You Are Where You APP: An Assessment on Location Privacy of Social APPs

Fanghua Zhao*, Linan Gao*, Zeyu Wang†, Yang Zhang‡ and Shanqing Guo*
*School of Software
Shandong University, Jinan, China
†School of Computer Science and Technology
Shandong University, Qingdao, China
‡CISPA
Saarland University, Saarland, Germany

***Abstract*—The development of positioning technologies has digitalized people's mobility traces for the first time in history. With GPS sensors equipped with mobile devices, people can share their positions by allowing APPs to access their location. The large amount of mobility data can help to build appealing applications, e.g., location recommendation. Meanwhile, location privacy has become a major concern.**

In this paper, we design a general system to assess whether an APP is vulnerable to location inference attacks. We utilize a series of automatic testing mechanisms, including UI match and API analysis to extract the location information (distance) an APP provides. According to different characteristics of these Apps, we classify them into two categories, corresponding to two kinds of attacks, namely attack with distance limitation (AWDL) and attack without distance limitation (AWODL). By evaluating 800 APPs, we found that 24.7% of them are vulnerable to the AWDL attack while 11.0% to AWODL attack. Moreover, some APPs even allow us to modify the parameters in Http requests which largely increases the scope of the attacks. In addition, 5 APPs directly expose the exact geo-coordinates of the potential victims.

***Keywords*-Android APPs, Location Privacy, Automatic Testing, Relative Distance, Trilateration**

I. Introduction

The fast development of ICT technologies has digitalized people's mobility traces for the first time in human history. Nowadays, with GPS-equipped mobile devices, such as smart phones and tablets, users can directly share their locations through various social network platforms such as Facebook, Twitter, and Instagram. Meanwhile, many APPs resided in a mobile device ask the users to grant them the access to their location data. Multiple parties could benefit from the large-scale mobility data: industry can use the data to build appealing applications, such as location recommendation systems; governments can use the data to improve traffic condition and reduce air pollution; meanwhile, academia can use the data to gain a deeper understanding of many fundamental questions in the society, such as epidemiology.

While bringing a lot of benefits, location data also raise the severe threat to people's privacy. In [1], the authors have pointed out that location is the most sensitive data being collected from each individual. Several studies show that knowing the locations users have visited can leak their attributes [2]–[5], and social relations [6]–[8]. In addition, being able to infer/track a user's location allows an adversary to stalk the user [9]. In particular, the privacy threat is severe for users of certain mobile APPs whose functionality heavily relies on location information, such as Tinder, Skout, and Whisper.

To mitigate the privacy threat, many location-based mobile APPs have taken countermeasures. The most common approach is only displaying the distance between two users, instead of showing their exact locations (geo-coordinates). For instance, a Whisper user can only know how far away a certain user is to her. However, the authors of [9] have shown that by simply modifying Whisper's API for location spoofing, a user's location can be inferred through triangulation. Other works in this direction include [8], [10], [11]. However, all these previous works have studied the location vulnerability of only one or a few APPs. To fully assess the location privacy threat, a large-scale study on most of the popular APPs is necessary, which, to our surprise, is still missing.

We crawled the top 800 social APPs from Google Play (500) and Wandoujia (300) and 109 of them passed the automatic testing and was roughly identified having location clues (i.e., "Distance") by triggering all these APPs' activities and monitoring them on a customized Android system [12]. 24.7% of these APPs are vulnerable to the AWDL attack while 11.0% of the them can leak users' locations with the AWODL attack. Moreover, some APPs even allow users to modify the original Apps to send crafted requests, which means the database could be disclosed by web API misuse. In addition, we identified 5 APPs which directly expose the exact geo-coordinates of the potential victims.

We further perform simulations on inferring users' locations. We have conducted our experiments in two cities, New York and Beijing. The evaluation shows that with only three location queries (based on trilateration), we are able to track more than 90% users.

We conduct a study on popular social APPs with respect to the location privacy threat. It turns out that various measures in the transmission and protection of location data are taken and a lot of them seems effective but actually invalid. For example, the relative distance is designed to protect the victims' geographic coordinates from being positioned by attackers. The server should send users' relative distances to clients when requested. However, some APPs seem to provide relative distances on user interface but actually give the latitudes and longitudes of other users to attackers and let clients do the math.

In summary, this paper makes the following contributions :

- We propose a series of testing mechanisms, including UI match and API analysis to automatically evaluate a certain social APP on location privacy leakage.
- We discover an effective way to filter the location-based APPs by parsing the APK files and intercepting Http packages. There are 109 social APPs that successfully pass the automatic testing. We discover that 24.7% of these APPs are vulnerable to the AWDL attack while 11.0% of the APPs can leak users' locations with the AWODL attack. We further show that some APPs' API design allows us to perform even stronger attacks.
- We perform simulations to demonstrate the feasibility of our location inference attacks. Experiments was conducted mainly in Beijing and New York on three APPs, Feeling, SKOUT and Blued, which turns out that three location queries, based on trilateration algorithm, can track more than 90% users.

Roadmap.The remainder of this paper is structured as follows: Section II elaborates our motivation and provides an example of the social APPs which leak users' location privacy. By then, Section III details an overview of our assessment system and proposes two attack models. The default settings and overall results of our experiments are presented in Section IV. And we also propose three real-world simulation of different APPs on different regions in this section. Section V introduces the previous works related to us on privacy protection of LBS services. Moreover, a discussion of the challenges, insights, and future works is manifested in Section VI. Finally, we conclude our work in Section VII.

II. Motivation and an Example

A. Motivation

The security of a portable device is not up to the best software installed on it, but the worst one. The majority of mobility researches focused on popular APPs like Facebook, Twitter, WeChat and so on. These APPs may do better in location protection and possess a large number of users. Nonetheless, location-based APPs are universal while a considerable number of them are deficient in privacy protection mechanisms. At this point, we propose an assessment on location privacy of mobile social APPs and develop a system to detect each specific APP.

B. An Example-SKOUT

To illustrate the problem clearly, we employ SKOUT, a typical location-based global social application which has got attention of several researchers in recent years [13], [14], as an instance. The number of SKOUT installations (50-100 million on Google Play) is not as overwhelming as Facebook. However, we have no reason to doubt that hunderds of APPs like SKOUT are capable of covering a considerable scale of users.

