PLAM: A Privacy-Preserving Framework for Local-Area Mobile Social Networks

Rongxing Lu[†], Xiaodong Lin[‡], Zhiguo Shi[§], and Jun Shao[*]
[†]School of Electrical and Electronics Engineering, Nanyang Technological University, Singapore 639798
[‡]Faculty of Business and Information Technology, University of Ontario Institute of Technology, Ontario, Canada
[§]Zhejiang University, Hangzhou 310027, China; [*]Zhejiang Gongshang University, Hangzhou 310018, China
Email: rxlu@ntu.edu.sg; xiaodong.lin@uoit.ca; shizg@zju.edu.cn; chn.junshao@gmail.com

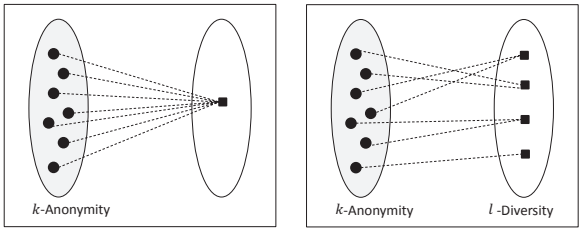
***Abstract*—In this paper, we propose a privacy-preserving framework, called PLAM, for local-area mobile social networks. The proposed PLAM framework employs a privacy-preserving request aggregation protocol with k-Anonymity and l-Diversity properties while without involving a trusted anonymizer server to keep user preference privacy when querying location-based service (LBS), and integrates unlinkable pseudo-ID technique to achieve user identity privacy, location privacy. Moreover, the proposed PLAM framework also introduces the privacy-preserving and verifiable polynomial computation to keep LBS provider's functions private while preventing the provider from cheating in computation. Detailed security analysis shows that the proposed PLAM framework can not only achieve desirable privacy requirements but also resist outside attacks on source authentication, data integrity and availability. In addition, extensive simulations are also conducted, and simulation results guide us on how to set proper thresholds for k-anonymity, l-diversity to make a tradeoff between the desirable user preference privacy level and the request delay in different scenarios.**

***Keywords* – Privacy-preserving, preference privacy, location-based services, mobile social network**

I. Introduction

Mainly due to the increased popularity of smartphones, like iPhone, Samsung Galaxy, and Blackberry, mobile social network (MSN), as an extension of online social network (OSN), has received unprecedented interests today [1]. Different from the traditional OSN, MSN can not only enable mobile users to access the existing online social networks, like Facebook and Twitter, from smartphone anywhere anytime, but also open a newly emerging mobile social network paradigm, where mobile users can utilize their location information to access some location based service (LBS) nearby, e.g, to query on "where is the nearest restaurant to their taste", or directly share information with other mobile users of similar interests in close proximity to them.

Although MSN is very popular, it also raises significant privacy concerns, especially the location privacy. For example, if Alice often reveals her locations close to a hospital when requesting LBS, these location information could be used by a *privacy-curious* LBS provider to infer that Alice may have some health problems. Although pseudo-ID technique can hide Alice's real identity, if the location is not protected, the pseudo-ID is insufficient to preserve Alice's sensitive health information [2]. Therefore, many privacy-preserving techniques have recently been studied in MSN to achieve location privacy in LBS Query, e.g., Geographic masking [3], Redundant query [4], and spatial k-anonymity [5], [6] techniques. In addition to the location privacy, user preference privacy is also important in MSN. If user preference privacy is not protected, more sensitive information could still be disclosed. Follow up the above example, we can see, even though the location information that "Alice is close to a hospital" is protected, i.e., the inference that Alice may have some health problems can be blocked, if Alice often requests the service on the knowledge of diabetes, then more sensitive information regarding Alice's health could still be disclosed, i.e., the LBS provider can infer that Alice may suffer from diabetes. Despite significant efforts made on location privacy already, most existing privacy-preserving techniques [3]–[6] are insufficient to protect user preference privacy in LBS Query. For example, even though there are k users simultaneously requesting the LBS, their preference privacy could be still inferred by the LBS provider once they request the same preference-relevant service, as shown in Fig. 1(a). Thus, user preference privacy is more challenging than the location privacy in LBS Query.



(a) preference privacy inferred (b) preference privacy protected

Fig. 1. Preference privacy preservation versus k-anonymity, l-diversity

As is known to all, k-Anonymity is originally an important technique to protect privacy in relational database. But, when k-Anonymity does not guarantee privacy, i.e., all sensitive values in an equivalence class lack diversity, the concept of l-diversity is further proposed, which ensures each equivalence class has at least l well-represented sensitive values [7]. Motivated by the concept of l-diversity in relational database, if we can ensure that k users request at least l services in LBS Query, as shown in Fig. 1(b), then the *privacy-curious* LBS provider cannot link a specific service to a specific

978-14799-3360-0/14/$31.00 ©2014 IEEE

user. As thus, user's preference privacy can be protected. However, when we implement this idea in MSN, new arising challenge that we have to face is how to calculate l-diversity while keeping each user's individual request privacy. In [8], a trusted anonymizer server is introduced to calculate l-diversity. However, the anonymizer server is usually costly, and once it is attacked, the single point of failure would paralyze the whole systems. Therefore, privacy-preserving l-diversity calculation without involving a trusted anonymizer server is still challenging in MSN. To the best of our knowledge, there is no approach so far to the challenge.

Local-Area MSN is referred to as a mobile social network formed by a group of mobile users roaming in a local area ($\mathcal{LA}$) together with a local LBS provider located in $\mathcal{LA}$. When a mobile user is roaming in $\mathcal{LA}$, he can request the LBS provider with his private inputs, and the latter will invoke the corresponding services to compute and return the results to the user. Just like accessing a Local-Area Network, e.g., WiFi, when a mobile user accesses the LBS provider, he cannot deny he is currently located in $\mathcal{LA}$. However, in order to keep his sensitive personal privacy, he hopes to hide his past and future locations and keep his preference privacy when querying LBS. But, as we discussed above, without a costly and trusted anonymizer server, it is still challenging to achieve user preference privacy. In this paper, aiming at addressing the above challenge, we propose a Privacy-preserving framework for Local-Area Mobile social networks, called PLAM. The proposed PLAM framework is characterized by employing a privacy-preserving request aggregation protocol to obtain k-Anonymity and l-Diversity without involving a trusted anonymizer server for achieving user preference privacy; integrating unlinkable pseudo-ID technique to protect user past and future locations; and introducing privacy-preserving and verifiable computation to protect LBS provider's privacy. To summarize, the main contributions of this paper are threefold.

- First, we propose a privacy-preserving request aggregation protocol in PLAM, which enables a group of mobile users to cooperatively aggregate their requests with k-Anonymity and l-Diversity properties for achieving their preference privacy. Compared with those previous works on addressing the same goal, the proposed protocol does not need a trusted anonymizer server. To protect a mobile user past and future locations, pseudo-ID technique is integrated, where frequently changing pseudo-IDs at different locations makes user's past, current and future locations unlinkable, as desired.
- Second, we introduce the privacy-preserving and verifiable computation functions in PLAM, which enables the LBS provider to output verifiable results while keeping the functions of LBS private. Because the functions of LBS are usually regarded as the private assets of the LBS provider, they won't be open, and thus the privacy-preserving and verifiable computations are desirable.
- Third, we carry out extensive simulations to examine the relationship between k-Anonymity, l-Diversity and the request delay in privacy-preserving request aggregation protocol, and the results guide us to set proper thresholds for k-Anonymity, l-Diversity to make a tradeoff between the desirable user preference privacy and the request delay in different scenarios.

The remainder of this paper is organized as follows. In Section II, we formalize the system model, security model, and identify our design goals. We present the details of our proposed PLAM framework in Section III, followed by the security analysis and performance evaluation in Section IV and Section V, respectively. Section VI reviews the related works, and Section VII closes the paper with the conclusion.

