Privacy-Aware Proximity Based Service using Hide & Crypt Protocol: Implementation

L. Ertaul, B. F. Imagnu, S. Kilaru.
California State University, East Bay, Hayward, CA, USA

Abstract-**Proximity based services are location based services (LBS) in which the service adaptation depends on the comparison between a given threshold value and the distance between a user and other (possibly moving) entities. While privacy preservation in LBS has lately received much attention, very limited work has been done on privacy-aware proximity based services. This paper describes the main privacy threats that the usage of these services can lead to, and explains the implementation of a privacy preserving protocol, Hide&Crypt. The use of simple and well-known encryption algorithms is also mentioned which is used to hide messages exchanged between users.**

Index Terms- Hide&Crypt, LBS, SP-Filtering, Proximity-based services, Privacy-Aware LBS

1.Introduction

Location based services (LBS) are becoming popular thanks to the advances in positioning technologies and to the diffusion of mobile devices with data communication capabilities. Proximity based services are a special class of LBS in which the service adaptation depends on the comparison between a given threshold value and the distance between a user and other (possibly moving) entities. The so-called "friend- inder" services are an example: Alice would like to be alerted whenever her friend Bob is nearby, so that they could get in contact and possibly meet. The proximity service considered in this paper is a generalization of "friend-finder" in which each user is part of one or more possibly large and dynamically changing groups of users (called buddies) related to hobbies, sports, religious or cultural interests. Technically, a proximity query is a spatial range query on the database of moving buddies in which the range is defined by the circle centered at the issuer's location and having the proximity threshold as radius. [1][2][3][4][5].

A major privacy concern with the use of LBS is the release to untrusted third parties of the user precise location information. This concern applies to proximity services as well: Alice would like to use the proximity service without necessarily releasing her exact position to the service provider (SP). In some cases, she may even wish not to provide the exact location to the buddies, although she may be willing to reveal whether she is in proximity. For example, she may agree to let Bob know that she is in a neighborhood near Bob's location, but keep the specific address hidden from Bob. In practice, this may avoid the situation in which buddies can directly walk to other buddies, as the goal of the service is usually to enable communication that may only eventually lead to meetings in person. A solution to the above privacy concern can be to allow each user to specify certain minimum location privacy requirements both with respect to the service provider and to the buddies. Note that these are minimum requirements, and a system should be designed with the goals to (1) guarantee the satisfaction of the minimum privacy requirements, and (2) reveal as little location information as possible. This paper provides the design of proximity service protocol toward these two goals. Several LBS privacy preserving techniques have been recently proposed [1][2][3][4][5].

The rest of the paper is organized as follows. ***Section 2*** describes the services, the privacy preferences, the threats, and the evaluation metrics. In ***Section 3*** we illustrate the implemented protocol, Hide&Crypt and its formal properties. In ***Section 4*** explanation on the implementation of Hide&Crypt and in ***Section 5*** we report experiments, in ***Section 6*** we show screenshots from our implementation and finally in ***Section 7*** we conclude with a short discussion.

2. Proximity services and privacy concerns

In this section we first formalize the proximity services and the related privacy requirements. We then describe how these requirements may be violated, and how different techniques avoiding these privacy threats could be compared.

2.1. The proximity service

The proximity service informally described in the introduction can be more precisely defined by considering a typical service provisioning session: a user A sends her own location information, acquired via GPS or other techniques, and requests to be alerted whenever a buddy1 reports a location that is in proximity while it was not in proximity before, or vice versa is not anymore in proximity. Here, proximity is defined as being within a distance threshold given by A, denoted δ_A, i.e., A is interested in all the buddies B satisfying the following condition:

$$d(loc(A);loc(B)) \leq \delta_A \qquad (1)$$

where d(loc(A); loc(B)) denotes the Euclidean distance between the reported locations of A and B. When (1) is true, we say that B is in the proximity of A. Since B may be different from A, this proximity relation may not be symmetric. In order to provide proximity service, it is convenient to have a service provider SP, especially when group sizes can be large and membership of the groups can dynamically change. Indeed, under these conditions, directly computing distances between A and every buddy can be extremely inefficient or even infeasible. Henceforth, we assume the existence of SP in providing proximity service.

With the presence of SP, and in the absence of privacy concerns, a simple protocol can be devised to implement the service: The SP receives from each user's automatic location updates and stores their last known positions, as well as the distance threshold δ_A for each user A. While in theory each user can define a different value δ_A for each buddy, in this paper, for the sake of simplicity, we consider the case in which

each user defines a single value of δ_A. When the SP receives a location update, it may recompute the distance between A and each buddy (possibly with some filtering/indexing strategy for efficiency). If any proximity relation changes, A is notified. In a typical scenario, if B is in proximity, A may contact him directly or through SP; however, for the purpose of this paper, we do not concern ourselves as what A will do once notified.

