Cover Locations: Availing Location-Based Services Without Revealing the Location

Sai Teja Peddinti
Computer Science and Engineering
Polytechnic Institute of New York University
Brooklyn, NY USA
psaiteja@cis.poly.edu

Avis Dsouza
Computer Science and Engineering
Polytechnic Institute of New York University
Brooklyn, NY USA
adsouz01@students.poly.edu

Nitesh Saxena
Computer and Information Sciences
University of Alabama, Birmingham
Birmingham, AL USA
saxena@cis.uab.edu

ABSTRACT

Location-Based Services (LBSs) have been gaining popularity due to a wide range of interesting and important applications being developed. However, the users availing such services are concerned about their location privacy, in that they are forced to reveal their sensitive location information to untrusted third-parties. In this paper, we propose a new privacy-preserving approach, *Cover Locations*, which allows a user to access an LBS without revealing his/her actual location. Based on its current location, the user's device queries for a few specifically chosen surrounding locations and constructs the results corresponding to its location from the results obtained for each queried location. Since the user location *does not leave* the user's device – as either a latitude and longitude pair, or as an obfuscated region – the user is guaranteed very high level of privacy. The Cover Locations approach only requires minimal changes on the user's device and can be readily deployed by privacy-conscious users. An adversary, trying to identify the user location, can only resolve the location to few triangular regions and not to the actual location itself. We evaluate the privacy provided by Cover Locations based on the number of locations queried and the total area under the resolved triangular regions. We also ascertain the robustness of Cover Locations approach when the adversary has access to a short-term user history, employing machine learning techniques. Overall, our results show that the proposed solution, which requires minor computations without the need for any out-of-band information such as traffic densities in a region or the road network information, is superior to other client-based solutions.

Categories and Subject Descriptors

K.6.5 [**Management of Computing and Information Systems**]: Security and Protection.; K.4.1 [**Computers and Society**]: Public Policy Issues—*Privacy*

WPES'11, October 17, 2011, Chicago, Illinois, USA.

General Terms

Design, Security, Experimentation

Keywords

Location-based Services, Location Privacy, k-anonymity

1. INTRODUCTION

Over the recent years, there has been a burgeoning adoption of mobile and wireless devices followed by an increase in the number of applications available for these devices. Many of these applications provide a certain service to the users based on some user input. One subclass of these applications, called Location-Based Services (LBSs), take as an input the user location and provide some service pertaining to the geographical location of the user. These services might report nearby points of interest [8], locate nearby friends [2], support locality-based gaming [1], or offer navigational help [3].

The LBSs are provided by independent service providers, and the way they handle the user reported locations is not transparent. Once the user location information leaves the user's device, the user is in no control as to how this information could be used by the LBS provider. The LBS provider can, for instance, aggregate all the locations obtained from the user and track the user's daily commute patterns, identify his/her home and work locations, and sell the information to third parties for targeted advertising, thereby invading the user's privacy. One recent example highlighting the location privacy issue is the tracking of user location on iPhones [9]. In fact, many smartphone users have shown serious concerns about their location privacy [26].

1.1 Our Contributions

In this paper, we propose a novel location privacy-preserving technique, called *Cover Locations*, which allows the user to access the location-based services without leaking out his/her exact location. In this technique, the user does not query the results for his location, but instead queries the results for specifically chosen nearby locations, called cover locations. These obtained results are then processed on the user's device to generate the results corresponding to the user location. Since the user location *does not leave* the user device, the user can be assured of the level of privacy protection offered by Cover Locations approach. The adversary can only identify a number of potential regions in

one of which the user could be located, but it cannot obtain any information as to the current exact location of the user. The approach is randomized in that two invocations of the underlying algorithm on the same input (user location) will generate independent sets of cover locations.

We evaluate the Cover Locations approach by simulating an adversarial LBS attempting to break the system. Given a set of k locations received by the LBS, we show that it is not possible to identify a small triangular region in which the user is located with a probability better than $\frac{1}{k-2}$. In fact, the average probability is much less than $\frac{1}{k-2}$. As an example, for $k = 8$, the average probability of guessing user's location is only about 0.06, while the upper bound is 0.17.

Even if the adversarial LBS has access to prior user history, using machine learning techniques, we additionally demonstrate that the adversary cannot make a better decision than the earlier case whereby the history was not available. This is an improvement over prior location privacy mechanisms, such as [25, 30, 23] (reviewed in the following section), which have been shown to be vulnerable to this class of attacks.

1.2 Paper Outline

We initially review existing location privacy-preserving techniques in Section 2. Next, we present our threat model in Section 3. In Section 4, we describe our new approach, Cover Locations, followed by a discussion of its implementation in 5. In Section 6, we evaluate the proposed technique and estimate the level of privacy provided to the end user under our threat model.

2. RELATED WORK

In this section, we review existing location privacy-preserving mechanisms.

One direct approach to provide location privacy to the user is by making it hard to attribute location queries to their respective users. This can be achieved by stripping the location queries off of any user identifying information and assigning a pseudonym. However, such methods are not completely effective, as the attacker can still relate all the queries with the same pseudonym and identify the home (and thereby the user details from the street address listings [24]) from the pseudonymized GPS tracks [21]. Even when the location queries are completely anonymized (by not using a pseudonym), it has been shown that all the queries belonging to a trip could be related [19].

Another approach is to distort the actual user locations by adding random noise [14, 7] and making it hard for the adversary to identify the actual location. However, prior studies, such as [24], indicate that the amount of random noise to be added to prevent any tracking is large. Instead of distorting the individual locations, one can query for a region such that at least k users of the service are present in that region. Such k-anonymity based spatial cloaking techniques [18, 12, 15] make it hard for the adversary to determine where the user is exactly located within the region. These spatial cloaking services require additional third-party infrastructure to aggregate the individual user locations and generate the regions for querying the LBS. When the LBS sends the response, these third party servers are responsible for de-multiplexing/filtering the results corresponding to each user location and forwarding them to the users. For the spatial cloaking techniques, the value of k is the security parameter, determining the level of anonymity provided to the end users.

