Paper **Five**

Privacy-Preserving Location-Proximity for Mobile Apps

Simonas Stirbys, Omar Abu Nabah, Per Hallgren and Andrei Sabelfeld

Location Based Services (LBS) have seen alarming privacy breaches in recent years. While there has been much recent progress by the research community on developing privacy-enhancing mechanisms for LBS, their evaluation has been often focused on the privacy guarantees, while the question of whether these mechanisms can be adopted by practical LBS applications has received limited attention. This paper studies the applicability of Privacy-Preserving Location Proximity (PPLP) protocols in the setting of mobile apps. We categorize popular location social apps and analyze the trade-offs of privacy and functionality with respect to PPLP enhancements. To investigate the practical performance trade-offs, we present an in-depth case study of an Android application that implements InnerCircle, a state-of-the-art protocol for privacy-preserving location proximity. This study indicates that the performance of the privacy-preserving application for coarse-grained precision is comparable to real applications with the same feature set.

Published in the 25th Euromicro International Conference on Parallel,
Distributed and Network-based Processing
PDP 2017, St. Petersburg, Russia, March 6-8, 2017

1 Introduction

Location Based Services (LBS) have seen a tremendous growth in recent years. A single resource lists over 2900 services at the time of writing [1]. The growth is boosted by the increasing spread of mobile devices, as Internet usage by mobile devices has come to dominate over desktop both by the number of users [2] and time spent [3]. Thanks to these developments, LBS-based mobile applications (or apps) have come to be a lucrative and thriving market.

LBS in mobile applications lets users accomplish a variety of tasks, such as planning a route from one location to another or obtaining information about entertainment venues in the vicinity. By obtaining the location of their users, LBS are able to provide a personalized experience to their users. Unfortunately, location disclosure endangers the privacy of the user, opening up for a plethora of attacks. These attacks are typically classified into external and internal.

The most intuitive kind of attacks are *external*, where the attacker has a black-box view of the system and can act as an ordinary user. External attacks have been seen in many widely used commercial applications such as Foursquare [4], Tinder [5] and Grindr [6]. These attacks often rely on *trilateration* techniques to precisely position users based on multiple distance queries. In these situations, the service provider is a Trusted Third Party (TTP) while the information being disclosed among users needs to be limited.

Secondly, *internal* attackers have full access to the system, which makes the typical internal attacker the LBS providers themselves. The smartphone app Uber, connecting passengers with private drivers, has been the subject of much privacy debate. Uber and its employees have been allegedly involved in privacy-violating activities from stalking journalists and VIPs to tracking one-night stands [7]. Given how powerful they are, internal attackers are significantly harder to protect against.

In an effort to mitigate attacks such as those mentioned above, there has been substantial progress on privacy-enhancing LBS [8–10]. Some approaches separate some parts of the system [11] in order to make it harder for an attacker to gain full white-box access to the service. For these approaches, the same foundational issue remains: the user needs to trust (a part of) the service, thus not addressing internal attacks. Other promising tracks have emerged using cryptographic techniques [12, 13] where the user retains control of their data through (homomorphic) public-key cryptography (homomorphic encryption is detailed further in Section 3).

While these studies address increasingly more powerful attackers, their evaluation has been often focused on the privacy guarantees. At the same time, the ques-

tion of whether these mechanisms can be adopted by practical LBS applications has received limited attention.

The focus of this paper is on the location proximity problem, i.e., the problem of computing whether one user is within a distance from another user. The privacy goal is to reveal the proximity and nothing else about the location of the users. Cryptographic approaches can provably protect against internal attackers, while disclosing only proximity mitigates the external attacker. Such solutions are referred to as *privacy-preserving location-proximity (PPLP)* [14–18, 12, 13, 19] protocols.

Note that different existing solutions protect different parts of user's data. Many approaches provide k-anonymity [11, 10], where the location of the user is indistinguishable among a set of users. The primary objective of these solutions is to protect the identity from the attacker, while allowing them to learn k distinct possible locations of the user. For the scope of this study, only solutions where the location of the user is secret are considered.

This paper studies the applicability of PPLP in the setting of mobile apps. We categorize popular location social apps and analyze the trade-offs of privacy and functionality with respect to PPLP enhancements.

To investigate the practical performance trade-offs, we present an in-depth case study of an Android application that implements InnerCircle [19], a state-of-the-art protocol for privacy-preserving location proximity. This study indicates that the features of PPLP fits several scenarios of real-world LBS and that the performance of such protocols is, for coarse-grained precision, comparable to real applications. To summarize the main contributions of the paper:

1. We evaluate to what extent a state-of-the-art protocol can be applied to mobile applications without limiting their functionality. The study uses popular location-based social apps from the Google Play Store.
2. We investigate performance trade-offs by performance measurements of an implementation of InnerCircle [19], a state-of-the-art privacy-preserving location-proximity protocol in an Android application. The study compares the performance of the implementation to real-world applications

The paper is organized as follows. Section 2 studies the applicability of privacy-preserving proximity-testing protocols to real-world LBS by investigating the privacy vs. functionality trade-offs. Section 3 presents necessary background for the InnerCircle protocol. Section 4 describes the architecture of the Android-based implementation of InnerCircle. Section 5 studies the performance of the protocol and compares it with the performance of the real-world apps. Section 6 discusses the related work. Section 7 offers concluding remarks.

