks to identify app candidates as supplementary.

In total, we have identified $23,546$ dating apps (out of 61,133 dating app candidates) that contain in-app purchase services.

App Clustering For the dating apps that embed in-app purchase services, we then cluster them based on resource similarity and code similarity. Considering that one app (package name) corresponds to several APKs as our crawler downloads different versions of apps during a 4 months span, we remove reduplicate APKs according to the package name in the app clustering process.

We first take advantage of the open source system FSquaDRA2 [8] to measure the resource similarity of each app pair based on a feature set of resource names and asset signatures (the MD5 hash of each asset of an application excluding its icon and XML files).

We then use an app clone detection tool WuKong [9] to measure code-level similarity. For the apps with resource similarity scores higher than 90% and code-level similarity scores higher than 85%[4], we group them into the same cluster. Figure 3 shows the distribution of app clusters, the cluster size varies from 1 to 496. Finally, we select the cluster whose size is equal or bigger than 2. In total, we identified $5,547$ candidate apps (1824 unique package names) into 226 clusters (size $>= 2$)[5].

Manual Inspection For apps in each cluster, we manually select three apps[6] (596 apps in total). We installed them on smartphones and then registered real accounts to check whether they have the typical characteristics such as abnormal user behaviors (many users will start conversations even though our registration information is empty or totally unattractive), irrelevant messages and premium services.

With the help of manual inspection, we eventually flag 22 out of the 226 clusters as FD app families.

2.1.3 Statistics of Fraudulent Dating Apps

As shown in Table 3, we have identified 3,697 FD apps (APKs) that share 967 unique package names. These apps account for 6% of dating apps in our dataset, which is much higher than we expected. For each family, we choose the keyword from the package name that has the most number of downloads as the family name. For example, the family `Youyuan` includes the most number of unique package names

3. 5 Keywords in both Chinese and English: purchase & VIP, privilege, subscription service, unlimited chat, recharge account

4. we choose the threshold empirically based on the previous studies [8], [9]

5. Note that we did not consider 1-app clusters as possible FD app candidates in this paper, mainly due to two reasons. On one hand, we have manually selected 10 clusters for each cluster size (1, 2, 3, 4, 5, $[6, 10]$, $[11, 20]$, > 20) for inspection, and found no fraudulent behaviors in the 1-app clusters. On the other hand, as suggested by many previous studies, malicious developers usually release malicious apps in the form of app repackaging or code reuse.

6. If the cluster size is 2, we select both apps.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2019.2908939, IEEE Transactions on Dependable and Secure Computing

Request

URL: **http://ad.aiaihunlian.com/(35394BB62360F5086774681D1CE1EBC9)/search/getYuanfen.go**
Post body: {"area": {"areaId": 0, "cityId": 0, "provinceId": 1, "provinceName": "BeiJing"},
{"platformInfo":{**"pkgName": "com.aahl.dskya"**, "platform": "3"},"pageNum": 1,
"pageSize": 100,",…}

Response

{
"listYuanfen": [{
"isSayHello": 0, "userBase": { "age": 19, "area": {"provinceName": "Beijing",……..,}, "gender": 1,
"height": 172, "id": 7056, **"nickName": "Beautiful Mirror"**, "image": {"id": 0, **"imageUrl": "http://img.aiaihunlian.com/do?uri=/user/201506/23/13/35/1435037725100A4F49E4_c.jpg&w=300&h=300&s=1"**,.....}},}]
}

Fig. 4: Protocol Analysis: the request and response messages to retrieve user information.

TABLE 4: Server Address Analysis of 22 FD App Families.

Family	Addr of User List Request	Download Addr of Avatar
Youyuan	hulu.youyuan.com	ptw.youyuan.com
Appforwhom	aus.appforwhom.com	img0.appforwhom.com
Youairen	api.youairen.cn/api/users	cdnimg.365yf.com
Yueaiapp	napi.yueaiapp.cn	nimgup.yueaiapp.cn
Tanliani	api.tanliani.com	img.miliantech.com
Wmlover	app.wmlover.cn	cdn.wmlover.cn
Tongchengsupei	jiaoyouappslb.tongchengsupei.cn	makefriends.oss-cn-qingdao.aliyuncs.com
Jiangaijiaoyou	jiangaijiaoyou.com	image.jiangaijiaoyou.com
Aiaihunlian	ad.aiaihunlian.com	img.aiaihunlian.com
Sipuhaiwei	app.sipuhaiwei.com	piccdn.sipuhaiwei.com
Yuanfenba	api2.app.yuanfenba.net	image.yuanfenba.net
Yuanlai	mobileapi.yuanlai.com	photo4.ylstatic.com
Qianshoulian	mpc5.qianshoulian.com	mpc5.qianshoulian.com
Erwanshenghuo	api.erwanshenghuo.com	pic.erwanshenghuo.com
Zlewx	zlewx.com	image.zlewx.com
Xiangyue	v1.5xiangyue.cn	7xkly7.com1.z0.glb.clouddn.com
99Paoyuan	api2.99paoyuan.com	img7.kainei.com
Qiaiapp	app.qiaiapp.com	photo.qiaiapp.com
Meiguihunlian	api.meiguihunlian.com	photo.meiguihunlian.com
Jucomic	yuemei.jucomic.com	photo.jucomic.com
Ailiaoba	friend.ailiaoba.com.cn	liaobaimg1.mosheng.mobi
Michun	api.michun.fallchat.com	youyu-michun.oss-cn-shenzhen.aliyuncs.com

and APKs, as roughly one third of the APKs and more than half of the packages belong to this family.

The distribution of FD apps for different markets is shown in Table 1. The Baidu Market hosts the most number of FD apps, where more than two-thirds of total FD apps are from this market. The official Google Play market contains the least number of FD apps, where only 7 apps are flagged.

2.2 User Profile Analysis

2.2.1 Protocol Analysis

It is non-trivial to harvest account profiles from FD apps, as they are not directly available on the devices. We hence resort to the network traffic traces to retrieve user profiles. Particularly, we randomly select three apps in each family and run them on real smartphones. We then leverage *tcpdump* to record the network traffic traces. Table 4 shows the server addresses of user profile requests and the download addresses of avatar files for each family. Surprisingly, all the three apps we analyzed for each family share the same server address, even if they have different package names or developer signatures, which further indicates that they belong to the same family and controlled/developed by a same group of people.

We further investigate the request and response messages for user information retrieval, as shown in Figure 4. The request messages usually contain information like geo-location data, platform information, the number of requested user information, etc. The response messages contain a list of users where each of them is represented by a unique identifier, a URL of the avatar file, and some other personal information (e.g., nickname, age, etc.).

By deeply looking into those request and response messages, we observe that some apps (e.g., app `com.hzsj.qmrl` and app `com.wanjiang.tcyasq` in the app family `Youyuan`) embed a fingerprint or package name in the request message to differentiate the apps (cf. Figure 4). Moreover, some apps (e.g., app `com.yuanfenapp.tcyyjiaoyou` and app `ltd.onedream.snsapp.moaiyueai` in the app family `Tongchengsupei`) may share exactly the same request and response messages, resulting in identical accounts.

2.2.2 Crawling Account Profiles

To crawl the account profiles, one straightforward approach is to simulate the protocol for each app. However, because account information is usually shown based on geo-location (e.g., you could only browse the user locations in the same city), and each request could only get limited number of users (e.g., one page), we need to analyze the request URLs to identify the corresponding fields, and construct request messages (e.g., change the city or the page number of user profiles) so as to crawl as many as possible user profiles. Take Figure 4 as an example, we need to change the value of "provinceId" and "provinceName" to get users that in different cities, and change the value of "pageNum" to get more users.

Unfortunately, app developers could use anti-crawling techniques such as embedding hash values in the request URLs, to keep us away from automatically harvesting their account profiles. In our dataset, account profiles of apps in families "Youyuan", "99Paoyuan" and "Jucomic" can not be harvested by simulating the protocol. Due to this reason, we propose to employ automated app testing techniques to infer user profiles. In particular, as the tested app only shows users who are in the same city with us, our crawler is equipped with a GPS simulator [10] to set our location as 50 major cities in china and 10 major cities in US. For each location of tested app, we leverage an automated UI testing tool DroidBot [11] to generate UI pull-down events and send to the tested apps to emulate real user behaviors of browsing the account list, we send pull-down events continuously until no new account profile appear. Note that we leverage *tcpdump* to record all the network traffic traces during our dynamic testing, and recover all the user profiles from the traffic.

