ISSN:0975-9646

Securing User Location in Geo Social Networking Using Coordinate Conversions

J Maruthi Nagendra Prasad[1], N.Penchalaiah[2]

[1,2]*Assistant Professor, Department of Computer Science & Engineering*
Annamacharya Institute of Technology & Sciences, Rajampet, Andhra Pradesh

***Abstract*- Using Geo-social networking like Apple's iGroups and Hot Potato., many people communicate with their neighbouring locations through their associates and their suggestions. Without sufficient location protection, however, these systems can be easily misused, In this paper, we introduce, a technique that provides location secrecy without adding complexity into query results. Our idea here is to secure user-specific, coordinate conversion to all location data shared with the server. The associates of a user share this user's secret key so they can apply the same conversion. This allows all spatial queries to be evaluated correctly by the server, but our privacy mechanisms guarantee that servers are unable to see or infer the actual location data from the transformed data or from the data access.**

***Keywords*-Security, location-based social applications, location conversion**.

I. INTRODUCTION

Geosocial networking application is using gps location services to provide a social interface to the physical world. Examples of popular social applications include freelancing networks are created with the specific purpose to allow users to find or post temporary employment opportunities. users establish and operate a professional profile and are able to connect with past and possible employers, employees, colleagues, classmates and friends and social rendezvous [1], local friend recommendations for dining and shopping [2], [3], as well as collaborative network services and games [4], [5]. the explosive popularity of mobile social networks such as scvngr [6] and foursquare (3 million new users in 1 year) likely indicate that in the future, social recommendations will be our primary source of information about our surroundings.

This functionality comes with significantly increased risks to privacy of the users. For current services with minimal privacy mechanisms, these data can be used to infer a user's detailed activities, or to track and predict the user's daily movements. In fact, there are numerous real-world examples where the unauthorized use of location information has been misused for economic gain [7], physical stalking [8], and to gather legal evidence [9]. Even more disturbing, it seems that less than a week after Facebook turned on their popular "Places" feature for tracking users' locations, such location data were already used by thieves to plan home invasion

Clearly, mobile social networks of tomorrow require stronger privacy properties than the open-to-all policies available today.

Existing systems have mainly taken three approaches to improving user privacy in geosocial systems: 1) introducing uncertainty or error into location data [11], [12], [13], 2) relying on trusted servers or intermediaries to apply anonymization to user identities and private data [14], [12], [15], and 3) relying on heavy-weight cryptographic or private information retrieval (PIR) techniques [16], [17], [18], [19]. None of them, however, have proven successful on current application platforms.

The challenge, then, is to design mechanisms that efficiently protect user privacy without sacrificing the accuracy of the system, To limit misuse, our goal is to limit accessibility of location information from global visibility to a user's social circle. We identify two main types of queries necessary to support the functionality of these geosocial applications:point queries and nearestneighbor (kNN) queries. Point queries query for location data at a particular point, whereas kNN queries query for k nearest data around a given location coordinate.

To address this challenge, in this paper, we propose location to index mapping), a novel approach to achieving user privacy while maintaining full accuracy in location-based social applications (LBSAs from here onward).

Our insight is that many services do not need to resolve distance-based queries between arbitrary pairs of users, but only between friends interested in each other's locations and data. Thus, we can partition location data based on users' social groups, and then perform transformations on the location coordinates before storing them on untrusted servers. A user knows the transformation keys of all her friends, allowing her to transform her query into the virtual coordinate system that her friends use. Our coordinate transformations preserve distance metrics, allowing an application server to perform both point and nearest-neighbor queries correctly on transformed data. However, the transformation is secure, in that transformed values cannot be easily associated with real-world locations without a secret, which is only available to the members of the social group. Finally, transformations are efficient, in that they incur minimal overhead on the LBSAs. This makes the applications built on LocX lightweight and suitable for running on today's mobile devices.

II. SCENARIOS AND REQUIREMENTS

Here we describe several scenarios we target in the context of emerging Geosocial applications that involve heavy interaction of users with their friends. We use these scenarios to identify the key requirements of a Geosocial location privacy preserving system.

