Protecting Movement Trajectories through Fragmentation

Marius Wernke, Frank Dürr, and Kurt Rothermel

Institute of Parallel and Distributed Systems (IPVS), University of Stuttgart
Universitätsstraße 38, 70569 Stuttgart, Germany
marius.wernke,frank.duerr,kurt.rothermel@ipvs.uni-stuttgart.de

Abstract. Location-based applications (LBAs) like geo-social networks, points of interest finders, and real-time traffic monitoring applications have entered people's daily life. Advanced LBAs rely on location services (LSs) managing movement trajectories of multiple users in a scalable fashion. However, exposing trajectory information raises user privacy concerns, in particular if LSs are non-trusted. For instance, an attacker compromising an LS can use the retrieved user trajectory for stalking, mugging, or to trace user movement. To limit the misuse of trajectory data, we present a new approach for the secure management of trajectories on non-trusted servers. Instead of providing the complete trajectory of a user to a single LS, we split up the trajectory into a set of fragments and distribute the fragments among LSs of *different* providers. By distributing fragments, we avoid a single point of failure in case of compromised LSs, while different LBAs can still reconstruct the trajectory based on user-defined access rights.
In our evaluation, we show the effectiveness of our approach by using real world trajectories and realistic attackers using map knowledge and statistical information to predict and reconstruct the user's movement.

Key words: Location management, fragmentation, trajectories, privacy

1 Introduction

Nowadays, most people carry a mobile device with an integrated positioning system such as GPS. In combination with cheap and fast mobile communication technologies, location-based applications (LBAs) like geo-social networks and points of interest finders have entered people's daily life.

LBAs can be classified into two categories: LBAs using single user positions, and LBAs using movement trajectories. The first category considers individual positions, for instance, provided when a user is "checking-in" to a restaurant, to show friends his current position. The second category considers LBAs relying on movement traces of mobile objects. Examples are applications for sharing jogging paths, community-based mapping, and real-time traffic monitoring.

Advanced LBAs rely on location services (LSs), which manage a huge number of mobile object positions and allow for sharing positions between multiple

applications. LSs provide functionalities for spatial queries and relieve mobile objects from sending their position individually to multiple LBAs.

While the position of a mobile object is essential for many applications, sharing position information may raise serious privacy concerns. As shown in [13], 55% of the 1.500 survey participants using LBAs were worried about loss of privacy. For instance, an attacker can determine habits, interests, or other sensitive information by analyzing movement trajectories. Furthermore, an attacker can misuse position information for stalking, mugging, or to trace user movement.

Existing privacy approaches protecting trajectories usually rely on a trusted third party (TTP). For instance, k-anonymity approaches [8] protecting the user identity rely on a *location anonymizer* to collect positions of different users to generate areas fulfilling the user-defined k-anonymity level. Mix zones [2] use a trusted middleware to change user pseudonyms within a predefined spatial region. Advanced obfuscation approaches [3] protecting the user trajectory use a TTP to calculate cloaked spatial regions. However, as shown by many incidents [5], even service providers that were deemed to be trusted could be compromised. Therefore, no service provider can guarantee to perfectly protect its stored information. Consequently, we have to consider that LSs are non-trusted.

To protect movement trajectories of mobile users without using a trusted third party, we present our novel Trajectory Fragmentation Algorithm (*TFA*). The general idea of our approach is to split up a user's trajectory into a set of smaller fragments that are distributed to different LSs of different providers. Thus, each LS stores only a part of the trajectory and no LS knows the complete movement trace of the user. Therefore, an attacker compromising an LS has only limited knowledge about the user's movement. To prevent an attacker compromising multiple LSs from correlating the retrieved fragments, *TFA* uses different pseudonyms for the fragments. Different clients of the LSs, for instance, different LBAs can access the trajectory of the user by querying fragments from multiple LSs based on the known pseudonym used for the corresponding LS.

In this paper, we make the following contributions: (1) A concept for distributing trajectory fragments among a set of LSs to avoid that a single LS can reveal the complete user trajectory. (2) An optimized algorithm that calculates a minimum number of fragments under the constraint of fulfilling a given level of privacy. (3) A privacy evaluation with attackers of different strength showing the effectiveness of our approach.

The rest of the paper is structured as follows: In Sec. 2, we present related work. In Sec. 3, we introduce our system model. Then, we formalize our problem statement and present *TFA* in Sec. 4 and Sec. 5. In Sec. 6, we present our privacy and performance evaluation. Finally, we conclude the paper in Sec. 7.