2.2.3 The Presence of Fake Account Profiles

It is difficult to measure how many fake accounts actually exist in each app since it is impossible for us to start a conversation with each account to check whether he/she is a real person or not. Motivated by the characteristics of Romance Scam fraud [12] that the scammers usually post profiles using stolen photographs of attractive people, we believe the fake accounts in FD apps may also use stolen/online photos too. To identify fake accounts in a fast manner, we regard the accounts with the same avatar photos but totally different account information (e.g., nickname, hometown, age, etc.) within the same app as *fake accounts*.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2019.2908939, IEEE Transactions on Dependable and Secure Computing

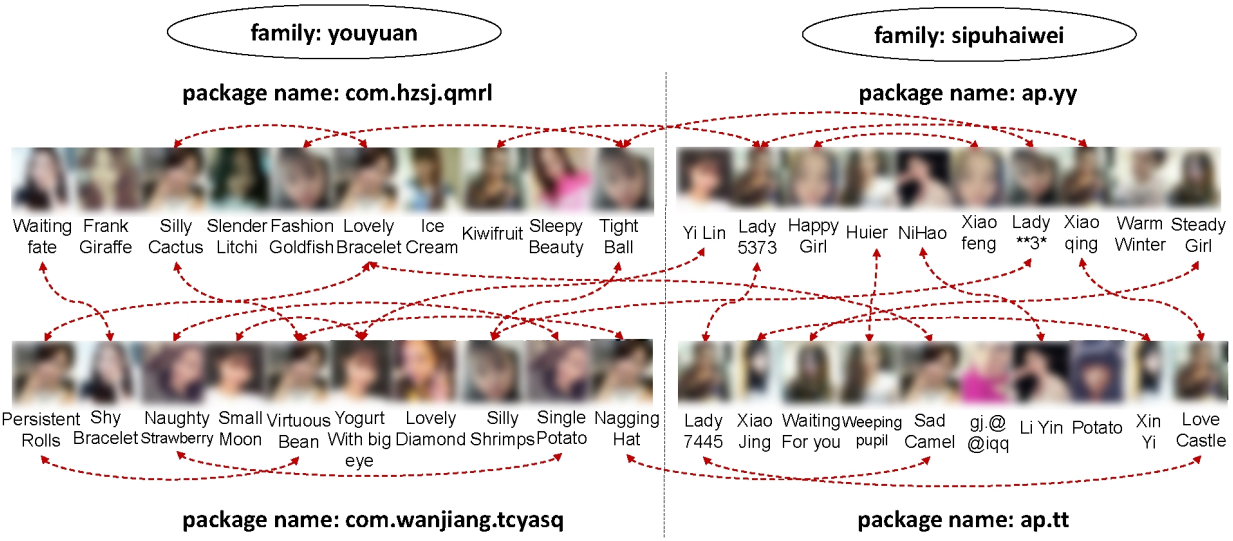


Fig. 5: Examples of fake account profiles.

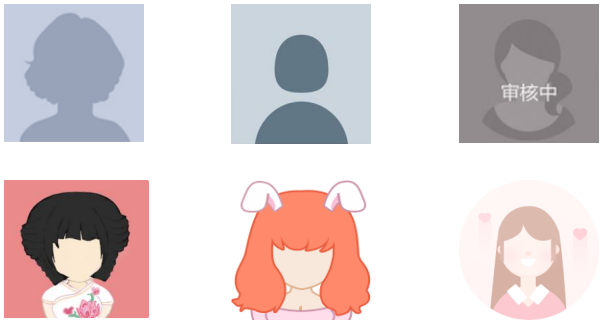


Fig. 6: Example of 6 default avatar photos.

Excluding Default Avatar Photos. Note that some apps offer default avatar photos during registration, which could mislead our detection. Thus we exclude all the default avatars from photo comparison. To get a full list of default avatars, we rely on two complementary approaches. On one hand, we extract all the pictures whose name is relevant to the avatar (e.g., default_avatar, girl, woman_normal) from the decompiled apks (mostly in the res directory). On the other hand, for each family, we crawl the default avatars when we register the accounts in the field study (cf. Section 2.3). In this way, we have identified 14 kinds of default avatars, Figure 6 shows examples of 6 popular default avatars. It is worthy noting that, if fake accounts use the default avatar, we may miss these fake accounts as we do not consider the default avatars. However, during our empirical analysis, we found that these fraudulent dating apps usually leverage attractive photos to lure users, and we did not identify the case that they use default avatars. Besides, although our method to identify the fake accounts is straightforward and easy to be passed by sophisticated developers, we could get the lower-bound of the percentage of fake users in this kind of apps. We will discuss the limitation of this method in the Section 7.3.1.

Identifying Duplicate Images. In this study, we use Dup Detector [13] to identify duplicate images. This approach is proved to be effective in finding duplicate and similar images by comparing image pixel data, and used in many other research studies [14] [15]. Note that, to cope with the cases that fake account adjusts the avatar photos by slightly changing the image (e.g., image clipping and resolution adjusting), we first use PIL [16], a python image processing library to clip the images, scale and zoom them into the same size. Then, we use a threshold of 95% in identifying duplicate images, according to our empirical evaluation on slightly changed images.

TABLE 5: The distribution of fake account profiles within the same app for each family.

Package Name	# Account Profiles	# Fake Account Profiles	Percent
Youyuan	57,779	7403	12.81%
Appforwhom	5,108	130	2.55%
Youairen	263,932	153,842	58.29%
Yueaiapp	2245	6	0.27%
Tanliani	969	0	0.00%
Wmlover	5516	683	12.38%
Tongchengsupei	3663	42	1.15%
Jiangaijiaoyou	1872	0	0.00%
Aiaihunlian	663	4	0.60%
Sipuhaiwei	23,946	1646	6.87%
Yuanfenba	3864	25	0.65%
Yuanlai	2178	8	0.37%
Qianshoulian	5431	312	5.74%
Erwanshenghuo	474	0	0.00%
Zlewx	3410	30	0.88%
Xiangyue	6451	38	0.59%
99Paoyuan	7829	411	5.25%
Qiaiapp	3422	2	0.06%
Meiguihunlian	7132	127	1.78%
Jucomic	4173	43	1.03%
Ailiaoba	3348	2	0.06%
Michun	5948	0	0.00%

For each family, we randomly choose an app and crawl all the account profiles. As shown in Table 5, for the app `com.jqyuehui.main` belonging to the `Youairen` family, we are able to crawl over 263,000 account profiles. We have crawled 419,352 account profiles in total. Figure 5 shows examples of fake account profiles we have found in our crawled data, from which we can observe that fake accounts may exist

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2019.2908939, IEEE Transactions on Dependable and Secure Computing

TABLE 6: The distribution of fake account profiles within the same family.

Family	# Account Profiles	# Suspicious Fake Account Profiles	Percent
Youyuan	137,497	130,558	94.95%
Appforwhom	15,324	15,324	100%
Youairen	431,697	237,476	55.01%
Sipuhaiwei	58,194	36,642	62.97%

TABLE 7: The distribution of fake account profiles across families.

Family	Youyuan	Appforwhom	Youairen	Sipuhaiwei
Youyuan	7403(12.81%)	137(2.68%)	3186(1.21%)	1435(5.99%)
Appforwhom	133(0.23%)	130(2.56%)	712(0.27%)	56(0.23%)
Youairen	3305(5.72%)	751(14.7%)	153842(58.29%)	2152(8.99%)
Sipuhaiwei	1316(2.28%)	58(1.14%)	1897(0.72%)	1646(6.87%)

within the same app, within the same family, or even across different families.

Fake Accounts within the Same App We first measure the fake account profiles within each app. As shown in Table 5, although we have identified fake account profiles in most of the apps, the percentage of fake account profiles within each app is not high. Only three apps have more than 10% of their account profiles detected as fake ones. Most of the apps have less than 1% of fake account profiles.

Note that, this is a conservative way to identify fake account since different fake accounts inside one app could use different avatar photos. We will analyze the interaction patterns in Section 2.3 to further detect the fake accounts.

Fake Accounts within the Same Family For each family, we choose three apps to examine fake account profiles across apps but within the same family.

Since it is generally time-consuming to analyze the protocol of a given app to simulate the request messages, in this work, we select four popular families (12 apps in total) to perform our measurements. For every app family considered, we ensure that the apps inside the family are different (i.e., has different package names). The type of account information (e.g., age, location) may slightly vary across different apps, we therefore regard the accounts with same avatar photos but different nicknames as *suspicious fake accounts.*

As shown in Table 6, the rate of suspicious fake accounts within the same family is significantly high. For example, around 95% of account photos of the selected three apps in `Youyuan` family are overlapped. The ratio of overlapping account profiles in `Appforwhom` family even achieves 100%. Further analysis reveals that all the apps in this family fetch user account profiles from the same server, besides they use the same protocol for accessing account information.

Fake Accounts Across Families We measure the ratios of account profiles overlapping across different families. As shown in Table 6, the overlapping ratios across different families are not high, where all of them are less than 10%. One possible reason is that apps in different families are from different developers and have different sources of account profiles.

Mostly Used Fake User Avatars We found many fake accounts using the same avatar photos but totally different account information. We list the top 20 popular fake user photos in Figure 7. All of the top 14 apps belong to `com.huizheng.yasq` (family `Youyuan`), while all of them have appeared in at least 130 different accounts. We also calculate the number of different accounts with each avatar photo and show it under the avatar in the Figure. It is interesting to see that the most popular avatar is associated to 186 different accounts. We further upload 100 popular avatars to four famous image searching engines[7] to see whether these avatars are crawled from the web, and found that more than 50% of them could be found on other web pages, but none of them are famous actor or actress.

2.3 Interaction Pattern Analysis

We now attempt to identify fake accounts from another perspective, i.e., the interaction patterns. If the accounts are real persons, then the messages should be relevant to the topic of the conversation. Thus, we perform a field study to analyze the interaction patterns of these fraudulent dating apps. For each family, we randomly choose an app and install it on a real device. Then we register two accounts (1 male user and 1 female user) to log in and start a conversation. Furthermore, we purchase the premium service for each app and compare the results before and after purchasing their services.

