Private Sharing of User Location over Online Social Networks

Julien Freudiger, Raoul Neu, and Jean-Pierre Hubaux

School of Computer and Communication Sciences, EPFL, Switzerland
`firstname.lastname@epfl.ch`

Abstract. Online social networks increasingly allow mobile users to share their location with their friends. Much to the detriment of users' privacy, this also means that social network operators collect users' location. Similarly, third parties can learn users' location from localization and location visualization services. Ideally, third-parties should not be given complete access to users' location. To protect location privacy, we design and implement a platform-independent solution for users to share their location in a private fashion over online social networks. Our solution relies on encryption to enforce access control and uses dummy queries and caching to protect localization and location visualization.

1 Introduction

Cellphone users increasingly share their location on the Internet with location-based services (LBSs) or online social networks (OSNs). These third-parties use location information to infer users' context and provide customized services [21]. We focus on *location-sharing services* (LSSs) that enable friends to share their locations with each other [18,27,38,39]: such services combine location-based services with online social networks.

While accessing LSSs, users often rely on two other components: a *localization* component, with which users obtain their location [4,12] and a *visualization* component, with which users render their location on a map and that of their friends [1,5,17,41].

Much to the detriment of users' privacy, third-parties running location-sharing services can collect users' location. Similarly, some localization services locate users based on wireless access points in users' proximity and thus learn users' location. Also, users often visualize their location and that of their friends on *online* maps thus revealing locations to the map operator. Because visited locations are correlated to users' identity, third-parties can not only *localize* users but also *profile* them over time [29,34,36,37]. Online social networks already know users' identity and their social graph, but with location information, they obtain a deeper insight into users' activities.

Ideally, location-sharing services should not be given complete access to users' location. Previous work on location-based services proposed to disclose minimum information relying on obfuscation and anonymization [20,30,32,33,34,35,42,44].

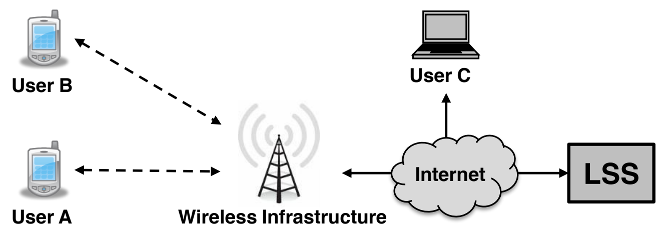


Fig. 1. System model. Users upload their location to a central server that provides location-sharing services. Users A, B and C can then see each others location.

These solutions aim at reducing the information shared with LBSs while still proposing a minimum quality of service. In the case of LSSs, the requirements differ: users aim at sharing their location with friends over third-parties that do not provide specific services but only facilitate users' interactions. Thus, obfuscation and anonymization are not viable options.

In this work, we design a platform-independent solution for users to share their location in a private fashion over LSSs. Our solution does not require any changes to the infrastructure and can operate with any third-party LSS. We implement our solution as a prototype application for mobile phones code-named PrivL (Private Locations). It uses encryption to protect users' location from third-party servers while still allowing them to share it with friends. It can also make use of distributed Hash tables (DHTs) to provide ephemeral data storage [28]. We also discuss the problem of protecting privacy with respect to localization and map visualization services and propose a solution based on dummy queries and caching. We discuss the challenge of generating dummy queries to confuse an adversary and use the concept of *virtual identities*.

2 Models

2.1 System Model

We study a network (Fig. 1) that involves mobile users equipped with wireless devices, third-party operators running location-sharing services and a wireless infrastructure. Let us define $\mathcal{U} = \{U_1, U_2, ..., U_n\}$ as the set of users in the network, where n is the total number of users. For simplicity, we consider that each user owns a single communication device. We study a discrete time system with initial time $t = 0$ and consider the location of users at each discrete time t.

Wireless devices feature localization technology such as GPS, or wireless triangulation that lets users locate themselves [4,12]. The geographic *location* of a user is denoted by $l = (lon, lat)$, where lon is the longitude, and lat is the latitude. The wireless infrastructure relies on technology such as WiFi, GSM or 3G to let users connect to the Internet. LSSs are operated by independent third-party entities that intend to provide services based on users' location.

