sensors

MDPI

Article

A Fine-Grained and Privacy-Preserving Query Scheme for Fog Computing-Enhanced Location-Based Service

Xue Yang *, Fan Yin and Xiaohu Tang

The Information Security and National Computing Grid Laboratory, Southwest Jiaotong University, Chengdu 610031, China; yinfan519@gmail.com (F.Y.); xhutang@swjtu.edu.cn (X.T.)

* Correspondence: xueyang.swjtu@gmail.com; Tel.: +86-136-7808-0943

Received: 31 May 2017; Accepted: 7 July 2017 ; Published: 11 July 2017

Abstract: Location-based services (LBS), as one of the most popular location-awareness applications, has been further developed to achieve low-latency with the assistance of fog computing. However, privacy issues remain a research challenge in the context of fog computing. Therefore, in this paper, we present a fine-grained and privacy-preserving query scheme for fog computing-enhanced location-based services, hereafter referred to as FGPQ. In particular, mobile users can obtain the fine-grained searching result satisfying not only the given spatial range but also the searching content. Detailed privacy analysis shows that our proposed scheme indeed achieves the privacy preservation for the LBS provider and mobile users. In addition, extensive performance analyses and experiments demonstrate that the FGPQ scheme can significantly reduce computational and communication overheads and ensure the low-latency, which outperforms existing state-of-the art schemes. Hence, our proposed scheme is more suitable for real-time LBS searching.

Keywords: location-based services (LBS); fog computing; low-latency; fine-grained; privacy-preserving

1. Introduction

With the rapid development of the Internet of Things (IoTs), fog computing has been presented by Cisco in 2012 [1] as a supplement to cloud computing. Based on its network edge computing feature, fog computing is able to provide location-awareness applications, where the location-based service (LBS) is the most popular [2,3]. With the assistance of fog computing, the LBS providers can outsource the LBS data to fog nodes and thus provide people a convenient life [4]. For instance, through the LBS provided by a local fog node, a mobile user can query and search points of interest (POIs) within a given distance to his/her location [5,6]. Actually, LBS query has become one of the killer applications in LBS, which allows mobile users to query a LBS provider (such as Google or Bing maps) to obtain the detailed information about POIs in their vicinity (e.g., restaurants, hospitals, etc.). For example, when a tourist arrives at a new place, it is very necessary to query some interesting places, such as hotels, restaurants, scenic spots and so on, based on the LBS. Hence, we focus on the LBS query in this paper.

Although LBS can benefit people a lot, the submitted queries may lead to some sensitive information about users being revealed, such as hobbies, current location and the location where mobile users will reach [7,8]. To clearly illustrate the privacy challenge in LBS query, we consider the following scenario, where a mobile user wants to find a hospital by exposing his location to a local fog node. Thus, this fog node can infer user's location where he will reach according to his request (the content of the matched POIs). However, even worse, the health condition that is very sensitive to this user, may be disclosed. Conceivably, without the privacy assurance in fog computing, the LBS can not continue to its flourish because users' personal trajectory and hobby may be exposed.

In addition, owing to the mobility of mobile users, the low-latency is very critical for the LBS query [9]. For example, when a driver wants to find the nearest restaurant on a heavy-traffic road, he has to keep driving while waiting for the response to ensure smooth traffic. If the response is slow, the result may already not satisfy the current spatial range of this diver, i.e., the driver has traveled far away. Therefore, latency should also be taken into account.

Recently, many privacy-preserving LBS schemes have been proposed to prevent the location privacy of LBS users. For examples, location obfuscation technology has been presented to protect users' precise locations from disclosing [10,11]. In addition, spatial cloaking technology has been proposed to conceal the precise location in a cloaking area [12,13]. However, the accuracy of the matched results in these schemes are not satisfactory, since the locations submitted by users are not precise. In order to ensure the data privacy and improve the accuracy of results simultaneously, privacy information retrieval (PIR) [14] has been applied to the LBS query [15,16]. Unfortunately, as mentioned in [17], PIR needs to transmit the entire LBS data to the mobile users, which brings about the heavy communication burden, especially for big data. In addition, a mobile user has to additionally access the LBS database's index data in a privacy-preserving way, which leads to the linear calculations of all LBS data. Hence, the communication and computational costs are tremendous, which is not cost-efficient for mobile users. To overcome this disadvantage, some efficient cryptographic algorithm-based schemes have been developed for the LBS query, where most of these schemes focused on cloud computing, as it can offer on-demand and scalable storage and unlimited computing capacity, especially for big data processing. For examples, Li et al. [18] discussed an efficient privacy-preserving location-based query over outsourced encrypted data by coding and bilinear pairing techniques; Zhu et al. [19] proposed a new efficient and privacy-preserving LBS query scheme in the outsourced cloud; Peng et al. [20] developed the collaborative trajectory privacy-preserving scheme for the trajectory privacy of continuous queries. However, with vast LBS requests generated everyday, the transmission of the extraordinarily huge volume data to the cloud has resulted in unbearable transmission latency, which will degrade the quality of service to end users, especially mobile users [9,21,22]. As a result, privacy-preserving LBS query schemes designed for cloud computing may not satisfy the low-latency.

Hence, the content of fog computing has been introduced to make up the disadvantage of the centralized processing in the cloud [1]. As far as we know, existing schemes in fog computing mainly concentrated on location privacy in wireless sensor networks [23] or real-time navigation of vehicular ad hoc networks [24]. Nevertheless, few fog computing-based schemes considered the privacy issue in the LBS query. Moreover, almost all state-of-the-art LBS query schemes only take the location matching into consideration without considering content matching, and thus this is not fine-grained. For instance, when a mobile user wants to query a hospital within 500 m, the returned results in existing schemes are all the LBS data meeting the spatial range (within 500 m). That is, in addition to the hospital, the result may include a restaurant, a park, or a mall and so on. In fact, these results are not necessary for LBS users, and it will lead to additional communication and computational costs, e.g., users have to decrypt all the returned ciphertexts to find the point of interest.

To not only ensure the privacy-preserving and low-latency, but also achieve the fine-grained query result, in this paper, we propose a fine-grained and privacy-preserving query scheme for fog computing-enhanced location-based service (FGPQ). The contributions of this paper are threefold as follows.

- First, we present our FGPQ scheme, which is characterized by employing the bilinear pairing [25] and the asymmetric scalar-product preserving encryption (ASPE) [26] to realize the LBS searching. In addition to satisfying the given spatial range, the searching result satisfies the given searching content, which is not considered in many up-to-date schemes.
- Secondly, we give detailed privacy analysis to show that our proposed FGPQ indeed achieves the privacy preservation of both the LBS provider and mobile users.

- Finally, we theoretically analyze the computational and communication overheads, and run extensive experiments to demonstrate that our FGPQ scheme is more efficient than the EPLQ scheme [18] and EPQ scheme [19]. In addition, the latency analysis shows that our FGPQ scheme is really low-latency, which is suitable for the real-time LBS query.

The remainder of this paper is organized as follows. In Section 2, we introduce our system model and design goals. Then, we describe some preliminaries in Section 3. In Section 4, we present our FGPQ scheme, followed by privacy analysis and performance analysis in Section 5 and Section 6, respectively. Finally, we draw our conclusions in Section 7.

2. System Model and Design Goals

In this section, we formalize the system model used in this paper, and identify design goals.