2 Related Work

Existing location privacy approaches can be classified based on their protection goal and whether they focus on single positions or movement trajectories. For an overview of existing location privacy attacks and approaches we refer to [15]. For

single positions, k-anonymity [6] can be used to protect the identity of the user by making the user indistinguishable from $k-1$ other users. Spatial obfuscation [4] protects the position of the user by decreasing the precision of the position provided to an LS. Position sharing [14] allows for providing different precision levels to different LBAs while LSs only manage positions of limited precision. However, all these approaches focus on singular positions instead of trajectories.

Trajectory k-anonymity [8] protects the identity of a user by making the user indistinguishable from $k-1$ other users for the complete trajectory. Mix zones [2] protect the identity of a user by changing pseudonyms within an area where no user provides position information to an LBA. Trajectory obfuscation [3] uses a group-based approach to collect positions of multiple users to calculate obfuscated areas for the complete trajectory. However, all of these approaches require a TTP, while we do not rely on a TTP.

Dummy approaches [12] provide false trajectories to the LS in addition to the real user trajectory such that the LS cannot distinguish the received trajectories. However, since identifying dummy trajectories is possible [10], user privacy can be decreased. The approach presented in [1] protects the start and the destination of a trajectory without assuming a TTP. With our approach, we focus on protecting the complete trajectory rather than only its start or destination.

3 System Model

The components of our system are depicted in Fig. 1. The **mobile object** (MO) has an integrated positioning system such as GPS that is used to determine the MO's current position π. The MO executes a local software component that provides π to a set of different **location servers** (LSs), which store and manage trajectories of several MOs. LSs provide an access control mechanism to manage access to the stored positions. Different clients, for instance, different LBAs get access to the positions on different LSs based on their access rights.

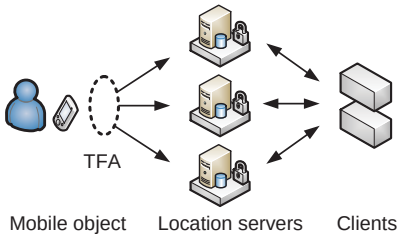

Fig. 1. System components

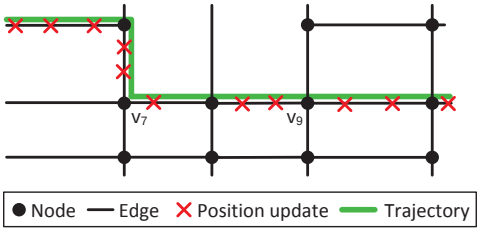

Fig. 2. Graph and trajectory example

The MO's position π is defined by its longitude and latitude values. Since MOs usually travel on streets, we map π to a graph representing the road network by using existing map-matching approaches. As shown in Fig. 2, the road network can be modeled as graph $G = (V, E)$ consisting of a set of nodes

V and a set of edges E. Each node $v_i \in V$ represents a junction or an intermediate node that models the shape of the road. Each edge $e_j \in E$ represents a road segment between two nodes. The MO's movement trajectory $T = \{(\pi_{start}, t_{start}), \ldots, (\pi_{end}, t_{end})\}$ represents a set of consecutive position fixes π_i where the MO is located at time t_i. An example for T is shown in Fig. 2. We assume that the destination π_{end} of T is already known at time t_{start} because the MO typically knows its movement destination.

4 Problem Statement

Since we have to consider that LSs are non-trusted, an attacker can compromise an LS with a certain probability greater than zero. In this case, the part of the trajectory provided to the LS is revealed. Thus, our goal is to protect the MO's trajectory by minimizing the revealed information of an attack on an LS.

We use a distributed approach to store and manage the MO's trajectory T. Our trajectory fragmentation algorithm splits up T into a set $F = \{f_1, \ldots, f_n\}$ of n trajectory fragments (or *fragments* for short). The number n of generated fragments corresponds to the number of used LSs to store the fragments and is either predefined by the MO or calculated by the fragmentation algorithm. While traveling, the MO uses the fragmentation algorithm to update its position over time on different LSs. Each position is provided to exactly one LS and each LS receives only the positions of a single fragment.

To measure the information exposed to an LS, we use weight function $\Phi(f_j)$ assigning each fragment f_j the value of its exposed information. More precisely, we measure the length of the trajectory that is revealed to an attacker, called the *revealed trace length* $\Phi(f_j)$.

