Speed Protection Algorithms for Privacy-aware Location Management

Marius Wernke, Frank Dürr, Kurt Rothermel
Institute of Parallel and Distributed Systems, University of Stuttgart
Universitätsstraße 38, 70569 Stuttgart, Germany
<marius.wernke|frank.duerr|kurt.rothermel>@ipvs.uni-stuttgart.de

***Abstract*—Nowadays, millions of users share their complete movement trajectory online when using real-time traffic monitoring applications, pay-as-you-drive insurances, or when sharing their last road trip with friends. However, many users still hesitate to use location-based applications as they are not willing to reveal, for instance, their driving behavior or the occurrence of a speeding violation.**

Therefore, we present novel speed protection algorithms protecting users from revealing a violation of given speed limits when using location-based applications. Our algorithms support time-based and distance-based position updates. To protect positions indicating a speeding violation, we either adjust temporal information by delaying position updates or adjust their spatial information. We evaluate our algorithms by using real world traces and show that the protected movement trajectory of the user is of high quality even after removing speeding violations.

***Index Terms*—Location-based applications, speed protection, movement trajectory, location management, location privacy.**

I. Introduction

Driven by the availability of accurate positioning systems such as GPS and powerful mobile communication technologies like UMTS or LTE, location-based services (LBS) like friend finders or geo-social networks attract millions of users today. Such LBSs can be classified into two basic categories: LBSs using singular positions and LBSs based on movement trajectories. An example of the first class are services based on sporadic location "check-ins" like Foursquare [6], where the user can document his presence at certain points of interest. Another example are friend finder applications, which notify users about geographically close friends based on knowledge about the current user position. In contrast to these LBSs based on single positions, the second class of LBSs requires knowledge about the complete movement trajectory of a user typically acquired as GPS traces recording the sensed positions plus the timestamps of the positions. For instance, several LBSs for sharing hiking trails, jogging paths, etc. exist [3]. Other examples include pay-as-you-drive insurances, community-based mapping based on collected GPS traces (as used by the OpenStreetMap project), or real-time traffic monitoring.

In this paper, we consider the second class of LBSs based on movement trajectories. Our focus is on the protection of private information that can be derived from such trajectories, in more detail, the protection of *speed information*. Although the user is typically aware of the fact that LBSs as the ones mentioned above collect movement trajectories, he is seldomly aware that this information can be used to derive further information beyond geographical positions. In particular, movement trajectories consisting of positions and timestamps can be used to calculate the speed of the user. Although this speed information is mandatory for many applications like real-time traffic monitoring—e.g., to detect traffic jams—, the user might involuntarily reveal information about his behaviour that he is not willing to share, in particular, information about when and where he might drove too fast.

At this point, we have to make clear that in the case of violations of speed limits, the protection of information is ambivalent. On the one hand, the protection of private information—in particular, location information—is a commonly accepted goal. On the other hand, it should be clear to everybody that speed limits are there to protect people, and therefore, monitoring speed information is an important measure for law enforcement. Our decision to design mechanisms to protect speed information are based on the commonly accepted principle that everybody should be able to control, which information about him he reveals to someone else. In other words: If the user is aware that the collected information might be used for detecting speeding violations—for instance, as part of a pay-as-you-drive insurance with special rates for safety-conscious drivers, or using a tachograph for trucks—, no protection mechanisms are necessary. On the contrary, the insurance or police might want to ensure that the driver does not manipulate the speed information using tamper-proof devices (which is a different research topic on its own right). However, if the driver does not explicitly agree on accurately monitoring his speed, our mechanisms will make sure that no information can be recorded that might later be used against him. This is a very important prerequisite for ensuring the acceptance of location-based services based on trajectories and speed information. Clearly, although everyone would assume that he obeys speed limits in general, the possibility to detect violations will deter users from participating in such services like automatic traffic jam detection. Even if the recorded information cannot be used by the police due to legal restrictions, private companies like car insurances might use information found on the Internet (e.g., in OpenStreetMap GPS traces) to screen their customers and adjust rates. This poses a serious psychological barrier in providing unfiltered trajectory information to LBSs.

Various cases from the past have shown that speed in-

Published in 2013 IEEE 9th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob). Lyon, France, 7-9 October 2013.

formation is indeed used without knowledge of the users. For instance, in 2001, a car rental company in the US fined customers for speeding violations using GPS-equipped cars [19]. One customer was billed $150 for each of his alleged speeding violations where the trace showed a speed faster than 79 mph. For tracking, the company installed GPS-devices in their cars. Nowadays, sensing and tracking technology of mobile devices and car navigation systems can be used to track users. For instance, new navigation systems provide real-time speed and location data to servers calculating real-time traffic conditions, which is clearly a service that many drivers find useful and would actively support. However, in 2010, a company providing such services sold their collected GPS records to the Dutch police, which used the data to target speed traps where they could catch most drivers [18]. Immediately, the company stressed that they only stored anonymous data such that individual speeders could not be identified by the police. However, as shown in [8], user identification from anonymized trajectories is possible if, for instance, an anonymized trajectory starts in front of an individual home. Therefore, identifying individual speeders would be possible.

As we can see from these examples, publishing user trajectories without protecting the speed information can have severe monetary and legal effects on the user if speeding violations can be revealed, as well as for the acceptance of LBSs based on trajectory information. To remove these concerns and, therefore, to support the acceptance of LBSs, we present our speed protection algorithms protecting the speed information of a user trajectory in real-time by adjusting the reported trajectory to the allowed speed limit such that users do not have to fear any negative impact due to speeding violations. To protect trajectories from indicating speedings, we either adjust temporal information by delaying position updates or adjust spatial information of positions. In our evaluation, we analyze real world traces and show that protecting speed information of movement trajectories is necessary. Furthermore, we show that the accuracy decrease introduced by our speed protection algorithms only affects few position updates such that the speed protected trajectory is of high quality.

