SSPA-LBS: Scalable And Social-Friendly Privacy-Aware Location-Based Services

Changsha Ma, Zhisheng Yan, *Member, IEEE*, and Chang Wen Chen, *Fellow, IEEE*

***Abstract*—Privacy-Aware Location-Based Service (PA-LBS) preserves LBS users' privacy but undesirably sacrifices service quality. In order to balance the two factors with satisfactory user experience, existing frameworks are faced with two barriers, i.e., scalability and social-friendliness. First, existing schemes do not enable LBS users to flexibly scale their privacy level on service provision. Such a lack of scalability easily results in either unacceptable service quality degradation or insufficient privacy protection and fails to meet the dynamic user requirements. Second, existing schemes handle privacy protection by merely considering the trust relationship between users and servers but ignore the complex trust relationships among users. As a result, users cannot preserve privacy in location-based social services that involve user-to-user interactions. In this paper, we present the first scalable and social-friendly PA-LBS (SSPA-LBS) system. In particular, we propose a novel camouflage algorithm with formal privacy guarantee that enables LBS users to expose their location information by scaling two privacy related factors, i.e., camouflage range and place type. Furthermore, we apply the Scalable Ciphertext Policy Attribute-Based Encryption (SCP-ABE) algorithm to enable LBS users to effectively control the access from other users to their location information. Moreover, we also demonstrated the operational efficiency of the proposed system through successful implementations on Android devices.**

***Index Terms*—Social media sharing, privacy, access control, SCP-ABE, scalable media format**

I. Introduction

Location Based Services (LBSs) have been shared by a large portion of mobile applications nowadays. By reporting real-time locations, LBS users can actively request information from LBS servers. These applications include navigation, points of interests (POIs) searching, public transportation schedule checking, etc. Alternatively, LBS servers can also actively push information to LBS users by accessing their real-time locations. A representative application of this case is nearby friend notification, which alerts LBS users when their friends are nearby. By accumulating the historical location records, LBS servers can also provide users with personalized services such as friend recommendation, POI recommendation, and personalized advertising [1], [2].

The prevalence of LBSs has resulted in a dramatic increase in the transmission of personal geo-data that usually embeds a high precision location information initially sensed by the positioning system [3], [4]. The lack of secure and reliable access control on these personal precise location information may result in serious privacy issues. One example is the real-time tracking, where malicious people track a LBS user by accessing his or her real-time precise location information. LBS users may also suffer off-line attacks through location analysis such as work/home place disclosure if the historical precise geo-data is leaked [5].

To prevent these privacy issues, it is imperative to develop privacy-aware LBS (PA-LBS) and enable users to securely control the exposure of their location information. However, PA-LBS has to sacrifice the service quality such as result accuracy and computation/communication cost, or even depreciate the service provision to achieve privacy protection. Despite the existing efforts to balance service availability/quality guarantee and privacy protection [6], [7], there are still two barriers preventing users from satisfactory user experience. First, LBS users are not provided with scalability when selecting their privacy levels in different services. This easily results in either unacceptable service quality degradation or insufficient privacy protection and fails to meet the dynamic user requirements. Second, existing schemes are not social-friendly. They merely handle the user privacy protection from untrusted servers, ignoring the complex trust relationships among users. As a result, users cannot preserve privacy in location-based social services involving user-to-user interactions. Handling privacy-aware user-to-user interactions is a challenge since trust relationships among users are not only varying in terms of social context but also dynamic in both time and space domain [8].

In this paper, we tackle these two technology barriers and propose a **S**calable and **S**ocial-friendly **PA-LBS** (SSPA-LBS) system. In particular, we propose a novel camouflage algorithm supporting scalability with formal privacy guarantee. LBS users can select their privacy level by designating a set of camouflage points, and scale up (scale down) by increasing (decreasing) the number of camouflage points. In addition, LBS users have two factors for scaling the privacy level, i.e., location range and location type. In this way, a LBS user can select the privacy guarantee and service quality in a more flexible way to adopt various LBS applications. Furthermore, we apply the Scalable Ciphertext Policy Attribute-Based Encryption (SCP-ABE) algorithm [9] to enable LBS users to effectively control the access from other users to their location information, providing friendliness.

We summarize the contributions of this research as below:

- We develop the first SSPA-LBS system that meets both scalable privacy protection requirement and social interaction requirement of the LBS users (Section 3).

This work is supported by NSF Grant ECCS1405594. Zhisheng Yan is the corresponding author.

C. Ma and C. W. Chen are with the Department of Computer Science and Engineering, University of Buffalo, Buffalo, NY, 14260 USA (e-mail: changsha@buffalo.edu, chencw@buffalo.edu).

Z. Yan is with the Department of Computer Science, Georgia State University, Atlanta, GA, 30303 USA (email: zyan@gsu.edu).

- We propose a novel camouflage algorithm that enables LBS users to disclose their location information by scaling two privacy related factors with formal privacy guarantee (Section 4).
- We propose a reliable location access control mechanism to effectively manage the exposure of the location information among LBS users (Section 5).

II. Related Works

A. Homomorphic Encryption

Homomorphic encryption (HE) enables specific computations in the cipher-text domain [10]. In HE-based PA-LBS framework, LBS users report the HE-encrypted locations for privacy protection. Unfortunately, HE-based mechanism supports very limited applications since only some specific operations can be conducted in the cipher-text domain. Furthermore, LBS users have to perform heavy computations in order to request services, which is especially problematic for resource-constraint mobile devices. Moreover, an excessive amount of all service related data need to be pre-encrypted and stored in the LBS server incurring significant overhead to update information in cipher-text domain. Since HE-based mechanism is intrinsically non-scalable, it does not enable LBS users to relieve the cost by reducing the privacy levels.

B. k-Anonymity

k-anonymity protects the identities of the LBS users by allowing a user to send requests from a region only when at least $k-1$ other users of the service are present in that region [14], [15]. The LBS server provides service based on the reported region instead of the precise location of the user. This implies that the LBS user is indistinguishable from at least $k-1$ other users by sacrificing the result accuracy.

By setting a larger k for better privacy guarantee or smaller k for more precise results, k-anonymity seems to be able to provide scalability. However, the selection of k highly depends on the nearby $k-1$ users. If there are no users within a close enough range, anonymization cannot be performed and hence no privacy-aware services can be provided. Additionally, k-anonymity depreciates all services that need identities since it protects user identities from LBS servers. Due to requirement of continuous report of exact locations to trusted third party (TTP), k-anonymity may result in serious privacy issues when the TTP is compromised [16].

C. k-Camouflaging

In k-camouflaging based PA-LBS, $k-1$ camouflaged locations are generated and reported with the real location to request service [17], [18]. The LBS server has to respond to each of the k service queries and cannot distinguish the true user location from $k-1$ camouflaged ones. LBS users can also obtain the accurate results from the server.

Such a mechanism supports scalable privacy guarantee in a more practical way compared to k-anonymity. However, a random selection of camouflage points with no sophisticated privacy strategy results in vulnerability in location analysis attacks [19]. For example, if a user sends multiple requests from one location, the probability for an attacker to guess it as the real location can be very high by observing multiple user requests. Additionally, k-camouflaging results in a waste of user side communication cost in both receiving and requesting services, since multiple requests are sent and multiple responses are received in service provision.

D. Single-Camouflaging

The idea of single-camouflaging is to generate a camouflage point within a camouflage range, to replace the real location of a LBS user [20]. The LBS server then provides services based on the obfuscated user locations. To relieve possible service quality degradation, the LBS server can enlarge the serving area [8], [21]. The LBS user will ultimately compute and recover the expected results from the returned results.

It is straightforward that a larger camouflage range results in more resource consumption and hence worse service quality. However, how to measure the achieved privacy guarantee is not trivial. In [22], the ratio between the optimal range in terms of service quality and the selected camouflage range is defined to measure the achieve privacy and result accuracy. However, this measurement is not enough since user privacy in LBS does not simply depend on the camouflage range. In [23], the entropy of region c is utilized to measure the achieved privacy. Practically, a LBS user will have to rely on a TTP to get $E(c)$ since it is dynamic resulting from user activities. This may cause privacy issues when the TTP is compromised, as what is faced by k-anonymity. Geo-indistinguishability has also been defined [21] and optimized [24] to achieve the privacy preservation. However, as we will discuss in Section IV A, these camouflage algorithms still suffer from several severe vulnerability issues.

E. Summary

We can see that each existing mechanism has its unique barrier in balancing privacy protection and service quality. In particular, all of them have not addressed the scalable privacy guarantee issue properly. Moreover, few existing works have dealt with the privacy-aware user-to-user interactions. The lack of scalability and social-friendliness undesirably limits the adoption of LBS applications, especially for the social applications that are receiving increasing popularity nowadays.

