2016 IEEE 22nd International Conference on Parallel and Distributed Systems

Towards a Framework to Facilitate the Mobile Advertising Ecosystem

Gong Chen[†], Shouling Ji[‡,†], and John A. Copeland[†]
[†]Georgia Institute of Technology, U.S.A.
[‡]Zhejiang University, China
gong.chen@gatech.edu, sji@gatech.edu, jcopeland@ece.gatech.edu

***Abstract*—To date, app developers are allowed to monetize their apps in two services: *in-app advertising* and *in-app billing*. Of these two, in-app billing is not prevalently used by users, whereas in-app advertising is considered an important funding source for developers. However, this service incurs a number of criticisms: (1) users must passively receive all mobile ads while using apps, (2) users get nothing from viewing or clicking ads, (3) ad networks transfer user private information to remote servers in an unencrypted format without user consent, and (4) negative impressions brought from irrelevant ads may harm the advertised brands. To overcome these problems, we propose *In-App AdPay*, a framework that combines the advantages of "in-app advertising" and "in-app billing" together so that ad networks can overtly ask users' permissions in order to serve more tailored ads, but in return, advertisers will pay targeted users' virtual transactions within the app (e.g., coins in mobile games) via a secure channel. While mobile users can be brought into the monetization loop, it will be technically and legitimately easier for ad networks to study users. We implemented the proof-of-concept framework and conducted a test with 42 volunteers. Based on these studies, we believe that "In-App AdPay" would balance user privacy and user experience without interfering with the existing monetization arrangements. Lastly, we reveal how tracked-by-consent users react in different test scenarios and value the permissions used in ad libraries.**

***Index Terms*—Mobile Advertising, Usable Privacy, Usability Testing & Surveys, User Interface, Unpaired Two-Sample T-Test**

I. Introduction

As of 2016, smartphone markets in the United States and all over the world are still prosperous. Likewise, mobile applications (apps) are widely prolific, and are usually developed by third-party developers and downloadable from official and/or unofficial app marketplaces. Although developers can sell their apps on app stores, most of them merely offer their apps for free. As a result, app developers generally make money through a combination of *in-app advertising* and *in-app billing*. In-app advertising has led the mobile ecosystem since 2009. After embedding one or more ad libraries provided by ad networks within their apps, developers may consistently earn money from the apps. According to the Interactive Advertising Bureau (IAB) [1], in-app mobile advertising in the United States totaled $20.7 billion and increased 65.6% in FY 2015 from the previous year. In 2011, Google came up with the in-app billing service modeled after Apple's iOS [2]. In-app billing improves user experiences by helping them purchase digital goods, either consumables or non-consumables. For mobile apps in Google Play, the in-app product costs range between $0.99 and $399.99. Actually, service providers of in-app billing are not limited to those mobile OS providers. For instance, TapJoy [3] also offers this monetization option. Most developers are eager to include the in-app billing service. According to [4], Candy Crush got over $1.3 billion from in-app billing in 2014.

However, according to public media [5] and our own interviews with 43 volunteers, users usually feel somewhat or even very uncomfortable with current advertisements (ads). Also, users may get annoyed by unsolicited ads and develop negative impressions towards advertised brands. As a result, AdBlock receives a high rating with over 500 million downloads [6]. As for in-app billing, it may not be widely accepted by mobile users. According to the survey from [7], over 80% of the surveyed developers estimate that no more than 10% of users make even one in-app purchase. [8] shows merely 0.15% of gamers might be delivering 50% of the revenue through in-app billing. [9] reveals a steady decline in number of in-app billings within popular games (e.g., Candy Crush).

Due to these facts, we develop In-App AdPay, a new monetization service to let users actively trade their information upon their consent via secure connections. In our framework, we not only minimize changes to the current ecosystems but also keep the existing monetization arrangements. Moreover, users are allowed to actively request mobile ads for getting digital goods. Meanwhile, in order to serve targeted ads, ad networks are allowed to explicitly track users with consent. Lastly, while ad networks get the same share from what developers earn as the existing models, mobile users can also benefit from viewing or clicking ads with virtual goods in our service. After implementing the monetization framework on Android, we conducted usability testing with 42 people from the same volunteer pool. Based on their perceptions and expectations, we found that In-App AdPay would harmonize the mobile monetization ecosystem. However, according to the feedback, over 70% of these 42 participants feel uncomfortable with giving up their private information. Therefore, we further investigated how advertisers could induce users to actively trade their information and get more tailored ads. We evaluate other factors (e.g., ad clicks) that affect user decisions. Particularly, our contributions include:

- We conducted a survey related to "in-app advertising" and "in-app billing" with 43 adult volunteers. We find that most participants have the same negative impression

1521-9097/16 $31.00 © 2016 IEEE
DOI 10.1109/ICPADS.2016.66

toward mobile ads in general, but share relatively diverse viewpoints to targeted ads. Moreover for the users in our sample, in-app billing is not popular.

- We implemented a prototype monetization service—In-App AdPay. With a few minor changes on the ad network side in the current "in-app advertising" framework, In-App AdPay not only allows users to actively take a "virtual" share from viewing and/or clicking ads, but also lets ad networks avoid liability for collecting users' private information. Meanwhile, with our framework, no change is necessary for advertisers: use the same console as before and pay the same amount of money as usual. However, for the sake of financial incentives, app developers would be motivated to secure network connections between mobile users and ad networks. As a result, more tailored ads will be served via secure connections with users' consent. We believe that In-App AdPay will facilitate the mobile monetization ecosystem.
- We studied feedback and analyzed user data. While receiving relatively positive feedback from these users, we recognize that people may still feel uncomfortable when actively giving up their private information. Therefore, we study how advertisers can induce users to trade their information with virtual goods, and then provide a quantitative value of each Android permission.

The rest of the paper is organized as follows. We describe the motivation related to in-app advertising and in-app billing, including workflows and user opinions, in Section II, followed by its prototyping in Section III and evaluation results in Section IV. After that, Section V discusses the benefits from In-App AdPay and the potential limitations and countermeasures for the framework. We provide related work in Section VI and conclude this paper in Section VII.

II. Motivation

A. Background

In-app advertising is popular in free apps. In this model, an ad network can aggregate app developers and advertisers to serve ads for users. Once the app developer's publisher ID is enclosed within the app, mobile ads can be refreshed every 12 to 120 seconds with a default rate of around 60 seconds [10] [11]. As a relatively new media for advertising, the rates of different mobile ads may depend on various factors (e.g., pricing model, device type, ad inventory, ad category, user value and time). For example, a newly registered user may cost an average of $9.54 for advertisers [12]. Also, the click-through rates (CTRs) are below 2.5% and varies based on devices and time [13]. Figure 1 depicts the in-app advertising workflow. As for money, it flows from advertisers to app developers through a designated ad network, which takes about 30% of the entire fund.