2.1. System Model

We illustrate an overall architecture of fog computing-enhanced LBS query in Figure 1, which includes a set of mobile users, a set of fog nodes $\{id_1, id_2, \ldots, id_n\}$ and an LBS provider.

- LBS provider: the LBS provider acts as a profit company, providing the location-based services for the registered mobile users. With the advantages of fog computing, the LBS provider prefers to outsource its LBS data containing service content and the corresponding geographic location to appropriate fog nodes based on the geographic location distance, which can provide the low-latency LBS for mobile users.
- Fog nodes $\{id_1, id_2, \ldots, id_n\}$: with the pay-as-you-use way, each fog node id_j stores the LBS data from the LBS provider and provides the fine-grained query services for mobile users.
- Mobile users: a mobile user who acts as a registered member of the LBS provider, sends a query that contains the searching content, current location information and searching spatial range, to a local fog node for requesting the nearby POIs satisfying both the given searching content and spatial range.

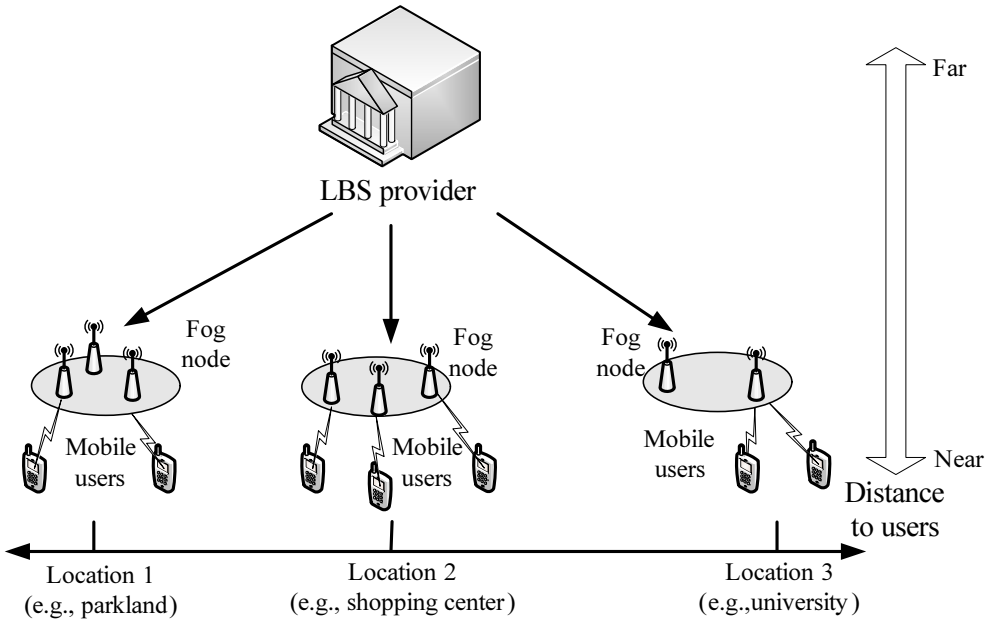


Figure 1. System model under consideration.

Similar to the most common assumption in research literature (see [27,28]), fog nodes are considered honest-but-curious, which honestly follow the underlying scheme, but are curious about

the privacy of mobile users and the LBS provider. In the LBS applications, the LBS provider or mobile users would not collude with fog nodes to obtain others' privacy. This is because if the LBS provider agree to collude with a fog node to obtain the content of a user' request, then this node can obtain some LBS data considered as provider's privacy from the LBS searching. For example, if a fog node obtain a mobile user's request (e.g., a hospital within 500 m) from the collusion attack, then it can obtain the real information of the matched LBS data, i.e., it knows that the service content of all matched LBS data is the hospital and the corresponding location is within 500 m of the current location. In other words, once the privacy of mobile users is disclosed, the privacy of the LBS provider will also be disclosed. Therefore, the LBS provider would not collude with fog nodes. Similarly, mobile users would also not collude with fog nodes. Note that, since the privacy preservation is our focus, some active attacks are beyond the scope of this work and will be discussed in the future.

2.2. Design Goals

The goals of our proposed FGPQ scheme are described as follows:

- Privacy preservation. As a profit company, the LBS data are considered as the LBS provider's own asset, which should be protected from disclosing. Therefore, the LBS data should be encrypted before being outsourced to fog nodes. For mobile users, a service query may contain some sensitive information, e.g., hobbies, current location and the location where mobile users will reach, which should also be protected from disclosing. Hence, mobile users should send the encrypted query request to the nearby fog node.
- Fine-grained query result. Besides the searching spatial range, the FGPQ scheme should satisfy the searching content, e.g., a hospital or a restaurant. That is, the query result should satisfy the given searching content and searching spatial range simultaneously.
- Efficiency. Owing to the mobility of mobile users, the low-latency is very critical for the LBS [9]. Hence, computational costs and communication delay should be as less as possible.

3. Preliminaries

In this section, we first outline the bilinear pairing [25], which will be used for generating the index of the service content to complete the content matching. Then, we describe the asymmetric scalar-product preserving encryption algorithm (ASPE) [26], which will be applied to achieve the encrypted location matching.

3.1. Bilinear Pairing

Given a security parameter $\kappa \in \mathbb{Z}^+$, a bilinear-parameter generation algorithm $\mathcal{G}(\kappa)$ outputs a tuple $(p, g, \mathbb{G}, \mathbb{G}_T, e)$, where p is a κ-bit prime number, $\mathbb{G}$ and $\mathbb{G}_T$ are two multiplicative cyclic groups of order p, g is a generator of $\mathbb{G}$, and $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a bilinear pairing with the following properties:

- Bilinearity: for all $u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p^*$, we have $e(u^a, v^b) = e(u, v)^{ab}$.
- Non-degeneracy: $e(g, g) \neq 1$.

The definitions of bilinear pairing parameter generator and the related complexity assumptions are described below [25,29].

Definition 1. *(Bilinear generator) The bilinear parameter generator $Gen(\cdot)$ is a probabilistic algorithm that takes a security parameter κ as input and outputs a* 5*-tuple $(\mathbb{G}, \mathbb{G}_T, p, g, e)$.*

Definition 2. *(Computational Diffie–Hellman (CDH) problem) Given $(g, g^a, g^b) \in \mathbb{G}$, for unknown $a, b \in \mathbb{Z}_p^*$, compute $g^{ab} \in \mathbb{G}$.*

Definition 3. *(Discrete logarithm (DL) problem) Given $Q \in \mathbb{G}$, compute $a \in \mathbb{Z}_p^*$ such that $g^a = Q$.*

3.2. The Asymmetric Scalar-Product Preserving Encryption (ASPE)

Wong et al. [26] developed an asymmetric scalar-product preserving encryption (ASPE), which has been widely applied to the outsourced range matching in the ciphertext [30,31]. Hence, this encryption algorithm can also applied to the outsourced spatial range query in the ciphertext.

