ACCEPTED MANUSCRIPT

Implications of Smartphone User Privacy Leakage from the Advertiser's Perspective

Yan Wang[a,1], Yingying Chen[b], Fan Ye[c], Hongbo Liu[d,1], Jie Yang[e]

[a]*Department of Computer Science, Binghamton University, 4400 Vestal Parkway East Binghamton, NY 13902*
[b]*WINLAB, Rutgers University, 671 Route 1 North Brunswick, NJ 08902*
[c]*Department of Electrical and Computer Engineering, Stony Brook University, 100 Engineering Bldg., Stony Brook, NY 11794*
[d]*Department of Computer Information and Technology, IUPUI, 799 West Michigan Street, ET 301 Indianapolis, IN 46202*
[e]*Department of Computer Science, Florida State University, Tallahassee, FL, 32306*

Abstract

Many smartphone apps routinely gather various private user data and send them to advertisers. Despite recent study on protection mechanisms and analysis on apps' behavior, the understanding about the consequences of such privacy losses remains limited. In this paper, we investigate how much an advertiser can infer about users' social and community relationships. After one month's user study involving about 190 most popular Android apps, we find that an advertiser can infer 90% of the social relationships. We further propose a privacy leakage inference framework and use real mobility traces and Foursquare data to quantify the consequences of privacy leakage. We find that achieving 90% inference accuracy of the social and community relationships requires merely 3 weeks' user data. Finally, we present a real-time privacy leakage visualization tool that captures and displays the spatial-temporal characteristics of the leakages. The discoveries underscore the importance of early adoption of privacy protection mechanisms.

Keywords:
Smartphone, Privacy, Social Relationship

ACCEPTED MANUSCRIPT

1. Introduction

The huge success of smartphones is largely fueled by the availability of millions of phone apps that provide functions covering all aspects of our lives. A large portion of these apps are free. Their developers get financial support from advertisers by embedding their advertisement libraries to display mobile advertisements to users. Many advertisers exist and some of the

Email addresses: `yanwang@binghamton.edu` (Yan Wang), `yingche@scarletmail.rutgers.edu` (Yingying Chen), `fan.ye@stonybrook.edu` (Fan Ye), `hl45@iupui.edu` (Hongbo Liu), `jyang5@fsu.edu` (Jie Yang)

[1]This work was conducted during his Ph.D. study at Stevens Institute of Technology.

This is the author's manuscript of the article published in final edited form as:
Wang, Y., Chen, Y., Ye, F., Liu, H., & Yang, J. (2019). Implications of smartphone user privacy leakage from the advertiser's perspective. Pervasive and Mobile Computing, 53, pp 13-32. https://doi.org/10.1016/j.pmcj.2018.12.006

ACCEPTED MANUSCRIPT

major players include Google, DoubleClick and AdMob [1, 2]. To gain better understanding of user habits and behaviors for accurate ad targeting, these apps customarily scavenge private user data, ranging from the phone's IMEI number, MAC addresses of nearby access points, the user's location, even the contact list, and send it to advertisers [3, 4]. Ultimately, these "free" apps are not entirely free: users pay the price of their privacy.

There has been quite some recent work that investigates the privacy leakage and potential defense mechanisms. TaintDroid [3] can track the flow of different kinds of private information (e.g., IMEI, location) within an app and log the leaking of such information through network interfaces. Barrera et al. [5] and Felt et al. [6] examined permissions requested by about 1,000 apps and found requests for unnecessary permissions commonly exist. A number of tools [7, 8] can help users manage permissions granted to apps such that they do not have access to certain private information. Sowayway [9] and Kirin [10] can detect over-privileged apps or identify requests of dangerous combinations of permissions. Beresford et al. and Zhou et al. proposed MockDroid [11] and TISSA [12] respectively to obfuscate private information so that adversaries only receive empty, fake or anonymized information. Agarwal et al. [13] proposed a crowdsourcing based mechanism to help users decide proper privacy settings.

In this paper, we seek to answer an important but different question: how much does the advertiser know about the user, in particular, her social and community relationship (e.g.,family, colleagues and friends) from the leaked private data? This is motivated by a couple observations. First, there is only limited study of apps' dynamic leakage behavior at run-time. Existing study [14, 9, 4, 11, 15, 12, 5, 6] is mostly on the static aspects of apps' permissions. TaintDroid [3] can be used to log the leaking activities but the paper did not focus on a systematic study on the destinations, frequencies and types of apps' run-time privacy leakages. Second, the *consequences* of such leakage, especially when an advertiser gathers such private data from many users and across many apps, is not known either. It is easy to conjecture that the advertiser may gain additional information when cross-examining private data, but exactly what can be learnt remains an open issue.

We focus on one important aspect of that perspective, the social and community relationships of a user, such as her family, colleagues and friends. Such knowledge is an important channel for the advertiser to push relevant advertisements since people tend to take note on things their acquaintances have done (e.g., bought). For example Facebook has largely relied on people voluntarily publicizing such relationship. However, many real world relationships are not publicized online yet they are equally important to advertisers; and there is a trend for

ACCEPTED MANUSCRIPT

Facebook users of various age groups to go for other "small-circle" social networks, or become less and less active due to privacy concerns [16, 17].

In particular, we quantify to what extent an advertiser can learn and infer users' relationships by developing a privacy leakage inference framework. Our systematic study on privacy leakage inference involves both real experiments with multiple volunteers as well as trace-driven studies with human mobility traces obtained from two data sets, namely MIT reality trace [18] and Foursquare trace [19]. By examining the privacy leakages of participants from a diverse background ranging from academia to city environments (i.e., our real experiments and the MIT trace are academia whereas the Foursquare trace represents a city environment), we discover that the privacy leakage enables an advertiser to infer a significant portion of a user's real world relationships that have physical interactions. Our privacy leakage model and inference framework could server as a foundation for estimating the potential social relationship leakage of a particular user based on his/her app usage.

Specifically, we make the following contributions:

- We conduct a manual study of the frequencies, destinations and types of the *run-time* privacy leakages of nearly 200 most popular apps across 19 categories in Google Play. We discover that major advertisers can easily gather all types of private data in short time from many users.

- We model the relationship inference process in a three-layer framework and define the concept of *connection*, which is exemplified by two users sharing similar patterns in their leaked data (e.g., common Wi-Fi access points). We conduct a one-month real experiment of 10 participants of family, colleague and friend relationships, using various apps in their daily lives. We find that by aggregating data across users and apps, an advertiser can infer over 90% of the relationships from the "connections".

- We further propose two models for users' temporal privacy leakage profiles based on the experimental study. To verify the generality of findings from the real experiments based on privacy leakage inference, we conduct trace-driven studies by populating the derived user profiles to the human mobility traces in the MIT reality [18] and the Foursquare datasets [19]. We find that the advertiser can infer 80-95% of a regular user's relation in academic and city environments after gathering only 3 weeks of private data.

- Finally, we build a visualization tool that captures and displays the spatial-temporal statistics of different types of privacy leakage on both per app and per destination basis

ACCEPTED MANUSCRIPT

ACCEPTED MANUSCRIPT

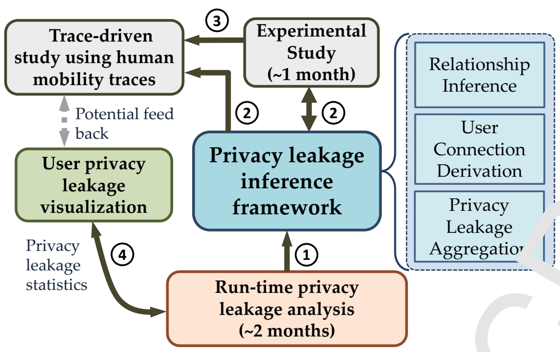

Trace-driven study using human mobility traces
③
Experimental Study (~1 month)
Relationship Inference
User Connection Derivation
Privacy Leakage Aggregation
②
②
Potential feed back
User privacy leakage visualization
Privacy leakage inference framework
Privacy leakage statistics
④
①
Run-time privacy leakage analysis (~2 months)

Figure 1: We take a four-step approach to understand the advertiser's perspective on users' social and community relationships.

in *real time*, which helps users gain better insights on the scope and degree of privacy losses. This tool can serve as a complement to recommendation-based user privacy protection mechanisms [13].

2. Approach Overview

To facilitate the understanding on the consequences of privacy leakages, we take a four-step approach: run-time privacy leakage study, privacy leakage inference and profile modeling via experimental study, inference framework evaluation via trace-driven study, and user privacy leakage visualization, as depicted in Figure 1.

From the advertiser's perspective, we study two types of relations: *social relationship* and *social community*. The social relationship is defined as a pair-wise relationship between two users with certain kind of physical interactions such as colleagues, families, and friends (not virtual friends from online social networks). Whereas a social community involves more than two users, who usually appear at a location during the same time period for certain common interests. For example, a group of students taking the same class twice every week or people eating in the same restaurant every Tuesday. We believe identification of both kinds of relationships help advertisers design better targeted advertising strategies.

Run-Time Privacy Leakage Study. First we want to systematically understand the destinations, frequencies and types of the leaking behavior of apps to understand the flow of common practice of privacy leakages in apps. We study their spatial and temporal privacy leakage characteristics to complement the existing work, which focuses on static aspects of permissions [12, 7, 20], or provides the capability of logging the leakage but stops short of a systematic study [3]. The Wall Street Journal (WSJ) study [21] in 2010 investigated the types of leakages for about 50 most popular apps, but not destinations and frequencies, and

ACCEPTED MANUSCRIPT

ACCEPTED MANUSCRIPT

it was a bit outdated given the fast pace of the mobile market. Thus we conduct a series of experiments over a about two-month period to obtain the most up-to-date picture of privacy leakages, where we also find changes in the types of leakages for about half of the apps studied in [21].

Privacy Leakage Modeling. To understand the consequences of privacy leakages when an advertiser combines the data received from different users, we develop a three-layer privacy leakage inference framework (as depicted in Figure 1) including *Privacy Leakage Aggregation*, *User Connection Derivation* and *Relation Inference*.

We introduce an important concept *connection*, which exists when two users share similarities in leaked data. The *connection* helps bridge the gap between raw privacy leakage data and higher level relationship inference. The intuition is that each type of particular relationship has certain temporal-spatial patterns in users' physical interactions, which can be captured by *connection*. For example, two family members usually stay together at home during late night and early morning; while classmates encounter each other frequently in classrooms during the daytime of weekdays. Although exceptions to such patterns exist, it can usually identify most relationships and is the standard practice widely adopted in social community inference [22, 23]. We conduct an additional experimental study with 10 participants for over one-month time period to confirm the effectiveness of our privacy leakage model, which also reveal that the temporal-spatial similarities between people who have friend relationship is not as regular as that between people of other relationships that have repetitive interactions (e.g., colleagues and families).