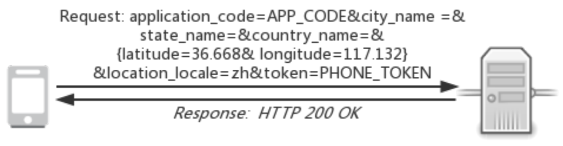

Request: application_code=APP_CODE&city_name =&
state_name=&country_name=&
{latitude=36.668& longitude=117.132}
&location_locale=zh&token=PHONE_TOKEN
Response: HTTP 200 OK

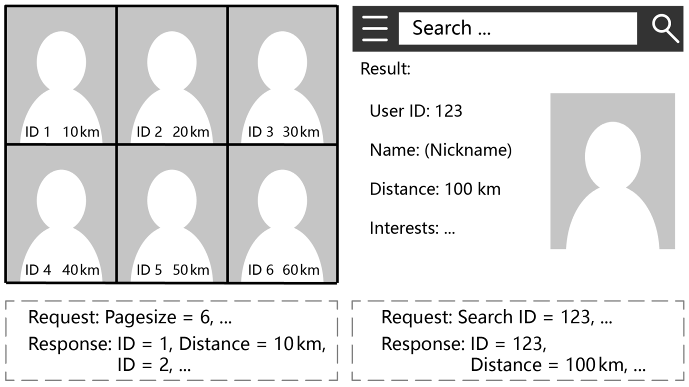
Fig. 1: Http request & response

Like most dating APPs, SKOUT provides multiple ways to help users connect with others all over the world. In general, once a user logs in by ID and correct password, the server would issue a certain **token** for the client as a part of signer authentication. If GPS is permitted to be available to this app, client will automatically send an Http request with location information to the server to specify the location of the current user and get access to server by the **token** as Figure 1. The period of validity of the token varies from an APP to another and the period of SKOUT is long enough to actualize replay attack. We modified latitude and longitude parameters values (e.g., latitude=110.908 & longitude=98.770) and resent this modified request successfully, which makes it possible to relocate a user anywhere an attacker needs.

There are two approaches to find new guys who are available to be friends, one is freshing the people nearby as Figure 2 (a), the other is searching for a sepecific ID as Figure 2 (b). Figure 2 (a) is an abstract from the SKOUT interface showing nearby users, there are 6 users loaded to the client one time and their basic information fetched from the server are shown on UI. The "Distance" parameter included in Http response which can also be interrupted demonstrates the **relative distance**. As a protection technique, server provides the distance between the user and others instead of exposing the geographic coordinate of them. Those relative distances can be used to locate victims in trilateration algorithm [15]. In trilateration locating algorithm, if we get three relative distances of one victim to three other users, we can perform location attack. In particular, SKOUT also provides the function to search any user by ID, which refers attackers can obtain detailed information by user ID (e.g.,90644621) as Figure 2 (b) with relative distances and cast the trilateration on it to trace the location of this user.

We use Burp Suite as a proxy server to capture requests and responses between the client and the server. As can be noticed, the location information of SKOUT users is reflected in Http requests and responses as follows.

- **No encryption in Http requests and responses**.
  All the location information contained in network traffic is unencrypted during transmission. Once network traffic is intercepted, all users' data would be exposed.
- **Weak protection for location Infomation**.



ID 1 10km
ID 2 20km
ID 3 30km
ID 4 40km
ID 5 50km
ID 6 60km
Search ...
Result:
User ID: 123
Name: (Nickname)
Distance: 100 km
Interests: ...
Request: Pagesize = 6, ...
Response: ID = 1, Distance = 10 km, ID = 2, ...
Request: Search ID = 123, ...
Response: ID = 123, Distance = 100 km, ...

(a) (b)

Fig. 2: Http request & response, abstract UI features of nearby users list (a) and a selected user (b)

Although the relative distance is a kind of protection for users to some extent, it could be attacked easily by trilateration algorithm.

Location-based social applications like SKOUT, which is weak in the protection of user information, are quite a lot in various APP stores. What is more, there are social APPs that directly expose the geo-coordinates of users. To that point, we propose an assessment of multiple location-based social applications and an evaluation system to detect the leakages of every new location-based social APP.

III. SYSTEM OVERVIEW

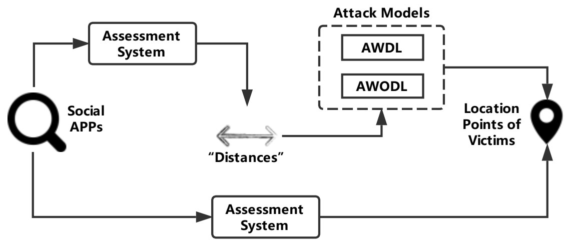


Fig. 3: The whole workflow

At present, there are a large number of social applications providing relative distance service (relative distance from a nearby user or any user searched in APP) like SKOUT. Our work attempts to find these applications and evaluate whether they can reveal users' locations. As shown in Figure 3, our ultimate goal is to obtain the location point of victims, the assessment system is mainly aimed to find location information leaking cues (like "Distance") or directly location point (like geo-coordinates) and the attack models are designed to help get location points when cues are detected.

A. Assessment System Overview

An overview of the assessment system is presented in Figure 5. There are three key components: (1) Keywords Search, a static decompile analysis process to preliminarily determine whether those APPs have location-related functionalities. (2) UI Automatic Matching, a dynamic analysis process to substantially confirm location-related functionalities according to UI features we defined in the following. (3) Automatic Analysis of APIs. Collecting all the APIs invoked by location-related functionalities to analyse and utilize API parameters' values to perform attacks.

1) Keywords Search: Many applications require access to the users' location when installed, but do not necessarily have the function to use it. By observation and testing, we found social applications with location-related functionalities always have similar features on the user interface (UI).

The programmers prefer to avoiding hard coding (embedding an input or configuration data directly into the source code of a program or other executable object, or fixed formatting of the data) in the development practice of Android projects. Usually all the strings (maybe in different languages) that appeared on the user interface (e.g.login) would be replaced by standard string names to cater to and promote international standards. There is normally a file namely $string.xml$ in the res directory of each Android Package (Apk) that transfer all the string names, which would be shown on the UI, into programmable values.

The first step is to analysing the string.xml file. Any word in any language which needs to be shown on the interface for users must appear in this file. So all the location related words could be detected to be attached to which variable values. In this process, we scan all the strings used by an application to preliminarily estimate whether it has location-related functions. Through static analysis, decompiling APKs with the help of apktool and dex2jar, we can get all resource files of APKs, where the $string.xml$ points out all the global strings. We decompile the given apk, get the file string.xml and scan all the contents of labels (e.g., "location" in $< stringname = "contents\_location" > location < /string >$) to find out if keywords including "location", "nearby", "distance"and corresponding Chinese characters appear. This process is allowed to search any strings in any languages. We selected four words to filter the location-related APPs based on empirical knowledge. The result was quite delightful: nearly half of the APPs were detected to be location-related.

2) UI Automatic Matching: Keywords search can preliminarily screen out applications with location-related functions by analysing the static code. But plenty of the functionality are available only if the APP gets access to be online. UI automatic matching will trigger all these functions to substantially confirm them and get the interfaces these functions call. Based on static analysis of APKs, this process involves dynamic automatic testing to install and run applications and trigger the location-related functions. Firstly, the UI features are as follows.

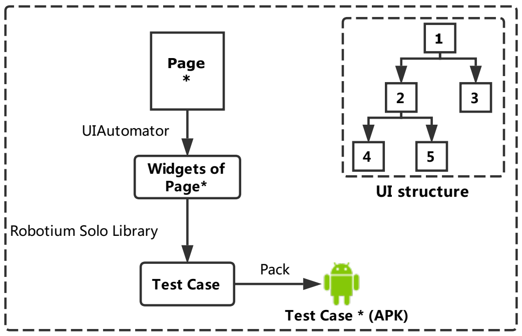


Fig. 4: Automatic testing

In social applications, (i) F_1 refers to the UI like Figure 2 (a), with the functionality to display nearby users and relative distances from them. (ii) F_2 refers to the search bar UI in Figure 2 (b), with the functionality to search any users. (iii) F_3 refers to UI like the Textview widget (e.g., $Distance$: $100km$) in Figure 2 (b), with the functionality to show other users' location-related information.