A. Geosocial Application Scenarios

Scenario 1. Alice and her friends are excited about exploring new activities in their city and leveraging the

"friend referral" programs offered by many local businesses to obtain discounts. Alice is currently in downtown and is looking to try a new activity in her vicinity. but she also wants to try an activity that gives her the most discount. The discounts are higher for a user that refers more friends or gets referred by a friend with high referral count. As a result Alice is interested in finding out the businesses recommended by her friends and the discounts obtained through them, within her vicinity. in addition, she is also interested in checking if there are discounts available for her favourite restaurant at a given location.

Scenario 2. Alice and her friends are also interested in playing location-based games and having fun by exploring the city further. So they setup various tasks for friends to perform, such as running a few miles at the Gym, swimming certain laps, taking pictures at a place, or dining at a restaurant. They setup various points for each task, and give away prizes for the friends with most points. For Alice to learn about the tasks available near her, she needs to query an application to find out all tasks from friends near her and the points associated with them. The scenarios above, while fictitious, are not far from reality. Groupon and LivingSocial are some example companies that are leading the thriving business of local activities. SCVNGR [6] offers similar services as locationbased games. But none of these services provide any location privacy to users: all the locations visited by the users are known to these services and to its administrators. Our goal is to build a system that caters to these scenarios and enables users to query for friends' data based on locations, while preserving their location privacy. We want to support: 1) point query to query for data associated with a particular location, 2) circular range query to query for data associated with all locations in a certain range (around the user), and 3) nearest-neighbor query to query for data associated with locations nearest to a given location. Finally, while it is also useful to query for data that belongs to non friends in certain scenarios, we leave such extensions for future.

B. System Requirements

The target scenarios above bring out the following key requirements from an ideal location-privacy service:

- Strong location privacy. The servers processing the data (and the administrators of these servers) should not be able to learn the history of locations that a user has visited. .
- Location and user unlinkability. The servers hosting theservices should not be able to link if two recordsbelong to the same user, or if a given record belongs to a given user, or if a given record corresponds to a certain real-world location.
- Location data privacy. The servers should not be able to view the content of data stored at a location.
- Flexibility to support point, circular range, and nearest-neighbor queries on location data.
- Efficiency in terms of computation, bandwidth, and latency, to operate on mobile devices.

The need for each of these requirements becomes clearer when we describe the related work and their limitations in more detail in the next section. In our proposed system, LocX, we aim to achieve all these requirements.

III. RELATED WORK

A. Prior Work on Privacy in General Location-Based Service

There are mainly three categories of proposals on providing location privacy in general LBSs that do not specifically target social applications. First is spatial and temporal cloaking [11], [12], [13], [22], [15], wherein approximate location and time is sent to the server instead of the exact values. The intuition here is that this prevents accurate identification of the locations of the users, or hides the user among k other users (called k-anonymity [12], [13], [22]), and thus improves privacy. This approach, however, hurts the accuracy and timeliness of the responses from the server, and most importantly, there are several simple attacks on these mechanisms [23], [24], [25], [26] that can still break user privacy. Pseudonyms and silent times [27], [14] are other mechanisms to achieve cloaking, where in device identifiers are changed frequently, and data are not transmitted for long periods at regular intervals. This, however, severely hurts functionality and disconnects users. The key difference between these approaches and our work is that they rely on trusted intermediaries, or trusted servers, and reveal approximate real-world location to the servers in plain text. In LocX, we do not trust any intermediaries or servers. On the positive side, these approaches are more general and, hence, can apply to many location-based services, while LocX focuses mainly on the emerging geosocial applications. The second category is location transformation, which uses transformed location coordinates to preserve user location privacy. One subtle issue in processing nearestneighbor queries with this approach is to accurately find all the real neighbors. Blind evaluation using Hilbert Curves [21], unfortunately, can only find approximate neighbors. To find real neighbors, previous work either keeps the proximity of transformed locations to actual locations and incrementally processes nearest-neighbor queries [28], or requires trusted third parties to perform location transformation between clients and LBSA servers [29]. In contrast, LocX does not trust any third party and the transformed locations are not related to actual locations. However, our system is still able to determine the actual neighbors, and is resistant against attacks based on monitoring continuous queries [30], [31]. The third category of work relies on PIR [16] to provide strong location privacy. Its performance, although improved by using special hardwares [17], is still much worse than all the other approaches, thus it is unclear at present if this approach can be applied in real LBSs.