As shown in Table 8, we have observed several interesting findings:

1) *The registration process is quite easy, and most apps do not need any personal information*. As shown in the second column of Table 8, only 4 (out of 22) apps require the phone number, social networking or email account during registration.
2) *Several apps use template-based conversations.* As shown in the third column of Table 8, 3 (out of 22) apps use template-based conversations, which could be found in the resource files of the app.
3) *There is a huge difference between male users and female users*. For male users, when they are online, many female accounts will reach to them within a short time (see column #4). For example, more than 10 girls talked to our registered user for the app `com.yueai.ya007` (family: `Yueaiapp`) and `com.myhoney` (family: `Sipuhaiwei`) within five minutes during our experiment. However, for female users, there was no one trying to initiate conversation for almost all the apps during our experiment (see column #5). Specifically, for some apps (e.g., `com.liaoba`), the the default gender is male during registration and users cannot change it. This suggests that these apps are mainly targeting male users.
4) *For more than 70% of the apps, users cannot reply to the messages unless premium services are purchased*. For the remaining 6 apps, users could only respond to at most 3 messages.
5) *Irrelevant messages are prevalent in the conversations.* Before we purchased the premium services, we were

7. https://www.google.com/imghp?, https://pic.sogou.com/, https://image.baidu.com/, https://www.tineye.com/

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2019.2908939, IEEE Transactions on Dependable and Secure Computing

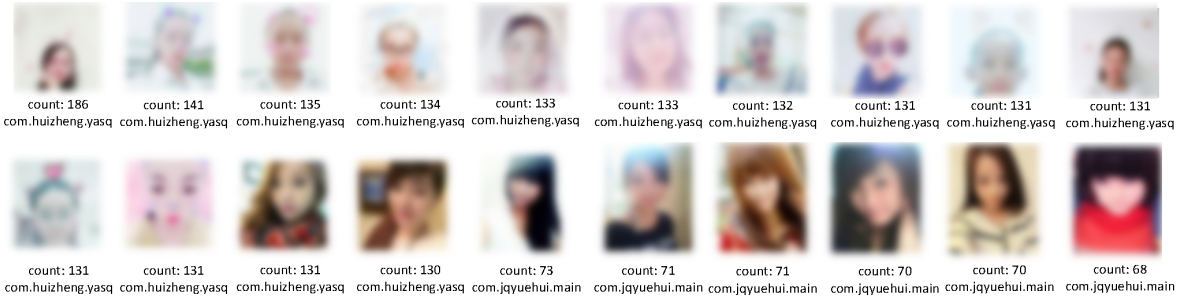


Fig. 7: Top 20 mostly used fake user avatars.

TABLE 8: Field study of different FD app families.

Family (package name)	Phone Num SN account Email	Template based	# Chatups (M)	# Chatups (F)	# Free Messages	Content Relevance	After Purchase
Youyuan (com.huizheng.dsya)	✗	✓	6	0	✗	NA	✗
Appforwhom (cn.com.qncnew.aus)	✗	✗	7	5	✗	NA	✗
Youairen (com.jqyuehui.main)	✗	✗	3	0	Total 3	✗	✗
Yueaiapp (com.yueai.ya007)	✗	✗	12	0	1 Per User	✗	✗
Tanliani (com.blsm.miyou)	✗	✗	6	0	Total 3	NA	✗
Wmlover (cn.umuad.dsaq)	✗	✗	5	0	✗	NA	✗
Tongchengsupei (ltd.onedream.snsapp.moaiyueai)	✗	✗	5	0	✗	NA	✗
Jiangaijiaoyou (com.jiangaihunlian.danshenyuehui)	✗	✗	6	0	✗	NA	✗
Aiaihunlian (com.aahl.jmyhb)	✗	✓	3	0	✗	NA	✗
Sipuhaiwei (com.myhoney)	✗	✗	14	0	✗	NA	✗
Yuanfenba (com.xiaochen.android.fate_it)	✗	✗	4	0	✗	NA	✗
Yuanlai (com.yuanlai)	✓	✗	7	0	✗	NA	✗
Qianshoulian (com.xiangqinqin.app)	✗	✗	2	0	1 Per User	✗	✗
Erwanshenghuo (com.syty.todayDating)	✗	✗	3	0	✗	NA	✗
Zlewx (com.jshy.tongcheng)	✗	✗	5	0	✗	NA	✗
Xiangyue (com.luren.xiangyueai)	✗	✓	5	0	✗	NA	✗
99Paoyuan (com.lingnei.kaikai)	✓	✗	6	✗	✗	NA	✗
Qiaiapp (com.jiaoyou.jqya)	✗	✗	4	0	✗	NA	✗
Meiguihunlian (com.meiguihunlian)	✗	✗	4	0	✗	NA	✗
Jucomic (cn.nineox.yuejian)	✓	✗	2	0	✗	NA	✗
Ailiaoba (com.liaoba)	✓	✗	3	✗	1 Per User	NA	✗
Michun (com.youyu.michun)	✗	✗	8	0	1 Per User	✗	✗

only able to reply to the messages in 6 apps. However, only 4 accounts in these apps replied to us and the response messages were totally irrelevant.

6) *After purchasing the premium services, the app stops responding to messages.* In this field study, we spent roughly 176 US dollars to purchase premium services for these 22 apps. Unfortunately, once we had purchased the premium services, all apps stopped responding to our messages. It appears that the sole mission of these apps is to lure users into purchasing its so-called premium services, which in reality do not exist at all.

2.4 Summary

Based on the results of user profile and interaction pattern analysis, we suspect that the accounts (except for the victims) in the apps are chatbots, instead of real persons. First, the account profiles in different apps in a family are mostly identical. These account profiles may be automatically generated. Second, our suspicion can be further confirmed by the interaction patterns. For instance, the messages in the conversation for the apps we evaluated are irrelevant to the topic, and no messages will be received after purchasing the premium services. These patterns are more likely generated from computer programs instead of real persons.

3 Business Model Analysis

We now analyze the business model of FD apps aiming at revealing the involved parties in the ecosystem, so as to answer the research question: *what are the involved parties and how they make a profit?*

We first retrieve the signature of the developer's key inside the app. If the signatures are the same in two different apps, we assume that these two apps are developed by

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2019.2908939, IEEE Transactions on Dependable and Secure Computing

TABLE 9: Multiple parties in the FD app ecosystem.

Family	#Pkg	# Developer Signature	# Companies	# Legal Persons
Youyuan	496	48	113	107
Appforwhom	70	43	21	10
Youairen	140	133	27	21
Yueaiapp	20	5	4	4
Tanliani	26	10	11	10
Wmlover	32	31	9	9
Tongchengsupei	42	5	16	16
Jiangaijiaoyou	13	6	7	7
Aiaihunlian	10	5	3	3
Sipuhaiwei	13	8	5	5
Yuanfenba	11	10	5	5
Yuanlai	12	1	3	3
Qianshoulian	272	12	7	7
Erwanshenghuo	25	1	3	3
Zlewx	16	1	2	2
Xiangyue	53	22	6	5
99Paoyuan	44	5	3	3
Qiaiapp	27	3	6	6
Meiguihunlian	65	2	1	1
Jucomic	22	6	3	3
Ailiaoba	7	1	2	2
Michun	54	3	1	1

the same developer[8]. These developers are referred as `app producers`.

We then collect the released company names of the FD apps from the app markets. Usually, the company name is a required entry when publishing apps to an app market. We also collect the name of the legal representative [17] of the company, based on the public records from the corresponding government agencies. These companies are the ones who publish apps and obtain payments from victims. We call them `app publishers`.

Table 9 shows the data of multiple parties involved in the FD app ecosystem. In particular, the first column shows the family name of the app, and the second column shows the number of distinct package names in each family. The third column shows the number of distinct developer signatures for the apps in each family, while the last two columns show the number of distinct company names and that of legal representatives of the companies. The number of legal representatives is less or equal to the number of companies since the same person can serve as the legal representatives of multiple companies.

Based on the data in Table 9, we obtain the following observations.

- The number of developer signatures is usually much fewer than the number of distinct package names in each family. This evidence indicates that the developers of different apps may be the same person. We also find the case that even the developer signatures of some apps are different, the names of the RSA files inside the META-INF directory of those apps are identical.
  For example, there are 130 apps in the family `youairen` sharing the same signature file name, namely *KEY_KEYS.RSA*. For each signature, the values of the `CN` (Common Name) and `OU` (Organization Unit) fields are meaningless strings such as `estituan`, `umgfoubq`. We believe even though the signatures of these 130 keys are different, they are probably automatically generated by the same person. Table 10 shows the randomly generated `CN` and `OU` fields of 20 developers' signatures.
- The apps belonging to the same family are published by multiple companies. This could be explained by the fact that the producers may sell their apps to different companies for publishing. For instance, as shown in the code snippet of Listing 1, the app developer hard-coded the relationship between the package names and the payment accounts that are used to receive payments from victims. By doing so, app producers could sell the same app with different package names to different companies (or publishers).
- One legal representative could own multiple companies so that FD apps can be published multiple times via different companies. If the apps from one company are removed from app markets due to user complains, the apps from other companies can still survive.

8. This is a reasonable assumption since the leakage of the developer's key is not a common case in practice.

TABLE 10: The randomly generated OU and CN field in the developers' signatures.

