# Privacy-preserving Verifiable Proximity Test for Location-based Services

Gaoqiang Zhuo*, Qi Jia*, Linke Guo*, Ming Li†, and Yuguang Fang‡
*Department of Electrical and Computer Engineering, Binghamton University,
State University of New York, Binghamton, NY 13902, USA
†Department of Computer Science and Engineering, University of Nevada, Reno, NV 89557, USA
‡Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611, USA
Email: {gzhuo1, qjia1, lguo}@binghamton.edu, mingli@unr.edu, fang@ece.ufl.edu

***Abstract*—The prevalence of smartphones with geo-positioning functionalities gives rise to a variety of location-based services (LBSs). Proximity test, an important branch of location-based services, enables the LBS users to determine whether they are in a close proximity with their friends, which can be extended to numerous applications in location-based mobile social networks. Unfortunately, serious security and privacy issues may occur in the current solutions to proximity test. On the one hand, users' private location information is usually revealed to the LBS server and other users, which may lead to physical attacks to users. On the other hand, the correctness of proximity test computation results from LBS server cannot be verified in the existing schemes and thus the creditability of LBS is greatly reduced. Besides, privacy should be defined by user him/herself, not the LBS server. In this paper, we propose a privacy-preserving verifiable proximity test for location-based services. Our scheme enables LBS users to verify the correctness of proximity test results from LBS server without revealing their location information. We show the security, efficiency, and feasibility of our proposed scheme through detailed performance evaluation.**

***Index Terms*—Proximity-test, Privacy, Correctness Verification, LBSs**

## I. Introduction

Location-based services (LBSs) have recently gained a lot of attention in daily life, which can provide various kinds of services, like finding the nearest restaurant and sharing current location with friends. As a major branch of LBSs, proximity test enables a LBS requesting user to test if there are some other LBS users within a certain proximity [1]–[3]. There are two cases of proximity test results. In one case, Alice only cares about her friends' locations. Alice will be notified as soon as her friend, e.g., Carol, is within her physical proximity or even can be shown in the map the real-time location of Carol, e.g., *Google Latitude, iPoki* and *Fire Eagle* [1], [4]. In the other case, Alice focuses on finding strangers who are in her proximity. Alice will be informed that a stranger, e.g., Bob, is within her proximity when Bob responds to her. Proximity-based mobile social networks (PMSNs) [5], [6] can be an extended service based on the latter case. In PMSNs, Alice first launches a proximity test request to the LBS server when she wants to make new friends. If a stranger Bob is interested in making friends with Alice, he responds to Alice's request via the LBS server. Alice may further build a secure communication channel with Bob if she is also interested in Bob [7].

No matter how promising proximity test looks, it will not be widely used unless the following issues are well addressed. On the one hand, LBS users' smartphones are lacking of computation capabilities when the number of LBS responding users increases, then LBS users may need to ask LBS server for help. On the other hand, LBS users do not want to reveal their exact location information to others, because directly revealing real location information may open them up to serious physical attacks [8]–[11]. In the current situation, LBS server is responsible for performing the intensive proximity test computations for requesting users [12]. However, this approach may have serious security and privacy breaches. First, current schemes pre-define the location privacy for the LBS users, i.e., users should encrypt their exact locations into ciphertexts or should mask their locations with a fixed size of area [13]. However, pre-defined location privacy is not secure enough. For example, Alice and Bob are LBS users and they are at the same location. If Alice launches the proximity test with the encrypted location and Bob responds with his encrypted location, then either the LBS server or Alice will know Bob is at the same location with side information. Besides, since every user has a different definition on privacy, he/she should have the privilege to define location privacy by him/herself. Second, the outsourced location information may be manipulated at LBS server. Under situations like software or hardware error and attacks from opponents the computation results will not be reliable [14]–[17]. As far as we know, no current proximity test scheme provides a mechanism for LBS users to verify the correctness of the computation results. A LBS requesting user should be able to detect the error from the computation results. To address the above issues, we propose a privacy-preserving verifiable proximity test scheme for location-based services. In our scheme, a user is able to define his/her own privacy level, namely, the discoverable range (DR), which represents the range that a user can be discovered and is varied from user to user. In addition to this, discoverable range also conveys a user's willingness to make friends with others and the larger a discoverable range is, the stronger the will is. Our scheme not only preserves the privacy of LBS users' location during the proximity test computation, but also enables LBS users to verify the correctness of the outsourced computation results.

**Related Works :**

**Private Proximity Test :** Private proximity test has been well studied in the past years [1], [2], [7]. In [1], a private and flexible proximity detection scheme in mobile social networks has been proposed. This scheme can notify a user if her friends enter the *vicinity region*. In [2], they present three protocols that can achieve private proximity test. In this paper, *location*

This work was partially supported by US National Science Foundation under grant CNS-1423165.

*tags* is novelly exploited to avoid location cheating between users. However, these protocols will either introduce privacy leak or involve too much computation and communication overhead. Yao *et al.* [7] use more sources of location tags to achieve private proximity test. In this paper, secure communications can also be established among strangers with pre-shared secret. Based on [2], Lin *et al.* [3] reduce the threshold set intersection on location tags to equality testing by using a de-duplication technique. However, none of these works allow LBS users to define their own privacy and the computation results of proximity test cannot be verified by requesting users either.

