A Collaborative K-anonymity Approach for Location Privacy in Location-Based Services

Hassan Takabi
School of Information Sciences
University of Pittsburgh
Pittsburgh, PA, USA
hatakabi@sis.pitt.edu

James B. D. Joshi
School of Information Sciences
University of Pittsburgh
Pittsburgh, PA, USA
jjoshi@sis.pitt.edu

Hassan A. Karimi
School of Information Sciences
University of Pittsburgh
Pittsburgh, PA, USA
hkarimi@sis.pitt.edu

***Abstract*—Considering the growth of wireless communication and mobile positioning technologies, location-based services (LBSs) have been generating increasing research interest in recent years. One of the critical issues for the deployment of LBS applications is how to reconcile their quality of service with privacy concerns. Location privacy based on *k*-anonymity is a very common way to hide the real locations of the users from the LBS provider. Several *k*-anonymity approaches have been proposed in the literature, each with some drawbacks. They need either a trusted third party or the users (or providers) to trust each other in collaborative approaches. In this paper, we propose a collaborative approach that provides *k*-anonymity in a distributed manner and does not require a trusted third party nor the users (or providers) to trust each other. Furthermore, our approach integrates well with the existing communication infrastructure. A user's location is known to only his/her location provider (e.g., cell phone operator). By using cryptographic schemes, user with the help of location providers determines whether the *k*-anonymity property is satisfied in a query area or not. We start with a simple scenario where user and location providers are honest-but-curious and then we progressively extend our protocol to deal with scenarios where entities may collude with each other. Moreover, we analyze possible threats and discuss how our proposed approach defends against such threats.**

I. Introduction

Considering the growth of wireless communication and mobile positioning technologies, location-based services (LBSs) have been generating increasing research interest in recent years. LBSs are convergence technologies resulting from the recent developments in several fields including mobile communication and computing, spatial database systems, Internet technology, geographical information systems (GIS), and others. Their applications in transportation, health care, vehicle to vehicle communications, social networks and other fields already exist and are growing rapidly. One of the critical issues for the deployment of LBS applications is how to reconcile their quality of service with privacy concerns. Users may be concerned about the possibility that an attacker will relate their identities with the information contained in their requests, including location among other information. In general, any connection between a request and the real identity of the person who issued it can be considered a privacy threat.

There are generally three different architectures for achieving the privacy in LBSs [3]. In the *non-cooperative* approach, the users use their own ability to hide their location using hiding techniques such as pseudonymity and dummies [18], [19], [20], [21], [22]. The *centralized trusted third party* (TTP) approach relies on a trusted third party that anonymizes the location of the user requests for the services with the anonymized location and returns the result to the users [9], [10], [11], [12]. In the *peer-to-peer cooperative* approach, a group of users cooperatively hide their location information. In this case, the union of the users leads to their anonymization [13], [14], [15], [16], [4]. The first approach is simple in design but vulnerable to several attacks [3]. The second approach suffers from TTP being a bottleneck, although it is the most accurate approach and provides the highest privacy level [3]. In the last approach, users collaborate to hide their location information in a distributed manner in order to achieve privacy.

K-anonymity is a well-known approach to preserve privacy [9], [10], [11], [12], [13], [14], [15], [16], [4]. By using this technique, a user's location is cloaked to ensure that there are at least $k-1$ other users within the cloaked area. So, the service provider is not able to distinguish the user from a set of k users because they share the same masked location. The idea of k-anonymity has been extensively applied to location privacy [9], [10], [11], [12], [13], [14], [15], [16], [4], [2]. In some of these approaches a trusted third party computes a cloaked area that has the k-anonymity property [9], [10], [11], [12]. More recently, some researchers have proposed to eliminate the trusted third party and to have users jointly compute a cloaked area satisfying the k-anonymity property [13], [15], [16], [4], [2]. Most of these approaches are based on cooperation among users, each with its own drawbacks. Some of them need the users to trust other users; others suffer from collusion among the users or collusion between the users and the service providers. Moreover, they are not secure against malicious users [15]. Another drawback of existing approaches is that they do not integrate well with existing LBS communication infrastructures. Many existing LBSs are aimed at cell phone users, and since cell phone operator knows the location of its customers, we can take advantage of that to provide k-anonymity in LBSs.