In this paper, we build the SSPA-LBS system to: 1) provide scalability from two key perspectives to improve the flexibility in balancing between privacy guarantee and service quality; 2) provide social friendliness to support LBS applications that involve user-to-user interactions.

III. System Overview

We now overview the SSPA-LBS and its system models.

A. Location Assumption

We first introduce the assumptions about the camouflage location information in SSPA-LBS. We assume that the possible locations of a LBS user form a set of POIs $\mathcal{X}$. A LBS user located at $A \in \mathcal{X}$ reports the camouflage location information to request services. The camouflage location information in SSPA-LBS is composed of camouflage point A', camouflage range R, and place type T (e.g. restaurant). Disclosing

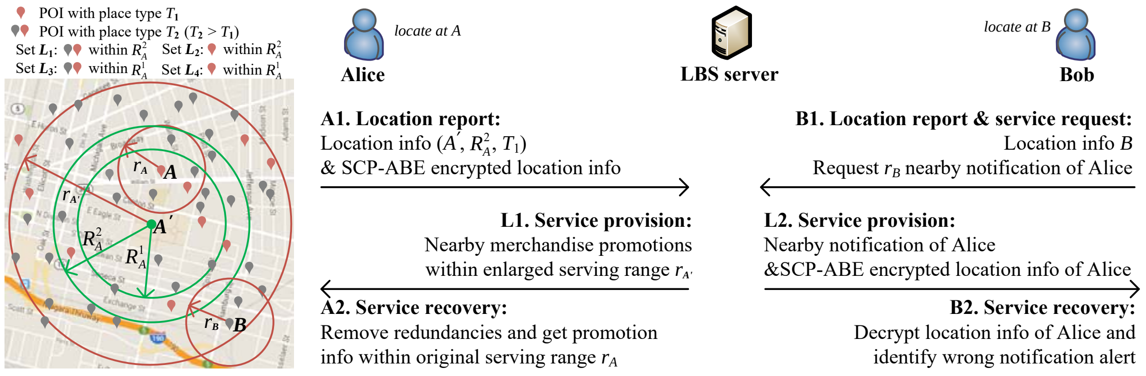
POI with place type T_1
POI with place type T_2 ($T_2 > T_1$)
Set L_1: within R_A^2
Set L_2: within R_A^2
Set L_3: within R_A^1
Set L_4: within R_A^1
r_A
A
$r_{A'}$
A'
R_A^2
R_A^1
r_B
B
locate at A
Alice
LBS server
locate at B
Bob
A1. Location report:
Location info (A', R_A^2, T_1)
& SCP-ABE encrypted location info
B1. Location report & service request:
Location info B
Request r_B nearby notification of Alice
L1. Service provision:
Nearby merchandise promotions
within enlarged serving range $r_{A'}$
L2. Service provision:
Nearby notification of Alice
&SCP-ABE encrypted location info of Alice
A2. Service recovery:
Remove redundancies and get promotion
info within original serving range r_A
B2. Service recovery:
Decrypt location info of Alice and
identify wrong notification alert

Fig. 1. Illustration of typical message flows in SSPA-LBS system

(A', R, T) results in user location observations $\mathcal{L} \subseteq \mathcal{X}$, which contains POIs of all type T located within R centered at A'. As camouflage range and place type are both intrinsically scalable, the user can adjust $\mathcal{L}$ by scaling R and T and disclose location information with different precisions.

It is worth to notice that existing schemes merely consider camouflage point and camouflage range and ignore the place type as part of location information when protecting user privacy. However, place type significantly impacts both service provision and privacy protection. First, the context of place type is critical to enable personalized services. Furthermore, place types can disclose information with various sensitivities, in different scenarios and to different people.

Moreover, a specific (A', R_i, T_i) determines an unique set $\mathcal{L}_i$ of observations on user locations. In the example shown in Fig. 1, the LBS user Alice sets four privacy levels by choosing two camouflage ranges, i.e. R_A^1 and R_A^2 ($R_A^1 < R_A^2$), and two place types, i.e. T_1 and T_2 (T_1 is more precise than T_2, noted as $T_1 < T_2$). One example of T_1 and T_2 could be *fast food* and *restaurant* where the first one belongs to the second one and is more precise. Disclosing (A', R_A^2, T_2), (A', R_A^2, T_1), (A', R_A^1, T_2), and (A', R_A^1, T_1) result in observations $\mathcal{L}_1$, $\mathcal{L}_2$, $\mathcal{L}_3$, $\mathcal{L}_4$, respectively, where $\mathcal{L}_4$ discloses the most precise location information and results in the lowest privacy level.

B. Adversary Model

We assume that the adversary aims at locating the LBS user's current real location. He/she will pick one point from the observation set $\mathcal{L}$. We also assume that the adversary has the user's historical access profile and is aware of the prior probability distribution of points in $\mathcal{L}$. In the case of Fig. 1, such prior knowledge of the adversaries could be expressed as $P_k(i)$ ($\sum_{i \in \mathcal{L}_k} P_k(i) = 1; k = 1, 2, 3, 4$) where k indicates privacy levels and i indicates POIs in the camouflage range.

C. Trust Model

We assume that LBS servers are semi-trusted by LBS users. A LBS server is trusted to provide correct and complete information for LBS users based on the reported location information, and is trusted to conform to the mechanisms in the SSPA-LBS model. However, the LBS server is not trusted to keep the location information of LBS users private from other parties. Moreover, the trust relationship between a LBS user and the LBS server vary with applications. Additionally, we assume that LBS users do not trust each other and that the trust relationship varies among users and across applications.

D. Message Flow Mechanism

We introduce the typical message flows of SSPA-LBS and illustrate them in Fig. 1.

1) User-to-Server Message Flow: User-to-server message flow preserves the privacy of a LBS user who is provided services by the LBS server. It contains three typical steps, i.e. privacy level selection by LBS user, service provision by LBS server, and service recovery by LBS user.

To begin with, a LBS user can select different privacy levels to the LBS server in different scenarios. For example, the user can expose the real location to the server if the current location is not sensitive at all, and report camouflage location information in order to preserve privacy. Note that the real location can be treated as a special case of camouflage location where camouflage point is the real location, camouflage range is zero, and place type is the precise place type. In the case of Fig. 1, Alice chooses to expose (A', R_A^2, T_1) to the LBS server. For the server, Alice could be at any type T_1 POIs (e.g. fast food restaurants) within R_A^2 from A', i.e. points in $\mathcal{L}_2$.

The LBS server then provides services based on the camouflage location information. In order to include the complete service information in the service response, the LBS server may need to enlarge the serving area. In Fig. 1, the LBS server notifies Alice the nearby merchandise promotions since the historical records shows Alice often goes for shopping after visiting a fast food restaurant. Suppose the usual serving area is within r_A centered at A, the LBS server then has to enlarge it as within $r_{A'} = R_A^2 + r_A$ centered at A' in order to fully cover Alice's desired serving area.

Upon receiving the response, the LBS user will have to remove redundant service information resulted from enlarged serving area. In the case shown in Fig. 1, Alice remove merchandise promotions out of the range r_A from A.

2) User-to-User Message Flow: User-to-user message flow handles privacy protection of one LBS user from other users, in the case that the service provision for other users requires

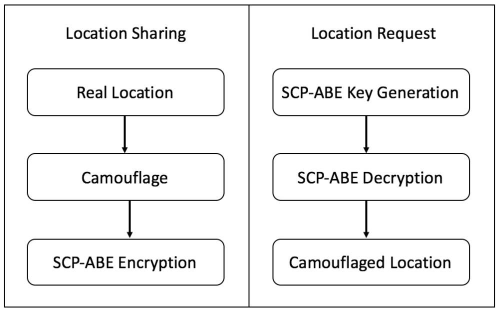

Fig. 2. Functional modules of location sharing and request

the location information of that LBS user. The functional modules in this user-to-user mode is illustrated in Fig.2.

First, a LBS user in SSPA-LBS system designates different privacy levels for different users according to their trust relationships, and shares camouflage location information with various precisions to other users accordingly. We further let LBS users make access policies for these camouflage location information based on attributes such as social relationships and relative distance, encrypt them using the proposed SCP-ABE algorithm, report the encrypted information to the LBS server, and rely on the server to distribute their location information. The reason of selecting attributes-based access control instead of identity-based access control is to adapt to the dynamic trust relationships among users.

Upon receiving request from a LBS user, i.e., Bob requesting nearby notification of Alice within a radius of r_B as in Fig. 1, the LBS server provides services using the camouflage location information shared between it and Alice, i.e., (A', R_A^2, T_1). The server then notifies Bob that Alice is nearby whenever A' is no more $r_B + R_A^2$ far away from B. The utilization of camouflage location information in service provision result in redundancies, which are wrong alerts in this case. To calibrate the service results, the server sends the SCP-ABE encrypted location information of Alice to Bob along with the notification.