2 Applicability

This section studies the applicability of privacy-preserving proximity-testing protocols to real-world LBS. First, a new set of *features* for LBS is outlined. These features can be used to assign an LBS into a *category*. A privacy-preserving protocol is able to serve a fixed set of categories.

2.1 LBS Features and Categories

We identify features and categories of location-based services in order to aid the applicability analysis of privacy-preserving protocols for LBS. First, mobile LBS applications vary based on whose location information they provide to the user, herein called the *target type* feature: venues, acquaintances, and strangers. Second, the applications also vary based on the precision of the location information provided, called the *precision* feature: exact location, precise distance, or a boolean proximity result. Using these application features we will determine to what extent a given privacy-preserving mechanism is applicable to each application.

Important to note about the venue *target type* is that in most cases, a venue's location is normally not secret. Thus, for most common applications, there is little to gain by using privacy-preserving protocols towards a venue, as in this case the instigator can be told the venue's coordinates and then run all computations locally. Although there is no need for a privacy-preserving protocol to handle the venue's position in this case, the privacy of the requester's location needs to be protected by the implementation as to prevent location leaks to internal attackers such as via the IP address.

We define three app categories: *Point-of-Interest based (PoI), Friend-Finding (FF),* and *People-Discovery (PD)* apps. PoI apps are common venue-locator applications, e.g. where people wish to meet each other or find a shop of some kind. Friend-Finding apps are for keeping track of the whereabouts of close friends and family. People-Discovery apps are for locating new people to interact with.

Table 1 reflects what kinds of applications belong to which category. As expected, the PoI applications disclose the precise location of the venue. This is also the case for Friend-Finding applications, though one could imagine scenarios where users would not want their precise location known even to friends and family, e.g. when buying a gift. Surprisingly, many applications that facilitate interaction between strangers also disclose the exact location of the users to each other.

For any privacy-preserving proximity-testing protocol to be adopted, the application must have proximity precision. Further, it cannot be a venue target type,

Table 1. Categorizations for Location-Based Services

Precision \ Target Type	Venues	Acquaintances	Strangers
Exact Location	PoI	FF	PD
Precise Distance	–	–	PD
Proximity Boolean	–	–	PD

as then the location can be publicized instead. The services which could most easily adopt privacy-enhancing technologies are thus the Friend-Finding and People-Discovery, both with proximity precision. However, some applications might in their current state reveal more location information than strictly necessary. As such, applicability of a privacy-preserving protocol is grouped into three classes:

- Not Applicable: the mechanism sets overly strict limits on the information disclosure to support the features of the application.
- Partly Applicable: to incorporate the mechanism the application would require minor modifications to its features but would still be able to maintain its core purpose.
- Applicable: the mechanism can easily be incorporated into the mobile application without hampering the functionality of the mobile application.

2.2 Real-World Applications

This section examines how real-world mobile applications utilize LBS to determine if their functions can be supported by a privacy-preserving protocol. We focus on analyzing popular applications, as indicated by top hits from searching for "location social networking" on Google Play Store. Examining the applications has revealed several trends, discussed below. Table 2 summarizes the results, but before looking at the results the different applications are outlined.

MeetUp a PoI application that allows users to find meet-up venues and meet new people with similar interests. The user is able to search for meet-up events by specifying a radius and receiving exact location of the venue. As highlighted earlier, PoI applications cannot easily adopt privacy-preserving technologies, making them Not Applicable.

FourSquare a PoI venue finding application which allows users to search for various entertainment venues, providing their exact location. PPLP can not be incorporated by FourSquare without significant changes in the application, and are Not Applicable.

Family Locator is another Friend-Finding application. It allows users to locate their family members and friends and see their exact location. Modifying the application to only display proximity boolean would go against the intent of the application and as such the PPLP is deemed Not Applicable.

Badoo and *Singles Around Me* are dating applications (PD) allowing users to find matches in their area, displaying their location on a map. In order to implement PPLP, they would need to forego this feature which would be a relatively important change in its functionality. Therefore PPLP is deemed Not Applicable for these application.

LINK is a PD app that allows users to connect with others and form groups based on interests. LINK provides its users with the exact distance between two strangers. Similarly to LINK, *SKOUT* enables strangers to search for others based on various criteria and displays to instigator the target's precise distance. Although such functionality is not directly supported by PPLP, a minor change in the application's features would fix the issue. Hence we deem PPLP to be Partly Applicable for these applications.

MeetMe is a People-Discovery application which enables their users to chat with strangers with similar interests in their area. *Tagged* is another People-Discovery app allowing users to find and chat with people in vicinity. In both cases, proximity checks occur mainly between strangers and only proximity boolean is revealed to the users, maintaining all other location information concealed. As these features are well within the limits of PPLP, the protocol is deemed to be Applicable to these to applications.