Cellphone users send their location to an LSS operator through the wireless infrastructure. For each request sent, users may have to identify themselves to the LSS using proper credentials. For example, some services may require users to have an account, and to provide the corresponding username and password with each request, whereas other services may be open to everyone and use HTTP cookies to recognize several visits from the same user. In general, we assume that users are identified with *pseudonyms* (i.e., fictitious identifiers), such as their username, their HTTP cookie or their IP address. In this work, we focus on LSSs enabling users to share their location with friends such as Google Latitude [38], FireEagle [26], and Loopt [39]. We define a function $F(U_i)$ that provides the set of friends of every user U_i.

Each location sample shared by a user is called an *event*. Each event is denoted by a triplet $< i, t, l >$, where i is the pseudonym of a user, t is the time instance at which the event occurred, and l is the location of the event. We consider a simple communication model in which users upload their location to LSSs with a post request:

1. U: Compute location l
2. $U \rightarrow LSS$: Post $< i, t, l >$
3. $LSS \rightarrow U$: Ack

Users can obtain the location of their friends with a get request:

1. $U \rightarrow LSS$: Get location of $F(U)$
2. $LSS \rightarrow U$: $< j, t, l >, \forall j \in F(U)$
3. U: Visualize on online map

Users then visualize their location and that of their friends on an online [1,5,17] or offline [41] map based on event triplets.

2.2 Threat Model

In this work, we investigate the ability of third-parties to learn users' location. We consider the potential privacy threat of each component of LSS applications. In the localization component, a third-party operator running the localization service learns users' location [13]. In the sharing component, LSS operators passively collect information about the locations of pseudonymous users over time. In the visualization component, a third-party operator running an online map learns users' location.

We consider that the adversary passively collects data from users and may be active or retro-active [28]: the adversary can interact with users to obtain their *current* location or interact with the servers storing the location data to obtain their *past* location. The adversary knows the probability that users visit specific locations on a map based on statistical information obtain from various sources such as Citysense [2], geographic information systems [23], or the Census bureau [29]. The adversary may also host a social network (e.g. Loopt [39]) and thus knows users' social graph and identity.

3 Our Solution

We rely on well-known privacy-preserving mechanisms to protect the privacy of LSS users: we use broadcast encryption to share users' location over third-parties and dummy queries with caching to obtain and visualize users location. Our solution does not require any changes from the infrastructure, and can operate with any LSS. We implement these mechanisms in a prototype application for mobile phones.

3.1 Privacy-Preserving Location Sharing

We propose to use encryption mechanisms to protect data privacy. We make use of cellphones communication capabilities to establish the keying material. We also suggest the use of ephemeral storage to thwart retroactive adversaries.

Encryption Mechanisms Cryptographic mechanisms can limit the access of LSSs to location information. By encrypting the location information, only qualified users (e.g., friends) can decrypt it: a user U_i can share his location only with his friends $F(U_i)$. To do so, we use *Broadcast encryption* schemes [25]. The challenge of broadcast encryption is to limit the size of the communication packet, of the encryption overhead, the number of keys and to allow for the revocation of users. Several solutions exist offering various tradeoffs among these constraints.

Symmetric Scheme The simplest broadcast encryption scheme consists in using symmetric cryptography: U_i establishes a pair-wise secret with each of his friends. User U_i then posts its location encrypted with the pair-wise secrets.

Still, this approach does not scale well: the number of key establishments, the packet size and the computation overhead increase linearly with the number of friends.

Trivial Asymmetric Scheme Assume that each user owns a public and private key pair. A simple broadcast encryption scheme based on asymmetric cryptography consists in using *hybrid encryption*: location information is encrypted with a secret and the secret is encrypted with the public keys of each user.

This scheme simplifies the establishment and management of secrets for groups of users. Still, as before, it generates packet sizes and computation overhead linear in the number of friends.

Dynamic Scheme Recently, advanced schemes were proposed achieving shorter cipher texts and key length [22] and offering to dynamically add new users to the set of friends at a low cost [24]. Nevertheless, such schemes still lack efficient implementations and are thus not easy to deploy in practice.

In LSSs, we believe that users will most of the time share their location with a subset of all their friends. Consequently there will be a small number of destinations, and as shown in [24], the trivial asymmetric scheme and the symmetric scheme are preferable in such case. Hence, our solution makes use of the symmetric scheme and the trivial asymmetric scheme implemented with OpenSSL.

Key Establishment Secret keys can be retrieved from a central repository or directly exchanged between users when they are in physical proximity using peer-to-peer wireless communications.