In this paper, we propose a distributed cooperative k-anonymity approach that does not need a trusted third party nor the users (or providers) to trust each other, and that

Digital Object Identifier: 10.4108/ICST.COLLABORATECOM2009.8374
http://dx.doi.org/10.4108/ICST.COLLABORATECOM2009.8374

integrates well with the existing infrastructures. We consider existing infrastructures with multiple location providers (e.g., cell phone operators) each having the location information of a subset of the users (e.g., its customers), where the subsets are disjoint. By using cryptographic schemes, a user with the help of location providers determines whether the k-anonymity property is satisfied in a given query area or not. First, we propose a protocol for simple scenario where the users and the location providers are honest-but-curious and then we extend it to deal with scenarios where entities may collude with each other. Finally, we analyze different threat scenarios and show how our proposed approach deals with them. Briefly, Our contribution is to propose an approach including a set of protocols that:

- Does not rely on the use of a trusted third party.
- Users (or location providers) do not need to trust each other.
- Is distributed.
- Integrates well with the existing infrastructures.
- Is robust against the collusion among users and location providers.
- Is modular, so users can use the protocols that they need depending on their current requirements.

The remainder of this paper is organized as follows: Section 2 describes the system model and the threat model. In section 3, we discuss our proposed approach. Section 4 reviews how the proposed approach defends against possible threats, and discusses how our approach can be extended to defend against malicious entities. Section 5 discusses the related work. Finally, Section 6 concludes the paper.

II. System and Threat Model

In this section, we describe our system model as illustrated in Figure 1. Then, we describe the threat model.

A. System Model

A *coverage area* is divided into a grid of equal size cells. The size of cells should be chosen such that there is a realistic chance that for most of the cells, multiple users are located in the cell. In order to provide scalability, there exist multiple coverage areas, each associated with the area covered by a particular instantiation of our system.

Generally there are four entities in the system: location providers, users, and directory servers. Below, we discuss each of these entities in detail.

1) Location Provider: The location providers are responsible for keeping track of current location of users in the coverage area. There are multiple location providers in a coverage area; each keeps track of the location information of a subset of users. These subsets of users are mutually disjoint. A location provider can provide full or partial coverage in the coverage area. For example, a cell phone network operator would likely cover most cells, whereas a WiFi network would provide coverage only for a subset of the cells [2]. Different providers are maintained by different organizations. Users

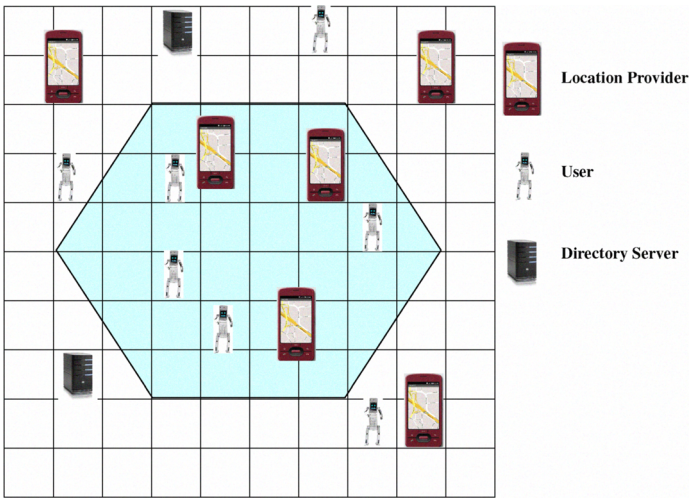

Fig. 1. The System Model

carry a mobile device that is able to locate itself by using, for instance, a GPS or nearby WiFi base stations.

2) User: A user is associated with exactly one of the location providers that keeps track of his/her location. Since the provider of the communication service (e.g., cell phone operator) accessed by the user's device already knows the his/her location, it is likely that the location provider is associated with that user.

3) Directory Server: The directory server is responsible for publishing contact information for the location providers in the coverage area. It also publishes information about that location providers including information about which location provider covers which cells in the coverage area.

4) LBS Provider: The location based service provider receives a query from a user. The query includes the type of service the user needs and the area where that service is. The LBS provider processes the query and sends a reply to the user based on the location included in the query.