This cloaking of the user locations into a region and demultiplexing the obtained results needs to be performed for every user query, resulting in performance and scalability issues. Moreover, these third-party servers might become single-points-of-failure and easy attack targets, that can bring down the whole anonymization scheme. Furthermore, replacing the user locations with regions might not be supported by certain applications for which the accurate location information is necessary. Care should be taken while generating these cloaked regions to prevent any kind of correlation attacks [16] (where the adversarial LBS can correlate the regions sent by the anonymization server and break the k-anonymity) and other attacks possible because of certain background information available to the LBS [32].

As an alternative to these third party infrastructures, peer-to-peer solutions have been developed, which require the participation of at least k peers to ensure the requirement of k-anonymity is satisfied [13, 17]. However, it is not always possible to rely on other users of the service. Private Information Retrieval (PIR) protocols, such as [16, 22, 20], have been developed which do not require any third party infrastructure or additional users; but offer provable privacy protection to the end user. However, these PIR protocols require changes to be made on both the client (user device) and the server, and they are currently not feasible to be deployed in practice due to their high computation and communication overhead.

Some decentralized and autonomous k-anonymity based solutions, such as [25, 30, 23], also exist. These solutions accompany many false queries along with the real user query sent to the LBS. When the LBS responds to each of these queries, the user simply filters the response to her actual query and discards the rest. By ensuring that these fake queries are in sufficient number and that they look realistic, the probability of a LBS identifying the real query could be reduced to a desired level. This approach is more attractive because it does not require any additional infrastructure (e.g., additional third party servers or presence of other users) or changes at the server, and are available for ready deployment by privacy-conscious users. However, recent research [29] shows that user location queries may stand out among the mix of noisy and real queries, and can be extracted with high probabilities (much higher than $1/k$) by utilizing machine learning techniques and correlation attacks. This implies that it is hard to generate realistic looking noisy queries which are close to user query patterns.

Each of the above solutions provides different degree of privacy to the end user. In most cases, as these solutions are based on totally different underlying principles, it is hard to compare the level of privacy provided by these solutions. However, recently, attempts have been made to quantify the level of privacy provided by different solutions [31].

3. THREAT MODEL

We adopt a threat model similar to that of [29], used in the context of query obfuscation techniques. In our model, the LBS itself is the adversary, wanting to track the users from the reported locations for its own personal reasons. These reasons could range from causing physical harm to performing a robbery, or even selling the information to third party advertisers. With the use of Cover Locations technique, the

LBS receives many different (cover) locations at the same time. The LBS's goal is to extract the user location from these reported locations. In our model, we assume that the adversary is passive and only tracks the locations reported by the user without manipulating the responses. Alternatively, an active adversarial LBS can manipulate the location query results in such a way as to lure the user in revealing his location. Moreover, we assume that the LBS can relate all the queries coming from a user (using identifiers). This implies that the user is not making use of an anonymizing network, such as Tor, or proxy servers which hide the query source from the LBS. It is to be noted that if there are no application specific identifiers, and network level identifiers are masked (because the mobile network service provider itself acts as a proxy or if the user uses an anonymizing network), then the user already enjoys a certain level of anonymity obviating the need for additional tools. Apart from the location history passively recorded by the LBS, we assume that the attacker does not use any external or out-of-band information (like searching in Google or yellow pages).

Based on the scenarios in which the Cover Locations technique is deployed, we can determine if any additional information is available to the adversary. If the anonymization technique is available to new mobiles or new navigational devices alone, then the adversary does not enjoy any advantage. However, if the Cover Locations technique is released as a software update or as an application onto the existing mobile or navigational devices, then it is valid to assume that the LBS possesses a short-term history of location-based queries posed by a user prior to using the Cover Locations tool. The adversary could leverage this additional information (i.e. location history of the user) to identify the current location of the user. The second scenario is realistic since we can not expect the users to reinvest in new devices just for protecting their privacy. We note that such an attacker model has previously been incorporated in the context of location privacy [29] and web search privacy [27, 28].

4. COVER LOCATIONS: A NEW ANONYMIZATION TECHNIQUE

4.1 Basic Idea and Overview

In the privacy-preserving methods proposed so far (Section 2), we can see that either the user location is being revealed to the LBS accompanied with many fake location queries, or the accuracy of the location is being decreased (like querying a region instead of a point) so as to provide privacy to the end user. We propose a new method, Cover Locations, which enables the user to obtain nearby Points of Interest (POI) without revealing the user's exact location. The idea is to query for specifically chosen nearby locations; the results of these locations are then processed on the user's device to generate the results corresponding to the user location. Since the user location does not leave the device (either as an exact latitude-longitude value or as an inaccurate region), the chances of privacy leakage are greatly reduced.

When we query for nearby POI by sending the Latitude-Longitude location, say l, the LBS returns the results within a certain radius r around the reported location l (let us call it *User Circle*). (Let us denote the results for location l by $result_l$.) We take advantage of this functionality. Let the actual user location be l; we initially choose three locations l_1, l_2, l_3, such that the circles of radius r around these points completely cover or overlap the User circle (as shown in Figure 1). Since the circle of radius r around l is completely overlapped, we can say that

$$result_l \subseteq (result_{l_1} \cup result_{l_2} \cup result_{l_3}).$$

Hence, instead of leaking out user location l, we query locations l_1, l_2, l_3, and compute the results corresponding to the user location l from the results of these three locations. As indicated in Figure 2, it is possible to have just two circles (with radius r around two reported locations l_1, l_2) cover the User Circle. However, the reported locations would be too close to the actual user location, allowing for easy identification. Hence, we believe that at least 3 locations need to be queried for completely covering the User Circle. Let us call these locations responsible for covering up the User Circle as *cover locations*. It is possible to incorporate more than three cover locations (like four cover locations, each at the vertex of a square, covering the entire User Circle), but we show that three locations are sufficient to provide acceptable levels of privacy.