- Key generation ($Gen(d)$). Given a security parameter d, two $d \times d$ invertible matrices M_1 and M_2, and a d-dimensional binary vector S are chosen as the private key, denoted as $sk = (S, M_1, M_2)$. Note that the binary vector S is a splitting indicator to split the plaintext vector into two random vectors, where $S[i]$ is the i-th bit in S.
- Tuple encryption function $E_T(\cdot)$. Consider a d-dimensional vector P in a database. Firstly, split P into two d-dimensional vectors (P_a, P_b) based on the splitting indicator S. Specifically, if $S[i] = 0$ $(i = 1, 2, \ldots, d)$, $P_a[i] = P_b[i] = P[i]$; if $S[i] = 1$ $(i = 1, 2, \ldots, d)$, the value of $P[i]$ will be randomly split into $P_a[i]$ and $P_b[i]$ such that $P[i] = P_a[i] + P_b[i]$. Then, the encrypted value of P can be calculated as
$$E_T(P, sk) = (P_a M_1, P_b M_2). \tag{1}$$
- Query encryption function $E_Q(\cdot)$. Consider a d-dimensional vector Q, split it into two d-dimensional vectors (Q_a, Q_b): if $S[i] = 0$ $(i = 1, 2, \ldots, d)$, the value of $Q[i]$ can be randomly split into $Q_a[i]$ and $Q_b[i]$ with $Q[i] = Q_a[i] + Q_b[i]$; if $S[i] = 1$ $(i = 1, 2, \ldots, d)$, $Q_a[i] = Q_b[i] = Q[i]$. The encrypted value of Q can be generated as
$$E_Q(Q, sk) = (M_1^{-1} Q_a^{\mathrm{T}}, M_2^{-1} Q_b^{\mathrm{T}}), \tag{2}$$
where Q_a^{T} and Q_b^{T} denote the transpose of Q_a and Q_b, respectively.
- Outsourced scalar-product calculation. With two ciphertexts $E_T(P, sk)$ and $E_Q(Q, sk)$, the outsourced scalar-product can be calculated as
$$\begin{aligned} E_T(P, sk) \cdot E_Q(Q, sk) &= (P_a M_1, P_b M_2) \cdot (M_1^{-1} Q_a^{\mathrm{T}}, M_2^{-1} Q_b^{\mathrm{T}}) \\ &= P_a M_1 \cdot M_1^{-1} Q_a^{\mathrm{T}} + P_b M_2 \cdot M_2^{-1} Q_b^{\mathrm{T}} \\ &= P_a \cdot Q_a^{\mathrm{T}} + P_b \cdot Q_b^{\mathrm{T}} \\ &= P \cdot Q. \end{aligned}$$
The correctness of the scalar-product calculation can be referred to [26].
- Decryption Function $D(\cdot)$. Consider an encrypted value $(P_a M_1, P_b M_2)$. Firstly, compute the inverse matrices M_1^{-1} and M_2^{-1}, and then extract two vectors P_a and P_b, i.e., $P_a = P_a M_1 \cdot M_1^{-1}$, $P_b = P_b M_2 \cdot M_2^{-1}$. Finally, recover the original d-dimensional vector P with the splitting indicator S:
$$P[i] = \begin{cases} P_a[i] = P_b[i], & if \quad S[i] = 0, \\ P_a[i] + P_b[i], & if \quad S[i] = 1. \end{cases}$$

The Properties of ASPE Algorithm: For any encrypted tuple $E_T(P_i, sk)$ and an encrypted query $E_Q(Q, sk)$, the ASPE algorithm satisfies the following two properties:

- For any P_i encrypted by $E_T(\cdot)$ and any query Q encrypted by $E_Q(\cdot)$:
$$P_i \cdot Q = E_T(P_i, sk) \cdot E_Q(Q, sk). \tag{3}$$
- For any P_i and P_j encrypted by $E_T(\cdot)$:
$$P_i \cdot P_j \neq E_T(P_i, sk) \cdot E_T(P_j, sk). \tag{4}$$

Note that the actual dimension d may be too small to resist the brute-force attack. Therefore, it is better to improve the system security by making d larger, which can be achieved by adding *artificial* dimensions to the data vector. In particular, extend a d-dimensional vector to a d'-dimensional ($d \leq d'$) by padding artificial data such that the scalar-product over the added data is 0. In addition, $d' \geq 80$ is large enough to make the system security. The details can be referred to [26]. Unless other specification, the encryption $E_T(P, sk)$ and $E_Q(Q, sk)$ are denoted by $E_T(P)$ and $E_Q(Q)$ for short in the rest of this paper, respectively.

4. Proposed FGPQ Scheme

This section presents a fine-grained and privacy-preserving query scheme for fog computing-enhanced LBS (FGPQ), which mainly consists of two parts: system initialization and privacy-preserving LBS query. In particular, the LBS provider mainly initializes the LBS system in the system initialization part, which includes the generation of public and private keys, and LBS data encryption. The privacy-preserving LBS query completes the LBS query process, which includes query request generation, LBS searching and result retrieval. The details are shown as follows.

4.1. System Initialization

The LBS provider initializes the system by following steps:

1. Key generation. The LBS provider generates the public and private keys, which are used to encrypt the LBS data and the query request. Specifically, the LBS provider first generates the bilinear pairing tuple $(p, g, \mathbb{G}, \mathbb{G}_T, e)$ and the ASPE algorithm's private key (S, M_1, M_2), respectively. Then, it chooses a large random number $\alpha \in \mathbb{Z}_p^*$ as the master-key, and a cryptographic hash function $h : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$. In addition, in our proposed scheme, the actual dimension d of the ASPE algorithm is 7, which can not ensure the sufficient security; thus, we should extend d to $d' = 80$. To this end, the LBS provider also chooses $(d' - d)$ random numbers $\omega_{d+1}, \omega_{d+2}, \ldots, \omega_{d'}$ as the private key of the ASPE algorithm. After that, the LBS provider publishes $(p, g, \mathbb{G}, \mathbb{G}_T, e, h)$ as the public key. Once a mobile user is registered in the LBS provider, the LBS provider will assign the private key $(\alpha, S, M_1^{-1}, M_2^{-1}, \omega_{d+1}, \ldots, \omega_{d'})$ to mobile users through the secure channel.
2. LBS data encryption. As mentioned in [18,19], the LBS provider has all LBS data in the system, which can be denoted as $\{(m_1, (X_1, Y_1)), \ldots, (m_k, (X_k, Y_k))\}$. In particular, m_i and (X_i, Y_i) denote the service content and the corresponding location for i-th service, respectively. In order to protect the privacy, the LBS provider encrypts the LBS data and separately sends them to appropriate fog nodes. Concretely, for each service item $(m_i, (X_i, Y_i))$, where $i = 1, 2, \ldots, k$, the LBS provider conducts the following calculations.

    - For the service content m_i, the LBS provider chooses a random number $r_i \in \mathbb{Z}_p^*$, and generates the ciphertext as
    $$E(m_i) = (g^{\alpha h(m_i) r_i}, g^{r_i}).$$
    - For the location information (X_i, Y_i), the LBS provider first generates a 7-dimensional vector as $W^{(i)} = (1, 1, X_i^2, Y_i^2, -2X_i, -2Y_i, -1)$, and extends it to a d'-dimensional vector $\widehat{W^{(i)}}$, where the first seven dimensions are copied from $W^{(i)}$, and for $j = 8$ to $d' - 1$, set $\widehat{W^{(i)}}[j] = r_j$ (r_j is a random number), and $\widehat{W^{(i)}}[d'] = \frac{-\sum_{j=8}^{d'-1} \omega_j r_j}{\omega_{d'}}$, i.e., $\widehat{W^{(i)}} = (1, 1, X_i^2, Y_i^2, -2X_i, -2Y_i, -1, r_8, r_9, \ldots, r_{d'-1}, \frac{-\sum_{j=8}^{d'-1} \omega_j r_j}{\omega_{d'}})$. Then, the LBS provider encrypts $\widehat{W^{(i)}}$ with the private key (S, M_1, M_2) by means of the tuple encryption function $E_T(\cdot)$ (the Equation (1)), that is,
    $$E_T(\widehat{W^{(i)}}) = (\widehat{W_a^{(i)}} \cdot M_1, \widehat{W_b^{(i)}} \cdot M_2),$$

where $\widehat{W_a^{(i)}}$ and $\widehat{W_b^{(i)}}$ are two split vectors of $\widehat{W^{(i)}}$.