B. Threat Model

In this section, we describe our threat model. We assume that the location providers honestly follow the protocol, but are curious about learning location information.

1) Location Provider: A location provider is able to learn:

- the location of users who are associated with this particular provider, and
- the number of users in a cell who are associated with this particular provider.

However, a location provider should not be able to learn:

- the total number of users in a cell associated with each of the location providers.
- the locations of users who are associated with any other location provider.

2) User: A user is able to learn only:

- his own location, and
- whether the number of people in the query area is at least k, where k is a value of his choice.

Digital Object Identifier: 10.4108/ICST.COLLABORATECOM2009.8374
http://dx.doi.org/10.4108/ICST.COLLABORATECOM2009.8374

We assume that a user carries only one mobile device that faithfully reports its location to a location provider.

3) Directory Server: The directory server should not be able to learn any location information about the users. However, the server might misbehave. It might provide inaccurate information about location providers; for example, it may list a location provider multiple times as providing coverage for a single cell or fail to list some location providers.

4) LBS Provider: The LBS provider is able to learn only a cloaked location that includes at least k users. It should not be able to learn the exact locations of the users.

What we discussed so far, are general threats from involved entities. We will discuss threat scenarios in detail in Section IV and show how to defend against such threats.

III. Proposed Collaborative Approach

In this section, we describe our approach that shows how a user can learn whether there are at least k users including himself/herself in a given query area, where k is a value chosen by the user. The query area initially corresponds to user's current cell and if the user finds out that there are fewer than k users in this cell, he/she can enlarge the query area and re-execute the protocol for the enlarged area. This process can be repeated multiple times. Generally speaking, we assume that location providers do not collude with each other. Since cell phone network operators know their users' locations and could easily share this information with each other, technical enforcement means are inadequate here and this can be enforced using legal means such as privacy laws or a contract between a user and a location provider. Similarly, it is assumed that location providers do not collude with users. However, we take into account scenarios where entities (users, location providers) may collude with each other. The simplest scenario is where the user and location providers are *honest-but-curious* and it is extended to consider scenarios where entities may collude with each other.

A. Misbehaving Directory Service

The user identifies all location providers that cover the query area. In order to do this, the user should download the entire directory or its recent changes from the server on a regular basis, such as once a day. If the user connects to the directory server and asks for a list of location providers, the server could learn user's location. As we have already mentioned, the directory server might misbehave. In order to prevent the directory service from misbehaving, two solutions could be used: to have the directory service to sign the directory which allows retroactive detection of misbehavior or to have multiple directory servers, where users accept information only if it is signed by a threshold of the servers. The user then executes our cooperative k-anonymity protocol with the related location providers.

To run our protocol, there should be at least two location providers in the coverage area; one is the location provider that the user is associated with and another is chosen randomly among other location providers in the coverage area. These location providers are called LP_1 and LP_2, respectively.

B. Notations

Our protocol uses the public key cryptography primitives to preserve privacy. The notations we use in this paper are as follows:

- m and c represent message (plaintext) and cyphertext respectively.
- PK represents public key in a cryptosystem that supports homomorphic properties and PK_l represents public key of l.
- E_{PK} is encryption function using public key PK.
- $E_{PK}(m)$ is encryption of message m under public key PK.
- $E_{PK_2}(E_{PK_1}(m)$ is encryption of message m under public keys PK_1 and PK_2. First, the message m is encrypted using public key PK_1 and then the result is encrypted using public key PK_2.
- RPK represents public key in a generic cryptosystem (does not necessarily support homomorphic properties) and RPK_l represents public key of l.
- E_{RPK} is encryption function using public key RPK.

The cryptosystem we use in our protocol should support homomorphic addition and subtraction of ciphertexts. Given two ciphertexts $c_1 = E_{PK}(m_1)$ and $c_2 = E_{PK}(m_2)$, one can efficiently compute $E_{PK}(m_1 + m_1)$.

$c_1 * c_2 = E_{PK}(m_1) * E_{PK}(m_1) = E_{PK}(m_1 + m_1)$

Similarly, given two ciphertexts $c_1 = E_{PK}(m_1)$ and $c_2 = E_{PK}(m_2)$, one can efficiently compute $E_{PK}(m_1 - m_1)$.