2.2. User minimum privacy requirements

The privacy we are considering in this paper is location privacy, i.e. we assume that users are concerned about other persons obtaining information about their exact location at specific times. In the considered services, users may prefer the service provider to have as little information about their location as possible, and the buddies not to know her exact position, even when proximity is revealed. In general, the level of location privacy can be represented by the uncertainty that an external entity has about the position of the user, and this uncertainty can be formally represented as a geographic area in which no point can be ruled out as a possible position of the user. In principle each user could express her privacy preferences, by specifying for each other user (or class of users perceived as adversaries) a partition of the geographical space defining the minimal uncertainty regions that she wants to be guaranteed. For
example, Alice specifies that Bob should never be able to find out the specific building where Alice is within the campus, i.e., the entire campus area is a minimal uncertainty
region. The totality of these uncertainty regions can be formally captured with the notion of *spatial granularity*.

While there does not exist a formal definition of spatial granularity that is widely accepted by the research community, the idea behind spatial granularities is simple. Similarly to temporal granularity [6], a spatial granularity can be considered a subdivision of the spatial domain into a discrete number of non-overlapping regions, called granules. In this paper, for simplicity, we consider only granularities that partition the spatial domain. In principle, granules of the same granularity can have any shape and don't need to have the same size or shape. Each granule of a granularity G is identified by an index (or a label). We denote with G(i) the granule of the granularity G with index i.

Users specify their minimum privacy requirements via spatial granularities, with each granule being a minimum uncertain region. In the following of this paper we assume that each user specifies two granularities

$$G^{SP}{}_A \text{ and } G^{U}{}_A$$

defining the minimum location privacy requirements for SP and for any other user, respectively, as the two categories of potential adversaries. The two extreme cases in which a user requires no privacy protection and maximum privacy protection, respectively, can be naturally modeled. For example, if a user A does not want her privacy to be protected with respect to other buddies (in this case A can tolerate other buddies to know her location at the maximum available precision) then A will set $G^{U}{}_A$ to the bottom granularity $\perp$ (a granularity that contains a granule for each basic element, or pixel, of the spatial domain). Similarly, if A wants to impose the maximum privacy protection with respect to the SP, then A sets $G^{SP}{}_A$ to the top granularity T (the granularity that has a single granule that covers the entire spatial domain).

2.3. Privacy threats

In order to formally identify the privacy threats, it is crucial to first specify the assumptions about the available external knowledge and about the behavior of considered adversaries. In this paper we consider both buddies and SP (as well as external entities that may have taken control of one of them) as potential adversaries, and we assume the following: (a) there is no external knowledge on the location of users other than the one exchanged during the protocol, and (b) buddies and SP do not collude. The formal proofs of our results also assume that the involved entities are not malicious, in the sense that they follow the protocols defined in the service.

Observe the simple protocol described in Section 2.1; it is easily seen that even under the above assumptions the location privacy of users is at risk. When the SP is considered as a potential adversary, an SP-threat can be identified: Since the exact location of user A is stored by the SP, if $G^{SP}{}_A$ is not the bottom granularity, A's minimum privacy requirement is violated. When a buddy is considered as a potential adversary, a buddy-attack can be identified as follows. Suppose A is a buddy of B who sets a value of in a way such that the circular region of radius δ_B (centered at B's location) is properly contained in a granule of $G^{U}{}_A$. Then, if A happens to enter that circular region, the SP will notify B, and A's minimum privacy requirement would be violated.

2.4. Privacy protection performance

In the sequel, we present techniques to protect user privacy. The minimum goal of our techniques is to guarantee the satisfaction of users' minimum privacy requirements, which all of our protocols provide. However, there are three additional performance goals to be considered.

The first is based on the observation that more privacy is generally more desirable by users; we should strive to provide larger uncertainty region than the minimum ones given by the user. Hence, the first performance measure of the protection is the size of the uncertainty region. The larger the uncertainty region, the better.

The second performance goal is to minimize the system costs, including computation and communication. As we show later, there is a trade-off between the privacy level and the costs.

The third performance metrics is the service precision. Due to the user minimum privacy requirement, there may be uncertainty whether user B is actually in proximity of A. We take a conservative approach, namely when it's uncertain, we report to A that B is in proximity. The service precision is then defined as the percentage of times that B is indeed in A's proximity when alerted (based on the reported locations of A and B) among all the times A is alerted. Obviously, the higher precision, the better.

3. Privacy preserving Techniques

3.1 SP-Filtering

SP-Filtering[7] is a three-party protocol that computes the proximity of B to A with a certain approximation,

guaranteeing the satisfaction of the minimum location-privacy requirements of both A and B.

