Bind Your Phone Number with Caution: Automated User Profiling Through Address Book Matching on Smartphone

Yao Cheng
Institute of Software
Chinese Academy of Sciences
Beijing 100190, China
chengyao@is.iscas.ac.cn

Lingyun Ying
Institute of Software
Chinese Academy of Sciences
Beijing 100190, China
yly@is.iscas.ac.cn

Sibei Jiao
Institute of Software
Chinese Academy of Sciences
Beijing 100190, China
jiaosibei@is.iscas.ac.cn

Purui Su
Institute of Software
Chinese Academy of Sciences
Beijing 100190, China
supurui@is.iscas.ac.cn

Dengguo Feng
Institute of Software
Chinese Academy of Sciences
Beijing 100190, China
feng@is.iscas.ac.cn

ABSTRACT

Due to the cost-efficient communicating manner and attractive user experience, messenger applications have dominated every smartphone in recent years. Nowadays, Address Book Matching, a new feature that helps people keep in touch with real world contacts, has been loaded in many popular messenger applications, which unfortunately as well brings severe privacy issues to users. In this paper, we propose a novel method to abuse such feature to automatically collect user profiles. This method can be applied to any application equipped with Address Book Matching independent of mobile platforms. We also build a prototype on Android to verify the effectiveness of our method. Moreover, we integrate profiles gathered from different messenger applications and provide insights by performing a consistency and authenticity analysis on user profile fields. As our experiments show, the abuse of Address Book Matching can cause severe user privacy leakage. Finally, we provide some countermeasures for developers to avoid this issue when designing messenger applications.

Categories and Subject Descriptors

D.4.6 [**Security and Protection**]: Invasive software; K.4.1 [**Public Policy Issues**]: Privacy

General Terms

Experimentation, Security

Keywords

Smartphone Applications, Privacy, User Profiling, Address Book Matching

ASIA CCS'13, May 8–10, 2013, Hangzhou, China.

1. INTRODUCTION

Messenger applications (e.g., WeChat [2] and Kik Messenger), which provide instant messaging service for mobile users, have become killer applications in mobile application markets (e.g., App Store and Google Play). It is reported that WeChat, a wildly popular instant messaging application for mobile devices, has doubled its users base in only 6 months, from 100 million in March 2012 to 200 million in September 2012. Messenger applications are popular for their flexible multimedia communication experience and economic boon. Before enjoying such convenience, users are asked to register an application account by uploading their personal information. For easily being discovered by friends, users prone to register with genuine personal information which is supposed to be visible to friends, however, in the real world this situation is not exactly as expected. Besides, messenger applications often recommend a list of other users by exploring contact lists in smartphones, which highs up the probability of exposing users' privacy to the wild.

Our work is motivated by the fact that an increasing number of messenger applications start to add Address Book Matching to their features. This new feature allows user to upload his contact list to the application server from which returns basic personal information of a list of other users who exist in the contact list, for further identification.

Our study shows that the abuse of Address Book Matching, equipped by most popular messenger applications, will bring about severe leak of user privacy in a large scale. Specifically, we make the following contributions: 1) We propose a novel approach to automatically collect user profiles via messenger applications on mobile platform for the first time in this area. Our method discovers a connection explicitly set up between user's virtual account and his real-world phone number, both of which would be revealed to strangers; 2) We implement a prototype to verify our approach. Through single-application analysis and cross-application integration, the experiments show the effectiveness of our method. Moreover, we conduct a consistency and authenticity analysis, which discloses wider genuine information of victimized users; 3) After analyzing main factors that make this attack possible, we provide some suggestions for

mobile application developers to evade user privacy leakage caused by abusing Address Book Matching feature.

2. PROBLEM FORMULATION

The basic idea of our method is presented in this section. Address Book Matching will first be introduced, along with the discussion of its potential security risk.

