Location Obfuscation for Privacy in Navigation Applications

Vaibhav Ankush Kachore, J. Lakshmi, S. K. Nandy
Supercomputer Education and Research Centre
Indian Institute of Science, Bangalore, India
vaibhav_kachore@ssl.serc.iisc.in, jlakshmi@serc.iisc.ernet.in, nandy@serc.iisc.ernet.in

Abstract—**Advances in wireless internet, sensor technologies, mobile technologies, and global positioning technologies have renewed interest in location based services (LBSs) among mobile users. LBSs on smartphones allow consumers to locate nearby products and services, in exchange of their location information. Precision of location data helps for accurate query processing of LBSs but it may lead to severe security violations and several privacy threats, as intruders can easily determine user's common paths or actual locations. Encryption is the most explored approach for ensuring security. It can give protection against third party attacks but it cannot provide protection against privacy threats on the server which can still obtain user location and use it for malicious purposes. Location obfuscation is a technique to protect user privacy by altering the location of the users while preserving capability of server to compute few mathematical functions which are useful for the user over the obfuscated location information. This work is divided into two parts. First part mainly concentrates on LBSs which wants to know the distance travelled by user for providing their services. This study proposes various methods of location obfuscation for GPS location data which are used to obfuscate user's path and location from service provider. Second part handles security issue in another type of LBS i.e. navigation application. The intuition behind the novel architecture is to get trusted service by making use of different independent untrusted services. The proposed architecture provides user privacy in navigation applications by using neighbourhood finding LBS and an anonymizer. This architecture makes use of novel protocol which provides protection against various attacks. Our work shows that user privacy can be maintained without affecting LBSs results, and without incurring significant overheads.**

Keywords: Location data protection; Location Based Services; Path Obfuscation Techniques; User Privacy; Navigation Applications.

I. Introduction

Location based services (LBSs) are services that are associated with processing of information in and around a specific location. The location of interest is mostly that of a person who is asking for information. Popular examples of LBSs are seeking navigational information from current location to that of a destination, knowing good eating places in and around a given location, seeking information on emergency services like hospitals or police stations nearest to that specific location, fitness applications which keep track of daily workout of user by using its global positioning system (GPS) information. Real world examples of LBSs are Local search and discovery applications such as Foursquare [1], Zomato [2], fitness applications such as RunKeeper [3], SportsTracker [4], Runtastic [5], etc.

Although LBSs are ubiquitous and provide convenience, they are threat to the privacy and security of their users [6]. The privacy threat comes from the fact that LBS providers rely mainly on an implicit assumption that users agree to share their private locations to get services. In other words, a user trades her privacy when availing the service. With untrustworthy servers, such a model poses several privacy threats. For example, an employer may check on her employee's behaviour by knowing the places where he visits and the time of each visit, the personal medical records can be inferred by knowing which clinic a person visits, or someone can track the locations of his acquaintances. In fact, in many cases, GPS devices have been used in stalking personal locations [7], [8].

Encryption is the most explored approach for ensuring security. Generally, to provide user privacy, encryption techniques are used. Obfuscation is another way to achieve the same purpose. Obfuscation is a technique to protect user privacy by altering the location of the users while preserving capability of server to compute few mathematical functions which are useful for the user over the obfuscated location information. So, for a user who wants to preserve his privacy, there can be two modes in which LBSs operate:

- Encrypted Service
- Obfuscated Service

In Encrypted Service mode, user location information is encrypted and then it is sent to LBSs server. Encryption and decryption of information is an overhead in this mode. These overheads contribute to latencies in the service response and can potentially reduce the utility of the service.

In Obfuscated Service mode, actual location information is not sent to LBSs server. Transformed (obfuscated) information is sent to server which is useful enough for functionality of LBS.

So, in this paper obfuscation techniques are proposed and evaluated. After that comparison between encryption and obfuscation techniques is made in terms of latency and it is concluded that obfuscation technique provide user privacy without much overhead.

The rest of the paper is organised as follows. Motivation and problem statement which is handled in this paper in described in Section 2. Section 3 highlights related work to this paper. Attacker model is explained in Section 4. Section 5 describes definitions and notations used. Obfuscation functions for user privacy are proposed in Section 6. Novel architecture for ensuring user privacy in navigation application is proposed in Section 7. Experiments and results are in shown Section 8. Section 9 provides conclusion and future work.

II. Motivation

Many enterprises are considering to purchase geo-location data from LBS, and use them to analyse potential customer preferences. So, the information with LBS is of monetary significance [9]. This makes location information more vulnerable to intruder attacks.

Though many location-based applications are successfully running, there is still a great concern about privacy in terms of the user's location. A contradiction is present in this situation: users want to receive satisfactory service, but they do not want the service provider to either know or preserve their exact location. This contradiction has triggered the appearance of techniques like location obfuscation [10].

Many of LBSs do not need to know the exact position of user, but they just need to calculate some mathematical function of the location information, such as the distance that has been covered by the tracked user or object, or the average time to move from a point to another point. Fitness applications are generally used to track user workouts in about any form of distance-based outdoor fitness endeavour and analyse the results. Another example of LBSs which want to know distance travelled by user is pay as-you-go insurance. These applications don't need exact user location coordinates for their functionality. So, there is a scope for improving user privacy in such applications. This is the motivation behind the proposed research work. The paper gives solution for user privacy in such applications. Obfuscation functions proposed in this paper converts original location data into obfuscated location data but preserves distance between location data points.