Evaluation of User Privacy Inference. To understand whether the above observations can be generalized to larger scale user population with various backgrounds, we extract user privacy leakage profiles, apply them to user mobility traces generated from two datasets with over 500 participants. We verify that using 3 weeks of private data, an advertiser can infer colleague-based relationships of regular users at around 90% accuracy in an academia environment, and friend-based relationships above 95% in a city environment. When an advertiser uses hierarchical clustering to infer social communities, $80 - 90\%$ of those of regular users' are revealed in academia environments, and over 80% in a city environment.

User Privacy Leakage Visualization. Finally, we believe that explicit presentation of privacy leakages to users will help them gain better understanding of the privacy loss. As a starting step, we leverage TaintDroid [3] to develop a tool that visualizes the temporal and spatial characteristics of the leakages in real-time on a mobile device. We note that the privacy

ACCEPTED MANUSCRIPT

ACCEPTED MANUSCRIPT

Table 1: Counts of six types of frequent privacy leakages to top-five appeared destinations during one day (including three time periods) testing for each app.

Destination	Contact	Phone Number	IMEI	GPS Location	Network-based Location	Accelerometer
Google	70	75	49	90	110	13
DoubleClick	5	6	3	9	11	3
Mixpanel	0	0	0	2	9	2
Flurry	2	2	0	4	7	0
Amobee	0	0	6	0	4	0

leakage inference model together with the privacy leakage data can potentially augment the visualization tool so that users can see the consequences on their relationship in real-time, which hopefully can guide their usage of apps (e.g., stop using Yelp for the rest of the day or resetting the ad tracking ID once a threshold of relationships are exposed).

3. Run-Time Privacy Leakage Study

Starting in Android API level 23, users can grant permissions to apps even when they are running. App permissions could be manually revoked by users, and unauthorized communications among apps are prohibited. Although the permission control mechanism seem to be friendly to users, they are not effective in protecting users from malicious apps that request to collect data irrelevant to the main function of the app. Moreover, once the application is granted access, the OS does not have further control on when a type of private data is accessed and how it is used by the application at run-time. The application is free to access it as frequently as possible and send it to wherever it wants over the Internet [3, 9, 1].

The most similar study to ours is the WSJ one [21] in 2010, which focuses on 5 privacy leakage types (i.e., *contacts, location, phone id,* and *phone number*) among 50 most popular apps from Google Play. The results are interesting but there is no analysis on the frequencies and destinations of privacy leakages. Our study aims to provide a more comprehensive and up-to-date analysis including the frequencies and destinations of the leakages. We also investigate how much private data an advertiser can collect and aggregate from multiple apps. This helps the user understand the scope and extent of privacy leakages when running apps; it also serves as the basis for the formulation of the privacy leakage inference in the next section.

3.1. Methodology

We choose the top 10 most popularly downloaded applications from each of the 19 categories in Google Play as of January 2013, totaling 190 applications. We expect that these most popularly downloaded apps are installed by the majority of users, thus their behavior analysis is

ACCEPTED MANUSCRIPT

representative to the majority of users. Our app behavior study is grounded on TaintDroid [3], which helps to track and log the privacy leakages of the applications. Once an application accesses private data, TaintDroid generates a taint log in the Android system log.

We use an Android application called CatLog to capture the system log and save the leaked information as a text file in the smartphone's memory card. The followings are recorded for each leaking event: application name, the leaked private data, time of the leakage, the destination to which the data is sent, the event/operation triggering the leakage, and an optional memo. The saved information is downloaded to a computer to do offline processing, which includes discriminating privacy leakages and calculating the frequency of privacy leakages. Some of the apps crash during the test, and we are able to gather complete results for 145 apps.

We use Google Nexus One and Google Nexus S, both running Android OS 2.3.4, and the whole study lasts for about two months. To capture the application's behavior at different times in a day, we test about 4-5 applications in three different time periods (i.e., morning, noon and night) in each day. During each period, the selected applications are tested one at a time: we first reboot the device to make sure all cached information is cleaned up; then we install the application and perform normal usage for about 5 minutes; finally we uninstall the application and reboot the device to test the next application. This procedure helps to avoid interference between applications.

3.2. Findings

Per App Privacy Leakage. During our study, we find that about 50% of the apps in the WSJ report have changed their privacy leakage behavior. In particular, we find that 97 out of the 145 applications send out private data of the user. The data includes *GPS location, network-based location* (provided by Android based on cellular ID and Wi-Fi networks), *WiFi Access Point SSID list, contact list, phone number, International Mobile Station Equipment Identity (IMEI), accelerometer readings*, but not those of microphone, camera and text message log, which are accessed by the applications but not leaked out. We also find that there are 8 applications sending Wi-Fi SSID list (scanned by the smartphone) through SSL, and 3 of the 8 applications (i.e., Compass, CNN App, Yelp) send this information to the same destination (with IP address 173.194.73.104 belonging to Google according to www.iplocation.net). This type of WiFi AP list is most likely the company's effort to build a WiFi address database for geo-location purposes [24]. It could be employed to infer the user's location, thus potentially her mobility pattern during a day.

. **Per Destination Privacy Leakage.** We further investigate how the private data

ACCEPTED MANUSCRIPT

ACCEPTED MANUSCRIPT

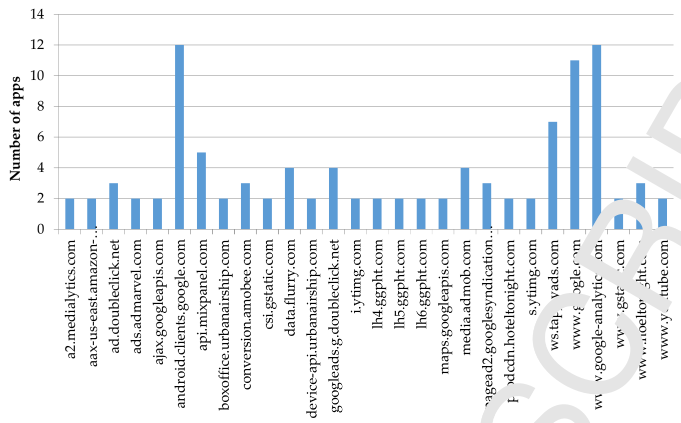


Figure 2: Destinations that collect private data from more than one app.

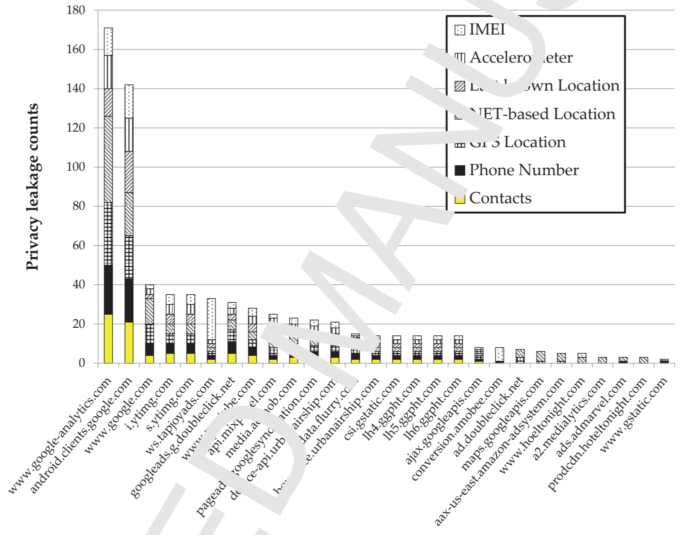


Figure 3: Privacy leakage counts of destinations that collect private data from more than one app.

could be collected by a single destination (e.g., an advertiser's server) through multiple apps. Table 1 summarizes the number of times that 6 types of private data are leaked to the top 5 most frequent destinations with one day test for each app. It is not surprising that Google and DoubleClick (bought by Google in 2008) dwarf the other three much smaller players due to their dominance to the mobile advertising market. We also observe that the location information is the most frequent leaked type, followed by phone number and contact list.

We show the destinations receiving data from more than one app in Figure 2 (called "common" destinations). We identify 28 such destinations in 19 categories. Specifically, we find that three Google destinations collect private data from more than 10 applications, and www.tapjoyads.com from 7 applications. Among all 97 applications showing privacy leakages, 26 of them send 7 different types of privacy data to Google, while 9 applications send 7 differ-

ACCEPTED MANUSCRIPT

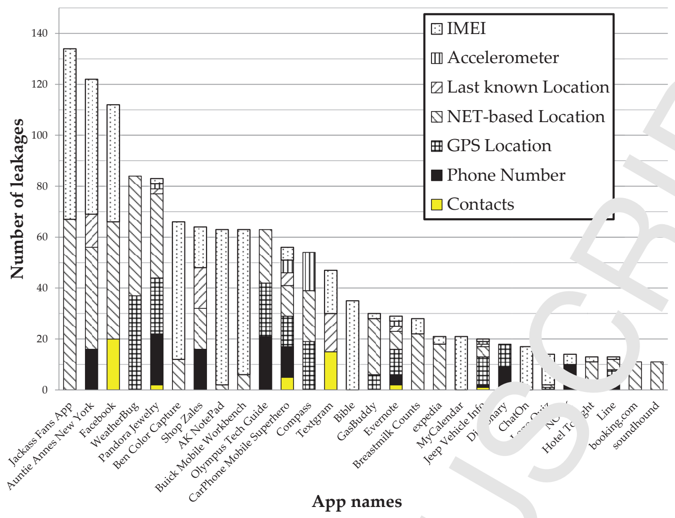


Figure 4: Privacy leakage counts of apps leaking more than 10 times in a 5-minute continuous testing.

ent types to DoubleClick. Figure 3 shows that during one day's testing of the 97 applications, most of the 28 destinations collect more than 3 types of private data, and Google is the most active one among these destinations.

Furthermore, we find that among applications sending to common destinations, they usually send out different types of private data. For instance, we observe that app Pandora Jewelry sends out IMEI, phone number and contact list whereas app Evernote sends out phone number and location information. This confirms our conjecture that an advertiser can combine the private data from different apps to gain a more complete picture of the user's behavior.

Privacy Leakage Frequency. During our testing, we find that the Location and IMEI are the first and second most common privacy leakage types, which involves 71 and 61 applications respectively. We present the leakage count decomposition of 28 apps that leaks more than 10 times in its 5-minute usage in Figure 4. We observe that at least three different types of private information are leaked for most of these apps. Specifically, the combination of IMEI and *NET-based location* is the most leaked information. We also notice that both Facebook and Textgram have about 20 times leakages of address book when the user queries his friends in the social network.