**Correctness Verification :** Lu *et al.* [18] use the verifiable polynomial computation to keep LBS provider's functions private and prevent the provider from cheating in computation. The accumulation tree was first introduced in [19], with which we can verify if the value of a leaf node is has been modified or not. In [20], the accumulation tree is used to verify the correctness of the original sets, which are used to compute the intersection.

**Our Contributions :** We list our contributions as follows.

- With our scheme, a proximity test requesting user can find the number of responding users in a privacy-preserving way.
- The LBS server performs the proximity test computation for the users, while learning nothing about users' location information from the results.
- Our scheme enables users to verify the correctness of proximity test results from the LBS server.

The remainder of this paper is organized as follows. Section II introduces preliminaries, assumptions and problem formulation. Section III describes the system model, security model and design objectives. The proposed scheme is presented in detail in Section IV, followed by the protocol evaluation in Section V. Finally, Section VI concludes the paper.

## II. Preliminaries and Problem Formulation

### A. Preliminaries

*1) Bilinear Pairing:* A bilinear pairing is a map $e : G \times G \rightarrow G_T$, where $G$ and $G_T$ are two multiplicative cyclic groups of the same prime order $p$ and $G$ is generated by $g$. The pairing $e$ has the following properties [21] [22]:

- **Bilinearity :** $e(u^a, v^b) = e(u, v)^{ab}$ for all $u, v \in G$ and random numbers $a, b \in Z_p^*$;
- **Computability :** For all $u,\ v\ \in\ G,\ e(u, v)$ can be computed efficiently;
- **Non-degeneracy :** For $g \in G$, $e(g, g) \neq 1$.

Here, we assume the hardness of DLP [23], CDH [24] and bilinear q-strong DH [20].

*2) Accumulation Tree [20] :* An accumulation tree $T$ has $\lceil 1/\epsilon \rceil$ levels and $n$ leaves, where $0 < \epsilon < 1$ is a constant selected at setup. The degree of each internal node is $n^\epsilon$. Let $s$ be a random number from $Z_p^*$, for a set $A_i \subseteq Z_p - \{s\}$, the accumulation value $acc(A_i)$ of $A_i$ is defined as $acc(A_i) = g^{\prod_{a \in A_i}(a+s)}$ [20]. Each leaf node contains a *digest* $d(v_i)$ of the accumulation value $acc(A_i)$ of a set $A_i$. Each internal node contains the *digest* of all its children nodes' *digests* [20]. For example, if $\epsilon = 1/3$ and there are 9 sets $A_1, A_2, ..., A_9$, the accumulation tree is shown in Fig. 1.

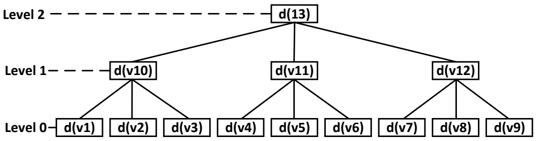
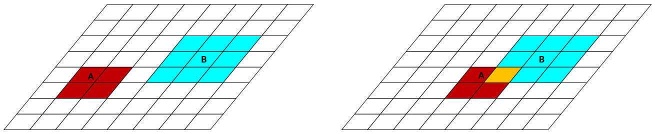


Fig. 1. Accumulation Tree for 9 sets and $\epsilon = 3$

### B. Problem Formulation

We briefly introduce our intuition for our proximity test. The user in our system can define his/her own *discoverable range* (DR), which is an extended area based on his/her current location. The size of DR reflects a user's willingness to communicate with others. In proximity test, if two users, Alice and Bob, have DRs as shown in Fig. 2(a). It is obvious that there is no overlap between these two DRs, then Alice and Bob will not consider each other as a potential friend. However, if their DRs overlap with each other, then Alice/Bob knows that Bob/Alice is also a person who likes to make new friends, as shown in Fig. 2(b). If Alice finds that there are more than one user want to communicate with her, she can choose the one who has larger overlap with her. Larger overlaps may represent stronger communication willingness.



(a) No Overlapping

(b) Overlapping

Fig. 2. Alice's and Bob's Discoverable Ranges

## III. System Model

In this section, we present our system model, security model and design objectives.

### A. System Model

There are three entities in our system, the trust authority (TA), LBS Server and mobile users (MU), as shown in Fig. 3. All the entities are within a service area. The mobile users (MUs) can roam inside the service area.