The rest of the paper is structured as follows: First, we present our system model and location model in Sec. II. Then, we introduce our problem statement in Sec. III. In Sec. IV, we introduce our speed protection algorithms. Our proof of correctness and our evaluation are presented in Sec. V and Sec. VI. In Sec. VII, we present related work. Finally, we conclude the paper in Sec. VIII.

II. System Model and Location Model

Our system model is depicted in Fig. 1 and consists of mobile objects, location servers, and clients.

The **mobile object** (MO) has an integrated positioning system, such as GPS, to determine the current user position π. The MO provides π to a **location server** (LS) storing and managing position information of several MOs. To this end, the MO sends a position update to the LS by using function $update(\pi, t)$ consisting of its position π and the

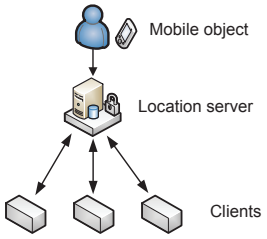

Fig. 1. System model

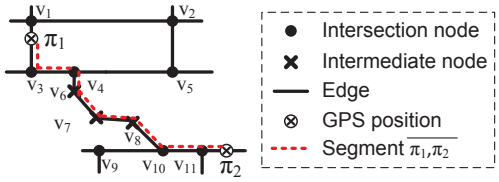

Fig. 2. Modeled road network

temporal information defining the corresponding time t. The LS provides different **clients**, for instance, different location-based applications, with stored positions. Using an LS relieves the MO from sending π individually to different clients. Position π has to be sent only once to the LS, while several clients can access the stored information on the LS. The LS controls access to the stored positions by using an access control mechanism such as [2]. Thus, only clients with the corresponding access rights can access the stored information.

To guarantee that no other component than our speed protection component can directly access the MO's position π locally on the mobile device, existing trusted mobile computing approaches such as [7] using, for instance, trusted hardware components can be used. Otherwise, a local client that could access π by directly querying the positioning system could maliciously reveal a speeding violation of the MO. For a detailed description on trusted computing approaches we refer to [7]. In the following, we focus our description on remote clients querying the remote LS and mention local clients only if they behave differently.

The MO's position π is defined by its longitude and latitude value. Since vehicles typically move on streets, we map positions to a graph representing the road network. As shown in Fig. 2, a road network can be modeled by a weighted graph $G = (V, E)$. Each node $v \in V$ represents either a junction or defines an intermediate node that is used to model the shape of a road. Each edge $e \in E$ represents a road segment between two nodes and has an assigned maximum allowed speed limit.

To map "raw" GPS positions to the underlying road network map matching algorithms can be used. For an overview of existing map matching algorithms we refer to [16]. In the following, we do not consider the map matching any more, but assume that π is located on a road segment of the graph.

A movement trajectory T of the MO is defined as sequence $T = \{(\pi_{start}, t_{start}), \ldots, (\pi_{end}, t_{end})\}$ of different positions π_i where the MO was located at time t_i. Trajectory T can be split up into a set of segments, where each segment $\overline{\pi_i, \pi_{i+1}}$ defines the path the MO traveled from π_i at time t_i to π_{i+1}

at time t_{i+1}. We assume that MOs travel on fastest paths, since usually a MO intends to reach its destination as quick as possible. Therefore, $\overline{\pi_i, \pi_{i+1}}$ is the fastest path between π_i and π_{i+1}. The distance between π_i and π_{i+1} is

$$distance(\overline{\pi_i, \pi_{i+1}}) = length(FP(\pi_i, \pi_{i+1})) \quad (1)$$

defining the length of the fastest path (FP) from π_i to π_{i+1}.

The time the MO traveled on segment $\overline{\pi_i, \pi_{i+1}}$ is

$$time(\overline{\pi_i, \pi_{i+1}}) = t_{i+1} - t_i. \quad (2)$$

III. Problem Statement

We tackle the problem that real-time position updates of a MO's movement trajectory T reveal speeding violations of the MO. To solve this problem, we use our speed protection algorithms transforming trajectory T into the speed protected trajectory $\widehat{T}$ (introduced below) guaranteeing that nobody can determine any speeding violation by analyzing $\widehat{T}$.

For trajectory $T = \{(\pi_{start}, t_{start}), \ldots, (\pi_{end}, t_{end})\}$, a speeding violation is detected if for at least one segment $\overline{\pi_i, \pi_{i+1}} \in T$ the MO traveled from π_i to π_{i+1} within shorter time than it takes the MO when driving at the maximum allowed speed of the segment:

$$time(\overline{\pi_i, \pi_{i+1}}) < timeMaxSpeed(\overline{\pi_i, \pi_{i+1}}). \quad (3)$$

This concept is also known as "section control", where the average speed over a certain known distance is calculated. Here, $timeMaxSpeed(\overline{\pi_i, \pi_{i+1}})$ defines the time it takes a MO driving from π_i to π_{i+1} at the maximum allowed speed.

An attacker is interested in finding segment $\overline{\pi_i, \pi_{i+1}} \in T$ indicating that the MO was speeding:

$$\begin{aligned} &\exists\ \overline{\pi_i, \pi_{i+1}} \in T : \\ &time(\overline{\pi_i, \pi_{i+1}}) < timeMaxSpeed(\overline{\pi_i, \pi_{i+1}}). \end{aligned} \quad (4)$$

As stated previously, $\overline{\pi_i, \pi_{i+1}}$ defines the fastest path from π_i to π_{i+1}. If a speeding violation is detected for $\overline{\pi_i, \pi_{i+1}}$, then the MO is also speeding for all other slower paths from π_i to π_{i+1}. Thus, we call this a definitely speeding semantic, where the MO has no plausible way of denying a speeding violation.