OU	CN	OU	CN	OU	CN	OU	CN
hyuhzwhl	fstxzj	btxetuui	pfsmdm	oymorwlp	vrnnuv	estituan	hvhskr
umgfoubq	vvshla	mdjgjpuc	tkwcpk	agggefmk	zhmipw	kovkokxi	jzmmeh
pbvdysyg	kychif	uzqoaawn	usfzum	gfpsbhmz	sejwql	dgvcmhxh	sqsqrf
pebbomoy	dacyfr	xwvoirgr	pgupio	vdsqjkvk	alzrly	ugqljbld	swkcom
memqwnon	nmscvv	wcjhpfkb	lxkruk	xdttilqo	zaamyd	hrzwtnzj	hvsbwp

```
1  str=package_name;
2  if ("com.huizheng.lasq".equals(str)) {
3      com.app.tencent.QQConstants.APP_ID="
         wxba66413d0f792ffa";
4      com.app.tencent.QQConstants.PARTNER_ID="1296280201";
5      return;
6  }
7  if ("com.youyuan.lrxq".equals(str)) {
8      com.app.tencent.QQConstants.APP_ID="
         wxf54f037e28294cc6";
9      com.app.tencent.QQConstants.PARTNER_ID="1296283001";
10     return;
11 }
12 ...
```

Listing 1: The code snippet showing the app package names and their corresponding WeChatPay accounts used to receive payments from victims

Figure 8 shows the business model of the ecosystem. Specifically, the app producers develop apps and sell them to publishers. The publishers usually register multiple companies and use these companies to distribute their apps, e.g., via app markets. In order to promote these apps, the ranking fraud techniques are used (Table 11 in Section 4.1). Moreover, the publishers could also pay the advertising network to distribute their apps (Section 4.2). When the victims are lured into installing these apps and buying the premium services, publishers will receive the money and gain a profit. Note that, the producers and publishers in some cases may come from the same companies, and act as both roles in the ecosystem.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2019.2908939, IEEE Transactions on Dependable and Secure Computing

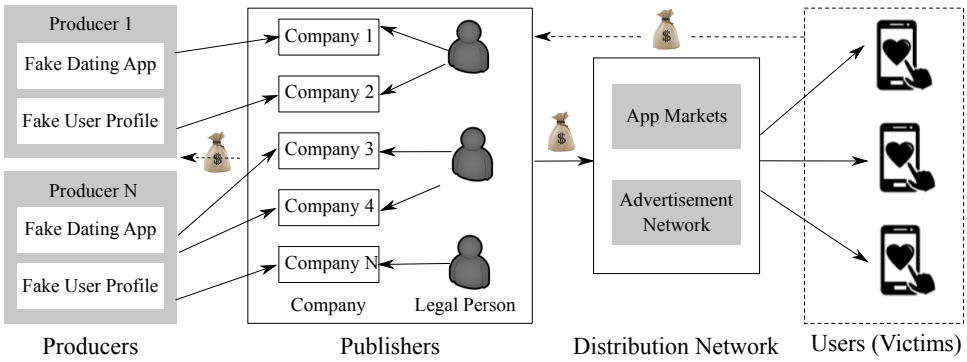


Fig. 8: The business model of the FD apps.

TABLE 11: Ranking fraud based on user reviews.

Family	# Reviews	Reviews # with Five-star Rating	# Repeated Reviews	# Fake Reviews	% Fake Reviews	Fake Reviews % with Five-star Rating	# Users	# Fake Users	% Fake Users
Youyuan	375,436	337,140	188,811	285,448	76.03%	95.82%	195,256	145,151	74.34%
Appforwhom	94,693	91,760	78,382	89,945	94.99%	98.73%	26,603	23,799	89.46%
Youairen	380,722	379,560	221,936	347,486	93.76%	99.26%	135,710	121,801	89.75%
Yueaiapp	20,920	19,562	13,490	17,990	85.99%	98.63%	15,895	13,811	86.89%
Tanliani	54,951	52,032	31,427	44,812	76.03%	97.74%	38,154	29,835	78.20%
Wmlover	103,162	100,183	71,787	91,081	88.29%	99.25%	62,352	52,005	83.41%
Tongchengsupei	4,746	4,594	2,007	4,230	89.13%	99.21%	3,959	3,479	87.88%
Jiangaijiaoyou	3,043	1,733	890	1,397	45.91%	78.38%	2,195	1,002	45.65%
Aiaihunlian	19,299	18,864	11,124	18,406	95.37%	99.09%	12,179	11,439	93.92%
Sipuhaiwei	46,036	45,337	27,691	44,292	96.21%	99.56%	15,603	14,287	91.57%
Yuanfenba	21,065	20,426	12,057	19,899	94.46%	98.73%	15,747	14,814	94.08%
Yuanlai	18,702	17,126	14,991	16,221	86.73%	97.95%	8,429	6,558	77.80%
Qianshoulian	38,514	37,006	24,571	34,867	90.53%	98.91%	26,346	23,509	89.23%
Erwanshenghuo	174	119	114	114	65.52%	85.96%	162	118	72.84%
Zlewx	1,004	959	775	921	91.73%	99.11%	972	911	93.72%
Xiangyue	13,408	13,108	10,604	12,865	95.95%	99.36%	9,645	9,257	95.98%
99Paoyuan	9,505	8,954	6,126	8,529	89.73%	97.74%	7,428	6,652	89.55%
Qiaiapp	8,955	8,521	6,537	7,362	85.23%	98.51%	5,780	5,088	88.03%
Meiguihunlian	1,692	1,506	758	1,289	76.18%	91.81%	1,237	1,013	81.89%
Jucomic	1,540	1,428	836	1,134	73.64%	98.21%	1,285	996	77.51%
Ailiaoba	626	592	66	306	48.88%	97.45%	611	300	49.10%
Michun	6,239	5,915	4,947	5,759	92.31%	98.15%	2,292	2,143	93.50%
Total	1,224,432	1,166,425	729,927	1,054,623	86.13%	96.70%	483,231	383,181	82.92%

4 Distribution Network Analysis

App publishers usually distribute their apps through both app markets and advertising networks (Figure 8). In this section, we analyze the distribution of the FD apps and answer the following question: *how these apps are distributed, and what techniques are used by publishers to promote these apps in app markets?* In particular, we collect the names and user reviews of these apps in app markets and monitor the app distribution through an online service [18].

4.1 App Markets

As expected, we find that app markets are the primary choice for app publishers to distribute apps (see Table 1 for the app markets in which FD apps are detected). Unfortunately, the ranking system of app markets could be manipulated (known as ranking fraud) by the owners of FD apps so as to attract more victims.

Ranking Fraud Ranking fraud [19] refers to the behaviors that aim to promote the ranking of apps inside app markets. Based on the ranking mechanisms of app markets, this could be achieved by manipulating the user reviews and rating of an app. For each app in our study, we therefore crawl the reviews and the ratings (if it is available) of the app in each market as well as the names of users who have posted the reviews, aiming at identifying fake reviews and fake reviewers.

Specifically, our analysis is based on the following reasonable heuristic: 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 11.

First, we remove the reviews that have less 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 reviews and log the number in the fourth column. The users who have posted repeated reviews

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2019.2908939, IEEE Transactions on Dependable and Secure Computing

are classified as fake reviewers correspondingly (shown in the ninth column).

Third, we further mark all the reviews from fake reviewers as fake reviews, which is shown in the fifth column. This step is added 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 calculate the percentage of fake reviews and users in the sixth and last column. 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 90% of reviews in 10 families are fake and more than 95% (96.70%) of the user ratings in the fake reviews are five-star, demonstrating that the ranking system is actively manipulated by the publishers of those FD apps.

4.2 Advertising Networks

Previous research [20] has revealed that mobile malware could be distributed through mobile advertising networks. In this study, we perform an initial investigation to check whether FD apps have been distributed through this channel. Our study leverages a third-party online service AppGrowing [18] to collect the corresponding data. In particular, given an app, AppGrowing provides a report containing whether this app has been distributed through an advertising network, and if so which networks have been involved. Note that, their data is through *sampling* the traffics of advertisement SDKs and thus is not complete. Nevertheless, the data still provide some insights of the distribution of FD apps.

Based on the three-month data from October to December 2017, the following advertising networks have been involved in the distribution of FD apps: the Cheetah Ad [21], IntelligentTui [22], Tencent Social Ad [23] and Baidu Ad [24]. The first one belongs to the Cheetah Mobile [25], a company listed in New York Stock Exchange, and the second and third advertising network belong to Tencent [26], the producer of QQ and WeChat and one of the largest Internet and technology companies in the world. The last one belongs to Baidu [27], the largest searching engine and one of the biggest mobile advertising networks in China. Table 12 shows the 26 apps we monitored and the advertising networks that have been leveraged to distribute those apps. In the table, we use the following abbreviations CH, IT, TD and BD to denote the Cheetah Ad, IntelligentTui, Tencent Social Ad and Baidu Ad, respectively.