The idea of the algorithm is that when a user A performs a location update, instead of providing her exact location to the SP, she sends a generalized location that is computed as a function of GU A and the granule $G^{SP}_A(i)$ where A is located. More precisely, A sends to SP the location LA(i) that is computed as the union of the granules of G^{U}_A that intersects with $G^{SP}_A(i)$. Formally

$$L_A(i) = \bigcup_{i' \in \mathbb{N} | G^U_A(i') \cap G^{SP}_A(i) \neq \emptyset} G^U_A(i') \quad (2)$$

Each buddy B does the same when location is updated with LB(j) similarly defined, where j is the index such that the location of B is in $G^{SP}_B(j)$. Then, the SP can compute, for each buddy B of A, the minimum and maximum distance between any two points of LA(i) and LB(j).We denote with d and D the minimum and maximum distance, respectively.

3.2 Hide&Crypt

Hide&Crypt works as follows. First, A computes the set S' of indexes of granules of GUB that intersects with the circle C centered in the location of A with radius δA. Then, in order to hide to B the cardinality of this set, A creates a new set S by adding to S0 some negative numbers. The aim of negative numbers is to increase the cardinality of S without affecting the result of the computation. The cardinality of S should be increased so that it is as large as the number SMAX that represents the maximum number of granules of GUB that intersect with any circle with radius δA. Note that SMAX can be computed off-line since its values depend only on GUB and δA. Then, A encrypts all the elements of S with an encryption function E and a private key KA and sends the result to B. User B encrypts again, using his private key KB, each element in the set he receives and sends it back to A together with the encryption of the index j such that B is located in GUB (j). Finally, A encrypts again EKB(i) using the key KA and checks if the result is contained in EKB(EKA(S)). Encryption function E is such that EKA(EKB(i)) ϵ EKB(EKA(S)) if and only if j ϵ S. Since negative numbers are not valid indexes, j ≥ 0, and hence j ϵ S if and only if j ϵ S'. Therefore A computes whether B is in her proximity or not.[7]

Protocol Hide&Crypt

Prerequisites: A and B are running the SP-Filtering protocol. User A knows G^{U}_B, a private key KA, the circle C centered in A's location with radius δ_A, and the value S_{MAX}. B knows a private key K_B, and the granule $G^{U}_B(j)$where B is located.[7]

Protocol:

1: A receives "B is possibly in proximity" from the SP.
2: A computes: S' = {jϵN s.t. $G^{U}_B(j)$ n C≠ϕ
3: A computes: S'' as a set of SMAX-|S'| random negative numbers.
4: A computes: S = S' U S''
5: A sends "starting two-parties protocol $E_{KA}(S)$" to B
6: B sends $[E_{KB}(E_{KA}(S));E_{KB}(j)]$[1] to A

Note-$[E_{KB}(E_{KA}(S));E_{KB}(j)]$-hold true for any commutative symmetric encryption algorithm. ***We used Vernam Encryption algorithm for the implementation***

7: A computes: $E_{KA}(E_{KB}(j))$
8: if ($E_{KA}(E_{KB}(j))$ ϵ $E_{KB}(E_{KA}(S))$) then
9: A computes that B is in proximity
10: else
11: A computes that B is not in proximity
12: end if

4. Hide&Crypt Implementation

The process starts from User A which request to know if User B is in proximity or not by updating its location to the SP. The SP will then do some simple calculations and comparisons to notify User A whether User B is in proximity, not in proximity or might be in proximity.

In the last case, proximity is decided by secure communication between individual Users, not the SP.

A user updates its location in some interval of time or when they query the SP in order to know the proximity of their friends. SP uses the last known user location for proximity calculations.

In this implementation the granule of every individual user is represented 8 points which are calculated by the user itself using its own gps location using latitude and longitude. User's current location and the method used to calculate the 8 points that represent user's own location are kept secret of only the user. The SP is only provided with these 8 points from every user which updates their locations. The calculations done by the SP are completely based on these 8 points. For example, between the 8 points of User A and the last updated 8 points of user B.

The step-by step calculations used to create the eight points by user A for our implementation is as follows.