Finally, based on the geographic location distance, the LBS provider outsources all k encrypted LBS data $(E(m_i), E_T(\widehat{W^{(i)}}))$, where $i = 1, 2, \ldots, k$, to appropriate fog nodes.

4.2. Privacy-Preserving LBS Query

This section achieves the privacy-preserving LBS query by means of the bilinear pairing and the ASPE algorithm, which includes LBS query request generation, LBS searching and request result decryption. Before that, we first define a range query used for location range matching.

Definition 4. *Given the location coordinate (x_*, y_*) and the searching spatial range T, a location coordinate (x_i, y_i) is within the searching spatial range if and only if*

$$R = \sqrt{(x_* - x_i)^2 + (y_* - y_i)^2} \leq T, \tag{5}$$

where R is the Euclidean distance used to compute the distance between two coordinates [32].

4.2.1. LBS Query Request Generation

When a mobile user wants to search for a POI within a certain range, e.g., a hospital within 1000 m of the current location, he should send the query request that contains the searching content m_* (a hospital), the current location (X_*, Y_*) and the searching spatial range T_* (within 1000 m), to a local fog node (denoted as id_j). In order to ensure the privacy, he encrypts the request information $(m_*, (X_*, Y_*), T_*)$ firstly, and then sends the encrypted query request to the fog node id_j. The details are described as follows.

1. For the searching content m_*, the user chooses a random number $r_* \in \mathbb{Z}_p^*$, and generates the ciphertext as
$$E(m_*) = (g^{\alpha h(m_*) r_*}, g^{r_*}).$$
2. For the location information (X_*, Y_*) and the searching spatial range T_*, the user first generates a 7-dimensional vector as $L = (X_*^2, Y_*^2, 1, 1, X_*, Y_*, T_*^2)$, and extends it to a d'-dimensional vector $\widehat{L}$ where the first seven dimensions are copied from L, and for $i = 8$ to d', set $\widehat{L}[i] = \omega_i$, i.e., $\widehat{L} = (X_*^2, Y_*^2, 1, 1, X_*, Y_*, T_*^2, \omega_8, \omega_9, \ldots, \omega_{d'})$. Then, the user chooses a random large positive number β to confuse $\widehat{L}$, and computes $\widehat{L}' = \beta \cdot \widehat{L}$. After that, the user encrypts $\widehat{L}'$ with the private key (S, M_1^{-1}, M_2^{-1}) by means of the query encryption function $E_Q(\cdot)$ (the Equation (2)) as
$$E_Q(\widehat{L}') = (M_1^{-1} \cdot \widehat{L}'_a,\ M_2^{-1} \cdot \widehat{L}'_b),$$
where $\widehat{L}'_a$ and $\widehat{L}'_b$ are two column vectors split from $\widehat{L}'$.

Finally, the user sends the encrypted query request $\{E(m_*), E_Q(\widehat{L}')\}$ to the fog node id_j.

4.2.2. LBS Searching

After receiving the encrypted query request $\{E(m_*), E_Q(\widehat{L}')\}$, the fog node id_j completes the LBS searching to find the most matched LBS item, which not only meets both the searching content and spatial range, but also has the shortest distance from the user's location. Specifically, suppose the fog node id_j receives k_j encrypted LBS items from the LBS provider, denoted as $\{(E(m_1), E_T(\widehat{W^{(1)}})), (E(m_2), E_T(\widehat{W^{(2)}})), \ldots, (E(m_{k_j}), E_T(\widehat{W^{(k_j)}}))\}$. For each $(E(m_i), E_T(\widehat{W^{(i)}}))$, where $i = 1, 2, \ldots, k_j$, the fog node id_j executes the following operations.

1. For $E_T(\widehat{W^{(i)}})$, the fog node id_j first computes

$$\begin{aligned} R_i &= E_T(\widehat{W^{(i)}}) \cdot E_Q(\widehat{L}') = (\widehat{W_a^{(i)}} M_1,\ \widehat{W_b^{(i)}} M_2) \cdot (M_1^{-1}\widehat{L}'_a,\ M_2^{-1}\widehat{L}'_b) \\ &= \widehat{W_a^{(i)}} M_1 \cdot M_1^{-1}\widehat{L}'_a + \widehat{W_b^{(i)}} M_2 \cdot M_2^{-1}\widehat{L}'_b \\ &= \widehat{W_a^{(i)}}\widehat{L}'_a + \widehat{W_b^{(i)}}\widehat{L}'_b \\ &= \widehat{W^{(i)}} \cdot \widehat{L}' \\ &= \beta((X_* - X_i)^2 + (Y_* - Y_i)^2 - T_*^2 + \omega_8 r_8 + \cdots + \omega_{d'-1} r_{d'-1} - \sum_{j=8}^{d'-1} \omega_j r_j) \\ &= \beta((X_* - X_i)^2 + (Y_* - Y_i)^2 - T_*^2). \end{aligned} \tag{6}$$

Then, it checks whether $R_i \leq 0$ holds. If not, it means that the LBS item $(E(m_i), E_T(\widehat{W^{(i)}}))$ does not satisfy the searching spatial range.

2. If $R_i \leq 0$ holds, similarly to [33], the fog node id_j checks whether

$$e(g^{\alpha h(m_i) r_i}, g^{r_*}) = e(g^{\alpha h(m_*) r_*}, g^{r_i}) \tag{7}$$

holds. If not, it means that the service content m_i does not satisfy the given searching content m_*. Otherwise, the LBS item $(m_i, (X_i, Y_i))$ satisfy the searching condition given by mobile users.

Eventually, the fog node id_j sends the matched result $E_T(\widehat{W^{(i)}})$ to the user. Note that if the service m_* has multi-matched locations, the fog node id_j always returns the nearest location from the user (the location with the smallest R_i).

4.2.3. Request Result Decryption

After receiving the result, e.g., $E_T(\widehat{W^{(i)}}) = (\widehat{W_a^{(i)}} M_1,\ \widehat{W_b^{(i)}} M_2)$, the user recovers the vector $\widehat{W^{(i)}}$ by means of the decryption function D, and then extracts the first seven dimensions in it, i.e., $W^{(i)} = (1, 1, X_i^2, Y_i^2, -2X_i, -2Y_i, -1)$. According to $W^{(i)}$, the user can obtain the matched location (X_i, Y_i) about the searching content m_*.

5. Privacy Analysis

Following our design goals, we discuss how the proposed FGPQ scheme achieves the privacy preservation of the LBS provider and mobile users.