In this work, we consider that users communicate their keys in a peer-to-peer fashion using Bluetooth, text messages (SMSs), or phone calls. In other words, we rely on the existing cellular infrastructure or physical proximity to guarantee the authenticity of the keying material.

Distributed Hash Tables (DHTs) The encrypted location data can be stored directly on the third-party LSS. However, a retro-active adversary may try to obtain the location data from the LSS in the future. As the data is stored encrypted, the retro-active adversary will also have to obtain the secret used for encryption to obtain users' location.

The encrypted location data can alternatively be stored on an ephemeral medium such as a DHT. Distributed Hash Tables are a peer-to-peer (P2P) storage network consisting of multiple participating nodes. They offer a natural time to live (TTL) of data as peers enter and exit the DHT over time. DHTs have two advantages: the encrypted location information cannot be obtained by a retro-active adversary [28] and if an LSS refuses to host encrypted material, we can upload on the LSS links to a DHT. In other words, DHTs offer a temporary storage solution in which the LSS becomes a transparent server that maintains the relation with friends but does not contain any location information.

In our solution, users can choose to upload their location either on a DHT or on the LSS directly.

3.2 Privacy-Preserving Localization

Most cell phones are equipped with GPS and can locate themselves privately. Yet, GPSs are battery consuming, work poorly indoors and are slow to obtain users' location. Hence, mobile devices sometimes rely on wireless signal triangulation for localization: users reveal contextual information such as the list of WiFi access points in proximity to a localization server that computes users' location [4,12].

A mobile device can protect its location privacy with respect to localization servers by altering localization requests in two ways: i) modify the list of WiFi access points in proximity or ii) send multiple queries in parallel. The first solution will degrade the localization precision. The second solution can be implemented using dummy localization requests and caching.

Caching In order to avoid contacting localization servers altogether, users can cache all access points locations in predefined regions of interest. We use publicly available sources of information to create such caches. In practice, users tend to have repeated mobility patterns, and caching should thus significantly reduce the number of queries sent to localization servers.

However, caching may not scale well as mobile devices may not have enough memory to store all the required information. Similarly, the cache may not contain localization information for some regions because the publicly available source does not contain localization information about that region. In this case, dummy requests could be used.

Dummy Queries Following the principle of k-anonymity, users can create $k-1$ queries to the localization server. This way, the server will be uncertain as to where exactly the user is. To create queries, users can rely on public databases of existing access points [16]. The adversary may still be able to infer the real location of users out of the k queries using statistical inference attacks. We distinguish between two scenarios: users can be *traceable* or *untraceable*.

Traceable Virtual Identities Users are often traceable by third-party providers based on their IP address. The IP address tends to remain the same on data connections of mobile networks (3G and GSM). HTTP cookies can also be used by third-parties to link multiple interactions with the same user.

By linking multiple user requests, the localization service can statistically obtain users' real location out of the k possible locations by analyzing queries over time. For example, if $k-1$ locations of dummy queries are chosen at random, the adversary can estimate that they do not correspond to real *human-generated* queries based on the probability of visiting certain locations.

To avoid such attacks, users can choose $k-1$ queries in a strategic fashion. For example, users can select frequently visited locations. Nevertheless, the adversary may still statistically determine if locations in queries correspond to *human mobility* by looking at the mobility patterns of visited locations. To avoid such inference, users have to maintain a set of *virtual* identities. Each virtual identity will have its own querying profile, matching human mobility statistics.

In summary, virtual identities must respect a number of constraints. *Spatially*, the locations visited by virtual identities should correspond to realistic human locations. *Temporally*, the distance between two consecutive locations should be feasible according to the average speed of humans. Finally, the mobility of virtual identities should *statistically* match traditional mobility statistics.

Untraceable There are multiple techniques to reduce user traceability. Users can rely on DHCP to force IP address changes, or use Tor to obtain a larger IP addresses anonymity set. Similarly, users can erase their HTTP cookies.

If users are not traceable, the localization server must do a double inference. It must first estimate the relation between multiple user requests over time and then statistically derive users' possible locations.

3.3 Privacy-Preserving Visualization

In the case of online maps, the visualization of location information reveals users' location and that of his friends to the map operator. We suggest to make use of dummy map requests and local map caching to protect location privacy.