These results indicate that an advertiser can potentially use the combination of private data (such as IMEI and NET-based location) to identify the location of the user from such apps, and further obtain a fine-grained picture of the user's social life with the assistance of the leaked contact information. Additionally, we observe that 7 apps shown in Figure 4 have leaked more than 4 types of private information, 3 of them have even leaked all 7 types of

ACCEPTED MANUSCRIPT

ACCEPTED MANUSCRIPT

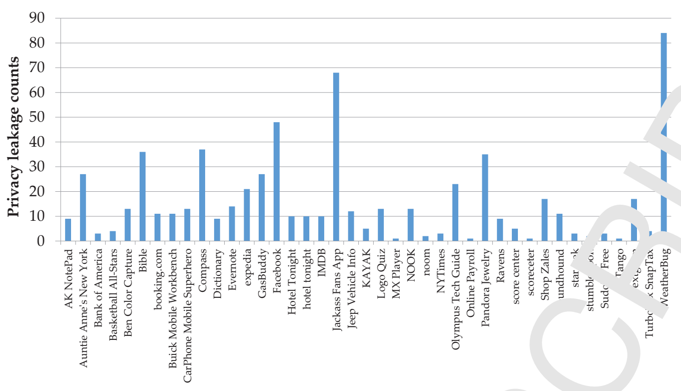


Figure 5: App leakage frequency to destination study: top 41 apps having leakages to multiple common destinations in 5-minute usage for each app.

private information (i.e., Pandora Jewelry, Carphone Mobile Superhero, and Evernote). This equips the advertiser with a comprehensive view of the users' private data through their daily phone usage.

We also find that it is common for different applications to have similar combinations and frequencies of privacy leakages. For example, GasBuddy and Breastmilk Counts have the same number of Net-based location and IMEI leakages in the 5-minute usage. Such applications have equivalent ability of revealing the user's private data. As the user continues using such "ability-equivalent" apps, the leaked private data keeps at similar levels all the time. The situation becomes more severe when these applications send the private data to the same destination.

Leakage to Common Destinations We take a closer look at the leakage frequency from different apps to a "common" destination. We first show the number of privacy leakages for the top 41 apps sending to common destinations in a 5-minute usage in Figure 5. We observe that WeatherBug and Jackass Fans are the top two apps with the most frequent leakages: they leak about 80/70 times during the 5 minute period. We further summarize the leakage frequency in app categories to common destinations and observe that the Weather category exhibits the highest privacy leakage frequency, partly due to their needs to know the user's location, and the Social category is the second. This again confirms that various apps leak private information to multiple common destinations, which allows the advertiser to piece together the user's social picture at fine temporal granularity through multiple apps.

4. User Privacy Leakage Modeling and Experimental Study

In this section, we present a privacy leakage inference framework that *quantifies to what extent an advertiser can learn and infer users' relationships.* We then run real experiments with multiple participants to analyze the consequences of the privacy leakage from the advertiser's

ACCEPTED MANUSCRIPT

ACCEPTED MANUSCRIPT

perspective and abstract privacy leakage user profiles based on the experiments.

4.1. Privacy Leakage Modeling

We first define the concept of *connection*. A *connection* between two users exists if the same type of privacy leakage from the two users share certain spatial, temporal or content similarities. A few examples are:

Contact list: A connection instance exists between two users if they are in each other's contact list, or they share common contacts. (However, we note that contact lists are not sufficient for social relationship inference simply a person does not necessarily to have close relationship with anyone in his or her contact list.)

Wi-Fi Access Point list: A connection instance exists between two users when they share common leaked access points at the same time.

GPS location: A connection instance exists between two users if two GPS locations leaked around the same time are close by within a certain threshold.

Network-based location: A connection instance exists between two users if the leaked network-based locations are close by within a certain threshold around the same time.

The *connection* bridges the gap between the privacy leakage information and the users' relationship inference. In particular, to quantify the consequences of the privacy leakage from the advertiser's perspective, we design a privacy leakage inference framework, which consists of three virtual layers: *Privacy Leakage Aggregation*, *User Connection Derivation*, and *Relationship Inference* as shown in Figure 6.

Such a framework facilitates us to perform a systematic study to understand the advertiser's perspective of user privacy: (1) The Privacy Leakage Aggregation layer deals with the raw privacy leakage information. An advertiser can combine the privacy leakage data from multiple apps across different users over time. For example, the users can be identified by the IMEI or phone number. The aggregated privacy leakage data of each user can then be categorized into different types, such as contact list, AP list, GPS location, and Network location. (2) In the User Connection Derivation layer, the advertiser correlates the data from different users and identifies connections [2] between any two users. By correlating different types of privacy leakages across the users over time, the connection frequency between any two users can be derived. (3) In the Relationship Inference layer, the user's social and community relationships, such as family, colleagues and friends, are inferred based on the connections between users.

[2]We use "connections" to refer to connection instances later in the paper.

ACCEPTED MANUSCRIPT

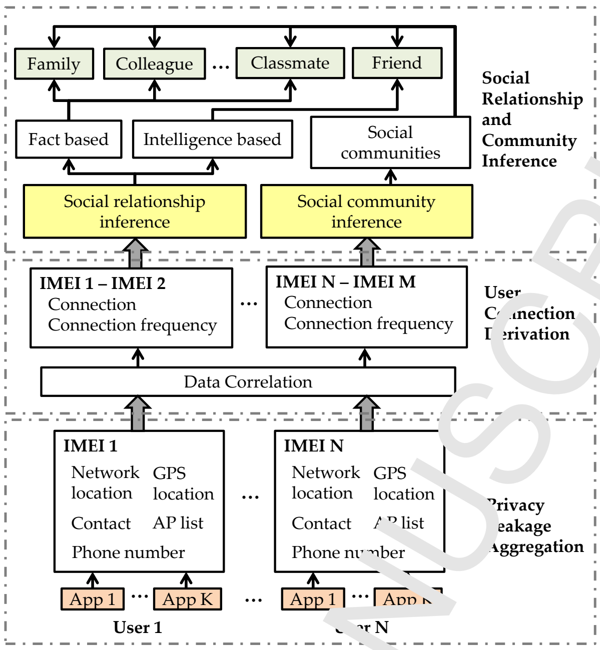


Figure 6: Privacy leakage inference framework with three virtual layers to quantify the advertiser's perspective of smartphone user privacy.

The type of relationship is usually determined by examining the temporal and spacial patterns of the connections (e.g., family members usually have connections at home in the morning and at night, whereas colleagues have connections in office during working hours). We next conduct an experiment to study the effectiveness of our privacy leakage model using this framework.

4.2. Experimental Study

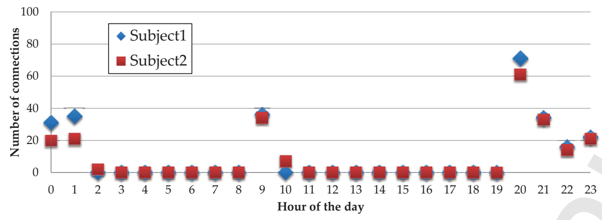
4.2.1. Design of Experiments

Our experiment involves 10 volunteer students and their family members over one month period, among which five types of relationships exist: colleague, collaborator, classmate, friend, and family. The ten volunteers are all graduate students between 21 to 24 years. Eight of them major in Computer Engineering and the other two major in non-computer-related major. All of them have been using smartphones for over two years and have moderate understandings of information technology. To clarify, collaborators are usually colleagues that actively work together, usually at regular times such as weekly meetings.

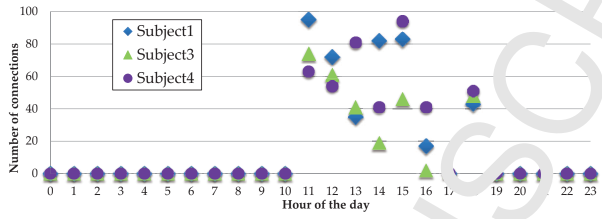
During the experiments, we distribute smartphones with our visualization tool (which is presented in Section VI) and the top 10 popular apps across 19 categories in Google Play sending to common destinations (shown in Figure 5) installed. Because the experimental smartphones are not replacements of the volunteers' regular phones, they are asked to use their experimental smartphones at least three times a day. There is no restriction of how and when

ACCEPTED MANUSCRIPT

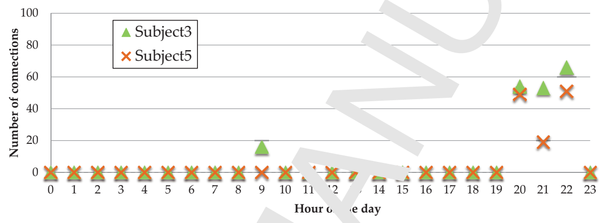
ACCEPTED MANUSCRIPT



Home, Fact based relationship (family)



Lab, Fact based relationship (colleague)



Coffee shop, Intelligence based relationship (friend)

Figure 7: Comparison of location leakage patterns of different relationships with respect to locations.

to use the apps in the experiments, and the volunteers are encouraged to use whichever apps they are interested in without knowing the purpose of this experiment. After the experiments we extract the leaked privacy data logged by our tool to quantify to what extent an advertiser can infer a user's relationships.

4.2.2. Observations

User Connection Derivation. Figure 7 shows one example on the temporal patterns of the derived user connections at three different locations based on GPS location leakage. We observe obvious spatial and temporal patterns of connections corresponding to the ground truth of the social relationships. In particular, subject1 and subject2 have frequent connections in the morning (around 10AM) and at night (through 9PM-2AM) at a residential area. The advertiser can thus infer the two subjects most likely have a family relationship. Further, subject1, subject3 and subject4 have connections frequently during working hours (through 11AM-7PM) in an office building. This follows a typical pattern of colleagues or collaborators. Additionally, the connections of subject3 and subject5 usually happen at early night (9PM-10PM) but do not show up late night (12AM-6AM) or morning. Such a pattern is more like friends hanging out together. From the above examples, the advertiser can infer users' social

ACCEPTED MANUSCRIPT

ACCEPTED MANUSCRIPT

relationship once it has accumulated enough privacy leakages over time.

Definition of Two Types of Social Relationships. From the experimental results, we observe that while some relationships (e.g., family, colleagues, collaborators and classmates) exhibit repetitive connection patterns, some others like friends do not. This is because family and colleague based relationships naturally carry similar spatial-temporal patterns dictated by the relationship. For example, families live together at night while colleagues work together during the day, whereas friendship does not necessarily carry such inherent patterns. Two friends that do not hang out for a while are still friends. We distinguish these two categories of relationships as *Fact Based Relationship* and *Intelligence Based Relationship*, which covers traditional social relationships. The *Fact-based Relationship* includes colleagues, classmates, roommates, families that carry inherit similar, regular and repetitive spatial-temporal connection patterns as dictated by the relationship, whereas the *Intelligence-based Relationship* includes friends, which do not necessarily carry such patterns.

4.3. Making Inference based on Thresholding

We next investigate how accurate an advertiser could infer about the user's social relationship such as colleagues, families and friends by utilizing connections between users derived from different types of privacy leakages.

