Privacy in Mobile Computing for Location-Sharing-Based Services

Igor Bilogrevic[1], Murtuza Jadliwala[1], Kübra Kalkan[2], Jean-Pierre Hubaux[1], and Imad Aad[3]

[1] Laboratory for Communications and Applications, EPFL, Lausanne, Switzerland
[2] Faculty of Engineering and Natural Sciences, Sabanci University, Istanbul, Turkey
[3] Nokia Research Center, Lausanne, Switzerland
firstname.lastname@{epfl.ch, nokia.com}, kubrakalkan@sabanciuniv.edu

Abstract. Location-Sharing-Based Services (LSBS) complement Location-Based Services by using locations from a group of users, and not just individuals, to provide some contextualized service based on the locations in the group. However, there are growing concerns about the misuse of location data by third-parties, which fuels the need for more privacy controls in such services. We address the relevant problem of privacy in LSBSs by providing practical and effective solutions to the privacy problem in one such service, namely the fair rendez-vous point (FRVP) determination service. The privacy preserving FRVP (PPFRVP) problem is general enough and nicely captures the computations and privacy requirements in LSBSs. In this paper, we propose two privacy-preserving algorithms for the FRVP problem and analytically evaluate their privacy in both passive and active adversarial scenarios. We study the practical feasibility and performance of the proposed approaches by implementing them on Nokia mobile devices. By means of a targeted user-study, we attempt to gain further understanding of the popularity, the privacy and acceptance of the proposed solutions.

1 Introduction

From Google to Facebook, online service providers are increasingly proposing sophisticated context-aware services in order to attract new customers and improve the user-experience of existing ones. Location-based services (LBS), offered by such providers and used by millions of mobile subscribers every day [8], have proven to be very effective in this respect.

Place check-ins and location-sharing are two popular features. By checking into a place, users share their current location with their families or friends, and the ones who do it frequently may also obtain special deals, provided by the nearby businesses, as incentives for sharing their locations [9]. Facebook, for instance, recently launched such a service by which users who want to check-in can look for on-the-spot discounts and deals [7]. Services based on *location-sharing*, already used by almost 20% of mobile users [18], are undoubtedly becoming popular. For instance, one recently announced application that exploits location data from different users is a taxi-sharing application, offered by a global telecom operator [19]. In order to share a taxi, users have to reveal their departure and destination points to the server.

Determining a suitable *location* for a set of users is a relevant issue. Several providers already offer variants of this service either as on-line web applications ([16,17]) or as stand-alone applications for mobile devices [17]. Not only is such a feature desirable, but it also optimizes the trade-off between convenience and cost for the involved parties.

However, there are growing concerns about how private information is used and processed by these providers. We conducted a study on privacy in location-sharing-based services (LSBS) with 35 participants (college students and non-scientific personnel), and according to the results 88% of them believe it is important to protect their location privacy from unauthorized uses. Similar results have been obtained in a different study on location-based services (LBS) [18]. Without effective protection, even sparse location information has been shown to provide reliable information about a user's private sphere, which could have severe consequences on the users' social, financial and private life [12]. For instance, a web service [21] has shown how thieves may misuse users' location updates (from a popular online social network) in order to rob their residences while they are not at home. In the taxi-sharing application, if the server is not fully trusted by all users, revealing sensitive locations (such as users home/work addresses) could pave the way for inference attacks by third-parties. Thus, the disclosure of location data to potentially untrusted third-parties and peers must be limited in any location-sharing-based service.

In this paper, we highlight the privacy issues in LSBS by studying one practical and relevant instance of such a general scenario, which is the determination of a *fair rendez-vous point (FRVP)* in a privacy-preserving way, given a set of user-provided locations. This is a novel and potentially useful problem for LSBS applications, which captures the essence of the computations that are generally required in any LSBS, and mitigates their inherent and important privacy issues. Our user-study indicates that 51% of the respondents would be very interested in such a service based on location-sharing.

Our contributions are as follows. First, we present the results of our targeted user-study on location-sharing and privacy in mobile services. Second, motivated by the results of this study and the need for privacy in LSBSs, we design and analyze two practical solutions to the FRVP problem, which do not reveal any additional information to third parties or other peers. The proposed solutions are independent of any underlying service or network provider, and can be included in existing location-sharing-based services. Third, we evaluate the robustness and resilience of our schemes to both passive and active attacks through a privacy analysis of the proposed solutions. Fourth, by implementing our proposed algorithms on a testbed of real mobile devices, we show that their performance in computing the rendez-vous point is acceptable, and that users do not incur in significant additional overhead due to the inherent privacy features.

2 Background and User Study

Background Novel LSB services, such as deals and check-ins, are offered by large service providers such as Google and Facebook. In order to assess users' opinions about the potential and challenges of such services, we conducted a targeted user study on 35 respondents, sampling a population of technology-savvy college students (in the age group of 20-30 years) and non-scientific personnel. The questionnaires are based on the privacy and usability guidelines from [5,13].

User-Study The entire study consisted of three phases; the goal of Phase 1, during which respondents answered a first set of 22 questions without knowing the subject of the study, was to assess the participants' level of adoption of mobile LSBS and their sensitivity to privacy issues in such services. The answers to these questions are either "Yes" or "No", or on a 4-point Lickert scale (where 1 means *Disagree*, 4 is *Agree*). In Phase 2, the respondents were instructed to use our prototype mobile FRVP application. Finally, in Phase 3, the participants answered the second set of 12 questions, choosing from a 4-point Lickert scale, after having used our application. The goal of this phase was to obtain feedback on the usability and privacy features of our prototype. The results of Phase 1 are described next, whereas Phase 2 and 3 are discussed in Section 7.2.

Phase 1 Results The majority of the respondents are males in the 20-25 year-age. Around 86% of them use social networks, and 74% browse the Internet with a mobile device. Although only 14% are aware of existing LSBS, 51% would be very or quite interested in using a LSBS such as the FRVP. However, people are sensitive to privacy (98%) and anonymity (74%) in their online interactions, especially with respect to the potential misuse of their private information by non-specified third-parties (88%). Due to space constraints, we are unable to include here the full details of the study.

These results indicate that, although rare at the moment, LSBSs are perceived as interesting by the majority of the sampled population, which is also the most likely to adopt LBS technologies [18]. With respect to privacy, people agree that it is crucial for the acceptability of such services, and thus LSBS should work properly by requiring a minimum amount of personal information.

In the next sections, we introduce the system architecture, the FRVP problem and our two solutions for computing the FRVP in a privacy-preserving way.

3 System Architecture

We consider a system which is composed of two main entities: (i) a set of users[4] (or mobile devices) $\mathbb{U} = \{u_1, \ldots, u_N\}$ and (ii) a third-party service provider, called *Location Determination Server (LDS)*. The N users want to determine the fair rendez-vous location that is computed by the LDS.

Each user's mobile device is assumed to be able to establish communication with the LDS either in a P2P fashion or through a fixed infrastructure-based

[4] Throughout this paper, we use the words *users* and *devices* interchangeably. The meaning is clear from the context, unless stated otherwise.

Internet connection. The mobile devices are able to perform public-key cryptographic operations, and each user u_i has means of determining the position $L_i = (x_i, y_i) \in \mathbb{N}^2$ of his preferred rendez-vous location (or his own location) by using a common coordinate system. We consider a two-dimensional position coordinates system, but the proposed schemes are general enough and can easily be extended to other practical coordinate systems. For instance, such definition of L_i can be made fully compliant with the UTM coordinate system [27], which is a plane coordinate system where points are represented as a 2-tuple of positive values (distances in meters from a given reference point).