The goal of our speed protection algorithms is to transform trajectory T into a protected trajectory $\widehat{T}$ such that for every segment $\overline{\pi_i, \pi_{i+1}}$ at least one possible path exists where the MO could have traveled from π_i to π_{i+1} without speeding. Then, the definitely speeding semantic is not fulfilled for $\widehat{T}$:

$$\begin{aligned} &\forall\ \overline{\pi_i, \pi_{i+1}} \in \widehat{T} : \\ &time(\overline{\pi_i, \pi_{i+1}}) \geq timeMaxSpeed(\overline{\pi_i, \pi_{i+1}}). \end{aligned} \quad (5)$$

Obviously, transforming the original trajectory T into the speed protected trajectory $\widehat{T}$ introduces artificial inaccuracies. Therefore, another goal is to alter T as less as possible to reduce the introduced inaccuracy. Later, we will show that the speed protection algorithms either lead to spatial or temporal inaccuracies for positions of $\widehat{T}$. Therefore, the spatial or temporal difference should be as small as possible.

By guaranteeing that the definitely speeding semantic is not fulfilled for the published trajectory $\widehat{T}$, the MO can plausibly deny a speeding violation since the MO could have traveled on each segment of trajectory $\widehat{T}$ without driving faster than the allowed speed. In combination with the "in dubio pro reo" principle, this fulfills our goal to protect the speed of a MO's movement trajectory $\widehat{T}$ because no evidences of a speeding violation exists as long as the MO could have traveled along $\widehat{T}$ on the fastest path without any speeding violation.

IV. Speed Protection Algorithms

In this section, we present our speed protection algorithms (*SPA* for short) guaranteeing that the continuously updated trajectory $\widehat{T}$ of the MO does not contain any speeding violation.

A. Overview

The general idea of *SPA* is to slow down the speed of trajectory $\widehat{T}$ to the maximum speed the MO is allowed to drive on each road segment. Since location update protocols are usually either time-based or distance-based, we use the methods of *position adjustment* (PA) and *temporal delay* (TD) to support both kinds of protocols.

We use position adjustment in *SPA-PA* to support time-based position update protocols, which periodically update π_{i+1} after a predefined update time period *TP*. The constant update rate is independent of the MO's position change and sends position updates even if the position of the MO did not change. Our position adjustment shifts position π_{i+1} to position $\widehat{\pi}_{i+1}$ if a speeding violation happened between the last updated position $\widehat{\pi}_i$ and the currently sensed position π_{i+1}. Here, position adjustment protects the speed of the MO by decreasing the spatial accuracy of π_{i+1} such that the speed protected position $\widehat{\pi}_{i+1}$ is updated instead of π_{i+1} at time t_{i+1}. As soon as the speed of the MO is below the speed limit, the spatial accuracy is increased until position $\widehat{\pi}_{i+1}$ is equal to π_{i+1} and the accurate position of the MO can be updated again without revealing a speeding violation.

We use temporal delays in *SPA-TD* to support distance-based position update protocols taking the traveled distance of the MO into account. With distance-based protocols a new position π_{i+1} is updated whenever the Euclidean distance to the last reported position π_i reaches a given threshold distance D. If a speeding violation occurred between the last updated position π_i and the new sensed position π_{i+1}, the update of position π_{i+1} is delayed until time $\widehat{t}_{i+1}$ such that no speeding violation can be recognized between π_i updated at time $\widehat{t}_i$ and π_{i+1} updated at time $\widehat{t}_{i+1}$. Our temporal delay keeps the spatial information of the position accurate and decreases the temporal accuracy of the update. As soon as the speed of the MO is reduced and the MO drives slower than the allowed speed limit, the temporal accuracy is increased until the timestamps of the sensed and updated positions are identical and no delay is needed any more.

Figure 3 shows an overview of the complete process. The MO senses its position π_{i+1} at time t_{i+1} using the positioning sensor. For time-based updates, *SPA-PA* uses the position adjustment method and provides position $\widehat{\pi}_{i+1}$ at time t_{i+1}

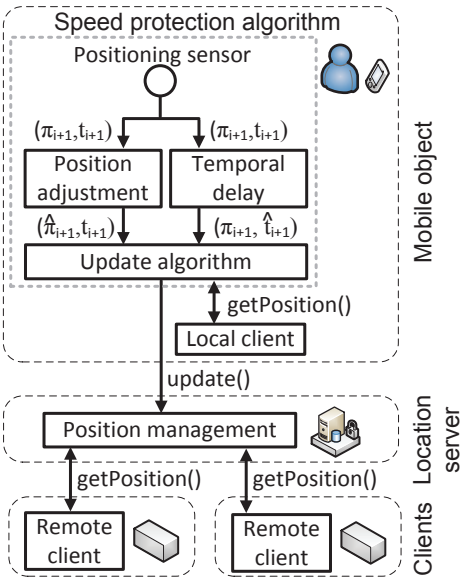

Fig. 3. Process overview

to the update algorithm. For distance-based updates, *SPA-TD* uses the temporal delay method to provide position π_{i+1} at the delayed time $\widehat{t}_{i+1}$ to the update algorithm. Then, function $update(\widehat{\pi}_{i+1}, t_{i+1})$ respectively $update(\pi_{i+1}, \widehat{t}_{i+1})$ is used by the update algorithm to provide the new position information of the MO to the LS. Finally, clients can access this information by using function $getPosition()$. Next, we present the two protection methods in more detail.

B. SPA with Position Adjustment

The detailed speed protection algorithm *SPA-PA* for time-based updates is shown in Fig. 5. First, the initial position π_{start} is sensed at time t_{start} and updated as position $\widehat{\pi}_{start}$ by using function $update(\widehat{\pi}_{start}, t_{start})$ at time t_{start}. Afterwards, a new position π_{i+1} is sensed at time $t_{i+1} = t_i + TP$ based on the MO-selected update time period *TP*. Next, *SPA-PA* uses the last updated position $\widehat{\pi}_i$ to evaluate $time(\overline{\widehat{\pi}_i, \pi_{i+1}}) < timeMaxSpeed(\overline{\widehat{\pi}_i, \pi_{i+1}})$. If the MO reached π_{i+1} from $\widehat{\pi}_i$ within shorter time than it takes a MO driving at the maximum allowed speed, then updating π_{i+1} would reveal a speeding violation. In this case position π_{i+1} has to be adjusted before it can be published. To this end, *SPA-PA* uses function $getReachablePosition(\widehat{\pi}_i, \delta t, T)$ to calculate position $\widehat{\pi}_{i+1}$ as the position that can be reached from the last updated position $\widehat{\pi}_i$ within time $\delta t = time(\overline{\widehat{\pi}_i, \pi_{i+1}})$ when driving at the maximum allowed speed on segment $\overline{\widehat{\pi}_i, \pi_{i+1}}$. In case no speeding violation is detected, position π_{i+1} is used as position $\widehat{\pi}_{i+1}$. Finally, the calculated position $\widehat{\pi}_{i+1}$ is updated using function $update(\widehat{\pi}_{i+1}, t_{i+1})$.

