2016 12th European Dependable Computing Conference

A System-level Architecture for Fine-grained Privacy Control in Location-based Services

Arielle Moro* Benoît Garbinato†
Distributed Object Programming Laboratory
University of Lausanne
CH-1015 Lausanne, Switzerland
Email: *arielle.moro@unil.ch †benoit.garbinato@unil.ch

***Abstract*—We introduce a system-level architecture providing fine-grained control over user privacy, in the context of location-based services accessed via mobile devices. In contrast with most mobile platforms today, users only have coarse-grained control over their privacy, either accepting to unconditionally stream their locations in order to use a service, or renouncing the service altogether. However, not all location-based services do require the same level of location accuracy and the same level of privacy renouncement. With this architecture, the user can adapt the tradeoff between location privacy and location accuracy. To achieve this, our architecture relies on three main elements: a trusted module extending the underlying mobile platform, a secure protocol between that module and untrusted applications offering location-based services, and a tree capturing user's zones of interest and organizing them in various accuracy levels. Untrusted mobile applications no longer receive user locations directly: the trusted module intercepts them to compute user's zones of interest and create the tree. The user can then decide what level of accuracy will be disclosed to what application. We evaluate this architecture from a privacy preserving point of view by comparing well-known blurring mechanisms and our tree.**

***Keywords*—location privacy, location-based services, privacy tree, zones of interest, system-level architecture**

I. Introduction

In less than a decade, mobile devices, in particular smartphones, have radically changed the way we consume digital services, be they web searching, online gaming, media streaming, etc. Basically, we can now access such services whenever we want and wherever we are. Furthermore, the ability of mobiles devices to locate themselves, either via the Global Positioning System (GPS) when outside or via WiFi or 3G/4G when indoor, has given birth to a new breed of digital services, so-called *location-based services*.

A. For better or for worse?

There is little doubt that location-based services can be very useful, ranging from simple one-shot queries about one's surroundings, e.g., when looking for nearby restaurants using applications such as Google Maps or Apple Maps,[1] to continuously tracking one's movements, e.g., while jogging using applications such as Runkeeper or Runtastic.[2] Location-based services come however at a price: *loss of location privacy*.

It is important to stress that location privacy is just one of many facets of privacy as a whole, yet a crucial one, and that this paper focuses exclusively on location privacy. Indeed, locations generated by mobile devices offer a powerful means to link virtually any data associated with a user to a very tangible aspect of her life: her physical location in the real world. For this reason, most mobile platforms, such as Apple iOS or Google Android, are trying to make users aware of this potential privacy loss, by explicitly asking them whether they are willing to share their locations with applications requesting them, either at installation time or at run time.

Unfortunately, this choice is usually binary: either the application is granted full access to the user locations or no access at all. Furthermore, once access has been granted, the user has no control over what the application will do with her location information. It could theoretically confine this information to the user mobile device, which is almost never the case, or forward it to some backend server, in order to compile statistics, use it later or even sell it to some third party.

A key problem here is that by giving away location information, users are actually revealing a lot about themselves, most of the time without even realizing it. In [22] for instance, it is shown that although most users may occasionally exhibit spontaneous behaviors, their moves bear strong regularity, which leads to high predictability of their future locations. Furthermore, according to [6] only four spatio-temporal coordinates are enough to uniquely identify 95% of the users of the dataset. As mentioned in the paper, this dataset contains locations, location antenna more specifically, of a large number of users, approximately 1.5 M. These locations were caught when users used their mobile device (e.g., when a user receives or initiates a call or a text message). Additionally, another very interesting paper [23] demonstrates the consequence of privacy leakage via a quantification of social inference from it. They found 90% inference accuracy concerning social and community relationships with only three week's user data by using their leakage inference framework they create. The dataset used in this paper contains real mobility traces and Foursquare data.

B. Location accuracy vs. privacy

When it comes to locations, a rather straightforward way to control *privacy* consists in controlling *accuracy*: the lower the location accuracy, the higher the location privacy. Obviously, knowing that Alice is located within a 100 meters range around the cathedral of Notre-Dame de Paris damages her privacy more than knowing that she stands within a 10 kilometers range (which boils down to simply say that Alice is somewhere

[1] http://google.ch/maps, http://apple.com/ios/maps
[2] http://runkeeper.com, http://runtastic.com

978-1-5090-1582-5/16 $31.00 © 2016 IEEE
DOI 10.1109/EDCC.2016.24

in Paris). Furthermore, the privacy level associated with Alice's location is not only dependent on its geographical dimension but also on its time dimension: knowing that Alice was located within a range of 100 meters around Notre-Dame de Paris *between 8:00 AM and 9:00 AM yesterday* compromises her privacy more than knowing that she was there *between 8:00 AM and 8:00 PM, some day last week*.

Based on this privacy-accuracy duality, a key question is the following: what is the level of accuracy a location-based service requires to fulfill its function and hence the level of privacy loss one has to accept to benefit from that service. A related key question is then: *given the level of accuracy required by some service, how can the mobile platform ensure that only the corresponding level of privacy will be lost?* This is precisely the question we address in this paper by proposing a system-level architecture for fine-grained control over privacy, in the context of location-based services. As implied above, this architecture must be implemented at the operating system level.

C. Contributions and roadmap

As already pointed out, today's mobile platform only offer the binary choice of revealing all or nothing in terms of user locations.[3] With our architecture in contrast, the user can decide what level of location privacy she wants to retain, which then translates to the level of accuracy the location-based service will be able to offer.

The remainder of this paper is structured as follows. After presenting our system model and defining the problem we address in Section II, we describe the conceptual architecture and generic protocol proposed by this architecture to solve this problem in Section III. Section IV then focuses on the notion of privacy tree, which is at the heart of our approach, while Section V presents key aspects of the implementation of the architecture on a mobile device. Then, Section VI details how the location privacy is ensured by using this architecture. Section VII proposes a quantitative evaluation of the privacy tree and compares it with existing approaches. In doing so, we introduce a spatial and temporal probabilistic measure of the privacy loss induced whenever some location information, even partially inaccurate, is provided to a location-based service. Finally, we discuss research results relating to our approach in Section VIII and conclude the paper with future research directions in Section IX.

II. System model and problem statement

We consider a user moving on the surface of the earth with a mobile device that has the ability to locate itself, typically via the Global Positioning System (GPS)[4] or some other positioning means, e.g., WiFi positioning (WPS)[5]. The architecture of this mobile device is depicted in Figure 1: Alice, the user, sits on the top and interacts with both the underlying *trusted operating system* and some untrusted location-based service. At the bottom, the system-level location provider is continuously pushing raw location information to the location access protocol above it. The latter processes this information according to some privacy policy and feeds the resulting altered location information to the untrusted location-based service, which can potentially forward this information across the network to some equally untrusted remote server.

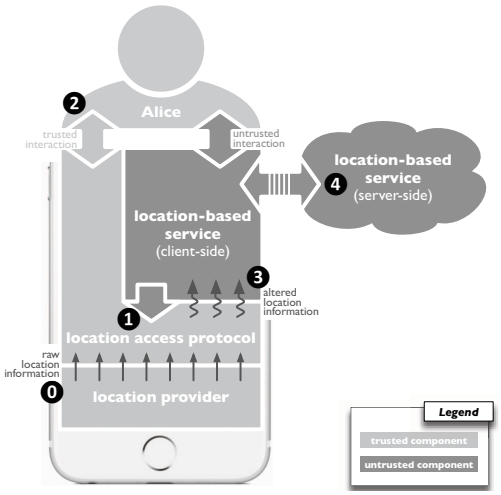

Fig. 1. Generic system architecture

We model the stream of raw locations originating from the location provider as sequence $L = \langle loc_1, loc_2, \cdots, loc_n \rangle$, where $loc_i = (\phi, \lambda, t)$ is a tuple representing an individual location. In this tuple, $\phi, \lambda \in \mathbb{R}$ represents a latitude and a longitude respectively, while $t \in \mathbb{N}$ represents the time when the location was captured. In the following, we sometimes use the notation $loc.\phi$, $loc.\lambda$ and $loc.t$ to designate specific parts of tuple loc. In addition, the duration between two consecutive locations in L does not exceed a constant Δt_{limit}, i.e., $loc_{i+1}.t - loc_i.t \leq \Delta t_{limit}$. This means that locations are captured in a regular manner.