- Feature-1 (F_1): Multiple Textview widgets, with text attributes " * km"," * m" or "* mi";
- Feature-2 (F_2): EditText widget as a Search bar, with the hint attribute "ID" or "username" and other similar tips;
- Feature-3 (F_3): One Textview widget, with the text attributes "* km" or "* m" or "* mi";

The UI automatic matching needs the following tools: UIAutomator, which we use to obtain all the widgets of UI; Junit regression testing framework, which we use to generate automatic test cases for every UI; Solo library of Robotium, automatically triggering widgets in test cases generated by Junit regression testing.

To automatically traverse the UI structure (all pages can be triggered) of application, we follow the principle of breadth priority (the test order is $1 \to 2 \to 3 \to 4 \to 5$ in UI structure of Figure 4). Our system aims to cover all the UIs to find the UI features (F_k, k=1,2,3) we defined.

For each page, we designed a test case in the form of an APK to dynamically trigger all the widgets and our system will automatically install and run the APK on the Android device. As shown in Figure 4, the test case (APK) embodies all widgets of the tested page obtained by UIAutomator and the use of it is to trigger all those widgets drawing support from solo library of Robotium. In detail, all the widgets obtained by UIAutomator are stored in a queue, once one widget is triggered, it will be cleared. The test case will continue to trigger all the widgets until the queue is empty. If any widgets trigger a new page, we preserve it for the next test case generation. In the meanwhile, we maintain a path to save the test case triggering process from the start page to any other page.

Algorithm 1: UI Automatic Matching Algorithm

```
Input: Set $S_{feature}$ = $\Phi$, $F_k(k = 1, 2, 3)$: 3 kinds of
       feature.
Output: Features Set $S_{feature}$ of Current APP
for each APP do
    Install its APK via Android Debug Bridge (ADB);
    Set $G$ = the initial page;
    for page* in $G$ do
        Set $T$ = all the UI widgets of page*;
        for each $i \in T$ do
            if $i$ matches $F_k$ (k=1,2,3) then
                $S_{feature} \leftarrow S_{feature} \cup \{F_k\}$;
            end
            if $i$ triggers a page $P_j \notin G$ then
                $G \leftarrow G \cup \{P_j\}$;
            end
        end
    end
end
```

UI automatic matching algorithm (shown in algorithm 1) includes UI features (F_k, k=1,2,3) matching and automatic testing. During automatic testing, where all widgets are retrieved and triggered, we discern the characteristics and attributes of those widgets, and take features we defined to match. The algorithm input is three features F_k (k=1,2,3), the output is matched features $S_{feature}$ set.

Definition:

- Category-I: Output $S_{feature}$ = F_1.
- Category-II: Output $S_{feature}$ = F_2, F_3.
- Category-I -II: Output $S_{feature}$ = F_1, F_2, F_3.

We roughly divide APPs to three categories in this section, If $S_{feature}$ returns F_1, the tested APP belongs to Category-I, it is very likely that it supplies relative distances from nearby users. If $S_{feature}$ returns F_2, F_3, the tested APP belongs to Category-II and it is that it can supply relative distance from arbitrary user searched. If $S_{feature}$ returns F_1, F_2, F_3, the tested APP belongs to Category-I -II and has the above two functionalities.

3) Automatic Analysis of APIs: In SKOUT, utilizing parameters (e.g., Distance, ID) that intercepted in HTTP requests and responses make it possible to perform trilateration attack. The ultimate goal of the assessment system is also to discover useful interface information and implement location attack. So, as UI automatic matching has triggered the location-related functionalities, we can make sure which APIs are invoked. Therefore, this process includes collecting APIs and the automatic analysis of them.

At the same time, we intercept all APIs of the tested Android phone with the help of the proxy in Burp Suite. If the $S_{feature}$ is not empty for an APP, we export the APIs according to their host name and separately store for different APPs and also as the input of API automatic analysis algorithm (shown in Algorithm 2).

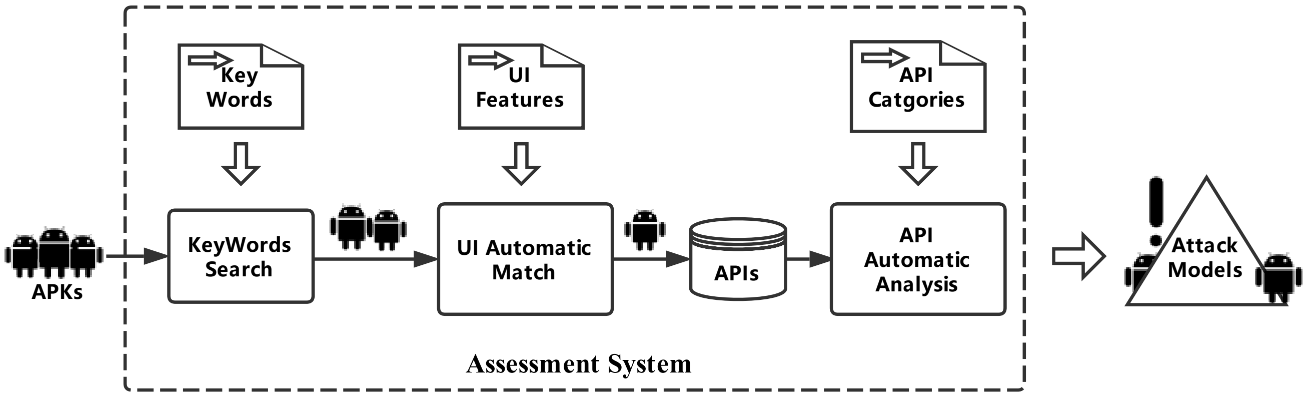


Fig. 5: A high-level overview of assessment system

Similarly, the analysis and utilization of the interface can not be separated from several types of typical APIs. We define three typical APIs that can be used to perform the attack, the detail of http requests and responses forms are shown in Figure 6. Category-I always has API-1 and API-2, Category-II always has API-1 and API-3, Category-I-II has API-1, API-2, and API-3.