Bob removes wrong alerts by generating a specific SCP-ABE key and then decrypting the location information of Alice. If the attributes of Bob enable him to obtain (A', R_A^1, T_1), (A', R_A^2, T_1), or (A', R_A^1, T_2), Bob can identify the alert as a wrong one. If Bob can only access (A', R_A^2, T_2), he will not be able to correctly identify the wrong alert.

Note that the procedure described in Fig. 1 can be directly scaled to the cases with multiple users (either multiple A or multiple B) by sending multiple location reports A1 or B1 to the server. Furthermore, the server is prevented from learning the location of Alice shared with LBS users since the SCP-ABE algorithm guarantees that only the users with designated attributes can decrypt the corresponding location information.

IV. Camouflage Algorithm

In this section, we introduce the scalable camouflage algorithm that provides LBS users with formal privacy guarantee.

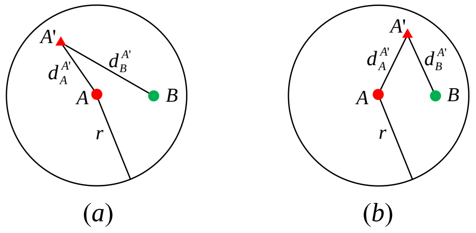

Fig. 3. Two possible locations of the camouflage point A' generated by Miguel Andrés's camouflage algorithm

A. Preliminary

1) Geo-Indistinguishability: Based on the differential privacy theory [28], geo-indistinguishability [21] has been defined to provide privacy guarantee for LBS users. Specifically, if Alice generates a camouflage point A' within radius r with a probability $P(A'|A)$ and the other user Bob at an arbitrary location B generates the same camouflage point A' with a probability $P(A'|B)$, then A' reveals little information about whether Alice's real location is A or B. Based on such an assumption, ϵ-geo-indistinguishability is defined in (1) and, correspondingly, the user enjoys ϵr-privacy within r.

$$\frac{P(A'|A)}{P(A'|B)} \leq e^{\epsilon r} \; (\forall r > 0, \forall A, B : d_A^B \leq r) \tag{1}$$

2) Typical Camouflage Algorithms: A typical camouflage algorithms has been proposed based on ϵ-geo-indistinguishability [21]. In the algorithm, a camouflage point $x \in \mathbb{R}^2$ is generated for a user who locates at $x_0 \in \mathbb{R}^2$ utilizing the function *planar laplacian centered in* x_0 as in (2). Specifically, $\epsilon^2/2\pi$ is a normalization factor, and $d_x^{x_0}$ is the distance between the center x_0 and the generated point x.

$$D_\epsilon(x_0)(x) = \frac{\epsilon^2}{2\pi} e^{-\epsilon d_x^{x_0}} \tag{2}$$

Now let us consider the examples indicated in Fig. 3, where A is the real location of the user who expects ϵr-privacy within r, and B is another point in $\mathcal{X}$ within this area. Note that the distance between A and B, i.e. d_A^B, will be no more than r. Suppose the reported location is A', which is $d_A^{A'}$ away from A, and $d_B^{A'}$ away from B. According to (2), by observing A', the probability that A generates A' should be $P(A'|A) = \frac{\epsilon^2}{2\pi} e^{-\epsilon d_A^{A'}}$ and the probability that B generates A' should be $P(A'|B) = \frac{\epsilon^2}{2\pi} e^{-\epsilon d_B^{A'}}$. Therefore, we have the following

$$\frac{P(A'|A)}{P(A'|B)} = \frac{e^{-\epsilon d_A^{A'}}}{e^{-\epsilon d_B^{A'}}} = e^{\epsilon (d_B^{A'} - d_A^{A'})} \leq e^{\epsilon d_A^B} \leq e^{\epsilon r} \tag{3}$$

According to (1), this implies that the user who is located at A enjoys ϵr-privacy within r.

3) Vulnerability Analysis: Despite the wide acceptance, we observe that this typical type of camouflage algorithm introduces three vulnerabilities when applied into the proposed scenario where the distance between the real location A and the camouflaged point A' needs to be less than a given r.

First, these algorithms cannot guarantee geo-indistinguishability that is defined in (1). As we have discussed, generating a camouflage point following planar laplacian distribution provides $\epsilon r-$privacy for users by satisfying (3). It is easy to show that both of the two cases in Fig. 3 satisfy (3) according to triangle inequality. However, since $d_B^{A'} > r$ while a generated camouflage point is no more r away from the real location, $P(A'|B)$ is actually 0. As a result, $P(A'|A)/P(A'|B) = \infty$, and hence no ϵ exists to provide users with ϵ-geo-indistinguishability as defined in (1).

Second, the algorithm is not robust to resist location analysis attack. We define success rate of resisting attack Suc as the ratio between the probability of guessing other locations and the probability of guessing the real location. Additionally, we define $dis(A, i)$ as $P(A'|A)/P(A'|i)$ representing the rate of distinguishing the real location A from an arbitrary possible locations i ($i \in \mathcal{X} \setminus A$) by generating A' as the camouflage point, where $\mathcal{X}$ is the possible location of A. Furthermore, we assume that the attacker who performs location analysis has the prior distribution of $\mathcal{X}$ and that $P(A) + \sum_{i\in\mathcal{X}\setminus A} P(i) = 1$. When the camouflage point A' is observed, Suc can be represented as in (4). $i \in \mathcal{X} \setminus A$

$$\begin{aligned}(Suc)_{A'\ observed} &= \frac{\sum_{i\in\mathcal{X}\setminus A} P(i|A')}{P(A|A')} \\ &= \sum_{i\in\mathcal{X}\setminus A} \frac{P(i|A')}{P(A|A')} \\ &= \sum_{i\in\mathcal{X}\setminus A} \frac{P(A'|i)P(i)}{P(A'|A)P(A)} \\ &= \frac{1}{P(A)} \sum_{i\in\mathcal{X}\setminus A} dis(A,i)^{-1} P(i)\end{aligned} \tag{4}$$

From (4) we can see that distinguishability and prior distribution of possible points should be jointly considered to guarantee Suc. In other words, the attacker's side knowledge should be considered for optimal privacy protection. However, existing camouflage algorithm merely provides an upper bound $e^{\epsilon r}$ for each $dis(A, i)$ and will not be able to address the attacker's prior knowledge.

Third, the algorithms do not allow users to disclose their location information with arbitrary precision, i.e., the algorithms are lack of scalability. Let us consider the example shown in Fig. 4. Suppose that Alice who locates at A wants to preserve ϵr_1-privacy for Bob and preserve ϵr_2-privacy for others. Using existing camouflage algorithm, Alice will have to generate A'_1 within r_1 from A and disclose it to Bob, and A'_2 within r_2 from A and disclose it to others. Correspondingly, Alice should be located within r_1 from A'_1 by Bob, and within r_2 from A'_2 by others. However, if Bob additionally request less precise information (A'_2), which is reasonable in access control, he will be able to locate Alice in the shaded area, which is more precise than what Alice intended to disclose.

B. The Proposed Scheme

In this section, we propose a novel camouflage algorithm for the proposed framework. First, the algorithm guarantees privacy for users based on the calibrated definition of ϵ-geo-indistinguishability. Second, the algorithm leverages the prior distribution of points in $\mathcal{X}$ to resist location analysis attack. Third, the algorithm enables various privacy guarantees.

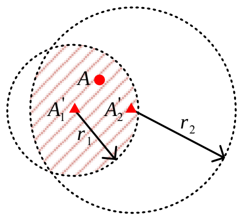

Fig. 4. Two camouflage points with ϵr_1-privacy and ϵr_2-privacy generated by Miguel Andrés's camouflage algorithm

1) Calibrated Definition of Geo-Indistinguishability: In order to provide privacy guarantee within a certain region, which is also necessary to facilitate service provision, truncation has to be utilized with the laplacian mechanism, i.e. generating camouflage point (A') within a specific range (r) from the real location (A) instead of everywhere in the plane. However, truncation also results in lack of ϵ-geo-indistinguishability guarantee defined as in (1), since (5) cannot be satisfied under the assumption that $\mathcal{L}$ are POIs within r from A.

$$d_i^{A'} \le r \quad (\forall i \in \mathcal{L}) \tag{5}$$

Such a contradiction is actually resulted from the inappropriate assumption of $\mathcal{L}$. In practice, it is infeasible for users to choose $\mathcal{L}$. Instead, $\mathcal{L}$ should be determined by attackers after the observation of the camouflage point, which is composed of POIs within r from A'. By calibrating the definition of geo-indistinguishability as in (6), we can see that (5) is always satisfied. As a result, the laplacian mechanism with truncation proposed by Miguel Andrés et al can guarantee ϵ-geo-indistinguishability for users. The proposed camouflage algorithm leverages Miguel Andrés's mechanism and guarantees privacy based on the calibrated definition of ϵ-geo-indistinguishability shown in (6).

$$\frac{P(A'|A)}{P(A'|B)} \le e^{\epsilon r} \ (\forall r > 0, \forall A, B : d_A^{A'} \le r, d_B^{A'} \le r) \tag{6}$$

2) Setup: In this step, the user (Alice) sets the expected privacy guarantees, i.e. how precise the location information is to disclose, for service request in different scenarios. In particular, the privacy guarantee can be scalable in terms of both camouflage range and place type. A higher privacy level corresponds to a relatively larger camouflage range and less precise context of place type, and vice versa. For ease of presentation, we use the example shown in Fig. 1 to present the algorithm. Specifically, Alice sets four privacy levels by selecting camouflage ranges R_A^1, R_A^2 ($R_A^1 < R_A^2$), and place type context T_1, T_2 ($T_1 < T_2$).