The generic architecture presented in Figure 1 allows us to model most (if not all) relevant location privacy approaches. On iOS and Android for instance, the stream of raw locations received by the location access protocol is initially kept safe from the untrusted location-based service client (0). Then, the latter requests access to location information to the operating system (1), either at run time or at installation time. This request is forwarded to Alice in the form of a binary choice, in order to draw her attention on the potential loss of privacy (2). Assuming she accepts to grant access, the stream of raw locations is simply forwarded, unaltered, to the requesting location-based service client (3), which might then propagate it to its remote counterpart (4).

Yet finer-grained access protocols are both desirable and possible, which do not merely block or forward all raw location information. Rather, such access protocols should involve a parameterized alteration of the location information eventually provided to the untrusted location-based service and a more

[3]Most mobile platforms also allow to grant location access to certain applications only when they are running in the foreground (while others can access them even when running in background). Still, users only have the choice to either accept or refuse continuously sharing their locations.

[4]http://www.schriever.af.mil/GPS

[5]http://en.wikipedia.org/wiki/Wi-Fi_positioning_system

subtle interaction between trusted and untrusted components of the system. Providing such a location access protocol is precisely the problem we address in this paper.

III. The proposed architecture including ShareZ service

Our system-level architecture proposes a location access protocol that allows user to share their locations with location-based services, via zones of interest of *various granularities*, in order control the level of privacy loss incurred by this sharing. That is, the notion of *privacy granularity* captures the accuracy-privacy duality inherent to sharing zones of interest: coarse-grained zones offer lower accuracy but higher privacy than sharing fine-grained zones.

Figure 2 sketches the proposed architecture, the included service called ShareZ as well as its location access protocol, which are assumed to be integrated into the underlying mobile operating system, as prescribed by our generic system architecture. To illustrate the description hereafter, we assume that Alice is shopping in Paris and wants to be notified when passing by a shop with special offers. For this, she relies on a mobile application acting as the local client of some location-based shopping service.

(0) ShareZ continuously computes zones of interests of various granularities using raw locations.

(1) The location-based service client requests location information in the form of zone of interests.

(2) ShareZ asks Alice to choose the privacy granularity and the zones of interest to share with that service.

(3) ShareZ pushes the selected zones with the chosen privacy granularity to the service.

(4) The client forwards those zones to the server, which computes related contextual information, i.e., coupons associated with special offers in our example.

(5) The server sends the contextual information with its period(s) of validity related to the zones it received back to the client.

(6) The client pushes this contextual information to ShareZ, with the associated zones.

(7) ShareZ monitors incoming raw locations and checks if they match some valid contextual information according to its period(s) of validity.

(8) As soon as ShareZ receives a raw location positioned in a zone associated with some contextual information, a coupon in our example, the latter is directly displayed to Alice by ShareZ.

(9) At this point, Alice might or not decide to act based on this information, e.g., use the coupon to benefit from some special offer. If she does, the location-based service has finally the ability to precisely locate Alice, directly or indirectly. If she however decides not to use the coupon, the service will never know that Alice was in that area.

One might of course argue that knowing Alice's zones of interest is already an important breach in her privacy. Note however that Alice has the ability to prevent certain zones from being disclosed to the location-based service and to blur

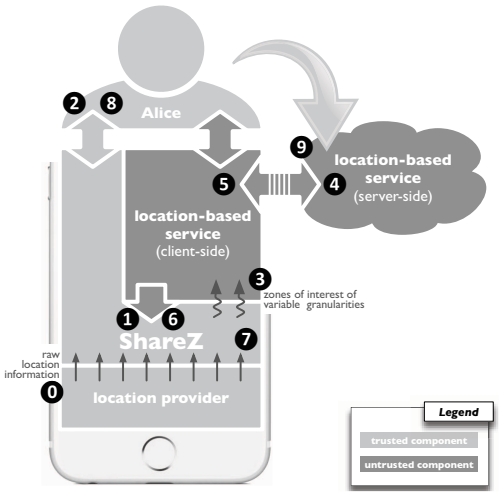

Fig. 2. The ShareZ location access protocol

the zones she accepts to share, based on the privacy granularity she chose. In addition, a rather simple strategy that ShareZ can apply to further protect Alice's privacy consists in adding one or more fake zones to those passed to the service in Step 3. Of course, such fakes zones are completely ignored during the location monitoring in Step 7. Yet the location-based service has no way to distinguish real zones of interest from fake ones (see research work in [13] and [5]).

In Step 2, it is indicated that Alice must choose a privacy granularity as well as the zones of interest that she wants to share with the location-based service. This means that Alice expresses the maximum accuracy she accepts to reveal about her location, which is also the minimum privacy. In our internal architecture, this translates into a level in the privacy tree. In terms of user interface, a map highlighting the different privacy tree levels could be presented to Alice in order to let her choose this maximum accuracy in a graphical manner. Since her choice may be not adapted to the functionality of the location-based service, she can modify her choice at any time.

The main characteristics of the proposed architecture, including its service ShareZ, and its location access protocol are summarized hereafter.

Local. The protocol minimizes the communication with the location-based service client and hence drastically reduces the risk of location information leakage to remote parties.

Flexible. ShareZ allows users to set distinct privacy preferences for different location-based services and for different contexts, e.g., one might use a certain privacy granularity while shopping and another one when jogging.

Adaptive. ShareZ constantly updates the zones of interest it manages, based on the stream of raw locations received from the underlying location provider.

Encapsulated. ShareZ is the sole recipient of raw locations

originating from the location provider and the protocol never forwards them directly to mobile applications.

IV. PRIVACY TREE

At the heart of the architecture, more specifically in the service SHAREZ, and its concept of flexible privacy granularity lies the notion of *privacy tree*. In the following, we first formally define what zones of interest are, as well as the notion of *privacy distance*. We then introduce the notion of privacy tree and its significance when it comes to support various privacy granularities.

A. Zone of interest

In order to precisely define a zone of interest, we must introduce other fundamental definitions based on the definition of a user location (see Section II) such as the definitions of cluster and cluster group.

1) Cluster: Intuitively, a cluster gathers locations sharing several common characteristics in terms of space and time. Let $\Delta d_{max} \in \mathbb{R}$ be a constant representing a distance (expressed in meters) and $\Delta t_{min} \in \mathbb{N}$ be a minimum time threshold (expressed in seconds). Moreover, we consider the two following functions: *centroid*$(\langle loc_1, loc_2, ..., loc_n \rangle)$, which computes and returns a location representing the centroid of the set of locations passed as parameters (i.e., the mean of the locations) and *distance*(loc_i, loc_j), which simply computes and returns an Euclidian distance between two locations passed as parameters.