We define the set of the preferred rendez-vous locations of all users as $\mathbb{L} = \{L_i\}_{i=1}^N$. For the sake of simplicity, we assume a flat-Earth model and we consider line-of-sight Euclidian distances between preferred rendez-vous locations. Even though the actual real-world distance (road, railway, boat, etc.) between two locations is at least as large as their Euclidian distance, the proportion between distances in the real world is assumed to be correlated with the proportion of the respective Euclidian distances. Location priorities, which are not discussed in this paper, can be used for isolated or unsuitable locations.

We assume that each of the N users has his own public/private key pair $(K_P^{u_i}, K_s^{u_i})$, certified by a trusted CA, which is used to digitally sign the messages that are sent to the LDS. Moreover, we assume that the N users share a common secret that is utilized to generate a shared public/private key pair $(K_P^{M_v}, K_s^{M_v})$ in an online fashion for each meeting setup instance v. The private key $K_s^{M_v}$ generated in this way is known only to all meeting participants, whereas the public key $K_P^{M_v}$ is known to everyone including the LDS. This could be achieved through a secure credential establishment protocol such as in [3,4,15].

The LDS executes the FRVP algorithm on the inputs it receives by the users in order to compute the FRV location. The LDS is also able to perform public-key cryptographic functions. For instance, a common public-key infrastructure using the RSA cryptosystem [22] could be employed. Let K_P^{LDS} be the public key, certified by a trusted CA, and K_s^{LDS} the corresponding private key of the LDS. K_P^{LDS} is publicly known and users encrypt their input to the FRVP algorithm using this key; the encrypted input can be decrypted by the LDS using its private key K_s^{LDS}. This ensures message confidentiality and integrity for all the messages exchanged between users and the LDS. For simplicity of exposition, in our protocols we do not explicitly show these cryptographic operations involving LDS's public/private key.

3.1 Threat Model

Location Determination Server The LDS is assumed to execute the algorithms correctly, i.e., take all the inputs and produce the output according to the algorithm. However, the LDS may try to learn information about users' location preferences from the received inputs, the intermediate results and the produced outputs. This type of adversarial behavior is usually referred to as *honest-but-curious* adversary (or semi-honest) [11]. In most practical settings, where service providers have a commercial interest in providing a faithful service to their customers, the assumption of a semi-honest LDS is generally sufficient.

Users The participating users also want to learn the private location preferences of other users from the output of the algorithm they receive from the LDS. We refer to such attacks as passive attacks. As user inputs are encrypted with the LDS's public key K_P^{LDS}, there is a confidentiality guarantee against basic eavesdropping by participants and non participants. In addition to these attacks, participating users may also attempt to actively attack the protocol by colluding with other users or manipulating their own inputs to learn the output.

4 The *Rendez-vous* Problem

In this work, we consider the problem of finding, in a privacy-preserving way, the rendez-vous point among a set of user-proposed locations, such that (i) the rendez-vous point is a point that is *fair* (as defined in Section 5.1) with respect to the given locations, (ii) each of the users gets to know only the final rendez-vous location and (iii) no participating user or third-party server learns private location information about any other user involved in the computations. We refer to an algorithm that solves this problem as *Privacy-Preserving Fair Rendez-Vous Point (PPFRVP)* algorithm. In general, any PPFRVP algorithm A should accept the inputs and produce the outputs, as described below.

- *Input*: a transformation f of private locations L_i: $f(L_1)||f(L_2)||\dots||f(L_N)$. where f is a one-way public function (based on secret key) such that it is hard (success with only a negligible probability) to determine the input L_i without knowing the secret key, by just observing $f(L_i)$.
- *Output*: an output $f(L_{fair}) = g(f(L_1), \dots, f(L_N))$, where g is a fairness function and $L_{fair} = (x_l, y_l) \in \mathbb{N}^2$ is the fair rendez-vous location that has been selected for this particular set of users, such that it is hard for the LDS to determine L_{fair} by just observing $f(L_{fair})$. Given $f(L_{fair})$, each user is able to compute $L_{fair} = f^{-1}(f(L_{fair}))$ using his local data.

The fairness function g can be defined in several ways, depending on the preferences of users or policies. For instance, users might prefer to meet in locations that are close to their offices, and their employers might prefer a place that is closest to their clients. In Section 5.1 we describe one such fairness function that minimizes the maximum displacement of any user to all other locations. Such function is globally fair and general enough, as it captures the essential computations required for optimization It can be extended to include more complex constraints and parameters.

5 Proposed Solutions and Analysis

In this section, we present our solution to the PPFRVP. First, we discuss the mathematical tools that we use in order to model the fairness function g and the transformation functions f. In order to achieve the integration between resource-constrained mobile devices and the client-server network paradigm, our solutions have to be efficient in terms of computations and communication complexities.

In order to separate the optimization part of the PPFRVP algorithm A from its implementation using cryptographic primitives, we first discuss the fairness function g and then the transformation function f.

5.1 Fairness Function *g*

In this work, we consider the fairness criterion that has been widely used in operations research to solve the *k-center* problem. In the k-center problem, the goal is to find $L_1, \ldots, L_k$ locations among N given possible places, in order to optimally place k facilities, such that the maximum distance from any place to its closest facility is minimized. For a two dimensional coordinate system, the Euclidian distance metric is usually employed.

As the PPFRVP problem consists in determining the fair rendez-vous location from a set of user-desired locations, we focus on the *k-center* formulation of the problem with $k = 1$. This choice is also grounded on the fact that not choosing L_{fair} from one of the location preferences $L_1, \ldots, L_N$ might potentially result in a location L_{fair} that is not suited for the kind of meeting that the participants require. The solution can easily be extended or integrated with mapping applications (on the users' devices) so that POIs around L_{fair} are automatically suggested for the meeting. Figure 1 shows an example PPFRVP scenario modeled as a k-center problem, where four users want to determine the fair rendez-vous location L_{fair}.

The k-center formulation considers the Euclidian distances, but it does not encompass other fairness parameters, such as accessibility of a place and the means of transportation. In this work, we focus on the pure k-center formulation as the essential building block of a more complete model, which can be extended when such an application is to be deployed in existing services.

Let $d_{ij} \geq 0$ be the Euclidian distance between two points $L_i, L_j \in \mathbb{N}^2$, and $D_i^M = \max_{j \neq i} d_{ij}$ be the maximum distance from L_i to any other point L_j. Then, the PPFRVP problem can be formally defined as follows.

Definition 1. *The* PPFRVP *problem is to determine a location* $L_{fair} \in \mathbb{L} = \{L_1, \ldots, L_N\}$, *where* $fair = arg \min_i D_i^M$

A solution for the PPFRVP problem finds, in a privacy-preserving way, the fair rendez-vous location among the set of proposed (and user-desired) locations, such that the distance of the furthest desired location to the fair one is minimized.

There are two important steps involved in the computation of the fair location L_{fair}. The first step is to compute the pairwise distances d_{ij} among all users $i, j \in \{1, \ldots, N\}$ participating in the PPFRVP algorithm. The second step requires the computations of the maximum and minimum values of such distances.