A user U_i must render the map of his own location and also render the map corresponding to location of his friends: $< j, t, l > \forall U_j \in F(U_i)$. We call m_j the map request to the online map provider containing the *lat*, *lon* coordinates of U_j's location, a radius and a zoom level.

Caching Again caching can be used to reduce the number of queries sent to the map server. A priori, users can define regions of interest where map caching is done. The size of the maps being quite large, this solution may also have problems to scale.

Dummy Queries Inspired by the k-anonymity principle, U_i can use dummy m's to confuse the map operator. In this setting, dummy queries must protect the location privacy of user U_i *and* that of U_i's friends as well.

Like in the localization case, the adversary can use statistical information to infer the real location of a user out of the k queries. In the map visualization scenario, the adversary can even correlate the mobility of a user with that of his friends.

Social Virtual Identities To avoid such attacks, users can maintain sets of virtual identities for them and their friends that have a *social life.* Indeed, the *interaction* between virtual identities must also be modeled to appear as a realistic profile for the adversary.

In our prototype application, we implement a dummy request algorithm that chooses $k - 1$ other queries from a set of frequently visited locations. We also cache every query made by the user locally. We are currently investigating the use of virtual identities in our application.

3.4 Implementation

In order to test the feasibility of the proposed privacy-preserving mechanisms, we implemented a prototype client running on mobile phones code-named PrivL.[1]

We have chosen the Symbian platform because it is a popular platform for mobile phones and it supports QT [10], a cross-platform application framework (Fig. 2). We show in Fig. 3 (a) & (b) two screenshots of the resulting application. Note that in the settings, users can establish security associations, connect to LSSs, give a preference on the positioning mode and privacy level.

[1] The source code will be open sourced and available at privl.sourceforge.net .

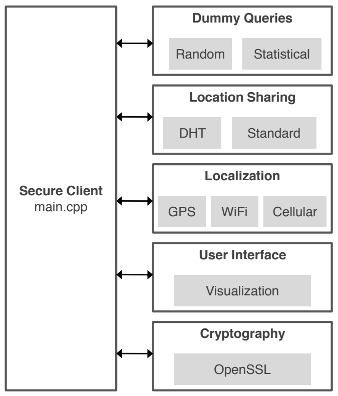


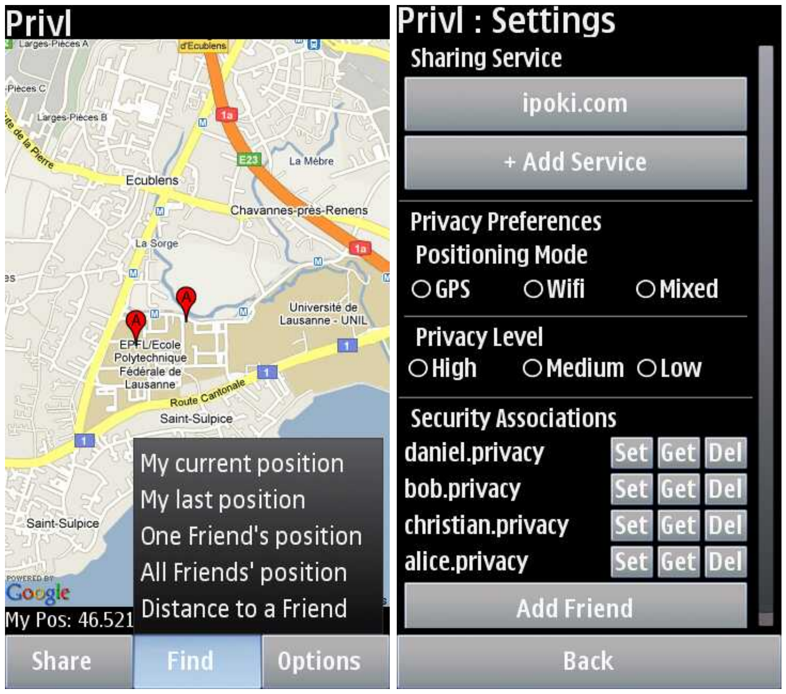
Fig. 2. Implementation overview.

Architecture The current version of the application uploads users' location on any third-party LSS. We have currently implemented an interface for ipoki [6] because it provides an open and detailed API. Users must first login in order get their Friend list and to post their location. We provide users with a new feature which allows them sharing their encrypted location: they upload the ciphertext to a DHT and a reference to it on ipoki's servers. Users have to generate random pair-wise AES session keys and share them with friends using peer-to-peer communications. Cryptographic operations as well as dummy queries are completely transparent to users and do not affect the performances of the LSS application. Our prototype was successfully tested on multiple standard mobile devices.