A subset $l \subseteq L$ is a cluster iff the three following conditions are met:

$$\forall loc_i, loc_{i+1} \in l : \\ distance(centroid(loc_1, \cdots, loc_i), loc_{i+1}) \leq \Delta d_{max} \quad (1)$$

$$loc_n.t - loc_1.t \geq \Delta t_{min} \quad (2)$$

$$\nexists l' \neq l : l \subset l' \quad (3)$$

This first part of the clustering process is based on a well-known technique described in [8] and presented as *density-time cluster (DT cluster)*. A cluster is a tuple $c = (\phi, \lambda, \Delta r, l)$, where $\phi \in \mathbb{R}$ represents a latitude, $\lambda \in \mathbb{R}$ represents a longitude, $\Delta r \in \mathbb{R}$ represents its radius in meters and l the subset of locations. The centroid (ϕ, λ) of the cluster is simply the mean of all ϕ and λ of the locations contained in l. The notation $c.centroid$ is used to designate the centroid of the cluster c. Hereafter, $C = \{c_1, c_2, \cdots\}$ is the set of clusters associated with the user, based on her sequence of locations.

2) Cluster group: Intuitively, a group of clusters contains all the clusters that can be gathered iff there exists an intersection between these clusters. Two clusters $c_i, c_j \in C$ are gathered in the same group of clusters g iff we have:

$$distance(c_i.centroid, c_j.centroid) \\ - \ (c_i.\Delta r + c_j.\Delta r) \ < \ 0 \quad (4)$$

So a group of clusters(s) is defined as tuple $g = (\phi, \lambda, \Delta r, \{c_1, c_2, \cdots\})$, where $\phi \in \mathbb{R}$ represents a latitude, $\lambda \in \mathbb{R}$ represents a longitude, $\Delta r \in \mathbb{R}$ represents its radius in meters and the array of clusters from which g is formed. The centroid of a group of clusters is represented by tuple (ϕ, λ), which is simply the mean of the centroids of the clusters in g. Hereafter, $G = \{g_1, g_2, \cdots\}$ is the set of cluster groups associated with the user, based on her set of clusters C.

3) Zone of interest: Intuitively, a zone of interest is a delimited zone that is frequently visited by a user in everyday life. Let $minVisitNb \in \mathbb{N}$ be a constant representing the minimum number of visits and let $visitThreshold \in \mathbb{N}$ that is a maximum threshold of visits. Moreover, let *size*(g) be a function that computes and returns the number of clusters of the group passed as a parameter and let *meanVisitNb*(G) be a function that computes and returns the mean number of visits among all the cluster groups passed as parameters. The number, returned by *meanVisitNb*(G), frequently changes over time depending on the mobility behavior of the user if we consider that the discovery process works in a sequential manner. It is important to note that $minVisitNb$ must be equal to the value returned by *meanVisitNb*(G) until reaching the $visitThreshold$, which is the maximum number of visits to transform a cluster group into a zone of interest. A group of clusters $g \in G$ becomes a zone of interest z iff it follows Equation 5:

$$size(g) \geq minVisitNb \\ | \ \ minVisitNb = meanVisitNb(G) \\ AND \ \ minVisitNb <= visitThreshold \quad (5)$$

Formally, a zone of interest z is a tuple $z = (\phi, \lambda, \Delta r, g)$, where $\phi \in \mathbb{R}$ represents a latitude, $\lambda \in \mathbb{R}$ represents a longitude, $\Delta r \in \mathbb{R}$ represents its radius in meters and g the group of clusters. The centroid of z is represented by (ϕ, λ), which is simply the centroid computed from group g. Hereafter, $Z = \{z_1, z_2, \cdots, z_n\}$ is the set of zones of interest associated with the user, based on her set of cluster groups G.

B. Privacy distance

The *privacy distance* is simply a cursor expressing the *accuracy* of the shared location information, which perfectly illustrates the inherent tradeoff between accuracy and privacy. With SHAREZ, the user is responsible for setting this cursor. Let $\Delta d_a \in \mathbb{R}$ be a constant representing this accuracy (in meters): the greater Δd_a, the higher the location privacy offered by zones of interest respecting this privacy distance.

C. Privacy tree structure

Privacy trees follow a similar hierarchical structure as R-trees, which were introduced by Guttman to index geo-located objects (i.e., leaves of the tree), by grouping and representing them via minimum bounding rectangles at each hierarchical level [10]. In SHAREZ, the zones of interest of a user are the leaves of her privacy tree and each hierarchical level closer to the root covers a set of zones found at the lower levels, as illustrated in Figures 3 and 4. So each level of the tree represents a different privacy granularity.

Figures 3 and 4 present a privacy tree based on five zones of interest. The process of building this tree goes as follows. The first step consists in trying to gather close zones by computing the smallest distance between two zones among all of them. A group containing close zones gives birth to a new upper zone with a new centroid. If a zone of interest cannot be gathered with another one, an upper zone is created around it. For example, zone z3 has the same centroid as its upper zone z2'. Note also that the radius of various zones found at a given privacy level are not necessarily equal.

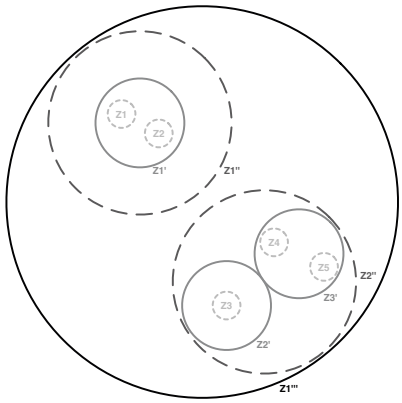

Fig. 3. Privacy tree – Spatial structure

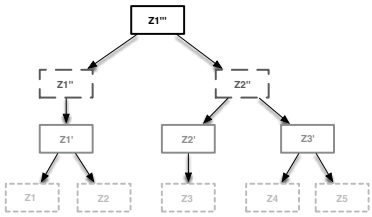

Fig. 4. Privacy tree – Tree structure

Since this process is realized sequentially in real time (i.e., each time that a new user raw location is caught), it may take time to highlight all zones of interest of the user. In this context and previously mentioned in Section II, the $minVisitNb$ and the $visitThreshold$ enable to discover zones of interest related to the user when she starts using her mobile device. Since $minVisitNb$ corresponds to the current mean number of visits, this value is obviously 0 at the beginning. Then, $minVisitNb$ evolves over time until reaching the $visitThreshold$, which is the maximum number of visits. If this mechanism was not included in the discovery process, the latter would not be able to discover zones of interest of the user from the beginning.

V. Implementation of the architecture on a mobile device

The implementation of the architecture has an impact on the two main components presented in Figure 2: trusted and untrusted components, i.e., the operating system including the service SHAREZ and the location-based service respectively. In this section, we present the key aspects of the implementation of this architecture on a mobile device from the two following points of view: operating system and location-based service.

A. On the operating system side

In order to implement our architecture at the operating system level, we need to create SHAREZ as well as the application programming interface (API) that can be used by developers to receive zones of interest of user and share contextual information. SHAREZ is able to directly interact with the location provider in order to discover the zones of interest of the user and update the related privacy tree. The location provider, which gets location via GPS, WiFi or 3G/4G, provides the stream of raw locations to SHAREZ, which is necessary to update the privacy tree. This service must also be able to store the privacy preferences set by the user for each application she is using (e.g., the privacy distance, the zones of interest already shared). In addition, this service must also memorize the contextual information, provided by the location-based service, and display it when it is appropriate according to the current location of the user and the validity period of the content. As described in Figure 2, we consider that SHAREZ is safe because it is included at the operating system layer, which is also a trusted component. In order to prevent some low level kernel attacks we could also implement a TrustZone technique to isolate critical data transactions but it is out of the scope of the paper. We assume that both the operating system including SHAREZ and the location provider are trusted components. Moreover, there is no data alteration during the data sharing between the trusted and untrusted components under the assumption that this exchange is realized in a safe manner. The data sharing implementation is also out of the scope of the paper.