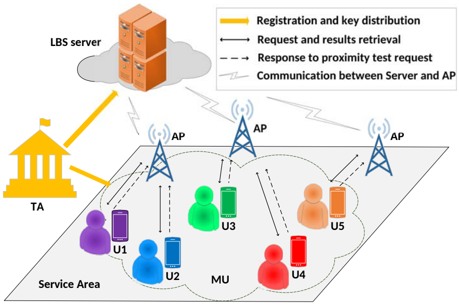

LBS server
Registration and key distribution
Request and results retrieval
Response to proximity test request
Communication between Server and AP
AP
AP
AP
TA
U1
U2
U3
U4
U5
MU
Service Area

Fig. 3. System Model

- Trust Authority (TA): TA is responsible for initializing the whole system which includes registering users and LBS server, generating public parameters, distributing keys to users and maintaining the system.
- Location-based service server (LBS server): The LBS server has enormous storage and computation capability. It will receive proximity-test request from users and broadcast the request to other users via access points (APs). The APs only act as relays among LBS server and users. After getting responses from other users, server

will perform the proximity test computation and return the computation results to the requesting user together with certain proof information.

- LBS Mobile Users (MU): Mobile users are represented by users with mobile devices in our system. Users may move in the service area randomly. When a user wants to know if there are some users who want to communicate with her and are close to her, she will send a proximity-test request to the LBS server via AP and wait for the results. When a user gets the results from LBS server, she can know the size of the overlap with responding users. She can also verify the correctness of the results.

### B. Security Model

The TA is fully trusted and it will not be compromised by any attacker. Both users and LBS server are honest but curious. The LBS Server will follow the protocol honestly but it will also try to know user's location privacy and proximity privacy. All mobile users will not allow others to know their exact locations. The proximity-test requesting user will be able to know the location of the overlapping area but the exact location information of the responding users will be revealed. Meanwhile, collusion attacks between users or between users and LBS server are not considered in our paper.

### C. Design Objectives

We mainly have three design objectives. First, only the requesting user can know the size of the overlapping of discoverable ranges, which means neither the LBS Server nor the responding users can learn any information about this. Second, users' private locations should not be revealed. Third, the requesting user should be able to verify the correctness of the proximity test results.

## IV. Our Proposed Scheme

In this section, we will explain our privacy-preserving verifiable proximity test scheme for location-based services in detail.

### A. Overview

In our scheme, the service area is divided into small square cells and each cell has the same size $L \times L$, as shown in Fig. 4. Each cell has a number. Supposing Alice wants to launch the proximity-test request, at the beginning she needs to generate her discoverable range. The discoverable range is generated through her smartphone, which can pin her location in the service area. After knowing her location on the service area, Alice can select the cells which compose her DR. The DR is then represented by the center points of the corresponding cells. In Fig. 4 Alice's location is in $a_1$ and DR is represented by the DR Set (DRS) $S_A := \{a_1, a_2, ..., a_4\}$. Similarly, Bob, located in $b_5$, chooses his DR Set $S_B := \{b_1, b_2, ..., b_9\}$. Different users may choose DRs with different sizes and shapes. However, to simplify the explanation, we assume that all DRs are squares but may differ in size, e.g. $(|S_B| > |S_A|)$. After generating the $S_A$, Alice encrypts every element in $S_A$ using the ElGamal encryption [2] and obtains $C_A$. She sends $C_A$ together with the ElGamal public key $pk_A$ to the LBS Server as a request. Then Alice will wait for the computation results from LBS Server. LBS Server broadcasts Alice's public encryption key $pk_A$ to all users within the service area via the APs and informing all users to give a response if they want to. If Bob is interested in communicating with Alice, he will encrypt his DR Set $S_B$ with Alice's $pk_A$ and output $C_B$. Bob also needs to generate an authenticated data structure $auth(S_B)$ based on his DRS. Then he sends $C_B$ and $auth(S_B)$ to LBS server. After receiving all the responses for Alice's request, LBS server performs computation based on $C_A$ and $C_B$ and outputs $R_{AB}$. Here, to simplify the explanation, we assume only Bob gives a response. LBS server also needs to generate the proof information $Pr_B$ for every element in $S_B$ based on Bob's authenticated data structure $auth(S_B)$ and outputs $Pr_B$. LBS server puts $R_{AB}$ and $Pr_B$ together and sends them back to Alice. Alice obtains the intersection of $S_A$ and $S_B$ based on $R_{AB}$ and verifies the correctness of the computation results by using the proof information $Pr_B$.

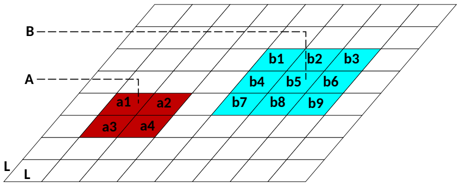


Fig. 4. Set Representation of Discoverable Ranges

### B. System Setup

In this phase, TA will generate the necessary parameters for the system. Besides, TA will register users and LBS Server into the system. We present the two steps as follows.

*1) General Setup:* Given the security parameter $\xi$, TA generates the bilinear pairing parameters $(g, p, G, G_T, e)$. TA also generates a field $Z_p$ of order $p$ and a master key $mk := s \in Z_p$. TA computes public parameters $g^s, g^{s^2}, ..., g^{s^K}$, which will be used by users to build the accumulation tree and $K$ is the total number of cells. A hash function $h(\cdot) : G \rightarrow Z_p$ is defined. TA publishes the public key $pk := (g, n_0, G, G_T, e, Z_p, h(\cdot), g^s, g^{s^2}, ..., g^{s^K})$ and keeps the master key $mk := s$.