TABLE 12: The package name of FD apps and their corresponding distributing advertising networks.

package name	AD Networks	Package Name	AD Networks
com.hzsj.kya	CH	com.aahl.zrl	BD
com.huizheng.tcyyhz	IT, CH	com.huizheng.kya	IT, TE
com.huizheng.dsyyh	IT, CH	com.mimivip.missyou	TE
com.huizheng.lasq	IT, CH, TE	com.hzsj.bdya	IT
com.youyuan.yyhl	IT	com.solo.peanut	IT
com.hzsj.dsjy	IT, TE	com.huizheng.tcxax	IT
com.dllingshang.tcjy	IT	com.yuanju.night	IT
com.futuredo.quickdate	CH	com.lingai.gaybar	IT
com.youyuan.yhb	IT, TE	com.meimei.yulove	IT
com.tanliani	CH	com.dljh.fjlxy	IT
com.youyuan.lrxq	IT	com.huizheng.jrtt	IT
com.dlkuaidu.tcayh	IT	com.prd.tosipai	IT
com.lingshang.ls	IT	com.hzsj.zxzdxz	BD

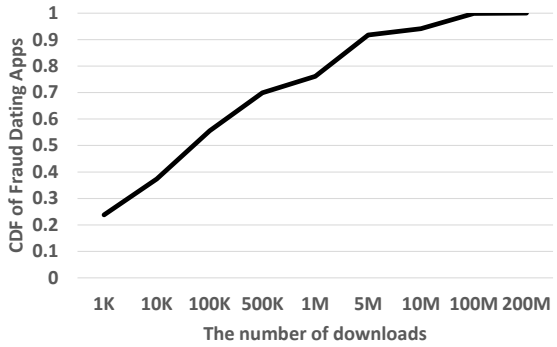


Fig. 9: The cumulative distribution of the number of app downloads.

5 User Impact Analysis

In this section, we analyze the user impact of the FD apps from the following three aspects. First, we measure the upper- and lower-bound of the number of victims. In particular, we first measure the downloads distribution of these apps, which could be used to calculate the upper-bound of the number of victims. Then we crawl the negative comments of these apps from app markets and pinpoint the number of users that complain that they have been cheated to purchase the premium services, which could be used to estimate the lower-bound of the number of victims. Second, we estimate the overall revenue of these apps based on several reports that disclose the revenue model of these FD apps. Third, we upload all these apps to VirusTotal to understand how many of them could be flagged by existing anti-virus engines.

5.1 Number of Victims Estimation

Downloads Analysis: the Upper-bound We first analyze the number of accumulated downloads for these apps crawled from 10 app markets. Figure 9 shows the distribution. Surprisingly, around 50% of apps have been downloaded more than 100K times, and roughly 25% of them have the number of downloads over 1 million. App `cn.feichengwuyue` has the most number of downloads (143 million).

We then examine the downloads distribution across families. As shown in Table 13, there are 6 families that have achieved more than 100 million accumulated downloads, in which the family `Youyuan` has accumulated downloads of 784 million. The total number of downloads for these 967 apps is surprisingly high, which has achieved 2.4 billion. Each app has an average of 2.5 million of downloads.

Negative Review Analysis: the Lower-bound Although we have shown that the positive reviews of these apps are mainly manipulated by ranking fraud techniques, the negative reviews are usually about the complaints of victim users. We have analyzed the negative reviews of these apps. Figure 10 shows the word cloud for the negative reviews.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2019.2908939, IEEE Transactions on Dependable and Secure Computing

TABLE 13: User impact analysis.

Family	#DLs	#Avg DLs	# Negative Ratings	# Victims	Profit Est by #DLs ($/Month)	Profit Est by #Rating ($/Month)	Avg Profit Est by #DLs ($/Month)	Avg Profit Est by #Rating ($/Month)
Youyuan	784.8M	1582K	22899	14795	399K	203K	804	409
Appforwhom	175.9M	2513K	2637	1320	89K	23K	1277	334
Youairen	684.7M	4890K	6393	5373	378K	57	2485	404
Yueaiapp	35.4M	1769K	1186	952	18K	11K	899	525
Tanliani	189.2M	7275K	2161	1538	96K	19K	3697	736
Wmlover	177.5M	5548K	2490	1860	90K	22K	2820	689
Tongchengsupei	5.3M	126K	136	115	3K	1K	64	29
Jiangaijiaoyou	6.2M	478K	1210	833	3K	11K	243	824
Aiaihunlian	4.9M	486K	358	269	2K	3K	247	317
Sipuhaiwei	65.2M	5013K	570	415	33K	5K	2548	388
Yuanfenba	9.1M	831K	591	468	5K	5K	422	476
Yuanlai	13.7M	1145K	1400	973	7K	12K	582	1033
Qianshoulian	92.4M	6163K	1055	800	47K	9K	3132	623
Erwanshenghuo	11.7M	2937K	55	47	6K	0.5K	1492	122
Zlewx	2.5M	496K	40	29	1K	0.4K	252	71
Xiangyue	143.4M	5735K	259	193	73K	2K	2915	92
99Paoyuan	6.3M	1255K	481	392	3K	4K	638	852
Qiaiapp	4.7M	365K	389	294	2K	3K	186	265
Meiguihunlian	0.8M	412K	89	43	0.4K	0.8K	209	394
Jucomic	10.9M	1357K	72	53	6K	0.6K	689	80
Ailiaoba	1.5M	759K	29	22	0.7K	0.3K	386	128
Michun	4.1M	1374K	252	148	2K	2K	698	744
Overall	2.4B	2.5M	44752	30932	1235K	396K	1277	410

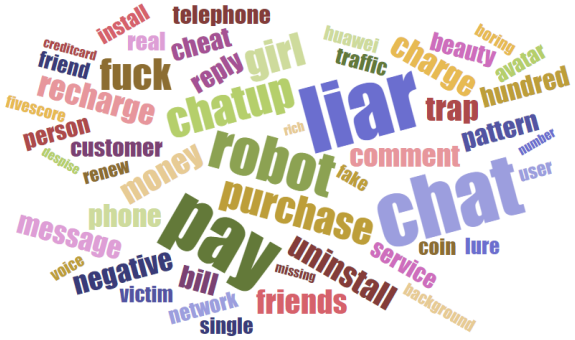


Fig. 10: Top words in the 44,752 negative reviews.

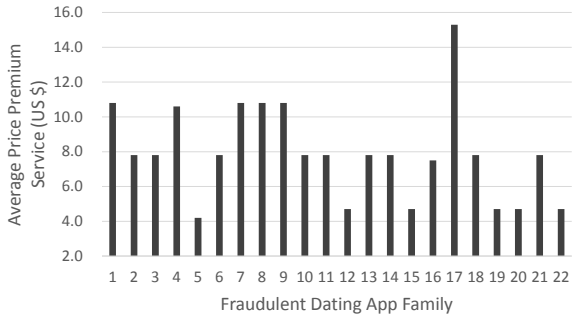


Fig. 11: The average subscription price for premium services.

Almost all of the users complain that they have been cheated to purchase the premium services.

Thus we measure the number of victims based on the negative comments. As shown in Table 13, we have collected 44,752 negative reviews in total, and the family `Youyuan` has occupied more than half of the negative comments. To estimate the lower-bound of the victims, we further try to pinpoint the number of users that complain that they have been cheat to purchase the services in the negative reviews. We manually summarized 10 keywords[9] and use them to search in the negatives reviews. At last, we found 30,932 users (69%) that are quite possible to be victims. We further use this result to estimate the lower-bound of the victims.

5.2 Payment Method Analysis and Profit Estimation

5.2.1 Price and Payment Method Analysis

For each family, we randomly choose three apps to analyze the price of the premium services they offered. As shown in Figure 11, the average price varies from 4.2 US dollars (family: "Tanliani") to 15.3 US dollars (family: "99Paoyuan"), and the median average price is 7.9 US dollars across the 22 families. We further analyze the payment method for each family. All the families support Alipay and WeChatPay, roughly 80% of them also embed the Union Pay service, while 30% of them provide the functionalities to send SMS to premium numbers.

5.2.2 Payment Identifier Analysis

To use WeChatPay, the merchant should define three necessary parameters: `appid`, `mch_id` and `secret key`. The appid and mch_id are usually hard-coded in the app by app developers, which are used as the identification of the merchant. Note that the appid is an 18-byte string with the prefix wx, the mch_id is a 10-byte digit string. Thus we first locate eligible strings in the decompiled code and then query the WeChatPay Web API to check whether we find the correct strings. At last, we have identified 232 unique WeChatPay identifiers (appid). With further analysis, we found that

9. 10 keywords in both English and Chinese: deceive, cheat, money, VIP, purchase, bill, fake, fooled, recharge, RMB

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2019.2908939, IEEE Transactions on Dependable and Secure Computing

one developer signature usually corresponds to several payment identifiers, while one payment identifier always corresponds to one company name (the distributor). This finding once again provides evidence suggesting that app developers might sell FD apps to different distributors.

5.2.3 Profit Estimation of FD Apps

It is non-trivial for us to estimate the profit of these FD apps. Several reports[10] have disclosed some information related with the revenue of this kind of apps. For example, it is reported that the Chinese polices have uncovered a case of fraudulent dating app at the early 2018, and the app was reported to have the revenue of more than 100 million US dollars one year. In this paper, we estimated the profit of FD apps using a fine-grained estimation approach based on some statistics provided by regulation departments in China.

Fine-grained Estimation. To accurate estimate the profit of FD apps, we further cooperated with one third-party payment regulation department and one anonymous app market regulation department in China to collect the profit data of FD apps. Note that we do not resort to exploit these apps ourselves to collect the unpublished revenue data due to ethical consideration. **We have reported all the FD apps we identified in this paper to the regulation departments in China. They have confirmed all the 22 FD app families. Then, they leveraged a recent work [28] to collect the profit data of FD apps under the guidance of a lawyer.**

Specifically, the WeChatPay allows the merchant to download their history income through a web API[11] with three necessary parameters: appid, mch_id and secret key. The appid and mch_id are usually hard-coded in the app by app developers, which are used as the identification of the merchant. WeChatPay adopts a hash function with the secret key to generate the message signature during the communication with the server. Thus the secret key should be kept safely and never leaked to others. However, Yang et al. [28] have found that some developers embed the secret key in the app, which could cause serious risks.