In this paper, two kinds of obfuscation functions are proposed:

- Non reversible obfuscation functions
- Reversible obfuscation functions

Non reversible obfuscation functions convert original path of user into obfuscated path in non reversible manner, i.e. it is not possible to get back original path from obfuscated path. Reversible obfuscation functions also transform original path of user into obfuscated path, but the transformation is reversible, i.e. it is possible to get back original path from obfuscated path. So, depending on user privacy needed and functionality of application any of these obfuscation functions can be used.

In this paper, 3 obfuscation functions are proposed, namely Ellipsoidal Random Obfuscation Function (EROF), Modified Random Obfuscation Function (MROF) and Grid Obfuscation Function (GOF). EROF falls in first category of obfuscation functions (i.e. non reversible obfuscation functions). This kind of obfuscation functions can be used when user's actual path is not of any significance and only distance travelled is of importance. EROF is a new obfuscation technique proposed in this paper. Whereas, MROF and GOF are of second kind of obfuscation functions. These kinds of obfuscation technique are used when user's actual path can be required in future. MROF and GOF are extension of work by [9]. These two techniques are reversible. So, any of these functions can be used for obfuscation depending on requirements of applications.

Later part of this paper proposes a novel architecture for preserving user privacy in navigation applications.

III. Related Work

Extensive research has been done for providing user privacy in LBSs. Kido et al. proposes a technique in which, for every location update, a user sends n different locations to the server with only one of them being true while the rest are dummies. Thus, the server cannot know which one of these locations is the actual one [11].

In [13] , D. Riboni et al. propose adding random noise to the data, or rounding the location based on a predefined set of landmarks or grid cells. This technique provides user privacy but it trade off accuracy of LBSs.

Marco Gruteser et al. presents a middleware architecture using anonymizer and algorithms that adjust the resolution of location information along spatial or temporal dimensions to meet specified anonymity constraints based on the entities who may be using location services within a given area. The main idea is to blur a user's exact location into a spatial area using either spatial or temporal cloaking [14]. Other works based on similar kind of techniques are described in [15], [16], [17], [18].

A lot of research has been done on giving user privacy with pseudonym. Pseudonym is a technique in which the real identity of user is replaced by fake identity [19]. In this approach, true location of user is sent to LBS, but true location information of user can lead to his true identity. Attacker can get true identity of user from this location information with the help of a public telephone directory which contains subscribers addresses [20], [8].

Researchers have tried to measure the distance travelled by a user using accelerometer and gyroscope sensors. This solution can provide user privacy. But, due to inherent accelerometer bias drift and inaccuracy of these sensors, the distance measured by using these sensors will not be accurate which will degrade performance of LBS [21], [22].

In another paper, Krishna P.N. Puttaswamy et al. introduces LocX, an alternative that provides significantly improved location privacy. In LocX, the friends of a user share their user's secrets with each other so they can apply the same transformation. This allows all location queries to be evaluated correctly by the server, but their privacy mechanisms guarantee that servers are unable to see or infer the actual location data from the transformed data or from the data access [23]. But, none of the above technique can be directly used for applications that we are targeting.

Wightman et al. [10] propose a location obfuscation technique, named Matlock. It is based on matrix obfuscation, it realizes location obfuscation in both spatial and temporal dimensions with low computational cost. But, in order to get complete user privacy, this method needs to be used with some obfuscation method in order to include a non-reversible layer that will protect location information.

Roberto Di Pietro et al. [9] formalizes the concept of obfuscation function, and proposes a solution that provides user privacy while allowing users to continue leveraging the services offered by the service provider. In [9], only reversible obfuscation techniques are proposed. Few methods described here are extension to their work. Moreover, in subsequent sections, novel irreversible obfuscation techniques will also be proposed.

It can be concluded from these papers that obfuscation techniques seem to be highly application dependent and hence choice of appropriate obfuscation technique is closely related to the application requirement. Hence, extensive research is going on in the field of obfuscation.

IV. Attacker Model

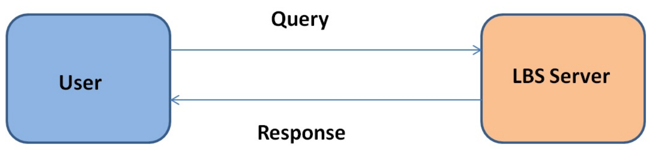


Fig. 1: User - Server Model

Figure 1 shows the user who is using a particular kind of LBS. Figure 1 shows that when a user queries to LBS server, a response comes from LBS server depending on functionality of LBS and a type of query. There are 2 kinds of attacks can occur in this model.

1) Attackers can spoof a link between user and LBS server. If this happens, attacker can track user and use this information for malicious purposes. This can be avoided if encrypted data is sent to server. Encryption will ensure that even if link between user and server is compromised by attacker, he will not be in position to get meaningful information out of that, provided encryption scheme is sufficiently strong.
2) In worst case, LBS server itself can be an attacker. Encryption cannot provide user privacy in such cases because server will get real location information after decryption. In obfuscation techniques which are proposed in this paper, actual location information of user is not sent to server. Obfuscation techniques which are proposed in this paper can provide user privacy even in such cases.

In this paper, we are considering both kinds of attacks. In subsequent sections, details about the proposed obfuscation schemes and how they provide user privacy will be explained.

V. Definitions and Notations

In this section, the definitions and notations used in subsequent sections are explained. These notations will be directly used in algorithms proposed in this paper.