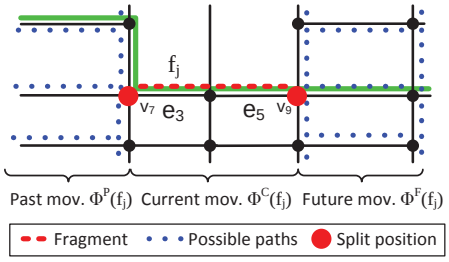

Fig. 3. Different information parts

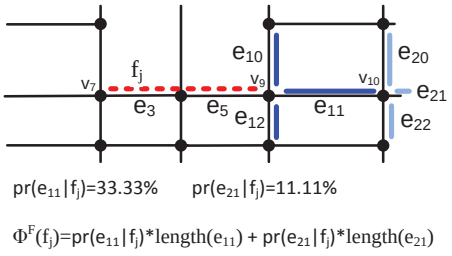

Fig. 4. Calculation of $\Phi^F(f_j)$

The revealed trace length $\Phi(f_j)$ is defined as the distance an attacker compromising fragment f_j can trace the MO. The value of $\Phi(f_j) = \Phi^C(f_j) + \Phi^F(f_j) + \Phi^P(f_j)$ is calculated as the sum of the following three parts (cf. Fig. 3): The first part of $\Phi(f_j)$ is the length of the current fragment $\Phi^C(f_j)$, i.e., the length of all edges $e_i \in f_j$. The second part of $\Phi(f_j)$ is the length of the predicted future movement $\Phi^F(f_j)$. This length is a probability measure, since multiple possible future paths exists. The probability that the MO travels along edge e_i after

traveling on f_j is $pr(e_i|f_j)$. Then, $\Phi^F(f_j)$ is calculated as the length of all edges e_i of the predicted future trajectory T weighted by the probability $pr(e_i|f_j)$. We assume that the probability $pr(e_i|f_j)$ is given for each edge e_i, for instance, by a statistical movement analysis, or by using map knowledge assuming a uniform distribution for all edges. An example to calculate $\Phi^F(f_j)$ based on map knowledge is shown in Fig. 4. The probability $pr(e_{11}|f_j)$ is 33.33%, since three alternatives exists to continue traveling after f_j. The probability $pr(e_{21}|f_j)$ is 11.11% taking also $pr(e_{11}|f_j)$ into account. The third part of $\Phi(f_j)$ is the length of the reconstructed past movement $\Phi^P(f_j)$. Here, $\Phi^P(f_j)$ is calculated as presented for $\Phi^F(f_j)$ by considering the past instead of the future movement.

In order to find a fragmentation of trajectory T such that each fragment has a minimum revealed trace length $\Phi(f_j)$, we have to minimize the maximum value of $\Phi(f_j)$. This means, no matter which LS is compromised, only the maximum trace length $\Phi(f_j)$ can be revealed. Formally, we have to solve the following optimization problem:

$$\begin{aligned} &\text{minimize} && \max_{j\in[1;n]}(\Phi(f_j)) \\ &\text{subject to} && \bigcup_{j\in[1;n]} f_j = \mathrm{T} \end{aligned}$$

Additionally, we want to answer the question of how many LSs are required to protect trajectory T from revealing a larger trace length than specified by a MO-defined maximum trace length Φ^{MO}_{max}. This means, we consider the problem of minimizing the number n of required LSs for Φ^{MO}_{max}. Formally, this optimization problem is defined as

$$\begin{aligned} &\text{minimize} && n \\ &\text{subject to} && \max_{j\in[1;n]}(\Phi(f_j)) \leq \Phi^{MO}_{max} \\ & && \bigcup_{j\in[1;n]} f_j = \mathrm{T} \end{aligned}$$

5 Trajectory Fragmentation Algorithm

In this section, we present our trajectory fragmentation algorithm *TFA*. We start with an overview of *TFA* and present the individual steps of *TFA* afterwards.

5.1 Process Overview

The concept of *TFA* consists of two parts: The *trajectory fragmentation* performed by the MO and the *trajectory reconstruction* performed by clients.