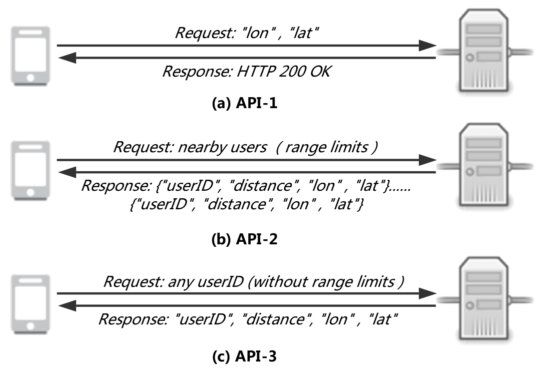

Request: "lon", "lat"
Response: HTTP 200 OK
(a) API-1
Request: nearby users (range limits)
Response: {"userID", "distance", "lon", "lat"}...... {"userID", "distance", "lon", "lat"}
(b) API-2
Request: any userID (without range limits)
Response: "userID", "distance", "lon", "lat"
(c) API-3

Fig. 6: Typical API categories: API-1, API-2, API-3

- API-1: API that is used to submit the latitude and longitude of the client.
- API-2: API that is used to get location information (e.g, distance) of a list of potential victims.
- API-3: API that is used to get location information (e.g, distance) of a certain potential victim searched by ID or user name.

B. Attack Models AWDL and AWODL

In the whole work flow, the attack models AWDL and AWODL are shown in Figure 7.

Definition:

Algorithm 2: API Automatic Analysis Algorithm

```
Input: Set $S_{API}$ = $\Phi$, Set $\Omega$ = {Requests}, Set $\Lambda$ = {Responses}.
Output: API Categories Set $S_{API}$ of Current APP
for each $i \in \Omega$ do
    if $i$ includes GPS coordinates then
        $S_{API} \leftarrow S_{API} \cup \{API-1\}$;
    end
end
for each $i \in \Lambda$ do
    if $i$ includes a list of location information then
        $S_{API} \leftarrow S_{API} \cup \{API-2\}$;
    end
    else if $i$ returns a location through searching ID/uesrname then
        $S_{API} \leftarrow S_{API} \cup \{API-3\}$;
    end
end
```

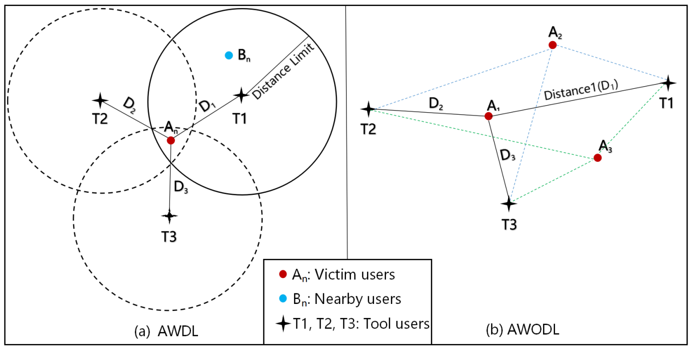


Fig. 7: Attack models AWDL & AWODL

- AWDL: Attack with distance limit (AWDL), is designed for the APPs offering relative distances of nearby users with a range limitation.
- AWODL: Attack without distance limit (AWODL), is designed for the APPs offering relative distances of a selected and searched user (which means almost any user) without any distance limitation.

In our system, we propose two attack models as the guide of assessment for the testing result, Category-I or Category-II. The attack model AWDL refers to the APPs offering nearby users lists with a range limitation mainly for Category-I. The attack model AWODL refers to the APPs offering relative distances of a selected and searched user without any distance limitation, mainly for Category-II and both AWDL and AWODL models for Category-I -II. These two attack models are both based on trilateration algorithm.

For the AWDL cases (as shown in Figure 7 (a)), T_1, T_2, T_3 are three real accounts we registered as tool users and their three tokens are used to replay web APIs to get location clues (relative distances). Victim users $(A_n; n \geq 0)$ are in the intersection of nearby users $(B_n; n \geq 0)$ of three tool users with distance limit. By replaying API-1, T_1, T_2, T_3 can be relocated to three position points and then get http response with B_n and relative distance values in it by replaying API-2 according to the position points. Set A_n is the intersection of three B_n sets. So for all the victims in A_n, three distances (D_1, D_2, D_3) can be obtained and as input to trilateration algorithm. In the attack model, distance limit restricts the attack range, tool users can't get relative distances to all users because of the limitation on nearby people size or regional scope limitation which means not all the users are victims. But by API-1, we can relocated T_1, T_2, T_3 to anywhere and perform attacks on users in any area.

For the AWODL cases (as shown in Figure 7 (b)), any users of an application are victim users. If any userID or other key words of one user is offered, replaying API-3 can find this unique user. Since there is no distance limit, by API-3, relative distance between the tool user and the searched user is included in the response. As similar to AWDL, by replaying API-1, T_1, T_2, T_3 are located to three position points. For any given victim $(A_i; i = 1, 2...n)$, API-3 can help get three relative distances (D_1, D_2, D_3) and D_1, D_2, D_3 are the input of trilateration algorithm to get longitude and latitude of the victim. Every one using the tested APP can be searched by unique ID or other key words (user name or telephone number). Hence, in AWODL, it is obviously probable to infringe on all users of an application to the maximum extent.

IV. Experiment

We introduce the details of our experiments including dataset, default settings, and overall results. A three-stage experiment is performed to evaluate our technique. On the first stage, we employ all the APPs we collected to our system and evaluate their safety level based on the previous definitions (AWDL and AWODL). Hereafter, we formulate a simulation on three typical APPs (SKOUT, Feeling, and Blued) aiming to assess different attack models: 1) AWDL, 2) AWODL, and 3) both AWDL and AWODL respectively. Last but not least, we propose a summarization for location tracking attack aimed to point out the effectiveness of our system.

A. Dataset collection and experiment environment

We crawled top 500 APPs from google play and top 300 APPs from Wandoujia in the social category. Removing the duplicate ones, we tested 737 APPs on our assessment system totally. By the process of location-based filtering, we got 365 apps in our dataset.

All of these apps were installed automatically to a LG Nexus 5 smartphone with Android 7.0 OS and the smartphone is connected to a Win10 PC running atop an Intel (R) Core (TM) i5-3470 3.20GHz CPU with 16.0GB RAM by Android Debug Bridge (ADB) commands. For the assessment process, the static process keyword search is executed just on Win10 PC without real Android phone, the static process of UI automatic matching is also executed on Win10 PC but dynamic process of UI automatic matching need the smartphone to step in. In automatic analysis of APIs, we set the proxy in BurpSuite to collect all the Http requests and responses of LG Nexus 5 smartphone and the data is stored on the PC. Our location inference attacks run on Genymotion Android Emulator installed on Win10 PC and we start three Android emulators at the same time to maintain all the tool-users online.

B. The overall result

The keywords search process running in a thread pool with five threads spent 4 minutes to filter 365 location-related social APPs from 737 in total with 372 not location-related. 109 of them were successfully installed and tested in the dynamic process of UI automatic matching. In these 109 APPs, 29 (24.7%) shown in successfully matched in UI matching and as shown in Figure 8, 27 of them match Feature 1 (category-I) when 12 (11.0%) of them match Feature 2 & Feature 3 (category-II) because 10 of them match all three features and belong to category-I & category-II in the meantime.