3) Camouflage Point Generation: This step generates a single camouflage point satisfying all levels of the expected privacy guarantee of Alice as in Algorithm 1.

First, Alice gathers the location information of points in $\mathcal{X}$. This can be simply achieved through POI retrieval, which is supported by many open APIs such as *Google Places* [30]. Alice even needs not to disclose her location information but simply designates a retrieval area to get the location information of POIs through the open APIs [31]. Additionally, Alice needs not to retrieve the whole $\mathcal{X}$, which is infeasible and unnecessary, but instead retrieves a subset of $\mathcal{X}$. Generally, Alice should guarantee that the final location observation sets should be all included in the retrieved set. In the case shown in Fig. 1, Alice only needs to retrieve all place type T_2 POIs within $R_A^2 + R_A^1$ from A, since the observation sets $\mathcal{L}_k (k = 1, 2, 3, 4)$ should be composed of points within R_A^2 from A' and A' is no more than R_A^1 from A. Note that this retrieval area is less precise than the location information disclosed by any privacy levels, and hence does not prevent us to preserve user privacy as expected.

Then, Alice selects the indistinguishability parameter ϵ, and generates n points as the candidates of camouflage point within R_A^1 from A. More specifically, Alice randomly chooses $\theta_1, \theta_2, ..., \theta_n$ uniformly from $[0, 2\pi)$, and $z_1, z_2, ..., z_n$ uniformly from [0,1), and computes $d_j = C_\epsilon^{-1}(z_j)(j = 1, 2, ..., n)$, where $C_\epsilon(z_j) = 1 - (1 + \epsilon z_j)e^{-\epsilon z_j}$. (d_j, θ_j) is then mapped to the closest point A'_j on the grid of discrete cartesian coordinates within R_A^1.

Furthermore, Alice constructs $\mathcal{C}_j \subset \mathcal{X}$ that contains type T_2 of POIs within R_A^2 from candidate A'_j. Therefore, $\mathcal{C}_j$ contains all intermediate observations. Alice then designates the prior distribution P_j for all the observations. Finally, Alice computes the value of the objective function utilizing the designated prior distribution as in (7). The candidate that results in the maximize value of (7) will be selected as the output camouflage point.

$$\textit{maximize} \sum_{i \in \mathcal{C}_j \setminus A} e^{\epsilon (d_A^{A'_j} - d_i^{A'_j})} P(i) \tag{7}$$

Note that the above objective function only guarantees that the camouflage point is optimal for the highest privacy level, since $\mathcal{C}_j = \mathcal{L}_1$. The gap between the optimal value and the value generated by the algorithm for other privacy levels can be presented as in (8), where P_{ji} and P_j are the priors of $\mathcal{L}_i$ $(i = 2, 3, 4)$ when the points outside of $\mathcal{L}_i$ are set as zero priors and non-zero priors respectively. Suppose that there are two points x_1 and x_2 in $\mathcal{L}_3$ and four points x_1, x_2, x_3 and x_4 in $\mathcal{L}_1$. If priors of points in $\mathcal{L}_3$ are $(\frac{1}{2}, \frac{1}{2})$, while priors of points in $\mathcal{L}_1$ are $(\frac{1}{3}, \frac{1}{3}, \frac{1}{6}, \frac{1}{6})$. As a result, the optimal objective for $\mathcal{L}_3$ is degraded with a factor $\frac{1}{2} - \frac{1}{3} = \frac{1}{6}$.

$$\sum_{k \in \mathcal{L}_i \setminus A} e^{\epsilon (d_A^{A'_j} - d_k^{A'_j})} (P_{ji}(k) - P_j(k)) \tag{8}$$

Such a gap is the sacrifice for scalability. In order to balance the gaps among different privacy levels, Alice can further assign weights for priors of points in different observation set. For example, if Alice wants to reduce the gap of optimal objective for the lowest privacy level, she can assign higher weights for priors in $\mathcal{L}_1$. $P(i)|_{i \in \mathcal{L}_4} > P(i)|_{i \in \mathcal{L}_2 \setminus \mathcal{L}_4} > P(i)|_{i \in \mathcal{L}_1 \setminus \mathcal{L}_3 \setminus \mathcal{L}_2}$. The objective function with weight assignment is represented as in (9).

$$\begin{aligned} \textit{maximize} & \sum_{i \in \mathcal{C}_j \setminus A} w_i e^{\epsilon (d_A^{A'_j} - d_i^{A'_j})} \\ \textit{s.t.} & \sum_{i \in \mathcal{C}_j \setminus A} w_i = 1 \end{aligned} \tag{9}$$

Algorithm 1 Camouflage Point Generation Algorithm

```
Input: Real location $A$, location set $\mathcal{X}$, number of camouflage
    point candidates $n$, camouflage range set $\mathcal{R}$, camouflage
    type set $\mathcal{T}$, indistinguishability parameter $\epsilon$
Output: Camouflage point $A'$
 1. Randomly choose $\theta_1, \theta_2, ..., \theta_n$ uniformly from $[0, 2\pi)$,
    and $z_1, z_2, ..., z_n$ uniformly from [0,1)
 2. Compute $d_j = C_\epsilon^{-1}(z_j)(j = 1, 2, ..., n)$, where $C_\epsilon(z_j) =$
    $1 - (1 + \epsilon z_j)e^{-\epsilon z_j}$
 3. Map $(d_j, \theta_j)$ to the closest point $A'_j$ on the grid of discrete
    cartesian coordinates within max($\mathcal{R}$)
 4. Create the observation sets centered with each $A'_j(j =$
    $1, 2, ..., n)$ for each pair from $\mathcal{R}$ and $\mathcal{T}$, and set priors for
    each POI within the observation sets
 5. Return $A'_j$ which maximizes (9)
```

4) Proof of Privacy Guarantees: First, the proposed camouflage algorithm guarantees ϵ-geo-indistinguishability based on the definition in (6). Since the algorithm generate camouflage point following planar laplacian distribution, the probability that A generates A' should be $P(A'|A) = \frac{\epsilon^2}{2\pi} e^{-\epsilon d_A^{A'}}$ and the probability that i generates A' should be $P(A'|i) = \frac{\epsilon^2}{2\pi} e^{-\epsilon d_i^{A'}}$ after observing A'. Suppose that A' is within r from A, then we have $d_i^{A'} < r$ for arbitrary i in the location observation set according to the proposed algorithm. Therefore, we have (10), which accords with the definition of ϵ-geo-indistinguishability.

$$\frac{P(A'|A)}{P(A'|i)} = \frac{e^{-\epsilon d_A^{A'}}}{e^{-\epsilon d_i^{A'}}} = e^{\epsilon (d_i^{A'} - d_A^{A'})} \leq e^{\epsilon d_i^{A'}} \leq e^{\epsilon r} \tag{10}$$

Second, the proposed camouflage algorithm resists location analysis attack with an optimized success rate. The camouflage point generated by the algorithm has the maximum value of objective function among the n candidates. The value approximates optimal when n approximates infinity. Under the laplacian mechanism, $e^{\epsilon (d_i^{A'_j} - d_A^{A'_j})}$ is equal to $dis(i, A) = dis(A, i)^{-1}$ with A'_j as the camouflage point. Additionally, the priors of points out of $\mathcal{C}$ is zero since the real location is no more R_A^2 away from A'_j. Therefore, the objective function in (7) accords with $\sum_{i \in \mathcal{X} \setminus A} dis(A, i)^{-1} P(i)$, which means that the generated camouflage point maximizes the success rate of resisting location analysis attack according to (4).

V. Location Access Control

Managing the exposure of the location information is still a challenge. First, the trust relationships among users are various and dynamic. Second, there are usually no secure channels shared between arbitrary two LBS users, as between a LBS user and a LBS server. In this section, an automatic location access control (LAC) mechanism based on our previously proposed SCP-ABE algorithm [9] is proposed.