*2) User and LBS server Registration:* Assume there are $N$ users in the system, $U_1, U_2, ..., U_N$. TA assigns each user $U_i$, $i = 1, 2, ..., N$, a key pair $(pk_i, sk_i)$, where $sk_i = x_i, x_i \in_R Z_p$ and $pk_i = g^{x_i}$. TA assigns LBS server a private key $sk_s = x_s, x_s \in_R Z_p$ and a public key $pk_s = g^{x_s}$.

### C. Request and Computation

Assuming Alice wants to launch the proximity-test request, first she uses her smartphone to generate the DRS $S_A := \{a_1, a_2, ..., a_m\}$, $m \in Z_p$ is the number of cells in her DRS. If Alice is $U_1$ then $pk_1 = g^{x_1}$ and $sk_1 = x_1$. Alice encrypts $S_A$ using the ElGamal encryption [2] with her public key $pk_1$ and random numbers $r_{i'} \in Z_{n_0}, i' = 1, 2, ..., m$, and gets the ciphertext $C_A$ as follows,

$$C_A := \{C_{a_1}, C_{a_2}, ..., C_{a_m}\}$$
$$C_{a_{i'}} = (g^{r_{i'}}, pk_1^{a_{i'}+r_{i'}}), i' = 1, 2, ..., m.$$

Let $l_0$ represents the string "Proximity-test request". Alice signs $l_0$ using her private key $sk_1$ and gets $\{l_0\}_{sk_1}$. Then, Alice sends $\{C_A, \{l_0\}_{sk_1}\}$ as the proximity-test request to the LBS Server. After receiving the proximity-test request from Alice, Server broadcasts $\{l_0\}_{sk_1}$ to all users within the service area informing them that Alice has launched a proximity-test request. Assuming Bob is interested in communicating with

Alice, then he wants Alice to check if his discoverable range has overlap with Alice's. Because if their DRs overlap, Alice might choose Bob as the one she will contact. Then Bob uses his smartphone to extract his DRS $S_B := \{b_1, b_2, ..., b_n\}$, $n \in Z_p$ is the number of cells in his DR. Bob encrypts his DRS $S_B$ using also the ElGamal encryption scheme [2] with Alice's public key $pk_1$ and random numbers $t_{j'} \in Z_p$, and gets the ciphertext $C_B$ as follows,

$$C_B := \{C_{b_1}, C_{b_2}, ..., C_{b_n}\}$$
$$C_{b_{j'}} = (g^{t_{j'}}, pk_1^{t_{j'}-b_{j'}}), j' = 1, 2, ..., n.$$

Bob also needs to compute the authenticated data structure $auth(S_B)$ for his DR set $S_B$ [20]. First, Bob computes the accumulation value $acc(b_{j'}) = g^{b_{j'}+s}$ for each element $b_{j'}$ in $S_B$. Second, Bob chooses a $\epsilon$, where $0 < \epsilon < 1$, and generates an accumulation tree $T$ with $l = \lceil 1/\epsilon \rceil$ levels and $n$ leaves, numbered $1, 2, ..., n$, where $n$ is the number of elements in $S_B$. Because $T$ has a *constant height*, the degree of any internal node is $m^\epsilon$. Third, Bob recursively computes the *digest* $d(v)$ of node $v$ as follows. If $v$ is a leaf node representing element $b_{j'}$, where $j' = 1, 2, ..., n$, $d(v) = acc(b_{j'})^{j'+s}$. Here, Bob accumulates the index $j'$ of element $b_{j'}$ into the *digest*, which can be used to prove that $acc(b_{j'})$ refers to $b_{j'}$. If $v$ is an internal node, then $d(v) = g^{\prod_{w \in Ch(v)}(h(d(w))+s)}$, where $Ch(v)$ represents the set of children of node $v$. Finally, Bob obtains the authenticated data structure $auth(S_B) = \{acc(b_{j'}), T, \{d(v)\}_{v \in T}\}, j' = 1, 2, ..., n$. Bob sets $d_0 = d(rt)$, where $rt$ is the root of $T$. Assuming Bob is user $U_2$, Bob signs $d_0$ as $\{d_0\}_{sk_2}$ and sends $\{C_B, auth(S_B), \{d_0\}_{sk_2}\}$ to CS as a proximity-test response. When LBS Server receives Bob's response $\{C_B, auth(S_B), \{d_0\}_{sk_2}\}$, it computes $R_{AB}$ as follows,

$$\begin{aligned}
R_{AB} &= \{C_A \times C_B\} \\
&= \{C_{a_1} \times C_{b_1}, C_{a_1} \times C_{b_2}, ..., C_{a_1} \times C_{b_n}, \\
&\quad C_{a_2} \times C_{b_1}, C_{a_2} \times C_{b_2}, ..., C_{a_2} \times C_{b_n}, \\
&\quad \cdots, \quad \cdots, \quad \cdots, \quad \cdots, \\
&\quad C_{a_m} \times C_{b_1}, C_{a_m} \times C_{b_2}, ..., C_{a_m} \times C_{b_n}\}.
\end{aligned}$$

where

$$\begin{aligned}
&C_{a_{i'}} \times C_{b_{j'}} \\
&= ((g^{r_{i'}} \cdot g^{t_{j'}})^{-1}, h^{a_{i'}+r_{i'}} \cdot h^{t_{j'}-b_{j'}}) \\
&= (g^{-(r_{i'}+t_{j'})}, h^{r_{i'}+t_{j'}+(a_{i'}-b_{j'})}).
\end{aligned}$$