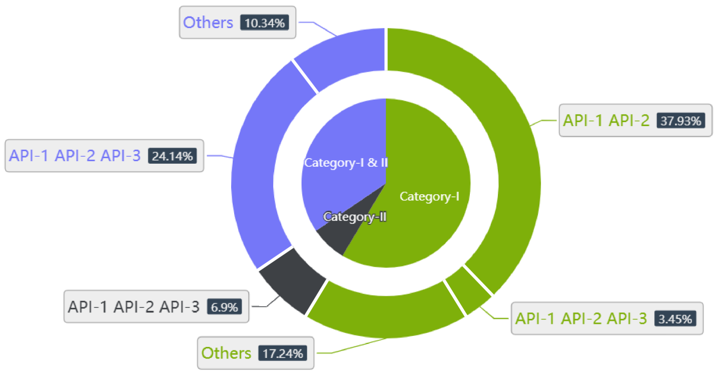


Fig. 8: UI and API categories for 29 high risk APPs

The Http requests and responses were extracted to XML format files from Burp suite. We extacted 29 XML files of those 29 APPs. As shown in Figure 8, our automatic analysis of APIs successfully detect 21 of them containing which categories of APIs we defined.

C. *Evaluation of detection result*

To evaluate the detection result of our system, in those 19 APPs detected fit for AWDL attack, we sampled 10 APPs to perform AWDL attack and 6 (cn.yourole, com.feeling, nnmcw.example.mozu, com.skout.android, com.soft.blued, com.roogooapp.im) of their users could be successfully located. Reasons of failure mainly includes : 1. the content of requests and responses are encoded (com.wodi.who), 2. the content is encrypted (com.android.lesdo, com.immomo.momo.apk), 3. the API-1 of them can't be replayed to locate tool-users (com.thel).

In the above 6 APPs in AWDL attack, 4 (nnmcw.example.mozu, com.skout.android, com.soft.blued, com.roogooapp.im) of them are also fit for AWODL attack. So we perform AWODL to them to assess the detection result of our system. Our evaluation result shows that potential victims in all those 4 APPs can be located.

D. *Simulations of location inference attacks,*

We perform simulations to demonstrate the feasibility of our location inference attacks, experiments performed in Beijing and New York on three APPs Feeling, SKouT, and Blued because of their respect characteristics.

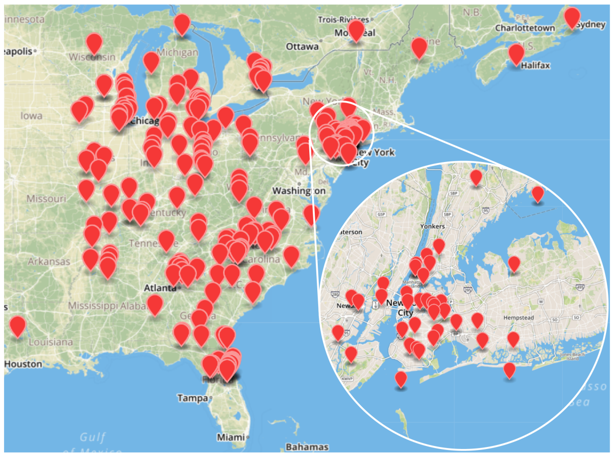


Fig. 9: Instance of AWODL attack in New York with SKOUT

1) Feeling APP in AWDL: In our assessment system, Feeling APP is detected matching F_1 and F_2 as well have API-1 & API-2, which is actually fits for AWDL. In each response, Feeling server gives back a list of nearby users and the parameter pageSize (pageSize = 20 in default) in API-2 is the limitation of the number of nearby people in each corresponding request.

We performed this simulation attack in Beijing based on Feeling's users' distribution.

As the attack result shows, we can obtain 19 attackable victims and locate 16 of them in trilateration, shown in Figure 10 (a).

For further discovery, we even found that the interface parameters can be modified. PageSize in API-2 can be reset to expand AWDL attack coverage. As shown in Figure 10, we gradually expanded the parameter pageSize from 20 to 1000, the discovery of parameter's change attack breaks the limitation to some extent. From Figure 10 (a) to Figure 10 (c), there is evident increasing of victims in the chosen area of Beijing. But when we continued to expand the parameter pageSize of the request, there is no obvious increase in the chosen area, which means the result in Figure 10 (c) are revealing all the users in the chosen area to some degree. Through continuous relocating three tool-users and repeating the above experiment, we are capable of locating all the users in any area of one APP which fits AWDL.

2) SKOUT APP in AWODL: In our assessment system, SKOUT is detected matching F_1, F_2, F_3 and have API-1 & API-2, means it fits both AWDL and AWODL. It offers an interested-people list which takes relative distance into consideration.

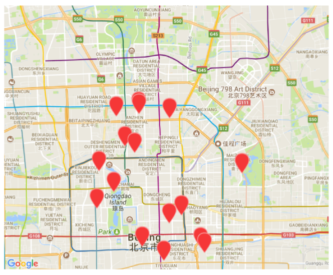
By modifying the pagesize to request more interested-people like the technique we utilized in AWDL, we broke the limitation and obtain a list of the interested-people information (especially, the IDs). Then we automatically extract the users' ID as the potential victims of AWODL. We continuously change the locations of those three tool-users (in or around New York) until the potential victim is located. The results show that SKOUT does have users in New York, and because SKOUT interested-people list takes relative distance as an aspect of "interesting", most of ID located in the east of America. we can return latitudes and longitudes for each of them and visualized them using geojson.io (in Figure 9).

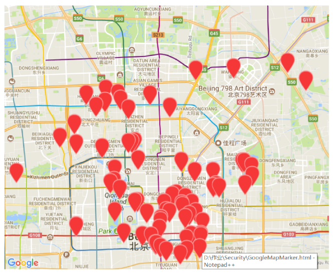
3) Blued APP in AWDL and AWODL: Blued, which is popular all over the world, is exactly a nearby-people APP and also provides the function to search any users. The parameter limit in API-2 is the limitation (default=60) of nearby people number in one request. Besides the same finding in those attacks performed in Feeling and SKOUT, we have some extra discoveries. For AWDL, Blued limits the number of nearby friends each response provided. Their initialized limitation is 60. By manually continuous resetting, we found that the biggest limitation can be accepted by Blued is 72. So Blued suits AWDL not very well. Nonetheless, the Blued unique IDs to identify users are continuous of uninterrupted integers (if exist 2027, then exists 2026, 2025......). Based on this point we can traverse all the ID in Blued database and locate them to GPS coordinates. In simulation experiment, we located 100 ID on a global scale of AWODL and the results are shown in Figure 11.