II. Models and Design Goal

In this section, we formalize the system model, security model, and identify our design goals.

A. System Model

In our model, we consider a typical Local-Area MSN which is composed of a set of mobile users $\mathcal{U} = \{U_1, U_2, \cdots\}$ roaming in a specific local area $\mathcal{LA}$, a LBS provider (LBSP) located in $\mathcal{LA}$, and a trusted authority (TA), as shown in Fig. 2.

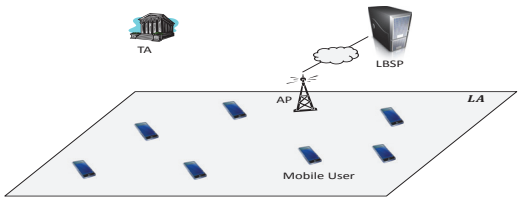


Fig. 2. System model under consideration

Trusted Authority: The trusted authority (TA) is fully trusted in the system, whose duty is to initialize the whole system, and register the LBSP and mobile users $\mathcal{U}$ in the system initialization phase. Generally, the TA is offline in other phases, however once there is any dispute arising, the TA can be involved to settle it down.

LBSP: The LBSP is located in $\mathcal{LA}$ with an access point (AP), which mainly provides a set of LBS relevant to the local area $\mathcal{LA}$. Given the inputs from mobile users, the LBSP uses the private LBS functions to compute and return the results to users. Since the *polynomial* functions can serve as the building blocks for a wide range of applications such as statistical analysis and scientific computing [9], we assume that all LBS functions offered by the LBSP are polynomials in our model.

Mobile Users $\mathcal{U} = \{U_1, U_2, \cdots\}$: All mobile users follow the same random mobility model, e.g., the random-way point model, to move in $\mathcal{LA}$ with the velocity $\mathbf{v}$. When two users encounter, they can communicate with each other within the transmission range. In addition, each user U_i, based on his preferences, may collect data of his interests during his movement. Once U_i is ready to request some LBS from the LBSP with his private collected data as inputs, he will submit his request to the access point (AP) of the LBSP and getting the corresponding response.

B. Security Model

In our security model, we mainly consider the privacy requirements of each role in our system model. In specific, we consider all roles, except the TA, are *honest-but-curious* in our model, i.e., they will faithfully follow the protocol, but could also snoop into other role's preference privacy on account of some side information available to them.

Mobile users: The privacy requirement of a mobile user include his preference privacy, location privacy and identity privacy, where the preference privacy indicates the user won't let others dope out his preferences, even when he provides data of his interests to request LBSP, the location privacy shows the user won't let others know his past and future location information, and the identity privacy means the user tries to keep his real identity secret. Meanwhile, each user is also *privacy-curious*, i.e., he tends to disclose the privacy of other users and LBSP from the side information available to him.

LBSP: The LBSP provides a set of polynomial LBS functions to users in $\mathcal{LA}$. Because polynomial functions are private assets of the LBSP, which may include some private and sensitive information, the LBSP won't disclose the concrete polynomials to users, but will provide users with the correct outputs upon their inputs. This is the privacy requirement of LBSP. Meanwhile, the LBSP is also *curious* of identifying users' preference privacy when users request services.

Note that, since all roles in our security model are *honest-but-curious*, they will follow the protocol and won't maliciously collude with each other to snoop into others' privacy. Therefore, the collusion among users and the collusion between users and the LBSP are beyond the scope of this paper. In addition, although the focus of this paper is on the privacy, in order to improve the security and reliability of PLAM, the protocols in PLAM should also be secure against possible attacks launched by an outside adversary $\mathcal{A}$ on source authentication, data integrity and availability in our model.

C. Design Goal

Our design goal is to develop a privacy-preserving framework to not only protect the privacy of both mobile users and the LBSP in Local-Area MSN, but also resist against other attacks launched by an outside adversary. Specifically, the following desirable goals should be achieved.

• *Achieving privacy-preserving LBS query in Local-Area MSN.* The proposed framework should achieve privacy requirements of mobile users and the LBSP. In particular, i) when a mobile user requests a LBS, neither the LBSP nor other users can identify the user's preference privacy; ii) when a mobile user gets the response from the LBSP, he can be convinced that the result of a polynomial function is correct, but cannot get to know the construction of the polynomial; and iii) both identity privacy and past & future location privacy of a mobile user will not be disclosed in the proposed framework.

• *Resisting other attacks launched by an outside adversary on source authenticity, data integrity and availability.* The proposed framework should also be secure and reliable in Local-Area MSN. Once an outside adversary launches some attacks, e.g., forging signature, replay and DoS attacks, on source authentication, data integrity and availability, the proposed framework should be able to detect them.

III. Proposed PLAM Framework

In this section, we propose our PLAM framework, which consists of four parts: system initialization, privacy-preserving request aggregation, privacy-preserving LBS computation, and privacy-preserving response verification. Before plugging into the details, we first review the preliminaries, including bilinear pairing [10] and BV homomorphic encryption [11], which will serve as the basis of the proposed PLAM framework.

A. Preliminaries

1) Bilinear Pairings: Let $\mathbb{G}$, $\mathbb{G}_T$ be two multiplicative cyclic groups with the same prime order q. Suppose $\mathbb{G}$ and $\mathbb{G}_T$ are equipped with a pairing, i.e., a non-degenerated and efficiently computable bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ such that $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab} \in \mathbb{G}_T$ for all $a, b \in \mathbb{Z}_q^*$ and any $g_1, g_2 \in \mathbb{G}$. In group $\mathbb{G}$, the Computational Diffie-Hellman (CDH) problem is hard, i.e., given (g, g^a, g^b) for $g \in \mathbb{G}$ and unknown $a, b \in \mathbb{Z}_q^*$, it is intractable to compute g^{ab} in a polynomial time. However, the Decisional Diffie-Hellman (DDH) problem is easy, i.e., given (g, g^a, g^b, g^c) for $g \in \mathbb{G}$ and unknown $a, b, c \in \mathbb{Z}_q^*$, it is easy to judge whether $c = ab \bmod q$ by checking $e(g^a, g^b) \stackrel{?}{=} e(g^c, g)$. We refer to [10] for a more comprehensive description of pairing technique, and complexity assumptions.

Definition 1: A bilinear parameter generator $\mathcal{G}en$ is a probabilistic algorithm that takes a security parameter κ as input, and outputs a 5-tuple $(q, g, \mathbb{G}, \mathbb{G}_T, e)$, where q is a κ-bit prime number, $\mathbb{G}, \mathbb{G}_T$ are two groups with order q, $g \in \mathbb{G}$ is a generator, and $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ is a non-degenerated and efficiently computable bilinear map.

2) BV Homomorphic Encryption: BV homomorphic encryption (BVHE) is built upon the hardness of Learning with Errors (LWE) problem [11], and can fully support both additive and multiplicative homomorphisms. The detailed key generation (KEYGEN), encryption (ENC), and decryption (DEC) of BVHE algorithm are as follows.

• KEYGEN: Given the security parameter ι, two primes $\hat{p}, \hat{q}$ are chosen, where $\hat{q} < \hat{p}$ and $|\hat{q}| = \iota$. After that, three Rings $(R, R_{\hat{p}}, R_{\hat{q}})$ are selected, where $R = \mathbb{Z}[x]/(x^2+1)$, $R_{\hat{p}} = \mathbb{Z}_{\hat{p}}[x]/(x^2+1)$ and $R_{\hat{q}} = \mathbb{Z}_{\hat{q}}[x]/(x^2+1)$. Finally, a discrete Gaussian error distribution χ with standard deviation σ is also chosen. Under this setting, a public/private key pair (pk_i, sk_i) is generated as below: randomly choose (s_i, e_i) from χ and $a_{i1} \in R_{\hat{p}}$, compute $a_{i0} = -(a_{i1}s_i + \hat{q}e_i)$, and set the public key as $pk_i = (a_{i0}, a_{i1})$ and the private key as $sk_i = s_i$.