We use a *threshold-based* approach to derive relationships based on the connections between users extracted from privacy leakage aggregation at the advertisement provider. If the connection count between two users exceeds a certain threshold in an observation window, we consider some relationship exists. Our framework utilizes the temporal and spatial patterns of the connections to classify the type of relationship: the connections of colleagues occur in work hours of weekdays, families in early morning and late night, while friends after working time and in weekends.

These simple rules may not be entirely reliable. Nevertheless, we show that an advertiser can make inference even with such simple rules. We note that such temporal and spatial patterns of the connections are the basis to statistically generate a user's privacy leakage profile, which will be described in Section 4.4.

In total we have 10 pairs of colleague relationship, 5 pairs of collaborator relationship, 1 pair of family relationship, 2 pairs of classmates, and 3 pairs of friend relationship among our 10 participants. That is 18 pairs of fact-based relationships and 3 pairs of intelligence-based relationship. During our experiments, we observe that by utilizing connection frequencies and patterns, an advertiser can infer over 90% social relationship correctly (i.e., all fact based

ACCEPTED MANUSCRIPT

ACCEPTED MANUSCRIPT

relationships and 2 pairs of intelligence based relationship). The only one friend relationship that is not correctly identified is because this pair of users only have one connection during the testing. This is inline with our expectation since the intelligence-based relationship does not carry repetitive spatial-temporal connection patterns, therefore their connection may not frequently repeat on different days, thus the relationship is harder to be identified.

4.4. Deriving Privacy Leakage User Profiles

Based on the experimental data collected over one month, we next build the privacy leakage user profile to statistically capture the temporal and spatial patterns. We develop two types of privacy leakage user profile, *activeness based profile* and *probability based profile*, which will be applied to our large-scale trace-driven studies on advertiser's perspective in the next section.

4.4.1. Activeness Based Profile

The activeness based profiles are generated based on privacy leakages from each participant in our experiments and aim to capture the fine-grained statistical view of the privacy leakages.

Step 1. We first derive the privacy leakage probability model of a particular user. Assume there are N types of privacy leakages observed in total. We divide the time in day d into T time windows as $\{w_t, t = 1, \cdots, T\}$. Then within a time window w_t, a vector $\Phi^{u,d,t}$ is defined to capture the numbers of occurrences of different privacy leakage types, and each element $\Phi^{u,d,t}(i)(i = 1, \cdots, N)$ corresponds to the number of times privacy leakage type i occurs. For example, when $\Phi^{u,d,t}$ equals to $[2, 1, 0]$, it means 2 occurrences of leakage type 1, 1 occurrence of leakage type 2 and 0 occurrence of leakage type 3 in time window W_t at day d for user u.

We then define $\gamma_i^{u,d,t}$ to indicate whether the privacy leakage type i appears in the vector $\Phi^{u,d,t}$ as:

$$\gamma_i^{u,d,t} = \begin{cases} 1, \Phi^{u,d,t}(i) \neq 0 \\ 0, \Phi^{u,d,t}(i) = 0. \end{cases} \tag{1}$$

The probability that type i leakage happens for user u in time window w_t across D days (e.g., $D = 7$ days) is defined as:

$$Prob_i^{u,t} = \frac{\sum_{d=1}^{D} \gamma_i^{u,d,t}}{D}. \tag{2}$$

Step 2. The number of occurrence of the privacy leakages affects the inference of a user's social community. Thus we capture the frequency of type i privacy leakage using the average number of occurrences over the days it happens in time window w_t across D days. Specifically,

ACCEPTED MANUSCRIPT

ACCEPTED MANUSCRIPT

the average rate $r_i^{u,t}$ is defined as:

$$r_i^{u,t} = \frac{\sum_{d=1}^{D} \Phi^{u,d,t}(i)}{\sum_{d=1}^{D} \gamma_i^{u,d,t}}. \tag{3}$$

The activeness based profile of user u consists of $Prob_i^{u,t}$ and $r_i^{u,t}$.

Example. We illustrate the generation of the activeness based profile of user u in Figure 8. We examine a privacy leakage dataset across 7 days (i.e., one week) with the time window w_t set to 5 minutes and 288 time windows in total per day. Assume 3 types of privacy leakage are under study. For w_t at day 2, if there are 2 occurrences of leakage type 2 and 8 occurrences of type 3 observed, we have $\Phi^{u,2,t} = [0, 2, 8]$ and $\gamma_2^{u,2,t} = 1$ shown as the green eclipse in Figure 8. In addition, if the leakage type 2 is only observed during day 1 and day 2 with $\Phi^{u,1,t} = [0, 5, 7]$ and $\Phi^{u,2,t} = [0, 2, 8]$, the privacy leakage probability of type 2 privacy leakage in time window w_t across 7 days for user u can be calculated using Equation 2 as: $Prob_2^{u,t} = \frac{\gamma_2^{u,1,t}+\gamma_2^{u,2,t}+\cdots+\gamma_2^{u,7,t}}{7} = \frac{1+1+\cdots+0}{7} = 0.286$. And the corresponding average rate can be obtained as: $r_2^{u,t} = \frac{\Phi_2^{u,1}+\Phi_2^{u,2}+\cdots+\Phi_2^{u,7}}{\gamma_2^{u,1,t}+\gamma_2^{u,2,t}+\cdots+\gamma_2^{u,7,t}} = \frac{5+2+\cdots+0}{1+1+\cdots+0} = 3.5$, which is shown in blue rectangles in Figure 8.

Categorization. Once the activeness based user profile is obtained, the advertiser could further categorize the profiles by the number of hours k_u the user u has privacy leakages in a one-day duration. There are three representative user categories, namely *active user category*, *regular user category*, and *inactive user category*. Assume two thresholding hours ρ_1 and ρ_2 with $\rho_1 > \rho_2$. If the user u has greater than ρ_1 hours with privacy leakages, his user profile is put into the active user category. If the user u has less than ρ_1 but larger than or equal to ρ_2 hours with privacy leakages, his user profile is then added into the regular user category. When the user u has less than ρ_2 hours privacy leakages, his user profile is then captured in the inactive user category. The categorization can be summarized as:

$$[illegible] = \begin{cases} 1 \text{ (active user category), if } k_u \geqslant \rho_1; \\ 2 \text{ (regular user category), if } \rho_2 \leqslant k_u < \rho_1; \\ 3 \text{ (inactive user category), if } k_u < \rho_2. \end{cases} \tag{4}$$

4.4.2. Probability Based Profile

The $\{Prob_i^{u,t}, r_i^{u,t}\}$ in each activeness based profile captures the leakage of the corresponding user u. To characterize the statistical average of the leakages of users in the same category, we design the probability based profile. Activeness based profiles in the same category are used to derive the leakage probability and rate for the probability based profile of that category.

ACCEPTED MANUSCRIPT

ACCEPTED MANUSCRIPT

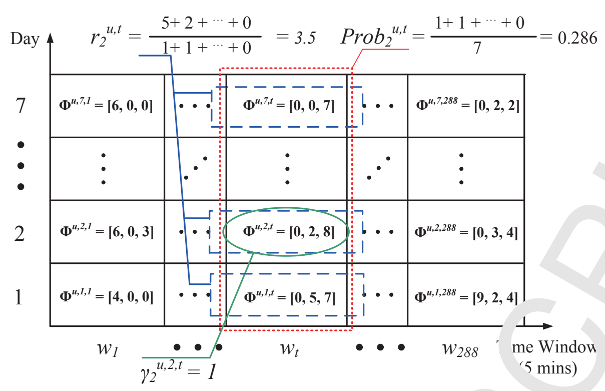

Day
$r_2^{u,t} = \frac{5+2+\cdots+0}{1+1+\cdots+0} = 3.5$
$Prob_2^{u,t} = \frac{1+1+\cdots+0}{7} = 0.286$
7
2
1
$\Phi^{u,7,1} = [6, 0, 0]$
$\Phi^{u,7,t} = [0, 0, 7]$
$\Phi^{u,7,288} = [0, 2, 2]$
$\Phi^{u,2,1} = [6, 0, 3]$
$\Phi^{u,2,t} = [0, 2, 8]$
$\Phi^{u,2,288} = [0, 3, 4]$
$\Phi^{u,1,1} = [4, 0, 0]$
$\Phi^{u,1,t} = [0, 5, 7]$
$\Phi^{u,1,288} = [9, 2, 4]$
w_1
w_t
w_{288}
Time Window
(5 mins)
$\gamma_2^{u,2,t} = 1$

Figure 8: Example of activeness based profile generation for user u with 3 types of privacy leakages when $D = 7$.

Step 1. We first define the $\delta_i^{u,t}$ to indicate whether there is a probability of type i leakage in the time window w_t for the profile of user u as:

$$\delta_i^{u,t} = \begin{cases} 1, Prob_i^{u,t} \neq 0 \\ 0, Prob_i^{u,t} = 0. \end{cases} \tag{5}$$

We calculate the average probability that leakage type i occurs in a time window for user u:

$$\lambda_i^u = \frac{\sum_{t=1}^{T} \delta_i^{u,t}}{T}. \tag{6}$$

Step 2. We then define the leakage probability of that category as the previous probability averaged over all users of the same category:

$$Prob_i^{\alpha} = \frac{\sum_{u=1}^{M_\alpha} \lambda_i^u}{M_\alpha}, \tag{7}$$

where M_α is the number of users belonging to the category α.

Step 3. The corresponding profile privacy leakage rate is calculated over all the users in one particular category α as

$$r_i^{\alpha} = \frac{\sum_{u=1}^{M_\alpha} \sum_{t=1}^{T} r_i^{u,t}}{\sum_{u=1}^{M_\alpha} \sum_{t=1}^{T} \delta_i^{u,t}}. \tag{8}$$

The probability based profile of a user category α then consists of $Prob_i^{\alpha}$ and r_i^{α}, which can be calculated based on the data from our experiments. We respectively name them as *high probability* ($Prob_i^{\alpha} = 0.87$), *medium probability* ($Prob_i^{\alpha} = 0.68$), and *low probability* ($Prob_i^{\alpha} = 0.44$) profiles.

Both types of profiles quantify the users' privacy leakage characteristics: the leakage probability $Prob_i^{u,t}, Prob_i^{\alpha}$ determine whether type i privacy leakage happens or not in a time window, whereas the average leakage rate $r_i^{u,t}, r_i^{\alpha}$ determine the number of type i leakages in

ACCEPTED MANUSCRIPT

ACCEPTED MANUSCRIPT

that time window should they happen at all. The profiles will be used in our large-scale trace-drive studies in the Section 5 to facilitate the understanding of the user relationship inference from the advertiser's point of view.

5. Social Relationship Inference Leveraging Privacy Leakages

In this section, we systematically study the consequence of the privacy leakages obtained by advertisers by applying the privacy leakage model to two human mobility traces. We build a simulator utilizing the privacy leakage inference framework to generate connections between users based on both the leakage profiles derived from Section 4 and the human mobility traces. The human mobility traces are used to discover connections between users in both an academia and a city environment.