If only the three cover locations are queried by the user, the adversarial LBS can try to identify the user location in the total covered area. For this, the adversary can determine the extent to which the User Circle can extend into each of the circles around the cover locations. Referring to Figure 3, the blue arcs indicates the total covered area, and the red circles indicate the extent to which the User Circles overlaps with each of the cover location circles. The centers of the red circles in the figure determine the bounds of the region where the user could be located (since the user should be able to construct his/her location results from the results of the three surrounding points). We can approximate this

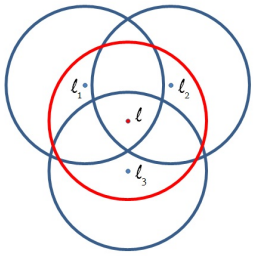

Figure 1: User Circle completely overlapped by surrounding circles

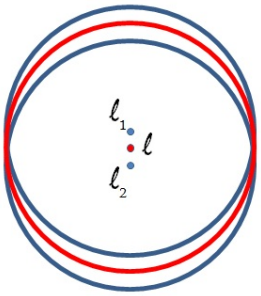

Figure 2: Two circles trying to cover the User Circle

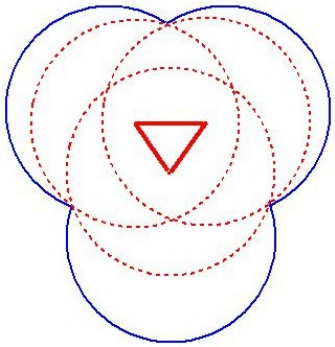

Figure 3: Triangular region indicating the position of the user

region to be a triangular region formed by the centers of the three red circles. The adversary will only be able to identify that the user should be located somewhere within this triangular region, but he cannot pinpoint the exact user location.

To make it even harder for the adversary, we can query *extra cover locations* (apart from the original cover locations) such that each additional location added would increase the number of probable cover location sets by at least one. One simple example can be seen in Figure 4, where adding one additional location would result in two possible cover location sets. This suggests that we can generalize our approach. Let k be the number of locations reported to the LBS. From the above discussion, it is clear that three locations would form the actual cover location set. The rest $k-3$ extra cover locations would result in *at least* $k-3$ additional cover location sets. Hence, reporting k locations would result in *at least* $k-2$ probable cover location sets – and therefore at least $k-2$ triangular regions where the user could possibly be present. The adversary will not be able to identify which of these triangular regions actually contains the user.

The above approach can be considered to be another flavor of *k-anonymity* based spatial cloaking [18, 12, 15] (reviewed in Section 2), where effectively the LBS provides the results for a region even though the user queries for few point locations. However, in contrast to the earlier spatial cloaking techniques, the entire processing happens on the user handset alone (involving few simple computations), without the need for any external infrastructure. This makes the new approach readily deployable by privacy-conscious users at their own discretion. Moreover, the new method enhances the anonymity further because the adversary can only identify regions where the user could possibly be, and not the actual locations.

4.2 Generating Locations

There could be many ways of forming the cover locations (such that the entire circle of radius r around the user location is covered), and the extra cover locations. We follow a simple approach which enables for randomizing each and every step of the procedure, so that any second invocation of the procedure for the same user location would result in a completely different set of surrounding locations being formed. For simplicity, we initially compute the points in a 2-dimensional space with the user location as the origin, but then map the points to the latitude-longitude format.

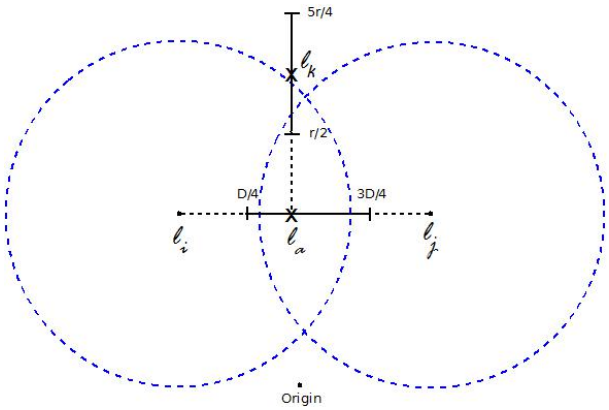

Figure 5: Generating extra cover locations

4.2.1 Generating Cover Locations

Given the user location is at the origin, we initially find the positions of the cover locations. If these cover locations form an *acute angled triangle*, they will be able to cover up the entire user circle in all cases. Moreover, having the origin at less than r distance from each of the cover locations would ensure that the entire circle of radius r around the origin is covered.

The first cover location (l_1) is chosen at a random distance of r_1 ($0 < r_1 < r$) from the origin and at a random angle of α_1 ($0^\circ < \alpha_1 < 120^\circ$) from the X-axis in the 2-dimensional space. The second cover location (l_2) is chosen at a random distance of r_2 ($0 < r_2 < r$) from the origin and at a random angle of α_2 ($\alpha_1+100^\circ < \alpha_2 < \alpha_1+140^\circ$) from the line joining l_1 and the origin. The third cover location l_3 is chosen at a random distance of r_3 ($0 < r_3 < r$) from the origin and at a random angle of α_3 ($\alpha_2 + 100^\circ < \alpha_3 < \alpha_2 + 140^\circ$) from the line joining l_2 and the origin. By ensuring these boundary conditions are met, we can say that the triangle formed by l_1, l_2 and l_3 is closely an acute angled triangle. Also, since all $\{r_1, r_2, r_3\} < r$, we can say that the circle of radius r around the origin is completely covered.