The trajectory fragmentation first calculates the MO's predicted trajectory T based on the given destination. Secondly, it splits up T into a set of fragments. Thirdly, each fragment is assigned to an LS. While traveling, the MO updates

Algorithm 1 TFA: Process overview

Function: $TFA(n, \pi_{end})$
1: $\pi_{start} \leftarrow getPosition()$
2: $T \leftarrow FP(\pi_{start}, \pi_{end})$
3: $F[1, \ldots, n] \leftarrow fragment(T, n)$
4: $ID[1, \ldots, n] \leftarrow getIDs(F)$
5: $\pi_i \leftarrow \pi_{start}$
6: **while** $\pi_i \in T$ **do**
7: $f_j \leftarrow getFragment(\pi_i)$
8: $LS_j \leftarrow getLS(f_j)$
9: $id_j \leftarrow getID(f_j)$
10: $update(\pi_i, LS_j, id_j)$
11: $\pi_i \leftarrow getPosition()$
12: **end while**

Algorithm 2 TFA: fragmentation

Function: $fragment(T, n)$
1: $G_F \leftarrow getGraph(T)$
2: $M \leftarrow getAdjacencyMatrix(G_F)$
3: $L \leftarrow n$
4: $M^L \leftarrow maxMatrixMult(M, L)$
5: $\Phi_{max} \leftarrow M^L[0, m]$
6: $M_{max} \leftarrow trimEdges(M, \Phi_{max})$
7: $M^L_{max} \leftarrow maxMatrixMult(M_{max}, L)$
8: $P \leftarrow calculateAllPaths(M^L_{max}, L)$
9: $S \leftarrow getRandomPath(P)$
10: $F[1, \ldots, n] \leftarrow getFragments(S)$
11: **return** $F[1, \ldots, n]$

its position on the LSs based on the calculated fragmentation. If the predicted trajectory deviates from the real trajectory, a new fragmentation is initiated using a new set of LSs.

The process of *TFA* is shown in Alg. 1. Since MOs normally travel on fastest paths to reach their destination as fast as possible, we predict the MO's trajectory T at time t_{start} as the fastest path from the current position π_{start} to the known destination π_{end}. Then, we split up T into set $F = \{f_1, \ldots, f_n\}$ of n fragments, one for each LS, by using function $fragment(T, n) = \{f_1, \ldots, f_n\}$, introduced below. For each fragment $f_j \in F$, we calculate a new pseudonym of the MO and select an LS to store f_j. The LS storing the positions of fragment f_j is denoted as LS_j. We use the notation $\pi_i \in f_j$ to denote that position π_i is part of f_j. As formalized in Sec. 4, the goal of function $fragment(T, n)$ is to minimize the maximum revealed trace length $\Phi(f_j)$. To solve the presented optimization problem, we use a dynamic programming approach presented below.

After calculating fragmentation F, the MO updates its position $\pi_i \in f_j$ to LS_j while traveling on f_j. As soon as a new position π_i is part of f_{j+1}, the MO changes the LS storing π_i from LS_j to LS_{j+1}. Furthermore, the used pseudonym is changed from id_j to id_{j+1}. In case the MO leaves the predicted trajectory T at position $\pi_k \notin T$, a new calculation of *TFA* is initiated for the new initial position $\pi_{start} = \pi_k$ and the destination π_{end} using a new set of LSs.

The trajectory reconstruction allows different clients to access the MO's real trajectory T by querying the LSs using the provided pseudonyms of the MO.

5.2 Trajectory Fragmentation

Next, we present function $fragment(T, n)$ calculating set $F = \{f_1, \ldots, f_n\}$ of fragments in Alg. 2. As introduced, each fragment $f_j \in F$ defines to which LS_j position $\pi_i \in f_j$ should be sent. The part of T belonging to fragment f_j is the part of T between two *split positions*. For example, fragment f_j in Fig. 3 is defined by the split positions of junction v_7 and v_9. We split up the MO's predicted trajectory T at nodes representing junctions instead of splitting up T

within an edge or at an intermediate node. This approach has the advantage that all positions belonging to the same edge e_i are assigned to the same fragment which is stored by only one LS.