B. On the location-based service side

If this architecture is implemented at the operating system level, developers of location-based services need to modify the way they obtain user locations. Their code must change in accordance with the new API exposed by the operating system library and, more specifically, by SHAREZ. This is already the case when a new version of the API of the operating system is released. Therefore, they must adapt their location-based service to the protocol of the architecture. This has an impact on the way to provide information to the user. In return, a location-based service improves its level of respect for user location privacy. One limit of our architecture is that it does not work with location-based service requiring frequent and precise location updates such as running applications and personal navigation applications. Regarding these specific cases, user should be able to share her raw locations, during a limited period of time, in order to reduce her location privacy loss.

VI. Location privacy ensured by the architecture

It is important to recall that we focus on location privacy only and not on other user privacy issues. Location privacy also aims at enhancing user privacy. As explained in the previous section, we assume that SHAREZ is implemented at the operating system level, which is considered as a safe and trusted component. Considering this, we discuss two main privacy levels, both aiming at ensuring location privacy of the user: at a low level with the privacy tree and at a high level with the architecture protocol.

A. At the level of the privacy tree

At the privacy tree level, the structure of the tree itself enables to protect the location privacy. With multiple levels of

location privacy, computed from the zones of interest of the user, the latter is able to select the most appropriate privacy distance for the location-based service. This tree is frequently updated according to the user mobility behavior. As a result, we cannot predict in advance its structure because it may evolve over time according to the new or outdated zones of interest. In addition, the privacy tree structure is only computed from the stream of user raw location, which is only accessible at the trusted component level. It is also important to indicate that we cannot prevent developers to use other way to obtain user locations such as beacons for instance. However, it is not a privacy threat because there does not exist a sufficient number of beacons used to locate users to extract a complete overview of the mobility behavior of a user.

B. At the level of the architecture protocol

At the architecture protocol level, the location privacy is mainly ensured by two crucial steps: the sharing of zones of interest (Step 0 to 3 in Figure 2) and the sharing of location-based content (Step 4 to 9 in Figure 2). Firstly, the sharing of zones of interest (from SHAREZ to the location-based service) protects the location privacy of the user, because no raw locations are sent to the location-based service. In addition, even if zones of interest of the lowest level of the privacy tree are shared, they do not correspond to precise locations. The user can also decide the zones of interest that she wants to share with the location-based service, that may reinforce the location privacy. Secondly, the sharing of content, from the location-based service to SHAREZ, aims at preserving the location privacy of the user because the content is gathered locally at the trusted component and only displayed when the current location of the user is in the zone of interest linked to this content. Consequently, the location-based service never knows the exact location of the user. However, if the user must interact with the location-based service to use a specific offer, the latter may infer the location of the user by linking the offer selected by the user with her related zone of interest.

VII. EVALUATION

This section has two different goals: (1) presenting a generic location privacy indicator according to space and time dimensions, and (2) comparing our privacy tree solution to three other location privacy preserving mechanisms: *Gaussian alteration*, *sampling* and *spatial cloaking*. It is important to indicate that we decide to assess the low privacy level, which concerns the privacy tree, because it is the heart of our architecture in terms of privacy preserving.

Firstly, we present the dataset we use for our evaluation, which is based on a real life experiment carried out by Nokia between 2009 and 2011. Secondly, we introduce the threat model as well as the privacy indicator. Further, we explain the classification of the users contained in the Nokia dataset. Next, we detail the chosen scenarios as well as the different blurring strategies we implemented. Finally, we present and discuss the obtained privacy indicator results of our experiments.

A. Nokia dataset

For this analysis, we use a dataset provided by Nokia and containing real mobility traces of users. This dataset was collected in the Lake Geneva region in Switzerland (Europe) from October 2009 to March 2011. A Nokia N95 mobile device was given to all volunteers participating to the data collection campaign. The whole process of this campaign is explained in detail in [16]. In addition, the data was of course anonymized in order to preserve user privacy, i.e., the dataset does not allow to infer the identity of the users who participated in this data collection campaign. To summarize, the dataset contains 188 users. This data comes from different sources, such as locations from GPS or GPS WLAN, phone and SMS logs, accelerometer, application usage, etc. Although this data is rich and abundant, we only use the raw location data coming from GPS or GPS-WLAN sources.

B. Threat model and privacy indicator metric

As already pointed out, we only focus in this paper on *location privacy*. In particular, we do not distinguish cases where the user identity is already known by the location-based service and the latter wants to discover her mobility pattern, from scenarios where the user identity is not known and the location-based service is trying to discover it based on her locations. Yet controlling the access to locations by location-based services can have a significant impact on privacy as a whole in both cases.

We assume a threat model including a possible adversary represented by a location-based service, which is an untrusted component as depicted in Figure 2. This adversary wants to infer personal information related to a user based on her locations received from the operating system layer. In this case, we consider that the locations shared with the location-based service are critical as well as their accuracy. Their accuracy inevitably influences the process used by the location-based service to infer sensitive user data. Consequently, we propose a metric that is a privacy indicator aiming at highlighting the degree of alteration of the user locations sent to the location-based service. This alteration takes space and time dimensions. The privacy indicator enables to compute this level of alteration and takes values between 0 and 1 included. The higher the value of the privacy indicator, the higher the protection of the user locations sent. Conversely, the lower the value of the privacy indicator, the lower the protection of the user locations sent.

Remember that in our generic architecture (Figure 1), the purpose of the location access protocol is to alter raw locations of the form (ϕ, λ, t) in order to achieve location privacy. The nature of this alteration depends on the location access protocol and can act on both the *spatial dimension* of locations, i.e., ϕ and λ, and on their *temporal dimension*, i.e., t. To model this, we introduce function F as follows:

$$F(\langle loc_1, loc_2, \cdots \rangle) \mapsto \{(z_1, \Delta t_1), \cdots, (z_n, \Delta t_n)\}$$

where $z_i = (\phi_i, \lambda_i, \Delta r_i)$ represents the *spatial alteration* and Δt_i represents the *temporal alteration* of the location information. It is important to note that these alterations are computed in parallel with the computation of the blurred locations directly sent to a possible adversary, which is the location-based service in our context. Both z_i and Δt_i are used to compute our global privacy indicator, as shown in Equation 6. That is, the global privacy indicator is the mean of the individual privacy indicators of all these alterations.

$$Privacy = \frac{1}{n} \times \sum_{i=1}^{n} Privacy(z_i, \Delta t_i) \tag{6}$$

Note that the number of $(z_i, \Delta t_i)$ tuples can significantly differ from the number of raw locations, e.g., in SHAREZ, the privacy tree is the actual result of alteration function F. The calculation of the privacy of a single $(z_i, \Delta t_i)$ tuple is described in Equation 7. This second Equation is the sum of the spatial alteration and the temporal alteration where α and β are respectively factors of them.

$$Privacy(z_i, \Delta t_i) = \frac{(\alpha \times P_{space}(z_i)) + (\beta \times P_{time}(\Delta t_i))}{(\alpha + \beta)} \tag{7}$$

The spatial alteration is presented in Equation 8, where the minimum between the area of the zone and the maximum area, called z_{max}, is divided by this maximum area. It means that, when this maximum area is reached, the user cannot lose more privacy because her privacy is fully ensured.

$$\begin{aligned} P_{space}(z_i) &= \frac{min(Area(z_i), Area(z_{max}))}{Area(z_{max})} \\ &= \frac{min(z_i.\Delta r^2, z_{max}.\Delta r^2)}{z_{max}.\Delta r^2} \end{aligned} \tag{8}$$