B. Prior Work on Privacy in GeoSocial Services

For certain types of geosocial services, such as buddy tracking services to test if a friend is nearby, some recent proposals achieve provable location privacy [18], [19] using expensive cryptographic techniques such as secure two party computation. In contrast, LocX only uses inexpensive symmetric encryption and pseudorandom number generators.

The closest work to LocX is Longitude [32], [33], which also transforms locations coordinates to prevent disclosure

to the servers. However, in longitude, the secrets for transformation are maintained between every pair of friends to allow users to selectively disclose locations to friends. As in, longitude can let a user reveal her location to only a subset of her friends. In contrast, LocX has a simpler threat model where all friends can access a user's information and hence the number of secrets that users have to maintain is only one per user. LocX can still achieve location and user unlinkability. In addition, LocX can provide more versatile geosocial services, such as location-based social recommendations, reminders, and others, than just buddy tracking as in the above prior work.

C. Anonymous Communication Systems

These systems, including Tor [34], provide anonymity to users during network activity. One might ask, then, why using Tor to anonymously route data to LBSA servers is not sufficient? This approach seems to provide privacy as the server only sees location data but not the identity of the user behind that data. However, recent research has revealed that hiding the identity of the users alone is not sufficient to protect location privacy. Even if Tor is used, it is possible for an attacker with access to the location data to violate our privacy and unlinkability requirements. For example, using anonymized GPS traces collected by the servers, it has been shown that users' home and office locations, and even user identity can be derived [23], [24], [25], [26]. LocX defends against such attacks and meets all our requirements.

D. Systems on Untrusted Servers

In the context of databases, recent systems proposed running database queries on encrypted data (stored on untrusted servers), using heavy-weight homomorphic [35] or asymmetric encryption [36] schemes. These approaches are suitable for spatial data outsourcing or data mining scenarios where the data are static and are owned by limited number of users. But they are less suitable for LBSAs, where the data are dynamic and personal, and thus cannot be encrypted under a single secret key. In the context of location and social applications, Persona [37] and Adeona [38] also relied on encrypting all data stored on untrusted servers to protect user privacy. Persona focused on privacy in online social networks, and Adeona focused on privacy in device tracking systems where there is no data sharing among users. Applying Persona's mechanisms to LBSAs directly would encrypt all location coordinates, making LBSAs unable to process nearest-neighbor queries. But if location is not encrypted, attacks using anonymized GPS traces, mentioned above, can succeed, making Persona insufficient to protect location privacy. Similarly, Adeona is useful for a user to retrieve her own data, but not the data from her friends. Our contributions complement these systems. Some techniques in these papers can help LocX as well, for example, Persona's approach to partition data shared with friends into fine-grained groups, and Adeona's hardware-assisted approaches to speed up crypto processing.

IV. System Design

A. Basic Design

the server should support different types of queries (point, circular range and nearestneighbor queries) on location data. For the server to be able to do this, we need to reveal the location coordinates in plain text. But doing so would allow the malicious server to break a user's location privacy.

To resolve this problem, we propose the idea of *coordinate transformation*. Each user u in the system chooses a set of secrets that they reveal only to their friends. These secrets include a rotation angle θu, a shift bu, and a symmetric key $symm_u$. The users exchange their secrets via interactions when friends meet in person, or via a separate trusted channel, such as email, phone etc. The secret angle and shift are used by the users to transform all the location coordinates they share with the servers. Similarly, the secret symmetric key is used to encrypt all the location data they store on the servers. These secrets are known only to the friends, and hence only the friends can retrieve and decrypt the data. For example, when a user u wants to store a review r for a restaurant at (x, y), she would use her secrets to transform (x, y) to (x^1, y^1) and store encrypted review E(r) on the server. When a friend v wants to retrieve u's review for the restaurant at (x, y), she would again transform (x, y) using u's secret (previously shared with v), retrieve E(r), and then decrypt it using u's symmetric key to obtain r. Similarly, v would transform (x, y) according to each of her friends' secrets, obtain their reviews, and read them. We only focus on point queries for now. Figure 1 depicts this basic design.