2.1 Abusing Address Book Matching

From a mobile application designer view, Address Book Matching is essentially introduced to set up connections between user's application account and his phone number which can be regarded as user's unique ID. It allows user to upload address books (mainly including names and phone numbers) to the application server end from which returns a list of user accounts that are bound with phone numbers in those address books. It is mandatory for a user to bind his own phone number with his application account before using Address Book Matching. It assumes that each user behaves regularly and the contacts truly reflect his real social relations. However, in reality attackers are not fabled. The exposure of the aforementioned binding exhibits high potential risk if is abused by attackers.

We propose an approach to reveal the danger of abusing Address Book Matching in terms of user privacy leakage. Our method is applicable to any messenger application featured with Address Book Matching. Specifically, we achieve this goal by uploading a forged address book which consists of contact information of a large number of recommended individuals and randomly chosen phone number owners. Thus we can obtain all the phone numbers along with corresponding account profiles in the response from the application server end. Practically, some applications store users' phone numbers via hashing. However, such protection does little impact on our method, since Address Book Matching completes the phone number match process in the view of regular users; phone numbers are stored as plain text in address book. Our approach totally abuses Address Book Matching, rather than reading hash values by our effort.

2.2 Information Integration and Analysis

The content and scheme of user profiles vary from applications to applications. The information returned from one application can give us a set of user profiles, while integration across different applications would give us a more comprehensive user profiles.

2.2.1 Horizontal Collaboration

Horizontal collaboration contains two parts: broad union and deep intersection. Broad union integrates all distinct accounts corresponding to different phone numbers, producing a large set of user profiles. Deep intersection aims at digging more comprehensive information of one single real identity. The key point behind deep intersection is that phone numbers, as globally unique identifiers for real identities, might connect with multiple accounts in different applications.

2.2.2 Vertical Penetration

Many large online service providers like Facebook and Tencent who also produce mobile applications open ID login services to other entities. As a result, some applications (e.g., instant messaging and social networking) share the same set of login IDs and many of them are also supported by the same vendor. Penetrating applications using the same login ID is called vertical penetration which can provides us with more detailed personal information of that ID than horizontal collaboration, yet at the expense of more manual work. A case of this will be analyzed and demonstrated later in Section 4.5.

2.2.3 Consistency and Authenticity Analysis

Applications aim at meeting users' different social needs, such as meeting new friends or contacting old friends. Different needs determine users' diverse profile filling patterns. Our analysis of the consistency and authenticity targets at multiple user profiles collected from different applications simultaneously. In terms of consistency, we trust the field value as long as it is consistent across all applications. For inconsistent ones, a winner-take-all scheme is employed. For authenticity analysis, particularly in terms of user name, we use *Bai Jia Xing*, a database containing hundreds of most commonly used Chinese surnames with coverage over 90%, and regular patterns to search for users' possible official names.

2.3 Malicious Usage

In online environment, people tend to trust accounts with detailed and reasonable-looking profiles. However, this common sense will lead to potential misjudgment if attackers collect a larger number of user profiles in an automated way. Generally, attackers may perform the following operations by abusing Address Book Matching:

- Attackers may obtain users' profiles and the associated phone numbers. In other words, a link between *real* identity and *virtual* account has been established. Deeper understanding about the phone number owners can assist greatly to performing further attacks, such as cracking user accounts and social engineering.

- Attackers, especially spammers, may determine whether particular phone numbers are in use or not by querying the server with a list of unidentified phone numbers. Meanwhile, with collected information (e.g., gender, region and hobby), they may also perform contextually targeted advertising.

- Through information integration, comprehensive personal data can be retrieved, such as school, department, working company and etc., with which virtual network fraud can be carried out readily. In this case, the more applications users register, the more personal information will be given away.

- Attackers may clone identities leveraging the profiles they obtained. Clone means duplicating identities on particular applications. The more comprehensive profiles gathered by attackers, the closer the clones are to real identities.

3. SYSTEM OVERVIEW

In this section, we present the overall system architecture of abusing Address Book Matching as shown in Figure 1, then we elaborate the abusing method and implementation details.