The problem is now how to find set $S = \{s_0, \ldots, s_n\}$ of split positions for T such that the corresponding set of fragments F is optimal considering the maximum revealed trace length $\Phi(f_j)$. To solve the proposed optimization problem, we first calculate the *fragmentation graph* $G_F = (V_F, E_F)$ as defined next: The set of nodes $V_F = \{v_i \in T\}$ consists of all possible split positions of T, i.e., the set of nodes representing a junction on T. The set of edges E_F is generated by calculating for each node $v_i \in V_F$ an edge to each node $v_j \in V_F$ if the MO will visit the junction of v_i before visiting the junction of v_j. The generated edge from v_i to v_j is denoted as $e_{ij} \in E_F$. The weight of e_{ij} is $\Phi(f(v_i, v_j))$ representing the revealed trace length of the fragment that is defined by the split position v_i and v_j. Then, we calculate the adjacency matrix M of G_F, which is of size $m \times m$ with $m = |V_F|$. Each value $M[i;j] = \Phi(f(v_i, v_j))$ defines the weight of edge e_{ij}. Since we aim for an optimal fragmentation using n LSs, we have to find a path in G_F consisting of n edges from the first split position s_0 to the last split position s_n minimizing the maximum edge weight. The L-th power of the adjacency matrix M is denoted as M^L and calculated using matrix multiplication. For each possible node v_k, we calculate the maximum value of $M^{L-1}[i;k]$ and $M^1[k;j]$. Then, we select the minimum value from all possible nodes v_k and store the determined maximum value in $M^L[i;j]$. Thus, $M^L[i;j]$ is the minimized maximum revealed trace length of a single fragment on the path of length L that leads from v_i to v_j. The maximum value for a path of length n from split position s_0 to s_n is the value of $M^n[0,m]$. This value is then stored in Φ_{max} and used to remove all edges in M with a higher value than Φ_{max}. The resulting adjacency matrix is M_{max}. To calculate all possible paths of length $L = n$ with a maximum single edge value below Φ_{max}, we incrementally calculate M^L_{max} using the introduced matrix multiplication. The nodes of the calculated paths represent the possible split positions for an optimal fragmentation. We randomly select one of all possible paths that were calculated and store the corresponding split positions in set S. Finally, we determine set F of fragments using the calculated split positions.

5.3 Minimizing the Number of Required LSs

After solving the problem of how to find an optimal fragmentation for trajectory T, we consider now the problem of minimizing the number n of required LSs to achieve a MO-defined maximum revealed trace length Φ^{MO}_{max}. To solve this problem, we adapt Alg. 2 as follows. Instead of using a fixed value of $L = n$ to calculate Φ_{max}, we stepwise increment L. As soon as $M^L[0,m] \leq \Phi^{MO}_{max}$, the minimum path length L and thus the minimum number of required LSs is found that can provide a maximum value of Φ^{MO}_{max}. After setting Φ_{max} to Φ^{MO}_{max} we can further use Alg. 2 without modification. If L reaches a value above L_{max}, which represents the maximum number of available LSs, no solution could be found for T and Φ^{MO}_{max}. Then, the MO has either to adjust Φ^{MO}_{max} or to use Alg. 2 to find an optimal fragmentation using $n = L_{max}$ LSs.

6 Privacy and Performance Evaluation

In this section, we evaluate the provided privacy of *TFA* and present our performance evaluation. Next, we introduce our attacker model and privacy metric.

6.1 Attacker Model

Nowadays, map knowledge is widely available, for instance, provided by the OpenStreetMap project [9]. Therefore, we assume that an attacker has map knowledge and knows the used fragmentation algorithm. In case an attacker compromises a client, all positions provided to the client are revealed. To limit the revealed information of a client, the MO can individually specify which part of the trajectory should be accessible for each client by defining access rights on the LSs. Because the access control mechanisms do not prevent that the stored information of an LS is revealed to an attacker compromising the LS, we further consider that an attacker compromises a single or multiple LSs. If an attacker compromises an LS, all positions assigned to the stored fragment are revealed.

The goal of an attacker is to derive as much information as possible from its known positions. Therefore, we consider attackers using state of the art movement prediction methods to predict and reconstruct the MO's trajectory. More precisely, we use the first order Markov model presented in [7] to simulate attackers using different turn probability estimations. First, we consider attacker A^{NC} that cannot correlate fragments that were provided to different LSs using different pseudonyms. Secondly, we consider attacker A^{AC} that can correlate adjacent fragments based on their spatiotemporal properties. That is, A^{AC} analyses the positions of two fragments and merges both fragments if the positions belong to two adjacent edges.

In addition to the ability of correlating fragments, we distinguish two attackers which are of different strength based on their known information. The first attacker A_{MAP} uses map knowledge to determine the MO's trajectory from his known positions. To this end, A_{MAP} estimates at each junction a uniform probability distribution where the MO probably came from or where the MO is probably going to. For instance, if three alternatives exist at a junction where the MO can continue traveling, A_{MAP} assigns each edge the probability of 33.33%. Then, A_{MAP} predicts and reconstructs the trajectory by selecting step by step adjoining edges to his known fragments based on the calculated probability distribution. The second attacker A_{SMI} uses statistical movement information gained from a road network traffic analysis from trajectories of other MOs. The goal of A_{SMI} is to improve its prediction by using more accurate turn probabilities.