In-app billing is flexible for users, which allows a payment agent to coordinate users with app developers. When opening the in-app billing portal, users may have four options to make purchases (i.e., add credit or debit cards, add paypal, enable carrier billing, and redeem gift card). It can even replace app

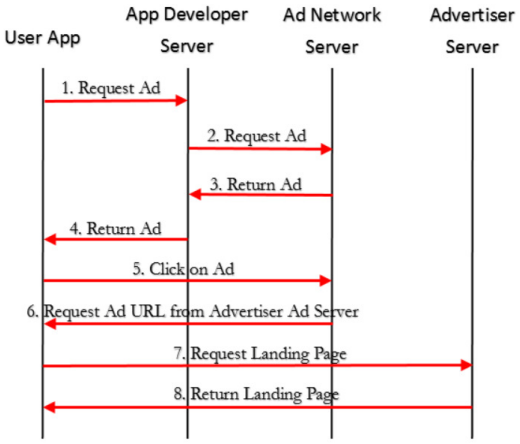

Fig. 1. Workflows of In-App Advertising

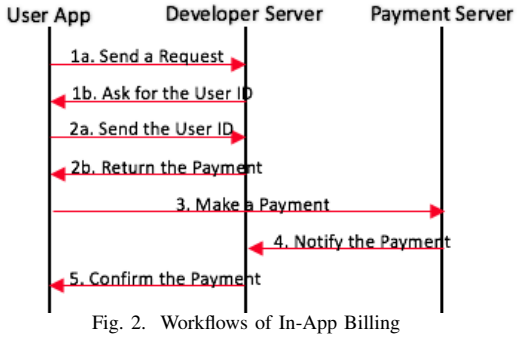

Fig. 2. Workflows of In-App Billing

store purchase in the future because of the "remove ads" option. The only reason that app store purchases are still there may be because that is a passive way to protect "app repackaging". In-app billing usually offers either consumable (e.g., virtual coins, props, or cloud storage for one month) or durable (e.g., new game level) digital transactions. No matter what type of digital purchase a user chooses, the payment workflow behind the scenes is always conducted via HTTPS, as depicted in Figure 2. Basically, users pay digital transactions to the related app developers, whereas the payment agent takes a commission fee of about 30% of the total amount.

B. User Opinions

In order to learn users' opinions (e.g., user satisfaction) on the two existing app monetization models, we conduct a survey with 43 adult volunteers, including students, scholars and staff, who have basic knowledge of smartphone usage. These respondents, aged from 18 to 45 from 10 countries across 4 continents, are either tech-savvy or lay people. The survey about user preferences on in-app advertising and in-app billing contains "Yes or No" questions, multiple choice questions, and open-ended questions. Every participant has signed the consent form for the current surveys and the subsequent studies related to In-App AdPay. All user-related studies were approved by our Institute Review Board (IRB).

Table I shows user answers towards in-app advertising. While mobile ads in general mostly receive negative impressions from users, around 20% of the respondents feel comfortable with personalized ads. But, none of them are glad

TABLE I
USERS' CHOICES ON IN-APP ADVERTISING

mobile ads in general	Feel	very comfortable	1
		somewhat comfortable	1
		neutral	15
		somewhat uncomfortable	18
		very uncomfortable	8
	Click	yes (intentional)	6
		yes (accidental)	14
		no	24
tailored ads	See	yes	13
		no	29
		unanswered	1
	Feel	very comfortable	1
		somewhat comfortable	8
		neutral	11
		somewhat uncomfortable	14
		very comfortable	8
		circled both somewhat's	1
	Click	yes	17
		no	25
		circled both	1
	Collect Info	neutral	7
		somewhat uncomfortable	12
		very uncomfortable	24

TABLE II
USERS' CHOICES ON IN-APP BILLING

Overall	very comfortable		6
	somewhat comfortable		6
	neutral		16
	somewhat uncomfortable		11
	very uncomfortable		4
Ever Used (14)	Single Transaction	<$1	2
		$1-$5	8
		$5-$10	2
		>$10	1
	Total Spending	<$20	9
		$20-$50	4
		>$100	2
Never Used (29)	will consider		9
	won't consider		20

to be tracked. Generally, one or more of the five attitudes is expressed by the participants:

- Accept: *"If it's tailored for me, maybe I'll click."* (P15)
- Understand: *"..., but I understand they're necessary for the survival of free apps."* (P38)
- Neglect: *"I don't pay attention to mobile ads."* (P43)
- Dislike: *"It's a little bit annoying when ..."* (P34)
- Counteract: *"I use ad blockers."* (P30)

Moreover, users' feelings on in-app billing are quite diverse. Table II shows that less than 1/3 of the respondents have utilized this monetization service, in which most of their single transactions are below $5. In-app billing is mainly used to unlock additional content (e.g., P26 and P30) or remove ads (e.g., P24). Among the users who have never used in-app billing in our sample, more than 2/3 may not consider the option. These users may *"not trust"* the service (e.g., P25) or simply *"don't use apps that require it"* (e.g., P9). Also, P36 mentions that *"I've seen a lot of comments about in-app purchases not working and people losing money"*.

According to these results, there is potential to improve both of the current monetization services from two different aspects. In-app advertising may focus on making users feel comfortable, whereas in-app billing can be used to increase the client pool. Therefore, in our proposed framework, one of our objectives is to ameliorate these two deficiencies.

C. Ad Library Permissions

Both Android and iOS devices require permissions for serving mobile ads. The difference between these two platforms is whether to explicitly declare permissions within apps. Android has a fine-grained permission system, which consists of over 130 permissions. In order to collect private information from mobile users, app developers are required to implant ad libraries with specific permissions, ranging from location to accounts. However, users are not able to distinguish between granted permissions that are used by the apps themselves, the mobile ads, or both. Therefore, here we focus on identifying the permissions usually asked by ad libraries.

Previous research works [14] [15] [16] [17] [18] listed over 25 third-party libraries, among which 14 are still in use with documentation in English[1]. Ad libraries usually ask specific permissions to work properly. Based on Mobile Rich Media Ad Interface Definition (MRAID) 2.0, about 21 permissions[2] may be utilized in Android. While Vdopia asks for GET_ACCOUNTS, InMobi lists 11 permissions in the current documentation (14 permissions when we checked in mid-2015). The INTERNET permission is undoubtedly always required by all ad libraries, and the Access_Network_State permission also comes along in most cases. Therefore, there are 19 ad-related permissions optionally used for ad libraries.