An example of *SPA-PA* for time-based updates is presented in Fig. 4. For simplicity, we use a fixed value of $maxSpeed(e) = 100\,\text{km/h}$ for each edge e and assume that a new position is sensed after a time period of $TP = 5\,\text{s}$. Without *SPA-PA*, trajectory T shown in Fig. 4 would be published. The MO drives with a speed of 80 km/h at time

1: $\pi_{start}, t_{start} \leftarrow$ sensed position ▷ Initial positioning
2: $T \leftarrow \pi_{start}, t_{start}$ ▷ Sensed trajectory
3: $\widehat{\pi}_{start} \leftarrow \pi_{start}$ ▷ Initial position
4: $update(\widehat{\pi}_{start}, t_{start})$ ▷ Initial position update
5: **while** report movement **do**
6: $\pi_{i+1}, t_{i+1} \leftarrow$ sensed position ▷ Triggered time-based
7: $T \leftarrow T \cup \pi_{i+1}, t_{i+1}$
8: $\widehat{\pi}_i, t_i \leftarrow$ last updated position
9: **if** $time(\overline{\widehat{\pi}_i, \pi_{i+1}}) < timeMaxSpeed(\overline{\widehat{\pi}_i, \pi_{i+1}})$ **then**
10: $\delta t \leftarrow time(\overline{\widehat{\pi}_i, \pi_{i+1}})$ ▷ Speeding detected
11: $\widehat{\pi}_{i+1} \leftarrow getReachablePosition(\widehat{\pi}_i, \delta t, T)$
12: **else**
13: $\widehat{\pi}_{i+1} \leftarrow \pi_{i+1}$ ▷ No speeding occurred
14: **end if**
15: $update(\widehat{\pi}_{i+1}, t_{i+1})$ ▷ Update $\widehat{\pi}_{i+1}$ at time t_{i+1}
16: **end while**

Fig. 5. SPA with position adjustment

t_3. At time t_4, the MO accelerates up to a speed of 120 km/h at time t_5 and travels at this speed until time t_6. Then, the MO slows down to 80 km/h at time t_7 that is kept until time t_{10}. The corresponding time-speed diagram is shown in Fig. 4 on the right. By using *SPA-PA*, the maximum speed between two updated positions in $\widehat{T}$ is limited to the maximum allowed speed of 100 km/h. The generated updates are shown in Fig. 4 with the corresponding time-speed diagram of the speed protected trajectory $\widehat{T}$. While actually a speeding violation of the MO occurred, the reported speed is limited to the maximum allowed speed limit. The time after the MO's speeding violation is used to reduce the spatial difference between the updated and the actual position of the MO.

C. SPA with Temporal Delay

The speed protection algorithm *SPA-TD* for distance-based updates is shown in Fig. 7. The positioning sensor provides a new position to *SPA-TD* as soon as the Euclidean distance between the last sensed position π_i and the new sensed position π_{i+1} reaches the MO-defined threshold distance D. The initial position is updated as described for *SPA-PA*. For the following position updates, it is evaluated whether position π_{i+1} sensed at time t_{i+1} can be reached in time $time(\overline{\pi_i, \pi_{i+1}}) = t_{i+1} - \widehat{t}_i$ from the last updated position π_i updated at time $\widehat{t}_i$. A speeding violation is detected if $time(\overline{\pi_i, \pi_{i+1}}) < timeMaxSpeed(\overline{\pi_i, \pi_{i+1}})$. Then, the update of position π_{i+1} must be delayed until π_{i+1} can be reached from π_i without exceeding the speed limit. To this end, *SPA-TD* calculates time $\widehat{t}_{i+1}$ at which the MO can reach π_{i+1} from π_i without speeding violation. *SPA-TD* assumes a MO driving at the maximum allowed speed from π_i to π_{i+1} and uses the minimum required time $timeMaxSpeed(\overline{\pi_i, \pi_{i+1}})$ to calculate $\widehat{t}_{i+1}$ as $\widehat{t}_{i+1} = \widehat{t}_i + timeMaxSpeed(\overline{\pi_i, \pi_{i+1}})$. The update of position π_{i+1} is then delayed until time $\widehat{t}_{i+1}$. Otherwise, if π_{i+1} can be reached from π_i without a speeding violation, time t_{i+1} is used to define $\widehat{t}_{i+1} = t_{i+1}$. Finally, position π_{i+1} is updated at time $\widehat{t}_{i+1}$ using function $update(\pi_{i+1}, \widehat{t}_{i+1})$. After

