Coordinate Transformation – A Solution for the Privacy Problem of Location Based Services?

Andreas Gutscher

Universität Stuttgart
Institute of Communication Networks and Computer Engineering (IKR)
Stuttgart, Germany
gutscher@ikr.uni-stuttgart.de

Abstract

Protecting location information of mobile users in Location Based Services (LBS) is a very important but quite difficult and still largely unsolved problem. Location information has to be protected against unauthorized access not only from users but also from service providers storing and processing the location data, without restricting the functionality of the system. This paper discusses why existing privacy enhancing techniques are insufficient to solve this problem and proposes a new approach basing on coordinate transformations. It shows how location information can be rendered illegible in such a way that it is still possible to perform processing operations required by LBS.

1 Introduction

Location Based Services (LBS) and Context Aware Systems (CAS) try to improve applications and services by adapting them to the location and the context of mobile users and objects in their environment.

Context information, especially location information of persons, is very sensitive data, because it may be possible to extract information like activities, habits and preferences of tracked users from it. Thus, protection of location information is an essential requirement in order to achieve user acceptability. However, there is a lack of effective solutions to solve these privacy problems arising when storing and processing location data in LBS and CAS.

In chapter 2 we give a closer description of the problem and list current approaches in related work as well as their problems in chapter 3. We describe then our approach basing on coordinate transformations in chapter 4, as well as known problems of this technique and solution approaches in chapter 5.

2 Problem Description

The Nexus [8] platform was used as an example to study the problem described below. We can nevertheless presume that the basic problem exists in all LBS with similar requirements.

2.1 LBS Scenario

The LBS consists of several location-aware client applications, typically running on mobile devices, and a platform service providing these client applications with the required location information. Apart from answering queries for the location of objects, the platform should also be able to notify applications whenever certain event conditions occur (this is useful for example to implement a reminder service or a friend finder). The platform thus consists of two services:

The Location Service (LS) [7] stores the location of static and mobile objects and answers requests from clients. Mobile users send location updates to the LS whenever their current location has changed significantly.

The Event Service (ES) [2] handles event registrations from clients, monitors the location information stored on the LS, and notifies the corresponding clients whenever an event condition becomes true.

Although LBS typically offer a richer interface for applications, we restrict our considerations to the following four basic request types:

Position Query: Query for the position of a certain object, i. e., "Where is object X?".

1-4244-0054-6/06/$20.00

Range Query: Query for the list of all objects within a certain area, i. e., "Which objects are located area A?".

On-Enter-Area Event: Notification request for the event that an object enters a certain area, i. e., "Notify me when object X enters area A!".

On-Meeting Event: Notification request for the event that the distance between two objects becomes smaller than a specified distance, i. e., "Notify me when the objects X_1 and X_2 meet!".

2.2 Attacker Model

In current LBS, the protection of location information typically relies on traditional access control mechanisms enforced by the service providers, but no additional privacy enhancing mechanisms are applied.

However, in large and rather open LBS operated by multiple service providers, it cannot be assumed in general, that all users of the system will trust in the benevolence and competence of all service providers. Hence, we argue that the location information has to be protected not only against access of unauthorized users and third parties but also *against the providers of the LS and the ES*.

For simplicity, we analyze only the worst case secanrio in which all providers are considered to be untrustworthy. Scenarios involving both trusted and untrusted providers will not be discussed.

2.3 Requirements

We are thus looking for a solution which ideally should

1. securely protect the location information of the users against unauthorized access of other users, third parties as well as the LS and ES providers,
2. *not* restrict the functionality of the LBS and
3. keep the additional processing, storage and communication costs as low as possible.

Requirements 1 and 2 obviously create a non-trivial problem: On the one hand, the protection mechanism must prevent the service providers from accessing and abusing the location information without any restraint, but on the other hand the LS and ES must be able to store and process this data. In order to find a solution to this contradistinction, the privacy problem has to be take into account already when designing the procedures for storing and processing the location data [6].