1. User A acquires its current latitude and longitude of his exact location. There are a lot of applications to get this information on mobile devices. Ex- Compass in Iphone, GPS Status in Android.
2. User A decides its minimum privacy distance, δ_A.
3. δ_A is changed to degrees of latitude and longitude, Ax and Ay respectively. Since 1^0 latitude = 69miles, 1^0 longitude = 53 miles.
4. Ax is added and subtracted to/from A's latitude, the same is done with longitude using Ay.
5. Choose two of the points results of Ax or Ay, but not one from each, from the four calculated location points in the previous step and add/subtract Ax or Ay to get locations the rest four points (add/subtract Ax if the two location points used are the sum/difference of Ax from A's current location and vice versa). As shown in Fig.1

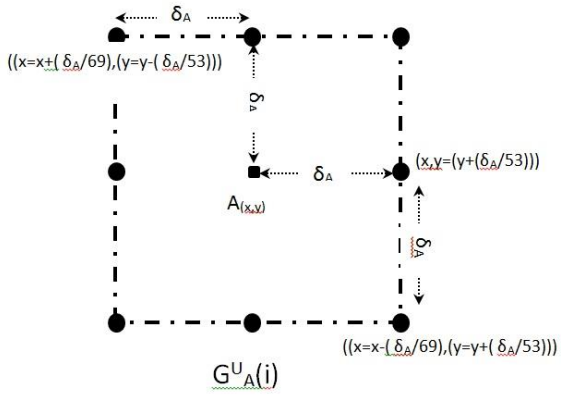

Figure 1. Granule of a user

After SP has acquired all the 16 points needed to calculate the proximity of B to A i.e. 8 points each calculated using their corresponding minimum privacy, δ_A and δ_B. SP will calculate the distance from every point of A to every point of B using simple calculation as shown below.

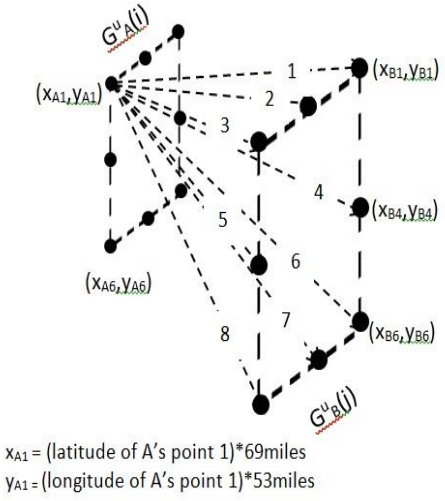

Figure 2. Calculating the distances between the granules of the two users

$$\text{Distance1}=\sqrt{(x_{A1}-x_{B1})^2+(y_{A1}-y_{B1})^2} \quad (3)$$

These 64 distance values will compare to each other to decide d and D.

d = the smallest distance from points of A to B.

D = the largest distance from points of A to B.

SP will decide the proximity based on the result of the comparison.

Case1.If, $D \leq \delta_A$ then "B is in proximity of A."

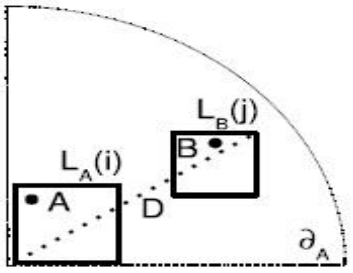

Figure 3. Case 1

Case 2. If $d \geq \delta_A$ then "B is not in Proximity of A."

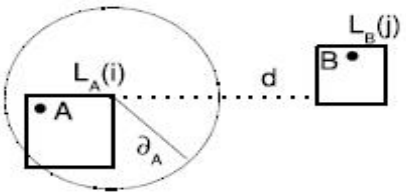

Figure 4. Case 2

Case 3.If $d \leq \delta_A \geq D$ the "B might be in Proximity of A" where A and B communicate securely to decide proximity.

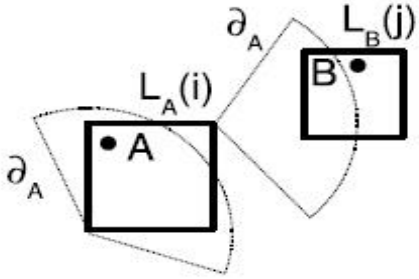

Figure 5. Case 3

If SP filtering results "B might be in proximity of A" then the SP divides the area, containing both A and B, into areas in the size of B's granule and index the divided areas. This table containing the indexes and their corresponding latitude and longitude values is sent to both users so that they can base the process of determining proximity through secure communication based on the same information (table).

The details of the preparation of this table are as follows. SP separate the minimum (min.) and maximum (max.) latitudes (Lat.) and longitude (Long.) from the 16 points that are sent from both users (8 from each) as shown in the Figure 6.

This area is divided to smaller squares starting from the point located at (max Lat., min Long.). Then each small area will be represented by the left top corner point i.e. the point with the largest latitude and lowest longitude among its four corner points.