Then LBS Server computes the proof information $Pr_{j'}$ for the accumulation values $acc(b_{j'})$, where $j' = 1, 2, ..., n$. Assuming $v_0$ stores $acc(b_{j'})$, there is a shortest path $v_0, v_1, ..., v_l$ from leaf node $v_0$ to root node $v_l$. $Pr_{j'} = (pr_1, pr_2, ..., pr_l)$, where $pr_j$ is the pair of the digest of node $v_{j-1}$ and the witness that can authenticate $v_{j-1}$, subject to node $v_j$. We define $pr_j$ as $pr_j = (\beta_j, \gamma_j)$, where

$$\beta_j = d(v_{j-1}), \ \gamma_j = g^{\prod_{w \in Ch(v_j)-\{v_{j-1}\}}(h(d(w))+s)}.$$

Finally, LBS Server sends back to Alice $\{R_{AB}, Pr_B = \{Pr_{j'}\}_{\forall b_{j'} \in S_B}, \{d_0\}_{sk_2}\}$, where $j' = 1, 2, ..., n$, and we require that $j'$ is in the increasing order from 1 to $n$.

### D. Results Retrieval and Correctness Verification

After receiving $\{R_{AB}, Pr_B, \{d_0\}_{sk_2}\}$ from LBS Server, Alice first decrypts $R_{AB}$ with her private key $sk_1 = x_1$ as follows,

$$\begin{aligned}
&D(R_{AB}) = \{D(C_A \times C_B)\} \\
&= \{D(C_{a_1} \times C_{b_1}), D(C_{a_1} \times C_{b_2}), ..., D(C_{a_1} \times C_{b_n}), \\
&\quad D(C_{a_2} \times C_{b_1}), D(C_{a_2} \times C_{b_2}), ..., D(C_{a_2} \times C_{b_n}), \\
&\quad \cdots, \quad \cdots, \quad \cdots, \quad \cdots, \\
&\quad D(C_{a_m} \times C_{b_1}), D(C_{a_m} \times C_{b_2}), ..., D(C_{a_m} \times C_{b_n})\}.
\end{aligned}$$

where

$$\begin{aligned}
&D(C_{a_{i'}} \times C_{b_{j'}}) \\
&= h^{r_{i'}+t_{j'}+(a_{i'}-b_{j'})} \cdot (g^{-(r_{i'}+t_{j'})})^{x_1} \\
&= h^{r_{i'}+t_{j'}+(a_{i'}-b_{j'})-(r_{i'}+t_{j'})} \\
&= h^{a_{i'}-b_{j'}}.
\end{aligned}$$

Alice checks if the following equation holds for all $i'$ and $j'$.

$$D(C_{a_{i'}} \times C_{b_{j'}}) \overset{?}{=} 1. \tag{1}$$

If the above equation holds for $i' = m'$ and $j' = n'$, Alice knows that $a_{m'} = b_{n'} \in I = S_A \cap S_B$. Assuming there are $\mu$ pairs of $(i', j')$ that satisfy the equation, namely, $(m_1, n_1), (m_2, n_2), ..., (m_\mu, n_\mu)$. Then, based on the structure of $D(R_{AB})$ Alice can easily find that $I = \{b_{n_1}, b_{n_2}, ..., b_{n_\mu}\}$ and $|I| = \mu$, which indicates that Alice knows how large the overlap is between her discoverable range and Bob's.

However, Alice cannot accept the results unless she can verify the correctness of the computation results. If the results are not correct, then Alice might make wrong decisions in choosing the responding users, which may breach her privacy. Alice verifies the correctness by checking if each element $b_{n_\kappa} \in I$ also belongs to $S_B$, where $\kappa = 1, 2, ..., \mu$. First, Alice computes the accumulation value of each element $b_{n_\kappa}$ as

$$acc(b_{n_\kappa}) = g^{b_{n_\kappa}} \cdot g^s = g^{b_{n_\kappa}+s}.$$

Then, for each $\kappa = 1, 2, ..., \mu$, Alice checks if the following equations hold based on $Pr_{n_\kappa}$,

$$e(\beta_1, g) \overset{?}{=} e(acc(b_{n_\kappa}), g^{n_\kappa} \cdot g^s) \tag{2}$$
$$e(\beta_j, g) \overset{?}{=} e(\gamma_{j-1}, g^{h(\beta_{j-1})} \cdot g^s) \tag{3}$$
$$e(d_0, g) \overset{?}{=} e(\gamma_l, g^{h(\beta_l)} \cdot g^s) \tag{4}$$

where $j = 2, 3, ..., l$. If all equations hold for each $\kappa = 1, 2, ..., \mu$, Alice accepts the results from LBS server. Otherwise, Alice discards the results.