According to requirement 2, the services must be able to process queries and event requests. They must therefore be able to decide for example whether or not an object resides within a given area or not. Thus, it is obviously not possible to find a 'strong' protection scheme because the service providers *have to* be able to determine the *relative* position of objects and areas to each other. Nevertheless, we want to find a possibility to protect at least the most sensitive part of the location information: the *absolute* position of the users.

3 Related Work

Several techniques for privacy protection in LBS have been proposed. However, none of them offers a satisfactory solution for a secure protection of location information without restricting the functionality of the system.

3.1 Reduced Accuracy

A simple approach is to reduce the spatial or temporal accuracy of the location data [5] (e. g., choosing a low resolution or adding a random error). As the quality of the data is reduced for attackers as well as for the applications, the technique is not applicable for applications which need more accurate data. Moreover, the protection level will be quite low, even if rather inaccurate data is sufficient.

3.2 Application Zones

In some scenarios (e. g., location-based games [9]) users will use location-aware applications only as long as they stay in certain areas, the so-called *application zones* [3]. Thus, it is sufficient to store the location information of mobile users only as long as they are located within an application zone.

This approach undoubtly decreases the amount of data collected, but many location-aware applications will show their real benefits especially if they are *not* restricted to small areas. It will thus not be applicable for all applications. While inside an application zone, the location information remains unprotected and can nevertheless be privacy-critical.

3.3 Data Encryption

A rather classical approach is to encrypt the location information before it reaches the database of the LS. The corresponding key can be made accessible to

authorizes users so that they can decrypt the response of position queries.

Encryption guarantees the secure protection of data, but processing the data becomes almost impossible. There exist propositions for order preserving encryption schemes [1] and aproaches to carry out simple arithmetic operations on encrypted data [4]. However, these schemes do not allow services to calculate the distance between two points and are therefore unsuitable to fullfil requirement 2.

The services will thus be unable to answer range queries and to detect events. If the users cannot do without these types of requests, then the client can use a local event service which has to periodically retrieve the position of all 'encrypted' objects and process range queries and events from applications by combining the decrypted location information with the not encrypted data accessible to the platform services. The big drawback of this approach is that processing range queries and events becomes quite inefficient due to the large number of required queries.

3.4 Dummy Data

An other approach for privacy protection is to mix the correct location information with a certain proportion of dummy data [3] so that the service providers are unable to distinguish between them. The necessary hint for authorized users to make this distinction can be provided in encrypted form in the meta-information of the location information.

The additional dummy data creates significant overhead, it multiplies the amount of data stored in the LS database, the number of objects returned for range queries and the frequency of events reported to the clients. Keeping the proportion of dummy data small is not easy because the dummy data must withstand attackers testing the data for plausibility, e. g., by calculating the average speed of the object between two successive location updates. Another sore spot is the fact that an attacker can determine where the user has *not* been.

3.5 Pseudonymization

Pseudonymization is a well-established privacy enhancing technique and can definitively contribute to protect the users' privacy in LBS. Users choose identifiers which do not link to their identity. When using services of a LBS, users always use solely these so-called *pseudonyms* or *virtual identities (VIDs)*. Thus, the service providers will be unable to find out (although they have access to the location information) because they cannot link the users to the VIDs. It is sufficient to reveal the pseudonym to all authorized users in order to enable other users to send position queries, register events and to interpret the responses of range queries.

Nevertheless, there are some drawbacks, though. An attacker can extract information (e. g., the place of residence and work) out of location traces of VIDs and try to link it to the identity of the user. This risk can be reduced by switching to a new VID from time to time [3] in order to avoid or break linkages between identity and VIDs.

3.6 Coordinate Transformation

In [10], Treu et al. propose to use coordinate transformations in LBS. However, in this approach all users share one single transformation function, it is thus only suitable for closed user groups in which all members trust each other. (The question whether a user can be member in more than one group is not discussed.) Moreover, event evaluation is limited to *relative* positions of the members towards each other, it ist not possible to process events refering to the *absolute* position of a user or an object. Thus, this approach does not meet the requirements described in chapter 2.

4 Privacy Protection using Coordinate Transformations

We propose to protect the location data before sending it to the LS by 'obfuscating' the location coordinates with the help of coordinate transformations. In order to retain the functionality of the system, the handling of the requests and responses has to be adapted in the LS, the ES and in the client.