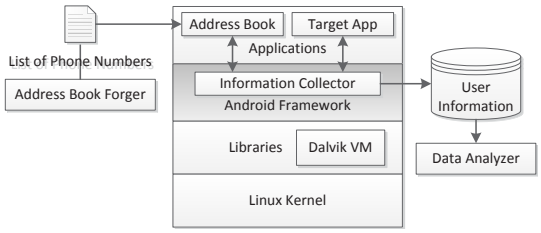

Figure 1: System Architecture

3.1 Method

Our method performs in three phases, i.e., address generation, information collection and data analysis which respectively achieve for forging address book, collecting information from the return list and correlating data for analysis.

Address Generation. This phase is to automatically generate lists of target phone numbers. Attackers typically enumerate all the interested phone numbers. The output of this phase will be used as input for querying target application servers; thus it should be formatted as a standard contact for compatibility.

Information Collection. We feed above mentioned contact lists to target applications via Address Book Matching. Applications then return a list of users within the contact lists, each of which has already registered an account and bound phone number with this account. Information Collector handles all user data returned from the applications. The implementation details are discussed in Section 3.2.

Data Analysis. This phase focuses on understanding the data obtained by previous phases more deeply. Our analysis includes two parts, i.e., single-application analysis and cross-application integration (see Section 4).

3.2 Implementation

This section elaborates the implementation of our system, particularly Information Collector. Different from crawling webpages, collecting user profiles from mobile devices is more complicated. The access to application data is often protected by mobile systems from other applications or processes for security purpose. We choose Android as the platform to build our prototype. At kernel level, each application has a low-privilege user ID and runs in its own process. At Android system level, when runs, each Android application is isolated by the Dalvik VM. The permission policies provided by Android system prevent applications from abusing each other's components, which makes our collection more complicated as the components of target application are not directly accessible.

To tackle this issue, we finally decide to monitor the Android APIs because: 1) local application database is often encrypted as well as the network traffic, however, decryption itself is an already hard problem with high cost; 2) hijacking the Intent is unworkable as the huge Intent messages are hard to target. Also, using Intent to deliver user information is not always the best choice for application developers; 3) modifying target applications to log down user data can avoid the decryption. However, this approach is subject to scalable problem since we need to rewrite specific modules for different applications, which is tedious and pleonastic. In fact, the Android framework wraps Linux system calls as specialized APIs for easily creating rich and novel applications, meaning that we do not need to make modifications for various applications. Instead, we just need to monitor each Android API the application invokes. Therefore, we monitor the APIs called by target applications which are responsible for handling user profiles. The monitor logs down user profile related data in real-time and in raw format, which will be post-processed for further analysis. Besides, for scalability purpose, we leverage ADB shell commands to automatically process the activity change by recording the motion and replaying it till the collection is done.

4. EXPERIMENT AND DATA ANALYSIS

In this section, we show our experiments and data analysis results. We first make some ethical considerations to prevent our method from being misused as best as we can.

4.1 Ethical Considerations

Our experiments randomly forge an address book and collect corresponding user profiles from messenger applications, which may bring about ethical and legal issues, similar with the cases in [9, 3]. However, this is the most efficient and effective way to conduct our experiments since 1) other methods like survey always suffer from inflating user concerns about privacy [6], by directly asking questions about privacy; 2) Comparatively, empirical experiments are more reliable. The collected data could reflect situations in the real world which are more convincing; 3) We carefully design experiments to avoid privacy issues, insuring the following principles fulfilled: a) The collected user data will never flow out of our experiment devices or be provided to any other irrelevant individual or organization; b) We will never perform any further penetration attempt to any account, except for proving our idea in this paper.

4.2 Single Application Analysis

We choose WeChat (Version 4.2 for Android), an aforementioned popular mobile messenger application with Address Book Matching, as our object to conduct the experiments.