- GPS=$\{(y,x) \in R^2 : -90 \leq y < 90, -180 \leq x < 180\}$; We denote an element "$p$" $\in GPS$ by the point, which is composed of latitude and longitude of a GPS coordinate.
- Let, $p = (y,x) \in GPS, t = (tLat, tLon) \in Q^2$; $w = p + t \ (mod\ GPS)$ means adding $\pm$(180, 0) and/or $\pm(0, 360)$ until $w \in GPS$. ($\because a \ (mod\ n)$ returns the congruent number of a in the $\{0, 1, ..., n-1\}$)
- The path is the ordered set of points recorded by a GPS-equipped device. A path is a n-tuple P = $(z_0, ..., z_{n-1})$, such that $z_i \in GPS$ for i $\in \{0, ..., n-1\}$.
- $C = \{C_0, C_1, \ldots\}$ are random number chains generated by Merkle tree [24] based approach which are one of the safest random number chains because it is computationally infeasible to calculate other random numbers from knowledge of few random numbers of C. This ensures backward security, forward security and the impossibility of collusion [9].
- $d(x,y)$ represents distance between point x and point y.
- $tLat$ and $tLon$ are latitude and longitude of initial point of obfuscated path which is chosen in such a way that it can be feasible location of user.

VI. Location Data Obfuscation Functions

There are two kinds of path obfuscating functions as described earlier. First kind of functions make use of non reversible algorithms to provide user security. Second approach is orthogonal to the first approach. These kinds of obfuscation functions make use of Merkle tree [24] based random number chain. Note that these methods are reversible but due to usage of safe random number chains [9], it provides security against various intruder attacks.

A. Ellipsoidal Random Obfuscation Function(EROF)

EROF is an irreversible obfuscation algorithm which will restrict intruder to get back real path from obfuscated path. The intuition of this method comes from two facts. First fact is multiplication of a vector by an orthonormal matrix preserves its 2-norm and second fact is sum of distance of any point from two foci of an ellipse is constant. Combination of these two facts have resulted in Algorithm 1.

This algorithm is two step algorithm. Orthonormal transformation is applied in the first step. Whereas, ellipsoidal transformation is applied in second step. Detailed implementation is shown in Algorithm 1. Here algorithm is explained with example in Fig. 2.

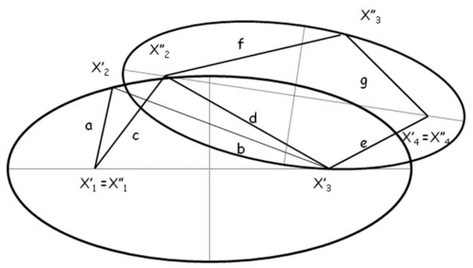


Fig. 2: Ellipsoidal Random Obfuscation Function Example

Let, Q be any orthonormal matrix. $(Q * Q^T = Q^T * Q = I)$

Let, R and O_1 be a real (original) path and obfuscated path 1 respectively.

Let, $(X_1, X_2, \ldots)$ be points on path R, where $X_i = [y_i, x_i]$ (Here y_i and x_i are latitude and longitude of point X_i which is on path R.)

Let, $(X'_1, X'_2, \ldots)$ be points on path O_1, where $X'_i = [y'_i, x'_i]$

Let, DR and DO_1 be distance travelled on path R and path O_1 respectively.

Transformation must satisfy the condition $DR = DO_1$. So, following transformation is done.

$$X'_i = Q \times X_i \tag{1}$$

Now, multiplication of a vector by orthonormal matrix does not change its 2-norm. Therefore,

$$| X'_i |_2 = | Q \times Xi |_2 \tag{2}$$

$$\begin{aligned} | X'_{i+1} - X'_i |_2 &= | Q \times X_{i+1} - Q \times X_i |_2 \\ &= | Q \times (X_{i+1} - X_i) |_2 \\ &= | X_{i+1} - X_i |_2 \end{aligned} \tag{3}$$

This implies that distance relationship between any pair of points is preserved in this transformation.

Let, "O_2" be an obfuscated path 2.

Let, $(X''_1, X''_2, \ldots)$ be points on path O_2, where $Xi = [y''_i, x''_i]$

Let, "DO_2" be distance travelled on path O_2.

Applying ellipsoidal transformation on path O_1,

Ellipse has property that sum of distances of any point on the ellipse from it's foci is constant.

In example of fig.2, path travelled by O_1 is [$X'_1 \rightarrow X'_2 \rightarrow X'_3 \rightarrow X'_4$] and path travelled by O_2 is [$X''_1 \rightarrow X''_2 \rightarrow X''_3 \rightarrow X''_4$].

$$DO_1 = a + b + e \tag{4}$$

$$DO_2 = c + f + g \tag{5}$$

Consider an ellipse with X_1 and X_3 as foci.

Using property of ellipse,

$$a + b = c + d \tag{6}$$

Similarly, for an ellipse with X_2 and X_4 as foci,

$$d + e = f + g \tag{7}$$

From (4), (5), (6) and (7),

$$DO_1 = c + d + e = c + f + g = DO_2 \tag{8}$$

Hence, distance travelled on path O_1 and O_2 is same. Note that points on path O_2 are sent to LBSs in place of points on real path R to achieve obfuscation.

Proof of irreversibility of EROF:

For calculating equation of ellipse whose major axis is at some angle with respect to X - axis , 3 points are needed because 3 parameters of the ellipse i.e. semi major axis "a", semi minor axis "b" and angle θ which its major axis is making with X - axis are unknowns. Now, to find path O_1 from O_2, if any 3 consecutive points are chosen on path O_2, then equation of ellipse can be found but it is not possible to know which point on this ellipse was there in path O_1. ($\because$ All points will satisfy distance criteria.) Hence, EROF is irreversible.