A. SCP-ABE Algorithm

The SCP-ABE algorithm was proposed to perform access control on multi-dimension scalable data. Fig. 5 shows an example of 2-by-2 scalable data structure, containing four

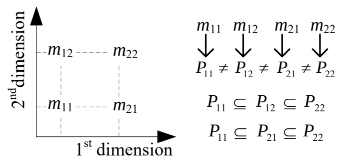
2nd dimension
1st dimension
m_{12}
m_{22}
m_{11}
m_{21}
m_{11} m_{12} m_{21} m_{22}
$P_{11} \neq P_{12} \neq P_{21} \neq P_{22}$
$P_{11} \subseteq P_{12} \subseteq P_{22}$
$P_{11} \subseteq P_{21} \subseteq P_{22}$

Fig. 5. 2-by-2 scalable data structure and the corresponding access policies in SCP-ABE

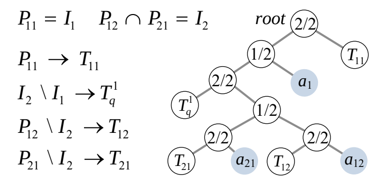
$P_{11} = I_1$ $P_{12} \cap P_{21} = I_2$
$P_{11} \to T_{11}$
$I_2 \setminus I_1 \to T_q^1$
$P_{12} \setminus I_2 \to T_{12}$
$P_{21} \setminus I_2 \to T_{21}$
root
2/2
1/2
T_{11}
2/2
a_1
T_q^1
1/2
2/2
2/2
T_{21}
a_{21}
T_{12}
a_{12}

Fig. 6. SCP-ABE access tree for 2-by-2 scalable LAC

messages m_{11}, m_{12}, m_{21}, and m_{22}, where (m_{11}) alone discloses the basic information. The disclosed information is enriched with the additional messages located at higher level (m_{12}, m_{21}, and m_{22}). Therefore, there are totally four access privileges under the 2-by-2 scalable data structure, i.e, m_{11} accessible, m_{11} and m_{12} accessible, m_{11} and m_{21} accessible, and all accessible. The access to each message is managed under a specific policy determined by attributes, i.e., P_{11}, P_{12}, P_{21}, and P_{22}. The access policy of a high-level message (e.g., m_{12}) is more restricted (contains more attributes) than that of a low-level message (e.g., m_{11}), and hence $P_{11} \subseteq P_{12} \subseteq P_{22}$ and $P_{11} \subseteq P_{21} \subseteq P_{22}$. The algorithm includes system setup, access tree construction, encryption, decryption, user key generation, and delegation.

B. SCP-ABE Based LAC Mechanism

Upon integrating the SCP-ABE algorithm with the SSPA-LBS system, we let the camouflage location information be the input messages of the SCP-ABE algorithm. In the example shown in Fig. 1, we input three messages including $m_{11} = (A', R_A^2, T_2)$, $m_{12} = (R_A^1)$, and $m_{21} = (T_1)$ to the SCP-ABE algorithm. The access of m_{11}, (m_{11}, m_{12}), (m_{11}, m_{21}), and (m_{11}, m_{12}, m_{21}) corresponds to the access of Alice's location information with four precisions. Moreover, the attributes used in SSPA-LBS system contain two types, i.e., LBS-related attributes and LBS-unrelated attributes. Specifically, LBS-related attributes may include relative distance, the number of co-visited places, the social relationship, etc. LBS-unrelated attributes refer to dummy attributes in SCP-ABE algorithm and only need to be authorized by the AA. By contrast, the LBS-related attributes, such as friendships and the relative distance between two users, will additionally require the authentication from the LBS server. Additionally, we assume there is a TTP playing the role of AA. The TTP does not store any location information of users, which is different from the role of TTP in other systems. Thus there is no location privacy leakage caused by TTP compromise.

We demonstrate the details of SCP-ABE based LAC mechanism utilizing the nearby friend notification example shown in Fig. 1, where Alice sets four privacy levels and Bob desires to receive the alert when Alice is nearby. The whole process involves initialization, location report, and location request.

1) Initialization: Alice runs initialization as below:

- Run the SCP-ABE setup algorithm to create the public key PK as in (11) and the master key MK as in (12).
- Select the LBS-related attribute set S_A^l and the LBS-unrelated attribute set S_A^n. Run the SCP-ABE access tree construction algorithm to build access tree $\mathcal{T}_A$, as shown in Fig. 6. Specifically, attributes from S_A^l are included in subtrees T_{11}, T_{12}, and T_{21}. S_A^n are a_1, a_{21}, and a_{21}.
- Cooperate with the AA, run the SCP-ABE user key generation algorithm to create user secret key SK^A for all attributes, where r and r_i are random numbers, and $attri_i$ is the context of attribute i.

$$PK = \{G_0, g, h = g^{\beta}, f = g^{1/\beta}, e(g,g)^{\alpha}\} \tag{11}$$

$$MK = \{\beta, g^{\alpha}\} \tag{12}$$

$$\begin{aligned} SK^A = \{&D = g^{(\alpha+r)/\beta}, \forall i \in S_A^n \cup S_A^l : \\ &D_i = g^r H(attri_i)^{r_i}, D_i' = g^{r_i}\} \end{aligned} \tag{13}$$

In LBS, user attributes change more frequently compared to that in other systems, due to the user movement. This requires timely update of the access tree or user secret keys. Therefore, Alice should periodically runs the initialization step.

2) Location Report: Alice runs the following steps to report camouflage location to servers.

- Run the camouflage algorithm to generate the camouflage location information (A', R_A^1, T_1), (A', R_A^2, T_1), (A', R_A^1, T_2), and (A', R_A^2, T_2), and select one to report $((A', R_A^2, T_2)$ in this example).
- Run the SCP-ABE encryption algorithm to encrypt (A', R_A^2, T_2), R_A^1 and T_1, and send the ciphertext CT_A along with the reported camouflage location to the server.

$$\begin{aligned} CT_A = (&\mathcal{T}_A, \forall i \in S_A^n \cup S_A^l : E_i = g^{p_i(0)}, E_i' = H(attri_i))^{p_i(0)} \\ &\tilde{C_{11}} = m_{11}^A e(g,g)^{\alpha(p_{R_{11}}(0)+s)}, C_{11} = h^{p_{R_{11}}(0)+s}, \\ &\tilde{C_{12}} = m_{12}^A e(g,g)^{\alpha(p_{R_{12}}(0)+s)}, C_{12} = h^{p_{R_{12}}(0)+s}, \\ &\tilde{C_{21}} = m_{21}^A e(g,g)^{\alpha(p_{R_{21}}(0)+s)}, C_{21} = h^{p_{R_{21}}(0)+s}) \end{aligned} \tag{14}$$

Specifically, s is the root secret, p_i is the polynomial corresponding to leaf node i, $p_{R_{11}}, p_{R_{12}}, p_{R_{21}}$ are the polynomials corresponding to subtree root nodes of T_{11}, T_{12}, and T_{21} respectively. These parameters are created in access tree construction algorithm [9].

On receiving Alice's location report, the server applies for the secret key SK^{Al} as in (15) from the AA. Note that these secret keys are for authorizing the LBS-unrelated attributes and delegating the corresponding user secret keys for other LBS users. The LBS server cannot successfully decrypt CT_A using SK^{Al}, which will be further explained in security analysis.

$$\begin{aligned} SK^{Al} = \{&D = g^{(\alpha+r)/\beta}, \forall i \in S_A^l : \\ &D_i = g^r H(attri_i)^{r_i}, D_i' = g^{r_i}\} \end{aligned} \tag{15}$$

3) Location Request: Suppose that Bob desires to receive an alert when Alice steps into the range with distance no more than r_B from his location. He sends the nearby friend notification request to the LBS server. Upon receiving the request, the LBS server tracks the location of Alice and Bob, and alert Bob whenever Alice steps into the serving range. To guarantee that there is no missed alert, the LBS server has to enlarge the serving range, which is $r_B + R_A^2$ in this example. The serving range could be further enlarged if Bob is also camouflaging. However, the enlargement on serving range will result in wrong alerts, which are identifiable or partially identifiable depending on Bob's access privilege on Alice's location information. Therefore, in order to identify wrong alerts when receiving the alerts, Bob requests for Alice's location information (CT_A) from the LBS server.

To obtain Alice's location information in plain text, Bob performs the following steps to obtain user secret key SK_B^A.