Table 2. Applicability of PPLP to Popular Mobile Applications

Application	Category	Not Applicable	Partly Applicable	Fully Applicable
MeetUp	PoI	X		
FourSquare	PoI	X		
Family Locator	FF	X		
Badoo	PD	X		
Singles Around Me	PD	X		
LINK	PD		X	
SKOUT	PD		X	
MeetMe	PD			X
Tagged	PD			X

As visible in Table 2, PPLP protocols are of limited use to applications which focus on interaction between people who already know each other as such applications place less importance on location privacy and allow their users to see the precise location of different users on a map. On the other hand, PPLP are very promising for applications that facilitate interaction with strangers before a possible meeting in reality. In such cases proximity between users is important as users want to know in advance if the potential meeting is possible, but at the same time want to keep their location private as they have not built enough trust yet to reveal it. As search on Google Play Store shows that such applications are relatively popular, good location privacy-preserving mechanism applicability means such mechanisms can become quite important in the future of the mobile application market.

3 A Concrete PPLP Protocol

Hitherto, we have only touched upon PPLP protocols in general; the enforcement of such is discussed in this section. There are many published works describing how to accomplish different flavors of PPLP [14–18, 12, 13, 19]. In this work, *InnerCircle* by AnonymousAthors [19] was implemented to evaluate efficiency of a recent PPLP protocol on a smartphone device. InnerCircle is a good representative of state-of-the-art PPLPs as it provides protection against internal attackers while disclosing only location proximity, which is a good countermeasure against external attackers. Further, the authors provide evidence that the protocol could be efficient enough for usage in smartphone applications. Other protocols, such as the work by Sedenka et al. [13] provides the same security guarantees, but requires the use of multiple cryptographic schemes and has several additional round-trips between the parties as compared to InnerCircle. Reducing the number of round-trips proved a good choice, as this can cause a blow-up in communication, as seen in Section 4.

The mobile application produced in this work uses the InnerCircle protocol [19]. This particular protocol was chosen as it preserves location privacy against both internal and external attackers, while completing in a single round-trip. The key concept used in InnerCircle is, as mentioned previously, homomorphic encryption, which avoids the need for TTP. In recent years homomorphic encryption has became a popular choice for creating privacy preserving protocols and as such, it is a good representation of much of the state-of-the-art technology in location-privacy.

The protocol considers two principals, Alice and Bob, where Alice is the instigator. When Alice wants to query Bob to check if they are in each other's proximity, Alice constructs a *location request*. The location request encapsulates Alice's coor-

dinates, encrypted under her public key. Bob uses the information in the location request together with his own coordinates to create a *location response*. A location response is an array which encodes a single boolean value, which can be decoded using Alice's private key. For the full protocol, the reader is referred to the original paper [19]. For the scope of this work, it suffices to view the protocol as consisting of three steps: request construction, response construction to encode the boolean result, and response interpretation to decode the boolean. The encoding step which constructs the lesser than comparison is henceforth referred to as `lessThan()`, while the decoding step where Alice finds out whether Bob is in her proximity is called `inProx()`. As shown in Section 5, the `lessThan()` and `inProx()` methods are the more time-consuming operations in the protocol.

The key concept used in InnerCircle is, as mentioned previously, homomorphic encryption, which avoids the need for TTP and in recent years has became a popular choice for creating privacy preserving protocols [14, 20, 21, 13, 19]. Homomorphic encryption allows for computations to be evaluated on encrypted data. Formally, given the plaintext space $\mathcal{M}$ and the ciphertext space of a homomorphic scheme $\mathcal{C}$ such that encryption is a function $E : \mathcal{M} \to \mathcal{C}$ and decryption is $D : \mathcal{C} \to \mathcal{M}$, for any arithmetic formula $f : \mathcal{M}^k \to \mathcal{M}^k$ it is possible to construct $g : \mathcal{C}^k \to \mathcal{C}^k$ such that $D(g(E(\overrightarrow{m}))) = f(\overrightarrow{m})$. I.e., for any arithmetic formula in the plain it is possible to construct another formula to compute the same in the ciphertexts. There are several flavors of homomorphic encryption. Normally homomorphic encryption signifies Fully Homomorphic Encryption (FHE) schemes [22, 23]. FHE schemes are extremely powerful, and can evaluate any formula as described above, but are rather inefficient. On the other hand, there are schemes that are more limited in what they can compute – such as Additively Homomorphic Encryption (AHE) – but which are far more efficient [24]. The Authors Of [19] present several cryptosystems which can be used to instantiate InnerCircle. In this research we have chosen to use the ElGamal's [25] encryption system using 1024 bit keys since it had a notably fast performance in the original implementation.

Of interest is also how an array is used in InnerCircle to encode a boolean. Of course, using an array requires much more communication, memory and computational resources. However, due to the limitations of AHE, the authors of InnerCircle found this the most efficient approach. In essence, the array a is the result of a less-than operation. To check if $x < y$, one can check if $\exists y_i < y : x - y_i = 0$. The protocol creates the array such that it contains only uniformly random numbers, except for the case when $x < y$, when it contains a single (random) slot which contains the encryption of 0. The decoding step is thus to decrypt the array and check

for the existence of a zero. Further, as square roots can not be computed using homomorphic encryption, the square of the distance between Alice and Bob is compared to the square of the radius, which yields an array which is quadratic in r.