Note that in algorithm 2, $x_1, x_2, x_3, temp$ are 2-D vectors. 1st and 2nd component of 2-D vector v is represented by $v(1)$ and $v(2)$ respectively.

Algorithm 1 $EROF_c(z_0, ..., z_{n-1}) = (w_0, ..., w_{n-1})$

```
1:  $t = (tLat, tLon)$
2:  $w_0 = t \ (mod\ GPS)$
3:  Initialize random orthonormal matrix "Q".
4:  $x_1 = w_0$
5:  $x_2 = Q \times (z_1 - z_0)$
6:  $x_3 = Q \times (z_2 - z_0)$
7:  **for** $i = 1 \rightarrow n - 2$ **do**
8:      $d_1 = d(x_1, x_2)$
9:      $d_2 = d(x_2, x_3)$
10:     $d_3 = d(x_1, x_3)$
11:     $a = \frac{d_1 + d_2}{2}$
12:     $x_c = \frac{x_1 + x_3}{2}$
13:     $cos(\phi) = \frac{x_3(1) - x_1(1)}{d_3}$
14:     $sin(\phi) = \frac{x_3(2) - x_1(2)}{d_3}$
15:     $\theta = \begin{bmatrix} cos(\phi) & -sin(\phi) \\ sin(\phi) & cos(\phi) \end{bmatrix}$
16:     $temp = \theta^T \times (x_2 - x_c)$
17:     $t = acos(\frac{temp(1)}{a})$
18:     $b = \frac{temp(2)}{sin(t)}$
19:     $x_1 = x_c + \theta \times \begin{bmatrix} a \times cos(C_{i+1}) \\ b \times sin(C_{i+1}) \end{bmatrix} \ (mod\ GPS)$
20:     $w_i = x_1 + t \ (mod\ GPS)$
21:     $x_2 = x_3$
22:     $x_3 = Q \times (z_{i+1} - z_0)$
23: **end for**
24: $w_{n-1} = x_3 + t \ (mod\ GPS)$
```

B. Modified Random Obfuscation Function(MROF)

This method is extension of work done in [9]. First method proposed in [9] is Random Obfuscation Function(ROF). In ROF, the first point is translated by a translation vector that depends on the chain C, while the other points are roto-translated in the following way: the i-th obfuscated point is placed according to the i-th angle (calculated by using the hash value C_{i+1}), and the circle with center the $(i-1)$-th obfuscated point and radius $|z_i - z_{i-1}|$, where z_{i-1}, z_i are points that belong to the original path.

The problem with this approach is that, the algorithm assumes that server is not checking whether a point (latitude, longitude) is a feasible point or not. As algorithm is choosing any random point to initialize the obfuscated path, it may happen that the path gets initialized from the point which is not feasible, e.g., the selected point might be in sea. There can be LBSs which check authenticity of location information prior to processing user's request. Such LBSs server will stop giving its services, if server is sure that user is doing something from its side (like obfuscating actual location of user) for its privacy. MROF can also handle such situation.

Consider a region having very high density of roads. In this situation, if sometimes user goes out of road (on obfuscated path), still server cannot be sure of the fact that user is changing its actual coordinates and sending obfuscated coordinates. This fact is justified because global positioning system (GPS) has inaccuracies of around 5-10 meters in many cases [25]. So, in such situation, ROF can also be used if trajectory of obfuscated path can be bound to the region having high road densities. In Algorithm 2, this bounding region is denoted by B and list L contains random numbers C_j which are not used for obfuscation. List L will be used for reverse transformation i.e. to convert obfuscated path back to original path. Implementation details of MROF are explained in Algorithm 2.

Algorithm 2 $MROF_c(z_0, ..., z_{n-1}, B) = (w_0, ..., w_{n-1}, L)$

```
$t = (tLat, tLon)$
$w_0 = z_0 + t \ (mod\ GPS)$
$j = 2$
for $i = 1 \rightarrow n-1$ do
    while $(true)$ do
        $\theta_j = \frac{360}{2^{m+1}} \times (2^m + C_j)$
        $w_i = w_{i1} + R_{\theta_j} \times (z_i - z_{i-1}) \ (mod\ GPS)$
        if $w_i \in B$ then
            $j = j + 1$
            break
        else
            Put $C_j$ in list L.
            $j = j + 1$
        end if
    end while
end for
```

C. Grid Obfuscation Function (GOF)

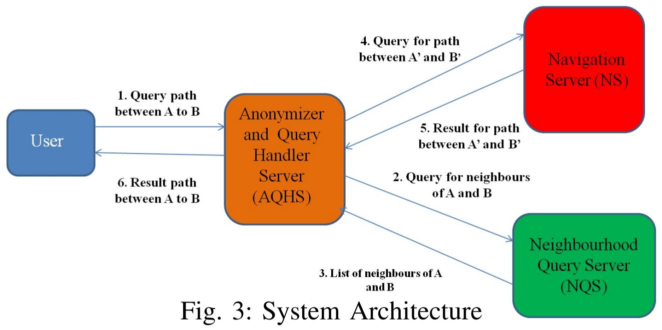
Linear Obfuscation Function (LOF) was proposed by [9]. In LOF, the user chooses two points p_1, $p_2 \in$ GPS making sure that the linear path between them is a realistic path. A possible choice can be a fairly straight strip of beach. The obfuscated path will be a sequence of points that goes forward and backwards from p_1 to p_2. The problem with LOF is that it is moving a point in obfuscated path on same path again and again. Hence, server may eventually come to know that user is obfuscating its actual coordinates.