4.1 Coordinate Transformations

A coordinate transformation is a mapping which converts the coordinates of any point in one coordinate system into coordinates of the same point in an other coordinate system. Let $\vec{c}_{p,A}$ be the coordinates of a point $\vec{p}$ in the coordinate system k_A and $\vec{c}_{p,B}$ the coordinates of $\vec{p}$ in k_B, then the transformation $t_{B,A}(\vec{x})$ converts the coordinates from k_A to k_B:

$$\vec{c}_{p,B} = t_{B,A}(\vec{c}_{p,A})$$

In the illustration in figure 1, the transformation function $t_{B,A}(\vec{x})$ is represented for simplicity by a single, two-dimensional translation $t_{B,A}(\vec{x}) = \vec{x} + \vec{d}_{B,A}$.

We propose to use transformations which may consist of arbitrary compositions of

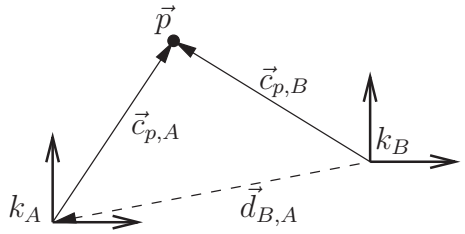

Figure 1. Representation of a point in two coordinate systems

- rotations: $t_{A,B}(\vec{x}) = R_{A,B} \cdot \vec{x}$ and
- translations: $t_{A,B}(\vec{x}) = \vec{x} + \vec{d}_{A,B}$

where $R_{A,B}$ represents a rotation matrix, e. g.,

$$R_{A,B} = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix}$$

and $d_{A,B}$ a translation vector (either for two-dimensional or three-dimensional coordinate systems). These transformations are *global isometries*, thus, each transformation $t_{B,A}$ has the following properties:

1. $t_{B,A}(\vec{x})$ is bijective, i. e., there exists an inverse transformation $t^{-1}_{B,A}(\vec{x}) = t_{A,B}(\vec{x})$
2. $t_{B,A}(\vec{x})$ is a distance and angle preserving map, i. e., the distances between points and the angles between lines remain invariant.
3. Compositions of these transformations can be represented as one transformation consisting of only one rotation and one translation: $t_{C,A}(\vec{x}) = t_{C,B}(t_{B,A}(\vec{x})) = R_{C,A} \cdot \vec{x} + \vec{d}_{C,A}$

Property 1 guarantees that the original location can be restored with the inverse transformation. Due to property 2, it is possible to calculate the distance between two objects, $\vec{p}_1$ and $\vec{p}_2$, in the coordinate system k_A solely from the coordinates from the coordinate system k_B, $\vec{c}_{p_1,B}$ and $\vec{c}_{p_2,B}$, without knowing the transformation $t_{B,A}(\vec{x})$. Property 3 ensures that the composition of two transformations $t_{C,A}(\vec{x}) = t_{C,B}(t_{B,A}(\vec{x}))$ can be disclosed without revealing the original transformations $t_{B,A}(\vec{x}) = R_{B,A} \cdot \vec{x} + \vec{d}_{B,A}$ and $t_{C,B}(\vec{x}) = R_{C,B} \cdot \vec{x} + \vec{d}_{C,B}$, because it is not possible to calculate the coefficients $R_{B,A}, R_{C,B}, d_{B,A}$ and $d_{C,B}$ from $R_{C,A}$ and $\vec{d}_{C,A}$.

4.2 Protection of Location Information

In conventional LBS all coordinates refer to a single, well-defined coordinate system (k_0). In order to protect location information, users should not send their location $\vec{p}_1$ in coordinates for k_0 (i. e., $\vec{c}_{p_1,0}$), but in coordinates for a different coordinate system k_A. These coordinates can be calculated with the corresponding transformation function $t_{A,0}(\vec{x})$:

$$\vec{c}_{p_1,A} = t_{A,0}(\vec{c}_{p_1,0})$$

The user then sends $\vec{c}_{p1,A}$ and a corresponding coordinate system identifier (*CSID*) A to the LS, but *not* the transformation function itself. The LS is unable to map coordinates for k_A to k_0 because it is unable to transform the coordinates back to k_0 without $t_{A,0}(\vec{x})$. By revealing $t_{A,0}(\vec{x})$ to all users authorized to access his location, the user gives them the possibility to interpret coordinates of k_A. The transformation function thus plays the role of a *shared secret* similar to a symmetric cryptographic key and can be distributed with the help of conventional key distribution protocols.