Phone numbers in mainland China consist of 11 digits, not including two digits Mainland China Country Code (+86). We forge a list of 100,000 contacts of two separate continuous phone number segments, i.e., +861521063 - +861521070 and +861521098 - +861521099. After binding our test phone number to a newly applied WeChat account, Address Book

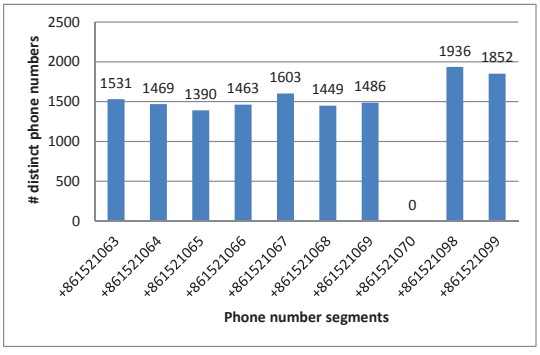

Figure 2: The Distribution of Active Phone Numbers Across all the Segments

Matching component of WeChat is activated. Upon uploading the forged address book to the server, WeChat begins to recommend all the accounts that already have been bound with phone numbers in the address book. As mentioned in Section 3.2, we modified Android system to monitor and record all the APIs specified to text and image display, thus we send a sequence of ADB shell "sendevent" commands to the simulator to start an ADB motion simulation which imitates a sequence of user behaviors of browsing the contacts profile. Finally, we succeed to obtain all fields of WeChat user profiles via Address Book Matching, they are WeChat ID, display name, phone number, gender, region, and "what's up". Generally, display name is close to the user's real name, as the display name is meant for identification by real world friends.

In the following step, we conduct a statistic analysis. Among 100,000 phone numbers, overall 14,179 registered users are matched (denoted as W), thus the user penetration rate is 14.18%. This number is consistent with the official released data 13.73%, which verifies the representative of our sampling. The distribution of the phone numbers in each segment is shown in Figure 2. Note that no user profiles correspond to the phone number segment +861521070, which is unallocated yet according to our further investigation. Among those 14179 users, not all of them upload their information completely. Over 55.13% of them have their profile fully completed as shown in Figure 3, indicating that most of the users indeed rely on social applications to maintain their social connections, thus they eager to put more details of their personal information online to facilitate the interaction with other people. Only 12.43% fill nothing but the required fields of WeChat ID and name. Even so, their phone numbers already connect with the WeChat accounts and may be leaked to attackers.

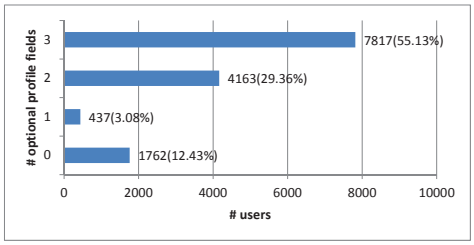

Figure 3: The Completeness of Optional Profile Fields

4.3 Horizontal Collaboration

Horizontal collaboration performs across multiple applications. We choose MiTalk Messenger [1] as our target which is a free and cross-platform messenger application whose users has already reached 17 million until August 2012. MiTalk provides a seamless connection between users' accounts and their address books to facilitate users' social connections.

In this phase, we use segment +861521098 which has uncovered maximum number of user profiles in previous experiment on WeChat. In the experiment, among those 10,000 phone numbers, 360 MiTalk user accounts have been returned (denoted as M) from MiTalk server and been added to our MiTalk friend list. MiTalk provides a more comprehensive profile scheme, including user name, gender, birthday, photo links, location he currently stays, and the names of schools and companies. Almost all the users have more than one photo link which can be visited directly via browser, and 30% of them maintain school and company name fields which are regarded as privacy for strangers.

As illustrated in Section 2.2, horizontal collaboration includes two parts: broad union and deep intersection. 116 new users are found in MiTalk. After broad union we totally have 2052 phone numbers and corresponding user accounts. In terms of deep intersection, we integrate user profiles obtained from WeChat and MiTalk and totally 244 users are found to have registered both applications. We then perform consistency and authenticity analysis based on these 244 users profiles (denoted as D).