While the permissions are asked in compliance with MRAID 2.0 for interactive rich media ads, most fall into the categories (i.e., accounts, location, phone status and identity, non-free services, file storages, hardware, and personal information) that users should be wary of [19] [20] [21].

After replacing the Android ID with the user-resettable advertising ID in 2013, users are able to select "Opt-out of Interest-based ads" within the Google Settings app to prevent receiving tailored ads across the device. Ad networks other than Google also enable such a feature. For example, InMobi uses the TRUSTe Ad Privacy Manager to provide in app opt-out [22].

III. FRAMEWORK

The preliminary studies motivate us to come up with a more user-friendly mobile monetization framework – In-App AdPay. In this section, we elaborate on our design philosophy and discuss the implementation details.

A. Methodology

When we look into the current monetization services, both models let mobile apps contact a third-party to initiate the designated service. But in in-app billing, all transactions are

[1]AdMob, Mopub, Vdopia, Mocean, Buzzcity, AppLovin, leadbolt, Mobfox, Smaato, Greystripe, MobClix (now Axonix), MillennialMedia (acquires jumptap), InMobi, Airpush

[2]Internet, Access_Network_State, Vibrate, Write_Contacts, Record_Audio, Get_Accounts, Read_Phone_State, Call_Phone, Write_External_Storage, Read_Calendar, Write_Calendar, Activity_Recognition, Camera, Wake_Lock, Read_Logs, Access_Coarse_Location, Access_Fine_Location, Send_SMS, Access_WiFi_State, Change_WiFi_State, Read_History_Bookmarks

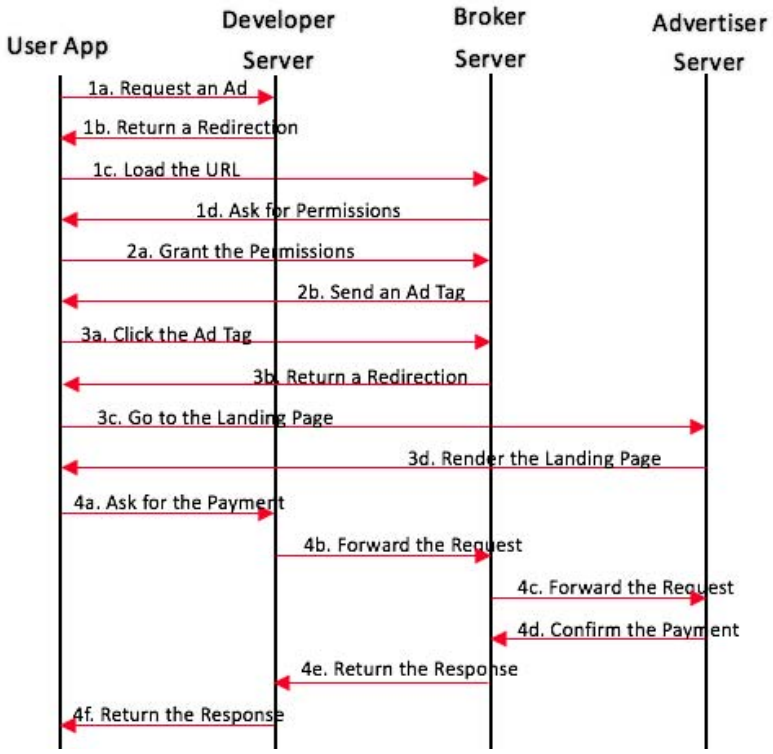

Fig. 3. Workflow of In-App AdPay

completed on the third-party side (i.e., the payment agent); whereas in in-app advertising, a fourth-party (i.e., an advertiser) will be selected by the third-party (i.e., the ad network) to serve ads for end users. While either ad networks or payment agents get a share from what app developers earn, none of the alternatives carry benefits to end users. Rather, in-app advertising may antagonize users who disagree with ad-supported apps that track their information without consent, which may potentially cause harm to the advertised brands.

The basic idea behind In-App AdPay is to let users get paid with virtual goods after viewing/clicking a tailored ad from a selected advertiser. In order to ensure ad personalization, users have to actively request ads but surrender their private information with consent. Given that both ad networks and payment agents are essentially brokers, we are able to combine the two roles into one in In-App AdPay. Moreover, financial incentives motivate the app developers to work with every ad network adapting In-App AdPay for securing the network connections between users and ad networks, although the vast majority of mobile ads served to AdMob have been served via HTTPS since 2015 [23]. While money still flows from advertisers to app developers, users are able to get paid with virtual goods from app developers. Lastly, although server redirection is mainly used in both existing services for controlling connection speeds, we may still adopt a few client redirections in our design to increase simplicity. It takes four steps for users to complete such a transaction: (1) after getting into the portal, the user can select an option, (2) the user must grant the requested permissions to get a tailored ad, (3) the user can click the ad to get into its landing page, and (4) when the user returns from the landing page, the payment will automatically be allocated. Figure 3 depicts the flowchart in detail.

B. Implementation

According to Figure 3, our implementation consists of three *node.js* servers providing an app developer, an ad network, and 14 advertisers[3], respectively, as well as an in-app user interface with only minimum required features to conduct further experiments. Each server runs its own *MongoDB* database and follows the REST architectural style for all HTTP/HTTPS connections. While the servers are hosted on a Mac mini (with OS X) and two PCs (one with Windows 10 and another with Ubuntu 14.04 Desktop), the user interface is implemented on a Moto X (with Android 5.1). All devices are connected by a switch and a wireless access point within a local network.

In order to complete a transaction, four steps are required: (1) once a user makes an ad request, the request is logged in the developer's server and redirected to the ad network's server which asks for permissions via the In-App AdPay portal; (2) once the user grants the permissions, a personalized ad tag is sent; (3) once the user clicks the ad, a landing page is rendered; (4) a payment request is automatically sent and then verified by remote servers. Also, the first three steps are manually handled by mobile users. Furthermore, five points are emphasized in our design, as shown in Figure 3. First, in step 1d, user consent is explicitly requested before

[3]We consider all advertisers' simple landing pages are hosted on the same server without a backend.