The time alteration is presented in Equation 9, where Δt_{max} is the time threshold beyond which the user cannot lose more privacy. The equation is therefore the division of the minimum between $z_i.\Delta t$ and Δt_{max} by Δt_{max}.

$$P_{time}(\Delta t_i) = \frac{min(\Delta t_i, \Delta t_{max})}{\Delta t_{max}} \tag{9}$$

C. User mobility behavior classification

We choose to classify users of the Nokia dataset according to their mobility behavior in terms of distance in order to see if the latter might have an influence on the results obtained for the different experiments that are explained in the next section. Among the 188 users contained in the dataset, we started by selecting all the users having GPS and GPS-WLAN data and found 184 users. Then, we decided to compute the average of all mean time difference between two successive locations of all users in order to find an appropriate threshold. The latter was presented in the system model in Section II as Δt_{limit}. After computation, we find a mean of 568 seconds. Consequently, we decided to set the value of Δt_{limit} to 600 seconds, which indicates that the locations are captured in a regular manner. We select all users having a mean time difference lower than 600 seconds and obtain a final set of 161 users. The tracking duration of all these users varies from less than 1 day to 567 days.

After exploring the radius of the largest zone of interest of each remaining user, obtained with the privacy tree algorithm, we find three groups of users as described in Figure 5. In order to illustrate our classification, Figures 6, 7 and 8 describe the mobility traces of users belonging to each group. To be more precise, User 1 travels very long distances (i.e., a radius of the largest zone of interest of approximately 180 km), User 2 travels medium distances (i.e., a radius of approximately 60 km) and User 3 travels short distances (i.e., a radius of approximately 19 km) belonging to group 3, 2 and 1 respectively. In addition, Figures 9, 10 and 11 show the three different privacy trees obtained for each user. Concerning these three users, their locations were taken during a time period of about 500 days.

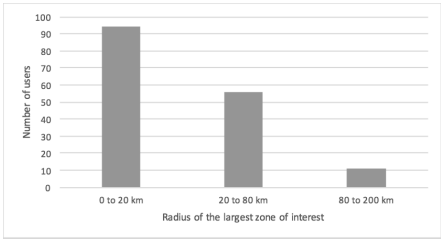

Fig. 5. User mobility behaviors

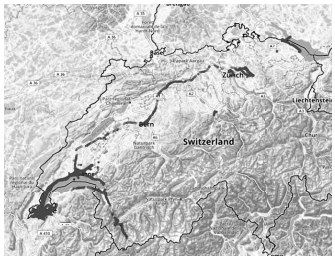

Fig. 6. Mobility traces of user 1

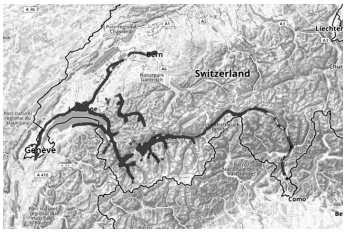

Fig. 7. Mobility traces of user 2

Fig. 8. Mobility traces of user 3

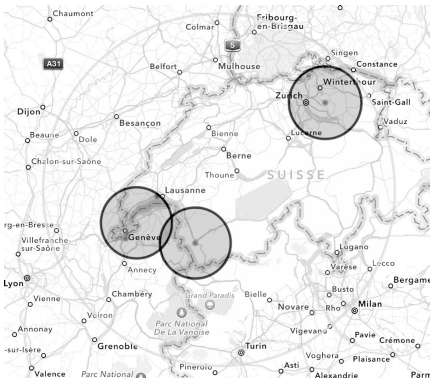

Fig. 9. Privacy tree of user 1 (7 zones of interest and 4 privacy tree levels)

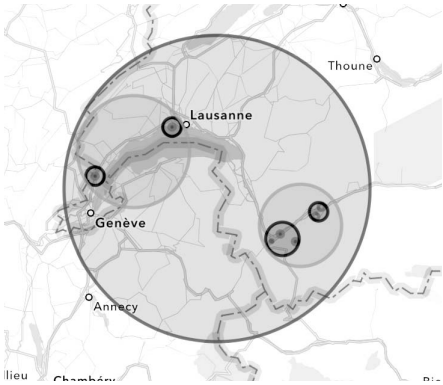

Fig. 10. Privacy tree of user 2 (10 zones of interest and 5 privacy tree levels)

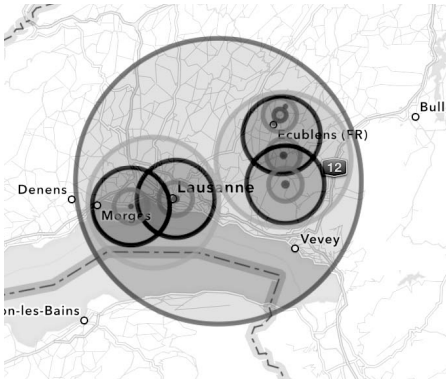

Fig. 11. Privacy tree of user 3 (7 zones of interest and 5 privacy tree levels)

D. Scenarios and blurring strategies

We consider two scenarios involving two distinct location-based services. The first scenario focuses on a social network that offers various location-based discounts (e.g., travel discounts, shopping discounts, etc.) according to their space/time context. In this context, we also consider that users select one offer per month on average. This amount is called *utility*, which is the utility perceived by the user when she uses the service. The second scenario concerns a green location-based mobile application that provides train and public transportation schedules to users according to their spatial and temporal context. The utility of this service is greater than the previous because users read the location-based information on an average of two times per day at least.

Regarding these two scenarios, we explore the impact of different blurring strategies applied to user locations. Furthermore, we also consider a case, i.e., worst case used as a reference, where all user locations are sent to the location-based service without any blurring strategy (i.e., spatial and temporal alterations are equal to 0 for all sent locations). The four selected blurring strategies are the following: Gaussian alteration, sampling, spatial cloaking and our privacy tree. Above all, it is important to note that the spatial cloaking technique is traditionally use to preserve the anonymity of users but we will use it as a blurring technique applied to one user as explained in Krumm's paper [14].

Gaussian alteration. The Gaussian alteration is a blurring strategy that consists in altering a location by adding Gaussian random noise on the latitude and the longitude of the original location. This Gaussian random noise depends on two variables: the mean which is the value where the curve is at the top (e.g., the latitude or the longitude of the original location) and the chosen standard deviation that may be expressed in meters. This strategy only has an impact on the space dimension of the location. In this context, each new location is spatially blurred and, therefore, a spatial alteration is generated. Regarding the spatial alteration, i.e., $z_i = (\phi_i, \lambda_i, \Delta r_i)$, the centroid of the zone corresponds to the blurred location and its radius is the distance between the blurred location and the original location. However, the temporal alteration, i.e., Δt_i, is 0 because there is no temporal alteration.

Sampling. The sampling strategy is a technique enabling to summarize several locations into a single location. In order to reach this goal, there exists different means of sampling. For our experiments, we decide to sample according to a time window. Consequently, we divide the user dataset into several sets of locations. Each set has a duration equal to the time window. Then, we summarize locations of each set into a single location. We create a new location by computing the mean of all the latitudes and the mean of all the longitudes. In this context, spatial and temporal alterations are generated for each blurred location, which is computed from one set of locations. Regarding the spatial alteration, the centroid of the zone equals to the blurred location and its radius is the maximum distance between the centroid of the blurred location and the farthest original location included in the set used to compute the sample. In this context, the temporal alteration corresponds to the duration between the first location and the last location of the set used to compute the sample.