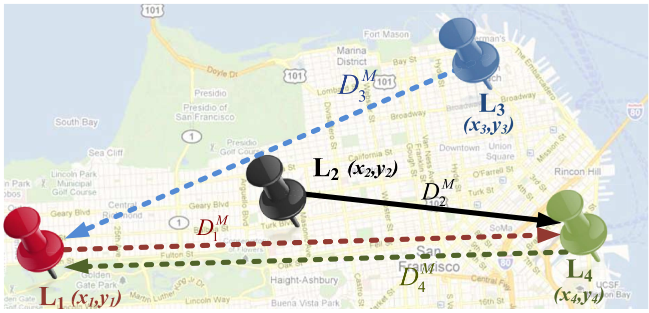


Fig. 1. PPFRVP scenario, where the fairness function is $g = \text{argmin}_i(D_i^M)$. The dashed arrows represent the maximum distance D_i^M from each user u_i to any user $j \neq i$, whereas the solid line is the minimum of all such maximum distances. The fair rendez-vous location is $L_{fair} = L_2 = (x_2, y_2)$.

5.2 Transformation Functions *f*

The fairness function g requires the computation of two functions on the private user-desired locations L_i: (i) the distance between any two locations $L_i \neq L_j$ and (ii) the minimum of the maximum of these distances. In order to achieve the final result and to preserve the privacy of the personal information, we rely on computationally secure cryptographic functions. In our protocol, we consider three such schemes: the *Boneh-Goh-Nissim* (BGN) [2], the *ElGamal* [6] and the *Paillier* [20] public-key encryption schemes.

What makes these schemes useful are their homomorphic encryption properties. Given two plaintexts m_1, m_2 with their respective encryptions $E(m_1), E(m_2)$, the multiplicative property (possessed by the ElGamal and partially by the BGN schemes) states that $E(m_1) \odot E(m_2) = E(m_1 \cdot m_2)$, where $\odot$ is an arithmetic operation in the encrypted domain that is equivalent to the usual multiplication operation in the plaintext domain. The additive homomorphic property (possessed by the BGN and the Paillier schemes) states that $E(m_1) \oplus E(m_2) = E(m_1 + m_2)$, where $\oplus$ is an arithmetic operation in the encrypted domain which is equivalent to the usual sum operation in the plaintext domain. Details about the initialization, operation and security of the encryption schemes can be found in [6,2,20].

Based on the three aforementioned encryption schemes, we now describe the distance computation algorithms that are used in our solution.

5.3 Distance Computations

In order to determine the fair rendez-vous location, we need to find the location L_{fair}, where $\mathit{fair} \in \{1, \ldots, N\}$, that minimizes the maximum distance between any user-desired location and L_{fair}. In our algorithms, we work with the *square* of the distances, as they are much easier to compute in an oblivious fashion using the homomorphic properties of the cryptographic schemes. The problem of finding the argument that minimizes the maximum distance is equivalent to finding the argument that minimizes the maximum distance *squared* (provided that all distances are greater than 1). Moreover, as squaring maintains the relative order, the algorithm is still correct.

BGN-distance Our first distance computation algorithm is based on the BGN encryption scheme. This novel protocol requires only one round of communication between each user and the LDS, and it works as follows. In Step 1, each user u_i, $\forall i \in \{1, \ldots, N\}$, creates the vectors

$$E_i(a) =< a_{i1}| \ldots |a_{i6} >=< E(x_i^2)|E(T-2x_i)|E(1)|E(T-2y_i)|E(y_i^2)|E(1) >$$
$$E_i(b) =< b_{i1}| \ldots |b_{i6} >=< E(1)|E(x_i)|E(x_i^2)|E(y_i)|E(1)|E(y_i^2) >$$

where $E(.)$ is the encryption of $(.)$ using the BGN scheme and $L_i = (x_i, y_i)$ is the desired rendez-vous location of user u_i. Afterwards, each user sends the two vectors $E_i(a), E_i(b)$ over a secure channel to the LDS. In Step 2, the LDS computes the scalar product of the received vectors by first applying the multiplicative and then the additive homomorphic property of the BGN scheme.

Paillier-ElGamal-distance An alternative scheme for the distance computation is based on both the Paillier and ElGamal encryption schemes, as shown in Figure 2. As neither Paillier or ElGamal possess both multiplicative and additive properties, the resulting algorithm requires one extra step in order to achieve the same result as the BGN-based scheme, i.e., obliviously computing the pairwise distances d_{ij}^2. The distances are computed as follows. In Step 1, each user u_i, $\forall i \in \{1, \ldots, N\}$, creates the vector

$$E_i(a) =< a_{i1}| \ldots |a_{i4} >=< Pai(x_i^2)|ElG(x_i)|Pai(y_i^2)|ElG(y_i) >$$

where $Pai(.)$ and $ElG(.)$ refer to the encryption of $(.)$ using the Paillier or ElGamal encryption schemes, respectively. Afterwards, each user u_i sends the vector $E_i(a)$ to the LDS, encrypted with LDS's public key. In the following steps of the protocol, the LDS computes the scalar products of the second and fourth elements of the received vectors (Step 2.1), randomizes (in an order-preserving fashion) the results and send a different set of values back to each user (Step 2.2). In Step 3, the users re-encrypt the values with the Paillier scheme and send it to the LDS, which then obliviously computes the pairwise distances (Step 4.1).

5.4 The *PPFRVP* Protocol

We now describe our protocol for the PPFRVP problem, as shown in Figure 3. The protocol has three main modules: (A) the distance computation module, (B) the MAX module and (C) the ARGMIN MAX module.

Distance computations The first module (distance computation) uses one of the two protocols defined in the previous subsection (BGN-distance or Paillier-ElGamal-distance). We note that modules (B) and (C) use the same encryption