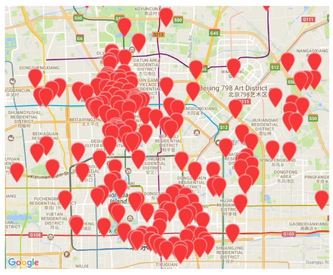
4) High risk APPs: Besides AWDL and AWODL attacks, by the analyse of APIs, we even find some social APPs take no steps on user location privacy protection and directly return the geo-coordinates (in the form $lat =$ "$23.****45$", $lon =$ "$123.****45$"), including com.redwolfama.peonylespark,

Package name	**Downloads(From GooglePlay or Wandoujia)**	**UI type**	**API type**
com.skout.android	50000000	category-I category-II	API-1 API-2 API-3
com.feeling	23000	category-I	API-1 API-2
com.soft.blued	21958000	category-I category-II	API-1 API-2 API-3
com.lchr.diaoyu	22772000	category-I	API-1 API-2
com.roogooapp.im	592000	category-I category-II	API-1 API-2 API-3
com.redwolfama.peonylespark	333000	category-I category-II	API-1 API-2 API-3
com.wodi.who	2015000	category-I	API-1 API-2
com.thel	725000	category-I category-II	API-1 API-2 API-3
com.hzsj.dsjy	12461000	category-II	API-1 API-2 API-3
com.ya.sq.yuanfen	1425000	category-I	API-1 API-2
com.yeyuetongcheng.main	1123000	category-I	API-1 API-2
com.yuedan	5738000	category-I	API-1 API-2
io.maper.android	50000	category-I	API-1 API-2

TABLE I: The detection classification of partial APPs


Page Size = 20
(a)


Page Size = 100
(b)


Page Size = 400
(c)

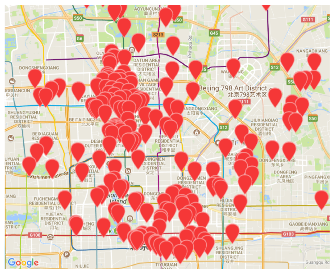

Page Size = 1000
(d)

Fig. 10: Instance of AWDL attack in Beijing with Feeling

nnmcw.example.mozu, com.esky.echat.apk, io.maper.android, com.hoolai.moca.apk. Those 5 APPs are high risk social APPs since they directly expose the exact latitudes and longitudes of victims. But the total user volume of them is between 8,113,494 and 8,163,494 (according to Google Play and Wandoujia), security threats caused by them should be concerned.

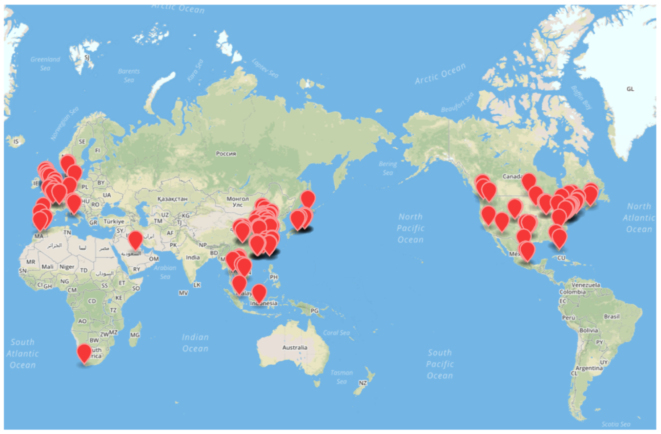


Fig. 11: Simulation of a global scale attack of Blued

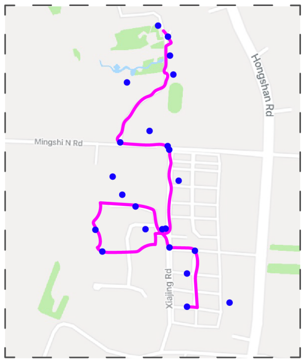
E. Tracking real-world users

Furthermore, we recruited 10 users for conducting a real-world attack to better evaluate our system. All the participants engaged in some sort of computer-oriented professions, such as programmer or cyber security engineer, and has been using the above three APPs for a period. These users were asked to wear a smart watch for recording their position precisely and backstage operate the procedure of three APPs. During this attack, participants provided us their user ID on these APPs, and we acquired their locations through searching ID or nearby people continuously by using our system. At the end, we draw our attained position connections and compared the similarity between the attack result with users' motion tracks recorded by smart watch. This study lasted for about half a day.

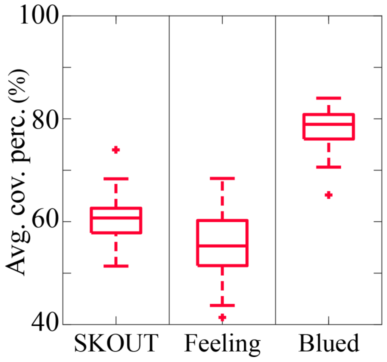
Figure 12 (a) presents an instance of the comparison results of tracking points by using Blued APP and the user's motion tracks in a period. The result obviously reveals that a quite amount of tracking positions located in the curve of the user's motion tracks, i.e., most blue points located nearby the pink curve.

For better comparison, we define the *average coverage distance* as $P_{cov}/P_{all} * 100\%$, where P_{cov} is the number of attacked positions that located nearby the motion curves (i.e., 100% for 0 meter and 0% for 100 meter), and P_{all} is the number of all tracked positions. Figure 12 (b) illustrates the boxplot of 10 users' average coverage percentage of three APPs, while it's worth to note that a number of attacked positions can be well-formed fit with motion tracks similarly.

Hence, this real-world attack indicates that we can successfully track users' location information on the vulnerable APPs by using our attack models. It's also worth to note that, during this study, most users walked very slow and did not pick some intense exercise such as running or taking taxi which may largely influence the accuracy.



(a)



(b)

Fig. 12: (a) shows an instance of real-world location comparison between our attack results (blue points) with user's motion tracks (pink curve) by using Blued APP. (b) indicates the boxplot of 10 users' average coverage distance of SKOUT, Feeling, and Blued.

F. Summary

Finally, we describe the highlight of our approach that revealed by this experiment in summary.

AWDL: everywhere. The AWDL attack model can attack and track locations by computing relative distances for nearby users. This model can reveal user privacy everywhere around the world. That is, we can attack any position and get the nearby user's information.

AWODL: everyone. The AWDL attack model can attack and track locations by computing relative distances for searched users. This model can attack everyone that we are interested in through select their nicknames or user IDs.

Both of the attack models can perform well with high risk APPs automated selected.

V. Related Work

In this section, we discuss the previous works about the location privacy protection of LBS services.

Location privacy protection on location-based services is a long-standing topic which is crucial to other security problems and has received a lot of attentions in the past years for the rapid rise and widespread concern of LBS applications [16]–[18].

During the past years, there are many investigations that aimed to emphasizing the importance and proposing the techniques for protecting privacy in LBS services [19]–[22]. While one of the most well-known and wide-applied strategy is to employ k-anonymity metric of privacy protection [23]. A typical k-anonymity based privacy protection mechanism is proposed by Gruteser et al. [24] by obscuring user's location to other *k-1* dummy locations. A mondrian multidimensional approach is proposed by LeFevre et al. [25]. Hereafter, Gedik et al. [26] presented a technique whose protection model permitted users to modulate the level of anonymity. Recently, a dummy location selection algorithm is introduced to improve the privacy level in terms of entropy. In addition, l-diversity [27] and t-closeness [28] are proposed because k-anonymity always cannot provide protection against probabilistic attack.