## V. Protocol Evaluation

In this section, we demonstrate the security, efficiency and feasibility of our scheme.

### A. Security and Privacy Analysis

*1) Correctness of Verification:* In our scheme, a requesting user Alice can verify the correctness of the computation results based on proof information from LBS server. Assume that an element $b_{n_\kappa} \in S_A$ and $\notin S_B$, but the decryption of the computation results show that $b_{n_\kappa} \in S_A \cap S_B$. This could happen when the server mistakenly compute the results or is comprised by attackers. Then, Alice will find that at least one of the equations $Eq.(2) - Eq.(4)$ will not hold. This verification is important, because if Alice's and Bob's intersection set has $\eta$ elements but the computation results show that there are $\eta^{'} > \eta$ elements, then the results may mislead Alice to believe that Bob's DR has more overlapping with hers. The q-strong DH assumption guarantees the server cannot generate proof information without users' authenticated data structure.

*2) Location Privacy:* In our scheme, every user has a set which represent his discoverable range. Before the proximity test, all elements in the set is encrypted with ElGamal encryption by using requesting user's public key. The LBS server will perform computations based on users' ciphertexts without knowing the corresponding plaintexts which are the real location information. Besides, LBS server cannot learn any information about the size of the intersection set, which protects the requesting user's and responding user's location privacy. In our scheme, only the requesting user can decrypt the computation results returned from the server and find the size of the overlapping cells between the discoverable ranges. The cardinality of the obtained intersection is smaller or equal to the real intersection. From the location privacy point of view, the possibility of miss detection can reduce the leakage risk of the requesting user's location and we consider preserving users' location privacy has higher priority than detecting the number of the responding users.

### B. Performance Analysis

*1) Simulation Setup:* We implement our simulation of computational cost based on the Pairing-based Cryptography (PBC) Library [25]. In our scheme, we use the Type A elliptic curve, which has the form of $y^2 = x^3 + x$. We generate a large prime $p$, and $|p| = 512$ bits. The base field of the curve is 512 bits and the order of the group is 160 bits, which provides the same security level as 1024-bit RSA. We use VMWare Workstation 10 Ubuntu 14.04 in a desktop with Intel Core i5 processor and 4G RAM. All the simulation results are averaged over 10 randomized runs.

TABLE I
SENARIO SETTINGS

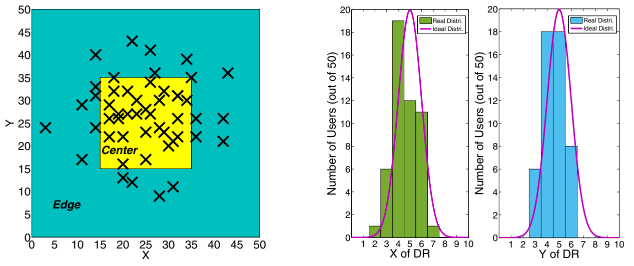
| Parameter | Setting |
|---|---|
| Simulation Area | $2km \times 2km$ |
| Number of cells | $2500, \{(1,1),(1,2),...(50,50)\}$ |
| Unit Cell Area | $40m \times 40m$ |
| Number of Users | 50 |
| User Distribution | $f_{xy} = \frac{1}{128\pi} e^{-\frac{(x-25)^2+(y-25)^2}{128}}$ |
| DR Distribution | $f_{xy} = \frac{1}{2\pi} e^{-\frac{(x-5)^2+(y-5)^2}{2}}$ |
| No. of Responses | $\{3,6,9,...,27,30\}$ |

In our scenario settings, we assume the service area is $2km \times 2km$ and it is evenly divided into 2500 square cells, labelled by $(1,1),(1,2),...(50,50)$ from bottom left corner to up right corner. Each cell's width $L = 40m$. There are $N = 50$ users normally distributed over the service area, as shown in Fig. 5(a). We define the yellow-colored area $15 \leq X, Y \leq 35$ as the *Center* and the rest as *Edge*. Each user's discoverable range also follows normal distribution, see Fig. 5(b). The detailed scenario settings are summarized in Table.I.

*2) Simulation Results:* We present our simulation results from three aspects.

- Performance on waiting time:

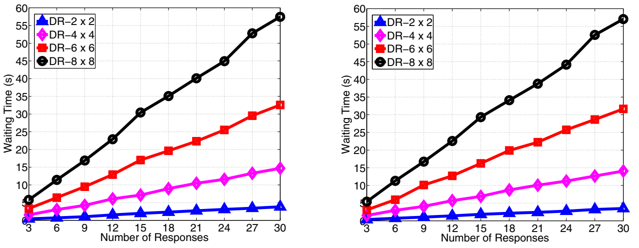
First of all, we focus on the overall performance of our proposed scheme. Under the same security and privacy level,