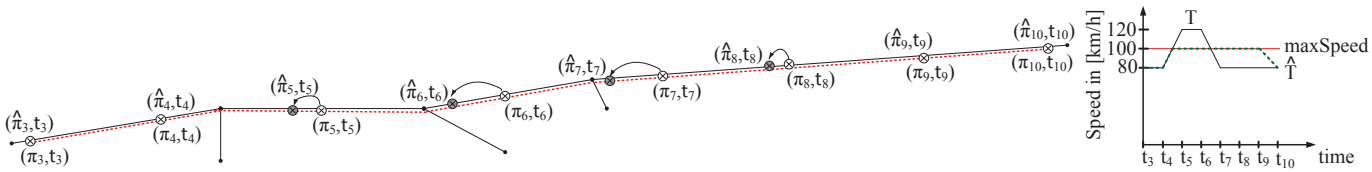

Fig. 4. Example for *SPA-PA* using time-based updates

1: $\pi_{start}, t_{start} \leftarrow$ sensed position ▷ Initial positioning
2: $\widehat{t}_{start} \leftarrow t_{start}$ ▷ Initial update time
3: $update(\pi_{start}, \widehat{t}_{start})$ ▷ Initial position update
4: **while** report movement **do**
5: $\pi_{i+1}, t_{i+1} \leftarrow$ sensed position ▷ Triggered distance-b.
6: $\pi_i, \widehat{t}_i \leftarrow$ last updated position
7: **if** $time(\overline{\pi_i, \pi_{i+1}}) < timeMaxSpeed(\overline{\pi_i, \pi_{i+1}})$ **then**
8: $\widehat{t}_{i+1} \leftarrow \widehat{t}_i + timeMaxSpeed(\overline{\pi_i, \pi_{i+1}})$ ▷ Speed.
9: **else**
10: $\widehat{t}_{i+1} \leftarrow t_{i+1}$ ▷ No speeding occurred
11: **end if**
12: $update(\pi_{i+1}, \widehat{t}_{i+1})$ ▷ Update π_{i+1} at time $\widehat{t}_{i+1}$
13: **end while**

Fig. 7. SPA with temporal delay

a speeding violation occurred and the MO is again driving at a speed below the speed limit, the temporal accuracy is increased until no delay is needed any more for a new position update.

An example for *SPA-TD* is shown in Fig. 6. For this example, we use the same MO movement and the same speed limitations as presented in our previous example for *SPA-PA*. Compared to the former example, the sensing of a new position is triggered based on the MO-defined threshold distance D that is set to $D = 100\,m$. The sensed positions and the corresponding position updates are shown in Fig. 6. As shown, the spatial position information of each sensed position is kept accurate, while the point in time used to update the position is adjusted to protect the speed information of the MO.

V. Proof of Correctness

In this section, we prove that the MO's movement trajectory $\widehat{T}$ generated by our speed protection algorithm does not indicate any speeding violation to an attacker. First, we consider *SPA-PA* and proceed afterwards with *SPA-TD*.

A. *Proof for* SPA-PA

We prove the correctness of *SPA-PA* by contradiction. Assume that there exists a segment in the published trajectory $\widehat{T}$ of the MO indicating a speeding violation. Furthermore, assume that trajectory $\widehat{T} = \{(\widehat{\pi}_{start}, t_{start}), \ldots, (\widehat{\pi}_{end}, t_{end})\}$ generated by *SPA-PA* consists of at least two position updates, which is the minimum number of position updates required to derive speed information. Following our assumption, at least one segment $\overline{\widehat{\pi}_i, \widehat{\pi}_{i+1}} \in \widehat{T}$ has to exists indicating a speeding violation of the MO. Without loss of generality, let position $\widehat{\pi}_{i+1}$ updated at time t_{i+1} be the first update indicating a speeding violation. As we can see in Fig. 5, $\widehat{\pi}_{i+1}$ can only be updated using function $update(\widehat{\pi}_{i+1}, t_{i+1})$ (cf. line 15). Moreover, our assumption requires that the MO drove faster than the allowed speed limit on segment $\overline{\widehat{\pi}_i, \widehat{\pi}_{i+1}}$. However, because *SPA-PA* detects the speeding violation for position π_{i+1} at time t_{i+1} (cf. line 9), the updated position $\widehat{\pi}_{i+1}$ is calculated as the position that can be reached from the last updated position $\widehat{\pi}_i$ without speeding violation (cf. line 11). Therefore, the updated position $\widehat{\pi}_{i+1}$ cannot indicate a speeding violation on segment $\overline{\widehat{\pi}_i, \widehat{\pi}_{i+1}}$. This contradicts our assumption that segment $\overline{\widehat{\pi}_i, \widehat{\pi}_{i+1}} \in \widehat{T}$ indicates a speeding violation. Thus, it is guaranteed that trajectory $\widehat{T}$ does not contain any speeding violations.

B. *Proof for* SPA-TD

For *SPA-TD* we can show in a similar way to *SPA-PA* that no segment $\overline{\pi_i, \pi_{i+1}} \in \widehat{T}$ can exist indicating a speeding violation of the MO. Assume trajectory $\widehat{T}$ provided to the LS consists again of at least two position updates. Moreover, assume that position π_{i+1} updated at time $\widehat{t}_{i+1}$ is the first position update indicating a speeding violation. The only function updating the MO's position π_{i+1} in Fig. 7 is function $update(\pi_{i+1}, \widehat{t}_{i+1})$ (cf. line 12). However, a speeding violation is detected by *SPA-TD* (cf. line 7) and the update of position π_{i+1} is delayed until time $\widehat{t}_{i+1}$ (cf. line 8). Therefore, the update of position π_{i+1} does not provide any information that the MO drove faster than the allowed speed limit. This contradicts our assumption that segment $\overline{\pi_i, \pi_{i+1}} \in \widehat{T}$ indicates a speeding violation of the MO and we can state that trajectory $\widehat{T}$ generated by *SPA-TD* protects the speed information of the MO.

VI. Evaluation

In this section, we present our real world trace evaluation analyzing the speed information of taxi cabs in the San Francisco Bay Area. Moreover, we evaluate the runtime performance of our speed protection algorithms using a prototype implementation on a state of the art mobile device.