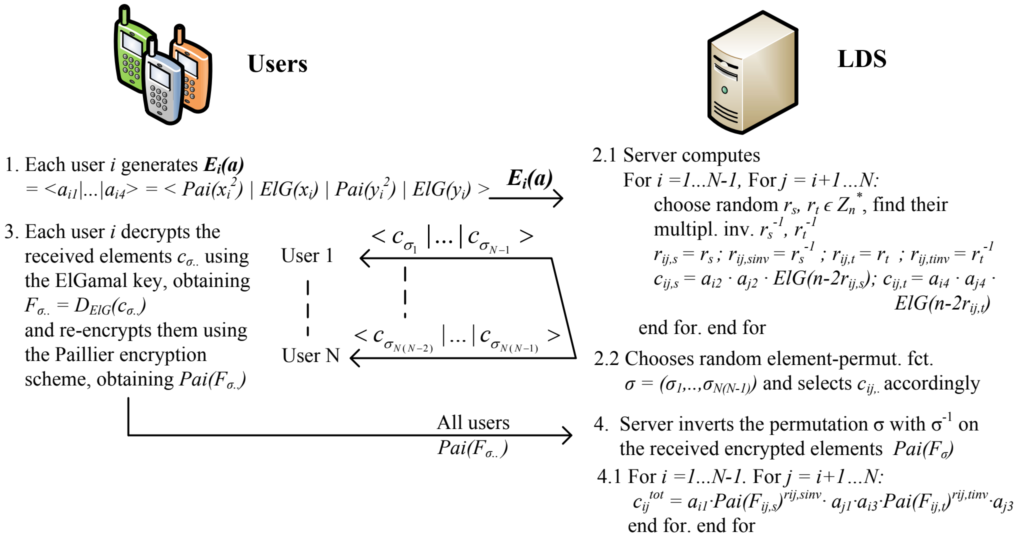

Users
LDS
1. Each user i generates $\boldsymbol{E_i(a)}$
$= <a_{i1}|...|a_{i4}> = < Pai(x_i^2) \mid ElG(x_i) \mid Pai(y_i^2) \mid ElG(y_i) >$
$\boldsymbol{E_i(a)}$
2.1 Server computes
For $i = 1...N-1$, For $j = i+1...N$:
choose random $r_s, r_t \epsilon Z_n^*$, find their
multipl. inv. r_s^{-1}, r_t^{-1}
$r_{ij,s} = r_s$; $r_{ij,sinv} = r_s^{-1}$; $r_{ij,t} = r_t$; $r_{ij,tinv} = r_t^{-1}$
$c_{ij,s} = a_{i2} \cdot a_{j2} \cdot ElG(n-2r_{ij,s})$; $c_{ij,t} = a_{i4} \cdot a_{j4} \cdot ElG(n-2r_{ij,t})$
end for. end for
3. Each user i decrypts the received elements $c_{\sigma..}$ using the ElGamal key, obtaining $F_{\sigma..} = D_{ElG}(c_{\sigma..})$ and re-encrypts them using the Paillier encryption scheme, obtaining $Pai(F_{\sigma..})$
User 1
$< c_{\sigma_1} |...| c_{\sigma_{N-1}} >$
User N
$< c_{\sigma_{N(N-2)}} |...| c_{\sigma_{N(N-1)}} >$
2.2 Chooses random element-permut. fct.
$\sigma = (\sigma_1,..,\sigma_{N(N-1)})$ and selects $c_{ij,.}$ accordingly
All users
$Pai(F_{\sigma..})$
4. Server inverts the permutation σ with σ^{-1} on the received encrypted elements $Pai(F_\sigma)$
4.1 For $i = 1...N-1$. For $j = i+1...N$:
$c_{ij}^{tot} = a_{i1} \cdot Pai(F_{ij,s})^{rij,sinv} \cdot a_{j1} \cdot a_{i3} \cdot Pai(F_{ij,t})^{rij,tinv} \cdot a_{j3}$
end for. end for

Fig. 2. Distance computation protocol based on the ElGamal and Paillier encryption schemes.

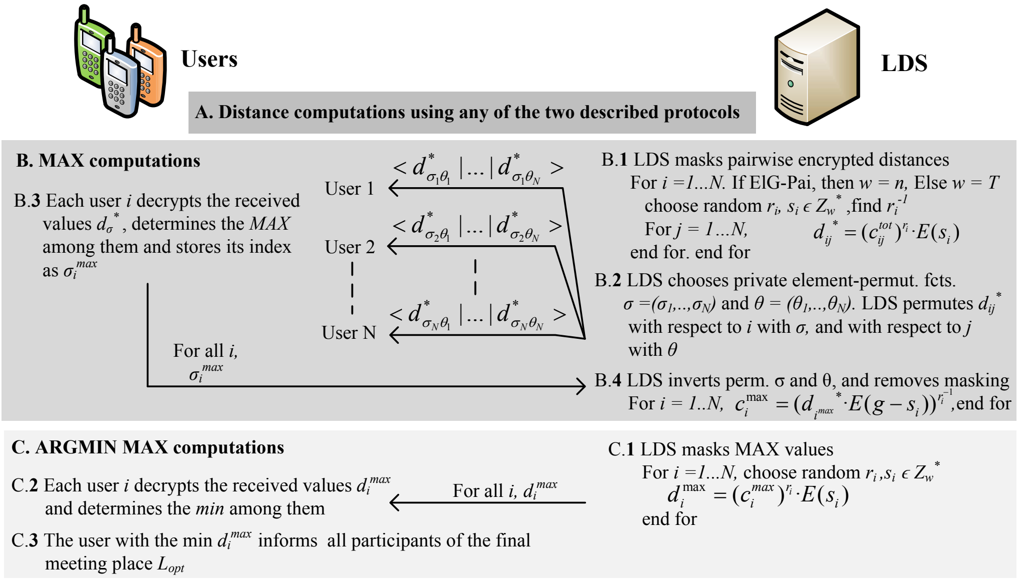

Users
LDS
A. Distance computations using any of the two described protocols
B. MAX computations
B.1 LDS masks pairwise encrypted distances
For $i = 1...N$. If ElG-Pai, then $w = n$, Else $w = T$
choose random r_i, $s_i \epsilon Z_w^*$,find r_i^{-1}
For $j = 1...N$, $d_{ij}^* = (c_{ij}^{tot})^{r_i} \cdot E(s_i)$
end for. end for
B.3 Each user i decrypts the received values d_σ^*, determines the MAX among them and stores its index as σ_i^{max}
User 1
$< d^*_{\sigma_1\theta_1} |...| d^*_{\sigma_1\theta_N} >$
User 2
$< d^*_{\sigma_2\theta_1} |...| d^*_{\sigma_2\theta_N} >$
User N
$< d^*_{\sigma_N\theta_1} |...| d^*_{\sigma_N\theta_N} >$
B.2 LDS chooses private element-permut. fcts.
$\sigma = (\sigma_1,..,\sigma_N)$ and $\theta = (\theta_1,..,\theta_N)$. LDS permutes d_{ij}^* with respect to i with σ, and with respect to j with θ
For all i, σ_i^{max}
B.4 LDS inverts perm. σ and θ, and removes masking
For $i = 1..N$, $c_i^{\max} = (d_{i^{max}}^* \cdot E(g - s_i))^{r_i^{-1}}$,end for
C. ARGMIN MAX computations
C.1 LDS masks MAX values
For $i = 1...N$, choose random $r_i, s_i \epsilon Z_w^*$
$d_i^{\max} = (c_i^{max})^{r_i} \cdot E(s_i)$
end for
C.2 Each user i decrypts the received values d_i^{max} and determines the min among them
For all i, d_i^{max}
C.3 The user with the min d_i^{max} informs all participants of the final meeting place L_{opt}

Fig. 3. Privacy-Preserving Fair Rendez-Vous Point (PPFRVP) protocol.

scheme as the one used in module (A). In other words, $E(.)$ of Figure 3 refers to the encryption of (.) using either the BGN or the Paillier encryption scheme.

MAX computations In Step B.1, the LDS needs to obliviously hide the values within the encrypted elements (i.e., the pairwise distances computed earlier), before sending them to the users, in order to avoid leaking any kind of private information such as the pairwise distance or desired locations to any user.[5] In order to obliviously mask such values, for each index i the LDS generates two random values r_i, s_i that are used to scale and shift the c_{ij}^{tot} (the encrypted square distance between L_i, L_j) for all j, obtaining d_{ij}^*. This is done in order to (i) ensure privacy of real pairwise distances, (ii) be resilient in case of collusion among users and (iii) preserve the internal order (the inequalities) among the pairwise distance from each user to all other users. Afterwards, in Step B.2 the LDS chooses two private element-permutation functions σ (for i) and θ (for j) and permutes d_{ij}^*, obtaining the permuted values $d_{\sigma_i \theta_j}^*$, where $i, j \in \{1, \ldots, N\}$. The LDS sends N such distinct elements to each user. In Step B.3, each user decrypts the received values, determines their maximum and sends the index σ_i^{max} of the maximum value to the LDS. In Step B.4 of the MAX module (B), the LDS inverts the permutation functions σ, θ and removes the masking from the received indexes corresponding to the maximum distance values.