4.2.2 Generating Extra Cover Locations

Once the cover location points l_1, l_2 and l_3 are formed, we form pairs of points and store them in an array A. We randomly remove one pair l_i, l_j from A and use these points to generate another point l_k, such that l_i, l_j and l_k can form another acute angled triangle, and thereby another possible cover location set. Let the distance between l_i and l_j be D. We randomly choose one point l_a at distance of d_a ($D/4 < d_a < 3D/4$) from l_i on the line joining l_i and l_j. We choose another point l_k which is at a distance of d_k ($r/2 < d_k < 5r/4$) from l_a, such that the line joining l_a and l_k is perpendicular to the line joining l_i and l_j. Also, the point l_k must lie away from the origin, since we want the points to spread out as much as possible. The new point l_k will form an acute angled triangle with the points l_i and l_j, and hence these points can form another set of cover location points. Figure 5 gives a pictorial explanation of this procedure.

After generating l_k, we add (l_i, l_k) and (l_j, l_k) pairs to the array A for generating additional cover locations. Since we randomly choose one pair from A and use it to generate new points, we do not always spread out in one specific direction

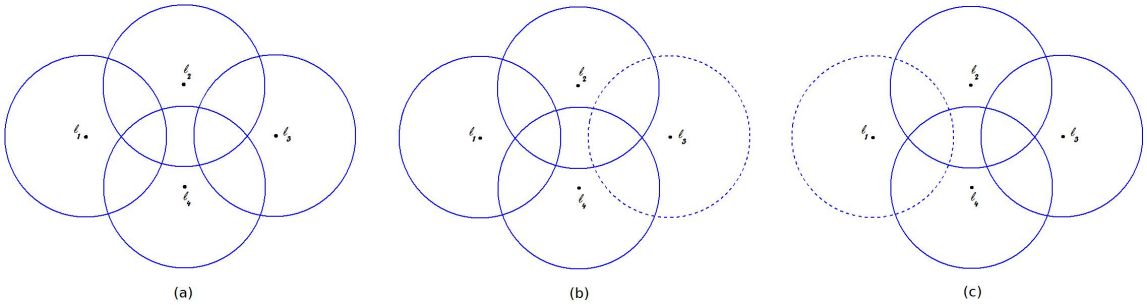

Figure 4: One additional point resulting in one additional cover location set

Algorithm 1 Generating Cover Locations

k = number of cover locations to be queried
r_1, r_2, r_3 = random numbers between 0 and r
α_1 = random angle between 0° and 120°
α_2 = random angle between $\alpha_1 + 100^\circ$ and $\alpha_1 + 140^\circ$
α_3 = random angle between $\alpha_2 + 100^\circ$ and $\alpha_2 + 140^\circ$
l_1 = point at a distance of r_1 from the origin and at an angle of α_1 from the X-axis
l_2 = point at a distance of r_2 from the origin and at an angle of α_2 from the line joining l_1 and the origin.
l_3 = point at a distance of r_3 from the origin and at an angle of α_3 from the line joining l_2 and the origin.
A = initialized with $[(l_1, l_2), (l_1, l_3), (l_2, l_3)]$
for $m = 1 \rightarrow k - 3$ **do**
  (l_i, l_j) = remove a random element from A
  D = distance between l_i and l_j
  d_a = random number between $D/4$ and $3D/4$
  l_a = point at a distance of d_a from l_i, on line joining l_i and l_j
  d_k = random number between $r/2$ and $5r/4$
  l_k = point at a distance of d_k from l_a, such that line joining l_a and l_k is perpendicular to line joining l_i and l_j
  $A = A \cup \{(l_i, l_k), (l_j, l_k)\}$
  $m \leftarrow m + 1$
end for

around the origin. This enables the additional feature that the cover locations could exist at the border of the reported locations or in the middle – allowing for additional confusion for the adversary. We repeat the procedure, until all the additional locations are generated. The algorithm for the generation of k cover locations is given in Algorithm 1.

4.2.3 Converting to Geographic Locations

Once the k surrounding cover locations are obtained in 2-dimensional space, we convert them into latitude-longitude pairs using simple transformations. Due to the small scale of the total area under consideration, we can treat the surface as a planar region neglecting the earth's curvature. However, the scales along the horizontal and vertical axes are not uniform. The distance corresponding to one degree across latitudes is not the same as the distance corresponding to one degree along the longitudes (1° latitude distance = 111.19km, 1° longitude distance = 94.29km). Given a user location in latitude-longitude form, we identify the offset of each of the k cover locations, along the horizontal and vertical axes, based on their values in the 2-dimensional space (since the user is assumed to be located at the origin). These offsets are then added to the user latitude and longitude, to generate the surrounding k cover locations to be sent to the LBS. We note that this simplistic mapping serves our purpose as we only require the total user circle to be covered, and do not require exact geographical mapping.

5. IMPLEMENTATION

We have developed a Java implementation of the Cover Locations technique. This implementation initially takes as an input the user location as a $< latitude, longitude >$ pair (available from the GPS device) and the value of k (the requested privacy level) from the user. Using Algorithm 1 with $r = 1$ km, we generate k locations in the 2-D space, which cover the results for the user located at the origin. These k locations are then converted to the latitude-longitude values. A distance of 1 km is equivalent to 0.008999 degrees across the latitudes and 0.011853 degrees along the longitudes. Hence, each of the k locations, represented as $< x_i, y_i >$, can be converted to $< lat_i, lon_i >$; where

$$lat_i = UserLatitude + y_i * 0.008999, \text{ and}$$

$$lon_i = UserLongitude + x_i * 0.011853.$$

In the above equations, having a minus sign before x_i and y_i would not make much difference except that the locations would be mirrored around the user location. Figure 6 and Figure 7 indicate the locations generated for different invocations of the algorithm with the same user location (Empire State Building, New York). These locations are categorized into three groups. Group1 (marked A) is the user Location, Group2 (marked B,C,D) is the cover location set and Group3 (the rest) is the extra cover location set.