• ENC: To encrypt a message $m \in R_{\hat{q}}$, i.e., $C = E_{pk}(m)$, three samples (u', f', g') are chosen from χ; then calculate the ciphertext $C = (c_0, c_1)$ as $c_0 = a_{i0}u' + \hat{q}g' + m$ and $c_1 = a_{i1}u' + \hat{q}f'$.

• DEC: To recover m from $C = (c_0, c_1)$, i.e., $m = D_{sk}(C)$, we compute $m = c_0 + c_1 s_i \mod \hat{q}$. The correctness is as

follows:

$$\begin{aligned} & c_0 + c_1 s_i \\ =&(a_{i0}u' + \hat{q}g' + m) + (a_{i1}u' + \hat{q}f')s_i \\ =&-(a_{i1}s_i + \hat{q}e_i)u' + \hat{q}g' + m + (a_{i1}u' + \hat{q}f')s_i \\ =&m + \hat{q}(g' + f's_i - e_i u') \\ \Rightarrow& m + \hat{q}(g' + f's_i - e_i u') \bmod \hat{q} = m \end{aligned} \quad (1)$$

It is also not difficult to verify that, given two ciphertexts $E(m_0)$ and $E(m_1)$, we have

$$\begin{aligned} \text{Additive:} \quad & E(m_0) + E(m_1) = E(m_0 + m_1) \\ \text{Multiplicative:} \quad & E(m_0) \times E(m_1) = E(m_0 \times m_1) \end{aligned} \quad (2)$$

We refer to [11] for a more detailed description of BVHE's additive and multiplicative homomorphisms.

B. Description of PLAM Framework

1) System Initialization Phase: For a single-authority Local-Area MSN under consideration, we assume a trusted authority (TA) will bootstrap the whole system. Specifically, given the security parameters κ, ι, TA first generates the bilinear parameters $(q, g, \mathbb{G}, \mathbb{G}_T, e)$ by running $\mathcal{G}en(\kappa)$ and BVHE parameters $(\hat{p}, \hat{q}, R, R_{\hat{q}}, R_{\hat{q}})$ as above. Then, TA chooses a secure symmetric encryption algorithm $\mathcal{E}nc()$, i.e., AES, and three collision-resistant cryptographic hash functions $\mathcal{H} : \{0,1\}^* \to \mathbb{G}$, $\mathcal{H}_i : \{0,1\}^* \to \mathbb{Z}_q^*$ with $i = 0, 1$. In addition, TA also chooses a random number $s \in \mathbb{Z}_q^*$ as the master key, and computes $P_{pub} = g^s$. Finally, TA keeps the master key s secretly, and publishes the system parameters $params = (q, g, P_{pub}, \mathbb{G}, \mathbb{G}_T, e, \hat{p}, \hat{q}, R, R_{\hat{q}}, R_{\hat{q}}, \mathcal{H}, \mathcal{H}_0, \mathcal{H}_1, \mathcal{E}nc())$.

USER REGISTRATION: For each user $U_i \in \mathcal{U}$, when he registers himself in the system, the following steps will be run between TA and U_i.

- TA first computes $s_0 = \mathcal{H}_0(s)$, and uses s_0 and $\mathcal{E}nc()$ to compute pseudo-IDs $\mathcal{PID}_i = \{PID_{i1}, PID_{i2}, \cdots\}$, where each pseduo-ID $PID_{ij} \in \mathcal{PID}_i$ is computed as $PID_{ij} = \mathcal{E}nc_{s_0}(U_i || r_{ij})$ with a fresh random number $r_{ij} \in \mathbb{Z}_q$. Then, for each PID_{ij}, TA calculates its corresponding private key $sk_{ij} = \mathcal{H}(PID_{ij})^s$. Finally, TA sends $\mathcal{PID}_i$ and the corresponding private keys back to U_i via a secure channel.
- After receiving pseudo-IDs $\mathcal{PID}_i$ and the corresponding private keys, U_i verifies the correctness of each private key sk_{ij} by checking $e(\mathcal{H}(PID_{ij}), P_{pub}) \stackrel{?}{=} e(sk_{ij}, g)$.

LBSP REGISTRATION: When the LBSP registers into the system, the following steps will be run between TA and LBSP.

- Suppose the LBSP serves in the local area $\mathcal{LA}$. TA calculates the private key $sk = \mathcal{H}(\mathcal{LA})^s$ and sends it back to LBSP via a secure channel. After receiving the private key $sk = \mathcal{H}(\mathcal{LA})^s$, LBSP verifies its correctness by checking $e(\mathcal{H}(\mathcal{LA}), P_{pub}) \stackrel{?}{=} e(sk, g)$.
- Assume the LBSP will offer a set of private polynomial LBS functions in $\mathcal{LA}$. Then, the LBSP should further register these polynomials to TA. For simplicity of description, we here just discuss how the LBSP registers a generalized bivariate polynomial $F(x, y)$ with bounded degree d for an example[1]. Let $\mathcal{S} = \{S_{ij} = x^i y^j | 0 \le i \le d, 0 \le i \le d\}$, a generalized bivariate polynomial $F(x, y)$ can be represented as

$$F(x, y) = \sum_{S_{ij} \in \mathcal{S}} c_{ij} x^i y^j \quad (3)$$

  with each coefficient $0 \le c_{ij} < q$. Then, the LBSP submits $F(x, y)$ to TA for registration.
- After receiving the bivariate polynomial $F(x, y) = \sum_{S_{ij} \in \mathcal{S}} c_{ij} x^i y^j$, TA chooses two random numbers $a, b \in \mathbb{Z}_q^*$, and computes g^a, g^b, $F(a, b) = \sum_{S_{ij} \in \mathcal{S}} c_{ij} a^i b^j$. Then, TA sets $L_{F(x,y)} = g^a || g^b || g^{F(a,b)}$ as the label of the polynomial $F(x, y)$, and makes a signature $Sig = \mathcal{H}(L_{F(x,y)})^s$ on the label. Finally, TA returns $a||b||g^a||g^b||g^{F(a,b)}||Sig$ to the LBSP.
- Since $F(x, y)$ is a private polynomial, i.e., the LBSP won't disclose $F(x, y)$ but will output the results of $F(x, y)$ to users. In order to convince users that the results are correct, the LBSP should provide a signature of correctness [9]. Thus, after receiving $a||b||g^a||g^b||g^{F(a,b)}||Sig$ from TA, the LBSP uses the following trick to create a validation polynomial $G(x, y)$ for generating the signature of correctness. Because $F(x, y) - F(a, b)$ can be written as

$$\begin{aligned} & F(x, y) - F(a, b) = \sum_{S_{ij} \in \mathcal{S}} c_{ij} x^i y^j - \sum_{S_{ij} \in \mathcal{S}} c_{ij} a^i b^j \\ =& \sum_{S_{ij} \in \mathcal{S}} (c_{ij} x^i y^j - c_{ij} a^i y^j) + \sum_{S_{ij} \in \mathcal{S}} (c_{ij} a^i y^j - c_{ij} a^i b^j) \end{aligned} \quad (4)$$

  In addition, there always exist two polynomials $G_1(x, y)$ and $G_2(x, y)$ such that

$$\begin{aligned} \sum_{S_{ij} \in \mathcal{S}} (c_{ij} x^i y^j - c_{ij} a^i y^j) &= (x - a) G_1(x, y) \\ \sum_{S_{ij} \in \mathcal{S}} (c_{ij} a^i y^j - c_{ij} a^i b^j) &= (y - b) G_2(x, y) \end{aligned} \quad (5)$$

  we have

$$F(x, y) - F(a, b) = (x - a) G_1(x, y) + (y - b) G_2(x, y) \quad (6)$$

  Thus, the LBSP can set $G(x, y) = [G_1(x, y), G_2(x, y)]$ as the validation polynomial of $F(x, y)$. After that, the LBSP keeps $F(x, y)||G(x, y)$ secret, and publishes the label $L_{F(x,y)} = g^a||g^b||g^{F(a,b)}$ and its signature Sig.