- Request for LBS-unrelated attribute authorization ($S_1 \subseteq S_A^n$) and the corresponding user secret key SK_{B0}^{An} as in (16) from AA.
- Request for LBS-related attribute authorization ($S_2 \subseteq S_A^l$) and the corresponding user secret key SK_B^{Al} as in (20) from the server, where r^B and r_i^B are random numbers specifically assigned by the LBS server for Bob, and r_i^B is associated with attribute i. r^B and r_i^B are kept secret by the server.
- Run the SCP-ABE delegation algorithm and creates the new user key SK_{B1}^{An} as in (17), where r^1 and $r_i^1 (i \in S_1)$ are randomly selected and kept secret by Bob..
- Send SK_{B1}^{An} to the server, and have SK_{B2}^{An} regenerated by the server. Derive SK_B^{An} from SK_{B2}^{An} as in (19), and finally obtains SK_B^A by (21).

$$SK_{B0}^{An} = \{\forall i \in S_1 : D_i = g^r H(attri_i)^{r_i}, D_i' = g^{r_i}\} \tag{16}$$

$$\begin{aligned} SK_{B1}^{An} =& \{\forall i \in S_1 : \\ & D_i^1 = D_i g^{r^1} H(attri_i)^{r_i^1}, D_i^{1'} = D_i' g^{r_i^1}\} \end{aligned} \tag{17}$$

$$\begin{aligned} SK_{B2}^{An} =& \{\forall i \in S_1 : \\ & D_i^2 = D_i^1 g^{r^B} H(attri_i)^{r_i^B}, D_i^{2'} = D_i^{1'} g^{r_i^B}\} \end{aligned} \tag{18}$$

$$\begin{aligned} SK_B^{An} =& \{\forall i \in S_1 : \\ & \tilde{D}_i = D_i g^{r^B} \cdot H(attri_i)^{r_i^B}, \tilde{D}_i{}' = D_i' g^{r_i^B}\} \end{aligned} \tag{19}$$

$$\begin{aligned} SK_B^{Al} =& \{\tilde{D} = D f^{r^B}, \forall i \in S_2 : \\ & \tilde{D}_i = D_i g^{r^B} \cdot H(attri_i)^{r_i^B}, \tilde{D}_i{}' = D_i' g^{r_i^B}\} \end{aligned} \tag{20}$$

$$\begin{aligned} SK_B^{A} =& \{\tilde{D} = D f^{r^B}, \forall i \in S_1 \cup S_2 : \\ & \tilde{D}_i = D_i g^{r^B} \cdot H(attri_i)^{r_i^B}, \tilde{D}_i{}' = D_i' g^{r_i^B}\} \end{aligned} \tag{21}$$

In the above process, the LBS server runs SCP-ABE delegation algorithm twice. However, the server reuse the random numbers in generating SK_B^{Al} to generate SK_{B2}^{An}. These random numbers are unique for each LBS user at each location request.

Bob then needs to decrypt CT_A using SK_B^A by running the SCP-ABE decryption algorithm. Suppose that Bob's access privilege guarantees him to access (A', R_A^2, T_2). Then Bob can decrypt each node x starting from the level of T_{11} as in (22). At the end, Bob will obtain $F_{R_{11}} = e(g,g)^{r p_{R_{11}}(0)}$ (F_x of T_{11} subtree root) and $F_{root} = e(g,g)^{rs}$ (F_x of tree root). Bob can further obtain K_{11} as in (23), where $r_B = r + r^B$. m_{11}^A can then be decrypted according to (24). Details of decryption process can be referred in [9]. Bob will eventually decrypt (A', R_A^2, T_2), which will facilitate the identification between the correct alerts and wrong ones.

$$\begin{aligned} F_x &= \frac{e(\tilde{D}_x, E_x)}{e(\tilde{D}_x', E_x')} \\ &= \frac{e(g^{r+r^B} \cdot H(attri_x)^{r_x + r_x^B}, g^{p_x(0)})}{e(g^{r_x + r_x^B}, H(attri_x)^{p_x(0)})} \\ &= \frac{e(g^{r+r^B}, g^{p_x(0)}) \cdot e(H(attri_x)^{r_x + r_x^B}, g^{p_x(0)})}{e(g^{r_x + r_x^B}, H(attri_x)^{p_x(0)})} \\ &= e(g,g)^{(r+r^B)p_x(0)} \end{aligned} \tag{22}$$

$$K_{11} = F_{R_{11}} \cdot F_{root} = e(g,g)^{r_B(p_{R_{11}}(0)+s)} \tag{23}$$

$$\begin{aligned} & \frac{\tilde{C_{11}}}{e(C_{11}, \tilde{D})/K_{11}} \\ &= \frac{m_{11}^A e(g,g)^{\alpha(p_{R_{11}}(0)+s)}}{e(h^{p_{R_{11}}(0)+s}, g^{(\alpha+r+r^B)/\beta})/e(g,g)^{r_B(p_{R_{11}}(0)+s)}} \\ &= \frac{m_{11}^A e(g,g)^{(\alpha+r_B)(p_{R_{11}}(0)+s)}}{e(g^{\beta(p_{R_{11}}(0)+s)}, g^{(\alpha+r_B)/\beta})} \\ &= \frac{m_{11}^A e(g,g)^{(\alpha+r_B)(p_{R_{11}}(0)+s)}}{e(g,g)^{\beta(p_{R_{11}}(0)+s)(\alpha+r_B)/\beta}} \\ &= m_{11}^A \end{aligned} \tag{24}$$

C. Reliability Analysis

The SCP-ABE based LAC mechanism provides reliability for both user access control and server access control.

1) Reliable User Access Control: The proposed LAC mechanism prevents users to decrypt location information that exceeds their access privileges through collusion.

The collusion means that two users collude with each other to derive more precise location information. Note that the collusion of directly exchanging decrypted location information is beyond the capability of any access control mechanism. Suppose that Bob owns the attributes satisfying the access policy P_{11} of Alice, and another user Clark owns the attributes satisfying $P_{21} \setminus P_{11}$. According to SCP-ABE, Bob can only obtain m_{11}^A, while Clark should not obtain any access keys due to the lack of attributes satisfying P_{11}. The goal of their collusion is to obtain m_{21}^A. Therefore, Bob and Clark will have to obtain K_{21}, which is computable from $F_{R_{21}}$ and F_{root} as in (23). However, Bob and Clark are not able to obtain K_{21} even if they exchange their intermediate computational results. More specifically, after separately running the SCP-ABE decryption algorithm, Bob has $F_{root} = e(g,g)^{r_B s}$ and $F_{R_{11}} = e(g,g)^{r_B p_{R_{11}}(0)}$ while Clark can obtain $F_{R_{21}} = e(g,g)^{r_C p_{R_{21}}(0)}$. Since r_B and r_C are random values specially designated for Bob and Clark, F_{root} computed by Bob and $F_{R_{21}}$ computed by Clark cannot be combined together to compute K_{21}. Further, Bob and Clark cannot obtain r_B and r_C since obtaining them from g^{r_B} and g^{r_C} are computationally infeasible according to Diffie-Hellman assumption [33].

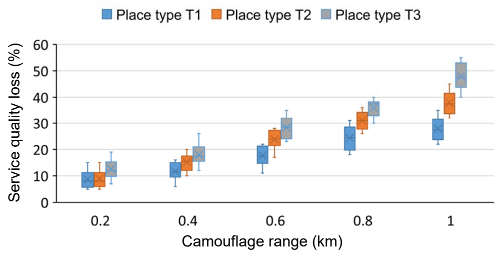

Fig. 7. Service quality loss varies with camouflage range and place type

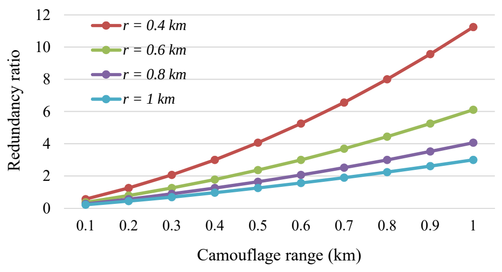

Fig. 8. Redundancy ratio varies with camouflage range

2) Reliable Server Access Control: The proposed SCP-ABE LAC mechanism can guarantee Alice's desired privacy level with respect to the LBS server.

First, the server is not able to decrypt any messages using its secret keys. In the proposed LAC mechanism, the LBS server owns all LBS-related attributes while not any LBS-unrelated attributes. As long as the access tree guarantees that all access privileges require LBS-unrelated attributes, the server can be prevented to obtain F_{root} and hence cannot decrypt any messages.

Second, the server are not able to derive the required secret keys from user secret keys. In order to perform the SCP-ABE decryption algorithm, Bob needs to use SK_B^{An} instead of SK_{B0}^{An} that is issued by the AA. If Bob directly sends SK_{B0}^{An} to the LBS server and requests the server to run delegation algorithm based on SK_B^{An}, the LBS server would obtain Bob's SCP-ABE user key. Along with SK_S^{Al}, the LBS server will have the access privilege no lower than Bob's. As a result, Alice may lose her desired privacy guarantee with respect to the server. However, in the proposed LAC mechanism, this reliability issue can be avoided because Bob first runs the delegation algorithm, and only sends the output SK_{B1}^{An} to the LBS server without disclosing r^1 and r_i^1. It is then computationally infeasible for the LBS server to derive $g^r H(attri_i)^{r_i}$ from $g^{r+r^1} H(attri_i)^{r_i+r_i^1}$, or g^{r_i} from $g^{r_i+r_i^1}$, which makes it impossible to obtain SK_{B0}^{An}.