In the following we show how the four different requests presented in chapter 2 can be processed.

4.3 Position Query

The user sends a position query for a certain object to the LS. The LS returns the coordinates $\vec{c}_{p,A}$ and the corresponding CSID A to the user. If the user has been authorized by the object, he can select the corresponding inverse transformation function $t^{-1}_{A,0}(\vec{x}) = t_{0,A}(\vec{x})$, and transform the coordinates back to k_0:

$$\vec{c}_{p,0} = t_{0,A}(\vec{c}_{p,A})$$

4.4 Range Query

In order to get a list of all objects located within a certain area, the user has to send one range query for unprotected objects, (i. e., using coordinates for k_0) and one range query for each coordinate system for which he knows the corresponding transformation function to the LS. The request must contain the CSID (e. g., B) and the coordinates of the concerning area a specified in coordinates for the corresponding coordinate system $\vec{c}_{a,B}$, i. e., he has to transform the k_0-coordinates $\vec{c}_{a,0}$ to k_B:

$$\vec{c}_{a,B} = t_{B,0}(\vec{c}_{a,0})$$

The LS selects all objects which are marked with B and which are located in the area specified by $\vec{c}_{a,B}$

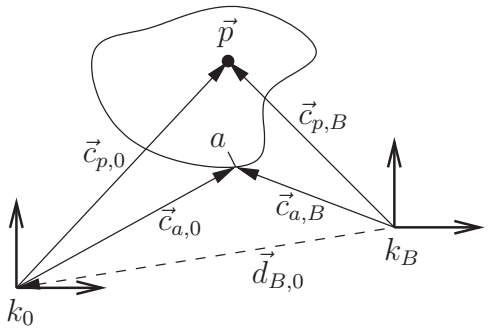

Figure 2. Range Query

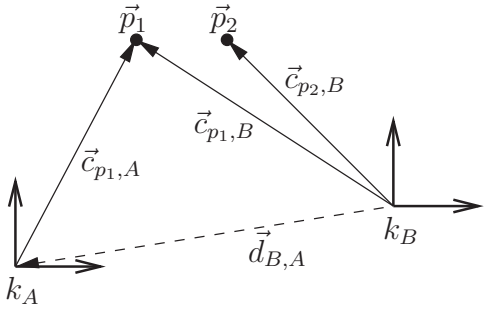

Figure 3. On-meeting event

(e. g., $\vec{p}$), and returns the list of these objects (as well as their k_B-coordinates) to the requester (see figure 2). The user can then transform the location of the returned objects back to k_0 as described in chapter 4.3.

4.5 On-enter-area Event

Similar to the range query case, the user has to register one on-enter-area event for unprotected objects and one on-enter-area event for each coordinate system, for which he was authorized, at the ES. For each event registration he must specify the CSID and the concerning area in coordinates for the corresponding coordinate system. Then, the ES starts observing whether on-enter-area events are triggered on the specified areas by objects with the corresponding CSID, and, if so, sends event notifications to the user. Again, the user can transform the location coordinates returned in the event notifications back to k_0 as described in chapter 4.3.

4.6 On-meeting Event

On-meeting events are more difficult to handle than the other request because the event request may involve two mobile objects using *different* transformations. Thus, we cannot apply the mechanism we used for range queries and on-enter-area events.

Instead, the user specifies the IDs of the two objects, the CSID of each object and the critical meeting distance in the event request. If the two objects use the *same* coordinate system, then the ES can monitor the distance between these objects (thanks to property 2) without requiring further information, and compare it to the distance specified in the event request.