Technical Details To obtain users' current position, we send nearby Cell Tower information and the MAC addresses of nearby WiFi access points to the Google Geolocation API [4]. While WiFi scanning is implemented with the SDK API plug-in from Symbian C++ [15], the Cell Tower information is provided by the QT Mobility API. The latter API also provides GPS support.

The localization and the visualization component require both caching of data. While this is achieved in the first case by downloading all available access points from Wigle.net within a user defined region, map caching is done by activating the browsers native cache. These two components also make use of dummy queries to enhance privacy. To make sure that dummy positions are valid, we use locations from the list of WiFi access point of Wigle.net. Users' position is shown on a map in a Webkit based browser. The map data is retrieved from Google's static map API [14] and we position placemarks with Javascript.

To provide ephemeral storage, we currently use the academic P2P Network OpenDHT [8]. The interaction with this network is done according to the XML-RPC protocol. The encryption of the user's position relies on the AES Symmetric



(a) (b)

Fig. 3. Interface overview. (a) Main window showing users' location with markers. (b) Settings window enabling users to establish security associations, to connect to LSSs and to setup their privacy requirements.

encryption algorithms which are provided by the EVP module of OpenSSL from Nokia's OpenC plugin.

The Graphical User Interface is implemented with QT's GUI framework: it is optimized for Symbian 5th Edition devices with touch screen. We rely also on the QT Mobility API for the connection to the Internet (WiFi or 3G) and for the interactions with the short message service (SMS).

4 Related Work

There are numerous types of location-based services [18,21,27,39] offering users to connect with friends, to discover their environment or to optimize their mobility. Hence, users can share their location in return for services. In this paper, we focus on services that enable users to share their location with friends.

The need for selective access control in online social networks (OSNs) has been identified in previous works and several solutions were proposed. Most of these solutions implement add-ons to existing OSNs in order to protect user privacy. NOYB [31], for example, encrypts personal information using a pseudo-random substitution cipher. FlyByNight [40] protects private data in Facebook by storing it in encrypted form. We rely on an encryption primitive similar to that discussed in [19] to encrypt data on LSSs.

Recently, a new breed of privacy-aware online social networks were proposed [3,11]. Basically, these OSNs offer precise settings for users to control their privacy. Effectively, all user data is still shared with the third-party running the OSN. Other works proposed fine-grained control over the sharing of user location [7]. This controls enable users to specify precisely with which friends they wish to share their location and how. In this work, we focus on the protection of users' location with respect to third-parties.

Earlier work on privacy preserving localization tried to solve the problem by caching on mobile devices the location of WiFi access points used for triangulation [9]. However, it is costly to store on mobile devices entire database of access points, and such databases may not offer a complete coverage of the regions visited by users. In this work, we extend these existing mechanisms by offering the ability to dynamically cache access points, and use dummy queries when areas are not covered.

The generation of dummy location samples was previously evaluated [43] but mostly with a focus on moving trajectories over a period of time. In this work, we identified several new constraints that must also be taken into account, in particular, we mentioned the use of virtual identities.

5 Conclusion

We have considered the problem of sharing user location privately over online social networks. We show that encryption techniques can be used to protect user location from LSSs. We note that this is not sufficient because other components, such as localization and map visualization services, may also obtain users' location. We show how dummy queries and caching can help protect location privacy. We observe that dummy queries generated at random are inefficient against an adversary with statistical knowledge about user mobility. In the worst case scenario (i.e., users are traceable), we argue that it is necessary to maintain virtual identities generating dummy queries that are statistically indistinguishable from real users. We implemented the ideas of this paper in a prototype client application code-named PrivL that can connect to any LSS.

In the future, we intend to test various dummy query and caching strategies and evaluate their effectiveness in providing location privacy. We will also evaluate the incentives of LSS operators to provide services to users that hide their location. In particular, users could strategically reveal sample of their visited locations or obfuscated location information to provide incentives to LSS operators.