ARGMIN MAX computations In Step C.1, the LDS masks the true maximum distances by scaling and shifting them by the same random amount, such that their order (the inequalities among them) is preserved. Then the LDS sends to each user all the masked maximum distances. In Step C.2 each user decrypts the received masked (randomly scaled and shifted) maximum values, and determines the minimum among all maxima. In Step C.3, each user knows which identifier corresponds to himself, and the user with the minimum distance sends to all other users his desired rendez-vous location in an anonymous way.

After the last step, each user receives the final fair rendez-vous location, but no other information regarding non-fair locations or distances is leaked.

6 Analytical Evaluation

6.1 Privacy Analysis

We define the privacy of a PPFRVP protocol as follows.

Definition 2. *A PPFRVP protocol A is* execution privacy-preserving *if a participating user cannot determine (with a non-negligible probability) (i) the preferred rendez-vous locations L_i (except L_{fair}), (ii) the mutual distances and (iii)*

[5] After the distance computation module (A), the LDS possesses all encrypted pairwise distances. This encryption is made with the public key of the participants and thus the LDS cannot decrypt the distances without the corresponding private key. The oblivious (and order-preserving) masking performed by the LDS at Step B.1 is used in order to hide the pairwise distances from the users themselves, as otherwise they would be able to obtain these distances and violate the privacy of the users.

coordinate relations of any user, after an execution of A. Moreover, the LDS (or any third-party) should not be able to infer any information about L_{fair}.

In our analysis, we consider two types of adversaries: Passive (honest-but-curious) and active adversaries. The passive try to learn as much information as possible from their inputs, the execution of the PPFRVP protocol and its output, without maliciously injecting or modifying data. The active adversaries, on the contrary, try on purpose to manipulate the data in order to obtain private information.

The aforementioned definition captures the privacy requirements of a single execution of a PPFRVP algorithm. By repeated interactions among a stable set of users, L_{fair} could be used to infer possible L_i of other users. The issue of *learning* from repeated interaction is inherent to any algorithm that, based on a set of private inputs, chooses one of them in particular, based on some criterion. For this reason, in this work we consider privacy for a single execution of the PPFRVP algorithm, or for repeated executions but with different sets of users.

Passive Adversary Under the passive adversary model, we have the following.

Proposition 1. *The BGN and ElGamal-Paillier based PPFRVP protocols are execution privacy-preserving.*

In simple words, Proposition 1 states that both proposed algorithms correctly compute the fair rendez-vous location, given the received inputs, and that they do not reveal any users' preferred rendez-vous locations to any other user, except the fair rendez-vous location L_{fair}. Moreover, the LDS does not learn any information about any user-preferred locations. In the Appendix we prove the proposition by considering a standard challenger-adversary game methodology that is usually employed for privacy proofs in cryptographic schemes.

Active Adversary We consider three main categories of active attacks against PPFRVP protocols, namely (i) the collusion among users and/or LDS, (ii) the fake user generation and/or replay attacks and (iii) unfair rendez-vous location.

Collusion Regardless of the protocol used or the encryption methods, in the case when users collude among themselves the published fair result (together with the additional information malicious users may get from colluders) can be used to construct exclusion zones, based on the set of equations and known parameters. An exclusion zone is a region that does not contain any location preferences, and the number of such exclusion zones increases with the number of colluders. We are currently working on quantifying this impact on our optimization and encryption methods. However, in the unlikely case of collusion between the LDS and the participants, the latter will be able to obtain other participants' preferences. In order to mitigate such a threat, the invited participants could agree on establishing a shared secret by using techniques from threshold cryptography [25]. The LDS should then collude with at least a given number of participants in order to obtain the shared secret and learn L_i.

Fake Users In case the LDS generates fake users, it would not be able to obtain the secret that is shared among the honest users and which is used to derive the secret key $K_s^{M_v}$ for each session v. This attack is more dangerous if a legitimate participant creates a fake, because the legitimate participant knows the shared secret. In this scenario, however, the LDS knows the list of meeting participants (as it computes the fair rendez-vous location) and therefore it would accept only messages digitally signed by each one of them. Here we rely on the fact that fake users will not be able to get their public keys signed by a CA. Replay attacks could be thwarted by adding and verifying an individually signed *nonce*, derived using the shared secret, in each user's meeting message.

Unfair RV The last type of active attack could lead to the determination of an unfair rendez-vous location. Maliciously modifying or untruthfully reporting the maximum masked values (Step B.3 of Figure 3) could deceive the LDS to accept the false received index as the maximum value, and therefore potentially lead to the determination of a subfair rendez-vous location. However, this is rather unlikely to happen in practice. For instance, even if in Step B.3 a user falsely reports one of his values to be the maximum when actually it is not, this would cause the algorithm to select a subfair rendez-vous location if and only if no other user selected a smaller value as the maximum distance.

6.2 Complexity Analysis

Table 1 summarizes the complexity results for our two protocols, both for the client devices and for the LDS. As it can be seen, the client complexity is in general $O(N)$, where N is the number of users. However, there is a notable exception for the BGN-based scheme; the number of exponentiation required for a single decryption is $O(\sqrt{T})$ [2], where T is the order of the plaintext domain. In Section 7 we show how this charateristic impacts the decryption performance.

Table 1. Asymptotic complexity of the proposed PPFRVP protocols, where N is the number of participants. The *Distance* protocol is the one used in the module A of Figure 3, whereas PPFRVP includes modules A,B and C.

CLIENT	**PROTOCOL**	**BGN** *(mod n)*	**ELGAMAL-PAILLIER** *(mod n^2)*	**LDS**	**BGN** *(mod n)*	**ELGAMAL-PAILLIER** *(mod n^2)*
Mult.	Distance	O(1)	O(N)	**Mult. Exp.**	O(N^2)	O(N^2)
	PPFRVP					
Exp.	Distance	O(1)	O(N)	**Bilinear mapping**	O(N^2)	-------
	PPFRVP	*O(N$\sqrt{T}$)*				
Memory	Distance	O(1)	O(N)	**Memory**	O(N^2)	O(N^2)
	PPFRVP	*O(N)*				
Comm.	Distance	O(1)	O(N)	**Comm.**	O(N)	O(N^2)
	PPFRVP	*O(N)*			*O(N^2)*	

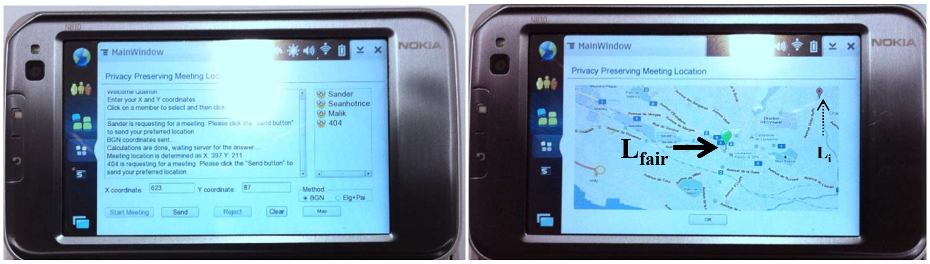
The LDS complexity for both protocols is in general $O(N^2)$, with the notable exception of BGN, where in addition to multiplications and exponentiations the schemes requires additional $O(N^2)$ bilinear mappings. These operations are required in order to support the multiplicative property of the BGN scheme.

7 Implementation Performance and User-Experience

In this section, we discuss the results of the performance measurements using implementations of the proposed algorithms on Nokia devices, and we present the related results of Phase 3 of our user-study on the prototype application.