$c_1 * c_2^{-1} = E_{PK}(m_1) * E_{PK}(m_1)^{-1} = E_{PK}(m_1 - m_1)$

There are several cryptosystems that support these properties, such as the Benaloh Cryptosystem [25].

It also should support multiplication and division of ciphertexts. Given two ciphertexts $E_{PK}(m_1)$ and $E_{PK}(m_2)$, one can efficiently compute $E_{PK}(m_1 \times m_1)$. There are several cryptosystems that support this property, such as the Boneh-Goh-Nissim Cryptosystem [26].

The public keys of location providers in the protocol are denoted as PK_{LP_1} and PK_{LP_2}, respectively.

C. Protocol 1

As we have already mentioned, in the simplest case, we assume that user and location providers are *honest-but-curious* and propose our protocol based on this assumption. In the protocol, after identifying location providers that cover the query area, the user asks each location provider for the number of users associated with it in the query area. Towards this, the user uses this location provider, LP_1, to query other location providers and LP_1 serves as a trusted proxy for the user. Each location provider encrypts the number of users associated with it using PK_{LP_1} and PK_{LP_2}, the public keys of LP_1 and LP_2; and sends it to the user. This way, the user cannot learn the computed number. If there are n_i users in the query area associated with location provider i, provider i sends $E_{PK_{LP_1}}(E_{PK_{LP_2}}(n_i))$ to the user. Then, the user sums up the received values from all location providers but he is not able to learn the sum. Particularly, the

Digital Object Identifier: 10.4108/ICST.COLLABORATECOM2009.8374
http://dx.doi.org/10.4108/ICST.COLLABORATECOM2009.8374

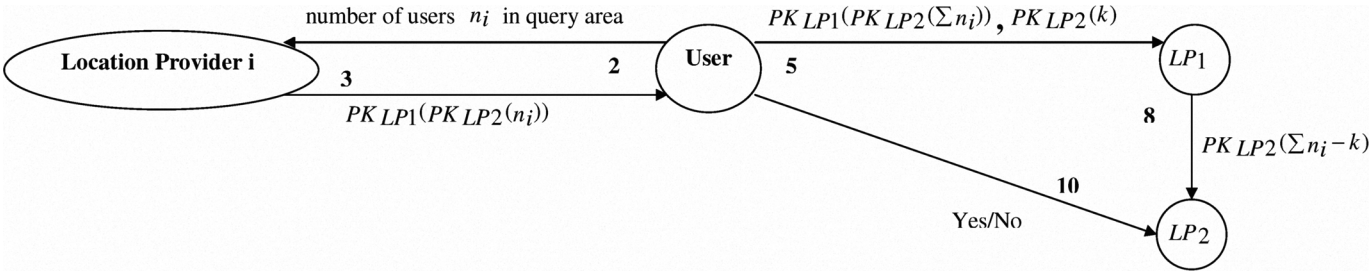

Fig. 2. The protocol for simplest scenario

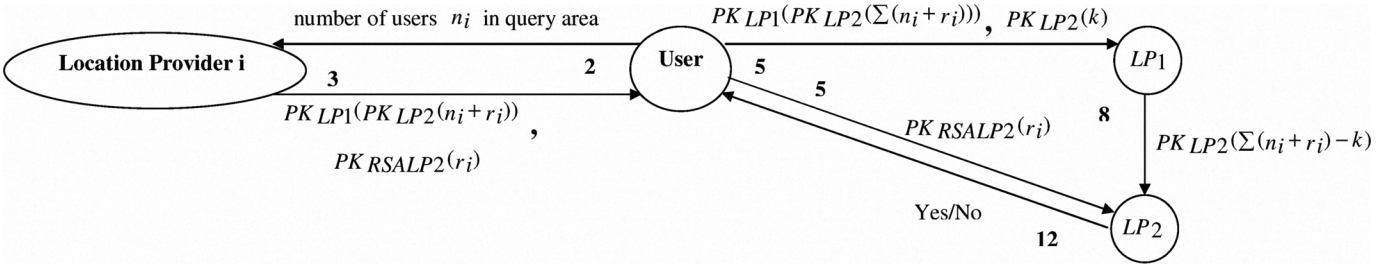

Fig. 3. The protocol to prevent user's binary search attack

user calculates $E_{PK_{LP_1}}(E_{PK_{LP_2}}(\sum n_i))$ using the additive homomorphic property of the encryption scheme. Then, the user sends the sum along with $E_{PK_{LP_2}}(k)$ to LP_1. Location Provider LP_1 decrypts the sum using its private key and extracts $E_{PK_{LP_2}}(\sum n_i)$; then using the homomorphic subtraction property of the encryption scheme, it computes $E_{PK_{LP_2}}(\sum n_i - k)$ which is sent to LP_2. Finally, location provider LP_2 decrypts the received value to extract $\sum n_i - k$ and sends "no" if the result is negative otherwise sends "yes".