(a) User Distribution (b) DR Distribution

Fig. 5. User and DR Distribution

a user cares most about the quality of LBS of a scheme and the waiting time for the proximity-test is a critical factor. We show the total *Waiting Time vs Number of Responses* in Fig. 6(a) and Fig. 6(b). In Fig. 6(a), the requesting user Alice is at the center of the service area, while in Fig. 6(b) Alice is at the edge. Here, *DR-2*$\times$*2* means that the requester's DR is a $2 \times 2$ square. As we can see from both Fig. 6(a) and Fig. 6(b), Alice's total waiting time is proportional to the number of responses for the same discoverable range (DR) regardless of her location in the service area. For the same number of responses and location, when the DR increases from $2 \times 2$ to $8 \times 8$ the waiting time also increases. Alice's location plays a less important role in the total waiting time, because Alice's location difference only leads to the difference in the size of the intersection and the intersection takes negligible time to decrypt. The waiting time at the center is slightly longer than the waiting time at edge. Consequently, if Alice wants a shorter waiting time, she should define a smaller discoverable range, e.g.when $DR = 3 \times 3$, the waiting time is less than 5 seconds. However, a smaller DR means that a user's exact location is easier to guess. As result, there is a tradeoff between the waiting time and the privacy level.

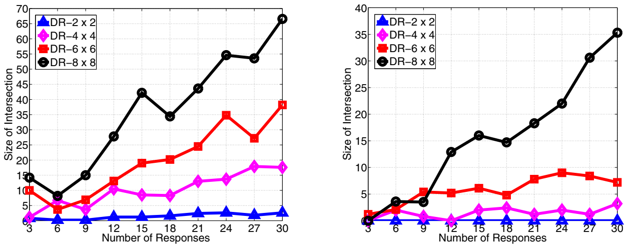


(a) Requesting User at Center (b) Requesting User at Edge

Fig. 6. Waiting Time vs Number of Responses

- Performance on intersection:

Second, we show in Fig. 7(a) and Fig. 7(b) the *Size of Intersection vs Number of Responses*. The size of intersection means the total number of overlapping cells between requesting user Alice's DR and responding users' DRs. The average size of intersection can reflect possibility of finding an *"valid"* responding user. For the same DR and location, the size of intersection increases with the number of responses. For the same number of responses and location, the size of intersection increases with the DR. Given the same DR and number of responses, Alice has a larger intersection with the responding users when she is at the center. As a result, if a user wants to increase the possibility of finding valid responding users, a larger DR is preferred or a location near the center is preferred given other conditions are the same. As we already discussed, the larger DR will lead to longer waiting time, which is another tradeoff.

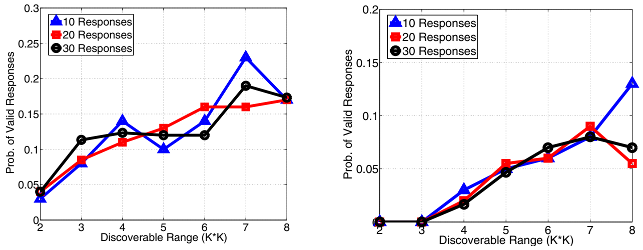


(a) Requesting User at Center (b) Requesting User at Edge

Fig. 7. Size of Intersection vs Number of Responses

- Performance on success probability:

Finally, we study the relationship between the *Probability of Valid Responses vs DR* in detail. The probability of valid responses is defined as the ratio between the non-empty intersection and the number of responses. For example, if there are 10 responses and 4 of them have overlapping with the requester's DR, then the probability equals to $4/10 = 40\%$. First of all, as we can see in Fig. 8(a) and Fig. 8(b), for the same number of responses and same location, the probability of valid responses increases with the DR overall but not always. Besides, for the same DR and location, the number of responses has little impact on the probability. What's more, even when the DR and number of responses increase at the same time, the probability may still decrease, e.g., in Fig. 8(a), the probability at $(DR = 6, 30\ \textit{Responses})$ is smaller than probability at $(DR = 4, 10\ \textit{Responses})$. If the requesting user Alice is near the center of service area, then she has higher probability of receiving valid responses.



(a) Requesting User at Center (b) Requesting User at Edge

Fig. 8. Prob. of Valid Responses vs DR

## VI. Conclusion

In this paper, we propose a privacy-preserving and verifiable proximity test scheme for location-based services. The proposed scheme enables LBS users to send proximity-test requests to the LBS server and get the proximity-test computation results from the LBS server in a privacy-preserving way. In particular, the LBS server learns nothing about the location information of the participants and the requesting user cannot learn the responding users' exact locations either. The requesting user can learn the total number of responding users and the portion of valid responses. Our scheme also provides users with a method to verify the correctness of the proximity-test results generated by the semi-trusted server.

## References

[1] L. Siksnys, J. R. Thomsen, S. Saltenis, and M. L. Yiu, "Private and flexible proximity detection in mobile social networks," in *Mobile Data Management (MDM), 2010 Eleventh International Conference on*. IEEE, 2010, pp. 75–84.

[2] A. Narayanan, N. Thiagarajan, M. Lakhani, M. Hamburg, and D. Boneh, "Location privacy via private proximity testing." in *NDSS*. Citeseer, 2011.