7.1 Performance Measurements

The tests were conducted on a testbed of Nokia N810 mobile devices (ARM 400 MHz CPU, Figure 4), and the LDS on a Linux machine (2 GHz CPU, 3 GB RAM). For the elliptic curve BGN-based PPFRVP protocol, we measured the performance using both a 160-bit and a 256-bit secret key, whereas for the ElGamal-Paillier-based one we used 1024-bit secret keys. As BGN is an elliptic curve-based scheme, much shorter keys can be used compared to ElGamal and RSA. A 160-bit key in elliptic curve cryptosystems is generally believed to provide equivalent security as a 1024-bit key in RSA and ElGamal [23].



Welcome Quentin
Enter your X and Y coordinates
Click on a member to select and then click

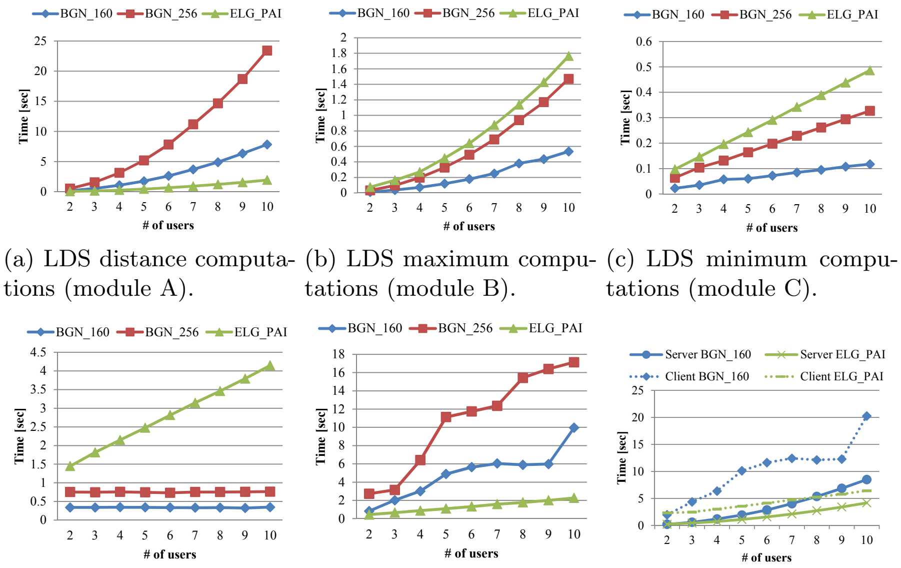
Sander is requesting for a meeting. Please click the "Send button" to send your preferred location
BGN coordinates sent...
Calculations are done, waiting server for the answer...
Meeting location is determined as X: 397 Y: 211
404 is requesting for a meeting. Please click the "Send button" to send your preferred location

X coordinate: 623
Y coordinate: 87
Method: ● BGN ○ Elg+Pai

Fig. 4. Prototype PPFRVP application running on a Nokia N810 mobile device. The image on the left is the main window, where users add the desired meeting participants. The image on the right is the map that shows the fair rendez-vous location (green pin) and the user-desired rendez-vous location (red pin).

LDS performance Figure 5(a), 5(b) and 5(c) show the computation time required by the LDS. We can see that such time increases with the number of users, and that the ElGamal-Paillier algorithm is the most efficient across all computations, requiring 4 seconds to execute the PPFRVP protocol with 10 participants. The two BGN-based algorithms are less efficient, but are still practical enough (9 seconds). The CPU-intensive bilinear mappings in BGN are certainly one important reason for such delays.

Client performance Figure 5(d) and 5(e) show the different computation times on the Nokia N810 mobile device. As it can be seen, thanks to the efficient use of the homomorphic properties of our BGN-based algorithm, this protocol



(a) LDS distance computations (module A).

(b) LDS maximum computations (module B).

(c) LDS minimum computations (module C).

(d) Client distance computations (module A).

(e) Client max/argmin computations (module B/C).

(f) Total client and LDS run times (modules A+B+C).

Fig. 5. Performance measurements.

is the most efficient for the distance computations, requiring only 0.3 seconds, independently of the number of users. On the contrary, the alternative protocol needs 4 seconds with 10 participants. However, the subsequent phases reverse such results, as the BGN protocol makes intensive use of bilinear mappings.

Overall, we can see that the ElGamal-Paillier protocol has a better performance than the BGN-based one, both on the client and on the LDS. Nevertheless, both schemes are practical enough and have acceptable time requirements in order to be implemented on current generations of mobile devices.

7.2 User-Experience

We present the PPFRVP application-related results of our user study introduced in Section 2. After using our application, all participants tend to agree (34%) or agree (66%) that our application was easy to use, and that they could quickly compute the task (97%). More than 71% appreciated that their preferred rendez-vous point was not revealed to other participants, and only 8% do not care about the privacy of their rendez-vous location preference. 26% of the respondents were able to identify to whom the FRVP location belonged to, which is expected. The users run our application in groups of 5 during the experimentation, and therefore there was always one person out of five that knew that the FRVP location was his preferred location.

From a software developer standpoint, this means that both ease of use and privacy need to be taken into account from the beginning of the application development process. In particular, the privacy mechanisms should be implemented in a way that does not significantly affect the usability or performance. The acceptance of LSBS applications is highly influenced by the availability of effective and intuitive privacy features.

8 Related Work

Hereafter, we present some works in the literature that address, without protecting privacy, strategies to determine the fair rendez-vous location. To the best of our knowledge, this is the first work to address such a problem in a privacy-preserving way.

Santos and Vaughn [24] present a survey of existing literature on meeting-location algorithms, and propose a more comprehensive solution for such a problem. Although considering aspects such as user preferences and constraints, their work (or the surveyed papers) does not address any security or privacy issues. Similarly, Berger et. al [1] propose an efficient meeting-location algorithm that considers the time in-between two consecutive meetings. However, all private information about users is public.

In the domain of Secure Multiparty Computation (SMC), several authors have addressed privacy issues related to the computation of the distance between two routes [10] or points [14,26]. Frikken and Atallah [10] propose SMC protocols for securely computing the distance between a point and a line segment, the distance between two moving points and the distance between two line segments. Zhong et al. [28] design and implement three distributed privacy-preserving protocols for nearby friend discovery, and they show how to cryptographically compute the distance between a pair of users. However, due to the fully distributed nature of the aforementioned approaches, the computational and communication complexities increase significantly with the size of the participants and inputs. Moreover, all parties involved in the computations need to be online and synchronized.

As both our protocols are centralized, most of the cryptographic operations are performed by the LDS and not by the mobile devices. Additionally, the proposed solutions do not require all users to be online at the same time, and they necessitate only minimal synchronization among the mobile devices.

9 Conclusion and Future Work

In this work, we address the problem of privacy in LSBS by providing practical and effective solutions to one such popular and relevant service. The PPFRVP problem captures the essential computational and privacy building blocks present in any LSBS offered on mobile devices. We designed, implemented on real mobile devices and evaluated the performance of our privacy-preserving protocols for the fair rendez-vous problem. Our solutions are effective in terms of privacy, have acceptable performance, and do not create additional overhead for the users. Moreover, our user-study showed that the proposed privacy features are crucial for the adoption of any such application, which reinforces the need for further exploration in privacy of LSB services. To the best of our knowledge, this is the first such effort in this direction.

Acknowledgment

We would like to thank Mathias Humbert for helping improving the quality of this work, as well as the Nokia Research Center for supporting this project.