A. *Analysis of Real World Traces for Speeding Violations*

For our evaluation, we use the mobility traces of taxi cabs in San Francisco, USA, provided by [15]. The dataset consists of the movement trajectories of approximately 500 taxi cabs collected over 30 days in 2008 in the San Francisco Bay Area. Each trajectory defines fine grained positions of a certain taxi cab. The position of each taxi cab is measured by its GPS receiver and updated within an update time of less than 10 s on average. We randomly select the time period of one day (2008/06/01) and analyze the behavior of the taxi cabs for

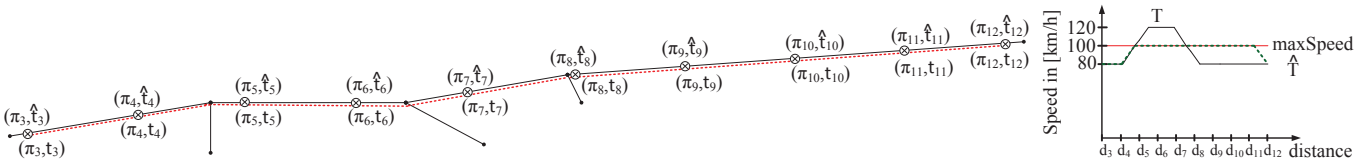

Fig. 6. Example for *SPA-TD* using distance-based updates

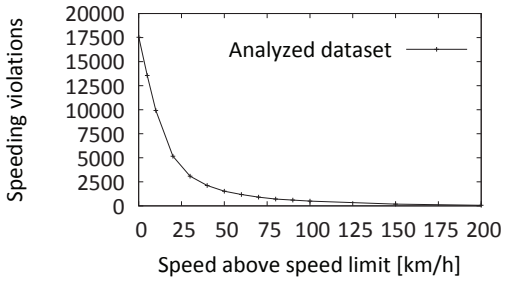

Fig. 8. Speeding violation analysis

this day. We use the map information of the OpenStreetMap-Project [14] providing road network data, speed limits, and further information to determine the speed limits and distances for each road segment.

First, we analyze the driving behavior and the occurred speeding violations that can be derived from the movement trajectories. To this end, we distinguish for each position update whether it indicates a speeding violation or not by analyzing the travel time and distance between succeeding updates as formalized in (3). Overall, we analyzed 365 348 position updates of 484 different taxi cabs. 347 818 of the updates (95.20%) conform to the local speed limits, while 17 530 position updates (4.80%) indicated a speeding violation.

We further analyzed these speeding violations and calculated the speed $\Delta v_{speeding} = v - v_{max}$ as the difference between the MO's speed v and the allowed speed limit v_{max}. The results are shown in Fig. 8, where we plotted the number of detected speeding violations over $\Delta v_{speeding}$. As we can see, the majority (87,93%) of speeding violations occurred for $\Delta v_{speeding} \leq 40$ km/h. On average, $\Delta v_{speeding}$ is 20.47 km/h for all detected speeding violations. From these results we can see that protecting the speed information of movement trajectories is a relevant problem.

B. Analysis of Introduced Inaccuracy

Next, we formalize the spatial and temporal inaccuracy introduced by *SPA-PA* and *SPA-TD*. The protection of the speed information leads to an artificial inaccuracy in case the MO drives faster than the allowed speed limit. We define $\Delta v^{max}_{speeding}$ as the maximum speed difference between the MO's speed v and the allowed speed limit v_{max}. Therefore, $\Delta v^{max}_{speeding}$ depends on the individual driving behavior of the MO. Furthermore, we define position $\pi_j \in T$ measured at time t_j as the last measured position in T where the MO drove below the allowed speed limit. Then, we can calculate for each position $\widehat{\pi}_k \in \widehat{T}$ updated at time $t_k \geq t_j$ the maximum spatial inaccuracy introduced by *SPA-PA* as

$$\Delta d_{max}(\pi_j, \widehat{\pi}_k) = time(\overline{\pi_j, \widehat{\pi}_k}) * \Delta v^{max}_{speeding}. \quad (6)$$

For *SPA-TD*, the maximum introduced temporal inaccuracy for each position $\pi_k \in \widehat{T}$ and the last measured non-speeding position π_j is calculated as

$$\Delta t_{max}(\pi_j, \pi_k) = \frac{distance(\overline{\pi_j, \pi_k}) * \Delta v^{max}_{speeding}}{(v_{max})^2 + (v_{max} * \Delta v^{max}_{speeding})}. \quad (7)$$

As we can see, the maximum spatial and temporal deviation introduced by our speed protection algorithms depends on the MO's driving behavior defining $\Delta v^{max}_{speeding}$ and the duration of the speeding violation of the MO respectively the traveled distance between position π_j and position π_k.

To get an insight into real world user driving behavior, we analyze the introduced spatial and temporal inaccuracy introduced by *SPA-PA* and *SPA-TD* for the introduced real-world dataset. To this end, we analyze the spatial inaccuracy introduced by *SPA-PA* by measuring the Euclidean distance between position $\widehat{\pi}_{i+1}$ calculated by *SPA-PA* and the original position π_{i+1} which would be updated without speed protection algorithm. Formally, the spatial inaccuracy is calculated for time t_{i+1} as

$$\Delta d(t_{i+1}) = distance_{eucl}(\widehat{\pi}_{i+1}, \pi_{i+1}). \quad (8)$$

The temporal inaccuracy introduced by *SPA-TD* is the time between t_{i+1} when position π_{i+1} is updated without *SPA-TD* and time $\widehat{t}_{i+1}$ *SPA-TD* updates π_{i+1}. The temporal inaccuracy is calculated for position π_{i+1} as:

$$\Delta t(\pi_{i+1}) = \widehat{t}_{i+1} - t_{i+1}. \quad (9)$$

Because the taxi cabs of the analyzed dataset triggered their position updates in an irregular manner, the used update strategy is neither strictly distance-based nor time-based. However, since the algorithms are also applicable for irregular update intervals and distances, we used the provided positions from the original dataset with the original timestamps.

We analyze *SPA-PA* by calculating the spatial inaccuracy $\Delta d(t_{i+1})$ for each position update π_{i+1} triggered at time t_{i+1}. As shown in Fig. 9, 93.3% of the position updates have an inaccuracy below 100 m. Therefore, we can state that the speed protected movement trajectories are of high quality, and only few updates have a low spatial accuracy.