In this way, none of the location providers can learn k. LP_1 can not learn k because it is encrypted using the public key of LP_2. Neither can LP_2 learn anything about k because what it has is a number that is the difference between exact number of users in the query area and k. Figure 2 illustrates the protocol.

The steps of the protocol are as follows:

- **Step 1.** The user identifies all location providers that cover the query area.
- **Step 2.** User asks all location providers for the number of users in the query area associated with them.
- **Step 3.** Each LP_i sends $E_{PK_{LP_1}}(E_{PK_{LP_2}}(n_i))$ to the user.
- **Step 4.** User sums up the received encrypted values from all location providers. The result is $E_{PK_{LP_1}}(E_{PK_{LP_2}}(\sum n_i))$.
- **Step 5.** User sends $E_{PK_{LP_1}}(E_{PK_{LP_2}}(\sum n_i))$ and $E_{PK_{LP_2}}(k)$ to LP_1.
- **Step 6.** LP_1 decrypts the first message; the result is $E_{PK_{LP_2}}(\sum n_i)$.
- **Step 7.** LP_1 subtracts the second one from it; the result is $E_{PK_{LP_2}}(\sum n_i - k)$.
- **Step 8.** LP_1 sends the result $E_{PK_{LP_2}}(\sum n_i - k)$ to LP_2.
- **Step 9.** LP_2 decrypts it to extract the value; the result is $\sum n_i - k$.
- **Step 10.** LP_2 sends yes or no to the user depending on the value of $\sum n_i - k$.

Although this protocol works well based on the assumptions made there are situations that it does not work without those assumptions; One situation is that user might be able to learn the exact number of users in a query area using binary search. Another sitution is that LP_1 and LP_2 might collude with each other. We discuss these flaws in detail and present our solution to address them in the following.

D. Protocol 2

One situation that the protocol proposed does not work is that a user might be able to learn the exact number of users in a query area using binary search. The user can perform a binary search for the actual value of $\sum n_i$ by adjusting the value of k in each run of the protocol. The user can send $E_{PK_{LP_1}}(E_{PK_{LP_2}}(\sum n_i))$ multiple times to the location provider LP_1 with different values of k and finally learn the the exact number of users in a query area.

To prevent this attack, we use time stamped tickets. Instead of sending $E_{PK_{LP_1}}(E_{PK_{LP_2}}(n_i))$ to the user, a location provider sends $E_{PK_{LP_1}}(E_{PK_{LP_2}}(n_i + r_i))$ and a ticket that contains $E_{RPK_{LP_2}}(r_i)$, where r_i is a random number changing with each request and $E_{RPK_{LP_2}}$ is the encryption function using public key $E_{RPK_{LP_2}}$ of the location provide LP_2. Note that this public key could be any kind of public key and does not need to have any homomorphic property. We use RPK notation to indicate regular public key. A location provider also includes an expiration time in the ticket and signs the ticket. The location provider LP_2 will decrypt all $E_{RPK_{LP_2}}(\sum r_i)$ and subtract $\sum r_i$ from $\sum(n_i + r_i)$. It also remembers tickets until their expiration time and refuses to reuse a previously seen ticket. Moreover, since the r_i value is different in each

Digital Object Identifier: 10.4108/ICST.COLLABORATECOM2009.8374
http://dx.doi.org/10.4108/ICST.COLLABORATECOM2009.8374

request, the user cannot use fresh tickets with a previously presented encrypted sum, so the location provider LP_1 cannot compute the correct input and the user cannot learn any useful information from this operation.