6.2 Privacy Metric

To measure the provided privacy of fragmentation F, we analyze the maximum revealed trace length Φ^{A}_{max} an attacker can derive from its compromised positions. For an attacker who is able to correlate adjacent fragments even if

different pseudonyms are used, the maximum revealed trace length Φ^A_{max} is in the worst case equal to the length of the complete trajectory. Therefore, we analyze also the probability $Pr(\Phi^A_{max} \leq \tau)$ that the maximum revealed trace length Φ^A_{max} is below or equal to a threshold value τ. The probability that attacker A successfully compromises the k LSs storing the k fragments of set $F_C \in F$ is $\alpha(F_C) = p^k * (1-p)^{n-k}$, where $p \in (0,1]$ is the probability that A can compromise a single LS and F_C is the set of compromised fragments. Then, we calculate for a given value τ the cumulated probability $Pr(\Phi^A_{max} \leq \tau)$ as the sum of the probability values $\alpha(F_C)$ for all sets $F_C \in F$ fulfilling $\Phi^A_{max} \leq \tau$.

6.3 Privacy Evaluation

We analyze the success of different attackers in predicting and reconstructing the MO's movement trajectory based on the compromised fragments by using current state of the art movement prediction methods. In our evaluation, we use the real-world dataset provided by [11], which consists of the traces of about 500 taxis collected for 30 days in the San Francisco Bay Area. The used map information is derived from the OpenStreetMap project [9]. The turn probabilities of attacker A_{MAP} consider a uniform distribution for all possible paths at each junction. For attacker A_{SMI}, we performed a road network traffic analysis where we analyzed the movement behavior of all taxis for a complete day (2008/06/01) and derived for each junction the corresponding turn probability distribution. For our evaluation, we selected from the next day (2008/06/02) a short trajectory of 4.49 km length within the city (denoted as *city*) and a long trajectory of 17.04 km length mainly using highways (denoted as *highway*). We assume a probability of $p = 10\%$ that a single LS is compromised and calculate the probability that multiple LSs are compromised as presented above. Next, we evaluate attacker A^{NC} and continue afterwards with attacker A^{AC}.

Attacker $\mathbf{A^{NC}}$: Since attacker A^{NC} does not use fragment correlation, the cases that A^{NC} compromises multiple LSs is identical to the case that A^{NC} compromises a single LS. To show the success of protecting trajectories against A^{NC} when using fragmentation, we measure the maximum revealed trace length Φ^A_{max} that is revealed to attacker A^{NC}_{MAP} and A^{NC}_{SMI} for the introduced trajectories. The results are shown in Fig. 5 and Fig. 6. For the *city* trajectory, Φ^A_{max} of A^{NC}_{MAP} decreases to $498m$ (11.07%) when using 15 LSs. By considering statistical movement information in addition to the map knowledge, Φ^A_{max} of A^{NC}_{SMI} decreases to 1148 m (25.55%). For the *highway* trajectory, Φ^A_{max} decreases to 5601 m (32.86%) respectively 9283 m (54.64%). As we can see, each trajectory has a limiting value of Φ^A_{max} such that even when increasing n, the maximum revealed trace length known to the attacker does not decrease any more. This is based on the fact that Φ^A_{max} cannot decrease below the maximum revealed trace length of a single road segment on the considered trajectory.

By comparing the relative values of Φ^A_{max} from the *highway* trajectory with the *city* trajectory, A^{NC}_{SMI} receives a higher value of Φ^A_{max} for the *highway* trajectory. This is based on the fact that the prediction of A^{NC}_{SMI} works well on the highway due to a high statistical probability that the MO stays on the

highway for longer times. For the *city* trajectory, many junctions exists with approximately equal turn probabilities for alternative roads such that the attacker cannot precisely predict the MO's movement.