For our analysis of *SPA-TD* we calculate the temporal inaccuracy $\Delta t(\pi_{i+1})$ for each position update triggered by position π_{i+1}. The evaluation results are shown in Fig. 10. As we can see, 94.56% of the position updates have a temporal

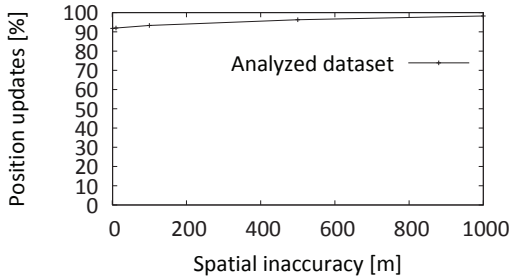

Fig. 9. Cumulative distribution of spatial inaccuracy

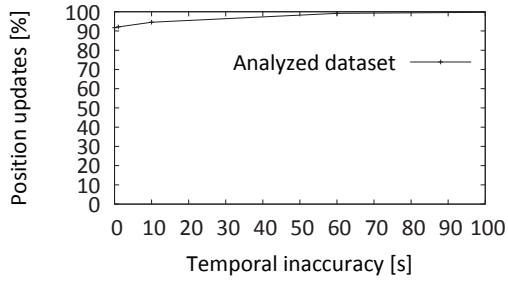

Fig. 10. Cumulative distribution of temporal inaccuracy

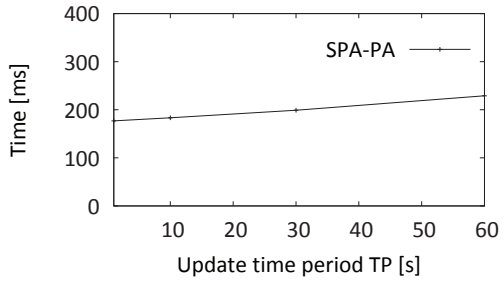

Fig. 11. Performance evaluation using time-based updates

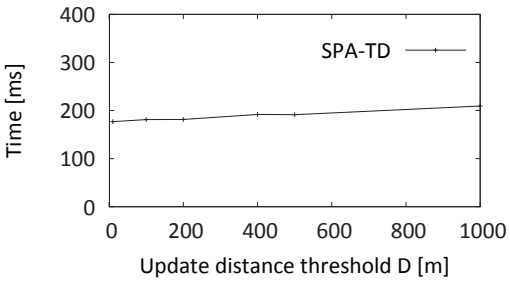

Fig. 12. Performance evaluation using distance-based updates

inaccuracy below 10 s. An inaccuracy of 60 s covers 99.14% of all updates. Therefore, we can state that the temporal inaccuracy introduced by *SPA-TD* is low.

C. Performance Evaluation

Next, we analyze the efficiency of *SPA-PA* and *SPA-TD*. Since the protection algorithms have to be executed online on the mobile device, which typically has relatively low computational power, efficiency is a critical factor for the algorithms. Moreover, a small computational overhead also reduces the energy consumption, which is desirable for battery-operated mobile devices.

We evaluate *SPA-PA* and *SPA-TD* using a prototype implemented on a state of the art mobile device (HTC Desire HD). The used map is the road network of Stuttgart, Germany. The time required to calculate a new speed protected position depends on the selected update protocol and its update parameter.

For *SPA-PA*, we analyze randomly selected positions on the map with different update time periods *TP* ranging from 1 s to 60 s. Figure 11 shows the evaluation results for a movement trajectory over several runs for each measurement. As we can see, the required time to determine a new speed protected position increases by increasing the update period *TP*. This is based on the fact that the speed protection algorithm has to calculate the fastest path on the road network from the last updated position to the currently evaluated position. By increasing *TP*, the traveled distance also increases. The measured time includes the time required to calculate the map matching on the road network for the sensed position of the MO. The map matching is required as position π provided by the positioning system has to be mapped to the road network before it can be used by our speed protection algorithms. On average, the map matching takes 150 milliseconds. As the introduced calculation of *SPA-PA* for time-based position updates is well below 250 milliseconds even for larger update time periods, we can state that protecting the speed information of a MO is possible without affecting the real-time property of position updates.

For *SPA-TD*, we vary the threshold distance D from 10 m to 500 m. As shown in Fig. 12, the processing time of *SPA-TD* stays well below 250 milliseconds. The time for the map matching part of *SPA-TD* is identical to the evaluation of *SPA-PA*. Generally, calculating the temporal delay of a position update in *SPA-TD* takes nearly the same time than calculating the next reachable position not indicating a speeding violation in *SPA-PA*. This is based on the fact that the functions used in the introduced algorithms mainly differ only in line 11 in Fig. 5 and line 8 in Fig. 7, where we calculate the next reachable position and the next reachable point in time that can be used without indicating a speeding violation for the next update. Therefore, *SPA-TD* can also be calculated very fast and support real-time position updates.

D. Analysis of Update Costs

As we can see from Fig. 5 and Fig. 7, each sensed position of the MO leads to exactly one (protected) position update. Therefore, the update costs of *SPA-PA* and *SPA-TD* are identical to the update costs without speed protection.

To reduce the number of required position updates sent to the LS, advanced update protocols using dead reckoning [12] can be used. Dead-reckoning is an optimization of the distance-based update protocols where the LS estimates the position of the MO based on the last known position, its speed, and its movement direction. The MO calculates the same estimation as the LS and updates its position as soon as the actual position differs from the estimated position by

more than a certain distance threshold DT. To this end, the update algorithm implementing the dead-reckoning algorithm periodically evaluates its update criteria based on the time-based update protocol of *SPA-PA*. Thus, *SPA-PA* can be used also for dead-reckoning based updates.