In GOF, off-line map is used for deciding the feasible direction of travel. Obfuscated path will start on any point which is on the road. It will move along the road and as soon as it reaches junction, it will randomly choose any road which is meeting at the junction. Algorithm 3 make sure that obfuscated path should not go beyond certain region by doing explicit checking. If obfuscated path is not restricted, then processing very large spatial data (map) will be required. .

By increasing size of bounded region, probability of detecting the fact that user is obfuscating its original coordinates by server can be reduced. But, the amount of space on mobile devices is limited, and hence keeping balance of both factors is important. So, by careful formulation, algorithm ensures security and avoids processing huge amount of map data.

Algorithm 3 $GOF_c(z_0, ..., z_{n-1}, B) = (w_0, ..., w_{n-1})$

```
$t = (tLat, tLon)$ = randomly choose any point which is
on actual road in map.
$w_0 = t$
Choose direction to move and start journey.
for $i = 1 \rightarrow n-1$ do
    while travelling distance $z_i - z_{i-1}$ do
        if Divergence is encountered then
            repeat
                Choose any road randomly out of roads
                which are meeting at that junction.
                Update $w_i$ appropriately.
            until $w_i \in B$
        end if
    end while
end for
```


User
1. Query path between A to B
6. Result path between A to B
Anonymizer and Query Handler Server (AQHS)
4. Query for path between A' and B'
5. Result for path between A' and B'
Navigation Server (NS)
2. Query for neighbours of A and B
3. List of neighbours of A and B
Neighbourhood Query Server (NQS)
Fig. 3: System Architecture

VII. Location Obfuscation for User Privacy in Navigation Applications

Before 2006, navigation software were accessed through specific devices which were able to receive a Global Positioning System (GPS) signal. However, with the beginning of smartphone era, devices which are able to track the position of users have increased significantly. Nowadays, more than 50% of the world population owns a smartphone and in some countries more than 90% of the active mobile phones have at least one interface to detect user location [26]. Hence, it is important to provide user privacy in such applications.

The intuition behind the novel architecture is to get trusted service by making use of different independent untrusted services. The proposed architecture provides user privacy in navigation applications by using neighbourhood finding LBS and an anonymizer. This architecture makes use of novel protocol which provides protection against various attacks. In system architecture shown in Fig. 3, following protocol is followed:

Step 1: User queries for path between A to B to Anonymizer and Query Handler Server (AQHS).

Step 2: AQHS removes the identification information from this request and queries for neighbourhood locations of point A and B to Neighbourhood Query Server (NQS).

Step 3: NQS gives a list containing neighbourhood points of point A along with distance of those points from point A.

Similar list is also sent for point B by NQS to AQHS.

Step 4: AQHS takes a random point from neighbourhood list of A. Let the name of point be A'. Similarly, AQHS takes a random point from neighbourhood list of B. Let the name of point be B'. After that, AQHS queries for path between A' and B' to Navigation Server (NS).

Step 5: NS sends an optimal path between A' and B' to AQHS.

Step 6: AQHS checks if point A and B are present on this path. If point A and B are present on this path, then AQHS filters out the path between A and B from optimal path between A' and B' (Fig. 4a) and sends that path to user.

Let us assume that filtered path is not optimal. In that case, just by replacing this path with optimal path between A and B, it will be possible to get another optimal path between A' and B' with lesser cost than the optimal path which is sent by NS. This is a contradiction. Hence, this filtered path is optimal path between A and B.

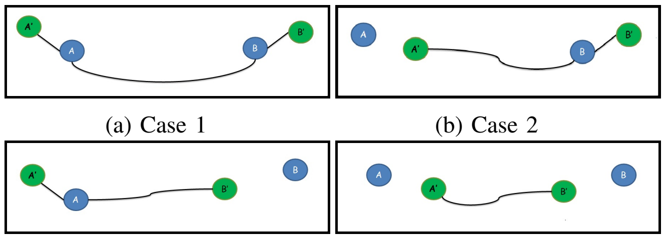
If in case, point A and/or B are not present on this path then following 3 cases can occur.

- Point A is not on the path between A' and B' (Fig. 4b): In this case, AQHS filters out the optimal path between A' and B from optimal path between A' and B'. Now as point A is in neighbourhood of point A', AQHS makes an approximation of concatenating point A in path A'B to get path AB and sends that path to user. Note that path AB may not be an optimal path but experiments show that for real world graph this approximation works really well.
- Point B is not on the path between A' and B' (Fig. 4c): In this case, AQHS filters out the optimal path between A and B' from optimal path between A' and B'. Now as point B is in neighbourhood of point B', AQHS makes an approximation of concatenating point B in path AB' to get path AB and sends that path to user.
- Both points A and B are not on the path between A' and B' (Fig. 4d): In this case, as point A and B is in neighbourhood of point A' and B', AQHS makes an approximation of concatenating point A and B in path A'B' to get path AB and sends that path to user.

In this protocol, as queries are sent through the AQWS, user's identity is hidden from NS. Moreover, actual location of source and destination is also obfuscated from NS. Hence, this architecture provides two layer protection to preserve user privacy.