They have applied the techniques introduced by Yang et al. [28] and they did find the hard-coded key in one FD app (`the name is anonymized due to ethical concern`), which could be used to estimate the victims and the profit that malicious developers could get. Note that the appid is an 18-byte string with the prefix wx, the mch_id is a 10-byte digit string, and the secret key is a 32-byte string with arbitrary content. Thus they first locate eligible strings in the decompiled code and then query the WeChatPay Web API to check whether find the correct strings.

They have obtained the daily incoming of the app with vulnerability from May 2017 to Jan 2018 and they provided the results to us. The result suggested that there are roughly 330 victims who have purchased the premium services during the 9 months with a total cost of roughly 2,800 US dollars for the app with vulnerability [12].

Overall Profit Estimation We assume that the profit is linearly proportional to the number of downloads and the number of negative reviews. Based on the provided real incoming of the app with vulnerability, we can estimate the profit of the overall FD apps in our system.

From the perspective of app downloads, the app with vulnerability has 610K downloads in total, which could generate 310 US dollars profit per month on average from the WeChatPay channel. As shown in Table 13, the accumulated number of downloads for all the apps is 2.4 billion, thus we estimate that the overall market of the FD apps is 1.2 million US dollars per month (310*2.4B/610K). Each app could earn 1,277 US dollars per month on average. We understand that, if the number of downloads has been manipulated, the profit could be overestimated using this method.

From the perspective of negative reviews, because the anonymized app with vulnerability has 35 negative comments (possibly from the victims) in total, we estimate the overall profits of the market is roughly 0.4 million US dollars per month (310*44752/35), while each app could earn 410 US dollars per month on average, at the least (i.e., lower-bound). **Note that we only consider the WeChatPay channel in the profit estimation.** As shown in Section 5.2.1, most apps embed various payment channels, including Alipay, WeChatPay, Union Pay, etc. Thus the real number of the profit for the ecosystem should be several times higher than the lower-bound we estimated.

5.3 Detection Results of VirusTotal

The number of FD apps labeled by VirusTotal. We upload all the identified FD apps to VirusTotal to explore how many of them could be flagged by existing anti-virus engines. The distribution of VirusTotal flags of these apps is show in Figure 12. Surprisingly, more than half of them are labeled as malware by less than 1 anti-virus engine, meaning that most anti-virus engines are not able to flag those FD apps. As some anti-virus engines on VirusTotal may not always report reliable results, thus some previous work use a threshold of 10 engines to identify malware. However, only 5% of the FD apps are flagged by more than 10 anti-virus engines. This result suggests that **the FD apps cannot be sufficiently identified by existing anti-virus engines**.

Malware Categories. We then analyze the distribution of malware categories labeled by VirusTotal, as shown in Table 14. It is interesting to see that although roughly 700 APKs (18.7%) are labeled as LoveFraud (PUA) by at least one engine, more than 80% of the apps in our dataset are not identified as LoveFraud, even if they share the same behaviors and belong to the same families.

Malware Families. We then use AVClass [29], a widely used tool to obtain the family name (label) of each identified malware. It is interesting to see that, 55.3% of apps that flagged by at least one VirusTotal engines are assigned

10. http://www.thatsmags.com/shenzhen/post/21999/sexy-girl-bots-scam-1-billion-from-dating-app-users-in-china
https://www.bbc.com/news/blogs-news-from-elsewhere-42609353
http://www.marketing-interactive.com/12-chinese-dating-apps-close-down-after-women-found-to-be-robots/
http://www.cngold.com.cn/zjs/20180109d1897n202734126.html
http://www.lc123.net/xw/tp/2018-01-12/852867.html
http://cj.sina.com.cn/article/detail/1642467340/548172

11. https://api.mch.weixin.qq.com/pay/downloadbill

12. Note that no private information of the victims is obtained and we did not get the raw data. The daily incoming information only contains a randomly generated payment ID and the price of the payment. We only got the total number of victims and incoming of the app from the regulation departments in China.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2019.2908939, IEEE Transactions on Dependable and Secure Computing

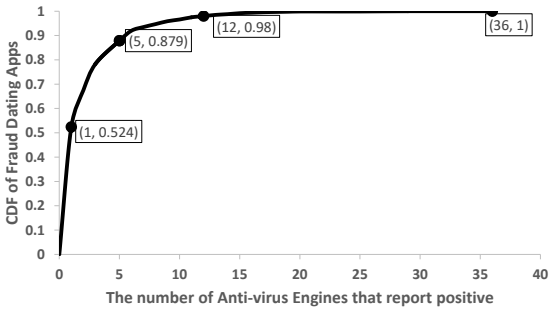


Fig. 12: The distribution of VirusTotal Flags.

TABLE 14: The distribution of malware samples labeled by VirusTotal.

Malware Signatures	Count	Percent
Trojan App	1138	30.7%
Android-PUP	813	21.9%
LoveFraud (PUA)	694	18.7%
Adware	377	10.2%
LustFishingMoney	259	7.0%
Riskware App	184	5.0%

"Singleton" as no family name was found in the AV labels. Families " smsreg" and " jiagu" are most popular, more than 5% and 4.4% of flagged malicious apps belong to them.

6 Post Analysis

All the app markets considered in this paper have strict policies to conduct security checks [30] [31] [32] [33] [34] [35] [36] [37] [38]. Yet our results reveal that they still host a significant amount of FD apps, even 7 FD apps found in Google Play. Also, the user review analysis (cf. Section 5.1) suggested that many users complain that they have been cheated to purchase the premium services. Thus, to understand the regulation process of these markets, we performed a second crawl of these FD apps for each app market at November 2018, in order to quantify whether the stores made any effort to remove these FD apps. As shown in the third column of Table 15, all the 7 FD apps[13] found in Google Play have been removed. However, the percentages of FD apps removal in Chinese alternative markets vary from 2% to 92%. For example, by November 2018, app "com.huizheng.tcyyhz" is still listed in 6 Chinese markets, with more than 54 million app installs in total.

7 Discussions

7.1 Implication

Besides showing the fact that there are many FD apps, our paper also delivers the following implications:

New approaches to detect FD apps. As demonstrated experimentally, FD apps cannot be sufficiently identified by VirusTotal, showing that our community needs to introduce new automated/semi-automated approaches to detect them. The various characteristics of FD apps summarized in this work could be helpful to create such detectors. For example, one malicious developer signature usually corresponds to several company names, which could be helpful in identifying suspicious apps in the markets. Besides, the detection results of FD apps could be used to help policies and regulators to identify the fraud rings behind-the-scenes.

App Vetting Process. Second, we find that FD apps are distributed through app markets and advertising networks. These include the app markets of leading phone vendors, e.g., Huawei, and advertising networks from world-class Internet companies, e.g., Baidu and Tencent. The detection result of VirusTotal shows that most anti-virus engines cannot detect these apps. These worrisome facts urge a more effective detection schema of these apps, and a better vetting process of apps in app markets and advertising networks.

A new and yet uncovered possible business model for developing and distributing malware. Empirically, we found that many FD apps share similar code implementation (e.g., unlikely be implemented independently) while being released with different package names by different companies. We hence hypothesize that these apps are bought (cloned) from some app developers by these so-called distributors (implemented once and sold many times). This sheds light on the possible business model of other kinds of malware (e.g., ransomware), though further investigations are expected.

Ranking fraud in app markets. FD apps use ASO methods (e.g., fake positive reviews) and various channels (app markets and ad networks) for app promotion and distribution, which could offer insights for general malware detection. Moreover, it raises the implication that market operators need to apply effective means to detect and hence avoid ranking fraud.

13. 7 FD apps found in Google Play: cn.relianre, com.jshy.tongcheng, com.lingnei.kaikai, com.mmfriends, com.yuanlai.ext, com.umuad, com.xiaochen.android.fate_it

TABLE 15: The distribution of removed FD apps.

Store	# of FD Apps	# of removed FD apps(%)
Baidu Market	673	478(71%)
360 Market	177	70(40%)
Tencent Myapp	432	8(2%)
Xiaomi	91	76(84%)
Wandoujia	285	261(92%)
Huawei	308	232(75%)
Lenovo	186	52(28%)
OPPO	305	102(33%)
Meizu	512	388(76%)
GooglePlay	7	7(100%)

7.2 Ethical Consideration

Indeed, any data collected from real users need to be carefully processed. We take a series of steps to preserve the privacy of (possible) involved users/malicious developers in our data set. First, all raw data collected for this study are open to public, we do not resort to exploit the apps ourselves to collect the unpublished data, and we cooperated with app market regular departments in China to get the revenue data under the guidance of a lawyer. Second, we preserve the privacy of (possible) involved mobile users according to the Research Data Management Policy of Beijing University of Posts and Telecommunications, including

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2019.2908939, IEEE Transactions on Dependable and Secure Computing

storage of data, sharing of data, and disposal of data. Third, we have obfuscated all the user photos shown in this paper to protect the privacy of original photo owners, even though all the user avatars we listed in the paper are convinced to be fake profiles and they could be found on the public INTERNET.