4 System Overview

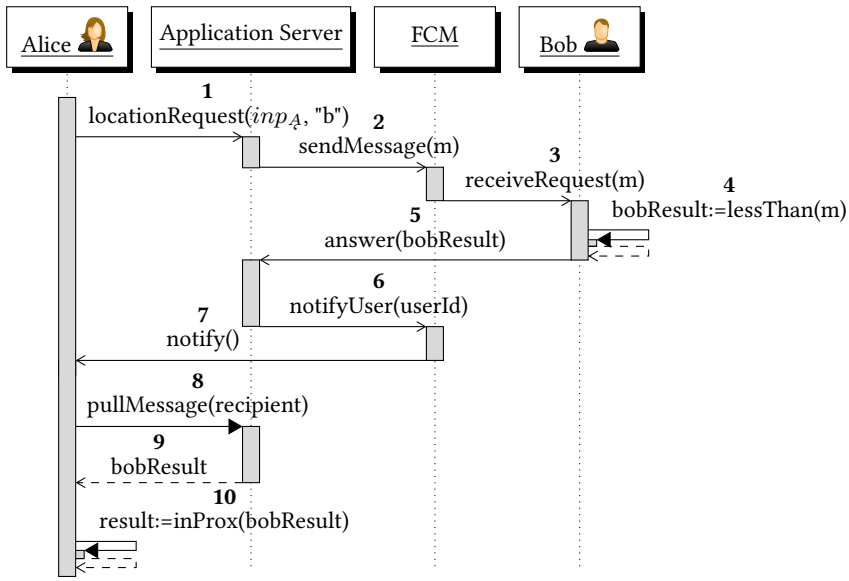

Fig. 1. Sequence diagram of a communication request

This section details the full system resulting from the implementation effort. In brief, the system consists of a mobile application for the Android operating system, an application server, and the messaging service used to send push notifications to the mobiles. Though network attackers are not considered in this work, such can easily be thwarted using HTTPS connections between all parties.

The original InnerCircle protocol finishes in a single round trip. As smartphones do not communicate in a peer-to-peer fashion the resulting implementation uses a larger protocol. The devices send messages to each other via push notifications, e.g. Firebase Cloud Messaging (FCM) on Android or Apple Push Notification Service (APSN) on iOS devices. Further, some of the messages are much too large to be sent via the FCM push service, which limits the size of the messages to 4 kilobytes. To send larger messages, a combination of the application server and the FCM service was used, where the FCM services were used to notify when a message is available

for on the server. This results in a protocol using 7 messages rather than 2 as intended by the original proposal. This communication overhead is much higher than the original prototype implementation by AnonymousAthors [19].

The resulting application thus involves communication between two clients and two servers. The clients are herein called Alice and Bob as in the original protocol, and the two servers the Application Server, and FCM. Messages exchanged during a single request are shown in Figure 1. Alice generates a location request which is sent to the Application Server (1), then from the Application Server to FCM (2), and from FCM to Bob (3). Bob then creates a proximity result using `lessThan()` (4), which is sent directly to the Application Server (5). The server notifies FCM (6), which in turn notifies Alice that the answer is ready to be retrieved (7). Alice then fetches the answer from the Application Server (8 & 9). When Alice retrieves the answer, she interprets ti using `inProx()` which will tell her if Bob is in her proximity (9).

InnerCircle assumes a euclidean plane, but the Android GPS interface provides longitude and latitude which have to be converted into Cartesian coordinates. The search radius and the coordinates input to InnerCircle are specified in an arbitrary distance unit. Thus, converting from GPS to the distance unit allows for discretization to take place, and the unit of the resulting Cartesian coordinate system can correspond to a millimeter, a yard, a meter or a kilometer. As the performance of InnerCircle is proportional to the radius, this is a useful tool to trade precision for efficiency.

5 Performance

This section presents the performance benchmarks performed in this study. First follows a brief description of the setup, after which efficiency both in terms of CPU and network usage is presented and discussed.

Testing was performed on two smartphone devices connected to a WiFi network, with the application server hosted on the same network. Each device is able to act as Alice as well as Bob to measure how the different phones handle both roles of the protocol. The devices used were two Samsung Galaxy S6 (model SM-G920F). The model was released in April 2015 and has a 64-bit Exynos 7 Octa 7420 system-on-chip, consisting of four 2.1 GHz Cortex-A57 cores, and four 1.5 GHz Cortex-A53 cores, and 3 GB of LPDDR4 RAM.

The protocol was executed 25 times with radius 25, 50, 75 and 100. The outcome total time taken to execute a request was measured as well as the CPU time spent computing by each party. CPU time measures the cryptographic parts of the proto-

col, with the two larger parts being `lessThan()` and `inProx()`. Encryption of Alice's coordinates and distance computation by Bob are included in the total CPU time but not displayed individually, as they are both relatively quick methods. Total Time represents the amount of time taken for the application to display the results to the user and includes the CPU time of both users as well as the network delay. Measurement starts when the user presses the locate button, and ends when the answer is available on the user's phone. The source code for both the application server and the smartphone app has been made publicly available[1].