The privacy preservation of mobile users: When a mobile user wants to search the POI satisfying the special spatial range and searching content, he or she sends an encrypted request $(E(m_*), E_Q(\widehat{L}'))$ to the fog node id_j. In particular, the ciphertext $E(m_*)$ is computed as $E(m_*) = (g^{\alpha h(m_*) r_*}, g^{r_*})$. Hence, based on hard problems of the CDH and DL in bilinear pairing (Definitions 2 and 3), the fog node id_j can not obtain any plaintext information from $E(m_*)$ without knowing r_* and α. As $E_Q(\widehat{L}')$ is a ciphertext of the ASPE algorithm, the privacy-preserving totally depends on the security of the ASPE algorithm. As analysed in [26], once the vector dimension d satisfies $d \geq 80$, it is generally very difficult for an attacker to successfully attack the ASPE algorithm. Therefore, under the parameter setting in Section 4.1, the location can be protected from disclosing. In other words, fog nodes can not obtain any plaintext information from the request $(E(m_*), E_Q(\widehat{L}'))$. In addition, even if the value of scalar-product (see Formula (6)) has been obtained, the fog node id_j cannot get the location, since a large random number β has been applied to confuse the real scalar-product value. Note that fog nodes do not collude with the LBS provider in this paper, so they will not directly forward users' encrypted requests to the LBS provider, thereby, the LBS provider will not obtain users' privacy. Consequently, the privacy-preserving of mobile users can be achieved.

The privacy preservation of the LBS provider: Under the aforementioned scheme, the LBS provider would encrypt all LBS data before outsourcing. Thus, fog nodes can only obtain the encrypted LBS items. Similarly, the fog node id_j can not obtain any plaintext information from the stored content information $E(m_i)$ $(i = 1, 2, \ldots, k_j)$ based on the CDH and DL problems. In addition, according to properties of the ASPE algorithm (Formulas (3) and (4)), even though the fog node id_j manages k_j encrypted LBS items, it can not obtain any plaintext information about locations. Conceivably, the privacy preservation of the LBS provider can be achieved.

6. Performance Analysis

In this section, we evaluate our proposed FGPQ scheme in terms of the communication overhead, computational costs and latency, respectively. Moreover, we give a comparison with the EPLQ scheme [18] and EPQ scheme [19].

6.1. Computational Costs

In our FGPQ scheme, the LBS provider needs to encrypt k LBS data, where each service content m_i is encrypted by the bilinear pairing technology, i.e., $e(g^{\alpha h(m_i) r_i}, g^{r_i})$, and the correponding location (X_i, Y_i) is encrypted by the tuple encryption function E_T of the ASPE algorithm. According to the empirical evaluation in [26,30], the computational costs for the ASPE algorithm are microseconds, which are considered negligible compared to the exponentiation and pairing operations. Hence, the computational costs of the LBS provider are $2k$ exponentiations in $\mathbb{G}$. When a mobile user wants to request the LBS searching, he or she encrypts the query request $(m_*, (X_*, Y_*), T_*)$, which costs two exponentiations in $\mathbb{G}$. After receiving the encrypted request, for each encrypted LBS item $(E(m_i), E_T(\widehat{W^{(i)}}))$, where $i = 1, 2, \ldots, k_j$, the fog node id_j first computes the Formula (6) to select the LBS items meeting the spatial range. After retrieving f LBS items matched the spatial range, it verifies the Equation (7), where each verification requires two pairing operations. Since the computational costs of the ASPE algorithm are negligible, the computational costs of the fog node id_j are $2f$ pairing operations. Finally, the total computational costs for the LBS provider, a mobile user and a fog node id_j will be $2k$ exponentiations in $\mathbb{G}$, two exponentiations in $\mathbb{G}$ and $2f$ pairing operations, respectively, in the proposed FGPQ scheme.

In the EPLQ scheme [18], to generate a query, a mobile user needs to encode the location and the searching spatial range into two N-dimensional vectors, and then encrypt these two vectors as $K_i = (g^{u''_{i,1}}, g^{u''_{i,2}}, \ldots, g^{u''_{i,N}}, e(g,g)^{h_i})$, where $(u''_{i,1}, u''_{i,2}, \ldots, u''_{i,N})$ is the encoded vector. Hence, the corresponding computational costs are about $2N$ exponentiations in $\mathbb{G}$. In order to ensure the privacy, the LBS provider also needs to encode the location of each LBS into a N-dimensional vectors, and then encrypt it as $C_j = (g^{v''_{j,1}}, g^{v''_{j,2}}, \ldots, g^{v''_{j,N}}, e(g,g)^{s_j})$, which requires N exponentiations in $\mathbb{G}$ and one exponentiation in $\mathbb{G}_T$. For all k LBS items, it totally spends $N * k$ exponentiations in $\mathbb{G}$ and k exponentiations in $\mathbb{G}_T$ to complete the encryption. In addition, to assist the cloud in checking the spatial range, the LBS provider still needs to compute $\tau_2 - \tau_1$ values $\Omega_i = Hash(e(g,g)^{(\alpha \times perm(i) + \beta)^d})$, where $i \in [\tau_1, \tau_2]$. That is, the computational costs are about $(\tau_2 - \tau_1)$ exponentiations in $\mathbb{G}_T$. Hence, the total computational costs for the LBS provider are about $N * k$ exponentiations in $\mathbb{G}$ and $k + (\tau_2 - \tau_1)$ exponentiations in $\mathbb{G}_T$. To determine whether the encrypted LBS items match the given query or not, it requires the cloud to compute $check(K_i, C_j)$, which costs N pairing operations and N multiplications in $\mathbb{G}_T$. Note that searching $\hat{ss}$-tree to find the matched LBS items requires traversing through about $\log k + f$ tree nodes with k LBS items and f items matched the spatial range. Therefore, the total computational costs for the cloud are about $(\log k + f)N$ pairing operations and $(\log k + f)N$ multiplications in $\mathbb{G}_T$.

In the EPQ scheme [19], the LBS provider outsources all encrypted LBS items to the cloud, where each location index is encrypted as $l_{s1} = e(g,g)^{q_1 {x_{s_0}}^2}$, $l_{s2} = e(g,g)^{q_1 {y_{s_0}}^2}$, $l_{s3} = g^{x_{s_0}} \cdot h^{r_{s_1}}$ and $l_{s4} = g^{x_{s_0}} \cdot h^{r_{s_2}}$. The corresponding computational costs for all k LBS items are about $4k$ exponentiations

in $\mathbb{G}$, $2k$ exponentiations in $\mathbb{G}_T$ and $2k$ multiplications in $\mathbb{G}$. To request a LBS query, a mobile user generates an encrypted query $(rq_1, rq_2, rq_3, rq_4) = (e(g,g)^{q_1({x_{s_0}}^2 - d^2)}, e(g,g)^{q_1 {y_0}^2}, g^{q_1 \cdot 2x_0}, g^{q_1 \cdot 2y_0})$, which requires two exponentiations in $\mathbb{G}$ and two exponentiations in $\mathbb{G}_T$. After receiving the query from the user, the cloud computes the search criteria T_s for each LBS item stored in it, which requires $2k$ pairing operations and $5k$ multiplications in $\mathbb{G}_T$.

We present the computational costs comparison of our FGPQ scheme, the EPLQ scheme [18] and the EPQ scheme [19] in Table 1, where k and f denote the number of LBS data items and the matched LBS data items, respectively. Obviously, k and f satisfy $f \leq k$, and the codeword length N satisfies $N \geq 7$ [18]. Note that C_e, C_{et}, C_p, C_{mt} and C_m denote the computational costs of an exponentiation in $\mathbb{G}$, an exponentiation in $\mathbb{G}_T$, a pairing operation, a multiplication in $\mathbb{G}_T$ and a multiplication in $\mathbb{G}$, respectively. From the table, it is obvious that the FGPQ scheme largely reduces the computational costs compared to the EPLQ scheme [18] and the EPQ scheme [19]. Moreover, for mobile users, the efficiency of our FGPQ scheme is better than the others. Thus, as declared in [18,19], it is convincing that our FGPQ scheme is more suitable for energy-poor mobile devices.