The steps of the protocol as illustrated in Figure 3 are as follows:

- **Step 1.** The user identifies all location providers that cover the query area.
- **Step 2.** User asks all location providers for the number of users in the query area associated with them.
- **Step 3.** Each LP_i sends $E_{PK_{LP_1}}(E_{PK_{LP_2}}(n_i + r_i))$ and a ticket to the user. The ticket includes $E_{RPK_{LP_2}}(r_i)$, where r_i is a random number changing with each request and $E_{RPK_{LP_2}}$ is the encryption function using public key RPK_{LP_2} of the location provide LP_2. It also includes an expiration time and is signed.
- **Step 4.** User sums up the received encrypted values from all the location providers. The result is $E_{PK_{LP_1}}(E_{PK_{LP_2}}(\sum(n_i + r_i)))$. It also sums up the random numbers in the tickets; the result is $E_{RPK_{LP_2}}(\sum r_i)$. It also sends expiration dates of tickets to LP_2.
- **Step 5.** User sends $E_{PK_{LP_1}}(E_{PK_{LP_2}}(\sum(n_i + r_i)))$ and $E_{PK_{LP_2}}(k)$ to LP_1 and sends $E_{RPK_{LP_2}}(\sum r_i)$ to LP_2.
- **Step 6.** LP_1 decrypts the first message; the result is $E_{PK_{LP_2}}(\sum(n_i + r_i))$.
- **Step 7.** LP_1 subtracts the second one from it; the result is $E_{PK_{LP_2}}(\sum(n_i + r_i) - k)$.
- **Step 8.** LP_1 sends the result $E_{PK_{LP_2}}(\sum(n_i + r_i) - k)$ to LP_2.
- **Step 9.** LP_2 decrypts the received value to extract the value; the result is $\sum(n_i + r_i) - k$.
- **Step 10.** LP_2 also decrypts $E_{RPK_{LP_2}}(\sum r_i)$; the result is $\sum r_i$.
- **Step 11.** LP_2 subtracts the result of step 10 from the result of step 9; the result is $\sum n_i - k$. It also checks the expiration dates of tickets and refuses to reuse a previously seen ticket.
- **Step 12.** LP_2 sends yes or no to the user depending on the value of $\sum n_i - k$.

E. Protocol 3

Suppose LP_1 and LP_2 collude with each other, i.e. they share their private keys then can easily learn $\sum n_i$, the exact number of users in the query area. In order to prevent LP_2 from knowing the total number of users in the query area, we use Gaussian noise with null average $\sim N(0, \sigma)$ [1]. By using a Gaussian pseudo-random number generator, each location provider LP_i can obtain a number N_i following the desired distribution [1]. Then this value is added to the real number of users and the location provider sends $n_i + N_i$ to the user instead of n_i. In this way, the real number of users is masked and even if LP_1 and LP_2 collude with each other they can not learn the real number of users in the query area.

The following equation shows that when we use Gaussian noise with null average, how N_i is removed from final result and does not affect the real number of users in the query area.

$$\frac{\sum(n_i+N_i)}{l} = \frac{\sum n_i}{l} + \frac{\sum N_i}{l} = \frac{\sum n_i}{l} + \overline{N} = \frac{\sum n_i}{l} + 0 = \frac{\sum n_i}{l}$$

where N_i is the number obtained from Gaussian pseudo-random number generator and l is a random number chosen by the user.

The steps of the protocol as illustrated in Figure 3 are as follows:

- **Step 1.** The user identifies all location providers that cover the query area.
- **Step 2.** User asks all location providers for the number of users in the query area associated with them and keeps the number of all location providers l.
- **Step 3.** Each LP_i sends $E_{PK_{LP_1}}(E_{PK_{LP_2}}(n_i + N_i))$ to the user.
- **Step 4.** User sums up the received encrypted values from all location providers and divide it by $E_{PK_{LP_1}}(E_{PK_{LP_2}}(l)$. The result is $E_{PK_{LP_1}}(E_{PK_{LP_2}}(\frac{\sum(n_i+N_i)}{l}))$.
- **Step 5.** User sends $E_{PK_{LP_1}}(E_{PK_{LP_2}}(\frac{\sum n_i}{l})$ and $E_{PK_{LP_2}}(\frac{k}{l})$ to LP_1.
- **Step 6.** LP_1 decrypts the first message; the result is $E_{PK_{LP_2}}(\frac{\sum n_i}{l})$.
- **Step 7.** LP_1 subtracts the second message from step 5 from result of step 6; the result is $E_{PK_{LP_2}}(\frac{\sum n_i - k}{l})$.
- **Step 8.** LP_1 sends the result $E_{PK_{LP_2}}(\frac{\sum n_i - k}{l})$ to LP_2.
- **Step 9.** LP_2 decrypts it to extract the value; the result is $\frac{\sum n_i - k}{l}$.
- **Step 10.** LP_2 sends yes or no to the user depending on the value of $\frac{\sum n_i - k}{l}$.