VII. Related Work

Location privacy approaches can be distinguished based on their used privacy principle and their protection goal. For an overview of different privacy approaches and possible location privacy attacks we refer to [21]. The most prominent privacy principle is k-anonymity [9], [13] trying to make the user indistinguishable from $k-1$ other users such that the identity of the user is protected. That is, k-anonymity tries to prevent an attacker from linking a trajectory—which is not modified and therefore contains all speeding violations—to an individual user. However, as shown in [8], spatial and temporal information can be used to identify individual users and therefore to identify individual speeders.

Spatial obfuscation approaches [1], [4] protect the position information of the user by decreasing precision. However, even obfuscated positions can possibly reveal speeding violations such that these approaches do not protect the speed information of the user. This applies also to position sharing approaches [5], [20] providing different precision levels to different clients while non-trusted LSs manage only positions of limited precision.

Dummy approaches [10], [17] send the true user position with additionally generated false positions to the LS to protect the movement trajectory of the user. However, dummy approaches taking user movement into account do not consider speed information of dummies. Therefore, if all dummies and also the true user trajectory contain speeding violations, an attacker knows that the user is definitely speeding. As the dummy approach is orthogonal to our approach, our speed protection algorithms could be added to protect dummies from revealing speeding violations.

VIII. Conclusion and Future Work

In this paper, we presented speed protection algorithms protecting users from revealing a violation of given speed limits when using location-based applications. The basic idea to protect time-based position updates is to adjust the spatial information of the user position such that the updated position does not indicate a speeding violation. For distance-based position updates the basic idea is to delay updates that would indicate that the user drove faster than the current speed limit.

For our evaluation we used a large real world dataset of taxi cabs providing precise position information. We analyzed this dataset and showed that protecting speed information is important to prevent revealing speeding violations. Furthermore, we used the dataset to measure the introduced spatial and temporal inaccuracy of our speed protection algorithms and showed that 99.14% of the distance-based updates are covered by a temporal inaccuracy of 60 s, and that 93.3% of the time-based updates are covered by a spatial inaccuracy of 100 m. Finally, we proved that our algorithms can run in real-time on a state of the art mobile device.

In future work, we will analyze the impact of our speed protection algorithms to advanced location update protocols such as map-based dead reckoning [11].

References

[1] C. A. Ardagna, M. Cremonini, and G. Gianini, "Landscape-aware location-privacy protection in location-based services," *Journal of Systems Architecture*, vol. 55, no. 4, pp. 243–254, April 2009.

[2] P. A. Bonatti and P. Samarati, "A uniform framework for regulating service access and information release on the web," *Journal of Computer Security*, vol. 10, no. 3, pp. 241–271, Jan. 2002.

[3] Buddyway, www.buddyway.com, Nov. 2012.

[4] M. Damiani, C. Silvestri, and E. Bertino, "Fine-grained cloaking of sensitive positions in location-sharing applications," *Pervasive Computing*, vol. 10, no. 4, pp. 64 –72, Apr. 2011.

[5] F. Dürr, P. Skvortsov, and K. Rothermel, "Position sharing for location privacy in non-trusted systems," in *Proceedings of the 9th IEEE International Conference on Pervasive Computing and Communications (PerCom '11)*, Mar. 2011, pp. 189–196.

[6] Foursquare, www.foursquare.com, Nov. 2012.

[7] P. Gilbert, L. P. Cox, J. Jung, and D. Wetherall, "Toward trustworthy mobile sensing," in *Proceedings of the 11th Workshop on Mobile Computing Systems & Applications (HotMobile '10)*, 2010, pp. 31–36.

[8] P. Golle and K. Partridge, "On the anonymity of home/work location pairs," in *Proceedings of the 7th International Conference on Pervasive Computing (Pervasive '09)*, 2009, pp. 390–397.

[9] 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*, vol. 19, no. 12, pp. 1719 –1733, Dec. 2007.

[10] H. Kido, Y. Yanagisawa, and T. Satoh, "An anonymous communication technique using dummies for location-based services," in *Proceedings of the International Conference on Pervasive Services (ICPS '05)*, Jul. 2005, pp. 88–97.

[11] A. Leonhardi, C. Nicu, and K. Rothermel, "A map-based dead-reckoning protocol for updating location information," in *Proceedings of the 16th International Parallel and Distributed Processing Symposium (IPDPS '02)*, 2002, pp. 193 –200.

[12] A. Leonhardi and K. Rothermel, "A comparison of protocols for updating location information," *Cluster Computing*, vol. 4, no. 4, pp. 355–367, Oct. 2001.

[13] M. F. Mokbel, C.-Y. Chow, and W. G. Aref, "The new casper: query processing for location services without compromising privacy," in *Proceedings of the 32nd international conference on Very large data bases (VLDB '06)*, 2006, pp. 763–774.

[14] OpenStreetMap, www.openstreetmap.org, Nov. 2012.

[15] 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 (COMSNETS '09)*, Jan. 2009, pp. 1–10.

[16] M. A. Quddus, W. Y. Ochieng, and R. B. Noland, "Current map-matching algorithms for transport applications: State-of-the art and future research directions," *Transportation Research Part C: Emerging Technologies*, vol. 15, no. 5, pp. 312 – 328, Oct. 2007.

[17] P. Shankar, V. Ganapathy, and L. Iftode, "Privately querying location-based services with sybilquery," in *Proceedings of the 11th Int. Conference on Ubiquitous computing (Ubicomp '09)*, no. 10, 2009, pp. 31–40.

[18] The Telegraph, http://www.telegraph.co.uk/technology/news/8480195/Police-use-TomTom-data-to-target-speed-traps.html, 2012.

[19] USA Today, http://www.usatoday.com/tech/news/2001-07-03-car-tracking.htm, 2012.

[20] M. Wernke, F. Dürr, and K. Rothermel, "PShare: ensuring location privacy in non-trusted systems through multi-secret sharing," *Pervasive and Mobile Computing*, vol. 9, no. 3, pp. 339 – 352, 2013.

[21] M. Wernke, P. Skvortsov, F. Dürr, and K. Rothermel, "A classification of location privacy attacks and approaches," *Personal and Ubiquitous Computing*, pp. 1–13, Nov. 2012.