VIII. Experiments and Results

Simulation studies and results of proposed obfuscation functions and novel architecture are presented in this section. All experiments were performed on a system with 4GB memory, Intel(R) Core(TM) i5-2430 CPU @ 2.40GHz processor. Servers are made using flask which is a microframework for Python based on Werkzeug, Jinja 2 [27]. For evaluation of novel architecture, clients are made using python based scripts. This section is divided into 3 subsections. In first subsection, proposed obfuscation techniques are evaluated. After proving the efficacy of proposed obfuscation techniques, comparison of encryption and proposed obfuscation techniques is done in second subsection. Third subsection evaluates proposed novel architecture for user privacy in navigation application.



(a) Case 1 (b) Case 2

(c) Case 3 (d) Case 4

Fig. 4: Different cases for query processing

A. Evaluation of proposed obfuscation techniques:

Random path is used for experimenting with user obfuscation functions. Real path in Fig. 5a is input to EROF and MROF which produces obfuscated path in Fig. 5b and 5c respectively. Green box in Fig. 5c shows bounding region. Yellow lines in Fig. 6a and 6b shows road network. Fig. 6a is a real path whose obfuscated path using GOF is shown in Fig. 6b.

After applying proposed obfuscation functions on random path, difference between original and obfuscated is analysed. Fig. 7a, 7b and 7c shows the variation of obfuscated path from original path. Red line in these figures shows the difference between X coordinates of original path and obfuscated path. Similarly, green line shows the difference between Y coordinates of original path and obfuscated path. Blue line shows the distance between original location and corresponding obfuscated location for every point. X - axis in these figures shows the serial number of location point in the same order as it is occurring in original path.

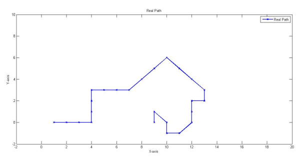
As described earlier, MROF and GOF are extension of Random Obfuscation Function (ROF) and Linear Obfuscation Function (LOF) proposed in [9]. Disadvantages of ROF and LOF are described in Section 5. MROF and GOF can have more time of execution in worst case, but they don't have disadvantages that ROF and LOF have.

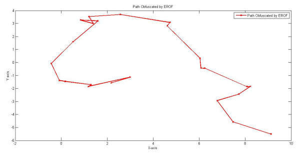
Note that obfuscation functions not only change shape of original path, but also change location of original path. It is observed that all obfuscation functions gives good user privacy by changing shape and location of original path. Hence, as per the privacy profile needed and server assumptions, these obfuscation functions can be used.

It is to be noted that the obfuscation schemes proposed in this paper preserves the distance travelled by user. So, the LBS results accuracy will not be affected due obfuscation schemes. The results given by obfuscated LBS and non-obfuscated LBS will be exactly same.

B. Encryption v/s Obfuscation:

In Fig. 8, comparison of AES (Advanced Encryption Standard) encryption function with proposed obfuscation is done


(a) Real path


(b) EROF obfuscated path

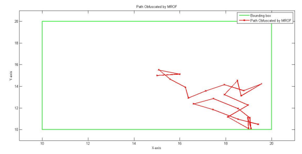
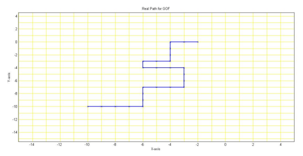

(c) MROF obfuscated path

Fig. 5: Real Path and Obfuscated Path of EROF and MROF


(a) Real path for GOF.

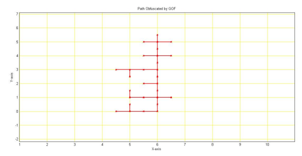
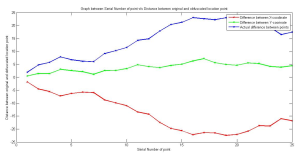
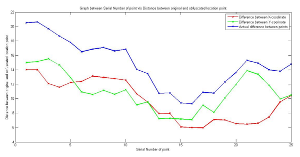

(b) GOF obfuscated path.

Fig. 6: Real Path and Obfuscated Path of GOF


(a) EROF


(b) MROF

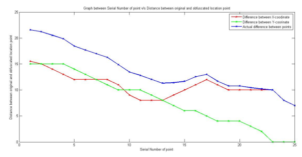

(c) GOF

Fig. 7: Difference between Real Path and Obfuscated Path

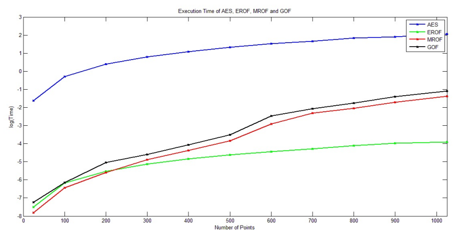


Fig. 8: Execution Time of AES, EROF, MROF and GOF

by calculating execution time. In this figure, X - axis shows the number of location points. Whereas, Y - axis shows the time of execution in logarithmic scale. Blue, green, red, black curves show execution time of AES, EROF, MROF, GOF respectively.

It can be seen that execution time of AES is far more than that of proposed obfuscated functions. So, latency of applications which uses encryption than obfuscation will be more. This would also result in more computing resources and battery if AES (encryption mode) is used in LBSs. Moreover, encryption mode has other problems as discussed in Section 4. So, to save battery and user privacy, it can be concluded that obfuscated mode of LBSs is better.

Name of Region (Graph)	No. of nodes	No. of edges
California and Nevada	1,890,815	4,657,742
Colorado	435,666	1,057,066
Northeast USA	1,524,453	3,897,636
Florida	1,070,376	2,712,798

TABLE I: Graphs used for Simulation.