F. Protocol 4

It is possible that the user, LP_1 and LP_2 collude together. In this case the user has private keys of both LP_1 and LP_2 and can find the number of users associated with each location provider in the query area. In order to prevent this attack, after identifying all location providers that cover the query area, we can have the user form a set of location providers other than LP_2 and LP_1. The user chooses a location provider randomly from the set $\{LP_3, LP_4, ..., LP_n\}$ where n is number of location providers and sends the set along with the query to the chosen provider.

The steps of the protocol are as follows:

- **Step 1.** The user identifies all location providers that cover the query area and forms a set of providers excluding LP_1 and LP_2. The set is $\{LP_3, LP_4, ..., LP_n\}$.
- **Step 2.** User chooses a location provider from the set and sends the set along with the query to the chosen location provider.
- **Step 3.** The location provider in step 2 removes its id from the set, chooses another provider from the set, and sends $E_{PK_{LP_1}}(E_{PK_{LP_2}}(n_i))$ to it where n_i is the number of users associated with this provider.

Digital Object Identifier: 10.4108/ICST.COLLABORATECOM2009.8374
http://dx.doi.org/10.4108/ICST.COLLABORATECOM2009.8374

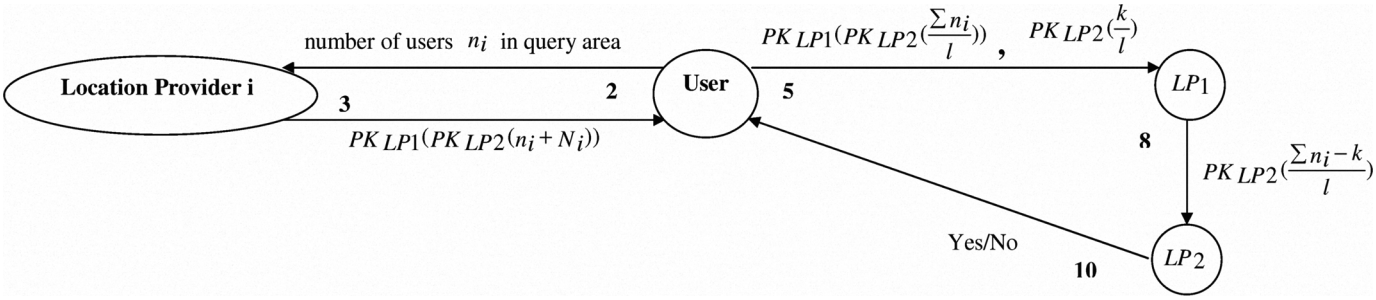

Fig. 4. The protocol to prevent collusion between LP_1 and LP_2

- **Step 4.** While the set is not empty, each location provider LP_i in the set removes its id from the set, ,adds $E_{PK_{LP_1}}(E_{PK_{LP_2}}(n_i))$ to the value received, chooses another provider from the set and sends the result to the chosen provider.
- **Step 5.** The last provider in the set, LP_l, after adding the number of its associated users sends the result, $E_{PK_{LP_1}}(E_{PK_{LP_2}}(\sum n_i))$, to LP_1.
- **Step 6.** The user sends $E_{PK_{LP_2}}(k)$ to LP_1.
- **Step 7.** LP_1 decrypts the value received, adds the number of users associated with it, $E_{PK_{LP_2}}(n_1)$, subtracts $E_{PK_{LP_2}}(k)$ from it and sends the result, $E_{PK_{LP_2}}(\sum n_i + n_1)$, to LP_2.
- **Step 8.** LP_2 decrypts the message and adds the number of users associated with it, n_2.
- **Step 9.** LP_2 sends yes or no to the user depending on the value of $\sum n_i - k$.