VI. Cost Evaluation

The proposed camouflage algorithm and the proposed LAC mechanism bring scalable privacy guarantee and social friendliness for LBS users, enabling flexible balance between privacy level and service quality in diverse applications. As a sacrifice, they result in additional cost in the LBS system, including communication cost and computation cost. Note that the proposed privacy preserving scheme is application/dataset agnostic. No matter what type of application is under consideration, the proposed scheme will be executed on a standard fixed data format for each location exchange. Therefore, we carry out the cost evaluation for each location exchange between two users or between one user and one server.

A. Service quality loss

To demonstrate the impact of both camouflage range and place type on service quality loss, we study the nearby friend notification application and conduct experiments on the gowalla data set, which contains 6,442,890 user check-ins from 196,591 users [25]. We first choose 100 check-ins from the date set. Utilizing the proposed camouflage algorithm, we generate a camouflage point for each check-in place, by setting camouflage range from 0.2 kilometer to 1 kilometer and place type context as T_1 (e.g. *starbucks*), T_2 (e.g. *cafe*), and T_3 (e.g. *cafe & restaurant*), respectively. In particular, the gowalla data set only provides T_1 while T_2 and T_3 are manually marked. In practice, the user can select APIs that provides hierarchical categories for POIs, such as foursquare and yahoo geoplanet [26]. The nearby friend notification is simulated in this way: centered at each check-in place A, we randomly select one place B within 0.5 kilometer (i.e. the notification range) as the location of the nearby friend. We then measure whether the "friend" can identify that the user is within 0.5 kilometer by learning the camouflage point A', the camouflage range R_B, and the place type T_B. Reflected by the algorithm, we can measure whether the place satisfying 1) place type is T_B; 2) within R_B from A'; 3) within 0.5 km from B, is unique. If yes, the nearby friend is identified, otherwise it is failed. We define service quality loss as the ratio of the fails and the total number of check-ins, and measure its minimum value, maximum value, first quartile, median, third quartile, and mean (marked as cross) with different camouflage ranges and place types among the 100 check-ins, as shown in Fig. 7. The figure clearly indicates that a larger camouflage range and a less precise place type both result in higher service quality loss. In particular, the impact of place type is generally more significant when the camouflage range is larger.

B. Communication Cost

In the SSPA-LBS system, LBS server enlarges its serving area in order to guarantee that all user expected results are included, which inevitably results in additional communication cost on both server side and user side. In particular, we define redundancy ratio $\mathcal{D}$, i.e., the ratio of the number of redundant results and the number of expected results, as in [8] to measure the communication cost of the scheme. A higher redundancy ratio brings more extra communication cost, and vice versa.

In general, $\mathcal{D}$ is affected by the privacy level, i.e., the precision of reported user locations. $\mathcal{D}$ can be represented as in (25), where C is the expected count of responses, and C_e is the extended count of responses. In nearby friend notification service, where alert is based on the distance between two users, the camouflage range itself determines C and C_e. In personal advertising where the place type is a key perspective for service provision, C and C_e additionally depend on the place

TABLE I
REDUNDANCY RATIO VARIES WITH PLACE TYPE

Expected type	Extended type	$\mathcal{D}_{median}$	$\mathcal{D}_{0.75}$	$\mathcal{D}_{0.25}$
cafe	*cafe&restaurant*	1.15	1.32	0.91
salon	*salon&spa*	0.88	1.05	0.74
swimming	*swimming&fitness*	1.21	1.50	1.00

type. We first use nearby friend notification application to illustrate how $\mathcal{D}$ varies with camouflage range R. Furthermore, suppose that all user locations are uniformly distributed, $\mathcal{D}$ can be represented as in (26), where r is the expected serving range, r_e is the extended serving range ($r_e = r + R$), and $\pi r_e^2 - \pi r^2$ is the area containing redundancies. From Fig. 8, we conclude that heavy communication cost can be avoided if the camouflage range does not exceed the expected serving range too much. We then use nearby restaurant recommendation application to show how $\mathcal{D}$ varies with place type T. More specifically, we choose 100 most popular check-ins from gowalla data set, and study nearby *cafe* vs *cafe&restaurant* recommendations within a serving range of 1 km, *salon* vs *salon&spa* recommendations within a serving range of 2 km, *swimming* vs *fitness&swimming* recommendations within a serving range of 3 km. In this case, C is the count of *cafe/salon/swimming* recommendations, C_e is the count of *cafe&restaurant/salon&spa/fitness&swimming* recommendations. Results are listed in Table I, where $\mathcal{D}_{median}$, $\mathcal{D}_{0.75}$, and $\mathcal{D}_{0.25}$ represent the median, the third quartile, and the first quartile of the redundancy ratio, respectively. We observe that extended types introduce the redundancy as expected. The swimming&fitness extended type has the highest $\mathcal{D}$. This is because in this particular dataset the number of fitness location over swimming location is higher than other two cases, indicating a potential higher density for fitness centers.

$$\mathcal{D} = \frac{C_e - C}{C} \tag{25}$$

$$\mathcal{D} = \frac{\pi r_e^2 - \pi r^2}{\pi r^2} \tag{26}$$

C. User Side Computation Cost

User side computation mainly comes from two phases: the location report phase that consists of the camouflage algorithm and the SCP-ABE encryption and the location request phase where data decryption and delegation algorithms are involved. We implement the user side algorithms on a Google Nexus 4 (1.5GHz quad-core Snapdragon S4 Pro with Krait CPUs, Android 5.0) and measure the involved computations.

Camouflage: The camouflage algorithm is computationally trivial. The process of the proposed camouflage algorithm indicates that the computation cost linearly increases with the number of the surrounding PoIs and the number of camouflage point candidates n. In general, the cost is about n times of that of Miguel Andrés's camouflage algorithm. In the experiment, we retrieve 30 popular places in entertainment category from google place API as the user locations. We set $\epsilon = 1$, camouflage range $R = 0.2km$, place type as $entertainment$, and candidate number $n = 6$. The average number of POIs

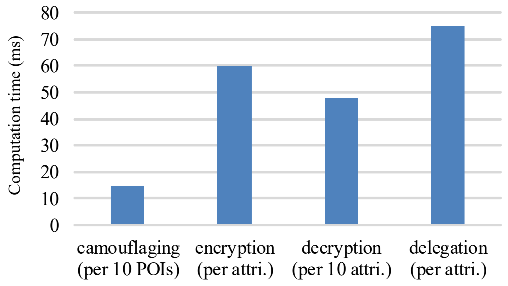

Fig. 9. Computation time of mobile side operations

that satisfies these settings for each user location is 10. The measurement of the average computation time of generating a camouflage point is ∼15 ms.

Location encryption: The top two types of heavy computations involved by SCP-ABE encryption are pairings and exponentiations on G_0 and G_1. As in [34], operations are conducted using a 160-bit elliptic curve group based on the curve $y^2 = x^3 + x$ over 512-bit finite field. The computation time of encrypting a message linearly increases with the number of attributes. By fixing the message length as 64 bytes, we run the SCP-ABE encryption algorithm ten times and obtain the average computation time as 60ms per attribute.

Location decryption: SCP-ABE decryption mainly contains two parts of computations, i.e. pairings and multiplications on G_1. For encryption, the cost of decryption is approximately linear with respects to the number of attributes. The average computation time is 48ms per ten attributes.

Delegation: This algorithm involves in exponentiations and multiplications on G_0 and G_1. The computation time also linearly increases with the number of attributes. The measured average time is 75ms per attribute.

Fig. 9 shows a summary of the average computation time of the above mentioned operations.

D. Server Side Computation Cost

The LBS server side computation cost comes from delegation. The AA side computation cost comes from user key generation. Both of the two algorithms involve in exponentiations and multiplications on G_0 and G_1. Since the user key generation and the delegation in the SCP-ABE algorithm is the same as in CP-ABE algorithm. We measure the cost based on the benchmark of CP-ABE algorithm [34]. As provided in [34], the computation time of user key generation is about 30ms per attribute on workstation with 64-bit 3.2 Ghz Pentium 4 processor. Besides, the computation time of delegation is almost the same due to similar computation operations.