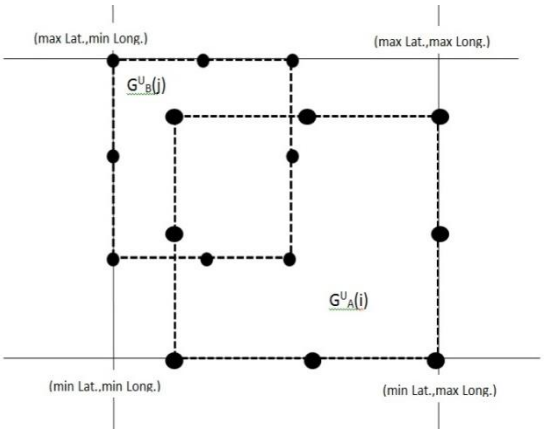

Figure 6. Calculating the coordinates

The latitude and longitudes of the individual cells will be calculated from the coordinate of the perivious cell except for the first one which is located at (max Lat., min Long.).The intervals for the latitude and longitude are calculated as follows.

Area 1= (max Lat.,min Long.)

Area 2=(max Lat,(min Long.+(δ_B/53)))

Area 3= ((max Lat.-(δ_B/69)),min Long.)

Area 4= ((max Lat.-(δ_B/69)),(min Long.+(δ_B/53))

The maximum number of cells resulted from the division of the whole area is,

Latitude m=(max Lat.- min Lat)/(δ_B/69)…. (4)

Longitude n=(max Long.- min Long.)/(δ_B/5)…(5)

Number of cells= m*n….(6)

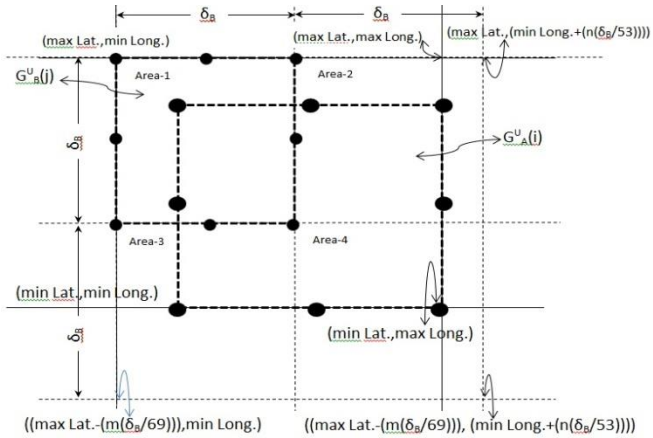

Figure 7 Dividing the area using the size of B's granule

Table 1.The indexed information

Index	m	n	Latitude	Longitude
Area1	1	1	max Lat.	Min Long.
Area2	1	2	max.Lat.-(m(δ_B/69))	min Long
Area3	2	1	max.Lat.-(m(δ_B/69))	min.Long.+(n(δ_B/53))
Area4	2	2	max.Lat.-(m(δ_B/69))	min.Long.+(n(δ_B/53))

After both users receive the table from Table 1 SP, proximity of B to A is decided by user A through a secure communication with B following the steps below.

1. User A will encrypt and send the indexes where its granule representing 8 points lie in using its secret key K_A, i.e. all the indexed areas 1,2,3,4, which is not always true.(two of them lie in area 1, three of them lie in area2,one lie in area3 and the last two lie in area 4)
2. B will encrypt the received message with its own secret key K_B and also encrypt and add the index of the area where it is located in and send all of it back to A.
3. A will encrypt the encrypted location of B and check with the other encrypted values. If a match is found then "B is in proximity of A" if not then "B is not in proximity of A".

Ex. Let B is located in indexed area 3 and $E_{KB}(3)=p$

$E_{KA}(1)=a$, $E_{KA}(2)=b$, $E_{KA}(3)=c$, $E_{KA}(4)=d$

$E_{KB}(a)=w$, $E_{KB}(b)=x$, $E_{KB}(c)=y$, $E_{KB}(d)=z$

$E_{KA}(1)=a$, $E_{KA}(2)=b$, $E_{KA}(3)=c$, $E_{KA}(4)=d$

1. $E_{KA}(1,2,3,4)=\{a,b,c,d\}$……..A sends to B
2. $E_{KB}(a,b,c,d,3)=\{w,x,y,z,p\}$……B sends to A
3. $E_{KA}(p)=\{y\}$…. A compare y with w,x,y,z

Since a match is found the "B is in Proximity of A."

5.Experimental Evaluation

We defined 5 levels of granularities for our experiment increasing in size as the level of granularity increase. The smallest and the largest size of granularities we considered are Level 0 which is 5milesX5miles square and Level 4 50milesX50miles as shown in Table 2.

Table 2. Five Level Granuluties

Parameter	Value
δ	5m,10m,25m,50m
Level of G^U Granularity	0,1,2,3,4
Avg number of buddies in a group	2,3,4,5

For simplicity reason we have forced all users to use the same minimum privacy requirement, δ for a given experiment.