5.1 CPU Performance

Table 3. Benchmark results (in milliseconds)

Radius	Total Time	CPU time `inProx()`	CPU time `lessThan()`	CPU time Total
25	4318.8	1500.2	1542.1	3124.1
50	11508.1	4978.0	5114.5	10174.2
75	21911.0	10235.4	10061.2	20376.4
100	35736.5	17355.1	16453.1	33887.0

The results of the benchmarks can be seen in Table 3, which shows the average time consumed for the collected data. As seen from Table 3, the time for protocol executions increases drastically with increase in radius. Mapping the average protocol execution times onto a line chart in Figure 2 suggests that the increase in time is more than linear, though less than the expected quadratic increase due to optimizations detailed in the original paper [19].

Table 4. Efficiency Results for Real Applications in Seconds

MeetUp	Foursquare	Badoo	Singles Around Me	LINK	SKOUT	MeetMe	Tagged
1.73	1.65	2.9	7.34	2.14	1.77	2.19	3.17

The performance of the Google Play applications was measured using a stopwatch, starting from the moment user opts to check location proximity to the moment the application displays its results to the user. The applications were tested 10

[1] https://bitbucket.org/innercircleandroid/

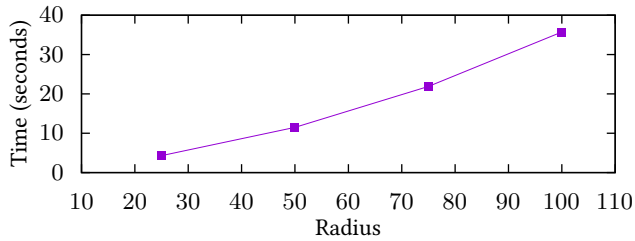

Fig. 2. Averages of execution times for protocol in different ranges

times each. The results can be seen in Table 4. These benchmarks can be compared to the total time in Table 3, allowing to determine if the efficiency of the protocol is sufficient enough to be successful in the market of mobile applications.

5.2 Network Usage

Additionally, network usage of the protocol is measured. Network usage is relevant for two reasons. First, users are often out of range of free network access, which means network usage has a financial cost. Secondly, SMC solutions, such as homomorphic encryption, are often limited either by computational or communications resources. Determining which is the limiting factor of the protocol is vital contribution of this work. Network usage is measured by testing the protocol 10 times at distances of 25, 50, 75 and 100 units and recording Bob's answer size in bytes.

Table 5. Average network usage in kilobytes

25	50	75	100
143.242	489.307	1025.198	1740.382

The protocol on average used 143 and 1740 kilobytes when transferring messages for 25 and 100 distance unit proximity checks respectively. The results of network usage can be seen in Table 5. As real applications would transfer unencrypted coordinates, network usage would be minimal and irrelevant. As such, network usage of real applications was not measured.

5.3 Discussion

The results shows the protocol yields an answer within 36 seconds for a radius of 100. Narayanan et al. [12] implemented a novel protocol in an Android application with an execution time of 46 seconds. Their results are comparable to our implementation, although their solution would likely perform better for larger radiuses. However, time frames over half a minute are unacceptable for most practical scenarios. Furthermore, it is reasonable to assume that Bob will not always be stationary in his position while the answer is generated on his phone. Given that on foot, at 5 km/h, it takes 72 seconds to cover a distance of 100 meters (more so by transportation vehicle), it is plausible that Bob's position can change to in/out of range before Alice receives an answer, invalidating the result.

On the other hand, using InnerCircle with smaller radiuses is definitely feasible in a mobile application; it took only 4.3 seconds to calculate proximity for proximity requests for a radius of 25 units. This is within the time frames of real applications, which provide output to the users in the range of 1 to 7 seconds. However, it is unlikely that Google Play application's response times are affected by the radius specified by the users. The majority of mobile applications focus on rather large distances. Thus, the distance unit used by InnerCircle must be tweaked to accommodate these, and should not be chosen as, for instance, 1 meter. Even though an implementation of InnerCircle would fall short of practical applicability if both high precision and large radius are needed, our study shows that state-of-the-art location-proximity protocols are efficient enough for integration in most mobile applications with more restricted precision than the applications are currently using.

As a concrete example, consider using InnerCircle to enhance the privacy of the previously mentioned Tagged application. Tagged allows users to check for other users within 100 kilometers. To utilize InnerCircle at such distances, the application would need to use a distance unit of approximately 4 kilometers to be able to use a radius of 25. The expected time for a proximity check would be only 4.3 seconds, which is comparable to the current time of the Tagged application.

The negative side of using a discretized plane where the unit is rather large is that it introduces an imprecise edge at the circle which denotes Bob's proximity to Alice. However, this error is fairly small relative to the proximity radius. Using the previous example, Alice is able to check the proximity of Bob in the radius of approximately 100 kilometers with the error range being 4 kilometers, which is an error of 4%. Although current applications most likely are checking proximity with higher accuracy at the same range, we believe the significance of the error is small in comparison to the radius.

In regards to network usage, the proximity result is at most 1.7 megabytes for proximity checks up to 100 distance units. Such sizes are comparable to average size of images, which the users rarely take into consideration when browsing Internet on their mobile devices. As such, we believe the protocol's network usage would be insignificant when incorporated into an application. Users could check proximity many times without much concern. This speaks favorably towards the protocol's implementation in mobile applications.