Spatial cloaking. As mentioned previously, spatial cloaking can be applied to one user only (see [14]) and, in our experiments, we use it as a blurring technique even if it is originally an anonymization technique. We assume that all zones of interest of a user, as precise as possible, are sensitive regions. The spatial cloaking works as follows: all

TABLE I. OVERVIEW OF ALTERATION STRATEGIES

Strategy	Spatial Alteration	Temporal Alteration
None (worst case)	No (real location sent)	No (real timestamp sent)
Gaussian alteration	Yes	No (real timestamp sent)
Sampling	Yes	Yes (time window)
Spatial cloaking	Yes/No	Yes/No
Privacy tree	Yes	Yes (no timestamp sent)

user locations located in cloaked regions are deleted. In our implementation of the spatial cloaking technique, we simply create cloaked regions for each zone of interest of a user that must always contain the centroid of the related zone of interest. If the radius of the cloaked region is very small, the centroid of the cloaked region is obviously very close to the centroid of the zone of interest. Every time a user enters in a cloaked region, the original locations are deleted and, spatial and temporal alterations are created. Regarding the spatial alteration, the centroid of the zone and its radius are the centroid and the radius of the cloaked region respectively. The temporal alteration is the duration between the first location and the last successive location deleted for a specific cloaked region considering that the blurring process works in a sequential manner. For all remaining locations, which are not located in a cloaked region, spatial and temporal alterations are equal to 0 because they are sent without any alteration.

Privacy tree. Section IV already explains the creation of a privacy tree as well as its different characteristics. It helps to blur user locations from the zones of interest of the user and an aggregation technique to build the privacy tree. In the analysis, we compute the alterations of all user's zones of interest of each selected level of the privacy tree (e.g. most precise, intermediate and less precise level explained below). Spatial and temporal alterations are generated for each zone of interest shared. Consequently, the spatial alteration corresponds to a zone having a centroid and a radius equal to the centroid and the radius of the zone of interest previously created. The temporal alteration is 1 (i.e., fully ensured) because no temporal information is revealed with the zones of interest.

Table I summarizes the chosen blurring strategies. In next section, the parameters of all blurring strategies are indicated.

E. Evaluation method, experimental settings and results

In this section, we present the experimental settings as well as the privacy indicator results achieved for each group of users.

1) Evaluation method and experiment settings: In order to run all the following experiments explained below, we create a cocoa application implemented in Objective-C containing the implementations of the four blurring strategies as well as the calculation of the privacy indicator. For all experiments, we chose a radius of z_{max} equal to 1000 meters under the assumption that it is a sufficient radius from which users consider that their location privacy is spatially ensured. This value is appropriate in highly dense urban areas but different radius should be chosen in other contexts, especially when users frequently travel. We also chose a Δt_{max} of the duration of one day meaning that when this threshold is exceeded, users think that their location privacy is temporally ensured. Consequently, when these two thresholds are exceeded, the location privacy is entirely preserved and the privacy indicator is equal to 1. Finally, for all the experiments we consider the factors α, i.e., spatial privacy factor, and β, i.e., temporal privacy factor, are equal (i.e., 0.5 for both factors).

The parameters of each blurring strategies are chosen in order to highlight when the privacy indicator results reach a maximum value, i.e., 0.5 or 1 depending on the strategy. Concerning the Gaussian alteration strategy, we select several standard deviations from 0.001 to 0.1 corresponding to different distance noise from approximately 120 to 11880 meters. Regarding the sampling strategy, we also take several different durations for the time window: from 30 minutes to 8 months. About the spatial cloaking strategy, we choose several different distances for the radius of the cloaked regions: from 1000 to 1000000 meters. And finally, regarding the privacy tree, three levels of privacy are taken into account: the most accurate, i.e., the level of the leaves of the tree (i.e., the zones of interest of a user), the intermediate level (i.e., approximately the mean between the most accurate level and the less accurate), and the level of the root node, which is the less accurate level. During an experiment, if a user has a privacy tree with only one level, we consider that the privacy indicator of the intermediate and that of the level of the root node also correspond to the level of the leaves. In addition, if a user has a privacy tree with two levels, the privacy indicator of the intermediate level is equal to that of the level of the leaves. User clusters are discovered with Δd_{max} of 60 meters and Δt_{min} of 900 seconds (i.e., 15 minutes). To highlight the zones of interest, we consider a $visitThreshold$ of 10. Finally, it is also important to mention that the zones of interest of the privacy tree are generated with the same parameters for all the users.

2) Results: The privacy indicator results obtained for each user group according to each blurring strategy are shown in Figures 12, 13, 14 and 15. The "worst" scenario for which all locations are sent without any blurring strategy is not indicated in the diagrams because the privacy indicator is equal to 0. To begin, we can clearly see that the privacy indicator results are very close for all groups of users in Figures 12 and 13, unlike Figures 14 and 15. Regarding the spatial cloaking strategy for a radius of 100000 meters, the obtained privacy indicator results are different for each user group. This difference is explained by the fact that user groups have different travel distances. About the privacy tree, the privacy indicator results show indeed that they are different for the intermediate level. This difference can be easily explained: if a user travels long distances, the created privacy tree is automatically larger than that of a user travelling medium distances. Consequently, the alterations of the users travelling long distances are most significant than those of users moving over short distances. We observe the same if we compare the results obtained for medium and small distances of the intermediate level of the privacy tree.

In Figure 12, we can see that it is impossible to reach a privacy indicator result greater than 0.5 for a Gaussian alteration. This is explained by the fact that there is no temporal alteration when this strategy is used, unlike other blurring strategies. Figure 13 shows we can reach a suitable protection from a sampling with a time window duration of approximately 2 days. However, it seems to be rather unrealistic to send user locations every 2 days or more. Regarding the spatial cloaking strategy in Figure 14, the larger the radius of the cloaked