It is interesting to note that, in Figure 7, even though location marker J lies on a water-body, the adversarial LBS cannot discard it. This is because by discarding J, the adversary will be missing at least one probable cover location set (C,F,J). Note that the user could actually be using the results of C, F and J to construct the results for his/her location, even though J points to water. Since the adversary knows that the user would not be located at any of the reported locations (given that the Cover Locations tool is being used), the adversary cannot discard any such improperly placed locations.

We interfaced this implementation with an open-source and free POI data service named SimpleGeo Places [4] (our LBS). This free service allows the user to send in a location

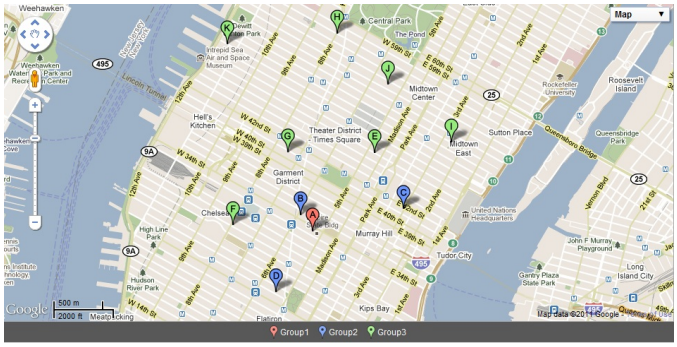

Figure 6: Surrounding Locations Set 1 (Red: user location; Blue: cover locations; Green: extra cover locations)

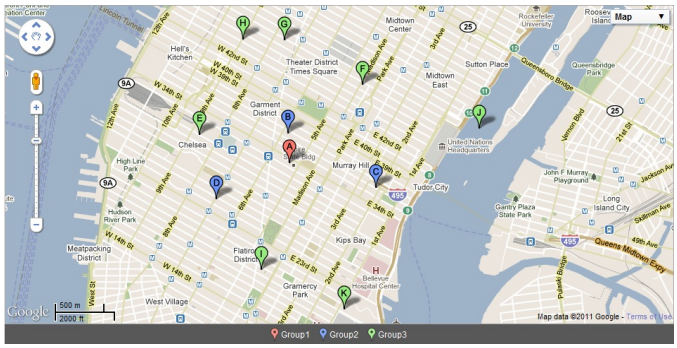

Figure 7: Surrounding Locations Set 2 (Red: user location; Blue: cover locations; Green: extra cover locations)

(in latitude-longitude form), an optional query (like Starbucks) and an optional category (like cafe) to obtain the nearby points of interest within a specified radius (in our case it is 1 km). We send k requests (one for each of the k locations generated) to the SimpleGeo service at the same time. Once the results are obtained, we filter them based on the condition that they should lie within a radius of 1 km around the user location, and sort the results before presenting them to the user such that the nearest ones appear first. This filtering has to be applied only to the results corresponding to the cover locations, and the results for the extra cover locations can be directly discarded. Also, this filtering is optional and we could just sort the results for the cover locations based on distance before presenting them to the user. If filtering is not applied, the user would be presented with more results compared to the case when he just searched with his actual location.

6. SECURITY EVALUATION

In this section, we evaluate the level of privacy provided to the user when using the Cover Locations approach. Lets assume k locations are reported to the LBS. The adversary knows that the user would be combining the results of three of these reported locations to construct the results corresponding to his location. Even if the adversary is able to identify (or guess correctly) the three cover locations, he would only be able to identify the triangular area which might contain the user. The adversary cannot obtain the current exact user location. So the privacy provided to the user is determined by two factors: the number of probable cover location sets formed by the k locations, and the total area of the region containing the user. In doing our analysis, we consider two types of attacker capabilities (as outlined in our threat model presented in Section 3). In one case, the adversary possesses only the k queried locations, and in the second case, the adversary is also equipped with a short-term location query history for the user.

6.1 Learning User Location Without User History

The area of the triangular region, containing the user, would inherently depend on the radius r, which in our case is 1 Km. Given a probable cover location set and the radius r, to determine the area of the approximate triangular region containing the user, the adversary needs to identify the vertices of this triangular region. These vertices are determined by identifying the extent to which the user circle can extend into each of the circles around the cover locations. Consider Figure 8; here we need to identify the center l_u of the User Circle that can extend maximum into the circle centered at l_1. If p_1, p_3 are the points of intersection of the circles centered at l_1 and l_2, and p_2, p_4 are the points of intersection of the circles centered at l_1 and l_3, then the maximum extending User Circle should pass through the points p_1 and p_4. We can identify p_1, p_3 and p_2, p_4 by solving the quadratic equations of the circles around l_1 and l_2, and l_1 and l_3, respectively. We can then determine the line equation passing through p_1 and p_4. This line would be the perpendicular bisector of the line passing between l_1 and l_u (since both

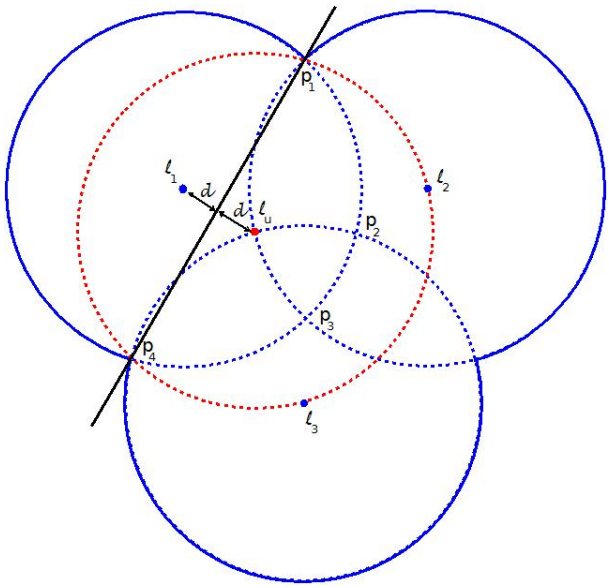

Figure 8: Identifying the vertices of a possible triangular region containing the user

the circles have the same radius). By calculating the distance d of l_1 from the line passing through p_1 and p_4, and identifying the equation of the line passing through l_1 and l_u, we can determine the center of the user circle l_u. We can repeat this procedure for each of the circles around l_2 and l_3, identify the vertices of the triangular region which contains the user location. Once the vertices are obtained, we can determine the area of the triangular region. The adversary can identify the area of the triangular region for each of the probable cover location sets, and the user could be located in any of the triangular regions with equal probability.