6 Related Work

The relevance of research on location-privacy has seen some debate, with studies showing mixed results on whether location-privacy is important to users of LBS or not. Barkhuus and Dey [26] compared the two scenarios of location-tracking services and location-aware services and performed an experimental case study with 16 participants. The participants had more privacy concerns regarding location tracking services compared to location-aware services, but in general were not overly concerned about privacy of their location data. Nevertheless, Barkhuus and Dey recommended focusing on developing services around location-aware concept. In case of location-tracking services, the researchers believe such services can still be acceptable as long as users have the option to turn-off the tracking capability at any time.

Xu and Gupta [27] developed a model to examine the impact of privacy concerns on intention to use LBS. They found that performance expectancy had a positive impact on participants' intention to use LBS and effort expectancy was positive only for inexperienced users, but privacy concerns had no direct effect. Interestingly, privacy concerns negatively impact performance and effort expectancy, thus indirectly affecting user decision to use LBS mobile services. This implies that privacy concerns are relevant to at least a limited extent to user of LBS applications.

Zickuhr [28] studied use of LBS in mobile apps by Americans. The findings show that the use of such applications is rapidly growing, from 55% of smartphone owners using LBS applications in 2011 to 74% of smartphone owners using LBS in 2012. Furthermore, taking into account that smartphone ownership itself quickly grew from 35% of adults in 2011 to 46% in 2012, it is safe to assume that the importance of LBS privacy concerns, even if relevance is currently debatable, will grow in the coming years.

6.1 Privacy-Preserving Technologies

There is significant literature on both protecting users' privacy against internal and external attackers. For internal attackers, there are several generic techniques not tied to LBS. For external attackers on LBS, there are two core tracks [29]. The first idea is to minimize each individual disclosure. For instance, by disclosing distances instead of positions, etc. Secondly, a good countermeasure is to discretize the location data by dividing the plane into a grid, such that many coordinates in a grid-cell are mapped to the same location. The grid cells need to be large enough that the imprecision is sufficient to provide privacy. While the first often allows a service to remain unchanged while providing better protection, the second can provide strict guarantees of how much information the attacker is able to learn.

Generic Privacy-Preservation For internal attackers, there are a number of different techniques. One popular strategy is the "k-anonymity model" [30–32], which hides the user among similar other users. This makes the original user indistinguishable from the rest of the population and thus anonymous. However, all efficient techniques for k-anonymity require a third party to be set up, which again opens up for internal attackers at this new party.

A generic approach to hide sensitive data from service providers is to utilize Secure Multi-party Computation (SMC), which is a research field of considerable size. SMC enables multiple parties to compute on private data without revealing their inputs. The ability to compute functions without revealing inputs allows for private data to remain confidential while being handled by 3rd parties, which completely removes the need for trusting a third party. There are three tracks in the literature that achieve SMC, each with its own large community: Secret Sharing (SS) [33], Garbled Circuits (GC) [34] and Homomorphic Encryption (HE) [22]. SS-based techniques show very promising performance, and are seeing some commercial use [35]. However, they typically require a set of non-colluding servers, which makes it unsuitable when the goal is to not put any trust in the service provider(s). Techniques based on GC have seen promising performance utilizing the Intel Advanced Encryption Standard Instructions through protocols tailored for this particular instruction set. As most mobile devices use ARM processors, it is unlikely that the performance results can be extrapolated to mobile devices. Further, GC offer a one-off solution, where any results (except the output) should not be reused in further computations.

As previously detailed, homomorphic encryption makes it possible to perform mathematical operations on encrypted data. The ground-breaking result by Gentry [23] presented the first Fully Homomorphic Encryption (FHE) scheme, which

is capable of computing arbitrary arithmetic formulas. Following Gentry's work, there's been numerous improvements for FHE [36–38]. However, so far there is no FHE scheme that is comparable in efficiency to schemes that are just additive or multiplicative. There have been many works that utilize homomorphic encryption to create privacy-preserving protocols in areas such as location-privacy and biometric authentication [20, 13, 19, 21, 14].

Privacy-Preservation in LBS Puttaswamy et al. [39] present a new technique for location privacy by coordinate system transformations, called LocX. Each user has a secret for which it's coordinate system is translated, and a set of friends. The secrets are distributed to each user's friends, such that only the user's friends may understand how coordinates are mapped. A prototype has been developed and it showed that it can be used in commercial applications with minimum overhead. However, unlike other protocols mentioned in this section, the user's exact location is revealed to all users with the secret, which forces the users to limit their social circle to users they trust with their location.

Further, to generate dummy data and present this to the LBS is a viable option to hide the user's location. Zhou et al. [40] propose a system called TISSA. TISSA allows users to choose what data an application can access. In case an application demands access to data that the user is unwilling to provide, the system sends dummy data as substitute, keeping the real data private. The system was tested in Android OS and successfully prevented leakage of information to restricted applications and caused no significant slow down to performance of the phone. However, using only dummy inevitably prevents the application from functioning properly. Kido et al. [41] propose a system which sends LBS providers real user data as mixed with dummy data. As the LBS providers cannot distinguish between real and fake data, the anonymity of the user is preserved. However, the solution causes large communication overhead as all users need to send many additional messages with dummy data for each real query.

There are not many works that provide an in-depth discussion of PPLP on Mobile Devices. Narayanan et al. [12] provides use cases where LBS mobile applications could be used and how their proposed protocol would relate to such applications. However, it is debatable whether the use cases themselves are realistic examples of LBS use and sufficient proof that the protocol could be applicable enough to be used in general applications.