[3] Z. Lin, D. F. Kune, and N. Hopper, "Efficient private proximity testing with gsm location sketches," in *Financial Cryptography and Data Security*. Springer, 2012, pp. 73–88.

[4] L. Šikšnys, J. R. Thomsen, S. Šaltenis, M. L. Yiu, and O. Andersen, "A location privacy aware friend locator," in *Advances in Spatial and Temporal Databases*. Springer, 2009, pp. 405–410.

[5] R. Zhang, J. Zhang, Y. Zhang, J. Sun, and G. Yan, "Privacy-preserving profile matching for proximity-based mobile social networking," *Selected Areas in Communications, IEEE Journal on*, vol. 31, no. 9, pp. 656–668, 2013.

[6] H. Zhu, S. Du, M. Li, and Z. Gao, "Fairness-aware and privacy-preserving friend matching protocol in mobile social networks," *Emerging Topics in Computing, IEEE Transactions on*, vol. 1, no. 1, pp. 192–200, 2013.

[7] Y. Zheng, M. Li, W. Lou, and Y. T. Hou, "Sharp: Private proximity test and secure handshake with cheat-proof location tags," in *Computer Security–ESORICS 2012*. Springer, 2012, pp. 361–378.

[8] M. Li, H. Zhu, Z. Gao, S. Chen, L. Yu, S. Hu, and K. Ren, "All your location are belong to us: Breaking mobile social networks for automated user location tracking," in *Proceedings of the 15th ACM international symposium on Mobile ad hoc networking and computing*. ACM, 2014, pp. 43–52.

[9] X. Liu, K. Liu, L. Guo, X. Li, and Y. Fang, "A game-theoretic approach for achieving k-anonymity in location based services," in *INFOCOM, 2013 Proceedings IEEE*. IEEE, 2013, pp. 2985–2993.

[10] L. Guo, X. Zhu, C. Zhang, and Y. Fang, "Privacy-preserving attribute-based friend search in geosocial networks with untrusted servers," in *Global Communications Conference (GLOBECOM), 2013 IEEE*. IEEE, 2013, pp. 629–634.

[11] L. Guo, C. Zhang, and Y. Fang, "Privacy-preserving revocable content sharing in geosocial networks," in *Communications and Network Security (CNS), 2013 IEEE Conference on*. IEEE, 2013, pp. 118–126.

[12] C. Wang, K. Ren, and J. Wang, "Secure and practical outsourcing of linear programming in cloud computing," in *INFOCOM, 2011 Proceedings IEEE*. IEEE, 2011, pp. 820–828.

[13] S. Mascetti, C. Bettini, D. Freni, X. S. Wang, and S. Jajodia, "Privacy-aware proximity based services," in *Mobile Data Management: Systems, Services and Middleware, 2009. MDM'09. Tenth International Conference on*. IEEE, 2009, pp. 31–40.

[14] B. Applebaum, Y. Ishai, and E. Kushilevitz, "From secrecy to soundness: Efficient verification via secure computation," in *Automata, Languages and Programming*. Springer, 2010, pp. 152–163.

[15] S. Benabbas, R. Gennaro, and Y. Vahlis, "Verifiable delegation of computation over large datasets," in *Advances in Cryptology–CRYPTO 2011*. Springer, 2011, pp. 111–131.

[16] D. Fiore and R. Gennaro, "Publicly verifiable delegation of large polynomials and matrix computations, with applications," in *Proceedings of the 2012 ACM conference on Computer and communications security*. ACM, 2012, pp. 501–512.

[17] L. Guo, Y. Fang, M. Li, and P. Li, "Verifiable privacy-preserving monitoring for cloud-assisted mhealth systems," in *INFOCOM, 2015 Proceedings IEEE*. IEEE, 2015.

[18] R. Lu, X. Lin, Z. Shi, and J. Shao, "Plam: A privacy-preserving framework for local-area mobile social networks," in *INFOCOM, 2014 Proceedings IEEE*, April 2014, pp. 763–771.

[19] C. Papamanthou, R. Tamassia, and N. Triandopoulos, "Authenticated hash tables," in *Proceedings of the 15th ACM conference on Computer and communications security*. ACM, 2008, pp. 437–448.

[20] ——, "Optimal verification of operations on dynamic sets," in *Advances in Cryptology–CRYPTO 2011*. Springer, 2011, pp. 91–110.

[21] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in *Advances in CryptologyCRYPTO 2001*. Springer, 2001, pp. 213–229.

[22] E.-J. Goh, "Encryption Schemes from Bilinear Maps," Ph.D. dissertation, Department of Computer Science, Stanford University, Sep 2007.

[23] A. J. Menezes, T. Okamoto, and S. A. Vanstone, "Reducing elliptic curve logarithms to logarithms in a finite field," *Information Theory, IEEE Transactions on*, vol. 39, no. 5, pp. 1639–1646, 1993.

[24] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Trans. Inf. Theor.*, vol. 22, no. 6, pp. 644–654, Sep. 2006.

[25] L. Ben, "The pairing-based cryptography, https://crypto.stanford.edu/pbc/," access January 20, 2015.