There are also some approaches proposed to protect the user's position privacy when they are enjoying the LBS service, such as obfuscation-based techniques [29], [30], mix zones related approaches [31], [32], and some other methods through adding dummy requests issued by fake location and indistinguishable from real requests [33]. There also are some other works tried to achieve the trade-off between the location privacy and utility of LBS networks [34]–[37].

However, all these protection approaches are trying to obscure the user's location or the identity of the user ID and aiming to increase the attack difficulty of the location attacker. But in location-based social applications, the users' demand is to share the nick name and exact location to do position check-in, search for a charming nearby person, or attain other location-based services. So the serious dependence of the functions in social applications on user location and the uncertainty of user willingness to share the position make the location privacy protection more and more difficult and important. Hence, we propose a series of assessment schemes, which are aimed to improve the awareness of location protection by users and developers and also put forward suggestions for developers and service providers.

VI. Discussion

A. Are the attackers too strong or the developers too weak?

The rise of social needs contributes to the booming of dating applications, which are the majority of location-based APPs. Those state-of-the-art company like facebook have comparatively rich experience and economic ability to design APPs with good defense against different kinds of attacks. A large number of following developers are busied in imitating successful APPs, producing APPs with similar services and putting them in the APP stores without considering the security of users' data or privacy. Through our assessment system, any user could be an attacker using a proxy server (Burp suite, Fiddler, Charles...) and a little bit mathematics without rich knowledge about cryptography. Currently, the cost of privacy is more and more concerned, developers should pay more attention to users' data protection.

B. Defendense Strategies.

From the results and evaluation of the experiment, great potential safety hazard about location privacy leak exists in large-scale social applications. Compared to APPs that preformed well in our assessment,we put forward the following suggestions for developers and service providers:

1) Https instead of Http: There is no encryption in Http, but it can be used in combination with SSL (Secure Socket Layer) to encrypt communications. After using SSL to establish a secure communication line, Http communication can be made on this line. The Http used in combination with SSL is called Https (Http Secure, Hypertext Transfer Security Protocol).

2) Reduce the period of the validity of the token: The validity of the original token request is vital to maintain access for attackers. One of the reason we conducted our experiment on SKOUT is that the SKOUT token is valid for over 12 hours.

3) Add hash value to messages: Adding the hash values to the end of corresponding messages is efficient to defend replay attack. The modification of readable messages would not pass authentication if attackers do not use the selected hash function to compute.

C. Limitations and Future Work.

First, the degree of automation of our assessment is still priamry. We manually registered all the apps that are location-based, which is really time-consuming. In the future work, we consider to use the single-sign-on (e.g., with Facebook Login [38]) to liberate labor, enhance accuracy and improve efficiency.

Second, our assessment only focuses on those social applications that offer relative distances, so our corresponding attack models are limited to two (AWDL and AWODL). In the future work, we will consider other attack models or level system to assess more social APPs. And trilateration has been proved that has drawbacks and wrongly recognizes a localizable graph as nonlocalizable in previous work [39]. But it still is an effective approach to locate victims appropriately when relative distance is used as a protective technique [40] and multiple recent researches continuously focused on GPS localization based on trilateration [41]–[43] .

Finally, location privacy leakage is a general problem in LBS APPs and not just limits to social APPs on Android. Currently, we only developed the automatic testing on Android, in the future, our scheme can be used to detect other APPs on other operating systems. Whatever, we believe our research result will alert other location-based services providers to pay more attention to protecting the users' location privacy.

VII. CONCLUSION

Users are more willing to authorize mobile APPs with their location data for more services nowadays. However, the incomplete user information preserving configurations of some APPs can largely raise a lot of privacy issues. Hence, we conducted an assessment on the location privacy threats of social APPs in this paper.

Dedicated to automatically evaluate the safety of different APPs, we propose a general privacy strategy detection framework. Our framework utilizes a series of testing techniques, especially UI matching and API analysis, to determine whether a social APP is vulnerable to reveal information in location tracking attacks. Hereafter, based on the position's format of different APPs which is relevant to their security strategies, we identify two attack models (AWDL and AWODL) and provide several available trilateration-based approaches to defense the proposed models.

Furthermore, we illustrated the overall assessment results, and three typical APPs (SKOUT, Feeling, Blued) were selected as instances for manifesting the validity. This experiment shows that our attack technique can successfully infer locations of interested users. Moreover, for some APPs with even worse privacy strategies such as Blued, indeed we can entirely simulate a global scale attack for attaining locations of all their users.

References

[1] Y.-A. D. Montjoye, C. A. Hidalgo, M. Verleysen, and V. D. Blondel, "Unique in the Crowd: The Privacy Bounds of Human Mobility," *Scientific Reports*, vol. 3, p. 1376, 2013.

[2] J. Pang and Y. Zhang, "DeepCity: A Feature Learning Framework for Mining Location Check-Ins," in *Proceedings of the 11th International Conference on Web and Social Media (ICWSM)*. The AAAI Press, 2017, pp. 652–655.

[3] M. Humbert, T. Studer, M. Grossglauser, and J.-P. Hubaux, "Nowhere to hide: Navigating around privacy in online social networks," in *European Symposium on Research in Computer Security*. Springer, 2013, pp. 682–699.

[4] J. Pang and Y. Zhang, "Location prediction: communities speak louder than friends," in *Proceedings of the 2015 ACM on Conference on Online Social Networks*. ACM, 2015, pp. 161–171.

[5] P. Golle and K. Partridge, "On the anonymity of home/work location pairs," in *Pervasive Computing*, H. Tokuda, M. Beigl, A. Friday, A. J. B. Brush, and Y. Tobe, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 390–397.

[6] S. Scellato, A. Noulas, and C. Mascolo, "Exploiting Place Features in Link Prediction on Location-based Social Networks," in *Proceedings of the 17th ACM Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 2011, pp. 1046–1054.

[7] M. Backes, M. Humbert, J. Pang, and Y. Zhang, "walk2friends: Inferring social links from mobility profiles." in *Proceedings of the 24th ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2017, pp. 1943–1957.

[8] M. Srivatsa and M. Hicks, "Deanonymizing mobility traces: Using social network as a side-channel," in *Proceedings of the 2012 ACM Conference on Computer and Communications Security*, ser. CCS '12. New York, NY, USA: ACM, 2012, pp. 628–637. [Online]. Available: http://doi.acm.org/10.1145/2382196.2382262

[9] G. Wang, B. Wang, T. Wang, A. Nika, H. Zheng, and B. Y. Zhao, "Whispers in the dark: analysis of an anonymous social network," in *Proceedings of the 2014 Conference on Internet Measurement Conference*. ACM, 2014, pp. 137–150.

[10] I. Polakis, G. Argyros, T. Petsios, S. Sivakorn, and A. D. Keromytis, "Where's wally?: Precise user discovery attacks in location proximity services," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2015, pp. 817–828.

[11] M. Xue, C. Ballard, K. Liu, C. Nemelka, Y. Wu, K. Ross, and H. Qian, "You can yak but you can't hide: Localizing anonymous social network users," in *Proceedings of the 2016 ACM on Internet Measurement Conference*. ACM, 2016, pp. 25–31.