7.3 Limitation

7.3.1 Identifying FD apps, fake accounts and fake reviews.

First, the method used to detect the FD apps is conservative and may miss some of them. For instance, we use the keywords and embedded in-app purchase libraries to find candidate apps. Though this method leads us to the discovery of $23,546$ candidate apps, the list of keywords and in-app purchase libraries may not be complete and could introduce false negatives. Second, we use a straightforward method to identify fake account profiles by comparing the avatar photos and account information, which may miss fake accounts that use sophisticated method. Nevertheless, our approach could identify the lower-bound of fake accounts in these apps, and more advanced approaches such as "user behavior" based approach [39] [40] could be exploited in the future work. Third, we use the heuristic to find the fake reviews. Specifically, we use the exact text matching to find the repeated reviews. This may miss the reviews that have same meanings but in different texts.

7.3.2 FD apps in other countries.

Our study is mainly focused on the apps in the Chinese app markets. Most of the fraudulent dating apps analyzed in this paper are targeting users in China, possibly due to the biased region distribution of apps in our dataset. However, we believe such kind of apps may exist in many other countries and in other languages as well, especially the places where app vetting is not strictly enforced when they are uploaded to an app market. In our future work, we plan to extend our crawler to download more apps from app markets in other countries.

8 Related Work

This paper is motivated by the work of Caballero et al. [41] and of Thomas [42], who have investigated the so-called "underground economy" associated with malicious apps (or unwanted software). FD apps fall into the same research line. To the best of our knowledge, the ecosystem of FD apps has not yet been investigated, even though it is reported in some media news [43]. Nevertheless, various studies have explored the general fraudulent behaviors in the mobile app ecosystem as well as the security aspect of dating apps.

8.1 Fraudulent Behaviors

Fraudulent behaviors have been widely explored in the mobile app ecosystem [44], [45], [46], [47], [48], [49]. The most common issue is ad fraud, where a miscreant's code fetches ads without displaying them to the user or "clicks" ads automatically [50], [51], [52], [53], [54]. For example, Crussell et al. [50] have revealed two fraudulent ad behaviors: (1) ads are requested by apps that are running in the background and (2) ads are clicked without user interaction (also known as click frauds [55]). More recently, Dong et al. [48] reveal seven types of ad frauds and further demonstrate that such ad fraudulent apps are also likely to violate the policy of app markets, resulting in risks to be removed from app markets [49]. Besides ad fraud, there are also other types of frauds disclosed by several researchers. For example, Liu et al. [46] have explored *usage fraud*, which is invented to boost usage statistics on third-party analytics like Google Analytics, resulting in inaccurate numbers that could eventually fool investors to make wrong decisions. Xie et al. [56] have analyzed the review fraud in mobile apps and found that some mobile app developers turn to the underground market to buy positive reviews. We also found in this paper that FD apps use fake positive reviews for app promotion. Our work is focusing on fraudulent dating Android apps and has revealed a new type of fraudulent behaviors, which have not been systemically studied.

8.2 Security and Privacy in Dating Apps/Online Dating Website

Dating apps have raised security and privacy-related concerns in recent years [57], [58], [59], [60], [61]. As shown by Shetty et al. [57], mobile dating apps are potentially vulnerable to security risks. For example, they have demonstrated that it is quite trivial to conduct a man-in-the-middle attack against most dating apps, resulting in private data leaks of app users. Similarly, Hoang et al. [62] argue that trilateration threatens location privacy of users of location-based mobile apps. They demonstrate that it is possible for an adversary to identify the location of an individual when she/he is using dating apps, even under the situation where location-hiding features are enabled. Similar findings have also been reported by Carman et al. [63], who have empirically presented their experiments on a popular dating app called Tinder. In addition to leaking location information directly, certain sensitive information (e.g., nearly usernames, profile pictures, messages, etc.) can also be recovered from user's devices based on the residual data generated by dating apps [58]. Our work is not towards the potential security and privacy concerns of legitimate dating apps, but revealing a new type of malicious dating apps, i.e., fraudulent dating apps.

Moreover, we want to clarify that, the fraudulent behaviors of these apps are different from the well-known *romance scam* [12], [64], [65], [66]. In particular, the fraudulent acts in romance scam are usually with the involvement of real persons, who are communicating with victims through phones, emails and try to access to victims' money or bank account. However, in FD apps, the chatbots instead of real persons, are communicating with victims. The main purpose is to lure the victim into buying premium service, not the financial information. Though, we find there are still some common aspects between them. For instance, seductive account profile avatars are used to attract victims in both romance scam and FD apps.

9 Conclusion

In this work, we perform a systematic study of fraudulent dating apps, including its characteristics, business model,

distribution networks and their impact on affected users. Our research has observed various findings that are previously unknown to the community. Due to the financial loss to victims and the fact that current anti-virus engines cannot detect most of these apps, we argue that an effective solution should be proposed to detect such apps or block the distribution of these apps in the first place to protect users.

REFERENCES

[1] W. Zhou, Y. Zhou, M. Grace, X. Jiang, and S. Zou, "Fast, Scalable Detection of "Piggybacked" Mobile Applications," in *Proceedings of the 3rd ACM Conference on Data and Application Security and Privacy*. ACM, 2013.

[2] Y. Zhou and X. Jiang, "Dissecting Android Malware: Characterization and Evolution," in *Proceedings of the 33rd IEEE Symposium on Security and Privacy*. IEEE, 2012.

[3] "Developer policy of Google Play," 2018, https://play.google.com/intl/zh-CN/about/monetization-ads/.

[4] "Google Play Billing API," 2018, https://developer.android.com/google/play/billing/api.

[5] "AliPay," 2018, https://www.alipay.com.

[6] "WeChatPay," 2018, https://pay.weixin.qq.com/index.php/core/home/.

[7] Z. Ma, H. Wang, Y. Guo, and X. Chen, "Libradar: fast and accurate detection of third-party libraries in android apps," in *Proceedings of the 38th International Conference on Software Engineering Companion*. ACM, 2016, pp. 653–656.

[8] O. Gadyatskaya, A. Lezza, and Y. Zhauniarovich, "Evaluation of resource-based app repackaging detection in android," in *Proceedings of the 21st Nordic Conference on Secure IT Systems*, ser. NordSec 2016. Springer, 2016, pp. 135–151.

[9] H. Wang, Y. Guo, Z. Ma, and X. 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*. ACM, 2015, pp. 71–82.

[10] "Fake GPS," 2018, https://github.com/xiangtailiang/FakeGPS.

[11] Y. Li, Z. Yang, Y. Guo, and X. Chen, "Droidbot: a lightweight ui-guided test input generator for android," in *2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)*. IEEE, 2017, pp. 23–26.

[12] "Romance Scams: Online Imposters Break Hearts and Bank Accounts," 2018, https://www.fbi.gov/news/stories/romance-scams.

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

[14] C. Newell, M. Wood, K. Costello, and R. Poetker, "Automatic story creation using semantic classifiers for images and associated meta data," in *U.S. Patent Application*, 2008, pp. No.11/758, 358.

[15] S. Anbalagan, "On occurrence of plagiarism in published computer science thesis reports at swedish universities," in *DiVA*, 2010, p. 68.

[16] "Python Imaging Library," 2018, http://www.pythonware.com/products/pil.

[17] "Legal person," 2018, https://en.wikipedia.org/wiki/Legal_person.

[18] "appgrowing," 2018, https://appgrowing.cn.

[19] H. Zhu, H. Xiong, Y. Ge, and E. Chen, "Discovery of Ranking Fraud for Mobile Apps," in *IEEE Transactions on Knowledge and Data Engineering*. IEEE, 2015.

[20] V. Rastogi, R. Shao, Y. Chen, X. Pan, S. Zou, and R. Ryan, "Are these Ads Safe: Detecting Hidden Attacks through the Mobile App-Web Interfaces," in *Proceedings of the Network and Distributed System Security Symposium*, 2016.

[21] "Cheetah Ad," 2018, http://ads.cmcm.com.

[22] "IntelligentTui," 2018, http://www.txzhihuitui.com/.

[23] "Tencent Social Ad," 2018, http://ads.tencent.com.

[24] "Baidu Ad," 2018, http://mssp.baidu.com/home.

[25] "Cheetah Mobile," 2018, https://en.wikipedia.org/wiki/Cheetah_Mobile.

[26] "Tencent," 2018, https://en.wikipedia.org/wiki/Tencent.

[27] "Baidu," 2018, https://en.wikipedia.org/wiki/Baidu.

[28] W. Yang, Y. Zhang, J. Li, H. Liu, Q. Wang, Y. Zhang, and D. 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.

[29] M. Sebastián, R. Rivera, P. Kotzias, and J. Caballero, "Avclass: A tool for massive malware labeling." in *International Symposium on Research in Attacks, Intrusions, and Defenses*. Springer, 2016.

[30] "Developer policy of Baidu Market," 2018, http://app.baidu.com/docs?id=18&frompos=401008.

[31] "Developer policy of 360 Market," 2018, http://dev.360.cn/wiki/index/id/18.

[32] "Developer policy of Tencent Myapp," 2018, http://wiki.open.qq.com/wiki.

[33] "Developer policy of Xiaomi," 2018, https://dev.mi.com/console/doc/detail?pId=879.