Table 1. Comparison of computational costs.

	Mobile User	**LBS Provider**	**Cloud/Fog Node**
EPLQ [18]	$2N \cdot C_e$	$N \cdot k \cdot C_e + (k + \tau_2 - \tau_1) \cdot C_{et}$	$(\log k + f)N \cdot (C_p + C_{mt})$
EPQ [19]	$2 \cdot (C_e + C_{et})$	$4k \cdot C_e + 2k \cdot (C_{et} + C_m)$	$2k \cdot C_p + 5k \cdot C_{mt}$
FGPQ	$2 \cdot C_e$	$2k \cdot C_e$	$2f \cdot C_p$

6.2. Communication Overhead

In our FGPQ scheme, the communication overhead can be divided into two parts: LBS provider-to-fog and user-to-fog. We first discuss the LBS provider-to-fog communication, where the LBS provider delivers all encrypted LBS items to each fog node id_j $(j = 1, 2, \ldots, n)$. The encrypted LBS item is in the form of $\{E(m_i), E_T(\widehat{W^{(i)}})\}_{i=1,2,\ldots,k_j}$ for the fog node id_j, and its size should be $(1024 + |E_T(\widehat{W^{(i)}})|)k_j$ bits if we choose 512-bit $\mathbb{G}$. As mentioned in [30], each dimension in the ASPE algorithm is a float number (the size of each float number is 32 bits). Thus, the total size of $E_T(\widehat{W^{(i)}}) = (\widehat{W_a^{(i)}} M_1, \widehat{W_b^{(i)}} M_2)$ is 5120 bits if the dimension is 80 (see Section 4.1). Accordingly, the communication overheads for LBS provider-to-fog communication are $\sum_{j=1}^{n}(6144 \cdot k_j)$ bits. As mentioned in Section 4.2.2, since all k encrypted LBS items are distributed to n fog nodes, we can obtain $\sum_{j=1}^{n} k_j = k$, where k_j denotes the number of fog node id_j's LBS items received from the LBS provider. That is, the value $\sum_{j=1}^{n}(6144 \cdot k_j)$ can be implicitly simplified as $(6144 \cdot k)$. Next, we consider the user-to-fog communication. Firstly, a mobile user sends an encrypted request $\{E(m_*), E_Q(\widehat{L'})\}$ to a fog node, which requires 6144 bits to transmit. Then, this fog node performs the LBS searching and returns the most matched encrypted location $E_T(\widehat{W^{(i)}})$ to this user, which costs 5120 bits. Therefore, the total communication overheads for user-to-fog communication are about 11,264 bits.

In the EPLQ scheme [18], the LBS provider outsources all encrypted LBS items to the cloud. The communication overheads are $(N+1)k \cdot 512 + k \cdot 128 + 128 \cdot (\tau_2 - \tau_1)$ bits if the bit length of p, standard symmetric encryption (e.g., AES) and the hash value $hash()$ satisfy $|p| = 512$ bit, $|AES| = 128$ bit and $|hash()| = 128$ bit, respectively. To achieve the LBS searching, a mobile user sends two encrypted tokens to the cloud, which requires $2(N+1) \cdot 512$ bits. After receiving the LBS searching request, the cloud performs LBS checking to find the LBS items matched the given spatial range. After that, the cloud returns the matched LBS data items to this user. If the number of matched LBS items is f, then the communication overheads are about $f \cdot 128$ bits, where each ciphertext of the symmetric encryption is 128 bits. Therefore, the LBS provider-to-cloud communication and the user-to-cloud communication require $(512 \cdot (N+1)k + 128 \cdot k + 128 \times 10^8)$ bits and $(1024 \cdot (N+1) + 128 \cdot f)$ bits, respectively.

In the EPQ scheme [19], in order to enjoy the location-based services, a mobile user sends an encrypted LBS query request to the cloud. The encrypted LBS query request is in the form of $(ID_{LBS}||E_{LQR}||U_i||TS||Sig_i)$, and its size is about 5280 bits if we set $\mid ID_{LBS}||U_i||TS \mid = 160$ bits. To provide the better quality of services, the LBS provider outsources all encrypted LBS data to the cloud. Since each outsourced LBS item is in the form of $(ID_{LBS}||T_j||l_{s_1}||l_{s_2}||l_{s_3}||l_{s_4}||E_s)$, the communication overheads for all k LBS items are about $4384 \cdot k$ bits. In addition, in order to help the cloud to complete the query in the ciphertexts, the LBS provider also sends the evaluation data set $EDS = \{ED_0, ED_1, \ldots, ED_i, \ldots, ED_{10,000,000,000}\}$ to the cloud, where $ED_i = H(PB^i)$. If the bit length of hash $H()$ is set as $|H()| = 128$ bit, then the size of the set EDS is about 128×10^{10} bits. Hence, the communication overheads for the LBS provider are about $(4384 \cdot k + 128 \times 10^{10})$ bits. After finding f matched LBS items, the cloud returns the query result $(E_{rq_1}(TRL)||ID_{CS}||TS||Sig_{CS})$ to the user, which requires $(128 \cdot f + 1184)$ bits to transmit. Hence, the total communication overheads between a mobile user and the cloud are about $(128 \cdot f + 6464)$ bits.

We present the communication overhead comparison of our FGPQ scheme, the EPLQ scheme [18] and the EPQ scheme [19] in Table 2, where "LBS provider-to-cloud (fog nodes)" and " User-to-cloud (a fog node) " denote the communication between the LBS provider and the cloud (or fog nodes) and the communication between a mobile user and the cloud (or a fog node), respectively. As described in the EPLQ scheme [18], the codeword length N satisfies $N \geq 7$, since the original dimension of the location vector is 7. Therefore, from Table 2, we can know that the communication overheads between the LBS provider and the cloud (or fog nodes) in these three schemes are about the same. However, in the user-to-cloud (a fog node) communication, the communication overhead of our FGPQ scheme is much less than that of other two schemes, as the number of LBS items matched the given spatial range is usually more than a few dozen, i.e., f is more than a few dozen, and f is likely to increase with k or the searching spatial range increasing.

Table 2. Comparison of communication overhead (Bits).