5.1. Methodology

5.1.1. Human mobility traces

We use two human mobility traces: the MIT trace [18], and the Foursquare trace [19]. We choose these two traces mainly because the users in these traces come from different backgrounds and have various relationships. Specifically, the MIT trace represents participants with similar background in an academia environment, where user relationships mostly represent research colleagues, office staff and classmates. In the Foursquare trace, participants have more diverse relationships; they may be colleagues, friends, and families in a city environment. The details of these two traces are introduced below:

Foursquare Trace. Foursquare is a company that helps people share life experiences based on locations such as restaurants. This trace is generated based on tipping information collected from different venues in Los Angeles (LA). A tip in a venue shows that one participant has carried out some essential activities (like dinning and shopping) at that venue. There are 104,478 tips left by 31,544 participants in this trace. We choose 354 participants from the top 10 venues (which are all restaurants) to generate encounter events between participants based on the time of the tipping in a 21 day duration.

MIT Trace. This trace is collected on MIT campus for 10 months by 107 participants with smartphones. Each smartphone scans (using Bluetooth) and records nearby smartphones every five minutes. The encounter happens when two participants are located in close physical proximity (e.g., shown in the Bluetooth scanned neighboring list in MIT Trace). And such an event is defined as an *encounter event*. There are 97 participants with valid data including staffs and students. In our study, we use 21 days' data which includes 91 participants for social relationship inference.

ACCEPTED MANUSCRIPT

5.1.2. Privacy Leakage Profile Population

To understand the impact of user profiles, we repeat the study using both activeness and probability based profiles. When activeness based ones are used, each participant is assigned a randomly selected profile in the chosen category (i.e., active, regular and inactive). When probability based ones are used, each participant is assigned the same probability profile (i.e., one of high, medium and low). When presenting our results, we will use terms like "active users" or "users of medium probability" to (loosely) refer to participants assigned of the activeness or probability based profiles.

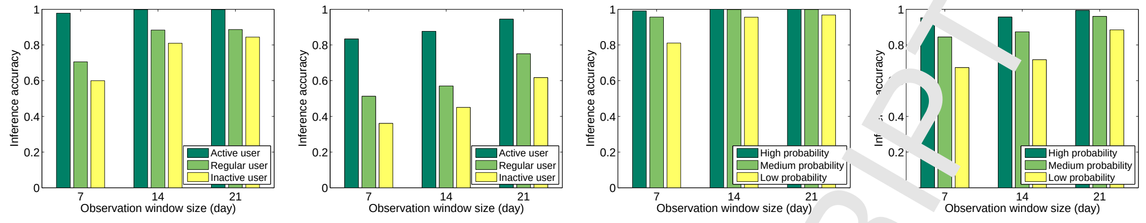
We then infer the relationships from the leakages over observation windows of different sizes (i.e., 7, 14 and 21 days). The inference is based on connections derived from the leaked data. From real experiments with 10 participants having known relationships, we find that different thresholds of connection counts in the observation window should be applied to derive different relationships. In addition, depending on the backgrounds of the environment where the datasets are collected (e.g., MIT trace collected on campus involves more academia relationships, and Foursquare trace collected in a city contains even more diversity of relationships), the thresholds could also be different.

The appropriate value for the threshold is a trade-off between accuracy and false positive [3]. If the threshold is too high, we may miss some real relationships and have low accuracy; if the threshold is too low, we will "identify" nonexistent relationships and have high false positives. In our work, we respectively use 3 days and 2 days for fact-based and intelligence-based relationships in both experiments and simulations. Such thresholds enable an advertiser to achieve over 90% inference accuracies for regular users with very small false positive rate, which is a good balance between the two.

Foursquare Trace In order to apply privacy leakage profiles to this dataset, we generate encounter events between participants as follows: for a particular venue, we give a visiting duration with a random length ranging from 30 minutes to 2 hours to each tipping user. In the overlapped period of the duration of two users, we generate encounter events with a fixed time interval of 30 minutes (e.g., in an 1 hour overlapped period, we generate 2 encounter events). For each encounter event, we first find out corresponding 5-minute time windows. Then we flip a coin with the privacy leakage probability in the users' profile (defined in Equation (2) or (7)) to decide whether privacy leakages should happen or not in that 5-minute time window.

[3]The accuracy and false positive rate are defined in Section 5.B.

ACCEPTED MANUSCRIPT



(a) Fact-based (b) Intelligence-based (c) Fact-based (d) Intelligence-based

Figure 9: Inference accuracy, MIT mobility trace, (a) and (b) are from the activeness based profiles using 3 days as the threshold, (c) and (d) are from the probability based profiles using 2 days as the threshold.

If they do, we use the privacy leakage rate defined in Equation (3) or (8) as the number of leakages revealed to the advertiser in that particular 5-minute time window. The leakages are used by the advertiser to derive connections and encounter events to infer users' relationship.

MIT Trace. The MIT trace records encounter events for each participant every 5 minutes. For each encounter event, we first find out the corresponding 5-minute time window. Then for each of the two participants, we flip a coin with the privacy leakage probability in his profile (defined in Equation 2 or Equation 7) to decide whether privacy leakages should happen or not in that 5-minute time window. If they do, we use the privacy leakage rate defined in Equation 3 or 8 as the number of leakages revealed to the advertiser in that particular 5 minute time window. The leakages are used by the advertiser to derive connections and eventually encounter events to infer users' relationship.

5.2. Metrics

In our evaluations, we study the inference accuracy of pairwise social relationship and the correlation between social communities extracted based on the connections of users.

Inference accuracy. This is the ratio between the successfully inferred relationship pairs and all relationship pairs.

Community correlation. This is the ratio between the common subjects within a community identified by our privacy leakage inference framework and the total number of subjects within the community.

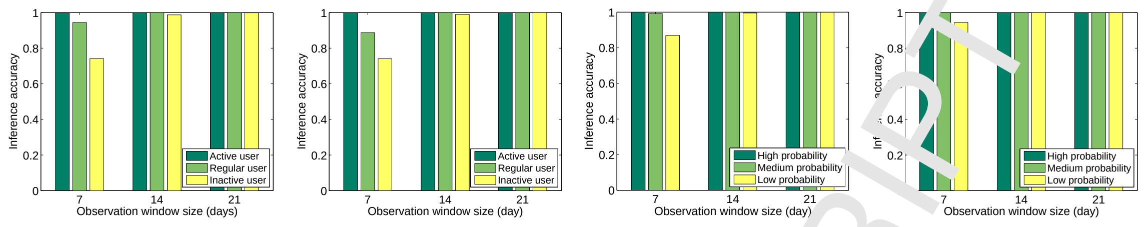
False positive rate. This is the ratio between the number of mistakenly identified members of inferred community and the total number of members within the inferred community.

5.3. Inference with Privacy Leakages

5.3.1. Combination of Privacy Leakages

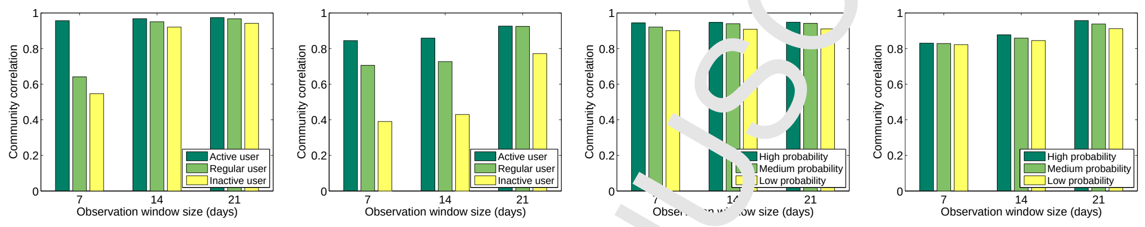
As we discussed in Section 4, an advertiser can utilize the temporal and spatial patterns of connections to infer users' relationship. There are multiple privacy leakages that can produce

ACCEPTED MANUSCRIPT



(a) Fact-based (b) Intelligence-based (c) Fact-based (d) Intelligence-based

Figure 10: Inference accuracy, Foursquare mobility trace, (a) and (b) are from the activeness based profiles using 3 days as the threshold, (c) and (d) are from the probability based profiles using 2 days as the threshold.



(a) Fact-based (b) Intelligence-based (c) Fact-based (d) Intelligence-based

Figure 11: Community correlation, MIT dataset, (a) and (b) are from the activeness based profiles using 3 days as the threshold, (c) and (d) are from the probability based profiles using 2 days as the threshold.

connections between users. In this study we focus on the {user identity, location} combinations of most popular privacy leakages including IMEI, phone number, GPS location, Wi-Fi AP list, and network-based location.

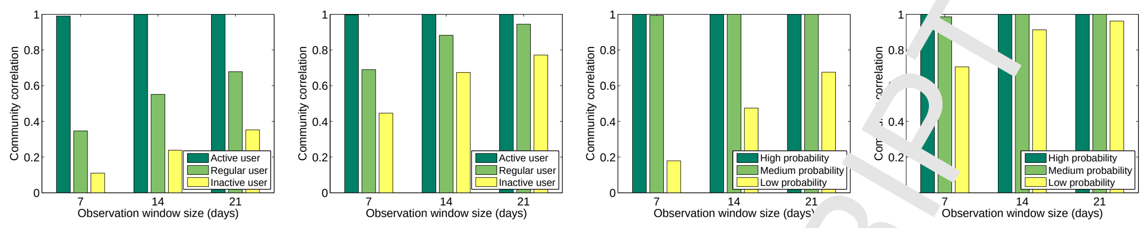
5.3.2. Pairwise Social Relationship Inference

Figure 9 compares the accuracy of pairwise social relationship inference (for both fact and intelligence based relationships) by applying activeness and probability based profiles to the MIT trace under different sizes of observation windows. We use 3 days and 2 days as the threshold for fact-based and intelligence-based relationship inference respectively, which is introduced in Section 4 (same threshold applies hereafter). The fact-based relationship has greater threshold because people having fact-based relationships are supposed to encounter each other more regularly than those having intelligence-based relationships (e.g., colleagues meet 3 days a week while friends meet 1 day a week).

From Figure 9 (a) and (b) we observe that for most cases, an advertiser can achieve over 80% inference accuracy for fact-based relationships with regular users, whereas it is around 60% for intelligence-based relationships. We also observe that the inference accuracy decreases for less active users, which is reasonable since less usage leads to less privacy leakages. In addition, for active users, Figure 9 shows that the inference accuracy for both fact-based relationship and intelligence relationship is high (i.e., above 90%).

ACCEPTED MANUSCRIPT

ACCEPTED MANUSCRIPT



(a) Fact-based (b) Intelligence-based (c) Fact-based (d) Intelligence-based

Figure 12: Community correlation, Foursquare dataset, (a) and (b) are from the activeness based profiles using 3 days as the threshold, (c) and (d) are from the probability based profile using 2 days as the threshold.