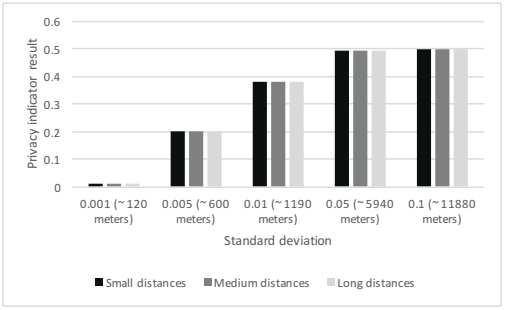

Fig. 12. Gaussian alteration - Privacy indicator results

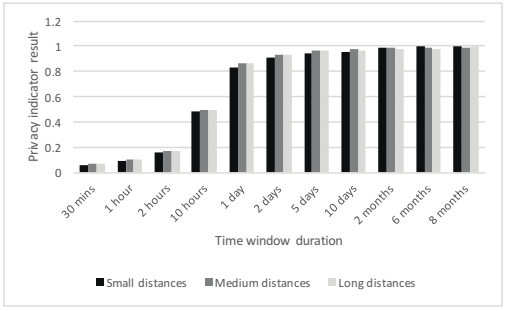

Fig. 13. Sampling - Privacy indicator results

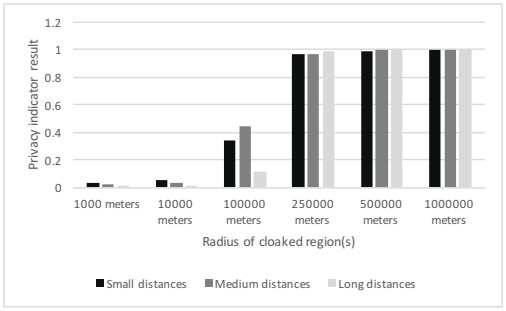

Fig. 14. Spatial cloaking - Privacy indicator results

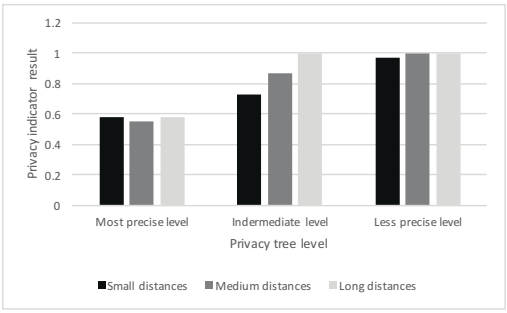

Fig. 15. Privacy tree - Privacy indicator results

regions, the higher the privacy indicator results. It also means that when 1 is reached, all user locations are contained in one large cloaked region. This strategy is interesting because it protects the sensitive regions of the user according to a specific radius distance around them. Concerning the privacy indicator results obtained for the privacy tree (see Figure 15), the most precise level (i.e., at the original zones of interest level of the users) already provides good results because they are around 0.5. These good results are explained by the non-disclosure of the temporal information. The results of the intermediate level strongly depend on the radius of the computed zones of interest of the intermediate level of the privacy tree. Then, the obtained results for the less precise level are very high, i.e., close to 1, because this level (i.e., at the root level of the privacy tree) mainly exceeds or is close to 1000 meters, which is the radius of the maximum area z_{max} of the space privacy (see Equation 8). Although the spatial cloaking strategy offers a good spatial and temporal protection, this strategy does not include a structure containing several encapsulated privacy levels. The privacy tree is a structure with various privacy levels, which makes it flexible. To conclude, the privacy tree strategy is an appropriate approach to protect location privacy of the user.

According to these results, if we resume the two scenarios described at the beginning with the two location-based mobile applications, we can now argue that the privacy tree is an appropriate blurring strategy to ensure location privacy. More specifically, we lose less privacy by only allowing a certain level of accuracy/privacy especially when the utility of the application is not high (i.e., in the case of the social network). In addition, thanks to the privacy tree we can also decide to lose more privacy for the green application, which is more useful for us (i.e., high utility), compared to the social network that is considered as less useful (i.e., low utility).

VIII. Related work

In the following, we discuss two research subjects included in our work: *location protection strategies*, and *privacy architectures and frameworks*.

A. Location protection strategies

There exists various strategies that enable to protect locations, such as mix-zones, spatial cloaking, perturbation, aggregation and many others as described in [8]. As presented in [11], [15], [8], [14], some of these strategies focus on anonymization (i.e., the impossibility of linking a user and an action) and others on location blurring or obfuscation (i.e., hiding a location). To begin with the anonymization strategies, Beresford and Stajano introduce the concept of *mix-zone* in [4] to achieve anonymization. In a mix-zone, precise locations of users are not computed. This guarantees the privacy of the user in this area and the anonymity of her moves from one zone to another zone thanks to a pseudonym mechanism. More specifically, when a user enters in a mix-zone, her pseudonym changes and the third party application does not obtain precise locations during her moves in the zone. Another technique helping to spatially anonymize a user is known as *spatial cloaking* [20]. If a user is located in a zone with at least k other

users, the entire area is returned and not her precise location. This strategy is based on the k-anonymity concept. However, although these anonymization strategies are well-known, they should not be used alone but complemented by other blurring strategies [24].

Other strategies aim at blurring locations through alteration techniques, which are used to modify the location by adding some noise to the coordinates of the location, typically Gaussian noise. In [2], authors present various spatial approaches including affine transformations, random perturbation as well as aggregation. In [19], a family of geometric data transformation methods (GDTMs) are introduced. The aggregation strategy enables to gather some close locations into a single one. For example, the clustering process can help to reach this goal as seen in [8], where several clustering algorithms are described such as density-joinable cluster, density-time cluster and time-density cluster. In our work, a density-time is used in order to find clusters at the beginning of our process. In doing so, we rely on a blurring strategy, aggregation more specifically.

To end this section, there also exist other strategies consisting in sending fake user locations. Kido et al. introduce this concept by presenting an anonymous communication technique based on the sending of false locations with the true location of the user as demonstrated in [13]. In [5], Bindschaedler and Shokri present a model to generate fake user locations to protect user location privacy. Their model is based on statistical metrics quantifying geographic and semantic aspects of the mobility of users.

B. Privacy architectures and frameworks

There basically exists two types of frameworks and architectures aiming at protecting user privacy: *decentralized ones*, where components are confined to the mobile device, and *centralized ones*, where components may be the mobile device and on some remote server [11]. In [17], Mokbel et al. introduce a framework called *Casper*, which is entirely centralized. The goal of *Casper* is to allow users to use location-based services without disclosing their location data. This framework consists of two distinct components: a location anonymizer and a privacy-aware query processor. The location anonymizer constantly receives user locations from the mobile device and blurs them using a spatial cloaking technique. It is important to note that the location anonymizer is not on the mobile device but on a remote server. Then, the blurred location is sent to the privacy-aware query processor, which enables to handle location-based requests such as *"Where is the nearest restaurant?"*. This last component is also located on a remote server. In [18], Myles et al. present *LocServ*, a middleware acting as a unifying location service. This location service relies on a remote server that collects user locations from different sources, as well as user privacy requirements, and provides answers to location requests from applications. The main goal of *LocServ* is to protect user locations gathered by various tracking systems. Users must subscribe to the location server *LocServ* and indicate their privacy preferences expressed with rules. Unlike the previous middleware, user anonymity is achieved by using multiple identifiers for one user in *LocServ*. In [9], a decentralized architecture, called *Prive*, is presented. *Prive* enables to preserve the anonymity of users using a decentralized version of k-anonymity. Although the architecture of *Prive* is decentralized, there are a certification server where users need to be registered as well as a centralized pseudonym service. The goal of *Prive* is also to answer to requests like *"Where is the nearest hospital?"*, while protecting user anonymity. In [12], Hong and Landay present *Confab*, a framework for privacy-sensitive ubiquitous computing applications. Its goal is to facilitate the development of privacy-sensitive applications by taking into account user privacy preferences. The architecture is composed of infospaces linked to users or things, e.g., an infospace for a specific user and another for a room. An infospace contains static information (e.g., user name, user address, etc...) and dynamic information extracted from different sensors (e.g., temperature). The architecture may be centralized, i.e., an infospace is managed by a remote server, or decentralized, i.e., an infospace is hosted on the user mobile device. As presented in the paper with three use cases, locations may be shared during a specific duration (i.e., time-to-live flag) and according to a pre-defined location levels (i.e., street level, room level, city level). In [7], Fawaz and Shin present *LP-Guardian*, a framework that helps to ensure location privacy. *LP-Guardian* takes into account the tracking context: background or foreground mode. For instance, when locations are caught in background, *LP-Guardian* is able to capture the location object being in the process of creation and blur it in order to protect the user. On the contrary, when the tracking is in foreground, the user may be notified in order to choose an appropriate location sharing option, e.g., hiding the place, revealing it during the application session, as well as revealing it always. In this case, it is not a binary choice as in a standard architecture, described in Figure 1, but it is also not a personalized option provided according to the behavior of the user, because the offered choices are not generated on the basis of the user locations. In addition to the blurring option, anonymization may always be enabled or disabled according to the context. Amini et al. offer a system called *Caché*, described in [1], enabling to pre-fetch and store locally location-based content (i.e., weather forecast data, bus schedule data, etc...) from various sources (e.g., web services) and for multiple areas. These areas must be pre-defined by the user and are regions of interests of the user. Beresford et al. introduce *MockDroid* in [3], which is a modified version of Android operating system enabling to change personal content shared with an application. It is an interesting way to make users aware that a large amount of personal data is shared with applications or services. Users are able to find a adapted tradeoff between privacy and service functionality they want to use. Finally the last remaining research to present is *FINE*, a framework proposed by Shao et al. in [21]. This framework enables to mainly ensure the confidentiality of location-based service data and user location privacy, and provide a fine-grained access control in the location-based service system. A cloud server is mandatory because it executes location queries and is a bridge between the location-based service provider and the users. By using a specific and adapted encryption technique, the framework ensures the confidentiality of data shared. The user must decide the range of location she wants to query but there is no information regarding how this range is chosen by the user. The context of *FINE* is not the same as our work because the location-based service provider is considered as a remote entity and not as a component of the user mobile device.