Since hide&crypt protocol comes to action only when the "might be in proximity" condition happen, our experiments are on the same case too. SP-filtering calculations are simple and made by server so it is not included in this paper. Another think we held constant is the 128 bits encryption key generated using AES-CTR. Every user has fixed encryption key to use with Vernam Cipher.

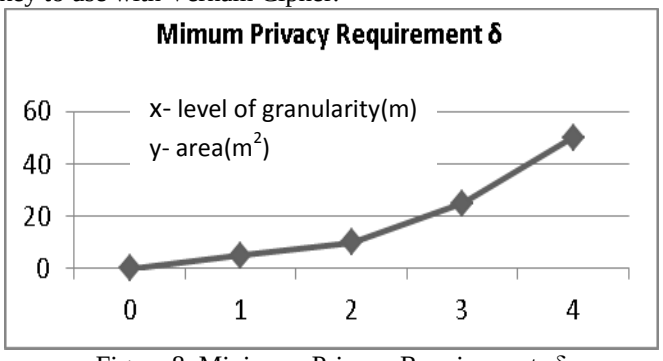

Figure 8. Minimum Privacy Requirements δ

As we can see from Figure 8 & 9 as δ increase the granularity of G^U increase exponentially making it cover a large area and cover many number of smaller granules the size of another user's granules, hence creating many indexes to be encrypted by users. This is shown in Figure 10.

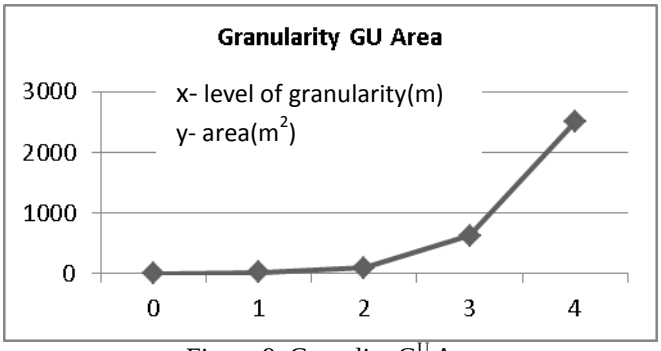

Figure 9. Granulity G^U Area

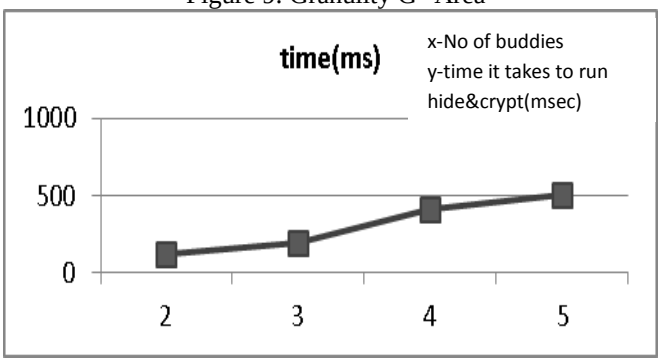

Figure 10. Total Time taken

6.Screenshots from Implementation

We used Dot Net, xml, html and javascript platform for our implementation.it is a web based program that works for all zipcodes in the USA. We used a tool in Visual Studio to change zipcodes into latitude and longitude.

Figure 11 shows the interface on which both users enter their zipcodes and their minimum privacy requirements to calculate their corresponding 8 representative points.

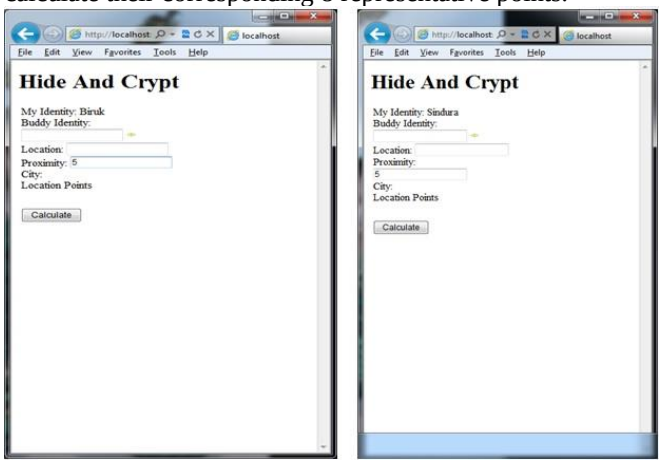

Figure11. Hide&Crypt Implementation Interface

The user calculated 8 points and the minimum privacy requirement are displayed in Figure12. This data is sent and proximity is determined by calculating the distances from each point of one user to each point of the other user. The smallest and largest values from these distances are selected and compared with the minimum privacy requirement of the user requesting the service for the decision. Proximity is decided by the SP and the message is sent to requesting user as shown in Figure 13.