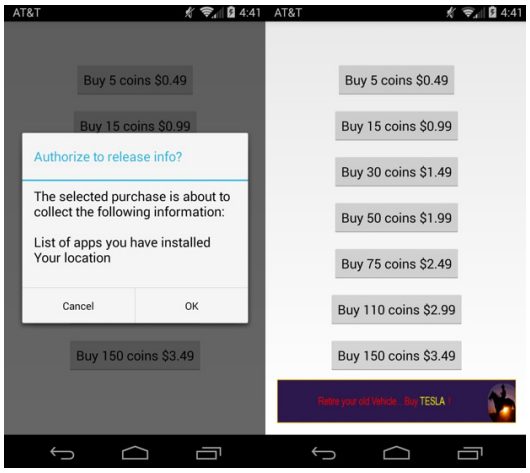

Fig. 4. User Interfaces

releasing private information. Second, in steps 1b and 3b, the developer and the ad network log transactions respectively during the redirections. Afterwards, in steps 2a and 4a, HTTPS connections are used when granting the permissions to the ad network and when asking for payment from the developer. Moreover, steps 4c and 4d are optional only if the advertiser adapts the Cost-Per-Action model[4]. Finally from step 4b to step 4e, server-to-server communications are performed during the long-polling phases.

We include both reasonable-case and worst-case scenarios simultaneously in our implementation. Figure 4 depicts the in-app user interface with seven price options between $0.49 and $3.49. These reasonable prices are set in regards to relatively low revenues that app developers can earn for a single click, since users' information is worth as much as advertisers are willing to pay for. When a user clicks on one of the buttons, a dialog box pops up to request permissions (as shown on the left of Figure 4). The higher the payment asked by the user, the more ad-related permissions that are asked. The ad network may also explicitly ask users for any permissions that advertisers are interested in – the exception being Internet and Access_Network_State, which are only for Internet connectivity. For the worst case, instead of using permission groups newly adopted in Google Play, we allow users to see each permission description. After granting the permissions, a randomly selected ad tag (size 468×60) is displayed on the bottom (as shown on the right of Figure 4).

IV. Evaluation

After implementing the framework, we conducted usability testing with 42 volunteers, which are evenly divided into 7 groups. These participants were asked to repeatedly select price options and decide whether to grant random Android permissions. All the users' actions and decisions are logged in the background with their consent. After analyzing the participants' feedback from a group, we adjusted our test scenario for the next group. Finally after all tests, we recognize that although user satisfaction rises, users still feel uncomfortable with information collection. Furthermore, after studying the data from our evaluators, we identify which permissions users care most about and how users can be enticed to give advertisers access to them.

[4]Advertisers pay higher than ad view/click, but only if a user conducts a closed sale or a particular action at the moment.

TABLE III
Seven Test Scenarios

Group	Times	No. of Permissions	Background	Button Permutation	No. of Permissions per Button	Click Tracking
1	30	37	white	yes	B1:1P, B2:2P, ..., B7:7P	no
2	50	37	white	yes	idem	no
3	50	19	Skype	no	idem	yes
4	50	19	white	no	idem	yes
5	50	19	white	no	B1&B2:1P, B3&B4:2P, B5&B6:3P, B7:4P	yes
6	50	19	white	no	idem	yes
7	50	19	white	no	1P/B	yes

A. Usability Testing & Data Collection

According to [24], five users per usability test is sufficient to assess the framework's usability. Therefore, for our 42 volunteers, we set six to a group. Each specific test scenario is conducted in each group. Moreover, for each button, the permissions are randomly selected with Fisher-Yates shuffle from our database. The seven scenarios and their differences are shown in Table III[5].

For Group 1 and Group 2, in order to prevent users from making their decisions in advance, seven buttons are re-permutated for each round. For the buttons in these two groups, Button 1 (i.e., $0.49) renders 1 permission, Button 2 (i.e., $0.99) renders 2 permissions, Button 3 (i.e., $1.49) renders 3 permissions, and the other four buttons follow the same rule. Both groups include 37 permissions, including the 19 permissions discussed in Section II-C along with another 18 similar permissions we picked out. Instead of the permissions themselves, their official descriptions are displayed in the user interface. While we start tracking ad clicks from Group 3, we set a blue background with the Skype icon in Group 3. Group 4 is the same as Group 3 except for the background. Group 5 and Group 6 have the same number of permissions per button in both cases. In Group 7, each button renders only one permission. All logs are collected within the mobile device. Each log consists of which button is clicked, what permissions are requested, time taken to make a decision, and the decision.

TABLE IV
USER OPINIONS ABOUT IN-APP ADPAY

User Perceptions	In-App AdPay	very comfortable	5
		somewhat comfortable	18
		neutral	8
		somewhat uncomfortable	9
		very uncomfortable	3
	Advertiser	very comfortable	17
		somewhat comfortable	18
		neutral	4
		somewhat uncomfortable	3
		very uncomfortable	1
	Ads	still memorized	25
		not memorized	18
	Permissions	still memorized	34
		not memorized	9
	Permission Description	helpful for decision	38
		helpless for decision	4
User Expectations	Still spend money	Yes	20
		No	22
	Will use In-App AdPay	Yes	20
		No	22
	Still uncomfortable with info collection	Yes	31
		No	11

B. Opinions & Results

After finishing usability tests with 42 volunteers, we conduct a survey related to In-App AdPay. Table IV depicts user opinions, including perceptions and expectations, related to our framework. Unsurprisingly, user perceptions were favorable: while more than half of the evaluators were comfortable with In-App AdPay, over 90% felt comfortable with advertisers. For example, P2 thinks *"it would work for majority of people"*, P24 feels that the service *makes every party feel incentive*, and P38 points out that *it's a nice idea to get money from the advertisers in a trade for some personal information as long as the users is clearly informed of which information is being exchanged*. It also shows that nearly 60% recalled ads and over 80% memorized permissions they have encountered during the test, and over 90% made decisions according to the permission descriptions. However, we find that over 70% of the participants are uncomfortable when consenting to collect their private information. After looking into their comments, we find that:

-P1: *"I don't want to sacrifice my privacy for money."*

-P22: *"I think anything more than approximate location is invasive. I like how it gives clear control of the permissions and how you can "earn" money for virtual transactions."*

-P29: *"I was unlikely to agree if I was uncertain of the permission's impact."*

-P26: *"I gave permissions that I thought were okay in keeping my privacy intact. It is inevitable in our world now. I would not mind as long as I feel that they respect my privacy to a certain level. I think it is a good experience given that some companies get these private information from people without paying them anyways."*

We believe that users may not wish to surrender their private information, but everyone sets prices for different categories of private information. Therefore, we conducted a log analysis