A limitation.

This basic design has one important limitation: the server can uniquely identify the client devices (for *e.g.*, using the IP address). Using this, the server can associate different transformed coordinates to the same user (using the IP). Sufficient number of such associations can break the transformations (as we show in Section 5). So maintaining unlinkability between different queries is critical. One approach to resolve this limitation is to route all queries through an anonymous routing system like Tor [34]. But simply routing the data through Tor all the time will be inefficient. Especially in the context of recent LBSAs, that adds larger multimedia files (pictures and videos) at each location. So we need to improve this basic design to be both secure and efficient.

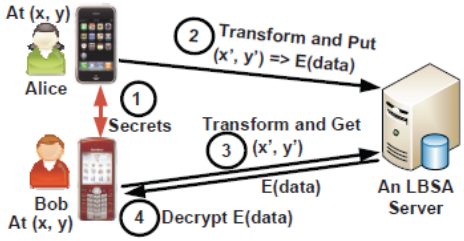


Fig. 1 A basic design

In the basic design
1) Alice and Bob exchange their secrets
2) Alice stores her review of the restaurant (at (x, y)) on the server under transformed coordinates,
3)Bob later visits the restaurant and queries for the reviews on transformed coordinates,
4) decrypts the reviews obtained.

B. Overview of LocX

LocX builds on top of the basic design, and introduces two new mechanisms to overcome its limitations. First, in LocX, we split the mapping between the location and its data into two pairs: a mapping from the transformed *location to an encrypted index* (called **L2I**), and a mapping from the *index to the encrypted location data* (called **I2D**). This splitting helps in making our system efficient. Second, users store and retrieve the L2Is via *untrusted proxies*. This redirection of data via proxies, together with splitting, significantly improves privacy in LocX. For efficiency, I2Ds are not proxied, yet privacy is preserved

1. *Decoupling a location from its data:*
Location data data(x, y) corresponding to the real-world location (x, y) is stored under (x, y) on the server. But in LocX, the location (x, y) is first transformed to (x^1, y^1), and the location data is encrypted into E(data(x,y)). Then the transformed location is decoupled from the encrypted data using a random index i via two servers as follows:
1) an L2I = [(x^1, y^1),E(i)],which stores E(i) under the location coordinate (x^1, y^1), and
2) an I2D = [i,E(data(x,y))], which stores the encrypted location data E(data(x,y)) under the random index i. The index is generated using the user's secret random number generator. We refer to the server storing L2Is as the *index server* and the server storing I2D as the *data server*. We describe these two as separate servers for simplicity, but in reality they can be on the same server, and our privacy properties still hold. This separation of location information into two components (L2I and I2D) helps us continue to efficiently run different types of location queries on L2Is and retrieve only relevant I2Ds.
Figure 2 depicts the design of LocX.

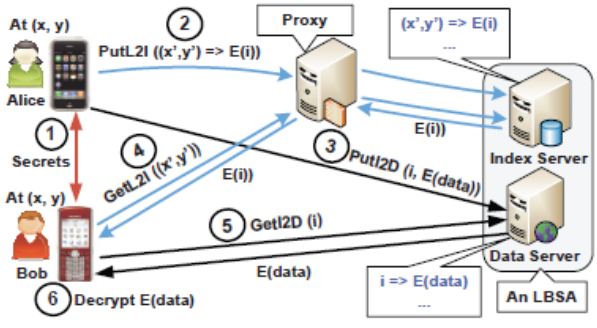


Fig. 2 design of LocX

1) Alice and Bob exchange their secrets, 2) Alice generates and L2I and I2D from her review of the restaurant (at (x, y)), and stores the L2I on the index server via a proxy. 3) She then stores the I2D on the data server directly, 4) Bob later visits the restaurant and fetches for L2Is from his friends by sending the transformed coordinates via a proxy, 5) he decrypts the L2I obtained and then queries for the corresponding I2D, 6) finally Bob decrypts Alice's review.