Based on Algorithm 1, the adversary knows that there are *at least* $k-2$ probable cover location sets – since initial 3 locations (l_1, l_2, and l_3) form one cover location set, and within the loop (in Algorithm 1) each extra cover location l_k can form at least one cover location set with l_i and l_j. In practice, the number of cover location sets could be larger than $k-2$. For example, consider the location set indicated in Figure 7. Since the cover locations form an acute angled triangle and do not contain any reported locations within the triangle, we obtain the following set of cover locations: {(B,C,D), (B,E,H), (B,E,G), (B,G,F), (B,C,F), (B,E,D), (B,E,I), (B,I,C), (C,F,J), (C,I,K), (E,H,F)}. We can clearly see that the size of the set is greater than 8 (since $k = 10$).

To determine how many cover location sets are generated (and the resulting total area of the triangular regions) in practice, we conducted the following experiment. For each value of k from 3 to 10 and a given user location l, we repeated the location generation algorithm 1000 times, and identified the number of cover location sets generated and the total area of the triangular regions. These results are indicated in Table 1. For each value of k, we report the lowest number of probable cover location sets, the average number of probable cover location sets formed, the average value of the total triangular area containing the user location, the maximum number of cover location sets and the maximum value of the total triangular area containing the user. From the table, we can clearly see that the average number of cover location sets is much higher compared to the lower bound on the number of cover location sets (which is nothing but $k-2$).

6.2 Learning User Location With User History

Prior research [29] showed that it is possible to identify the user patterns, when the user location history is available, and isolate/filter the noisy queries associated with the real queries with high probability using machine learning techniques. We set out to investigate how effective the Cover Locations approach is in protecting the user privacy when user location history is known to the adversary, besides just the posed queries (as studied in the previous subsection).

In order to pursue this analysis, we obtained the real mobility traces of 20 cabs in the San Francisco Bay area from the Cabspotting [10] service. This service tracks the location of the cabs using on-board GPS devices, which report the cab location to the server periodically. For simulation purposes, we treat each cab to be a user of the our Cover Locations technique. The cab data, we obtained, was spread over a period of 70 days. The first 45 days data was used as the cab location history available to the LBS adversary, which it can use to identify the cab location patterns. The next 25 days location data would be used for testing the Cover Locations technique. During the 25 days period, the LBS does not receive the actual location of each user query, but rather receives k (value of k chosen by the user/cab) cover locations generated using Algorithm 1. Having access to the user location patterns, we wanted to test if the LBS can identify the actual user locations with a higher probability, compared to the case when the user history was not available.

For each user, we generate cover locations using the new anonymization technique, for all the locations reported by the user during the 25 days period. For each set of k cover locations received by the LBS at an instant from a user (for a given user location query), the LBS tries to identify the number of probable cover location sets and the triangular regions containing the user. The user could be located in any of the triangular regions. The LBS can identify the centroids of these triangular regions as a rough estimate of the user location within these regions. Having obtained all the centroids, the LBS has to identify one centroid which could be the probable user location based on the user history. We consider the adversary to be successful, if he is able to identify the centroid closest to the user's current location. Let us call this special centroid as *User Centroid*. The adversary needs to identify these User centroids, generated by k cover locations for every user location query during the 25 days period.

We can formulate the above problem as a one class classification problem, where we have the training history for one class – 'C', and we need to identify the instances of this class from the mix of C and non-C class instances. In our case, the user history would be the user locations reported during the 45 day period, and we need to identify which of the calculated centroids might be the probable current user location based on the user history. One-class SVM classifier would be ideal for this scenario. In One-Class SVM, we try to fit a hypersphere (in a transformed space) around the

k	Lowest # of Cover Location Sets	Average # of Cover Location Sets	Average Value of Total Triangular Area (sq. km)	Highest # of Cover Location Sets	Highest Value of Total Triangular Area (sq. km)
3	1	1.00	0.0987	1	0.7151
4	2	3.19	0.2698	4	1.2116
5	3	5.99	0.4984	10	1.3307
6	4	9.51	0.7696	20	1.7067
7	5	13.21	1.0591	29	2.2830
8	6	17.41	1.3628	42	3.7206
9	7	22.37	1.7125	56	4.1627
10	8	27.57	2.0547	67	4.4620

Table 1: The level of privacy provided by Cover Locations

training data [11]. Any test data instance lying within the hypersphere would be labelled as an instance of class C. The test data instances lying outside the hypersphere would be labelled as not belonging to class C. We used the One-Class SVM implementation provided in [11] for our purpose.

For each user, the One-Class SVM classifier would classify and label each test data instance (centroid) as belonging to user class or non-user class. This would generate two subsets (user and non-user) of the test data instances. We consider the classification techniques to be successful if the probability of identifying a user centroid from the user subset is significantly higher compared to the probability of identifying a user centroid from the total test data set.

Additionally, we wanted to test how the classifier performance would vary as the value of k increases. Hence, for each of the 20 users, we try to identify the classifier performance for $k = 5, 6, 7, 8, 9$ and 10. The results of our analysis are indicated in Table 2. For each value of k, we indicate the average number of User centroids and average number of Non-User centroids corresponding to different sets (the Total Test Set, the User Set and the Non-User Set), aggregated across all the 20 users. We can observe that for all values of k, the number of Non-User centroids (T_n) are more than k times the User Centroids (T_u).