[5]B: Button, P: Permission

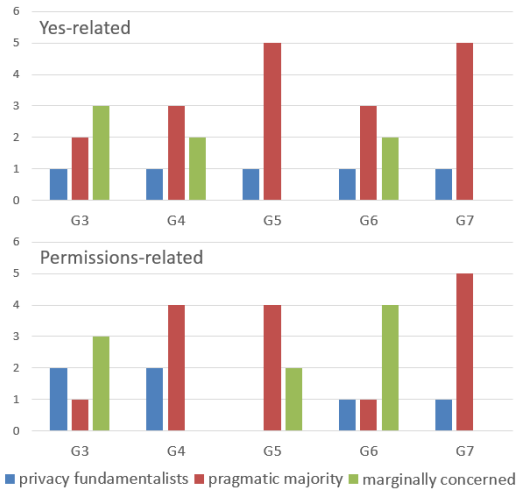

Fig. 5. Individual Differences

to reveal how users value their private information in different scenarios. The results will give advertisers a better way to induce users without violating the policies regarding data privacy. Moreover, all groups have slight differences, except Group 5 and Group 6 are the same. However, since Group 1 and Group 2 contain permissions not related to mobile ads, we focus on the other 5 groups.

Individual Differences Individuals do not view privacy uniformly. As used in [25], we classify our volunteers into three separate clusters: the privacy fundamentalist, the pragmatic majority, and the marginally concerned. While the privacy fundamentalists find it extremely unacceptable to give up their private information, the marginally concerned individuals feel indifferent. The pragmatic majority group falls in the middle. See Figure 5. On the left side, the volunteers with less than 10 "Yes" are considered as the privacy fundamentalists and the one with more than 40 "Yes" are treated as the marginally concerned. Whereas, on the right side, the privacy fundamentalists select less than 6 permissions, but the marginally concerned people allow more than 16 permissions. We notice that different settings (i.e., number of permissions per button) trend towards two opposite directions for user choices: When users are asked to grant more permissions for higher prices, as shown in Group 3 and Group 4, "Yes"-related demographics move towards privacy fundamentalists in permissions-related demographics. However, when asking for less permissions, as shown in Group 5 and Group 6, "Yes"-related demographics move towards the marginally concerned side in permissions-related demographics. As Group 7 asks only one permission per button, there is no significant difference between "Yes"-related and permissions-related demographics.

Influences of Trustful Apps Group 3 and Group 4 are the same, except for having different backgrounds: the volunteers in Group 3 were directed to use a service within Skype. We consider the null hypothesis (H_{0a}): *There is no difference in user satisfaction with advertisers between Skype background and white background*. We quantize user satisfaction into 5

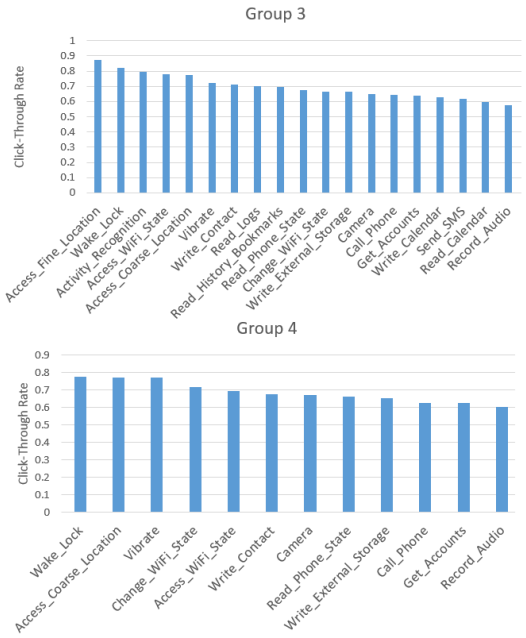

Fig. 6. Different Backgrounds (left: Skype, right: white)

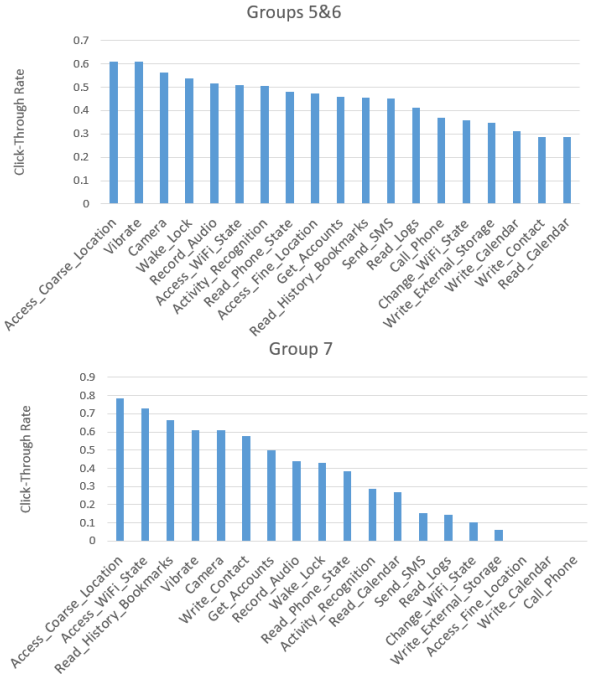

Fig. 7. Number of Permissions per Button (left: 1-4, right: 1)

values (i.e., very comfortable: 1, somewhat comfortable: 0.75, neutral: 0.5, somewhat uncomfortable: 0.25, and very uncomfortable: 0) and then weight the results with ad memorization (i.e., still memorized: 1.25, and not memorized: 0.8). An unpaired two-sample t-test[6] shows suggestive evidence that the null hypothesis does not hold (t=-1.84, p=0.095599, two-tailed). Therefore, we deduce that users may be more satisfied in using the service within a well-known app like Skype.

As we observe that in Group 3 and Group 4, there are more privacy fundamentalists in permissions-related than "Yes"-related demographics. We focus on the differences when using the service within a well-known app like Skype and an unknown app, as depicted respectively in Figure 6. In Group 4, users are more reluctant to click on higher paid options with granting more permissions. As a result, a few permissions do not appear due to the test's randomization. However, we cannot find any evident relation between granted permissions and their CTRs (i.e., ad clicked over ad requested). Therefore, users may be more satisfied with advertisers by granting more permissions for In-App AdPay within a well-known app.