Note that, since the LBSP usually registers and maintains a set of private polynomials, it is inefficient to make a signature on each label. Therefore, in order to flexibly maintain these polynomials' labels, the Merkle hash tree technique [14] can be utilized, as shown in Fig. 3. In such a way, only one signature is needed to authenticate the whole tree. Once some

[1]For the cases of generalized univariate and multivariate polynomials's constructions, please refer to [9], [12], [13].

polynomials are evolved for security or other reasons, it is also convenient to update the tree and generate a new signature for the updated tree root.

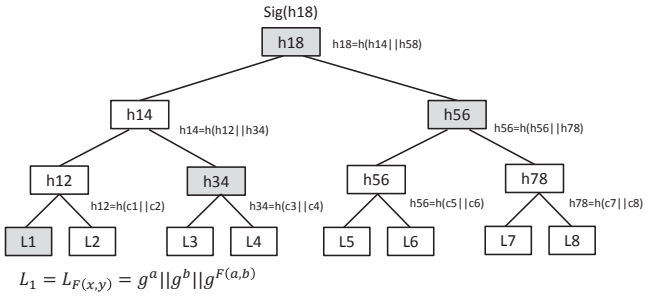


Fig. 3. Merkle hash tree technique is utilized to efficiently maintain polynomials' labels. To verify L_1, it only requires h_{34}, h_{56} and $Sig(h_{18})$.

2) Privacy-preserving Request Aggregation: Assume that the LBSP offers n private polynomial LBS functions, denoted as $\mathcal{L} = (L_1, L_2, \cdots, L_n)$ in the area $\mathcal{LA}$. When a mobile user, assumed U_1, is roaming in $\mathcal{LA}$, he wants to get a subset of LBS of $\mathcal{L}$ from the LBSP. However, in order to keep his preference privacy, U_1 won't directly disclose what services he will request to the LBSP. Therefore, a strategy for U_1 is to aggregate multiple users' requests with k-anonymity and l-diversity and send these requests together to the LBSP. In such a way, the LBSP cannot link a specific service to a specific user. Particularly, in order to improve user preference privacy, not only the number of involved users k should be larger than a threshold th_k, but also the number of requesting services l should also reach a threshold th_l. To achieve the k-anonymity and l-diversity requirements, U_1 and other users in proximity will cooperatively run the privacy-preserving request aggregation protocol by the following steps, which is characterized by aggregating all users' requests while keeping each individual user's requests privacy. For the clarity of presentation, we use $PID_i \in \mathcal{PID}_i$ to represent an unlinkable pseudo-ID of U_i. In addition, we denote $\mathbf{a}_i = (a_{i1}, a_{i2}, \cdots, a_{in})$, with each $a_{ij} \in \{0, 1\}$, as the requesting services of U_i in $\mathcal{LA}$. If $a_{ij} = 1$, U_i will request the service L_j, and won't request L_j when $a_{ij} = 0$.

Step 1. Based on the BVHE parameters $(\hat{p}, \hat{q}, R, R_{\hat{q}}, R_{\hat{q}})$, U_1 first generates a public/private key pair $(\mathsf{pk}, \mathsf{sk})$ as described in Section III-A2. Then, for his requesting service vector $\mathbf{a}_1 = (a_{11}, a_{12}, \cdots, a_{1n})$, U_1 uses pk to calculate $E_{\mathsf{pk}}(\bar{\mathbf{a}}_1)$ as

$$E_{\mathsf{pk}}(\bar{\mathbf{a}}_1) = \big(E_{\mathsf{pk}}(1 - a_{11}), \cdots, E_{\mathsf{pk}}(1 - a_{1n})\big) \tag{7}$$

Step 2. U_1 broadcasts the "request aggregation" message containing $\mathsf{pk}||E_{\mathsf{pk}}(\bar{\mathbf{a}}_1)$ to users in the proximity to him. If a user U_i in proximity receives the message and also wants to request some LBS from the LBSP with his requesting service vector $\mathbf{a}_i = (a_{i1}, a_{i2}, \cdots, a_{in})$, he will contact with U_1 and use the public key pk to calculate $E_{\mathsf{pk}}(\bar{\mathbf{a}}_i)$ as

$$E_{\mathsf{pk}}(\bar{\mathbf{a}}_i) = \big(E_{\mathsf{pk}}(1 - a_{i1}), \cdots, E_{\mathsf{pk}}(1 - a_{in})\big) \tag{8}$$

Step 3. Suppose users $(U_2, U_3, \cdots, U_k)$, with $k \geq 2$, are involved in the privacy-preserving request aggregation protocol, they will cooperatively exchange $E_{\mathsf{pk}}(\bar{\mathbf{a}}_i)$, for $i = 1, \cdots, k$, calculate $\prod_{i=1}^{k} E_{\mathsf{pk}}(\bar{\mathbf{a}}_i)$ as

$$\prod_{i=1}^{k} E_{\mathsf{pk}}(\bar{\mathbf{a}}_i) = \left(\prod_{i=1}^{k} E_{\mathsf{pk}}(1 - a_{i1}), \cdots, \prod_{i=1}^{m} E_{\mathsf{pk}}(1 - a_{in})\right) \tag{9}$$

and finally send the resulting $\prod_{i=1}^{k} E_{\mathsf{pk}}(\bar{\mathbf{a}}_i)$ back to U_1.

Step 4. After receiving $\prod_{i=1}^{k} E_{\mathsf{pk}}(\bar{\mathbf{a}}_i)$, U_1 calculates the number of requesting services l in the aggregation, i.e., the union $l = \sum_{i=1}^{n}(a_{1i} \vee a_{2i} \vee \cdots \vee a_{ki})$ by computing

$$l = n - D_{\mathsf{sk}}\left(\sum_{i=1}^{n}\left(\prod_{j=1}^{k} E_{\mathsf{pk}}(1 - a_{ji})\right)\right) \tag{10}$$

The correctness of l is as follows,

$$\begin{aligned}
l &= n - D_{\mathsf{sk}}\left(\textstyle\sum_{i=1}^{n}\left(\prod_{j=1}^{k} E_{\mathsf{pk}}(1 - a_{ji})\right)\right) \\
&\xrightarrow{\text{BVHE multiplicative homomorphism}} \\
&= n - D_{\mathsf{sk}}\left(\textstyle\sum_{i=1}^{n} E_{\mathsf{pk}}\left(\prod_{j=1}^{k}(1 - a_{ji})\right)\right) \\
&\xrightarrow{\text{BVHE additive homomorphism}} \\
&= n - D_{\mathsf{sk}}\left(E_{\mathsf{pk}}\left(\textstyle\sum_{i=1}^{n}\prod_{j=1}^{k}(1 - a_{ji})\right)\right) \\
&= n - \textstyle\sum_{i=1}^{n}\prod_{j=1}^{k}(1 - a_{ji}) \\
&= \textstyle\sum_{i=1}^{n}\left(1 - \prod_{j=1}^{k}(1 - a_{ji})\right) \\
&\xrightarrow{\because a_{ji \in \{0,1\}} \therefore 1 - a_{ji} \text{ can be written as } \overline{a}_{ji}} \\
&= \textstyle\sum_{i=1}^{n}(1 - \overline{a}_{1i} \wedge \overline{a}_{2i} \wedge \cdots \wedge \overline{a}_{ki}) \\
&= \textstyle\sum_{i=1}^{n}\left(\overline{a}_{1i} \wedge \overline{a}_{2i} \wedge \cdots \wedge \overline{a}_{ki}\right) \\
&= \textstyle\sum_{i=1}^{n}(a_{1i} \vee a_{2i} \vee \cdots \vee a_{ki}) \text{ as desired.}
\end{aligned} \tag{11}$$