4.4 Consistency and Authenticity Analysis

We find that 34.02% of users in D share similar display names in both applications, which inspires us to further find out in what degree their names are the official names. We leverage the Chinese *Bai Jia Xing* (aka. Hundred Family Surnames) data set to match official names. Typical Chinese names begin with one of the surnames in *Bai Jia Xing* as family name and followed by a given name of one or two Chinese characters length. We believe that display names in formal Chinese full name format are probably users' official names. Through regular pattern matching, we investigate the display names from W, M and D to find out the ratio of formal Chinese full name in respective dataset. The results are shown in Table 1 that in M 48.33% of users' display names are probably official names, which is quite higher than that of W, 31.24%, indicating that MiTalk users tend to register accounts using official names. This may be ascribed to the fact that MiTalk promotionally recommends users to provide personal information as truthful as possible when registering, so as to facilitate the contact with their friends. This "seamless" strategy makes users lean to trust other MiTalk users. They naturally believe that the uploaded profiles are only prepared for their friends by default. The case in D outperforms those in W and M that it reaches a ratio of 59.43%, which implies users, registered in both applications, are more adhered to and trust messenger applications; thus are more willing to use real personal information for registration. It also makes us realize that through deep intersection, more comprehensive and truthful user profiles can be aggregated not only because of the correlation but also the user habit.

We also spot another odd phenomenon - the gender inconsistency. Presented in Table 2, 54.1% of the users in D have their gender fields contradicted in both applications. This phenomenon is called inconsistency which has also been found in different types of social networks [3]. Moreover, among all "gender inconsistency" users, those who appear "male in MiTalk and female in WeChat" account for 68.9% (90/132). To further determine the likely-to-be-true gender of each user, the winner-take-all strategy which favors majority cases if more than two choices are available could be employed. Yet, it is still hard to confirm a user's gender just by comparing results from two applications. However, we have noticed that the proportion of male and female gender is more unbalanced in WeChat profiles. In addition, based on the name authenticity analysis presented above, we deem user profiles in MiTalk to be more trustworthy which is convincing after we manually check the display name field of inconsistency users in D.

Dataset	Total # Users	# Users with Real Name	Rate
WeChat (W)	14179	4430	31.24%
MiTalk (M)	360	174	48.33%
Deep intersection set (D)	244	145	59.43%

Table 1: Comparison of Name Authenticity Analysis Results from Three Dataset - WeChat, MiTalk and Deep Intersection Set

WeChat \ MiTalk	Male	Female
Male	37	42
Female	90	57

Table 2: Gender Consistency of Deep Intersection Set

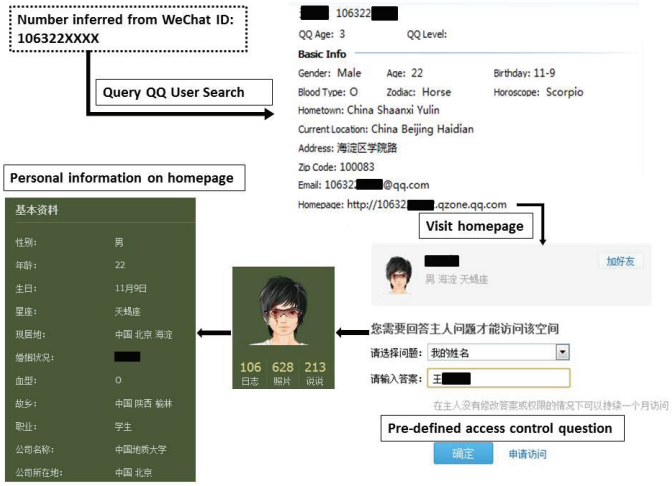

Figure 4: Vertical Penetration Process

4.5 Vertical Penetration

In this section, we manually conduct a case study using deep intersection dataset D to concisely demonstrate vertical penetration.