Relations between Permissions and Prices/CTRs Group 5 and Group 6 are exactly the same, which render 1-4 permissions when a button is clicked. Therefore, we consider the two groups as a whole (i.e., 12 people). As for Group 7, it shows only 1 permission per button click. In order to evaluate user satisfaction with advertisers in the groups asking for less permissions, we set Group 4 as the benchmark for the null hypothesis (H_{0b}): *There is no difference in user satisfaction with advertisers when providing different number of permissions per button click*. The unpaired two-sample t-test (i.e., Group 4 vs. Groups 5&6) surprisingly fails to reject the null hypothesis (t=-1.71616, p=0.105457, two-tailed) due to an individual who answers "somewhat uncomfortable" and "not memorized". When we exclude this individual's answers from our sample, there is a significant evidence that the null hypothesis does not hold (t=-3.15544, p=0.006539, two-tailed). Therefore, user satisfaction with advertisers in Groups 5&6 is different from that in Group 4. As for Group 7, it fails to reject the hull hypothesis (t=-0.03197, p=0.976657, two-tailed) when comparing with Group 4. Accordingly, three points are highlighted: (1) it's unavoidable to bring a high variance into usability testing by extremists, (2) it shows that 1-4 permissions per button click makes most users satisfied with advertisers, and (3) one permission per button click does not help increase user satisfaction with advertisers from the original settings in Group 4.

Figure 7 shows the CTRs of ad-related permissions in Groups 5&6 and Group 7. While the CTRs in Figure 6 are all above 50%, we witness a more diverse distribution in Figure 7. Specifically, the CTRs of 7 ad-related permissions are below 20% (i.e., Send_SMS, Read_Logs, Change_WiFi_State, and Write_External_Storage) and 0% (i.e., Access_Fine_Location, Write_Calendar, and Call_Phone) in Group 7. While most permission rankings in terms of CTR are changed between the two charts in Figure 7, Access_Coarse_Location remains first in ranking but gets a higher CTR in Group 7, and the CTR of Read_Calendar keeps within the same range. Since Group 7 allows only one permission per button click, we believe that the CTRs in Group 7 better reflect the actual user preferences for different ad-related permissions. However, we also deduce

[6]We use two separate sets of unpaired samples for a statistical hypothesis test in which the test statistic follows a Student's t-distribution under the null hypothesis. It can be used with extremely small sample sizes [26].

that the strategy of rendering permissions in a combination increases the overall CTRs and gets more user information.

Table V displays the median, mean and standard deviation of each permission's prices evaluated by the volunteers in Figure 7. Similarly, while one permission per button click in Group 7 may reflect the actual user viewpoints on each permission, the strategy of rendering permissions in a combination in Groups 5&6 results in a lower users' permission-value expectation, which helps advertisers develop a better approach to allocate their budgets when using In-App AdPay.

V. Discussion

In this section, we discuss the current situation in mobile advertising from three aspects (i.e., standard, network and operating system). Afterwards, we discuss three important questions. One, how can the "In-App AdPay" framework benefit every entity in the whole ecosystem? Two, what are the limitations of our prototype? And, three is whether there any countermeasures for these limitations?

MRAID, led by the IAB, defines a common API for mobile rich media ads with HTML5 and JavaScript. It enables ad makers to build rich media ads capable of running across different mobile apps. MRAID v1.0 and v2.0 were consecutively released in 2011 and 2012. However, in order to fully unlock the functionalities, an MRAID compliant SDK in Android has to add permissions such as RECORD_AUDIO or WRITE_EXTERNAL_STORAGE. On the mobile side, Google made a few modifications on Android related to mobile ads, directly (e.g., Google launched "HTTPS Everywhere" on mobile ads in 2015 [23].) or indirectly (e.g., Users can only view at most 17 groups of related permissions and no longer see the INTERNET permission when installing an app since Android 4.4 [27].). Furthermore, Android 6.0 is allowed to enforce permissions at runtime. All permission-related adjustments make it possible to design a more fine-grained authorization framework.

A. Benefits

Admittedly, a new advertising framework can be adapted only if it is not harmful for any role in mobile advertising. Also, it will be even a plus if the solution brings benefits to each player in the ecosystem, including users. In the current framework, money flow starts from an advertiser, passes through an ad network and then ends at a developer. This still happens in In-App AdPay: the advertiser still pays the same amount for an ad, through the ad network, to the designated app developer. However, in our model, the app developer should offer the user in-app virtual credits (e.g., virtual coins in a game). As these in-app virtual credits are generated by app developers for free, nothing has been changed with regards to money flows. Technology-wise, no change is necessary for advertisers (i.e., use the same console & pay the same amount), and there are only subtle modifications (i.e., combine roles of ad networks and payment agents) on the ad network side with a renewed ad library for app developers.

However, In-Ad AdPay comes up with additional benefits for different sides:

- User: (1) users are brought into the monetization loop, (2) users are allowed to actively trade their private information upon their consent, and (3) users can still get personalized ads via HTTPS.
- Advertiser: positive impressions from users may help the advertised brands.
- Ad Network: (1) ad networks will not face any potential ethical and legal issues when collecting user information and serving personalized ads, and (2) secure channels are automatically applied to secure virtual transactions.
- App Developer: (1) financial incentives motivate app developers to work with ad networks to secure connections, and (2) in order to earn more from mobile ads, developers expect to have a higher CTR. Although a solid measurement must be tested on the app marketplace, our primary test results show a leap in CTR.

B. Limitations

Three kinds of limitations may occur: experiment-related, technique-related and human-related.

While In-App AdPay concentrates on mitigating the existing issues of in-app advertising (e.g., bad impressions on advertisers), it may potentially cause users to neglect in-app billing's low price options. Moreover, although it is statistically sufficient to cover different test scenarios with merely 42 volunteers, it would be better to recruit more participants for in-depth studies for assessing the optimal value of the population's private data more precisely.

Two technical issues cannot be resolved by In-App AdPay: (1) click fraud initiated by app developers, and (2) inappropriate data collection conducted by ad networks. In order to increase their income, malicious apps may programmatically trick clicks on mobile ads. Whereas, in our framework, unless the privilege de-escalation for ad libraries is implemented, ad networks can still request all data granted by app permissions.

Our monetization framework gives users great flexibility to control ad requests. Therefore, we expect a significant reduction in accidental ad clicks. However, for the sake of earning "virtual" coins, users may intentionally request and then click ads, which results in burning advertisers' budgets. Also, users' private information (e.g., geolocation) may be changed over time. Furthermore, we also find three concerns from users' comments: (1) several users (i.e., P3, P8, P27 and P30) do not want their phones out of control until the service has been trusted, (2) P1 prefers to opt in permissions, and (3) P8 wants to consider only higher payment and fewer privacy-related permissions.