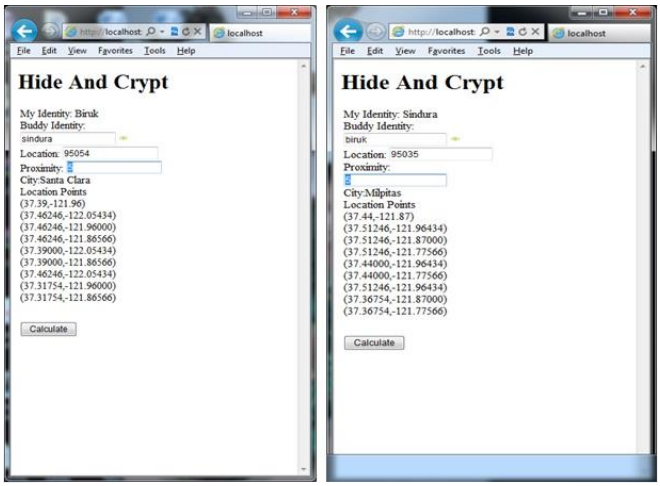

Figure 12. Eight Points Representing Location of Each User

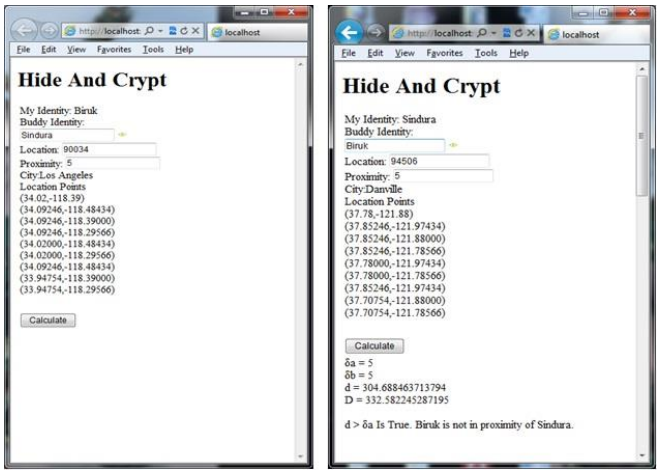

Figure13. Result of Hide&Crypt(Not in Proximity case)

In the case when one user might be in proximity of the other case, SP sends the index table as shown in Table 1 for both users. Then Hide & Crypt protocol starts communication between the two users. The process is described in detail in Section 4. Figure 14 shows the messages exchanged between the two users and the decision reached on proximity through this secure communication between them.

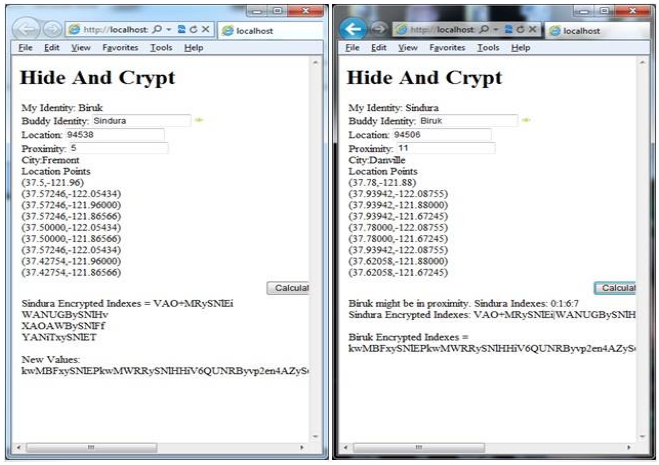

Figure 14. Encryption Process in Hide&Crypt

7.Conclusion

Hide&Crypt is very secure and reliable protocol. It uses simple and highly effective encryption algorithms for security. It does not reveal any information to a third party server, Service Provider or other buddy users about the exact location of service requesting user or vice versa.

Hide&Crypt seems to generate a lot repetitive calculations in the SP Filter but since SP Filter is a server provided by the Service Provider, computation is not time and resource consuming. But creating 8 points, generating new encryption key for every session and comparing ciphers and exchange of results between user devices might be power and resource consuming from which mobile devices are scares of.

Accuracy of this protocol depends on how granularity is defined i.e. we used 8 points to represent a granule, and when the minimum privacy requirement gets larger and larger these 8 points spread further from each other making the proximity determining calculations less and less accurate. But on the contrary it also gives users higher privacy when these points are far from each other making it much harder for attackers wanting to acquires users exact location. With additional methods of hiding users location and defining granules, this protocol's efficiency can be highly improved.