Therefore, the number of requesting services l, i.e., the union of all users' requests, can be privacy-preserving computed in a cooperative way. If both $k \geq th_k$ and $l \geq th_l$ are satisfied, users $(U_1, U_2, \cdots, U_k)$ will cooperatively run the next step. Otherwise, U_1 sends $\mathsf{pk}||\prod_{i=1}^{k} E_{\mathsf{pk}}(\bar{\mathbf{a}}_i)$ and waits more users to join until both $k \geq th_k$ and $l \geq th_l$ do hold. Note that, in order to protect newly-joining users' preference privacy, at least $n_j \geq 2$ users should cooperatively run the *Step 3* to compute and return the result to U_1. Otherwise, U_1 would use the private key sk and the existing results to infer a single user's preference privacy. For example, if $a_{1i} \vee a_{2i} \vee \cdots \vee a_{ki} = 0$ originally, but $a_{1i} \vee a_{2i} \vee \cdots \vee a_{ki} \vee a_{(k+1)i} = 1$, then U_1 can infer $a_{(k+1)i} = 1$ for a single newly-joining user. But, if at least two users are invited at the same time, the success probability of this kind of inference attack becomes low.

Step 5. Assume mobile users $(U_1, U_2, \cdots, U_k)$ finally cooperatively request the services from the LBSP. Then, in this step, each U_i prepares his inputs based on his requesting service vector $\mathbf{a}_i = (a_{i1}, a_{i2}, \cdots, a_{in})$. Concretely, for each $a_{ij} = 1$ in $\mathbf{a}_i$, U_i prepares an input $\mathtt{in}_{ij}$, chooses and keeps a random number $\mathtt{r}_{ij}$, and uses the ID-based encryption (IBE) algorithm [10], with $\mathcal{H}(\mathcal{LA})$ as the public key, to compute

$$C_{ij} = IBE_{\mathcal{H}(\mathcal{LA})}(L_j||\mathtt{r}_{ij}||\mathtt{in}_{ij}) \tag{12}$$

where L_j is the label of the requesting service.

After all requests of users $(U_1, U_2, \cdots, U_k)$ are prepared in an encrypted patten as above, these encrypted requests will be aggregated as $\mathcal{C}$.

Step 6. Let $\mathtt{TP}$ be the current timestamp. To ensure the authenticity and resist the possible replay and DoS attacks, each U_i uses the private key $sk_i = \mathcal{H}(PID_i)^s$ to make a signature (S_i, T_i) on $\mathcal{C}||\mathtt{TP}$, where

$$S_i = \mathcal{H}(\mathcal{C}||\mathtt{TP})^{r_i} \cdot \mathcal{H}(PID_i)^s, \quad T_i = g^{r_i} \tag{13}$$

with a random number $r_i \in \mathbb{Z}_q^*$.

After that, all signatures (S_i, T_i), for $1 \leq i \leq k$, will be aggregated as

$$S_k = \prod_{i=1}^{k} S_i, \quad T_k = \prod_{i=1}^{k} T_i \tag{14}$$

Finally, the aggregated request

$$\mathcal{C}||\mathtt{TP}||PID_1||PID_2||\cdots||PID_k||S_k||T_k$$

with k-anonymity and l-diversity from users $(U_1, U_2, \cdots, U_k)$ will be reported to the LBSP.

3) Privacy-preserving LBS Computation: Upon receiving the aggregated request, the LBSP first checks the timestamp $\mathtt{TS}$. If it is outdated, the request will be directly discarded. Otherwise, the LBSP runs the following steps.

Step 1. The LBSP first verifies the request's validation by checking

$$e(S_k, g) = e(T_k, \mathcal{H}(\mathcal{C}||\mathtt{TP}))e\left(P_{pub}, \prod_{i=1}^{k} \mathcal{H}(PID_i)\right) \tag{15}$$

If it holds, the LBSP is convinced that the request comes from a group of users with identities $PID_1||PID_2||\cdots||PID_k$ and will continue to perform the next step. Otherwise, the LBSP will reject the request. The correctness is as follows,

$$\begin{aligned} e(S_k, g) &= e\left(\prod_{i=1}^{k} S_i, g\right) = e\left(\prod_{i=1}^{k} \mathcal{H}(\mathcal{C}||\mathtt{TP})^{r_i} \cdot \mathcal{H}(PID_i)^s, g\right) \\ &= e\left(\prod_{i=1}^{k} \mathcal{H}(\mathcal{C}||\mathtt{TP})^{r_i}, g\right) e\left(\prod_{i=1}^{k} \mathcal{H}(PID_i)^s, g\right) \\ &= e\left(\prod_{i=1}^{k} g^{r_i}, \mathcal{H}(\mathcal{C}||\mathtt{TP})\right) e\left(g^s, \prod_{i=1}^{k} \mathcal{H}(PID_i)\right) \\ &= e(T_k, \mathcal{H}(\mathcal{C}||\mathtt{TP}))e\left(P_{pub}, \prod_{i=1}^{k} \mathcal{H}(PID_i)\right) \end{aligned} \tag{16}$$

Step 2. The LBSP uses the location-based private key $sk = \mathcal{H}(\mathcal{LA})^s$ to recover each request $L_j||\mathtt{r}_{ij}||\mathtt{in}_{ij}$ from $C_{ij} = IBE_{\mathcal{H}(\mathcal{LA})}(L_j||\mathtt{r}_{ij}||\mathtt{in}_{ij})$ in $\mathcal{C}$. Then, according to the label L_j to invoke the corresponding polynomial to compute, i.e., with the input of $\mathtt{in}_{ij}$, output the correct $\mathtt{out}_{ij}$ and verification information $\mathtt{witness}_{ij}$. Finally, the LBSP uses the random number $\mathtt{r}_{ij}$ to compute the response

$$R_{ij} = \mathcal{H}_1(\mathtt{r}_{ij})||\mathcal{E}nc_{\mathtt{r}_{ij}}(\mathtt{r}_{ij}||\mathtt{out}_{ij}||\mathtt{witness}_{ij}) \tag{17}$$

To illustrate the procedure more clear, we assume the request $L_j||\mathtt{r}_{ij}||\mathtt{in}_{ij}$ is in regard to the bivariate polynomial $F(x, y)$ discussed in Eq.(3), i.e., the label L_j is $L_{F(x,y)} = g^a||g^b||g^{F(a,b)}$ and the input $\mathtt{in}_{ij}$ is (x_0, y_0). Then, with the input $\mathtt{in}_{ij} = (x_0, y_0)$, the LBSP invokes $F(x, y)$ and $G(x, y) = [G_1(x, y), G_2(x, y)]$ to compute

$$\begin{aligned} \mathtt{out}_{ij} &= F(x_0, y_0) \\ \mathtt{witness}_{ij} &= [g^{G_1(x_0,y_0)}, g^{G_2(x_0,y_0)}] \end{aligned} \tag{18}$$

and form the response as

$$R_{ij} = \mathcal{H}_1(\mathtt{r}_{ij})||\mathcal{E}nc_{\mathtt{r}_{ij}}\left(\mathtt{r}_{ij}||F(x_0, y_0)||g^{G_1(x_0,y_0)}||g^{G_2(x_0,y_0)}\right) \tag{19}$$

Step 3. Once all requests are processed, the LBSP aggregates all responses as $\mathcal{R}$, and returns $\mathcal{R}$ to the users.