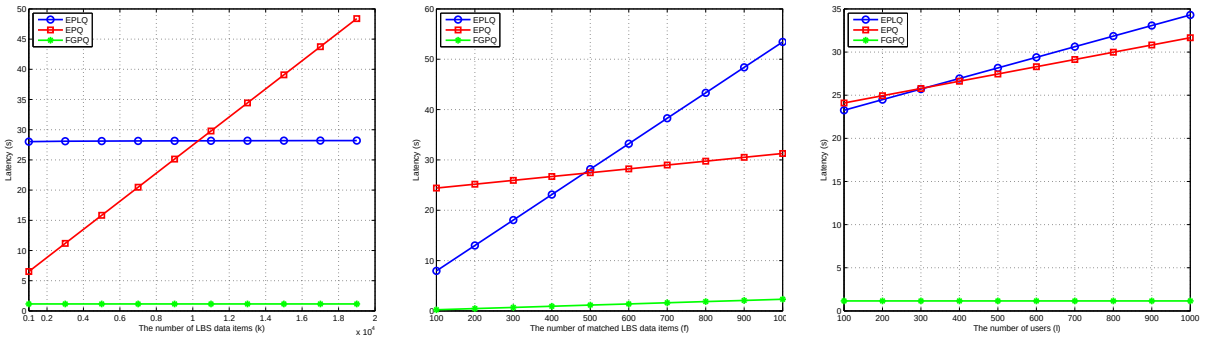
	LBS Provider-to-Cloud (Fog Nodes)	User-to-Cloud (A Fog Node)
EPLQ [18]	$512 \cdot (N+1)k + 128 \cdot k + 128 \cdot (\tau_2 - \tau_1)$	$1024 \cdot (N+1) + 128 \cdot f$
EPQ [19]	$4384 \cdot k + 128 \times 10^{10}$	$128 \cdot f + 6464$
FGPQ	$6144 \cdot k$	11264

6.3. Latency

In order to demonstrate the advantage of fog computing, this section discusses the latency of LBS searching, and gives a comparison with cloud computing-based schemes, e.g., the schemes proposed in [18,19]. In general, the service latency is a time interval from a user sending a request to receiving the feedback, which contains the online computing delay and communication delay. In our FGPQ scheme, since fog nodes are local, the communication delay between mobile users and fog nodes can be neglected, and thus the latency absolutely depends on the online computing time in fog nodes. As shown in Table 1, the online computing time of the fog node id_j is $2f \cdot C_p$ seconds, that is, the latency of our FGPQ scheme is $2f \cdot C_p$ seconds. Alternatively, if the cloud computing system is adopted, the latency depends on the online computing time in the cloud and the communication delay between users and the cloud. Specifically, in the EPLQ scheme [18], the online computing time in the cloud is $(\log k + f) \cdot N \cdot (C_p + C_{mt})$ seconds (See Table 1). In addition, based on Table 2, the communication overheads between each user and a fog node are $128 \cdot f + 1024(N+1)$ bits. Therefore, if l mobile users send requests to the cloud at the same time, the total network traffic is about $(128 \cdot f + 1024(N+1)) \cdot l$ bits. As analysed in [34], the communication delay for uploading 10 MB data takes about 28 s, thereby, the communication delay of the EPLQ scheme [18] is about $(128 \cdot f + 1024(N+1)) \cdot l \div 8388608$ s (1 MB =8,388,608 bits). Finally, the latency of the EPLQ scheme [18] is about $((\log k + f) \cdot N \cdot (C_p +$

$C_{mt}) + (128 \cdot f + 1024(N+1)) \cdot l \div 8388608)$ s. Similarity, based on Tables 1 and 2, the latency of the EPQ scheme [19] is about $(2k \cdot C_p + 5k \cdot C_{mt} + (128f + 6464)l \div 8388608)$ s.

Furthermore, we conduct the experiments with the pairing-based cryptograph library (PBC) [35] and the GNU multiple precision arithmetic library (GMP) [36] running on a 2.6 GHz-processor computer to study the latency. To this end, we set the security parameter $|\kappa| = 512$ bits. Furthermore, similar to [18], we set $N = 37$. With the exact operation costs, we depict the variation of computational costs in terms of the number of LBS items k, the number of matched LBS items and the number of mobile uses l who request LBS searching at the same time, in Figure 2. From the figure, it is obvious that our FGPQ scheme largely reduces the latency compared to the EPLQ scheme [18] and EPQ scheme [19]. In particular, in Figure 2a, our FGPQ scheme remains unchanged when k increases. This is because our FGPQ scheme takes advantage of a very efficient ASPE algorithm, and uses this algorithm to originally select f encrypted LBS items matched the spatial range. In addition, in practice, the number of LBS data stored in each fog node is far less than the total LBS data (i.e., $k_j \ll k$). Therefore, the latency of our FGPQ scheme only depends on the time verifying whether f LBS data satisfy the searching content, i.e., $2f$ pairing operations. In Figure 2c, our FGPQ scheme still remains unchanged when l increases. The main reason is that the communication delay between mobile users and local fog nodes can be neglected, and the computing operations for l users can be performed in parallel. According to Figure 2b, the latency of these three schemes increase with f increasing, but our scheme increase significantly slower.



(**a**)Different k ($f = l = 500$) (**b**)Different f ($k = 10,000$, $l = 500$) (**c**)Different l ($k = 10,000$, $f = 500$)

Figure 2. The comparison of latency for location-based service searching.

From the above analyses, our FGPQ scheme is indeed efficient in terms of computational and communication overheads, and also low-latency in fog computing-enhanced system, which is suitable for the real-time LBS searching for mobile users.

7. Conclusions

Fog computing-enhanced location-based services have been widely developed in the Internet of Things. In this paper, we have proposed a fine-grained and privacy-preserving query scheme in fog computing-enhanced location-based services. In our proposed scheme, mobile users can obtain the fine-grained searching result satisfying not only the given spatial range but also the searching content. Detailed privacy analysis shows that our proposed scheme indeed achieves the privacy preservation for the LBS provider and mobile users. In addition, extensive performance analyses and experiments demonstrate that our proposed scheme can significantly reduce computational and communication overheads and ensure low-latency, which outperforms existing state-of-the art schemes. Hence, our proposed scheme is more suitable for real-time LBS searching. In future work, we will consider a stronger adversarial model and design new solutions under the new model. In order to achieve more fine-grained LBS queries, we will consider the extensibility, such as searching the closest restaurant

that opens now and satisfies the five-star service score or has good sales volume, which makes the LBS query be more practical.

Acknowledgments: This work was supported in part by the National Science Foundation of China under Grant 61325005 and the Major Frontier Project of Sichuan Province under Grant 2015JY0282.

Author Contributions: The work presented in this paper was a collaboration of all authors. X.Y. conceived and designed the scheme, analyzed the data and wrote the paper; F.Y. performed the experiments; and X.T. commented on the work and modified the paper.

Conflicts of Interest: The authors declare no conflict of interest. The founding sponsors had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, and in the decision to publish the results.