2. *Proxying L2Is for location privacy:*
Users store their L2Is on the index server via *untrusted proxies*. These proxies can be any of the following: PlanetLab nodes, corporate NATs and email servers in a user's work places, a user's home and office desktops or laptops, or Tor [34] nodes. We only need a one-hop indirection between the user and the index server. These diverse types of proxies provide tremendous flexibility in proxying L2Is, thus a user can store her L2Is via different proxies without restricting herself to a single proxy. Furthermore, compromising these proxies by an attacker does not break users' location privacy, as (a) the proxies also only see transformed location coordinates and hence do not learn the users' real locations, and (b) due to the noise added toL2Is. To simplify the description, for now, we assume that the proxies are non-malicious and do not collude with the index server. But we will later describe our solution in detail to even defend against colluding, malicious proxies. With this high-level overview, we now describe our solution to store and query data on the servers in detail. We also explain the challenges we faced, and the tradeoffs we made in making our solution secure and efficient.

V. Evaluation

A. Implementation and setup

We implemented LocX in Java. We used AES with 128 bits keys for encryption and decryption. The implementation of nearest-neighbor queries was based on the R□-tree package from HKUST [45]. We configured each user to cache 1000 random number tags from each of her friends.
We measured LocX's performance on both desktops The index and data servers were run on the same Dell PowerEdge server equipped with Quad Core Xeon L5410 2.33Ghz CPU, 24GB RAM and 64 bit Federal Core 8 kernels. Clients were run on another machine with the same configuration. We used the same code base for both desktop and mobile tests. But we had to modify the code slightly for Android OS to deal with some missing libraries. In addition, we had to make certain optimizations to limit the memory usage to under 16MBs for LocX process in Android.
Workload. We used both synthetic and real-world LBSA workload datasets for our tests. The synthetic dataset with default parameters was created following empirical observation on popular geo-social sites such as FourSquare: First, we partitioned a two dimensional space into 100 cells, each of which is a city. In each city, we randomly generated 100 pairs of location coordinates. Then we assigned 1000

resident clients to each city. Each client had 100–1000 friends following a power law distribution with $\alpha = 1.5$ [52], among whom 70% friends were from the same city as the client and 30% were from other cities. Each client did 20 location puts, among which 70% puts were at locations in the client's resident city and 30% were at locations in other cities. Each location put message was randomly generated consisting of maximum 140 bytes, following the tweets in Twitter. As a result, each city had 20K location puts on average, and the total number of location puts was 2M. After all the puts, each client submits a point query and a nearest-neighbor query with 70% probability of being within the client's resident city and 30% probability of being in other cities. Each nearest-neighbor query requests for 10 nearest locations (we only evaluate nearest-neighbor queries, as we found in our preliminary tests that the performance of circular range queries to be similar to that of nearest neighbor queries). We set noise to a fixed 10 points per query for now, and study the impact of noise later. We crawled www.brightkite.com for real LBSA traces. We crawled using BrightKite's public APIs, at a rate slower than the rate specified in the API Terms of Use. Due to the slow rate, we distributed the crawling tasks to 20 machines, and crawled for about a month starting from August 20th, 2010. Starting with an initial seed of users, we crawled each user's profile, friends list, and check-in data. The crawled data in total had 25,314 users, 123,438 unique GPS coordinates with 259,775 check-ins by users. While using this data for experiments, we treated each check-in as a location put, and let each user query from one of her check-in locations. Since check-in messages were not available for us to crawl, we generated random messages of varying sizes.

Experiment setup. To evaluate the overhead that our approach is adding to today's LBSAs with no privacy, we compared LocX with random tags, referred to as LocX, with an implementation of a today's service that has social network on the server and directly maps a location to its data, referred to as L2D. In L2D, data is in plain-text, thus no encryption or decryption is needed. We measured the communication costs between clients and servers, the client processing time, the query completion time (including network latency), and the server processing time

B. Experimental Results

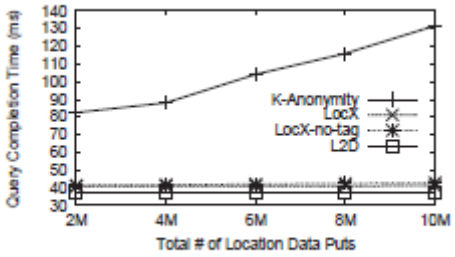
We report results from our tests on desktop computers

Performance of a location put. We present the cost of a single location put in synthetic dataset. A put in today's system (L2D) costs no processing time on clients as there is no crypto operation. But we can see that a put in LocX with encryption and additional index data only slightly increases the overhead, which is not even observable by users. The average message size was 84.5 in L2D, but it was increased to 140 in LocX. k-Anonymity, however, has even higher size due to the information regarding the cloaked spacial region in the message.