Created by the same company, WeChat is designed to share the same account database as QQ which is the most popular IM (Instant Messaging) software in China. We choose a user whose WeChat ID is "t106322XXXX" with last four digits replaced by letter 'X' for privacy concern. As shown in Figure 4, we query "user search function" of QQ using the last ten digits of the WeChat ID as QQ account ID by our speculation, and as expected a single match finally returns. We then collect additional profile fields except those from WeChat, from which we know that this QQ account belongs to a young man from Shannxi, China, born on November 9^{th} 1990, and currently stays in Beijing, China. In addition, we also find his personal home page address. However, until now we are not sure the owner of this QQ account is exactly the one has WeChat ID of "t106322XXXX". We then go one step further - visiting his personal home page, where a pre-defined question asking for the answer of "real name" shows up for access control. By inputting the display name of this WeChat account, we successfully log in! Until now, we can safely conclude that the aforementioned QQ account and WeChat account are indeed belong to the same person. Finally, the home page reveals more information about the young man as shown in Table 3 which also illustrates incremental profile fields we collect along each step.

Step	Field	Value
0	WeChat ID	t106322XXXX
	WeChat name	WangXX
	Phone number	+861521098XXXX
1	QQ name	TianXX
	Date of birth	1990.11.09
	Blood type	O
	Hometown	Yulin, Shannxi Province
	Address	Xueyuan Rd, Haidian Distinct
	Zip code	100083
	E-mail address	106322XXXX@qq.com
	Home page	http:// X.qzone.qq.com
2	Real name	WangXX
	Marital status	X
	Blogs	106 blogs
	Photos	628 photos
	Status	213 status
	Occupation	Student
	Name of school	China University of Geosciences

Table 3: Profile Fields Seized in Each Penetration Step

We can also infer from the pre-defined access control question that the home page is supposed to be visited by the friends who really know this young man. However, we, as strangers, have successfully obtained his personal information, even private secretes.

We argue that for attackers who intend to crack someone, manual work should be a price worth paying. As it can be seen that vertical penetration can provide a wealth of information which is crucial for earning trust when camouflaging, boosting the implementation of online fraud.

5. COUNTERMEASURE

By abusing Address Book Matching, we have successfully gathered users' phone numbers and corresponding user accounts in large scale, however, attackers will be no exception. It seems like throwing away the apple because of the core to disable this promising function. Therefore, in this section, we propose several feasible alternatives for developers to avoid this issue by tackling the key factor - the establishment of connection between real phone numbers and virtual user accounts in flawed messenger applications: 1) Make phone numbers be queried with a one-at-a-time constraint, which meets users' needs and meanwhile removes the chance for attackers to launch large scale attacks. 2) Only return the display names of application accounts, rather than detailed profiles nor corresponding phone numbers. This

could be applied to applications with real-name system in which users can identify their real world friends by the official names. 3) Return the phone numbers and the names which should be exactly identical to the ones stored in the address book. No account information should be accessed, making sure no virtual and real identity connections are acquired. 4) Recommend a user U those who also have U in their address books. This is the way particularly suitable for phone number based relationship management which is generally bilateral.

6. RELATED WORK

Privacy Analysis of Applications. Privacy issues of applications are widely studied. W. Enck et al. [8] implements TanitDroid, a dynamic taint analysis tool, that modifies Android's Dalvik VM and determines the violation of privacy when there is tainted sensitive data flowing out of the phone. PiOS [7] is another privacy leakage analyzing tool performing static taint analysis on iOS application binaries. In terms of preventing privacy leakage, works such as MockDroid [4] and TISSA [11] allow users to give resource access permission to an application, whereas the system reports empty or unavailable when application requests access. Differently, we leverage existing Address Book Matching function to gather private user profiles without considering the information flow internally.

Automated User Profiling. Leyla Bilge et al. [5] achieves automated identity theft via profiling users on SNS (Social Networking Services) by employing friendship relation. M. Balduzzi et al. [3] abuses registered e-mail address querying which is designed for users to search friends on SNS. They collected a large number of users' personal information by crawling and connecting their profiles. In our paper, we try to get user information from applications and the techniques are quite different from crawling webpages.