Furthermore, we find that longer observation windows help improve the inference accuracy, especially when the privacy leakage probability is low. This is observed in the probability-based approach shown in Figure 9 (c) and (d). It is because a longer window helps the advertiser to accumulate more data, resulting in more connections to identify users' relationships. Comparing Figure 9 (a) and (b) to Figure 9 (c) and (d) respectively, we observe that the inference accuracy of active users is similar to that of users have the profile with a high leakage probability. Figure 9 (d) suggests that in order to keep the inference accuracy of intelligence based relationship lower than 60%, the user has to keep his leakage probability smaller than *low probability* (i.e., 0.44).

Examining the Foursquare trace, we observe much higher inference accuracy for both fact and intelligence based relationships. We show the results using privacy leakage profile with different activeness in Figure 10 (a) and (b). For inactive users, the inference accuracy is about 70% for a 7-day window, and it goes over 95% for the 14-day window. This is because users in the Foursquare trace encounter each other more frequently than those in the MIT trace. Thus, more connections can be discovered when the users have the same intensity of app usage, leading to a higher inference accuracy. It is also the case when using the probability-based profile to infer pairwise relationship with the Foursquare trace (i.e., Figure 10 (c) and (d)).

5.3.3. Social Community Inference

We next study how the social community could be inferred by the advertiser using the privacy leakage profile. In particular, based on the inferred pairwise social relationships, a hierarchical clustering algorithm [25] is applied to obtain the social communities of users with similar relationships (e.g., collaborators, labmates, and classmates).

Community Correlation. Figure 11 shows the community correlations calculated when applying the activeness and probability based profiles on the MIT trace with three different observation windows. We observe that overall the community correlation is above 60% for

ACCEPTED MANUSCRIPT

Table 2: False positive rate for community correlation: MIT trace and Foursquare trace.

	Fact-based			Intelligence-based		
	7days	14days	21days	7days	14days	21days
MIT Trace						
Active users	0.035	0.022	0.016	0.149	0.129	0.098
Regular users	0	0	0	0.065	0	0
Inactive users	0	0	0	0.056	0	0
High prob.	0.069	0.054	0.052	0.17	0.145	0.12
Medium prob.	0.045	0.041	0.039	0.104	0.093	[illegible]
Low prob.	0.026	0.016	0.015	0.055	0.03	0.0257
Foursquare Trace						
Active users	0	0	0	0.019	0.07	[illegible]
Regular users	0	0	0	0.081	0.061	0.052
Inactive users	0	0	0	0.04	[illegible]	0.029
High prob.	0	0	0	0.19	0.078	0.078
Medium prob.	0	0	0	0.18	[illegible]	0.078
Low prob.	0	0	0	0.099	0.064	0.058

regular users. Furthermore, we observe that the community correlation also increase with longer observation windows, especially for regular users and inactive users. This is also because longer observation windows help the advertiser to aggregate more connections between users, which helps to more accurately identify their social communities.

We present the community correlations of both fact based relationship and intelligence based relationship when activeness and probability based profiles are applied to the Foursquare trace in Figure 12. We observe that the intelligence based community correlation is high for active users and regular users (i.e., over 80% on average). Similarly it is high for users with high probability and medium probability profiles (i.e., over 90%), which is comparable to those of the MIT trace. However, the community correlation of users having the fact-based relationship is much lower with inactive users and users with low probability profiles (i.e., ranges from 10% to 60%) than that of the MIT trace. This is because participants in Foursquare data are from much diverse background in the city environment and the locations are mostly restaurants which favors more to the intelligence-based relationship inference.

Discussion of False Positive. Table 2 shows false positive rate of community correlation for both MIT and Foursquare traces. Overall, the false positive rate is very small. In addition, larger observation window has smaller false positive rates. This indicates that longer observation window size has more information and can improve the inference performance. Furthermore, we observe that the Foursquare trace has much lower false positive rate than the MIT trace. This is because the Foursquare trace has more connections, thus leads to lower false positive rate.

To briefly summarize the major findings, 1) the advertiser can infer users' social relation-

ACCEPTED MANUSCRIPT

ships at high accuracy, e.g., over 90% on average for active users and over 80% on average for regular users; 2) the advertiser can also infer a significant portion of users' community relationships, e.g., over 90% on average for active users and over 60% on average for regular users, which reveals common interests or activities among users not necessarily with direct interactions. The results from the simulation confirm the findings from our real experiments with the relatively smaller number of participants and shorter term.

6. Privacy Leakage Visualization

Since advertisers can infer users' relationships and communities from the privacy leakages, a visualization tool that presents such leakages in real time may help users better adapt their app usage. Such tool would reveal the detailed temporal and spatial characteristics of privacy leakages to complement existing works, such as the crowdsourcing-based privacy setting tool proposed in [13].

6.1. Tool Features

We present our privacy leakage visualization tool that can display the leakage statistics at different levels of granularity. Existing visualization tools usually address each app individually or focus on a specific type of privacy leakages [26, 27]. For example, by displaying the privacy data that an app intents to access, existing work allows the user to block such privacy leakage [20] or helps her obfuscate the privacy information [11, 12].

In contrast, our tool provides spatial-temporal visualization of the leaked private data at multiple levels of granularity, including the *global level*, whereby the statistics over all apps and destinations (i.e., advertisers) is summarized; the *destination level*, whereby the privacy leakages received at each destination are shown; and the *app level*, whereby the leakage can be displayed per app and per user's location. Such comprehensive statistical visualization avoids presenting low-level raw data to users who are mostly not tech-savvy and cannot make sense out of the raw data. It facilitates a better understanding of the privacy leakage and provides users opportunities to adjust her app usage pattern to control such leakage.

Our tool utilizes TaintDroid [3] (with Android OS 2.3.4) to capture the leakage information. The tool runs in background and logs the details of each privacy leakage event including the privacy type, the content of leaked data, the app name, the destination, and a timestamp. It consists of five main components: *Global Viewer*, *Map Viewer*, *Statistics Viewer*, *App-Destination Viewer*, and *App Leakage Warning.* The main components and the flow of the tool are depicted in Figure 13.

ACCEPTED MANUSCRIPT

ACCEPTED MANUSCRIPT

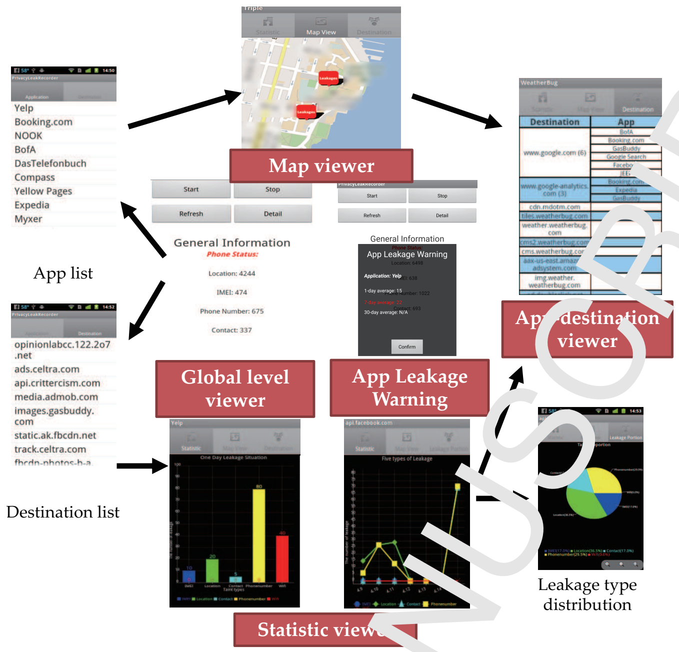


Figure 13: GUI flow of the visualization tool.

Global Viewer. It presents the overall privacy leakages of the mobile device by aggregating the leaked information from all apps in a user defined time period.

Map Viewer. It shows a detailed spatial distribution of leakage information including the type and occurred time over a map.

Statistics Viewer. It provides temporal statistics of different types of privacy leakage using scatter plots and pie charts.

App-Destination Viewer. It displays the connection between apps and destinations: what private data is leaked from which apps to the same destination. The destination can integrate the data from all these apps to infer knowledge about the user.

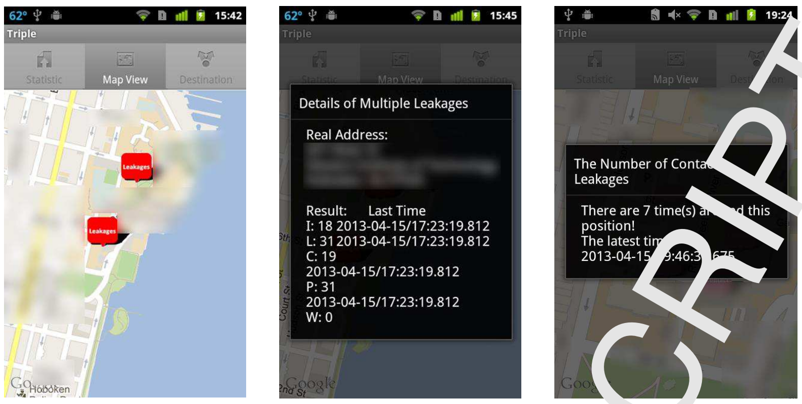
App Leakage Warning. The visualization tool dynamically sends active warnings to users based on the average leakages of apps. The warning message includes a summary of the average leakages and provides a shortcut link to the detailed statistics of the privacy leakages of the highlighted app.

6.2. Global Viewer

The visualization tool starts with the Global Viewer, which shows the total numbers of privacy leakages of each data type (i.e., Phone number, IMEI, Location, Contact list, and Wi-Fi AP list, as illustrated in Figure 13) over a user-defined time period or the total history of the phone usage. The Global Viewer has two branches: *App List* or *Destination List*, which show

ACCEPTED MANUSCRIPT

ACCEPTED MANUSCRIPT



Details of Multiple Leakages

Real Address:
[redacted]

Result: Last Time
I: 18 2013-04-15/17:23:19.812
L: 31 2013-04-15/17:23:19.812
C: 19
2013-04-15/17:23:19.812
P: 31
2013-04-15/17:23:19.812
W: 0

The Number of Contact Leakages

There are 7 time(s) around this position!
The latest time
2013-04-15 9:46:3 675

(a) Two places of multiple leakages (b) List of leakages with detailed information (c) Selected one detailed leakage

Figure 14: Screenshots of multiple privacy leakages found to be leaked at different locations.

the lists of apps that leak private data and destinations that receive it (shown in Figure 13). By observing the privacy leakage counts per app, a user can easily tell which app has the most frequent leakage of sensitive data.