C. Countermeasures

Both technical problems discussed previously have some solutions provided by other researchers. In Android, fraudulent developers are able to programmatically forge click events within their apps to increase ad revenue. In-App AdPay may also suffer from such attacks. To mitigate this issue, AFrame [28] and LayerCake [29] allow embedded user interfaces, such as ad libraries, to run as a separate process within an app. In AFrame, developers place ads in an isolated region running a

TABLE V
USERS' PRICE EXPECTATIONS ON AD-RELATED PERMISSIONS

Permission	Median		Mean		Stdev	
	Groups 5&6	Group 7	Groups 5&6	Group 7	Groups 5&6	Group 7
Access_Coarse_Location	0.890833	2.24	0.858594	2.04	0.10736	0.647109
Access_Fine_Location	0.86	2.156667	0.868629	1.698333	0.103363	0.909059
Access_WiFi_State	0.86625	2.99	0.87694	2.04	0.090729	1.30384
Activity_Recognition	0.8535	2.406667	0.808394	2.406667	0.139274	1.532065
Call_Phone	0.834347	N/A	0.861472	N/A	0.09486	N/A
Camera	0.89299	2.49	0.853298	2.61	0.147533	0.544977
Change_WiFi_State	0.7575	2.49	0.720243	2.49	0.170563	0
Get_Accounts	0.805	1.74	0.750668	2.003889	0.157695	0.530875
Read_Calendar	0.950764	2.24	0.937623	1.906667	0.069279	1.2829
Read_History_Bookmarks	0.884427	1.49	0.886677	1.79	0.094897	0.67082
Read_Logs	0.923542	2.24	0.932054	2.24	0.057893	0.353553
Read_Phone_State	0.8725	1.99	0.905853	1.865	0.062061	0.629153
Record_Audio	0.89996	2.281667	0.890471	2.281667	0.102696	0.648181
Send_SMS	0.838125	0.99	0.840602	0.99	0.105888	0
Vibrate	0.844333	2.49	0.866509	2.156667	0.119822	0.874007
Wake_Lock	0.897292	1.24	0.872635	1.656667	0.106897	0.946485
Write_Calendar	0.913889	N/A	0.926364	N/A	0.043033	N/A
Write_Contact	0.921094	0.49	0.8518	0.49	0.185564	0
Write_External_Storage	0.995	N/A	0.969583	N/A	0.043537	N/A

different process from the main activity, so that the host app cannot click on the ad activity. LayerCake uses the Android WindowManager to display the ad activity in a new window on top of the window of the main activity. Also, AdAttester [30] uses ARM's TrustZone to make sure unforgeable clicks and verifiable display. Furthermore, MAdFraud [17] automatically runs apps to trigger and expose two fraudulent ad behaviors, including clicking on ads without user interaction. In order to prevent unintentional data disclosure, since "In-App AdPay" allows ad networks to collect private information under user consent, we also need solutions, such as TaintDroid [31] to monitor sensitive information on smartphones with dynamic taint analysis. Furthermore, after a user grants permissions, methods such as VetDroid [32] and Permlyzer [33] are required to monitor permission use and data authentication.

Due to time constraints and budget limitations, human-related limitations are out of our scope. The potential adapters on the ad network side can add AI to mitigate user-initiated click fraud and to reward users' ad requests, and design a more sophisticated authorization system (e.g., control panel) to opt in/out permissions if necessary. Besides, it takes time to add trust for a new service.

VI. RELATED WORK

To the best of our knowledge, none of the previous works are directly related to our work, but we find our project is somewhat relevant to four technical areas. One is about three similar ad formats in the industry, and the other three have been studied in the research community.

Ad Types *Rewarded Video Ads* [34], a kind of reward-based app ads embedded in an app, offers rewards to users for clicking and viewing sponsored video ads promoting other mobile apps. While such unrelated ads reach a high average click-through rate of 13.64% [35], Apple started rejecting apps that reward video views [36]. Also, Google offers *In-App Purchase Ads* [37] to display multiple in-app purchase items for sale within a single mobile ad. Obviously, that ad format is different from our settings.

Security in App Monetization Researchers have been paying attention to smartphone security and privacy in different monetization methods since 2012. In [38], researchers examined 100K Android apps and identified 100 representative in-app ad libraries within 52.1% of these apps. AdRisk detects risks from uploading sensitive information to remote servers to executing untrusted code from Internet sources. Similarly, researchers in [16] investigated user privacy in 13 popular ad libraries by classifying the permissions specified in their documentation as well as figuring out the misused permissions and insecure JavaScript interfaces. As for in-app billing, VirtualSwindle [39] finds 60% of 85 popular Android apps are automatically and easily crackable when developers do not rigorously follow Google's guidelines. PEDAL [40] de-escalates privileges for ad libraries from host apps.

Android Permissions Mobile apps may be made by all types of developers. In order to get income and add features, developers may import third-party libraries into their apps. These libraries may demand specific permissions, such as sharing location data, to work properly. Android's permissions are used to inform users the potential dangers of installing apps. However, developers usually include more and more unnecessary permissions within their apps. Therefore, mobile privacy researchers looked into Android permissions. [41] finds about 5.8% of the 700 tested apps crash after a permission is removed and investigated the effect of removing a few popular permissions. Researchers in [42] revealed the market shares and the permissions used by various ad libraries over time (e.g., More and more permissions are used in ad libraries). [43] shows that a user's privacy preference is strongly influenced by the purpose of using such a permission.

Human Factors User studies in smartphones are more or less related to permissions. In [44], authors utilize two guiding principles to make a selection among four permission-

granting mechanisms (i.e., automatic granting, trusted UI, runtime consent dialogs, and install-time warnings). Felt et al. run two usability studies and revealed that current permission warnings in Android do not help users make decisions during installation [45]. Meanwhile in [46], Kelly et al. found a clearer privacy warning could direct users to install apps requesting fewer permissions. Researchers also looked into permissions in details. In [47], every dangerous permissions used in different scenarios (i.e., publicly, with friends, with advertisers, and sent to servers) are ranked by users. Also in [48], researchers revealed that in order to let users assess evaluate apps easily, risks should be decomposed into four different dimensions (e.g., personal information privacy, and data integrity).

VII. Conclusion

In this paper, we propose In-App AdPay, a new monetization service which allows advertisers "pay" targeted users for virtual transactions via secure connections. While keeping most of the existing framework's functionality unchanged, In-App AdPay brings mobile users into the monetization loop. After testing with 43 adult volunteers about in-app advertising and in-app billing, we conducted a usability test of In-App AdPay with 42 people from the same pool. According to their feedback, we witness a great improvement of user satisfaction on advertisers. Furthermore, a closer examination of user activities discovers how users view advertisers and value permissions in different test scenarios. Finally, we believe that In-App AdPay will favor all players and facilitate the mobile advertising ecosystem.