Since the distribution of centroids in the test data set was varying for different values of k, we determined the optimized parameters for the classifier, once for each value of k. Hence, the number of User centroids in the User Set (U_u) was varying for different values of k. Though the classifier was able to identify a large fraction of the user centroids, many of the non-user centroids were also falsely classified into the User Set, resulting in no improvement in the probability of identifying User Centroids. In fact, the classification slightly reduced the random probabilities of identifying the User centroids. This can be observed from the last column of the Table 2.

These classifier results show that having access to user history does not provide the adversary with any advantage in breaking the privacy offered by Cover Locations. Such classifications are not successful against the new anonymization technique because even the cover locations lie close to the actual user locations and are realistic. In other words, Cover Locations indirectly take into consideration the user query patterns (as recommended in the work of [29]), since it generates locations close to the real locations, at times the user visited those locations. Therefore, we can conclude that the Cover Locations provide an acceptable level of privacy even when the adversary has access to the user location history.

7. STRENGTHS AND LIMITATIONS

The Cover Locations technique only requires few changes on the user's client, without any need for additional third-party infrastructure or server-side modifications. This allows the solution to be immediately deployable by privacy-conscious users. There are various prior client-based solutions, which either send many fake queries along with the real location query [25, 30, 23] or periodically pre-fetch the location information from the LBS to a local database and answer the user location queries from the local database [6, 5]. The former set of solutions need to take care to prevent any correlation attacks across trips while generating the accompanying fake locations. Moreover, to generate realistic looking fake trips, they need additional out of band information, such as the traffic densities for a particular region or the road network information. For the latter method, the user needs to indicate the region that he might visit in advance to be able to download the Location information for that region. Pre-fetching everything in advance might not always be possible, and even if the user is visiting few locations within a region, the solution has to download a lot of information.

In contrast to the above solutions, Cover Locations does not require any out of band information or need to maintain any local database. It just needs to solve few straightforward mathematical equations to be able to generate the cover locations. This computation can be done in real time. Moreover, in reconstructing the response, the user only needs to filter the results corresponding to three cover locations, irrespective of the number of locations queried (value of k). Since the user location does not leave the user's device as either a latitude, longitude pair or as an obfuscated region, the user can be confident about the level of privacy provided. Based on the analysis shown in Section 6, the adversary can identify at least $k - 2$ probable cover location sets, and even if he is able to guess the real cover location set, he can only identify a region in which the user could be located, but not the actual user location. In addition, even having access to user history does not provide the adversary with any advantage. Thus, taking into consideration the resource constraints on smart phones and other hand-held devices, our solution offers reasonable level of privacy at the cost of minimal computation.

Using the Cover Locations technique, the user should be

Value of k	Total Test Data		User Set		Non-User Set		Random Probability in Total Test Set $\frac{T_u}{T_u+T_n}$	Random Probability in User Set $\frac{U_u}{U_u+U_n}$
	Average # of User Centroids (T_u)	Average # of Non-User Centroids (T_n)	Average # of User Centroids (U_u)	Average # of Non-User Centroids (U_n)	Average # of User Centroids (N_u)	Average # of Non-User Centroids (N_n)		
5	3166.15	16596.45	2579.35	13339.70	586.80	3256.75	0.191	0.162
6	3166.25	27767.05	2835.25	24555.80	331.00	3211.25	0.114	0.104
7	3166.25	40377.65	2920.75	36734.05	245.50	3643.60	0.078	0.074
8	3166.25	54588.55	2976.35	50602.90	189.90	3985.65	0.058	0.056
9	3166.25	70279.80	3019.90	66254.95	146.35	4024.85	0.045	0.044
10	3166.25	87660.70	3048.80	83532.00	117.45	4128.70	0.036	0.035

Table 2: Performance of the classifier in identifying User Centroids

able to obtain the same set of results (if not more, which happens when the results are not filtered by the user) when he queries for k cover locations and when he queries directly for his actual location. This is because the entire User Circle is covered by the circles of radius 1 Km around the cover locations. However, in practice, we found that the results do not always match. This is because the SimpleGeo service restricts the results to the nearest 25 POIs when the result set is large. This happens only in cases when the user sends in the latitude-longitude location alone and does not provide any search query term or a category to tailor the results. In those cases, using the Cover Locations technique, the user would obtain a different set of nearby POI which are within a radius of 1 kilometer but not necessarily all the nearest 25 POI results. This problem is specific to the targeted LBS's policies, and need not be a general issue if the LBS does not restrict the results to any fixed number.

If the user needs to periodically query nearby POIs (using the proposed Cover Locations) while he is on a trip, then it might be possible for the adversary to correlate the reported k locations and eliminate few probable cover location sets. One simple solution to prevent correlation attacks would be to use the same offsets for generating the surrounding cover locations throughout the trip. The user could indicate on his device if he is performing a one time query or if this query is part of a series of queries corresponding to a trip. The user can easily switch between different modes in real time. The proposed Cover Locations approach is applicable only to the class of LBSs which take a location as an input and return the nearby POIs. Currently, it is not clear how this approach could be extended to other classes of LBS such as the ones which help in generating navigation paths.

8. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed Cover Locations, a new approach to location privacy which allows the users to access the Location-Based Services without revealing their actual location. Based on his current location, the user queries for few specifically chosen surrounding locations and constructs the results for his location from the results of the surrounding locations. We have shown how to effectively generate these surrounding locations with very few mathematical computations. We evaluated the level of privacy provided by the current solution in terms of the number of cover locations queried and the total area of the triangular regions resolved by the adversary. We also demonstrated that the Cover Locations technique would protect the user privacy even when the user location history is available to the adversary.