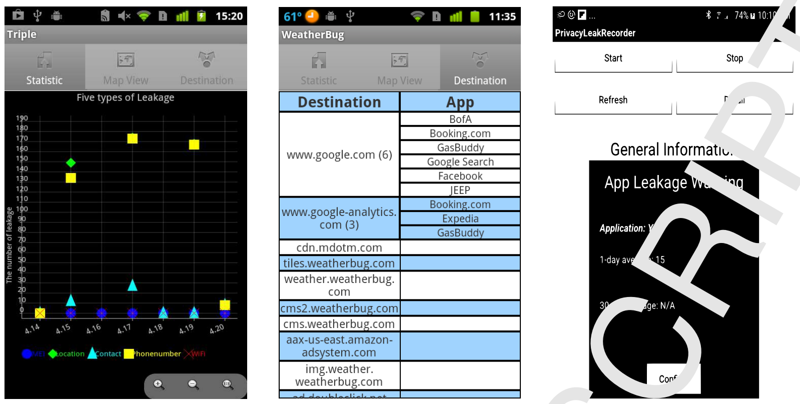
6.3. Map Viewer

The Map Viewer shows the physical locations where the private data is leaked. The Map Viewer can be accessed from the App list or the Destination list, which will show on the map all the leakages from one app or to one destination (e.g., Figure 14 (a)). The user can also further refine to only showing a certain type of privacy leakage per app or per destination. Clicking an icon will display the details of privacy leakage at that location. Figure 14 (b) and (c) illustrate the detailed information of all types of privacy leakages and the detailed information of contact information leakage, respectively. We translate the GPS coordinates into the corresponding physical address using Google Maps API for easy understanding to the user.

One implementation issue is how to display the privacy leakage events when a lot of them occur around the same location. This happens when the user stays at the same location for some time during which multiple leakage events of different types can happen. The Map Viewer combines events of the same privacy type and shows them as one icon on the map. It determines whether two locations can be combined if their distance is less than a threshold.

Different types of privacy leakages are displayed in icons of different colors. When too many events occur at one location, a single red icon is shown to avoid crowdedness. Figure 14 (a) displays multiple privacy leakages happening at two locations (with two red icons for aggregated privacy leakages). Clicking on the icon will display the detailed information about the privacy leakages, such as the physical address, historical counts of different types of privacy leakages, and the most recent leakage time (shown in Figure 14 (b)). Clicking on an icon of a specific

ACCEPTED MANUSCRIPT



Destination	App
www.google.com (6)	BofA
	Booking.com
	GasBuddy
	Google Search
	Facebook
	JEEP
www.google-analytics.com (3)	Booking.com
	Expedia
	GasBuddy
cdn.mdotm.com	
tiles.weatherbug.com	
weather.weatherbug.com	
cms2.weatherbug.com	
cms.weatherbug.com	
aax-us-east.amazon-adsystem.com	
img.weather.weatherbug.com	

(a) Privacy sinks (b) Other apps sharing the destination (c) App Leakage Warning message

Figure 15: Screenshots of (a) & (b) App-Destination Viewer and (c) App Leakage Warning.

type of privacy leakage will display the historical counts of that type of leakage and the most recent leakage time (shown in Figure 14 (c)).

6.4. Statistics Viewer

The Map viewer summarizes the spatial distribution of privacy leakages. The Statistics Viewer reveals the temporal patterns of each type of leakage events such as IMEI, GPS location, Wi-Fi Access Point SSID list, contact list, and phone number. The Statistics Viewer plots the numbers of different types of privacy leakages over a time window, whose sizes are 1-day, 7-day, and 30-day to show the leakage pattern in both short-term and long-term. Figure 15 (a) depicts the statistics over a 7-day period. Leakage patterns in fine-grained time slots (e.g., one-hour) are also available in bar graphs. These statistics potentially can help a user identify apps leaking private data aggressively and alter his/her usage patterns of these apps (e.g., use them less frequently or stop altogether).

6.5. App-Destination Viewer

The App-Destination Viewer presents how much private data each destination can obtain across multiple apps, which determines the scope and depth of the knowledge one advertiser may infer about the user. Figure 15 (b) presents an example of App-Destination Viewer for the app WeatherBug. The screen is separated into two columns: the left column lists the destinations to which the selected app has sent private data, and the right column shows other apps that also send private data to the same destinations.

6.6. App Leakage Warning

In addition to providing various interfaces for users to have the complete and detailed picture of the privacy leakages in their smartphone use, we design an *App Leakage Warning*

ACCEPTED MANUSCRIPT

mechanism, which can actively send a warning message to users about a particular app when the average leakages of the app have crossed a certain threshold. Figure 15 (c) presents an example of the App Leakage Warning message for the app Yelp. The message includes the name of the app, average leakages in different scales (i.e., 1-day, 7-day, and 30-day). The average leakage crossing the threshold would be highlighted in red. Users could directly go the detailed statistics by tapping the name of the app in the message.

7. Discussion

More Types of Private Data. There are potentially other types of private data available to advertisers. For example, Google has the access to its search terms and histories of many users. Although we do not find the 190 apps studied leaking text message logs, audio and video data, illegitimate access and disclosure of such information are not impossible. Our current study is based on the combination of most basic privacy leakages (i.e., identities and locations). Contact list carries important information. For example, a certain relationship most likely exists when two users share many common contacts. Unfortunately, because we do not have the contact lists of the subjects of the two mobility traces, we are not able to evaluate its impact on relationship inference. Studying the impact of more types of private data would be interesting future work.

Large Scale Evaluation. We are aware that our experiments (due to limited available manpower) may cause bias to our privacy leakage profiles and the evaluation may not cover all privacy leakage patterns in the real world. In particular, the ten volunteers in our experiments are most graduate students and only two of them are family-related. We mainly use this experiment to demonstrate the possibility of revealing users' social relationships by using the spatial and temporal information of the privacy leakages in their smartphone use. In addition, we are also aware that our experiments may be limited to independent app behaviors due to the limited number of apps used in our real experiments. Involving more apps running simultaneously in the same phone would better reveal the privacy leakages of apps having certain dependencies. Increasing the number of participants and apps in real experiments and building more sophisticated privacy leakage profiles are our future exploration as well.

Insufficient Metadata. Our evaluation is constrained by the availability of metadata descriptions of datasets. Neither the MIT nor Foursquare data has sufficient annotation to differentiate the relationships among users at fine granularity desired by us: colleague, collaborator, classmate, friend, and family. In lieu of that, we have to utilize the most commonly used technique for detecting social communities (i.e., hierarchical clustering algorithm [25]) and

ACCEPTED MANUSCRIPT

ACCEPTED MANUSCRIPT

use the results as the base of comparison, which is a common practice in social relationship research. In the future we hope such metadata would be made available when people conduct such experiments.

More Advanced Inference Algorithms. The thresholding algorithm that we use to infer relationships is based on quite simple heuristics. With the availability of large amount of data, advertisers can use more advanced inference algorithms, e.g., by utilizing data mining techniques. In addition, certain relationship may have similar spatial and temporal patterns that may be hard to distinguish by using the naive algorithm, such as family and friends if they live together. However, it may be possible to reveal the difference based on even finer-grained analysis. For example, family members usually have breakfast and dinner together at the same time while friends may have totally different schedules. We are aware of this limitation and plan to build advanced models more closely describing general cases.

Time Guidelines from the Visualization Tool. Our visualization tool provides spatial temporal statistics of privacy leakages, which gives the user better ideas of the scope and degree of privacy loss. Ideally, based on the type, destination, location and frequency of the leakages, an inference algorithm can be used to estimate how much of the user's relationships can be derived. The tool would then be able to provide app usage guidelines in real time, such as stopping using certain apps for some time, when certain conditions are triggered (e.g., a certain fraction of a kind of relationship is exposed). This estimation requires certain knowledge about the usage patterns of other related users, available to advertisers but not to us without a wide enough installation base of the tool. How to make educated guesses on such knowledge would be one interesting research issue.

Emerging Visualization Tool. We are also aware there are several visualization tools has been proposed recently [28, 29, 30]. XDroid*et al.* [28] is a machine-learning based tool that can detect malware based on apps' behaviors. Liu [29] proposed a personal privacy assistant that can build permission profiles based on people's choices and provide permission suggestions to users. Similarly, DroidNet [30] could help users to detect malware and make decisions on enabling apps' permissions based on the suggestions from experts through the cloud service. However, all these tools are focusing on permissions setup and do not reveal real-time privacy leakages and the more serious consequence of privacy leakages integrated at the same destination.

ACCEPTED MANUSCRIPT

ACCEPTED MANUSCRIPT

8. Related Work

Enck et al. [10] are the first to conduct permission analysis to identify dangerous functionalities using Android permissions, such as tracking user or voice eavesdropping. They propose Kirin, which performs inspections on Android API permissions during the app installation time. They examine 311 top free applications and identify several questionable applications sending out users' private information. Barrera et al. [5] perform permission analysis of the top 1,100 free applications and report many applications request only a small set of permissions. Felt et al. [6] study 100 paid and 856 free applications from the Android Market and find INTERNET permission is the most frequently requested. They later propose Sowayway [9] to detect over-privilege in applications and report 10 most common unnecessary permissions. Taylor et al. [35] analyze a privilege escalation attack, where the third-party libraries can get access to sensitive data from devices using intra-library collusion. And they prove that several popular libraries facilitate this attack by collecting enough sensitive data. Pennekamp et al. [36] provide a survey, which reviews the methods that have been proposed to check the applications permission and their access to sensitive information.

Static analysis analyzes the code of applications to infer what can happen to users' security. For example, PiOS [31] analyzes compiled Objective-C code to identify information leaks on the iOS platform, whereas ComDroid [32] uses disassembled DEX bytecode to identify vulnerabilities in Intent communication between applications. Enck et al. further propose the *ded* decompiler [34] to reverse Android applications to Java code for security analysis. FlowDroid [37] can detect private data leaks in Android apps by performing a data flow analysis, which becomes a highly popular static analysis framework. DroidSafe [38] uses the static analysis on Android information flow to report the potential leaks of sensitive information.

Less work has been done in dynamic analysis. TaintDroid [3] tracks the flow of privacy-sensitive data through third-party applications. It detects when sensitive data leaves the system via interfaces such as network connections. Agarwal et al. [13] propose ProtectMyPrivacy (PMP) which utilizes a crowdsourcing-based mechanism to help users decide proper privacy settings for iOS apps, such as denying access to private data or substituting anonymized data. He et al. [39] propose a privacy leakage analysis framework for third-party libraries in real time using both combined static and dynamic Xposed methods. Chen et al. [40] propose a method called HybriDroid which gets the advantage of both static and dynamic analysis methods to identify the private data leakage. Using the combination of both static and dynamic analysis, HybriDroid can detect data leakage for both inter and intra-app communications with the

ACCEPTED MANUSCRIPT

ACCEPTED MANUSCRIPT

high accuracy. Brandtzaeg et al. [41] also apply the dynamic analysis to analyze the data flow and measure activity in the apps over a long time, which strengthens the understanding of the complexity of privacy issues in mobile apps. Our work takes a different viewpoint by systematically analyzing what privacy-sensitive information the advertiser can collect and aggregate at run-time from multiple apps and infer the social relationship of a user. We utilize TaintDroid as a tool to track and log the run-time privacy leakage from apps.