4) Privacy-preserving Response Verification: After receiving the response $\mathcal{R}$ from the LBSP, each user can use the random numbers in hand to identify the responses to him, recover and verify the results. For example, if U_i sends the request $L_j||\mathtt{r}_{ij}||\mathtt{in}_{ij}$ to the LBSP, he can identify, recover and verify the response $R_{ij} = \mathcal{H}_1(\mathtt{r}_{ij})||\mathcal{E}nc_{\mathtt{r}_{ij}}(\mathtt{r}_{ij}||F(x_0, y_0)||g^{G_1(x_0,y_0)}||g^{G_2(x_0,y_0)})$ by running the following steps:

Step 1. With the random number $\mathtt{r}_{ij}$ in hand, U_i can easily compute and use $\mathcal{H}_1(\mathtt{r}_{ij})$ to identify the response $R_{ij} = \mathcal{H}_1(\mathtt{r}_{ij})||\mathcal{E}nc_{\mathtt{r}_{ij}}(\mathtt{r}_{ij}||F(x_0, y_0)||g^{G_1(x_0,y_0)}||g^{G_2(x_0,y_0)})$ by comparing the response head $\mathcal{H}_1(\mathtt{r}_{ij})$ in R_{ij}. After that, U_i uses $\mathtt{r}_{ij}$ as the key to recover

$$\mathtt{r}_{ij}||F(x_0, y_0)||g^{G_1(x_0,y_0)}||g^{G_2(x_0,y_0)} \tag{20}$$

from $\mathcal{E}nc_{\mathtt{r}_{ij}}(\mathtt{r}_{ij}||F(x_0, y_0)||g^{G_1(x_0,y_0)}||g^{G_2(x_0,y_0)})$. If the recovered $\mathtt{r}_{ij}$ is correct, U_i continues to verify the result $F(x_0, y_0)||g^{G_1(x_0,y_0)}||g^{G_2(x_0,y_0)}$ in next step. Otherwise, the response will be discarded.

Step 2. Because U_i requests the bivariate polynomial $F(x, y)$, U_i first retrieves the authenticated label $L_{F(x,y)} = g^a||g^b||g^{F(a,b)}$ from the public and authenticated Merkle hash tree, as shown in Fig. 3. After that, U_i uses the following equation to verify the correctness of the result $F(x_0, y_0)$

$$e\left(\frac{g^{F(x_0,y_0)}}{g^{F(a,b)}}, g\right) = e\left(\frac{g^{x_0}}{g^a}, g^{G_1(x_0,y_0)}\right) \cdot e\left(\frac{g^{y_0}}{g^b}, g^{G_2(x_0,y_0)}\right) \tag{21}$$

If it does hold, the result $F(x_0, y_0)$ is accepted; otherwise, $F(x_0, y_0)$ will be discarded. The reason is that, if $F(x_0, y_0)$ is correctly computed, then the relationship

$$F(x_0, y_0) - F(a, b) = (x_0 - a)G_1(x_0, y_0) + (y_0 - b)G_2(x_0, y_0)$$

in Eq. (6) is satisfied. Thus, Eq. (21) should hold as well. Otherwise, if $F(x_0, y_0)$ is wrongly calculated, Eq. (21) cannot hold. With the above steps, each user can obtain his results in a privacy-preserving and verifiable way.

IV. Security Discussion

In this section, we discuss the security and privacy properties of the proposed PLAM framework. In specific, following the design goals discussed early, we examine whether the proposed PLAM framework can achieve the desirable security and privacy requirements.

A. The proposed PLAM framework is privacy-preserving in local-area MSN.

1) Mobile user's preferences are privacy-preserving in the proposed PLAM framework: According to the k-anonymity and l-diversity requirements, the more mobile users are involved and the more services are requested at the same time, the lower probability for the LBSP to link a specific service to a specific user in the proposed privacy-preserving request aggregation protocol. Thus, the user preferences can be protected against the LBSP.

At the same time, when $l = \sum_{i=1}^{n}(a_{1i} \vee a_{2i} \vee \cdots \vee a_{ki})$ is calculated from $l = n - D_{\mathsf{sk}}\left(\sum_{i=1}^{n}\left(\prod_{j=1}^{k} E_{\mathsf{pk}}(1-a_{ji})\right)\right)$, only the number of newly-joining users is larger than 2, i.e., $n_j \geq 2$, then U_1 can only link a new requested service to a newly-joining user with probability $1/n_j$. For example, if $\mathbf{a}_j = (1,0,0,0,1)$, $\mathbf{a}_{j+1} = (0,1,0,1,0)$, $\mathbf{a}_{j+2} = (1,1,0,0,0)$, then Eq. (9) becomes $\prod_{i=j}^{j+2} E_{\mathsf{pk}}(\bar{\mathbf{a}}_i) = (E_{\mathsf{pk}}(0), E_{\mathsf{pk}}(0), E_{\mathsf{pk}}(1), E_{\mathsf{pk}}(0), E_{\mathsf{pk}}(0))$ and $l = 4$. From both $\prod_{i=j}^{j+2} E_{\mathsf{pk}}(\bar{\mathbf{a}}_i)$ and $l = 4$, user U_1 cannot exactly link a requesting service with a user U_j with one hundred percent. Thus, each mobile user preferences can be protected against the mobile user, i.e., U_1.

A possible solution to further improve the user preference privacy preservation is that other users directly compute and send the value $\mathsf{C}_x = \sum_{i=1}^{n}\left(\prod_{j=1}^{k} E_{\mathsf{pk}}(1-a_{ji})\right)$ to U_1. With the value C_x, U_1 can still calculate $l = n - D_{\mathsf{sk}}(\mathsf{C}_x)$ correctly, but cannot use the intermediate result in Eq. (9) to infer other user's preference privacy. Note that, though the collusion is not emphasized in the current version, this solution is also effective to resist against the collusion between U_1 and a small part of mobile users, but it requires all joined users to cooperatively compute a new value C_x each time when a user joins, which becomes not efficient.

We can further notice that, although the LBSP cannot link a specific service to a specific user, yet a mobile user U_i can use the random number $\mathtt{r}_{ij}$ in hand to identify the response $R_{ij} = \mathcal{H}_1(\mathtt{r}_{ij})||\mathcal{E}nc_{\mathtt{r}_{ij}}(\mathtt{r}_{ij}||\mathtt{out}_{ij}||\mathtt{witness}_{ij})$ from the LBSP to him. Therefore, the proposed PLAM framework is also workable while preserving user preference privacy.

2) The LBSP's polynomials are privacy-preserving but the correctness of each polynomial computation can be verified in the PLAM framework: In the proposed PLAM framework, the LBSP won't release the constructions of the polynomials to users, but just outputs the results of polynomial computations. In addition, the polynomials would be also evolved and updated frequently. Thus, it is hard for users to guess the polynomials, and as a result the polynomials of the LBSP are privacy-preserving. At the same time, in order to convince the user on the correctness of the result, the witness technique [9] has been applied. For example, upon receiving the response $R_{ij} = \mathcal{H}_1(\mathtt{r}_{ij})||\mathcal{E}nc_{\mathtt{r}_{ij}}(\mathtt{r}_{ij}||F(x_0,y_0)||g^{G_1(x_0,y_0)}||g^{G_2(x_0,y_0)})$, U_i checks

$$e\left(\frac{g^{F(x_0,y_0)}}{g^{F(a,b)}}, g\right) = e\left(\frac{g^{x_0}}{g^{a}}, g^{G_1(x_0,y_0)}\right) \cdot e\left(\frac{g^{y_0}}{g^{b}}, g^{G_2(x_0,y_0)}\right)$$

If it does hold, user U_i can accept the result $F(x_0, y_0)$. Therefore, the LBSP's polynomials are privacy-preserving but the correctness of each polynomial computation can be verified in the PLAM framework.