References

1. Bonomi, F.; Milito, R.A.; Zhu, J.; Addepalli, S. Fog computing and its role in the internet of things. In Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing, Helsinki, Finland, 17 August 2012; pp. 13–16.
2. Dinh, T.; Kim, Y.; Lee, H. A Location-Based Interactive Model of Internet of Things and Cloud (IoT-Cloud) for Mobile Cloud Computing Applications. *Sensors* **2017**, *17*, 489.
3. Ryu, K.; Koizumi, Y.; Hasegawa, T. Name-based geographical routing/forwarding support for location-basedr IoT services. In Proceedings of the 24th IEEE International Conference on Network Protocols, Singapore, 8–11 November 2016; pp. 1–6.
4. Lu, R.; Lin, X.; Shen, X.S. SPOC: A Secure and Privacy-Preserving Opportunistic Computing Framework for Mobile-Healthcare Emergency. *IEEE Trans. Parallel Distrib. Syst.* **2013**, *24*, 614–624.
5. Freeman, H.; Zhang, T. The emerging era of fog computing and networking [The President's Page]. *IEEE Commun. Mag.* **2016**, *54*, 4–5.
6. Yannuzzi, M.; Milito, R.A.; Serral-Gracià, R.; Montero, D.; Nemirovsky, M. Key ingredients in an IoT recipe: Fog Computing, Cloud computing, and more Fog Computing. In Proceedings of the 19th IEEE International Workshop on Computer Aided Modeling and Design of Communication Links and Networks, Athens, Greece, 1–3 December 2014; pp. 325–329.
7. Stojmenovic, I.; Wen, S. The Fog Computing Paradigm: Scenarios and Security Issues. In Proceedings of the 2014 Federated Conference on Computer Science and Information Systems, Warsaw, Poland, 7–10 September 2014; pp. 1–8.
8. Yi, S.; Li, C.; Li, Q. A Survey of Fog Computing: Concepts, Applications and Issues. In Proceedings of the 2015 Workshop on Mobile Big Data, Mobidata@MobiHoc 2015, Hangzhou, China, 21 June 2015; pp. 37–42.
9. Deng, R.; Lu, R.; Lai, C.; Luan, T.H.; Liang, H. Optimal Workload Allocation in Fog-Cloud Computing Toward Balanced Delay and Power Consumption. *IEEE Internet Things J.* **2016**, *3*, 1171–1181.
10. Ardagna, C.A.; Cremonini, M.; Damiani, E.; di Vimercati, S.D.C.; Samarati, P. Location Privacy Protection Through Obfuscation-Based Techniques. In Proceedings of the 21st Annual IFIP WG 11.3 Working Conference on Data and Applications Security, Redondo Beach, CA, USA, 8–11 July 2007; pp. 47–60.
11. Kido, H.; Yanagisawa, Y.; Satoh, T. An anonymous communication technique using dummies for location-based services. In Proceedings of the International Conference on Pervasive Services 2005, Santorini, Greece, 11–14 July 2005; pp. 88–97.
12. Gruteser, M.; Grunwald, D. Anonymous Usage of Location-Based Services Through Spatial and Temporal Cloaking. In Proceedings of the First International Conference on Mobile Systems, Applications, and Services, San Francisco, CA, USA, 5–8 May 2003.
13. Mokbel, M.F.; Chow, C.; Aref, W.G. The New Casper: Query Processing for Location Services without Compromising Privacy. In Proceedings of the 32nd International Conference on Very Large Data Bases, Seoul, Korea, 12–15 September 2006; pp. 763–774.
14. Chor, B.; Kushilevitz, E.; Goldreich, O.; Sudan, M. Private Information Retrieval. *J. ACM* **1998**, *45*, 965–981.
15. Ghinita, G.; Kalnis, P.; Khoshgozaran, A.; Shahabi, C.; Tan, K. Private queries in location based services: Anonymizers are not necessary. In Proceedings of the ACM SIGMOD International Conference on Management of Data, Vancouver, BC, Canada, 10–12 June 2008; pp. 121–132.

16. Yi, X.; Paulet, R.; Bertino, E.; Varadharajan, V. Practical k nearest neighbor queries with location privacy. In Proceedings of the IEEE 30th International Conference on Data Engineering, Chicago, IL, USA, 31 March–4 April 2014; pp. 640–651.
17. Olumofin, F.G.; Goldberg, I. Revisiting the Computational Practicality of Private Information Retrieval. In Proceedings of the 15th International Conference on Financial Cryptography and Data Security, Gros Islet, St. Lucia, 28 February–4 March 2011; pp. 158–172.
18. Li, L.; Lu, R.; Huang, C. EPLQ: Efficient Privacy-Preserving Location-Based Query Over Outsourced Encrypted Data. *IEEE Internet Things J.* **2016**, *3*, 206–218.
19. Zhu, H.; Lu, R.; Huang, C.; Chen, L.; Li, H. An Efficient Privacy-Preserving Location-Based Services Query Scheme in Outsourced Cloud. *IEEE Trans. Veh. Technol.* **2016**, *65*, 7729–7739.
20. Peng, T.; Liu, Q.; Meng, D.; Wang, G. Collaborative trajectory privacy preserving scheme in location-based services. *Inf. Sci.* **2017**, *387*, 165–179.
21. Lai, C.; Lu, R.; Zheng, D.; Li, H.; Shen, X.S. Toward secure large-scale machine-to-machine comm unications in 3GPP networks: Challenges and solutions. *IEEE Commun. Mag.* **2015**, *53*, 12–19.
22. Dastjerdi, A.V.; Buyya, R. Fog Computing: Helping the Internet of Things Realize Its Potential. *IEEE Comput.* **2016**, *49*, 112–116.
23. Huang, C.; Ma, M.; Liu, Y.; Liu, A. Preserving Source Location Privacy for Energy Harvesting WSNs. *Sensors* **2017**, *17*, 724.
24. Wang, L.; Liu, G.; Sun, L. A Secure and Privacy-Preserving Navigation Scheme Using Spatial Crowdsourcing in Fog-Based VANETs. *Sensors* **2017**, *17*, 668.
25. Boneh, D.; Franklin, M.K. Identity-Based Encryption from the Weil Pairing. In Proceedings of the 21st Annual International Cryptology Conference on Advances in Cryptology, Santa Barbara, CA, USA, 19–23 August 2001; pp. 213–229.
26. Wong, W.K.; Cheung, D.W.; Kao, B.; Mamoulis, N. Secure kNN computation on encrypted databases. In Proceedings of the ACM SIGMOD International Conference on Management of Data, Providence, RI, USA, 29 June–2 July 2009; pp. 139–152.
27. Lu, R.; Heung, K.; Lashkari, A.H.; Ghorbani, A.A. A Lightweight Privacy-Preserving Data Aggregation Scheme for Fog Computing-Enhanced IoT. *IEEE Access* **2017**, *5*, 3302–3312.
28. Yang, X.; Lu, R.; Liang, H.; Tang, X. SFPM: A Secure and Fine-Grained Privacy-Preserving Matching Protocol for Mobile Social Networking. *Big Data Res.* **2016**, *3*, 2–9.
29. Boneh, D.; Lynn, B.; Shacham, H. Short Signatures from the Weil Pairing. *J. Cryptol.* **2004**, *17*, 297–319.
30. Li, H.; Yang, Y.; Luan, T.H.; Liang, X.; Zhou, L.; Shen, X.S. Enabling Fine-Grained Multi-Keyword Search Supporting Classified Sub-Dictionaries over Encrypted Cloud Data. *IEEE Trans. Dependable Secur. Comput.* **2016**, *13*, 312–325.
31. Cao, N.; Wang, C.; Li, M.; Ren, K.; Lou, W. Privacy-Preserving Multi-Keyword Ranked Search over Encrypted Cloud Data. *IEEE Trans. Parallel Distrib. Syst.* **2014**, *25*, 222–233.
32. Liu, L.; Tamer Özsu, M. Lp Distances. In *Encyclopedia of Database Systems*; Springer: New York, NY, USA, 2009; p. 1662.
33. Jiang, T.; Chen, X.; Wu, Q.; Ma, J.; Susilo, W.; Lou, W. Secure and Efficient Cloud Data Deduplication with Randomized Tag. *IEEE Trans. Inf. Forensic. Sec.* **2017**, *12*, 532–543.
34. Aazam, M.; Huh, E. Fog Computing and Smart Gateway Based Communication for Cloud of Things. In Proceedings of the 2014 International Conference on Future Internet of Things and Cloud, Barcelona, Spain, 27–29 August 2014; pp. 464–470.
35. Lynn, B. PBC Library. Available online: https://crypto.stanford.edu/pbc/thesis.html (accessed on 19 April 2017).
36. Granlund, T. The Gnu MP Bignum Library. Available online: https://gmplib.org/ (accessed on 19 April 2017).