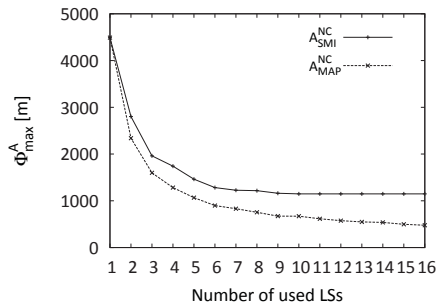

Fig. 5. City trajectory evaluation

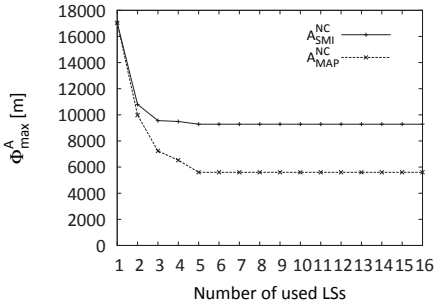

Fig. 6. Highway trajectory evaluation

Next, we analyze the minimum number of LSs that is required to reveal at most a trace length of Φ^{MO}_{max} to a single LS. Figure 7 shows which values of Φ^{MO}_{max} can be provided using the map based and the statistical movement based fragmentation. By decreasing Φ^{MO}_{max} from the maximum length of the considered trajectory, the minimum number n of required LSs to provide Φ^{MO}_{max} increases. Again, each trajectory has a limiting minimum value of Φ^{MO}_{max} such that smaller values of Φ^{MO}_{max} cannot be provided even when increasing n as presented before.

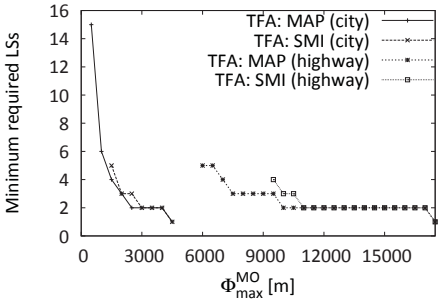

Fig. 7. Minimum n for trajectories

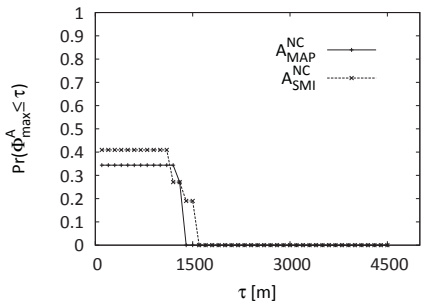

Fig. 8. Cumulated probability distr.

To show that attacker A^{NC} cannot derive a higher revealed trace length than specified by the MO in Φ^{MO}_{max}, we analyze for the *city* trajectory the cumulated probability distribution $Pr(\Phi^A_{max}(F_C) \leq \tau)$ using the fragmentation generated by optimizing n. We select a value of $\Phi^{MO}_{max} = 1.5\,\text{km}$, which leads to $n = 4$ LSs for the map based fragmentation and $n = 5$ for the statistical movement based fragmentation. As shown in Fig. 8, the probability that A^{NC}_{MAP} and A^{NC}_{SMI} can derive a maximum revealed trace length above Φ^{MO}_{max} is zero such that the defined privacy requirement is fulfilled and *TFA* prevents that an attacker can derive a higher revealed trace length than Φ^{MO}_{max}. Altogether, we can state that *TFA* effectively prevents that attacker A^{NC} can trace the MO over longer distances.

Attacker $\mathbf{A^{AC}}$: The powerful attacker A^{AC} can correlate adjacent fragments based on the spatio-temporal properties of the compromised positions as presented in Sec. 6.1. If A^{AC} can compromise all used LSs, the complete trajectory of the MO is revealed. This results in a maximum revealed trace length of Φ^A_{max} equal to the length of the trajectory. To better understand this kind of strong attacker, we show the success of A^{AC} to trace the MO over a certain distance τ by analyzing the cumulated probability that A^{AC} receives a maximum revealed trace length of τ by compromising a certain set of LSs. Figure 9 shows for the city trajectory the cumulated probability that attacker A^{AC}_{MAP} can derive a maximum value Φ^A_{max} of τ for different values of n. As we can see, increasing the number of generated fragments increases the probability that a small part of the trajectory is revealed, whereas the probability that longer parts are revealed decreases. For instance, A^{AC}_{MAP} can trace the MO for a value of $\tau = 1\,\text{km}$, which represents 22.27% of the trajectory, only with a probability of 1.08% when using 15 LS. Therefore, we can state that *TFA* can be used to prevent attacker A^{AC} from tracing the MO over longer distances with a high probability.

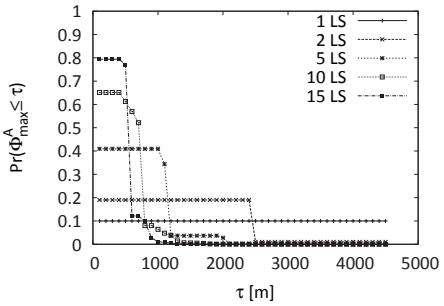

Fig. 9. Cumulated probability distr.