Acknowledgment

This work is partially sponsored by the National Key Research and Development Program of China under grant No. 2016YFB0800100 (2016YFB0800102) and the CCF-Tencent Open Research Fund.

References

[1] IAB Internet Advertising Revenue FY 2015, https://www.iab.com/wp-content/uploads/2016/04/IAB-Internet-Advertising-Revenue-Report-FY-2015.pdf
[2] Android's in-app billing is out to consumers, http://www.adweek.com/socialtimes/android-in-app-billing/508003
[3] Purchase tracking in Tapjoy, http://dev.tapjoy.com/sdk-integration/android/getting-started-guide-publishers-android/
[4] Candy crush saga players spent over $1.3 billion, http://www.macrumors.com/2015/02/13/candy-crush-saga-revenue-2014/
[5] Most Annoying, http://dazeinfo.com/2012/12/18/ads-on-mobile-apps-and-online-videos-are-most-annoying/
[6] Adblock Plus passes 500 million downloads, http://venturebeat.com/2016/01/22/10-years-in-adblock-plus-passes-500-million-downloads/
[7] Mobile app monetization trends, http://blog.desk.pm/wp-content/uploads/2015/05/App-Annie-IDC-Mobile-App-Advertising-Monetization-Trends-2013-2018-EN.pdf
[8] A long tail of whales, http://recode.net/2014/02/26/a-long-tail-of-whales-half-of-mobile-games-money-comes-from-0-15-percent-of-players/
[9] Not popular with gamers, http://www.csmonitor.com/Business/Saving-Money/2014/0302/In-app-purchases-not-popular-with-gamers
[10] Vallina-Rodriguez, N., et al.: Breaking for Commercials: Characterizing Mobile Advertising. IMC 2012.
[11] Falaki, H., et al.: A First Look at Traffic on Smartphones. IMC 2010.
[12] Liftoff releases q1 2015 mobile app engagement index, http://liftoff.io/liftoff-releases-q1-2015-mobile-app-engagement-index-details-cpa-trends-across-category-platform-and-gender/
[13] Digital Advertising Benchmarks 2014, http://www.slideshare.net/augustinefou/digital-advertising-benchmarks-2014-by-augustine-fou
[14] Book, T., Wallach, D.: A Case of Collusion: A Study of the Interface between Ad Libraries and Their Apps. SPSM 2013.
[15] Book, T., Wallach, D.: An Empirical Study of Mobile Ad Targeting. Technical report 2015.
[16] Stevens, R., et al.: Investigating User Privacy in Android Ad Libraries. MoST 2012.
[17] Crussell, J., et al.: MAdFraud: Investigating Ad Fraud in Android Applications. MobiSys 2014.
[18] Nath, S.: MAdScope: Characterizing Mobile In-App Targeted Ads. MobiSys 2015.
[19] Android's app permissions were just simplified, now they're much less secure, http://www.howtogeek.com/190863/androids-app-permissions-were-just-simplified-now-theyre-much-less-secure/
[20] A guide to understanding android app permissions, http://www.hongkiat.com/blog/android-app-permissions/
[21] Mobile apps collect information about users, http://www.pewresearch.org/fact-tank/2014/04/29/mobile-apps-collect-information-about-users-with-wide-range-of-permissions/
[22] Inmobi opt-out, http://www.inmobi.com/page/opt-out/
[23] "https everywhere", http://doubleclickadvertisers.blogspot.com/2015/04/ads-take-step-towards-https-everywhere.html
[24] How many test users in a usability study?, https://www.nngroup.com/articles/how-many-test-users/
[25] Ackerman, M., et al.: Privacy in E-Commerce: Examining User Scenarios and Privacy Preferences. EC 1999.
[26] De Winter, J.: Using the Students T-Test with Extremely Small Sample Sizes. PARE 2013.
[27] Review app permissions thru android 5.9, https://support.google.com/googleplay/answer/6014972?hl=en
[28] Zhang, X., et al.: AFrame: Isolating Advertisements from Mobile Applications in Android. ACSAC 2013.
[29] Franziska R.,Tadayoshi K.: Securing Embedded User Interfaces: Android and Beyond. USENIX Security 2013.
[30] Li, W., et al.: AdAttester: Secure Online Mobile Advertisement Attestation Using TrustZone. MobiSys 2015.
[31] Enck, W., et al.: TaintDroid: An Information-Flow Tracking System for Realtime Privacy Monitoring on Smartphones. USENIX OSDI 2010.
[32] Zhang, Y., et al., Wang, X., Zang, B.: Vetting Undesirable Behaviors in Android Apps with Permission Use Analysis. CCS 2013.
[33] Xu, W., et al.: Permlyzer: Analyzing Permission Usage in Android Applications. ISSRE 2013.
[34] Big growth for rewarded video ads, http://venturebeat.com/2015/03/01/supersonic-sees-big-growth-for-rewarded-video-ads-in-mobile-games-interview/
[35] Pros & cons of 5 mobile ad options, http://blog.kiip.me/developers/mobile-ad-options/
[36] Apple store policy of rejecting apps with rewarded video, http://techcrunch.com/2014/06/24/app-store-policy-of-rejecting-apps-with-rewarded-video-social-sharing-gets-rolled-back-with-a-few-caveats/
[37] In-app purchase ads, https://developers.google.com/admob/android/iap
[38] Grace, M., et al.: Unsafe Exposure Analysis of Mobile In-App Advertisements. WiSec 2012.
[39] Mulliner, C., et al.: Virtualswindle: An Automated Attack against In-App Billing on Android. ASIACCS 2014.
[40] Liu, B., et al.: Efficient Privilege De-Escalation for Ad Libraries in Mobile Apps. MobiSys 2015.
[41] Kennedy, K., et al.: Quantifying the Effects of Removing Permissions from Android Applications. MoST 2013.
[42] Book, T., et al.: Longitudinal Analysis of Android Ad Library Permissions. MoST 2013.
[43] Lin, J., et al.: Modeling Users Mobile App Privacy Preferences: Restoring Ssability in a Sea of Permission Settings. SOUPS 2014.
[44] Felt, A., et al.: How to Ask for Permission. HotSec 2012.
[45] Felt, A., et al.: Android permissions: User Attention, Comprehension, and Behavior. SOUPS 2012.
[46] Kelley, P., et al.: Privacy as Part of the App Decision-Making Process. CHI 2013.
[47] Felt, A., et al.: I've Got 99 Problems, but Vibration ain't One: A Survey of Smartphone Users' Concerns. SPSM 2012.
[48] Jorgensen, Z., et al.: Dimensions of Risk in Mobile Applications: A User Study. CODASPY 2015.