A few app tools are developed to provide a user with the privacy leakage information [11, 7, 8]. However, the information is provided in low-level and raw data format based on each app, which only gives the user a narrow view of the privacy leakage on a per-app basis. Our visualization tool seeks to provide a multi-level statistical view of privacy leakage and gives the user a better understanding of the implication of leaked private information.

Finally, some work develops smartphone platform based privacy protection mechanisms. Ongtang et al. propose Saint [26] and Porscha [27] by extending the functionality of the Kirin system to allow runtime permission inspection by defining runtime policies. Bugiel et al. propose XManDroid [42] to mitigate permission privilege escalation attacks in Android by tracking communication between components in different applications. To prevent smartphone applications from leaking phone identifiers and location information, MockDroid [11] and TISSA [12] provide empty, fake or anonymized information to applications. AppFence [20] builds on top of Taintdroid to actively block network transmissions containing user-defined sensitive information, which should be used on the device only. Our work focuses on a different aspect of gaining a systematic understanding of the social relationship inference consequences from the privacy leakage by an advertiser. Such understanding may motivate the user to adjust app usage pattern or adopt defense mechanisms to control the sensitive data leakage.

9. Conclusion

Privacy leakage by smartphone apps has attracted significant research efforts in recent years. The community has proposed various defense mechanisms, from permission management, code analysis, to obfuscated data. Nevertheless, the characteristics of apps' run-time privacy leakage behavior is still not well investigated, and the consequences of such privacy leakages have not attracted much attention. This paper serves as the first step towards a comprehensive understanding of the advertiser's perspective. In particular, we seek to discover what an advertiser can infer about users' social and community relationships by combining private data from many apps. Our analysis on the run-time privacy leakage behavior of nearly 200 most popular apps from 19 categories of Google Play shows that dominant advertisers

ACCEPTED MANUSCRIPT

can easily gather data from many apps. We propose a privacy leakage inference framework that describes a general method for inferring users' social and community relationships. Our experimental study over one month demonstrates that an advertiser can infer 90% of users' social relationship correctly using simple heuristics. This observation is further confirmed by human mobility trace driven studies of two large scale data sets. We finally build a visualization tool that can capture and display the spatial-temporal statistics of privacy leakage to different advertisers in real time. We hope our work will eventually lead to a complete picture of the advertiser's perspective.

10. Acknowledgement

This work is supported in part by the National Science Foundation Grants SES1450091 and CNS1217387.

[1] M. C. Grace, W. Zhou, X. Jiang, A.-R. Sadeghi, Unsafe exposure analysis of mobile in-app advertisements, in: Proceedings of the fifth ACM conference on Security and Privacy in Wireless and Mobile Networks, 2012.

[2] R. Stevens, C. Gibler, J. Crussell, J. Erickson, H. Chen, Investigating user privacy in android ad libraries, IEEE Mobile Security Technologies (MoST).

[3] W. Enck, P. Gilbert, B.-G. Chun, L. P. Cox, J. Jung, P. McDaniel, 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, 2010.

[4] A. P. Felt, E. Ha, S. Egelman, A. Haney, E. Chin, D. Wagner, Android permissions: User attention, comprehension, and behavior, in: Proceedings of the Eighth Symposium on Usable Privacy and Security, 2012, pp. 1–14.

[5] D. Barrera, H. G. Kayacik, P. C. van Oorschot, A. Somayaji, A methodology for empirical analysis of permission-based security models and its application to android, in: Proceedings of the ACM conference on Computer and communications security, 2010.

[6] A. P. Felt, K. Greenwood, D. Wagner, The effectiveness of application permissions, in: WebApps, 2011.

ACCEPTED MANUSCRIPT

ACCEPTED MANUSCRIPT

[7] X. Wei, L. Gomez, I. Neamtiu, M. Faloutsos, Profiledroid: multi-layer profiling of android applications, in: Proceedings of the 18th annual international conference on Mobile computing and networking, 2012, pp. 137–148.

[8] Lamian, LBE Privacy Gurad, https://play.google.com /store/apps/.

[9] A. P. Felt, E. Chin, S. Hanna, D. Song, D. Wagner, Android permissions demystified, in: ACM CCS, 2011.

[10] W. Enck, M. Ongtang, P. McDaniel, On lightweight mobile phone application certification, in: ACM CCS, 2009.

[11] A. R. Beresford, A. Rice, N. Skehin, R. Sohan, Mockdroid: trading privacy for application functionality on smartphones, in: Proceedings of the 12th Workshop on Mobile Computing Systems and Applications, 2011, pp. 49–54.

[12] Y. Zhou, X. Zhang, X. Jiang, V. W. Freeh, Taming information-stealing smartphone applications (on android), in: Trust and Trustworthy Computing, 2011.

[13] Y. Agarwal, M. Hall, Protectmyprivacy: detecting and mitigating privacy leaks on ios devices using crowdsourcing, in: MobiSys, 2013.

[14] M. Nauman, S. Khan, X. Zhang, Apex: extending android permission model and enforcement with user-defined runtime constraints, in: Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security, 2010, pp. 328–332.

[15] K. W. Y. Au, Y. F. Zhou, Z. Huang, D. Lie, Pscout: analyzing the android permission specification, in: ACM CCS, 2012.

[16] C. Wilson, B. Boe, A. Sala, K. P. Puttaswamy, B. Y. Zhao, User interactions in social networks and their implications, in: Proceedings of the 4th ACM European conference on Computer systems, 2009, pp. 205–218.

[17] A. Cruz, Farewell Facebook, and Good Riddance, http://abcnews.go.com/ABC_Univision/quitting-facebook/story?id=18668978#.UYVrQ7W854I.

[18] N. Eagle, A. S. Pentland, CRAWDAD trace set mit/reality/blueaware (v. 2005-07-01), Download from http://crawdad.cs.dartmouth.edu/mit/reality/blueaware (Jul. 2005).

ACCEPTED MANUSCRIPT

ACCEPTED MANUSCRIPT

[19] J. Bao, Y. Zheng, M. F. Mokbel, Location-based and preference-aware recommendation using sparse geo-social networking data, in: ACM SIGSPATIAL GIS, 2012, pp. 199–208.

[20] P. Hornyack, S. Han, J. Jung, S. Schechter, D. Wetherall, These aren't the droids you're looking for: retrofitting android to protect data from imperious applications, in: Proceedings of the ACM conference on Computer and communications security, ACM, 2011.

[21] S. Thurm, Y. I. Kane, Your Apps Are Watching You, http://online.wsj.com/article/SB10001424052748704694004576020083703574602.html.

[22] D. Knoke, S. Yang, Social network analysis, Vol. 154, Sage, 2008.

[23] P. Hui, E. Yoneki, S. Y. Chan, J. Crowcroft, Distributed community detection in delay tolerant networks, in: ACM MobiArch, 2007.

[24] Google.com, Configure access points with Google Location Service, http://support.google.com/maps/?hl=en.

[25] M. E. Newman, Detecting community structure in networks, EPJ B 38.

[26] M. Ongtang, S. McLaughlin, W. Enck, P. McDaniel, Semantically rich application-centric security in android, Security and Communication Networks 5 (6) (2012) 658–673.

[27] M. Ongtang, K. Butler, P. McDaniel, Porscha: Policy oriented secure content handling in android, in: Proceedings of the 26th Annual Computer Security Applications Conference, 2010.

[28] B. Rashidi, C. Fung, E. Bertino, Android resource usage risk assessment using hidden markov model and online learning, Computers & Security 65 (2017) 90–107.

[29] B. Liu, M. S. Andersen, F. Schaub, H. Almuhimedi, S. A. Zhang, N. Sadeh, Y. Agarwal, A. Acquisti, Follow my recommendations: A personalized privacy assistant for mobile app permissions, in: Symposium on Usable Privacy and Security, 2016.

[30] B. Rashidi, C. Fung, A. Nguyen, T. Vu, E. Bertino, Android user privacy preserving through crowdsourcing, IEEE Transactions on Information Forensics and Security 13 (3) (2018) 773–787.

[31] M. Egele, C. Kruegel, E. Kirda, G. Vigna, Pios: Detecting privacy leaks in ios applications, in: NDSS, 2011.

ACCEPTED MANUSCRIPT

ACCEPTED MANUSCRIPT

[32] E. Chin, A. P. Felt, K. Greenwood, D. Wagner, Analyzing inter-application communication in android, in: Proceedings of the 9th international conference on Mobile systems, applications, and services, 2011, pp. 239–252.

[33] L. Lu, Z. Li, Z. Wu, W. Lee, G. Jiang, Chex: statically vetting android apps for component hijacking vulnerabilities, in: ACM CCS, 2012.

[34] W. Enck, D. Octeau, P. McDaniel, S. Chaudhuri, A study of android application security, in: USENIX security, 2011.

[35] V. F. Taylor, A. R. Beresford, I. Martinovic, Intra-library collusion: A potential privacy nightmare on smartphones, arXiv preprint arXiv:1708.03520.

[36] J. Pennekamp, M. Henze, K. Wehrle, A survey on the evolution of privacy enforcement on smartphones and the road ahead, Pervasive and Mobile Computing.

[37] S. Arzt, S. Rasthofer, C. Fritz, E. Bodden, A. Bartel, J. Klein, Y. Le Traon, D. Octeau, P. McDaniel, Flowdroid: Precise context, flow, field, object-sensitive and lifecycle-aware taint analysis for android apps, Acm Sigplan Notices 49 (6) (2014) 259–269.

[38] M. I. Gordon, D. Kim, J. H. Perkins, L. Gilham, N. Nguyen, M. C. Rinard, Information flow analysis of android applications in droidsafe., in: NDSS, Vol. 15, 2015, p. 110.

[39] Y. He, B. Hu, Z. Han, Dynamic privacy leakage analysis of android third-party libraries, in: Data Intelligence and Security (ICDIS), 2018 1st International Conference on, IEEE, 2018, pp. 275–280.

[40] H. Chen, H.-f. Leung, B. Han, J. Su, Automatic privacy leakage detection for massive android apps via a novel hybrid approach, in: Communications (ICC), 2017 IEEE International Conference on, IEEE, 2017, pp. 1–7.

[41] P. B. Brandtzaeg, A. Pultier, G. M. Moen, Losing control to data-hungry apps: A mixed-methods approach to mobile app privacy, Social Science Computer Review (2018) 0894439318777706.

[42] S. Bugiel, L. Davi, A. Dmitrienko, T. Fischer, A.-R. Sadeghi, Xmandroid: A new android evolution to mitigate privilege escalation attacks, Technische Universität Darmstadt, Technical Report TR-2011-04.

ACCEPTED MANUSCRIPT