7 Conclusion

This study furthers the knowledge of how well currently existing cryptographic privacy-preserving protocols apply to real-world mobile apps. To this end, we have road-mapped popular location-based social maps and identified scenarios where privacy-preserving location proximity is desired. The category of People-Discovery apps turns out to be a particularly promising fit. We conclude that the protocol can be fruitfully applied to a number of popular applications, in particular for the ones that facilitate meetings in real life between strangers, such as meet-up and dating apps.

Further, we have implemented InnerCircle, a state-of-the-art privacy-preserving location proximity protocol and integrated it in an Android app. With respect to performance, we arrive at the conclusion that InnerCircle on Android matches real applications at radius values of 25 and 50 units while values at 75 units and above are not yet within a reach. The average network usage for 25 units is 143 KB and for 100 units is 1740 KB respectively. With less precise coordinates the protocol can check the radius of 100 kilometers, using 25 unit radius in only 4 seconds, which shows that the protocol is efficient enough for implementation in real applications.

Acknowledgments This work was partly funded by the European Community under the ProSecuToR project and the Swedish research agency VR.

References

1. AngelList, “Location based services startups,” Sep. 2014, https://angel.co/location-based-services.
2. A. Lella, “Number of Mobile-Only Internet Users Now Exceeds Desktop-Only in the U.S.” https://www.comscore.com/Insights/Blog/Number-of-Mobile-Only-Internet-Users-Now-Exceeds-Desktop-Only-in-the-U.S, Apr. 2015.
3. K. Dreyer, “Mobile Internet Usage Skyrockets in Past 4 Years to Overtake Desktop as Most Used Digital Platform,” http://www.comscore.com/Insights/Blog/Mobile-Internet-Usage-Skyrockets-in-Past-4-Years-to-Overtake-Desktop-as-Most-Used-Digital-Platform, Apr. 2015.
4. D. Coldewey, “"Girls Around Me" Creeper App Just Might Get People To Pay Attention To Privacy Settings,” http://techcrunch.com/2012/03/30/girls-around-me-creeper-app-just-might-get-people-to-pay-attention-to-privacy-settings/, Mar. 2012.
5. M. Veytsman, “How i was able to track the location of any tinder user,” http://blog.includesecurity.com/2014/02/how-i-was-able-to-track-location-of-any.html, Feb. 2014.

6. C. Paton, "Grindr urges LGBT community to hide their identities as Egypt persecutes nation's gay community," http://www.independent.co.uk/news/world/africa/grindr-urges-lgbt-community-to-hide-their-identities-as-egypt-persecutes-nations-gay-community-9757652.html, Sep. 2014.
7. C. Bessette, "Does Uber Even Deserve Our Trust?" http://www.forbes.com/sites/chanellebessette/2014/11/25/does-uber-even-deserve-our-trust/, Nov. 2014.
8. J. Krumm, "A survey of computational location privacy," *Personal and Ubiquitous Computing*, vol. 13, no. 6, 2009.
9. M. Terrovitis, "Privacy preservation in the dissemination of location data," *SIGKDD Explorations*, vol. 13, no. 1, 2011.
10. E. Magkos, "Cryptographic approaches for privacy preservation in location-based services: A survey," *IJITSA*, vol. 4, no. 2. [Online]. Available: http://dx.doi.org/10.4018/jitsa.2011070104
11. N. Talukder and S. I. Ahamed, "Preventing multi-query attack in location-based services," in *WISEC 2010*. [Online]. Available: http://doi.acm.org/10.1145/1741866.1741873
12. A. Narayanan, N. Thiagarajan, M. Lakhani, M. Hamburg, and D. Boneh, "Location privacy via private proximity testing," in *Proceedings of the Network and Distributed System Security Symposium, NDSS 2011*. [Online]. Available: http://www.isoc.org/isoc/conferences/ndss/11/pdf/1_3.pdf
13. J. Sedenka and P. Gasti, "Privacy-preserving distance computation and proximity testing on earth, done right," in *ASIA CCS '14, Kyoto, Japan - June 03 - 06, 2014*. [Online]. Available: http://doi.acm.org/10.1145/2590296.2590307
14. G. Zhong, I. Goldberg, and U. Hengartner, "Louis, lester and pierre: Three protocols for location privacy," in *Privacy Enhancing Technologies, 7th International Symposium, PET 2007*. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-75551-7_5
15. L. Siksnys, J. R. Thomsen, S. Saltenis, M. L. Yiu, and O. Andersen, "A location privacy aware friend locator," in *Advances in Spatial and Temporal Databases, 11th International Symposium, SSTD 2009*. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-02982-0_29
16. L. Siksnys, J. R. Thomsen, S. Saltenis, and M. L. Yiu, "Private and flexible proximity detection in mobile social networks," in *Eleventh International Conference on Mobile Data Management, MDM 2010*. [Online]. Available: http://dx.doi.org/10.1109/MDM.2010.43
17. D. Freni, C. R. Vicente, S. Mascetti, C. Bettini, and C. S. Jensen, "Preserving location and absence privacy in geo-social networks," in *19th Conference on Information and Knowledge Management, CIKM 2010*. [Online]. Available: http://doi.acm.org/10.1145/1871437.1871480
18. S. Mascetti, D. Freni, C. Bettini, X. S. Wang, and S. Jajodia, "Privacy in geo-social networks: proximity notification with untrusted service providers and curious buddies," *VLDB J.*, vol. 20, no. 4, 2011. [Online]. Available: http://dx.doi.org/10.1007/s00778-010-0213-7
19. P. A. Hallgren, M. Ochoa, and A. Sabelfeld, "Innercircle: A parallelizable decentralized privacy-preserving location proximity protocol," in *PST*. IEEE, 2015, pp. 1–6.

20. Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, I. Lagendijk, and T. Toft, "Privacy-preserving face recognition," in *Privacy Enhancing Technologies*, 2009.
21. A. Sadeghi, T. Schneider, and I. Wehrenberg, "Efficient privacy-preserving face recognition," in *Information, Security and Cryptology - ICISC 2009*. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-14423-3_16
22. R. L. Rivest, L. Adleman, and M. L. Dertouzos, "On data banks and privacy homomorphisms," *Foundations of secure computation*, vol. 32, no. 4, pp. 169–178, 1978.
23. C. Gentry, "Fully homomorphic encryption using ideal lattices," in *STOC*, 2009, pp. 169–178.
24. P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Advances in Cryptology - EUROCRYPT 1999*. [Online]. Available: http://dx.doi.org/10.1007/3-540-48910-X_16
25. T. E. Gamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," in *Advances in Cryptology, CRYPTO 1984*. [Online]. Available: http://dx.doi.org/10.1007/3-540-39568-7_2
26. L. Barkhuus and A. K. Dey, "Location-based services for mobile telephony: a study of users' privacy concerns," in *Human-Computer Interaction INTERACT '03: IFIP TC13 International Conference on Human-Computer Interaction, 2003, Zurich, Switzerland.*
27. H. Xu and S. Gupta, "The effects of privacy concerns and personal innovativeness on potential and experienced customers' adoption of location-based services," *Electronic Markets*, vol. 19, no. 2-3. [Online]. Available: http://dx.doi.org/10.1007/s12525-009-0012-4
28. Kathryn Zickuhr, "Three-quarters of smartphone owners use location-based services," May 2012, http://www.pewinternet.org/files/old-media/Files/Reports/2012/PIP_Location_based_services_2012_Report.pdf.
29. 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 CCS, 2015*. [Online]. Available: http://doi.acm.org/10.1145/2810103.2813605
30. M. Gruteser and D. Grunwald, "Anonymous usage of location-based services through spatial and temporal cloaking," in *MobiSys 2003*. [Online]. Available: http://www.usenix.org/events/mobisys03/tech/gruteser.html
31. B. Gedik and L. Liu, "Protecting location privacy with personalized k-anonymity," *IEEE Trans. Mob. Comput.*, vol. 7, no. 1. [Online]. Available: http://dx.doi.org/10.1109/TMC.2007.1062
32. C. Wu, C. Huang, J. Huang, and C. Hu, "On preserving location privacy in mobile environments," in *Ninth Annual IEEE International Conference on Pervasive Computing and Communications, PerCom 2011*. [Online]. Available: http://dx.doi.org/10.1109/PERCOMW.2011.5766939
33. A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, 1979. [Online]. Available: http://doi.acm.org/10.1145/359168.359176
34. A. C. Yao, "Protocols for secure computations (extended abstract)," in *23rd Annual Symposium on Foundations of Computer Science, 1982*. [Online]. Available: http://dx.doi.org/10.1109/SFCS.1982.38

35. D. Bogdanov, M. Jõemets, S. Siim, and M. Vaht, "How the estonian tax and customs board evaluated a tax fraud detection system based on secure multi-party computation," in *Financial Cryptography and Data Security - 19th International Conference, FC 2015*. [Online]. Available: http://dx.doi.org/10.1007/978-3-662-47854-7_14
36. Y. Huang, D. Evans, J. Katz, and L. Malka, "Faster secure two-party computation using garbled circuits," in *USENIX Security*, 2011.
37. A. López-Alt, E. Tromer, and V. Vaikuntanathan, "On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption," in *Symposium on Theory of Computing Conference, STOC 2012*. [Online]. Available: http://doi.acm.org/10.1145/2213977.2214086
38. C. Gentry, A. Sahai, and B. Waters, "Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based," in *CRYPTO (1)*, 2013.
39. K. P. N. Puttaswamy and B. Y. Zhao, "Preserving privacy in location-based mobile social applications," in *Eleventh Workshop on Mobile Computing Systems and Applications, HotMobile 2010*. [Online]. Available: http://doi.acm.org/10.1145/1734583.1734585
40. Y. Zhou, X. Zhang, X. Jiang, and V. W. Freeh, "Taming information-stealing smartphone applications (on android)," in *Trust and Trustworthy Computing - 4th International Conference, TRUST 2011*. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-21599-5_7
41. H. Kido, Y. Yanagisawa, and T. Satoh, "An anonymous communication technique using dummies for location-based services," in *Proceedings of the International Conference on Pervasive Services ICPS 2005*. [Online]. Available: http://dx.doi.org/10.1109/PERSER.2005.1506394