[12] J. Chen, X. Cui, Z. Zhao, J. Liang, and S. Guo, "Toward discovering and exploiting private server-side web apis," in *Web Services (ICWS), 2016 IEEE International Conference on*. IEEE, 2016, pp. 420–427.

[13] P. A. Quiroz, "From finding the perfect love online to satellite dating and 'loving-the-one-you're near': A look at grindr, skout, plenty of fish, meet moi, zoosk and assisted serendipity," *Humanity & Society*, vol. 37, no. 2, pp. 181–185, 2013.

[14] E. Toch and I. Levi, "What can 'people-nearby' applications teach us about meeting new people?" in *UbiComp*, 2012.

[15] F. Thomas and L. Ros, "Revisiting trilateration for robot localization," *IEEE Transactions on Robotics*, vol. 21, no. 1, pp. 93–101, Feb 2005.

[16] Z. Zhu and G. Cao, "Applaus: A privacy-preserving location proof updating system for location-based services," in *INFOCOM, 2011 Proceedings IEEE*. IEEE, 2011, pp. 1889–1897.

[17] K. G. Shin, X. Ju, Z. Chen, and X. Hu, "Privacy protection for users of location-based services," *IEEE Wireless Communications*, vol. 19, no. 1, 2012.

[18] M. Han, M. Yan, J. Li, S. Ji, and Y. Li, "Neighborhood-based uncertainty generation in social networks," *Journal of Combinatorial Optimization*, vol. 28, no. 3, pp. 561–576, 2014.

[19] R. J. Bayardo and R. Agrawal, "Data privacy through optimal k-anonymization," in *Data Engineering, 2005. ICDE 2005. Proceedings. 21st International Conference on*. IEEE, 2005, pp. 217–228.

[20] B. Hoh, M. Gruteser, H. Xiong, and A. Alrabady, "Preserving privacy in gps traces via uncertainty-aware path cloaking," in *Proceedings of the 14th ACM Conference on Computer and Communications Security*, ser. CCS '07. New York, NY, USA: ACM, 2007, pp. 161–171. [Online]. Available: http://doi.acm.org/10.1145/1315245.1315266

[21] B. Niu, Q. Li, X. Zhu, G. Cao, and H. Li, "Enhancing privacy through caching in location-based services," in *Computer Communications (INFOCOM), 2015 IEEE Conference on*. IEEE, 2015, pp. 1017–1025.

[22] R. Shokri, G. Theodorakopoulos, C. Troncoso, J.-P. Hubaux, and J.-Y. Le Boudec, "Protecting location privacy: optimal strategy against localization attacks," in *Proceedings of the 2012 ACM conference on Computer and communications security*. ACM, 2012, pp. 617–627.

[23] L. Sweeney, "k-anonymity: A model for protecting privacy," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 05, pp. 557–570, 2002.

[24] M. Gruteser and D. Grunwald, "Anonymous usage of location-based services through spatial and temporal cloaking," in *Proceedings of the 1st international conference on Mobile systems, applications and services*. ACM, 2003, pp. 31–42.

[25] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan, "Mondrian multidimensional k-anonymity," in *Data Engineering, 2006. ICDE'06. Proceedings of the 22nd International Conference on*. IEEE, 2006, pp. 25–25.

[26] B. Gedik and L. Liu, "Protecting location privacy with personalized k-anonymity: Architecture and algorithms," *IEEE Transactions on Mobile Computing*, vol. 7, no. 1, pp. 1–18, 2008.

[27] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam, "l-diversity: Privacy beyond k-anonymity," in *Data Engineering, 2006. ICDE'06. Proceedings of the 22nd International Conference on*. IEEE, 2006, pp. 24–24.

[28] N. Li, T. Li, and S. Venkatasubramanian, "t-closeness: Privacy beyond k-anonymity and l-diversity," in *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*. IEEE, 2007, pp. 106–115.

[29] P. Samarati, "Protecting respondents identities in microdata release," *IEEE transactions on Knowledge and Data Engineering*, vol. 13, no. 6, pp. 1010–1027, 2001.

[30] Z. Tu, K. Zhao, F. Xu, Y. Li, L. Su, and D. Jin, "Beyond k-anonymity: protect your trajectory from semantic attack," in *Sensing, Communication, and Networking (SECON), 2017 14th Annual IEEE International Conference on*. IEEE, 2017, pp. 1–9.

[31] A. R. Beresford and F. Stajano, "Mix zones: User privacy in location-aware services," in *Pervasive Computing and Communications Workshops, 2004. Proceedings of the Second IEEE Annual Conference on*. IEEE, 2004, pp. 127–131.

[32] J. Freudiger, R. Shokri, and J.-P. Hubaux, *On the Optimal Placement of Mix Zones*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 216–234. [Online]. Available: https://doi.org/10.1007/978-3-642-03168-7_13

[33] T. Xu and Y. Cai, "Feeling-based location privacy protection for location-based services," in *Proceedings of the 16th ACM Conference on Computer and Communications Security*, ser. CCS '09. New York, NY, USA: ACM, 2009, pp. 348–357. [Online]. Available: http://doi.acm.org/10.1145/1653662.1653704

[34] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, "Geo-indistinguishability: Differential privacy for location-based systems," in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*. ACM, 2013, pp. 901–914.

[35] Y. Wang, D. Xu, X. He, C. Zhang, F. Li, and B. Xu, "L2p2: Location-aware location privacy protection for location-based services," in *INFOCOM, 2012 Proceedings IEEE*. IEEE, 2012, pp. 1996–2004.

[36] K. Fawaz and K. G. Shin, "Location privacy protection for smartphone users," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2014, pp. 239–250.

[37] Q. Xiao, J. Chen, L. Yu, H. Li, H. Zhu, M. Li, and K. Ren, "Poster: Locmask: A location privacy protection framework in android system," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2014, pp. 1526–1528.

[38] R. Wang, S. Chen, and X. Wang, "Signing me onto your accounts through facebook and google: A traffic-guided security study of commercially deployed single-sign-on web services," in *2012 IEEE Symposium on Security and Privacy*, May 2012, pp. 365–379.

[39] Z. Yang, Y. Liu, and X. Y. Li, "Beyond trilateration: On the localizability of wireless ad hoc networks," *IEEE/ACM Transactions on Networking*, vol. 18, no. 6, pp. 1806–1814, 2010.

[40] Z. Yang and Y. Liu, "Quality of trilateration: Confidence-based iterative localization," *IEEE Transactions on Parallel and Distributed Systems*, vol. 21, no. 5, pp. 631–640, May 2010.

[41] K.-M. Cheung, G. Lightsey, and C. Lee, "Accuracy/computation performance of a new trilateration scheme for gps-style localization."

[42] A. Rafina Destiarti, P. Kristalina, and A. Sudarsono, "Swot: Secure wireless object tracking with key renewal mechanism for indoor wireless sensor network."

[43] W. Liu, J. Zhang, G. Huang, G. Wang, and Z. Zhang, "The indoor localization algorithm for combination of signal strength and anti-disturbance," in *China Satellite Navigation Conference*. Springer, 2018, pp. 341–353.