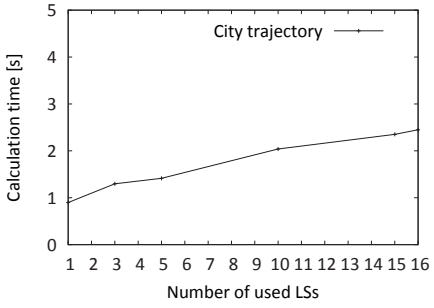

Fig. 10. Runtime performance

6.4 Performance Evaluation

Next, we evaluate the performance of *TFA* by measuring its runtime on a state of the art mobile device (HTC Desire HD). We measure the required time to calculate for the *city* trajectory the corresponding fragmentation using n LSs. As shown in Fig. 10, the calculation time stays below 3 s even for a larger number of fragments. The runtime of *TFA* for optimizing the number of required LSs for the introduced value of $\Phi^{MO}_{max} = 1.5\,\text{km}$ also stays below 1.4 s. Recall that the fragmentation is only calculated initially at the start of the MO's movement or if the MO leaves the predicted trajectory. While traveling, *TFA* only performs a simple lookup of the current position against the corresponding fragment before sending the position to the LS. Therefore, we can state that *TFA* supports real-time position updates and that the fragmentation time is reasonable.

7 Conclusion and Future Work

In this paper, we presented a novel approach to protect the user's movement trajectory in a non-trusted system environment. The basic idea of our approach is to split up the user's trajectory into a set of trajectory fragments that are distributed among LSs of different providers. In case an LS gets compromised, only the positions of the stored fragment are revealed instead of the complete trajectory. In our evaluation, we used real world trajectories to show the effectiveness of our approach to protect trajectories against attackers using map knowledge and statistical movement information. In future work, we will analyze how we can improve our approach by considering LSs of different trust levels.

References

1. C. Ardagna, G. Livraga, and P. Samarati. Protecting privacy of user information in continuous location-based services. In *IEEE 15th International Conference on Computational Science and Engineering*, pages 162–169, 2012.
2. A. R. Beresford and F. Stajano. Location privacy in pervasive computing. *IEEE Pervasive Computing*, 2(1):46–55, 2003.
3. C.-Y. Chow and M. F. Mokbel. Enabling private continuous queries for revealed user locations. In *Proceedings of the 10th International Conference on Advances in spatial and temporal databases*, 2007.
4. M. Damiani, C. Silvestri, and E. Bertino. Fine-grained cloaking of sensitive positions in location-sharing applications. *Pervasive Computing*, 10(4):64 –72, 2011.
5. DATALOSSDB. www.datalossdb.org, June 2013.
6. P. Kalnis, G. Ghinita, K. Mouratidis, and D. Papadias. Preventing location-based identity inference in anonymous spatial queries. *IEEE Transactions on Knowledge and Data Engineering*, 19(12):1719 –1733, 2007.
7. J. Krumm. A markov model for driver turn prediction. In *Society of Automotive Engineers (SAE) World Congress*, 2008.
8. M. E. Nergiz, M. Atzori, Y. Saygin, and B. Güç. Towards trajectory anonymization: a generalization-based approach. *Transactions on Data Privacy*, 2(1):47–75, 2009.
9. OpenStreetMap. www.openstreetmap.org, June 2013.
10. S. T. Peddinti and N. Saxena. On the limitations of query obfuscation techniques for location privacy. In *Proceedings of the 13th International Conference on Ubiquitous Computing*, 2011.
11. M. Piorkowski, N. Sarafijanovoc-Djukic, and M. Grossglauser. A parsimonious model of mobile partitioned networks with clustering. In *The First International Conference on COMmunication Systems and NETworkS*, pages 1–10, 2009.
12. P. Shankar, V. Ganapathy, and L. Iftode. Privately querying location-based services with sybilquery. In *Proc. of the 11th Int. Conf. on Ubiquitous Computing*, 2009.
13. Webroot. http://www.webroot.com/us/en/company/press-room/releases/social-networks-mobile-security, June 2013.
14. M. Wernke, F. Dürr, and K. Rothermel. PShare: ensuring location privacy in non-trusted systems through multi-secret sharing. *Pervasive and Mobile Computing*, 9:339 – 352, 2013.
15. M. Wernke, P. Skvortsov, F. Dürr, and K. Rothermel. A classification of location privacy attacks and approaches. *Personal and Ubiquitous Computing*, 16:1–13, 2012.