3) Mobile user's identity privacy and location privacy are preserved in the proposed PLAM framework: In the proposed PLAM framework, each user U_i uses pseudo-ID PID_{ij} in place of the real identity in mobile social network. Thus, the identity privacy can be achieved. As for the location privacy, since U_i is visiting in $\mathcal{LA}$ and requests the LBS in $\mathcal{LA}$, U_i cannot deny his current location information $\mathcal{LA}$ to the LBSP. However, since U_i can change his unlinkable pseudo-IDs at different locations, although the current location of U_i is disclosed in $\mathcal{LA}$, his past and future location information are unlinkable and can be preserved.

Note that, a strong adversary may be able to correlate a mobile user's pseudo-ID to real identity with some side information (e.g., from cameras) at some specific location. But, it is a hard task for the adversary to collect all side information at all locations that a mobile user visited. Thus, the user's locations are also unlinkable to the strong attacker. In addition, since each pseudo-ID of U_i is formed by $PID_{ij} = \mathcal{E}nc_{s_0}(U_i||r_{ij})$, once there is a dispute occurred, the TA can be involved and use $s_0 = \mathcal{H}_0(s)$ to track user's real identity U_i by decrypting $\mathcal{E}nc_{s_0}(U_i||r_{ij})$ in mobile social network.

B. The proposed PLAM framework also achieves the requirements of source authentication, data integrity, and availability.

In the proposed PLAM framework, once an outside adversary $\mathcal{A}$ wants to inject bogus information in the aggregated request, he should forge the aggregated signature (S_k, T_k). However, since (S_k, T_k) is provably secure in the random oracle model [15], the adversary $\mathcal{A}$ cannot launch this kind of attack. When the LBSP returns a response $R_{ij} = \mathcal{H}_1(\mathtt{r}_{ij})||\mathcal{E}nc_{\mathtt{r}_{ij}}(\mathtt{r}_{ij}||F(x_0,y_0)||g^{G_1(x_0,y_0)}||g^{G_2(x_0,y_0)})$ to U_i, although the LBSP cannot identify U_i, yet U_i can use the random number $\mathtt{r}_{ij}$ to identify the response R_{ij} to him. Because the random number $\mathtt{r}_{ij}$ is only shared between the LBSP and U_i, once an adversary $\mathcal{A}$ modifies the response R_{ij}, U_i can detect it. In addition, because the timestamp $\mathtt{TP}$ is embedded in the request $\mathcal{C}||\mathtt{TP}||PID_1||PID_2||\cdots||PID_k||S_k||T_k$, an adversary $\mathcal{A}$ cannot launch either replay attack or denial of service attack to paralyze the LBSP. Therefore, the proposed PLAM framework is also secure against the outside attacks on source authentication, data integrity, and availability.

V. Performance Evaluation

The essence of the proposed PLAM framework is to apply the privacy-preserving request aggregation with k-anonymity

and l-diversity to preserve user preference privacy. However, this kind of cooperation will inevitably cause some request delay. Therefore, in this section, we study the performance of the proposed PLAM framework in terms of the request delay. According to the privacy-preserving request aggregation protocol design, the request delay comes not only from the computation and communication costs among the users but also from the waiting time to meet the conditions $k \geq th_k$ and $l \geq th_l$. Since the request delay is dominated by the waiting time, we use simulations to examine how the waiting time (a.k.a., request delay) affects the parameters k and l.

A. Simulation Setup

We evaluate the performance of PLAM based on a custom simulator built in Java. The simulator implements the network layer and assumes each mobile user's smartphone has available resources to compute data and communicate with each other. To simulate both dense and sparse mobile social networks, $N = 100$ mobile users with transmission radius of $tr = 50$ are first deployed in a region $S = \{1\text{km}\times1\text{km}, 1.5\text{km}\times1.5\text{km}\}$. Then, with the same velocity $v = \{1, 2\}$ m/s, each user independently moves in the region S by following the random waypoint model. To evaluate the relationship between k-Anonymity, l-Diversity and the waiting time, we assume there is one user, e.g., U_1, stops and broadcasts the "request aggregation" message at time 0. Later, when another user U_i encounters U_1, with probability ρ, U_i will stop and participate in the request aggregation protocol; and with probability $1-\rho$, U_i will continue his own movement. If users participate in the protocol, we assume they have the same number of requesting services to LBSP, i.e., $\texttt{NRS} = \{1, 2, 3, 4\}$, and the requesting services are uniformly distributed in $\mathcal{L} = (L_1, L_2, \cdots, L_n)$ with $|\mathcal{L}| = 50$. The detailed parameters settings are summarized in Table I.

TABLE I
SIMULATION SETTINGS

Parameter	Setting
Simulation area S	$1\text{km} \times 1\text{km}$, $1.5\text{km} \times 1.5\text{km}$
Simulation duration	30 minutes
Number of services provided by LACS	$\lvert\mathcal{L}\rvert = 50$
Mobile user	
number, ρ	$N = 100$, $\rho = 50\%$
velocity, transmission radius	$v = \{1, 2\}$ m/s, $tr = 50$ m
mobility model	random waypoint model
number of requested services	$\texttt{NRS} = \{1, 2, 3, 4\}$

In the following, we conduct the simulations with different parameters. For each case, we run the simulations for 30 minutes, and the average performance results over 10000 runs are reported.

B. Simulation Results

Fig. 4 shows the relationship between the number of participating users k and the waiting time during the request aggregation protocol, and Fig. 5 also depicts the relationship between the number of requesting services l and the waiting time. From the figures, we can see both k and l increase as time increases in all cases. When the velocity is high, k and l also increase quickly. However, when the region S becomes large, the increases of k and l become slow. In addition, Fig. 5 also shows the positive effect of NRS on l. Therefore, in order to balance user preference privacy level and the request delay, we should set proper thresholds th_k and th_l in different scenarios. For example, if the network is dense and high dynamic, we can set a little higher thresholds, because the proposed PLAM framework works well in such network scenarios. However, if the network is sparse and low dynamic, the low thresholds would be more reasonable.

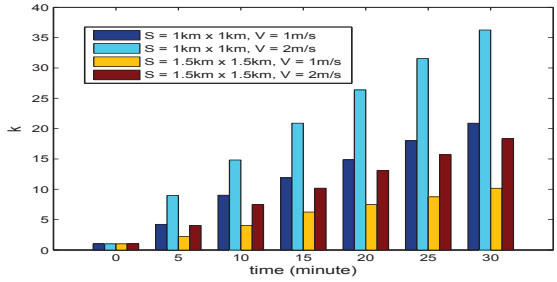


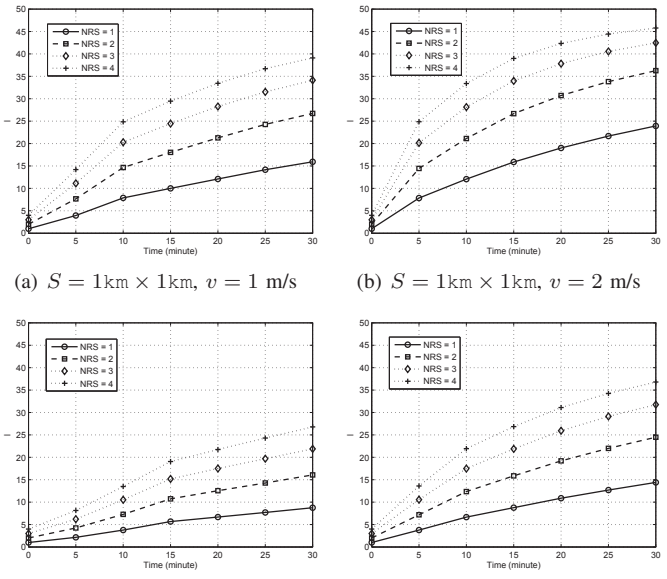
Fig. 4. The number of participating users k versus the waiting time.