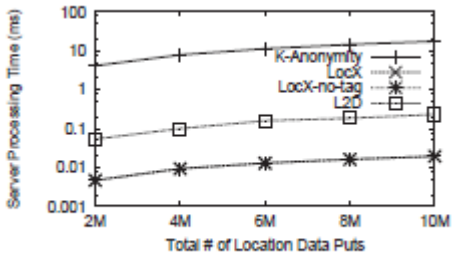
Query performance with increase in the # of puts.

We compared the performance of LocX (with random tags), LocX with no tags, k-Anonymity, and L2D for point queries and kNN queries. On synthetic dataset, we varied the number of location puts per client from 20 to 100, while fixing the amount of noise in a query to default 10 and message size to default maximum 140. Total number of clients was fixed at 100K. As location puts per client increases, the total data size increases, thus more data needs to be processed and the sizes of query answers increase. Figure 3 shows the increase in query answer sizes. Obviously, the response to a kNN query contains more data than a point query (by more than 6 times).

From Figures 3(a) and 4(a), we see that processing a query in LocX takes is comparable to that of L2D, in a LAN IEEE TRANSACTIONS ON MOBILE COMPUTING This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.12 setting. However, the other two approaches – k-Anonymity and 'Locx-no-tag' – perform poorly. k-Anonymity has higher overhead as the entire cloaked spacial region is included in the responses, which leads to increase in the query completion time, and server processing time or load (shown in Figures 3(b) and 4(b)). In 'LocX-no-tag', a client cannot differentiate between friends' and non-friends' messages, so the client tries to decrypt every single message received, which leads to costly computation and time to completion. This problem becomes particularly worse while processing nearest-neighbor queries, as shown in 4(a). The server time of LocX is actually better than L2D due to the fact that the application logic is moved to the clients and server simply needs to do lookups. The communication cost of LocX is no more than 3 times the communication cost of L2D for point queries and no more than 7 times the communication cost of L2D for nearest-neighbor queries, We also measured the client processing times. LocX, as expected, pays a slight processing cost on the client side in decrypting indices and location messages. But we find that this increase in overhead is actually negligible. Due to space limitation, we leave out the graphs for synthetic data. The results are similar in both cases

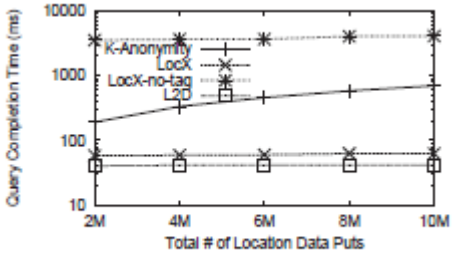


(a) Query Completion Time

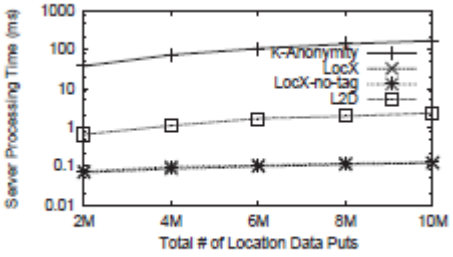


(b) Server Processing Time

Fig.3 The various costs of running point queries while varying the number of location puts in synthetic data



(a) Query Completion Time



(b) Server Processing Time

Fig-4 the various costs of running nearest neighbour queries while varying the number of location puts in synthetic data

VI. CONCLUSIONS

This paper describes the design, prototype implementation, and evaluation of LocX, a system for building location-based social applications (LBSAs) while preserving user location privacy. LocX provides location privacy for users without injecting uncertainty or errors into the system, and does not rely on any trusted servers or components. LocX takes a novel approach to provide location privacy while maintaining overall system efficiency, by leveraging the social data-sharing property of the target applications. In LocX, users efficiently *transform* all their locations shared with the server and encrypt all location data stored on the server using inexpensive symmetric keys. Only friends with the right keys can query and decrypt a user's data. We introduce several mechanisms to achieve both privacy and efficiency in this process, and analyze their privacy properties. Using evaluation based on both synthetic and real-world LBSA traces, we find that LocX adds little computational and communication overhead to existing systems. Our LocX prototype runs efficiently even on resource constrained mobile
phones. Overall, we believe that LocX takes a big step towards making location privacy practical for a large class of emerging geo-social applications.