Problems with Misusing Phone Numbers in Communication Applications. Sebastian Schrittwieser et al. [10] analyze several popular message and VoIP applications, finding that most such applications treat users' phone numbers as their account identifiers. They show some security flaws in terms of authentication mechanisms, such as account hijack or ID spoof. Differing from their work, our experiments show that contact matching technology, adopted by many messenger applications today, has exhibited severe privacy leakage problem because phone numbers are used as external identifiers of user accounts which can be exploited by attackers to seize other users' accounts.

7. CONCLUSION

We exploit Address Book Matching, a novel feature of smartphone messenger applications, to collect user profiles. To our best knowledge, this work is the first to collect user profiles via smartphone applications. The consistency and authenticity of collected user profiles are further analyzed. Results show that smartphone users prone to leave truthful personal information in mobile messenger applications, however, for different applications, users' confidence may vary due to different registration rules and promotion strategies. Finally, we analyze main factors that make this abusing possible and provide countermeasures for application developers. Developers should be cautious and do their best to prevent the exposure of the mapping information when users' phone numbers are linked with their application accounts.

8. ACKNOWLEDGMENTS

This work was supported by National Program on Key Basic Research Project (Grant No. 2012CB315804), National Natural Science Foundation of China (Grant No. 61073179 and No. 91118006), National Science and Technology Major Project (Grant No. 2011ZX03002-005-02) and Beijing Municipal Natural Science Foundation (Grant No. 4122086).

9. REFERENCES

[1] Mitalk messenger. `http://www.miliao.com/`.

[2] Wechat. `http://weixin.qq.com`.

[3] M. Balduzzi, C. Platzer, T. Holz, E. Kirda, D. Balzarotti, and C. Kruegel. Abusing social networks for automated user profiling. In S. Jha, R. Sommer, and C. Kreibich, editors, *Recent Advances in Intrusion Detection*, volume 6307 of *Lecture Notes in Computer Science*, pages 422–441. 2010.

[4] A. R. Beresford, A. Rice, N. Skehin, and R. Sohan. Mockdroid: trading privacy for application functionality on smartphones. In *Proceedings of the 12th Workshop on Mobile Computing Systems and Applications*, HotMobile '11, pages 49–54, New York, NY, USA, 2011.

[5] L. Bilge, T. Strufe, D. Balzarotti, and E. Kirda. All your contacts are belong to us: automated identity theft attacks on social networks. In *Proceedings of the 18th international conference on World wide web*, WWW '09, pages 551–560, New York, NY, USA, 2009.

[6] A. Braunstein, L. Granka, and J. Staddon. Indirect content privacy surveys: measuring privacy without asking about it. In *Proceedings of the Seventh Symposium on Usable Privacy and Security*, SOUPS '11, pages 15:1–15:14, New York, NY, USA, 2011.

[7] M. Egele, C. Kruegel, E. Kirda, and G. Vigna. PiOS: Detecting privacy leaks in iOS applications. In *Proceedings of the 18th Annual Network & Distributed System Security Symposium (NDSS)*, Feb. 2011.

[8] W. Enck, P. Gilbert, B.-G. Chun, L. P. Cox, J. Jung, P. McDaniel, and A. N. Sheth. Taintdroid: an information-flow tracking system for realtime privacy monitoring on smartphones. In *Proceedings of the 9th USENIX conference on Operating systems design and implementation*, OSDI'10, pages 1–6, Berkeley, CA, USA, 2010.

[9] M. Jakobsson, N. Johnson, and P. Finn. Why and how to perform fraud experiments. *IEEE Security and Privacy*, 6(2):66–68, Mar. 2008.

[10] S. Schrittwieser, P. Fruehwirt, P. Kieseberg, M. Leithner, M. Mulazzani, M. Huber, and E. Weippl. Guess who is texting you? evaluating the security of smartphone messaging applications. In *Network and Distributed System Security Symposium (NDSS 2012)*, 2012.

[11] Y. Zhou, X. Zhang, X. Jiang, and V. W. Freeh. Taming information-stealing smartphone applications (on android). In *Proceedings of the 4th international conference on Trust and trustworthy computing*, TRUST'11, pages 93–107, Berlin, Heidelberg, 2011.