Note that, we assume all users have the same number of requested services and each user's requested services are uniformly distributed in $\mathcal{L}$ in the simulations, but the equal number of requested services is not realistic and the uniform distribution may be also not the case in practice. For example, some services may be much popular, they are frequently accessed by users, then the number of requesting service l would be a little low in practice. By further observing Fig. 5, we can see, although l increases with the waiting time, the major increase lies in the first half of the waiting time, i.e., within 15 minutes. Therefore, when we set the thresholds, we should also consider these factors into accounts.

VI. Related Works

Recently, there have appeared several research works on privacy-preserving computing in MSN and verifiable polynomial computation [9], [11]–[13], [16]–[18], which are closely related to the techniques in the proposed PLAM framework.

Privacy-preserving computing in MSN. To support privacy-preserving location query service in MSN, Li and Tung first [16] propose a privacy-preserving distance (PPD) computation protocol, which allows two users to compute their distance while not disclosing their exact coordinates to each other. Then, they extend PPD computation protocol to PPD comparison protocol, which allows two users to privacy-preserving compare whether their distance is less than, equal to, or greater than a threshold value. To secure friend discovery in MSN, Dong et al. [17] and Li et al. [19] propose privacy-preserving scale product protocols, which can be used to compare two users' similarity while not disclosing individual's interest to each other. Although the above privacy-preserving protocols are ingeniously designed for MSN, they are based on Paillier



(a) $S = 1\text{km} \times 1\text{km}$, $v = 1$ m/s (b) $S = 1\text{km} \times 1\text{km}$, $v = 2$ m/s

(c) $S = 1.5\text{km} \times 1.5\text{km}$, $v = 1$ m/s (d) $S = 1.5\text{km} \times 1.5\text{km}$, $v = 2$ m/s

Fig. 5. The number of requesting services l versus the waiting time.

cryptosystem, which makes the computation inefficient when the the vector length is large. To achieve the efficiency, in our previous work [18], we propose a lightweight privacy-preserving scale product protocol, which does not utilize the Paillier cryptosystem, and can improve the efficiency greatly. Different from the the above privacy-preserving computing protocols, the PLAM framework introduces a privacy-preserving union computing protocol based on BVHE [11], which allows a group of users to compute the union of their requesting services but not disclosing individuals' requesting services. Thus, it can be applied to protect user preference privacy while not involving a trusted anonymizer server.

Verifiable polynomial computation. In [12], Kate et al. observe that, for any univariate polynomial $f(x)$, the polynomial $f(x)-f(a)$ is always divisible by $x-a$, i.e., there always exists another polynomial $w(x)$ such that $f(x)-f(a) = (x-a)w(x)$. Based on the observation, Kate et al. propose a publicly verifiable univariate polynomial commitment scheme, which can be widely applied to applications such as verifiable secret sharing, selective disclosure of signed data and credentials, and others. Quite recently, inspired by Kate's work, Papamanthou et al. [9] propose the signature of correct computation (SCC) protocol suitable for multivariate polynomials with optimal updated. In addition, Benabbas et al. [13] also develop methods for efficient verification of multivariate polynomial evaluation by using algebraic on-way functions. Motivated by the core of SCC, the PLAM framework introduces the privacy-preserving and verifiable polynomial computation in LBSP, enabling LBSP to keep the polynomials private but output verifiable computation results to mobile users.

VII. Conclusions

In this paper, we have proposed a privacy-preserving framework for local area MSN, called PLAM. The proposed PLAM framework mainly studies the cooperative privacy-preserving request aggregation with k-anonymity and l-diversity, and privacy-preserving verifiable computing to exploit how to protect both mobile users and LBSP's privacy while enabling mobile users to obtain content-verifiable results from LBSP. Detailed security analysis shows that the proposed PLAM framework can not only achieve user preference, location and identity privacy and LBSP privacy on its private polynomial functions, but also resists against outside attacks on source authentication, data integrity and availability. Moreover, through extensive performance evaluation, we also discuss how to set proper thresholds for k and l to make a tradeoff between desirable preference privacy level and the request delay. In our future work, we will take the collusion attacks into accounts and revise the PLAM framework within the new security model. In addition, since simulation-based evaluation generally cannot capture all factors in reality, we will also plan to carry on the real experiments to further evaluate the effectiveness of the proposed framework.

References

[1] X. Liang, X. Li, T. H. Luan, R. Lu, X. Lin, and X. Shen, "Morality-driven data forwarding with privacy preservation in mobile social networks," *IEEE T. Vehicular Technology*, vol. 61, pp. 3209–3222, 2012.
[2] C. Bettini, S. Mascetti, X. S. Wang, and S. Jajodia, "Anonymity in location-based services: Towards a general framework," in *MDM*, 2007, pp. 69–76.
[3] M. Leitner and A. Curtis, "A first step towards a framework for presenting the location of confidential point data on maps - results of an empirical perceptual study," *International Journal of Geographical Information Science*, vol. 20, no. 7, pp. 813–822, 2006.
[4] M. L. Yiu, C. S. Jensen, X. Huang, and H. Lu, "Spacetwist: Managing the trade-offs among location privacy, query performance, and query accuracy in mobile services," in *ICDE*, 2008, pp. 366–375.
[5] B. Gedik and L. Liu, "Location privacy in mobile systems: A personalized anonymization model," in *ICDCS*, 2005, pp. 620–629.
[6] P. Kalnis, G. Ghinita, K. Mouratidis, and D. Papadias, "Preventing location-based identity inference in anonymous spatial queries," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 12, pp. 1719–1733, 2007.
[7] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, "*L*-diversity: Privacy beyond *k*-anonymity," *TKDD*, vol. 1, no. 1, 2007.
[8] F. Liu, K. A. Hua, and Y. Cai, "Query l-diversity in location-based services," in *Mobile Data Management*, 2009, pp. 436–442.
[9] C. Papamanthou, E. Shi, and R. Tamassia, "Signatures of correct computation," in *TCC*, 2013, pp. 222–242.
[10] D. Boneh and M. K. Franklin, "Identity-based encryption from the weil pairing," in *Proceedings of CRYPTO'01*, 2001.
[11] Z. Brakerski and V. Vaikuntanathan, "Fully homomorphic encryption from ring-lwe and security for key dependent messages," in *CRYPTO*, 2011, pp. 505–524.
[12] A. Kate, G. M. Zaverucha, and I. Goldberg, "Constant-size commitments to polynomials and their applications," in *Advances in Cryptology-ASIACRYPT 2010*. Springer, 2010, pp. 177–194.
[13] S. Benabbas, R. Gennaro, and Y. Vahlis, "Verifiable delegation of computation over large datasets," in *CRYPTO*, 2011, pp. 111–131.
[14] R. C. Merkle, "Secrecy, authentication, and public key systems." 1979.
[15] C. Gentry and Z. Ramzan, "Identity-based aggregate signatures," in *Public Key Cryptography*, 2006, pp. 257–273.
[16] T. Jung and X.-Y. Li, "Search me if you can: Privacy-preserving location query service," *CoRR*, vol. abs/1208.0107, 2012.
[17] W. Dong, V. Dave, L. Qiu, and Y. Zhang, "Secure friend discovery in mobile social networks," in *INFOCOM*, 2011, pp. 1647–1655.
[18] R. Lu, X. Lin, and X. S. Shen, "Spoc: A secure and privacy-preserving opportunistic computing framework for mobile-healthcare emergency," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 3, pp. 614–624, 2013.
[19] M. Li, S. Yu, N. Cao, and W. Lou, "Privacy-preserving distributed profile matching in proximity-based mobile social networks," *IEEE Trans. on Wireless Commun.*, vol. 12, no. 5, pp. 2024–2033, 2013.