References

1. F. Berger, R. Klein, D. Nussbaum, J.-R. Sack, and J. Yi. A meeting scheduling problem respecting time and space. *GeoInformatica*, 2009.
2. D. Boneh, E.-J. Goh, and K. Nissim. Evaluating 2-dnf formulas on ciphertexts. In *Theory of Cryptography*. 2005.
3. C. Cachin and R. Strobl. Asynchronous group key exchange with failures. In *ACM PODC '04*, 2004.
4. C.-H. O. Chen, C.-W. Chen, C. Kuo, Y.-H. Lai, J. M. McCune, A. Studer, A. Perrig, B.-Y. Yang, and T.-C. Wu. Gangs: Gather, authenticate 'n group securely. In *ACM MobiCom '08*, 2008.
5. M. Chignell, A. Quan-Haase, and J. Gwizdka. The privacy attitudes questionnaire (paq): initial development and validation. In *Human Factors and Ergonomics Society Annual Meeting Proceedings*, 2003.
6. T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31, 1985.
7. Facebook Deals. http://www.facebook.com/deals/.
8. Facebook Statistics. http://www.facebook.com/press/info.php?statistics.
9. Foursquare for Business. http://foursquare.com/business/, Last visited 04.02.2011.
10. K. B. Frikken and M. J. Atallah. Privacy preserving route planning. In *WPES '04*, 2004.
11. O. Goldreich. *Foundations of cryptography: Basic applications.* Cambridge University Press, 2004.
12. J. Krumm. A survey of computational location privacy. *Personal and Ubiquitous Computing*, 13(6):391–399, 2009.
13. J. Lewis. IBM computer usability satisfaction questionnaires: psychometric evaluation and instructions for use. *International Journal of Human-Computer Interaction*, 7, 1995.
14. S.-D. Li and Y.-Q. Dai. Secure two-party computational geometry. *Journal of Computer Science and Technology*, 20, 2005.
15. Y.-H. Lin, A. Studer, H.-C. Hsiao, J. M. McCune, K.-H. Wang, M. Krohn, P.-L. Lin, A. Perrig, H.-M. Sun, and B.-Y. Yang. Spate: Small-group PKI-less authenticated trust establishment. In *MobiSys '09*, 2009.
16. MeetWays. http://www.meetways.com/.
17. Mezzoman. http://www.mezzoman.com/.

18. Microsoft survey on LBS. http://go.microsoft.com/?linkid=9758039, 2011.
19. Orange Taxi sharing app. http://event.orange.com/default/EN/all/mondial_auto_en/taxi_partage.htm.
20. P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. *EUROCRYPT '99*, 1999.
21. Please Rob Me. http://pleaserobme.com/.
22. R. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21, 1978.
23. M. Robshaw and Y. Yin. Elliptic curve cryptosystems. *An RSA Laboratories Technical Note*, 1997.
24. P. Santos and H. Vaughn. Where shall we meet? Proposing optimal locations for meetings. MapISNet'07, 2007.
25. B. Schoenmakers. A simple publicly verifiable secret sharing scheme and its application to electronic voting. In *CRYPTO '99*, 1999.
26. A. Solanas and A. Martínez-Ballesté. Privacy protection in location-based services through a public-key privacy homomorphism. In *Public Key Infrastructure*. 2007.
27. UTM coordinate system. https://www.e-education.psu.edu/natureofgeoinfo/c2_p21.html.
28. G. Zhong, I. Goldberg, and U. Hengartner. Louis, lester and pierre: Three protocols for location privacy. In *Privacy Enhancing Technologies*, pages 62–76, 2007.

Proof of Proposition 1

We express the privacy in terms of three probabilistic advantages that an adversary u_a (a user or a third-party) gains after an execution of a PPFRVP algorithm A. First, we measure the *identifiability advantage*, which is the probabilistic advantage of u_a in correctly guessing the preferred location L_i of any user $u_i \neq u_a$. We denote it as $Adv_a^{IDT}(A)$. Second, the *distance-linkability advantage* is the probabilistic advantage of u_a in correctly guessing whether the distance d_{ij} between any two users $u_i \neq u_j$ is greater than a given parameter s, without necessarily knowing any users' preferred locations L_i, L_j. We denote it as Adv_a^{d-LNK}. Finally, the *coordinate-linkability advantage* is the probabilistic advantage of u_a in correctly guessing whether a given coordinate x_i (or y_i) of a user u_i is greater than the corresponding coordinate(s) of another user $u_j \neq u_i$, i.e., x_j (or y_j), without necessarily knowing any users' preferred locations L_i, L_j. We denote it as Adv_a^{c-LNK}.

Challenger-Adversary Games

We describe hereafter the challenger-adversary game for the identifiability advantage $Adv_a^{IDT}(A)$ of any user u_a, $a \in \{1, \dots, N\}$, after executing the PPFRVP algorithm A:

1. Initialization: Challenger privately collects $\mathbb{L} = \{L_i\}_{i=1}^N$, where $L_i = (x_i, y_i)$ is the preferred rendez-vous location of user u_i, and $f(L_i)$, $\forall i \in \{1, \dots, N\}$.
2. PPFRVP algorithm: Challenger executes the PPFRVP algorithm A and computes $f(L_{fair}) = g(f(L_1), \dots, f(L_N))$. It then sends $f(L_{fair})$ to each user $u_i, \forall i \in \{1, \dots, N\}$.
3. Challenger randomly chooses a user u_a, $a \in \{1, \dots, N\}$, as the adversary.

4. u_a chooses $u_j \neq u_a$ and sends j to the challenger.
5. Challenge: Challenger chooses a random $k \in \{1, \ldots, N\}$ and sends L_k to the adversary. The challenge is to correctly guess whether $L_k = L_j$.
6. The adversary sends L_j^* to the challenger. If the adversary thinks that L_k is the preferred rendez-vous location of user u_j, i.e., if $L_k = L_j$ then the adversary sets $L_j^* = 1$. If the adversary thinks that L_k is not the preferred rendez-vous location of user u_j, then he sets $L_j^* = 0$. If $L_j^* = L_k$ the adversary wins the game, otherwise he loses.

The challenger-adversary game for the distance-linkability advantage $Adv_a^{d-LNK}(A)$ of any user u_a is defined as follows.

1. Initialization: Challenger privately collects $\mathbb{L} = \{L_i\}_{i=1}^N$, where $L_i = (x_i, y_i)$ is the preferred rendez-vous location of user u_i, and $f(L_i)$, $\forall i \in \{1, \ldots, N\}$.
2. PPFRVP algorithm: Challenger executes the PPFRVP algorithm A and computes $f(L_{fair}) = g(f(L_1), \ldots, f(L_N))$. It then sends $f(L_{fair})$ to each user $u_i, \forall i \in \{1, \ldots, N\}$.
3. Challenger randomly chooses a user u_a, $a \in \{1, \ldots, N\}$, as the adversary.
4. u_a chooses $u_j, u_k \neq u_a$ and sends (j, k) to the challenger.
5. Challenge: Challenger computes a value s, such as the average Euclidian distance $d = \sum_{n=1}^{N-1} \sum_{m=n+1}^{N} d_{nm}/(2N(N-1))$ between any two users $u_n \neq u_m$, and sends (j, k, s) to the adversary. The challenge is to correctly guess whether $d_{jk} < s$.
6. The adversary sends d^* to the challenger. If the adversary thinks that $d_{jk} < s$ then he sets $d^* = 1$, otherwise $d^* = 0$. The adversary wins the game if: (i) $d^* = 1 \wedge d_{jk} < s$ or (ii) $d^* = 0 \wedge d_{jk} \geq s$. Otherwise, the adversary loses.