When comparing the above architectures and frameworks to SHAREZ, *LP-Guardian* is close to our solution in the sense that it also offers a personalized location privacy protection per application. However, its blurring strategy is quite different from ours, which relies on the concept of a privacy tree. Our blurring approach is constantly adapted to the user behavior in order to offer appropriate levels of location privacy protection for each specific location-based service. Furthermore, in the architectures presented above, most of them are centralized and need a remote server to work. Although *Confab* framework enables to share locations according to different location levels that seems to be pre-defined according to the context (e.g., different location levels in a building related to a work place). Our blurring solution is obviously not pre-defined because the privacy tree is updated over time. Unlike *Caché*, our solution automatically computes user's zones of interest in real-time from user raw location data. Moreover, we propose an approach per location-based service that the user wants to use, and not an approach that catches location-based content from different sources. To finish with *MockDroid*, even if this approach is per application, the privacy preferences are limited and also not automatically adapted to the user mobility behavior.

IX. CONCLUSION AND FUTURE WORK

In this paper, we proposed a novel system-level architecture providing fine-grained control over their location privacy to mobile device users. Our solution consists in a system-level architecture, which includes a service SHAREZ, relying on a location access protocol that is strictly local. Our approach also offers a flexible blurring technique that is a privacy tree. Rather than forwarding all user locations to location-based services, SHAREZ only shares her zones of interest, according to her privacy preferences. A quantitative location privacy indicator was also introduced and used in order to compare our solution to more traditional blurring approaches. The results indicate that the privacy tree, built from the user zones of interest, is a valuable structure to flexibly protect user privacy, as it enables to meet the tradeoff between privacy and accuracy on a per-service and per-user basis. Future work could be focused on a real implementation of this architecture on a real mobile device in order to analyze several performance criteria and measure the tradeoff between location-based functionality and location privacy. Since the privacy tree is already implemented in Objective-C, we could implement the architecture on Apple iOS devices by modifying the operating system layer, which is the most challenging part of this possible future work.

Acknowledgment. This research is partially funded by the Swiss National Science Foundation in the context of Project 146714.

REFERENCES

[1] S. Amini, J. Lindqvist, J. I. Hong, M. Mou, R. Raheja, J. Lin, N. M. Sadeh, and E. Toch. Caché: caching location-enhanced content to improve user privacy. *Mobile Computing and Communications Review*, 14(3):19–21, 2010.

[2] M. P. Armstrong, G. Rushton, and D. L. Zimmerman. Geographically masking health data to preserve confidentiality. *Statistics in Medicine*, vol. 18:497–525, April 1999.

[3] A. R. Beresford, A. Rice, N. Skehin, and R. Sohan. Mockdroid: Trading privacy for application functionality on smartphones. In *Proceedings of the 12th Workshop on Mobile Computing Systems and Applications*, HotMobile '11, pages 49–54, New York, NY, USA, 2011. ACM.

[4] A. R. Beresford and F. Stajano. Location privacy in pervasive computing. *IEEE Pervasive Computing*, 2(1):46–55, 2003.

[5] V. Bindschaedler and R. Shokri. Privacy through fake yet semantically real traces. *CoRR*, abs/1505.07499, 2015.

[6] Y.-A. de Montjoye, C. A. Hidalgo, M. Verleysen, and V. D. Blondel. Unique in the Crowd: The privacy bounds of human mobility. *Scientific Reports*, 3, Mar. 2013.

[7] K. Fawaz and K. G. Shin. Location privacy protection for smartphone users. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, CCS '14, pages 239–250, New York, NY, USA, 2014. ACM.

[8] S. Gambs, M.-O. Killijian, and M. Núòez del Prado Cortez. Show me how you move and i will tell you who you are. *Trans. Data Privacy*, 4(2):103–126, Aug. 2011.

[9] G. Ghinita, P. Kalnis, S. Skiadopoulus, J. Joxan, G. Ghinita, P. Kalnis, and S. Skiadopoulos. Prive: Anonymous location-based queries in distributed mobile systems. In *In WWW '07: Proceedings of the 16th International Conference on World Wide Web*, pages 371–380. ACM Press, 2007.

[10] A. Guttman. R-trees: A dynamic index structure for spatial searching. *SIGMOD Rec.*, 14(2):47–57, June 1984.

[11] A. Holzer, B. Garbinato, and F. Vessaz. Middleware for location privacy: an overview. In *Proceedings of the 2012 ACM Research in Applied Computation Symposium*, RACS '12, pages 296–303, New York, NY, USA, 2012. ACM.

[12] J. I. Hong and J. A. Landay. An architecture for privacy-sensitive ubiquitous computing. In *MobiSys'04*, pages 177–189. ACM, 2004.

[13] 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 2005, ICPS '05, Santorini, Greece, July 11-14, 2005*, pages 88–97, 2005.

[14] J. Krumm. Inference attacks on location tracks. In A. LaMarca, M. Langheinrich, and K. N. Truong, editors, *Pervasive*, volume 4480 of *Lecture Notes in Computer Science*, pages 127–143. Springer, 2007.

[15] J. Krumm. A survey of computational location privacy. *Personal Ubiquitous Comput.*, 13(6):391–399, Aug. 2009.

[16] J. K. Laurila, D. Gatica-Perez, I. Aad, B. J., O. Bornet, T.-M.-T. Do, O. Dousse, J. Eberle, and M. Miettinen. The mobile data challenge: Big data for mobile computing research. In *Pervasive Computing*, 2012.

[17] M. F. Mokbel, C.-Y. Chow, and W. G. Aref. The new casper: query processing for location services without compromising privacy. In *VLDB '06*, pages 763–774, 2006.

[18] G. Myles, A. Friday, and N. Davies. Preserving privacy in environments with location-based applications. *IEEE Pervasive Computing*, 2(1):56–64, 2003.

[19] S. R. M. Oliveira and O. R. Zaïane. Privacy preserving clustering by data transformation. In A. H. F. Laender, editor, *SBBD*, pages 304–318. UFAM, 2003.

[20] N. Poolsappasit and I. Ray. Towards achieving personalized privacy for location-based services. *Trans. Data Privacy*, 2(1):77–99, Apr. 2009.

[21] J. Shao, R. Lu, and X. Lin. Fine: A fine-grained privacy-preserving location-based service framework for mobile devices. In *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*, pages 244–252, April 2014.

[22] C. Song, Z. Qu, N. Blumm, and A.-L. Barabási. Limits of Predictability in Human Mobility. *Science*, 327(5968):1018–1021, Feb. 2010.

[23] Y. Wang, Y. Chen, F. Ye, J. Yang, and H. Liu. Towards Understanding the Advertiser's Perspective of Smartphone User Privacy. In *Distributed Computing Systems (ICDCS), 2015 IEEE 35th International Conference on*, pages 288–297. IEEE, June 2015.

[24] H. Zang and J. C. Bolot. Anonymization of Location Data Does Not Work: A Large-Scale Measurement Study. In *Proc. of ACM Mobicom*, Sept. 2011.