Name of Region (Graph)	Actual query time (sec)	Protocol query time (sec)
California and Nevada	14903.71	14842.73
Colorado	3113.2097	3004.53346
Northeast USA	13080.988	13041.1418
Florida	8688.89201	8659.96642

TABLE II: Total query processing time for 1000 queries.

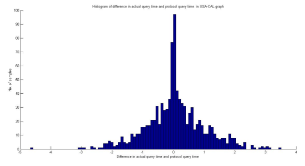
C. Evaluation of Proposed Architecture for User Privacy in Navigation Application:

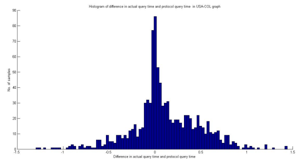
The architecture is evaluated on the basis of 2 evaluation matrices i.e. delay and error caused due to the proposed protocol. Table I lists the 4 USA road networks that are used for experimentation [28]. The nodes in the graph represent the intersection of roads. The edges of graph represent the roads in that particular region.

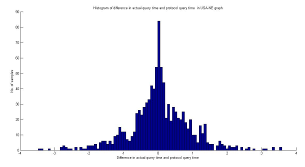
Actual query time is the time taken when a user queries path from source to destination directly to navigation server. Protocol query time is the time taken to serve query by using protocol proposed in this paper. Fig. 9 shows the histogram of difference in actual query time and protocol query time. In this figure, X - axis shows difference in actual query time and protocol query time in seconds. Whereas, Y - axis shows number of samples(queries). Fig. 10 shows histogram of percentage error in distance. In this figure, X - axis shows percentage error. Whereas, Y - axis shows number of samples(queries).

Complexity of single source - single destination Dijkstra algorithm depends on topology of graph. It is observed that in some cases, time required to get path AB is more than the time required to get path A'B' where A' and B' are neighbours of point A and B.

NQS gives the list of all nodes which are adjacent to node for which query has come. Time required to get neighbourhood points and communication delays are of the order of milliseconds. Hence, complexity of protocol is actually complexity of Dijkstra algorithm. Due to these facts, in Fig. 9, difference in actual query time & protocol query time follows a Gaussian


(a) California & Nevada


(b) Colorado


(c) Northeast USA

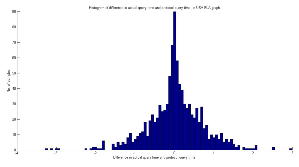
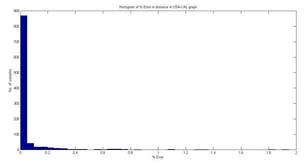
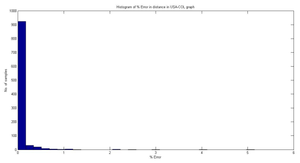
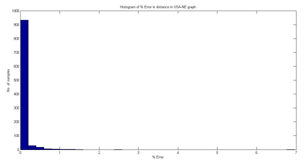

(d) Florida

Fig. 9: Histogram of difference in actual query time and protocol query time


(a) California & Nevada


(b) Colorado


(c) Northeast USA

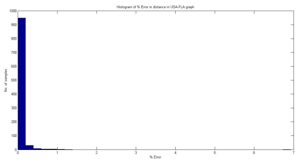

(d) Florida

Fig. 10: Histogram of percentage error in distance

distribution with center at 0.

It can be concluded from table II that on an average this system works as good as the system in which user queries path from source to destination directly to navigation server. Moreover, this system provides user privacy.

It is also observed that percentage error in distance due to approximation made in this protocol, is less than 0.2% in most of the queries (more than 90% of queries). The number of queries with percentage error more than this decreases exponentially with very high decaying rate. Thus, experiments show that user privacy can be maintained without affecting navigation results and without incurring significant overheads.

IX. Conclusion and Future Work

Popularity of location based services has been raising various privacy issues. The few existing solutions generally trade off location privacy with LBSs accuracy. Our solution provide user privacy without sacrificing service accuracy of LBS. Proposed approach is suitable for those services that need to evaluate distance travelled by user, without needing actual path travelled by user. Experimental evaluation shows that original and obfuscated path using our approach are quite different. Our approach allows obfuscation of user path and guarantees the user privacy without affecting LBSs results.

After showing the efficacy of obfuscation, comparison is made between encryption and obfuscation to find their performance in LBSs. It is observed that encryption is slower than obfuscation which can potentially degrades services offered by LBSs. Analysis also shows that obfuscation is less computationally intensive than encryption. Due to this, a lot of battery power can be saved by using obfuscation techniques.

This paper also presents a novel architecture for providing user privacy in navigation applications. Experimental results show that the error due to proposed protocol is within acceptable limits. Also average time to process queries is almost same as compared to actual query processing time.

It is also observed that obfuscation techniques are highly application dependent and hence choice of appropriate obfuscation technique is closely related to the application requirement. Hence, depending on type of application and its requirement, different kinds of obfuscation techniques need to used. Single obfuscation technique cannot provide user privacy in all kinds of applications. Therefore, there is lot of scope of innovation in this field. Obfuscation techniques proposed in this paper can be used for healthcare (fitness) application, pay as-you-go insurance application, navigation application etc. In future, privacy protection in other LBSs will be explored and suitability of obfuscation techniques for such applications will be evaluated.