However, if the objects in the on-meeting event request use *different* coordinate systems (e. g., $\vec{p}_1$ uses k_A and $\vec{p}_2$ k_B), then the ES will be unable to calculate the distance between them. Thus, the user has to provide another piece of information, but he should reveal neither $t_{0,A}(\vec{x})$ nor $t_{0,B}(\vec{x})$ in order not to reveal the k_0-coordinates of any object. The solution is to reveal only the delta transformation function $t_{B,A}(\vec{x})$ to the ES, but *not* the *absolute* transformation functions $t_{0,A}(\vec{x})$ and $t_{0,B}(\vec{x})$. The ES will thus be able to transform the coordinates of the object $\vec{p}_1$ using k_A to the coordinate system used by $\vec{p}_2$ (see figure 3):

$$\vec{c}_{p_1,B} = t_{B,A}(\vec{c}_{p_1,A})$$

The ES can then monitor the distance between $\vec{p}_1$ and $\vec{p}_2$ watching $\vec{c}_{p_1,B}$ and $\vec{c}_{p_2,B}$ and send event notifications including the current location of the two objects to the user if necessary. However, due to property 3, the ES will be unable to calculate the k_0-coordinates of the objects with the help of $t_{B,A}(\vec{x})$.

5 Known Security Problems and Proposed Solutions

Unfortunately, the presented approach does not provide a completely satisfying solution. Known problems and proposed solutions are discussed below.

5.1 Location Trace Patterns

An attacker (e. g., the LS provider) can analyze location traces and try to detect patterns (e. g., characteristic roads or buildings) he can match to known structures in k_0. If he succeeds, the transformation function can be calculated and subsequently validated by transforming the location traces to k_0 and verifying the plausibility of the traces. One approach is to change the used coordinate system frequently. The overhead for the distribution of the new transformation functions and the (necessary) re-registration of events can be kept small by using time-dependent transformation functions.

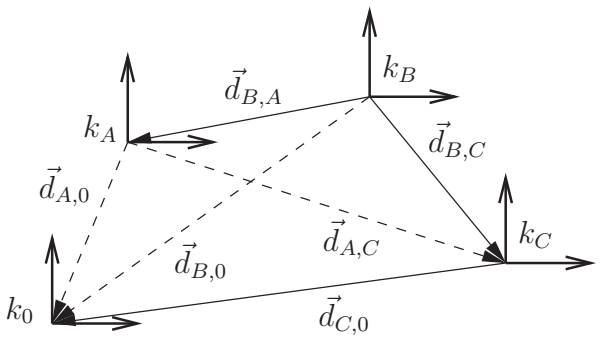

Figure 4. Delta Transformation Correlation

5.2 Request Patterns

Attackers (e. g., the LS or ES providers) may try to detect whether successive range queries or on-enter-area event registrations contain the same area represented in coordinates of different coordinate systems, e. g., by comparing the shape and extent of the area. In the case of success, attackers can calculate transformations between different coordinate systems (possibly including k_0).

5.3 Delta Transformation Correlation

An attacker with access to event registrations (e. g., the ES provider) can collect delta transformations from on-meeting-events of different users and try to systematically calculate more, previously unknown transformations. Knowing for example $t_{B,A}(\vec{x})$ and $t_{B,C}(\vec{x})$, an attacker can calculate also $t_{A,C}(\vec{x})$, which gives him the possibility to transform coordinates from k_A to k_C (see figure 4).

Even worse, if a user now registers an on-meeting-event between an object using k_C and an unprotected object, the attacker will learn $t_{C,0}(\vec{x})$ and successively also $t_{B,0}(\vec{x})$ and $t_{A,0}(\vec{x})$, which leads to a complete circumvention of the intended protection of the location information because the ES can now calculate the coordinates of all objects in k_0! The possibility to compose delta transformations causes severe problems, for which we try to sketch solutions approaches.

The chains of delta transformations can be broken by using different VIDs with different transformations for the same object in different event registrations. Unfortunately, this implies that mobile users have to use multiple VIDs. If the users of the LBS want to register n different events involving a certain mobile user, than he will have to register n different VIDs with different transformations, send n location updates to the LS every time he moves and distribute the n VIDs with corresponding transformation functions among the users according to the number of events they want to register.