REFERENCES

[1] M. Motani, V. Srinivasan, and P. S. Nuggehalli, "Peoplenet: engineering a wireless virtual social network," in *Proc. of MobiCom*, 2005.
[2] M. Hendrickson, "The state of location-based social networking," 2008.
[3] P. Mohan, V. N. Padmanabhan, and R. Ramjee, "Nericell: rich monitoring of road and traffic conditions using mobile smartphones," in *Proc. of SenSys*, 2008.
[4] G. Ananthanarayanan, V. N. Padmanabhan, L. Ravindranath, and C. A. Thekkath, "Combine: leveraging the power of wireless peers through collaborative downloading," in *Proc. of MobiSys*, 2007.
[5] M. Siegler, "Foodspotting is a location-based game that will make your mouth water," http://techcrunch.com/2010/03/04/foodspotting/.
[6] http://www.scvngr.com.
[7] B. Schilit, J. Hong, and M. Gruteser, "Wireless location privacy protection,"*Computer*, vol. 36, no. 12, pp. 135–137, 2003.
[8] F. Grace, "Stalker Victims Should Check For GPS," Feb. 2003,www.cbsnews.com.
[9] DailyNews, "How cell phone helped cops nail key murder suspect secret'pings' that gave bouncer away," Mar. 2006.
[10] "Police: Thieves robbed homes based on facebook, social media sites," WMUR News, September 2010, http://www.wmur.com/r/24943582/detail.html.
[11] M. Gruteser and D. Grunwald, "Anonymous usage of location-based services through spatial and temporal cloaking," in *Proc. of Mobisys*, 2003.
[12] M. F. Mokbel, C.-Y. Chow, and W. G. Aref, "The new casper: A privacyaware location-based database server," in *ICDE*, 2007.
[13] B. Gedik and L. Liu, "Location privacy in mobile systems: A personalized anonymization model," in *Proc. of ICDCS*, 2005.
[14] T. Jiang, H. J. Wang, and Y.-C. Hu, "Preserving location privacy in wireless lans," in *Proc. of MobiSys*, 2007.
[15] P. Kalnis, G. Ghinita, K. Mouratidis, and D. Papadias, "Preventing location-based identity inference in anonymous spatial queries," *TKDE*, 2007.
[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 Conference*, 2008.
[17] S. Papadopoulos, S. Bakiras, and D. Papadias, "Nearest neighbor search with strong location privacy," *PVLDB*, 2010.
[18] A. Narayanan, N. Thiagarajan, M. Lakhani, M. Hamburg, and D. Boneh, "Location privacy via private proximity testing," in *Proc. of NDSS*, 2011.
[19] G. Zhong, I. Goldberg, and U. Hengartner, "Louis, lester and pierre:Three protocols for location privacy," in *Proc. of PET*, 2007.
[20] N. Daswani and D. Boneh, "Experimenting with electronic commerceon the palmpilot," in *Financial Cryptography*. Springer, 1999.
[21] A. Khoshgozaran and C. Shahabi, "Blind evaluation of nearest neighbor queries using space transformation to preserve location privacy," in *Proc.of SSTD*, 2007.
[22] G. Ghinita, P. Kalnis, and S. Skiadopoulos, "Prive: anonymous locationbased queries in distributed mobile systems," in *Proc. of WWW*, 2007.
[23] P. Golle and K. Partridge, "On the anonymity of home/work location pairs," in *Proc. of Pervasive Computing*, 2009.
[24] B. Hoh, M. Gruteser, H. Xiong, and A. Alrabady, "Enhancing security and privacy in traffic-monitoring systems," in *IEEE Pervasive Computing Magazine*, 2006.
[25] B. Hoh *et al.*, "Preserving privacy in gps traces via uncertainty-aware path cloaking," in *Proc. of CCS*, 2007.
[26] J. Krumm, "Inference attacks on location tracks," in *Proc. of Pervasive Computing*, 2007.
[27] A. Beresford and F. Stajano, "Mix zones: User privacy in location-aware services," in *Proc. of Pervasive Computing*, 2004.
[28] M. L. Yiu, C. S. Jensen, X. Huang, and H. Lu, "Spacetwist: Managing the trade-offs among location privacy, query performance, and query accuracy in mobile services," in *Proc. of ICDE*, 2008.
[29] D. Lin, E. Bertino, R. Cheng, and S. Prabhakar, "Position transformation: a location privacy protection method for moving objects," in *Proc. Of Security and Privacy in GIS and LBS*, 2008.
[30] C.-Y. Chow and M. F. Mokbel, "Enabling private continuous queries for revealed user locations," in *SSTD*, 2007, pp. 258–275.