References

[1] Foursquare. [Online]. Available: https://foursquare.com/
[2] Zomato. [Online]. Available: https://www.zomato.com/
[3] Runkeeper. [Online]. Available: http://runkeeper.com/
[4] Sportstracker. [Online]. Available: http://www.sports-tracker.com/
[5] Runtastic. [Online]. Available: https://www.runtastic.com/
[6] N. Li and G. Chen, "Sharing location in online social networks," *Network, IEEE*, vol. 24, no. 5, pp. 20–25, 2010.
[7] K. Baum, *Stalking victimization in the United States*. DIANE Publishing, 2011.
[8] C.-Y. Chow, M. F. Mokbel, and W. G. Aref, "Casper*: Query processing for location services without compromising privacy," *ACM Transactions on Database Systems (TODS)*, vol. 34, no. 4, p. 24, 2009.
[9] R. Di Pietro, R. Mandati, and N. V. Verde, "Track me if you can: Transparent obfuscation for location based services," in *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2013 IEEE 14th International Symposium and Workshops on a*. IEEE, 2013, pp. 1–9.
[10] P. M. Wightman, M. A. Jimeno, D. Jabba, and M. Labrador, "Matlock: A location obfuscation technique for accuracy-restricted applications," in *Wireless Communications and Networking Conference (WCNC), 2012 IEEE*. IEEE, 2012, pp. 1829–1834.
[11] H. Kido, Y. Yanagisawa, and T. Satoh, "Protection of location privacy using dummies for location-based services," in *Data Engineering Workshops, 2005. 21st International Conference on*. IEEE, 2005, pp. 1248–1248.
[12] K. P. Puttaswamy and B. Y. Zhao, "Preserving privacy in location-based mobile social applications," in *Proceedings of the Eleventh Workshop on Mobile Computing Systems & Applications*. ACM, 2010, pp. 1–6.
[13] D. Riboni, L. Pareschi, C. Bettini, and S. Jajodia, "Preserving anonymity of recurrent location-based queries," in *Temporal Representation and Reasoning, 2009. TIME 2009. 16th International Symposium on*. IEEE, 2009, pp. 62–69.
[14] M. Gruteser and D. Grunwald, "Anonymous usage of location-based services through spatial and temporal cloaking," in *Proceedings of the 1st international conference on Mobile systems, applications and services*. ACM, 2003, pp. 31–42.
[15] B. Gedik and L. Liu, "Protecting location privacy with personalized k-anonymity: Architecture and algorithms," *Mobile Computing, IEEE Transactions on*, vol. 7, no. 1, pp. 1–18, 2008.

[16] W.-C. Peng, T.-W. Wang, W.-S. Ku, J. Xu, J. Hamilton *et al.*, “A cloaking algorithm based on spatial networks for location privacy,” in *Sensor Networks, Ubiquitous and Trustworthy Computing, 2008. SUTC’08. IEEE International Conference on*. IEEE, 2008, pp. 90–97.

[17] B. Bamba, L. Liu, P. Pesti, and T. Wang, “Supporting anonymous location queries in mobile environments with privacygrid,” in *Proceedings of the 17th international conference on World Wide Web*. ACM, 2008, pp. 237–246.

[18] R. Cheng, Y. Zhang, E. Bertino, and S. Prabhakar, “Preserving user location privacy in mobile data management infrastructures,” in *Privacy Enhancing Technologies*. Springer, 2006, pp. 393–412.

[19] A. Pfitzmann and M. Köhntopp, “Anonymity, unobservability, and pseudonymitya proposal for terminology,” in *Designing privacy enhancing technologies*. Springer, 2001, pp. 1–9.

[20] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K.-L. Tan, “Private queries in location based services: anonymizers are not necessary,” in *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. ACM, 2008, pp. 121–132.

[21] M. A. Lele and J. Gu, “Evaluation of solid state accelerometer sensor for effective position estimation,” in *Intelligent Control and Automation (WCICA), 2011 9th World Congress on*. IEEE, 2011, pp. 959–964.

[22] M. L. Anjum, J. Park, W. Hwang, H.-i. Kwon, J.-h. Kim, C. Lee, K.-s. Kim, and D.-i. Cho, “Sensor data fusion using unscented kalman filter for accurate localization of mobile robots,” in *Control Automation and Systems (ICCAS), 2010 International Conference on*. IEEE, 2010, pp. 947–952.

[23] K. P. Puttaswamy, S. Wang, T. Steinbauer, D. Agrawal, A. El Abbadi, C. Kruegel, and B. Y. Zhao, “Preserving location privacy in geosocial applications,” *IEEE Transactions on Mobile Computing*, vol. 13, no. 1, pp. 159–173, 2014.

[24] M. Knecht, W. Meier, and C. U. Nicola, “A space-and time-efficient implementation of the merkle tree traversal algorithm,” *arXiv preprint arXiv:1409.4081*, 2014.

[25] M. G. Wing, A. Eklund, and L. D. Kellogg, “Consumer-grade global positioning system (gps) accuracy and reliability,” *Journal of forestry*, vol. 103, no. 4, pp. 169–173, 2005.

[26] A. Saracino, D. Sgandurra, and D. Spagnuelo, “Addressing privacy issues in location-based collaborative and distributed environments,” in *Collaboration Technologies and Systems (CTS), 2014 International Conference on*. IEEE, 2014, pp. 166–172.

[27] Flask. [Online]. Available: http://flask.pocoo.org/

[28] Dimacs. [Online]. Available: http://www.dis.uniroma1.it/challenge9/download.html