Events involving one protected and one unprotected object should be avoided. One approach is to create different VIDs using non-k_0 coordinate systems (although they do not need to be protected themselves) and use only protected VIDs in event registrations.

Using different VIDs with different transformations for the same object involves a certain risk: It will probably be rather simple to determine whether or not two VIDs belong to the same object by comparing the number and timestamps of the location updates, the distance between successive location updates and the angle between two moves. If the ES succeeds in detecting which VIDs belong to the same object, the intended protection will be void. Although there are techniques to make it more difficult to find VIDs belonging together, e. g., by sending slightly different location update sequences for different VIDs, the protection level will remain low.

For users requiring a higher level of protection we propose the following, more secure, but unfortunately less efficient solution: When distributing the transformation function to authorized users, they should be notified that they are allowed to send any queries and events except for events requiring delta transformations, i. e., they may use on-meeting events on objects with the same, but not with different transformations.

If these users cannot do without these events, they can use a local event service (e. g., running on the mobile device) which can send all allowed queries and events to the LS and ES in order to collect information about the position of the involved objects and test locally whether a registered event has occurred.

6 Conclusions and Outlook

We have presented an approach demonstrating that it is basically possible to solve the major privacy problem of LBS and to protect the location information of mobile users even against malicious location and event service providers.

Our scheme provides a relatively 'weak' protection; however, we are not aware of a better solution. Note that some risks cannot be avoided without violating the requirements specified in chapter 2.3. Although we cannot offer a perfect solution, we believe that it will be sufficiently difficult, expensive and time-consuming for service providers to circumvent the protection mechanisms on a large scale in order to obtain reliable location information of all users.

It is obvious that the this scheme is not yet a completely satisfactory solution. Apart from adding complexity to the processing of location informations and generating overhead with respect to bandwidth, storage capacity and processing time, it is not free from from security risks endangering the dearly bought privacy.

Although solutions to avoid these risks have been pointed out, not all problems are solved yet. One of the most challenging remaining problem is to develop strategies for client applications to keep track of the different coordinate systems in order to avoid dangerous accumulations of delta transformations at the ES.

Acknowledgements

We would like to thank Silke Kurz for her help with the security evaluation. This work was funded by the German Research Foundation (DFG) through the Center of Excellence (SFB) 627.

References

[1] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu. Order preserving encryption for numeric data. In *Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data*, pages 563–574, 2004.

[2] M. Bauer. Event management for mobile users. Technical Report 2004/02, Universität Stuttgart, Institut für Parallele und Verteilte Systeme, 2004.

[3] A. R. Beresford and F. Stajano. Location privacy in pervasive computing. In *IEEE Pervasive Computing*, pages 46–55, 2003.

[4] J. Domingo-Ferrer. A provably secure additive and multiplicative privacy homomorphism. In *Proceedings of the 5th International Conference on Information Security*, volume 2433 of *Lecture Notes in Computer Science*, pages 471–483, 2002.

[5] M. Gruteser and D. Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *Proceedings of the The First International Conference on Mobile Systems, Applications, and Services*, pages 31–42, 2003.

[6] M. Langheinrich. Privacy by design – principles of privacy-aware ubiquitous systems. In *Ubicomp 2001 Proceedings*, volume 2201 of *Lecture Notes in Computer Science*, pages 273–291. Springer, 2001.

[7] A. Leonhardi and K. Rothermel. Architecture of a large-scale location service. Technical Report 2001/01, Universität Stuttgart, Institut für Parallele und Verteilte Systeme, 2001.

[8] D. Nicklas and B. Mitschang. On building location aware applications using an open platform based on the nexus augmented world model. In *Software and Systems Modeling*, 2004.

[9] D. Nicklas, C. Pfisterer, and B. Mitschang. Towards location-based games. In *Proceedings of the International Conference on Applications and Development of Computer Games in the 21st Century*, pages 61–67, 2001.

[10] G. Treu, A. Küpper, and P. Ruppel. Anonymization in proactive location based community services. In *Adjunct Proceedings of the Third International Conference on Pervasive Computing*, pages 49–53, May 2005.