We further measure the computation cost on the side of Alice, Bob, the AA, and the LBS server with various number of N attributes. Alice, who is camouflaging and hence performs the camouflage algorithm as well as data encryption, spends $15ms + N \times 60ms$ in the service. Bob, who is requesting Alice's location, needs to perform delegation and decryption. In particular, the delegation time is at most $2 \times (N/2) \times 75ms$ since $S_2 \in S_A^l$, while the decryption time is $(N/10) \times 48ms$. Additionally, the AA will spend $N \times 30ms$ assigning the

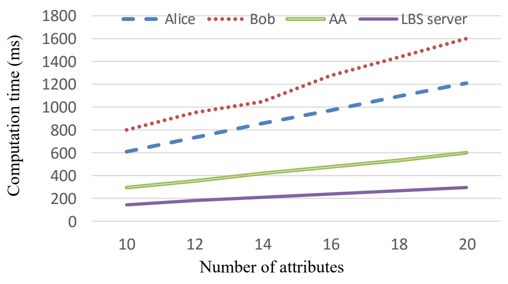

Fig. 10. Computation time with various number of attributes

user key for Alice. The LBS server will spend at most $(N/2) \times 30ms$ for delegating SK_B^{Al} to Bob.

By selecting N from 10 to 20, we evaluate the computation time on each communication party's side in Fig. 10. The results show that running the nearby friend notification service in the SSPA-LBS model costs around 1 seconds on the mobile side, which is at the same level of the average response time of general mobile applications[35]. Therefore, we conclude that the SSPA-LBS model could be efficiently applied on the mobile side, an important fact for privacy-aware LBSs.

VII. Conclusion And Future Work

In this paper, we have presented the SSPA-LBS system that meets both scalable privacy protection and social interaction requirements of the LBS users. Through experimental evaluation, we have demonstrated the cost efficiency of the SSPA-LBS system. Future work shall include two major aspects. First, it is essential to develop strategies for sustained privacy preservation on power-constrained mobile device. Although the proposed camouflage algorithm and LAC mechanism shows acceptable performance on mobile device, it is not efficient enough to let LBS users frequently camouflage and encrypt their locations whenever moving to a new place. Second, it is desirable to enable the LAC mechanism to handle the user key revocation more conveniently. In general, previous user secret keys need to be revoked when the user attributes are changed. In LBS, user movement is normal and frequent, resulting in dynamic LBS-related attributes. This brings a great challenge to develop efficient and reliable LAC.

References

[1] X. Wang et al., "Semantic-Based Location Recommendation With Multimodal Venue Semantics," *IEEE Transactions on Multimedia*, vol. 17, no. 3, pp. 409-419, March 2015.

[2] G. Zhao, X. Qian and X. Xie, "User-Service Rating Prediction by Exploring Social Users' Rating Behaviors," *IEEE Transactions on Multimedia*, vol. 18, no. 3, pp. 496-506, March 2016.

[3] I. Bisio, F. Lavagetto, M. Marchese and A. Sciarrone, "GPS/HPS-and Wi-Fi Fingerprint-Based Location Recognition for Check-In Applications Over Smartphones in Cloud-Based LBSs," *IEEE Transactions on Multimedia*, vol. 15, no. 4, pp. 858-869, June 2013.

[4] Z. Liu, L. Zhang, Q. Liu, Y. Yin, L. Cheng and R. Zimmermann, "Fusion of Magnetic and Visual Sensors for Indoor Localization: Infrastructure-Free and More Effective," *IEEE Transactions on Multimedia*, vol. 19, no. 4, pp. 874-888, April 2017.

[5] P. Golle and K. Partridge, "On the anonymity of home/work location pairs," *Pervasive Computing*, 2009.

[6] M. Yiu, C. Jensen, X. Huang, and H. Lu, "SpaceTwist: Managing the trade-offs among location privacy, query performance, and query accuracy in mobile services," *IEEE ICDE,* pp. 366-375, 2008.

[7] R. Shokri, G. Theodorakopoulos, J.Y.L. Boudec, and J.P. Hubaux, "Quantifying Location Privacy,*Proc. IEEE 32nd Symp. Security and Privacy,* pp. 247-262, 2011.

[8] C. Ma and C. W. Chen, "Nearby Friend Discovery with Indistinguishability for Stalkers," *11th International Conference on Mobile Systems and Pervasive Computing,* vol. 34, pp 352-359, 2014.

[9] C. Ma and C. W. Chen, "Attribute-Based Multi-Dimension Scalable Access Control For Social Media Sharing," *IEEE ICME*, 2016.

[10] Craig Gentry, "A fully homomorphic encryption scheme," *PhD thesis, Stanford University,* 2009.

[11] I. T. Lien, Y. H. Lin, J. R. Shieh and J. L. Wu, "A Novel Privacy Preserving Location-Based Service Protocol With Secret Circular Shift for K-NN Search," *IEEE Transactions on Information Forensics and Security,* vol. 8, no. 6, pp. 863-873, 2013.

[12] A. Solanas, A. Martínez-Ballesté, "Privacy protection in location-based services through a public-key privacy homomorphism," *4th European PKI Workshop: Theory and practice, Lecture Notes in Computer Science, Springer,* 2007, pp.362-368.

[13] A. Solanas, A. Martínez-Ballesté "A TTP-free protocol for location privacy in location-based services," *Computer Communications,* vol.31, no.6, pp. 1181-1191, 2008.

[14] L. Sweeney, "k-anonymity: a model for protecting privacy," *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.,* vol. 10, pp 557-570, 2002.

[15] A. Gkoulalas-Divanis, P. Kalnis, and V. S. Verykios, "Providing k-anonymity in location based services," *ACM SIGKDD,* vol. 12, no. 1, pp. 3-10, 2010.

[16] C. Y. Chow and M. F. Mokbel, "Enabling Private Continuous Queries For Revealed User Locations," *Proc. 10th international conference on Advances in spatial and temporal databases,* pp.258-273, 2007.

[17] S. Peddinti, A. Dsouza, and N. Saxena, "Cover Locations: Availing Location-Based Services Without Revealing the Location," *Proc. of the 10th annual ACM workshop on Privacy in the electronic society*, pp. 143-152, 2011.

[18] H. Kido, Y. Yanagisawa, and T. Satoh, "Protection of location privacy using dummies for location-based services," *IEEE 21st International Conference on Data Engineering Workshops,* pp. 1248-1248, 2005.

[19] H. Zang and J. Bolot, "Anonymization of location data does not work: a large-scale measurement study," *ACM MobiCom*, pp. 145-156, 2011.

[20] M. Duckham and L. Kulik, "A formal model of obfuscation and negotiation for location privacy,"*3rd international conference on pervasive computing, Springer,* pp. 152-170, 2005.

[21] M. Andrés, N. Bordenabe, K. Chatzikokolakis, and C. Palamidessi. "Geo-Indistinguishability: Differential Privacy for Location-Based Systems," *ACM CCS,* pp. 901-914, 2013.

[22] C. Ardagna, M. Cremonini, S. De Capitani di Vimercati, and P. Samarati, "An Obfuscation-Based Approach for Protecting Location Privacy," *IEEE TDSC,* vol.8, no.1, pp. 13-27, 2011.

[23] T. Xu and Y. Cai, "Feeling-based location privacy protection for location based services," *ACM CCS,* pp. 348-357, 2009.

[24] N. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, "Optimal geo-indistinguishable mechanisms for location privacy," *Proceedings of the 16th ACM conference on Computer and communications security,* pp.251-262, 2014.

[25] http://wiki.urbanhogfarm.com/index.php/Gowalla

[26] https://apievangelist.com/2012/03/04/overview-of-11-places-data-apis/

[27] R. Shokri, G. Theodorakopoulos, C. Troncoso, J. Hubaux, and J. Boudec, "Protecting Location Privacy: Optimal Strategy against Localization Attacks," *CCS,* pp. 617-627, 2012.

[28] C. Dwork. "Differential Privacy," *ICALP,* pp. 338-340, 2006.

[29] R. Dewri, "Local Differential Perturbations: Location Privacy under Approximate Knowledge Attackers," *IEEE TMC*, vol.12, no.12, pp.2360-2372, Dec. 2013.

[30] https://developers.google.com/maps/documentation/javascript/places

[31] http://developer.baidu.com/map/index.php?title=androidsdk

[32] M. D. Soete, "Attribute certificate,," *Encyclopedia of Cryptography and Security, H. C. A. Van Tilborg and S. Jajodia, Eds., Springer*, 2011, pp. 51.

[33] U. Maurer, "Towards proving the equivalence of breaking the Diffie-Hellman protocol and computing discrete logarithms", *Proc. of Crypto '94*, pp. 271-281.

[34] J. Bethencourt, A. Sahai, B. Waters, "Ciphertext-Policy Attribute-Based Encryption," *IEEE Symposium on Security and Privacy*, pp. 321-334, 2007.

[35] http://www.infoq.com/news/2014/03/mobile_app_performance benchmark