[31] E. O. Turgay, T. B. Pedersen, Y. Saygin, E. Savas, and A. Levi, "Disclosure risks of distance preserving data transformations," in *Proc. of SSDBM*, 2008.
[32] S. Mascetti, C. Bettini, and D. Freni, "Longitude: Centralized privacypreserving computation of users' proximity," in *Proc. of SDM*, 2009.
[33] S. Mascetti, C. Bettini, D. Freni, X. S. Wang, and S. Jajodia, "Privacyaware proximity based services," in *Proc. of MDM*, 2009.
[34] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The secondgeneration onion router," in *USENIX Security Symposium*, 2004.
[35] H. Hu, J. Xu, C. Ren, and B. Choi, "Processing private queries over untrusted data cloud through privacy homomorphism," in *ICDE*, 2011.
[36] W. K. Wong, D. W.-L. Cheung, B. Kao, and N. Mamoulis, "Secure kNN computation on encrypted databases," in *SIGMOD Conference*, 2009.
[37] R. Baden, A. Bender, N. Spring, B. Bhattacharjee, and D. Starin,"Persona: An online social network with user defined\ privacy," in *Proc. of SIGCOMM*, 2009.
[38] T. Ristenpart, G. Maganis, A. Krishnamurthy, and T. Kohno, "Privacypreserving location tracking of lost or stolen devices: Cryptographic techniques and replacing trusted third parties with DHTs," in *Proc. Of USENIX Security Symposium*, 2008.
[39] A. Mislove, K. Gummadi, and P. Druschel, "Exploiting social networks for internet search," in *Proc. of HotNets*, 2006.
[40] A. Mislove, A. Post, P. Druschel, and K. Gummadi, "Ostra: Leveraging trust to thwart unwanted communication," in *Proc. of NSDI*, 2008.
[41] T. Isdal, M. Piatek, A. Krishnamurthy, and T. Anderson, "Privacypreserving p2p data sharing with oneswarm," in *SIGCOMM*, 2010.
[42] M. Bellare, R. Canetti, and H. Krawczyk, "Keying hash functions for message authentication," in *CRYPTO96*.
[43] B. Greenstein, D. McCoy, J. Pang, T. Kohno, S. Seshan, and D. Wetherall,"Improving wireless privacy with an identifier-free link layer protocol," in *Proc. of MobiSys*, 2008.
[44] A. Guttman, "R-trees: a dynamic index structure for spatial searching," in *Proc. of SIGMOD*, 1984.
[45] D. P. group, "R-tree java implementation," http://www.rtreeportal.org/code/Rstar-java.zip.
[46] "Privoxy web proxy," http://www.privoxy.org/.
[47] B. Wong, I. Stoyanov, and E. Sirer, "Octant: A comprehensive framework for the geolocalization of Internet hosts," in *Proc. of NSDI*, 2007.
[48] P. Gill *et al.*, "Dude where's that IP? Circumventing Measurement-basedIP Geolocation," in *USENIX Security Symposium*, 2010.
[49] J. Manweiler, R. Scudellari, and L. P. Cox, "Smile: Encounter-based trust for mobile social services," in *Proc. of CCS*, 2009.
[50] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *CCS*, 2006.