References

[1]. Bugra Gedik and Ling Liu.*" Protecting location privacy with personalized k-anonymity: Architecture and algorithms."* IEEE Transactions on Mobile Computing, 7(1):1–18, 2008.

[2]. Gabriel Ghinita, Panos Kalnis, Ali Khoshgozaran, Cyrus Shahabi, and Kian-Lee Tan."*Private queries in location based services: anonymizers are not necessary"* In Proc. Of SIGMOD. ACM Press, 2008.

[3]. Panos Kalnis, Gabriel Ghinita, Kyriakos Mouratidis, and Dimitris Papadias. "*Preventing location-based identity inference in anonymous spatial queries."* IEEE Transactions on Knowledge and Data Engineering, 19(12):1719–1733, 2007.

[4]. Sergio Mascetti, Claudio Bettini, Dario Freni, and X. Sean Wang. "*Spatial generalization algorithms for LBS privacy preservation"* Journal of Location Based Services, 2(1), 2008.

[5]. Hua] Man Lung Yiu, Christian S. Jensen, Xuegang Huang, and Lu. Spacetwist: "*Managing the trade-offs among location privacy, query performance, and query accuracy in mobile services."* In Proc. of the 24th International Conference on Data Engineering. IEEE Computer Society, 2008.

[6]. Claudio Bettini, Xiaoyang Sean Wang, and Sushil Jajodia. "*Time Granularities in Databases, Temporal Reasoning, and Data Mining".* Springer, 2000.

[7]. Sergio Mascetti, Claudio Bettini ,Dario FreniDICo Universit`a di Milano, X. Sean Wang Department of CS University of Vermont, Sushil Jajodia CSIS,George Mason University, "*Privacy-Aware Proximity Based Services."*

[8]. Canalys.com, "*Gps smart phone shipments overtake pnds in emea*," November 2008. [Online]. Available: http://www.canalys.com/pr/2008/r2008111.html

[9]. ABI research, "*Location-based mobile social networking will generate global revenues of $3.3 billion by 2013,*August2008.[Online].Available:http://www.abiresearch.com/abiprdisplay.jsp?pressid=1204"*Stalk your friends withgoogle*,"2009.[Online].Available:http://features.csmonitor.com/innovation/2009/0/04/stalk-your-friends-with-google

[10]. S. Mascetti, C. Bettini, and D.Freni,"*Longitude: Centralized privacy-preserving computation of users' proximity.*" In *Secure Data Management*,2009,

[11]. P. Ruppel, G. Treu, A. Küpper, and C. Linnhoff-Popien"*Anonymous User Tracking for Location-Based Community Services*," in *LoCA*, 2006, pp. 116–133.

[12]. L. Šikšnys, J. R. Thomsen, S. Šaltenis, M. L. Yiu,and O. Andersen, "*A Location Privacy Aware FriendLocator*," in *SSTD*, 2009.

[13]. A. Amir, A. Efrat, J. Myllymaki, L. Palaniappan, and K. Wampler, "*Buddy Tracking - Efficient Proximity Detection Among Mobile Friends*," in *INFOCOM*,2004,

[14]. C. A. Ardagna, M. Cremonini, E. Damiani, S. D. C.di Vimercati, and P. Samarati, "*Location Privacy Protection Through Obfuscation-Based Techniques,"* in *DBSec*, 2007,

[15]. M. Duckham and L. Kulik, "*A Formal Model ofObfuscation and Negotiation for Location Privacy*," in *PERVASIVE*, 2005

[16]. M. Gruteser and D. Grunwald, "*Anonymous Usage of Location-Based Services Through Spatial and Temporal Cloaking*," in *USENIX MobiSys*, 2003

[17]. M. F. Mokbel, C.-Y. Chow, and W. G. Aref, "*The New Casper: Query Processing for Location Services without Compromising Privacy*," in *VLDB*, 2006

[18]. G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K.-L. Tan, "*Private Queries in Location Based Services: Anonymizers are not Necessary*," in *SIGMOD*.2008, pp. 121–132.

[19]. A. Khoshgozaran and C. Shahabi, "*Blind Evaluation of Nearest Neighbor Queries Using Space Transformation to Preserve Location Privacy*," in *SSTD*, 2007

[20]. K. Liu, C. Giannella, and H. Kargupta, "An Attacker'sView of Distance Preserving Maps for Privacy Preserving Data Mining," in *PKDD*, 2006, pp. 297–308.

[21]. T. Brinkhoff, "*A Framework for Generating Network-Based Moving Objects*," *GeoInformatica*, vol. 6, no. 2,