The challenger-adversary game for the coordinate-linkability advantage $Adv_a^{c-LNK}(A)$ of any user u_a is defined as follows.

1. Initialization: Challenger privately collects $\mathbb{L} = \{L_i\}_{i=1}^N$, where $L_i = (x_i, y_i)$ is the preferred rendez-vous location of user u_i, and $f(L_i)$, $\forall i \in \{1, \ldots, N\}$.
2. PPFRVP algorithm: Challenger executes the PPFRVP algorithm A and computes $f(L_{fair}) = g(f(L_1), \ldots, f(L_N))$. It then sends $f(L_{fair})$ to each user $u_i, \forall i \in \{1, \ldots, N\}$.
3. Challenger randomly chooses a user u_a, $a \in \{1, \ldots, N\}$, as the adversary.
4. u_a chooses $u_j, u_k \neq u_i$ and sends (j, k) to the challenger.
5. Challenge: Challenger chooses a coordinate axis $c \in \{x, y\}$ and sends (j, k, c) to the adversary. The challenge is to correctly guess whether $c_j < c_k$.
6. The adversary sends c^* to the challenger. If the adversary thinks that $c_j < c_k$ then he sets $c^* = 1$, otherwise $c^* = 0$. The adversary wins the game if: (i) $c^* = 1 \wedge c_j < c_k$ or (ii) $c^* = 0 \wedge c_j \geq c_k$. Otherwise, the adversary loses.

For the third-party (LDS) adversary, the game definitions are similar to those of the user adversary. However, as mentioned, the third-party shall not be able to infer (with a non-negligible probability) the L_{fair}, in addition to any L_i.

Proofs

Correctness Given the encrypted set of user-preferred locations $f(L_1), \dots, f(L_N)$, the proposed PPFRVP algorithms compute the pairwise distance between each pair of users d_{ij}, $\forall i, j \in \{1, \dots, N\}$, according to the schemes of the respective distance computation algorithms. Following the sequence of steps for such computation, one can easily verify that the ElGamal-Paillier based distance computation algorithm computes

$$\begin{aligned} Pai(d_{ij}^2) &= Pai(x_i^2) \cdot Pai(-2x_i x_j) \cdot Pai(y_j^2) \cdot Pai(y_i^2) \cdot Pai(-2y_i y_j) \cdot Pai(y_j^2) \\ &= Pai(x_i^2 - 2x_i x_j + x_j^2 + y_i^2 - 2y_i y_j + y_j^2) \end{aligned}$$

which is the same result that is achieved by the BGN-based distance algorithm.

After the pairwise distance computations, the PPFRVP algorithm computes the masking of these pairwise distances by scaling and shifting operations. The scaling operation is achieved by exponentiating the encrypted element to the power of r_i, where $r_i \in \mathbb{Z}_w^*$ is a random integer and r_i^{-1} is its multiplicative inverse. The shifting operation is done by multiplying the encrypted element with the encryption (using the public key of the users) of another random integer s_i privately chosen by the LDS. These two algebraic operations mask the values d_{ij}^2 (within the encrypted elements), such that the true d_{ij}^2 are hidden from the users. Nevertheless, thanks to the homomorphic properties of the encryption schemes, the LDS is still able to remove the masking (after the users have identified the maximum value) and correctly re-mask all maxima, such that each user is able to correctly find the minimum of all maxima.

In the end, each user is able to determine L_{fair} where $fair = \text{argmin}_i \max_j d_{ij}^2$ from the outputs of the PPFRVP algorithm, and therefore the PPFRVP algorithms are correct.

User Identifiability Advantage Using the previously defined challenger-adversary games, we define the identifiability advantage of an attacker u_a as

$$Adv_a^{IDT}(A) = \left| Pr[L_j^* = L_k] - 1/N \right|$$

where $Pr[L_j^* = L_k]$ is the probability of user u_a winning the game by correctly answering the challenge, computed over the coin tosses of the challenger, and $R(\nVdash/N$ is the probability of a random guess over the N possible user-preferred locations. Now, at the end of the PPFRVP protocol, the attacker knows L_{fair} and its own preferred location $L_a = (x_a, y_a) \in \mathbb{N}^2$. Assuming that all users other than u_a have executed the protocol correctly, u_a does not know any preferred location L_i, for $i \neq a$. Hence, the probability $Pr[L_j^* = L_k]$ of him making a correct guess j^* about the preferred rendez-vous location L_k of user u_k equals the probability of a random guess, which in this case is $1/N - 1$. Thus, the identifiability advantage of the attacker u_a is negligible.

User Distance-Linkability Advantage The distance-linkability of an attacker u_a is defined as

$$Adv_a^{d-LNK}(A) = |Pr[(d^* = 1] \wedge d_{jk} < s) \vee (d^* = 0 \wedge d_{jk} \geq s)] - \frac{1}{2}|$$

where $Pr[.]$ is the probability of the adversary u_a winning the game by correctly answering the challenge, computed over the coin tosses of the challenger, d^* is the guess of the adversary, d_{jk} is the distance between L_j, L_k and s is a parameter chosen by the challenger. In this case, the attacker has to guess whether the distance d_{jk} between two users j, k is greater than s, and clearly if he at some point in the protocol obtains any pairwise distance d_{jk}, his advantage is non-negligible. However, as explained in the correctness proof, each user gets to know only N masked (and anonymized) values of the squares of pairwise distances. Thus, the attacker wants to solve the following system of linear equations:

$$\begin{cases} C_{\sigma_a,\theta_1} & = r_a \cdot d^2_{\sigma_1,\theta_1} + s_a \\ & \vdots \\ C_{\sigma_a,\theta_N} & = r_a \cdot d^2_{\sigma_1,\theta_N} + s_a \end{cases}$$

where C_{ij} is the received masked value of the pairwise distances and r_a, s_a are random integers privately chosen by the LDS. Hence, possessing only the knowledge of his own preferred location and the fair fair rendez-vous location, the attacker cannot uniquely solve this system of equation, because it is still underdetermined. Therefore, the distance-linkability advantage of u_a is negligible.

User Coordinate-Linkability Advantage In order to have non-negligible coordinate-linkability advantage, an attacker u_a needs to have additional information regarding at least one of the two coordinates of any other user's preferred rendez-vous location. As discussed in the identifiability and distance linkability advantage proofs, after a private execution of the PPFRVP algorithm A, the attacker does not gain any additional information about any other user's locations. Therefore, not knowing any other user's coordinate, an attacker does not gain any probabilistic advantage on correctly guessing the relationship between their spatial coordinates. Hence, the coordinate-linkability advantage is negligible.

Third-party Advantages All elements that are received and processed by the LDS have previously been encrypted by the users with their common public key. In order to efficiently decrypt such elements, the LDS would need to have access to the private key that has been generated with the public key used for the encryption. As explained in Section 3, in most practical settings, where service providers have a commercial interest in providing a faithful service to their customers, the LDS would not try to maliciously obtain the secret key. Therefore, all the LDS does in the PPFRVP algorithm is to obliviously execute algebraic operation on encrypted elements, without knowing the values within the encrypted elements. Hence, the PPFRVP algorithms do not disclose any information the a third-party, such as the LDS, during or after its execution.