References

1. Bing maps. http://maps.bing.com.
2. Citysense. http://www.citysense.com.
3. Glympse. http://glympse.com.
4. Google geolocation api. http://code.google.com/apis/gears/api_geolocation.html.
5. Google maps. http://maps.google.com.
6. ipoki. http://www.ipoki.com.
7. Locaccino. http://locaccino.org.
8. OpenDHT. http://www.opendht.org.
9. A privacy-observant location system. http://www.placelab.org.
10. Qt. http://qt.nokia.com.
11. Rallyup. http://www.getupandrally.com.
12. Skyhook wifi positioning system. http://www.skyhookwireless.com.
13. Spotrank. http://www.skyhookwireless.com/spotrank.
14. Static maps api. http://code.google.com/apis/maps/documentation/staticmaps.
15. Symbian c++ sdk api plugin. http://wiki.forum.nokia.com/index.php/SDK_API_Plug-in.
16. Wireless geographic logging engine. http://wigle.net.
17. Yahoo maps. http://maps.yahoo.com.
18. Aka Aki. The discover of a lifetime. http://www.aka-aki.com.
19. F. Beato, M. Kohlweiss, and K. Wouters. Enforcing access control in social network sites. In *HotPETs*, 2009.
20. A. R. Beresford and F. Stajano. Mix zones: User privacy in location-aware services. In *PerSec*, March 2004.
21. Google Mobile Blog. http://googlemobile.blogspot.com/2010/01/finding-places-near-me-now-is-easier.html.
22. D. Boneh, C. Gentry, and B. Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In *CRYPTO*, pages 258–275, 2005.
23. Cloudmade. Makes maps differently. http://cloudmade.com.
24. C. Delerablée, P. Paillier, and D. Pointcheval. Fully collusion secure dynamic broadcast encryption with constant-size ciphertexts or decryption keys. In *Pairing*, pages 39–59, 2007.
25. A. Fiat and M. Naor. Broadcast encryption. In *CRYPTO*, pages 480–491, 1994.
26. Yahooo Fireeagle. Take your location to the web. http://fireeagle.yahoo.net.
27. Foursquare. Check-in, find your friends, unlock your city. http://foursquare.com.
28. R. Geambasu, T. Kohno, A. Levy, and H. M. Levy. Vanish: Increasing data privacy with self-destructing data. In *USENIX Security Symposium*, 2009.
29. P. Golle and K. Partridge. On the anonymity of home/work location pairs. In *Pervasive*, 2009.
30. M. Gruteser and D. Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *MobiSys*, 2003.
31. S. Guha, K. Tang, and P. Francis. Noyb: privacy in online social networks. In *WOSP*, pages 49–54, 2008.
32. B. Hoh and M. Gruteser. Protecting location privacy through path confusion. In *SECURECOMM*, 2005.
33. B. Hoh, M. Gruteser, R. Herring, J. Ban, D. Work, J.-C. Herrera, A. M. Bayen, M. Annavaram, and Q. Jacobson. Virtual trip lines for distributed privacy-preserving traffic monitoring. In *MobiSys*, 2008.

34. B. Hoh, M. Gruteser, H. Xiong, and A. Alrabady. Enhancing security and privacy in traffic-monitoring systems. *Pervasive Computing*, pages 38–46, 2006.
35. B. Hoh, M. Gruteser, H. Xiong, and A. Alrabady. Preserving privacy in GPS traces via uncertainty-aware path cloaking. In *CCS*, 2007.
36. iPhone Dev Center. App store tip: Enhance your app with core location. http://developer.apple.com/iphone/news/archives/2010/february/.
37. J. Krumm. Inference attacks on location tracks. In *Pervasive*, 2007.
38. Google Latitude. See where your friends are right now. http://www.google.com/intl/en_us/latitude/intro.html.
39. Loopt. Discover the world around you. http://loopt.com.
40. M. M. Lucas and N. Borisov. Flybynight: mitigating the privacy risks of social networking. In *WPES*, pages 1–8, 2008.
41. Ovi Maps. Navigation on your nokia. for free. forever. http://maps.nokia.com/ovi-services-and-apps/ovi-maps.
42. M. F. Mokbel, C.-Y. Chow, and W. G. Aref. The new casper: Query processing for location services without compromising privacy. In *VLDB*, 2006.
43. T. You, W. Peng, and W. Lee. Protecting moving trajectories with dummies. In *PALMS*, 2007.
44. S. Zhong, L. E. Li, Y. G. Liu, and Y. R. Yang. Privacy-preserving location-based services for mobile users in wireless networks. Technical report, State University of New York at Buffalo, 2005.