The proposed solution is a simple and efficient client-based privacy preserving approach. However, currently it can be used only for a particular category of Location-Based Services – which provide nearby points of interest based on the user location. In our future work, we plan to extend this architecture for navigation systems. Here, the exact source and destination of the user trip are hidden from the navigation service, but based on the paths generated for specifically chosen user trips, we plan to construct the efficient navigation path for the actual user trip. This architecture can also be useful in the context of web search privacy. The user can translate his web search query into a few cover queries and will generate the response of the former from the responses corresponding to the latter. We intend to explore how such cover search queries can be generated and how this approach will enhance the user privacy.

Acknowledgments

We are grateful to the Cabspotting service for providing us with the cab traces to conduct part of our evaluation of Cover Locations.

9. REFERENCES

[1] Geocaching. `http://www.geocaching.com/`.
[2] Google latitude. `http://www.google.com/latitude`.
[3] Google maps navigation. `http://www.google.com/mobile/navigation/`.
[4] Simplegeo places. `https://simplegeo.com/products/places`.
[5] S. Amini, J. Lindqvist, J. I. Hong, J. Lin, E. Toch, and N. Sadeh. Cache: caching location-enhanced content to improve user privacy. In *MobiSys 2011: the 9th International Conference on Mobile Systems, Applications and Services*, 2011.
[6] S. Amini, J. Lindqvist, J. I. Hong, M. Mou, R. Raheja, J. Lin, N. Sadeh, and E. Tochb. Cache: caching location-enhanced content to improve user privacy. *SIGMOBILE Mob. Comput. Commun. Rev.*, 14:19–21, December 2010.
[7] C. A. Ardagna, M. Cremonini, S. D. C. di Vimercati, and P. Samarati. An obfuscation-based approach for

protecting location privacy. *IEEE Transactions on Dependable and Secure Computing*, 8:13–27, 2011.

[8] AroundMe. `http://www.tweakersoft.com/mobile/aroundme.html`.

[9] N. Bilton. Nytimes: Tracking file found in iphones. `http://www.nytimes.com/2011/04/21/business/21data.html`.

[10] Cabspoting. `http://cabspotting.org/`.

[11] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[12] C.-Y. Chow, M. Mokbel, and X. Liu. Spatial cloaking for anonymous location-based services in mobile peer-to-peer environments. *GeoInformatica*, pages 1–30, 2009.

[13] C.-Y. Chow, M. F. Mokbel, and X. Liu. A peer-to-peer spatial cloaking algorithm for anonymous location-based service. In *ACM international symposium on Advances in geographic information systems*, 2006.

[14] M. Duckham and L. Kulik. A formal model of obfuscation and negotiation for location privacy. In *Pervasive Computing*, pages 152–170. 2005.

[15] B. Gedik and L. Liu. Protecting location privacy with personalized k-anonymity: Architecture and algorithms. *IEEE Transactions on Mobile Computing*, 7:1–18, 2008.

[16] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K.-L. Tan. Private queries in location based services: anonymizers are not necessary. In *SIGMOD international conference on Management of data*, 2008.

[17] G. Ghinita, P. Kalnis, and S. Skiadopoulos. Mobihide: a mobilea peer-to-peer system for anonymous location-based queries. In *Conference on Advances in spatial and temporal databases*, 2007.

[18] M. Gruteser and D. Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *Conference on Mobile systems, applications and services*, pages 31–42, 2003.

[19] M. Gruteser and B. Hoh. On the anonymity of periodic location samples. In *Security in Pervasive Computing*, volume 3450, pages 179–192, 2005.

[20] U. Hengartner. Hiding location information from location-based services. In *Mobile Data Management, 2007 International Conference on*, pages 268–272, May 2007.

[21] B. Hoh, M. Gruteser, H. Xiong, and A. Alrabady. Enhancing security and privacy in traffic-monitoring systems. *IEEE Pervasive Computing*, 5:38–46, 2006.

[22] A. Khoshgozaran, H. Shirani-Mehr, and C. Shahabi. Spiral: A scalable private information retrieval approach to location privacy. In *Mobile Data Management Workshop*, pages 55–62, Apri 2008.

[23] H. Kido, Y. Yanagisawa, and T. Satoh. Protection of location privacy using dummies for location-based services. *Data Engineering Workshops, 22nd International Conference*, 2005.

[24] J. Krumm. Inference attacks on location tracks. In *Proceedings of the 5th international conference on Pervasive computing*, PERVASIVE'07, pages 127–143, 2007.

[25] J. Krumm. Realistic driving trips for location privacy. In *Pervasive Computing*. 2009.

[26] Neilsen. Smartphone users do care about location privacy. Available at: `http://www.techgearx.com/nielsen-smartphone-users-do-care-about-location-privacy/`.

[27] S. T. Peddinti and N. Saxena. On the privacy of web search based on query obfuscation: A case study of trackmenot. In *Privacy Enhancing Technologies Symposium (PETS)*, 2010.

[28] S. T. Peddinti and N. Saxena. On the effectiveness of anonymizing networks for web search privacy. In *ACM Symposium on Information, Computer and Communications Security (AsiaCCS)*, 2011.

[29] S. T. Peddinti and N. Saxena. On the limitations of query obfuscation techniques for location privacy. In *To appear in International conference on Ubiquitous computing*, 2011.

[30] P. Shankar, V. Ganapathy, and L. Iftode. Privately querying location-based services with sybilquery. In *International conference on Ubiquitous computing*, 2009.

[31] R. Shokri, G. Theodorakopoulos, J.-Y. Le Boudec, and J.-P. Hubaux. Quantifying Location Privacy. In *IEEE Symposium on Security and Privacy (S&P)*. University of California, Irvine, 2011.

[32] R. Shokri, C. Troncoso, C. Diaz, J. Freudiger, and J.-P. Hubaux. Unraveling an old cloak: k-anonymity for location privacy. In *Proceedings of the 9th annual ACM workshop on Privacy in the electronic society*, WPES '10, pages 115–118, 2010.