[34] "Developer policy of Wandoujia," 2018, http://aliapp.open.uc.cn/wiki/?p=140.

[35] "Developer policy of Huawei," 2018, https://developer.huawei.com/consumer/cn/devservice/doc/50104.

[36] "Developer policy of Lenovo," 2018, http://open.lenovo.com/developer/adp/helpData/database_detail.jsp?url=http://open.lenovo.com/sdk/?p=796.

[37] "Developer policy of OPPO," 2018, https://open.oppomobile.com/wiki/doc#id=10071.

[38] "Developer policy of Meizu," 2018, http://open-wiki.flyme.cn/doc-wiki/index#id?110.

[39] H. Zhang, D. Yao, and N. Ramakrishnan, "Causality-based sense-making of network traffic for android application security," in *Proceedings of the 2016 ACM Workshop on Artificial Intelligence and Security*. ACM, 2016, pp. 47–58.

[40] K. Elish, X. Shu, D. Yao, B. Ryder, and X. Jiang, "Profiling user-trigger dependence for android malware detection," in *Computers Security*, 2015, pp. 255–273.

[41] J. Caballero, C. Grier, C. Kreibich, and V. Paxson, "Measuring pay-per-install: the commoditization of malware distribution." in *Usenix security symposium*, 2011, pp. 13–13.

[42] K. Thomas, J. A. E. Crespo, R. Rasti, J. M. Picod, C. Phillips, M.-A. Decoste, C. Sharp, F. Tirelo, A. Tofigh, and M.-A. Courteau, "Investigating commercial pay-per-install and the distribution of unwanted software." in *USENIX Security Symposium*, 2016, pp. 721–739.

[43] "Ashley Madison Code Shows More Women, and More Bots," 2015, https://gizmodo.com/ashley-madison-code-shows-more-women-and-more-bots-1727613924.

[44] K.-H. Yeh, N.-W. Lo, L.-C. Chen, and P.-H. Lin, "A fraud detection system for real-time messaging communication on android facebook messenger," in *2015 IEEE 4th Global Conference on Consumer Electronics (GCCE)*. IEEE, 2015, pp. 361–363.

[45] S. de los Santos, A. Guzmán, and C. Torrano, "Android malware pattern recognition for fraud detection and attribution: A case study," *Encyclopedia of Social Network Analysis and Mining*, pp. 1–9, 2017.

[46] W. Liu, Y. Zhang, Z. Li, and H. Duan, "What you see isn't always what you get: A measurement study of usage fraud on android apps," in *Proceedings of the 6th Workshop on Security and Privacy in Smartphones and Mobile Devices*. ACM, 2016, pp. 23–32.

[47] L. Li, T. F. Bissyandé, M. Papadakis, S. Rasthofer, A. Bartel, D. Octeau, J. Klein, and Y. Le Traon, "Static analysis of android apps: A systematic literature review," *Information and Software Technology*, 2017.

[48] F. Dong, H. Wang, Y. Li, Y. Guo, L. Li, S. Zhang, and G. Xu, "Fraudroid: An accurate and scalable approach to automated mobile ad fraud detection," *arXiv preprint arXiv:1709.01213*, 2017.

[49] F. Dong, H. Wang, L. Li, Y. Guo, G. Xu, and S. Zhang, "How do mobile apps violate the behavioral policy of advertisement libraries?" in *Proceedings of the 19th International Workshop on Mobile Computing Systems & Applications*. ACM, 2018, pp. 75–80.

[50] J. Crussell, R. Stevens, and H. Chen, "Madfraud: Investigating ad fraud in android applications," in *Proceedings of the 12th annual international conference on Mobile systems, applications, and services*. ACM, 2014, pp. 123–134.

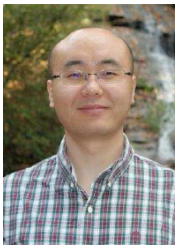
[51] T. Blizard and N. Livic, "Click-fraud monetizing malware: A survey and case study," in *2012 7th International Conference on Malicious and Unwanted Software (MALWARE)*. IEEE, 2012, pp. 67–72.

[52] R. Stevens, C. Gibler, J. Crussell, J. Erickson, and H. Chen, "Investigating user privacy in android ad libraries," in *Workshop on Mobile Security Technologies (MoST)*, vol. 10, 2012.
[53] B. Liu, S. Nath, R. Govindan, and J. Liu, "Decaf: Detecting and characterizing ad fraud in mobile apps." in *USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2014, pp. 57–70.
[54] G. Cho, J. Cho, Y. Song, D. Choi, and H. Kim, "Combating online fraud attacks in mobile-based advertising," *EURASIP Journal on Information Security*, vol. 2016, no. 1, p. 2, 2016.
[55] G. Cho, J. Cho, Y. Song, and H. Kim, "An empirical study of click fraud in mobile advertising networks," in *2015 10th International Conference on Availability, Reliability and Security (ARES)*. IEEE, 2015, pp. 382–388.
[56] Z. Xie and S. Zhu, "Appwatcher: Unveiling the underground market of trading mobile app reviews," in *Proceedings of the 8th ACM Conference on Security & Privacy in Wireless and Mobile Networks*. ACM, 2015, p. 10.
[57] R. Shetty, G. Grispos, and K.-K. R. Choo, "Are you dating danger? an interdisciplinary approach to evaluating the (in) security of android dating apps," *IEEE Transactions on Sustainable Computing*, pp. 1–1, 2017.
[58] J. Farnden, B. Martini, and K.-K. R. Choo, "Privacy risks in mobile dating apps," *arXiv preprint arXiv:1505.02906*, 2015.
[59] C. Spensky, J. Stewart, A. Yerukhimovich, R. Shay, A. Trachtenberg, R. Housley, and R. K. Cunningham, "Sok: privacy on mobile devices–its complicated," *Proceedings on Privacy Enhancing Technologies*, vol. 2016, no. 3, pp. 96–116, 2016.
[60] J. Imgraben, A. Engelbrecht, and K.-K. R. Choo, "Always connected, but are smart mobile users getting more security savvy? a survey of smart mobile device users," *Behaviour & Information Technology*, vol. 33, no. 12, pp. 1347–1360, 2014.
[61] G. Argyros, T. Petsios, S. Sivakorn, A. D. Keromytis, and J. Polakis, "Evaluating the privacy guarantees of location proximity services," *ACM Transactions on Privacy and Security (TOPS)*, vol. 19, no. 4, p. 12, 2017.
[62] N. P. Hoang, Y. Asano, and M. Yoshikawa, "Your neighbors are my spies: Location and other privacy concerns in glbt-focused location-based dating applications," *Advanced Communication Technology (ICACT)*, pp. 851–860, 2017.
[63] M. Carman and K.-K. R. Choo, "Tinder me softly–how safe are you really on tinder?" in *International Conference on Security and Privacy in Communication Systems*. Springer, 2016, pp. 271–286.
[64] J. Huang, G. Stringhini, and P. Yong, "Quit playing games with my heart: Understanding online dating scams," in *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*. Springer, 2015, pp. 216–236.
[65] A. Rege, "What's love got to do with it? exploring online dating scams and identity fraud," *International Journal of Cyber Criminology*, vol. 3, no. 2, p. 494, 2009.
[66] M. T. Whitty and T. Buchanan, "The online romance scam: A serious cybercrime," *CyberPsychology, Behavior, and Social Networking*, vol. 15, no. 3, pp. 181–183, 2012.

Yangyu Hu was born in 1991. He received the bachelor degree in applied physics from Beijing University of Posts and Telecommunications, Beijing, in 2012. Currently, he is a Ph.D student in the school of Cyberspace Security at Beijing University of Posts and Telecommunications. His research interest includes mobile application security and privacy.

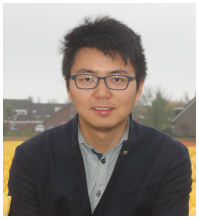
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2019.2908939, IEEE Transactions on Dependable and Secure Computing

Haoyu Wang received his PhD degree from Peking University in 2016. He is currently an Assistant Professor at Beijing University of Posts and Telecommunications. His research interests lie at the intersection of mobile system, privacy and security, and program analysis.



Yajin Zhou earned his Ph.D. in Computer Science from North Carolina State University. He is currently a Professor of Zhejiang University. His research mainly focuses on smartphone and system security, i.e., identifying real-world threats and building practical solutions. Currently He is working on interesting smartphone security projects.

Yao Guo received his PhD in Computer Engineering from University of Massachusetts at Amherst in 2007. He is a full Professor in the Institute of Software of the School of Electronics Engineering and Computer Science at Peking University. He has served as Vice Chair of Computer Science since 2013. His general research interests include operating systems, mobile computing and applications, low-power design and software engineering.



Li Li is a Lecturer at the Faculty of Information Technology, Monash University. Prior to join Monash, he was a research associate in Software Engineering at the University of Luxembourg, where he obtained his PhD degree in 2016. His research interests are in the fields of Android security, static code analysis, and machine learning. Dr. Li received a Best Paper Award at the ERA track of IEEE SANER 2016 and the FOSS Impact Paper Award at MSR 2018.

Bingxuan Luo is a student at Beijing University of Posts and Telecommunications. She was a research assistant at Professor Haoyu Wang's group. Her research interest is mobile security.

Fangren Xu is a student of Rivermont Collegiate. He was a research assistant at Professor Haoyu Wang's group. His research interest is mobile security.