IV. Threat Analysis

In this section, we review different threat scenarios and discuss how our proposed approach defends against the threats. As illustrated in Table I, we take into account scenarios where entities are *honest*, *honest-but-curious* and *malicious*. Based on different combinations we suggest what protocols should be used. If all the users and location providers are honest, we do not even need to use encryption and naively they can communicate via plain text. If we assume that user is honest while location providers are honest-but-curious, we can use the proposed protocol 1. In this protocol, we assume location providers do not interact with each other, so they cannot learn the location of users in the query area who are associated with other providers, not even the total number of users. The location provider that a user associates with can serve as the trusted proxy for contacting the other providers during a query operation.

In case the user is honest-but-curious, we use protocol 2, no matter location providers are honest or honest-but-curious. The user learns only whether the total number of users in his/her query area is at least k. The time stamped tickets prevent him/her from learning the actual number of users by using a binary search. In case one of the involved entities (user or location providers) are malicious, we need to use a combination of the proposed protocols and what we discuss in sections IV-A and IV-B as shown in Table I. In case location providers LP_1 and LP_2 collude with each other we use protocol 3 and if they collude with the user we use protocol 4 to preserve privacy.

The protocols gracefully deals with crashes of location providers other than LP_1 and LP_2. In case of crash, the user contacts the remaining providers, which might still report a sufficient number of users in the query area. Over time, the directory server will learn of the crash of a location provider and will remove it from the directory.

Now, we discuss how our approach can be extended to defend against malicious parties.

A. Malicious Providers

We can have a location provider log the random values used in its encryption process and also sign all its generated messages to achieve non-repudiation. If users suspect misbehavior, they, likely in collaboration with the directory server, can force the location provider to reveal the logged values and its private key to validate the server's computations.

A malicious location provider can misbehave while executing the protocol. For example, it can send the user a value different from the actual number of users associated with the query area. We can have location providers keep record of all their actions but it is problematic in terms of privacy, because this record would have to include users location information. An alternative approach is that if users suspect misbehavior by a location provider, they can report the set of location providers from which they received information to the directory server. So, the directory server will be able to remove misbehaving location providers. As we mentioned earlier, the user should not directly connect to the directory server, so it can use the location provider it associates with as the trusted proxy for contacting the directory server.

B. Malicious Users

Malicious users could report wrong locations to location providers. A complete defense against this attack is likely impossible, but we outline some mechanisms that make this attack harder. A location provider might be able to detect wrongly reported locations. For example, if a provider is the operator of a WiFi network, the operator can ensure that a reported location is close to the WiFi access point from which the registration request was sent. An operator of a cell phone

Digital Object Identifier: 10.4108/ICST.COLLABORATECOM2009.8374
http://dx.doi.org/10.4108/ICST.COLLABORATECOM2009.8374

TABLE I
SCENARIO ANALYSIS

Location Providers	Users	Approach
honest	honest	no encryption needed; communicate via plain text
honest-but-curious	honest	protocol 1
honest	honest-but-curious	protocol 2
honest-but-curious	honest-but-curious	protocol 2
malicious	honest	no encryption needed; discussed in section IV-A
honest	malicious	no encryption needed; discussed in section IV-B
honest-but-curious	malicious	protocol 1; discussed in section IV-B
malicious	honest-but-curious	protocol 2; discussed in section IV-A
malicious	malicious	discussed in sections IV-A and IV-B

network can verify whether the reporting device is close to a particular cell phone tower. An alternative is to ask the user for a credit card number, including his name and billing address. This option becomes especially attractive if the system charges its users in the first place. Billing for the usage of the system itself can become a mechanism for reducing misbehavior, because an attacker might not have the necessary resources for a large-scale attack.

C. Malicious Directory Services

The directory server cannot learn any location information, because users do not retrieve individual records for their current cell from the server. However, as we mentioned earlier it might misbehave. In order to prevent the directory service from misbehaving, two solutions could be used: to have the directory service to sign the published directory which allows retroactive detection of misbehavior or to have multiple directory servers, where users accept information only if it is signed by a threshold of the servers.

V. RELATED WORK

There are three general approaches for preserving privacy in LBS: non-cooperative, centralized trusted third party (TTP), and cooperative approach.

Non-cooperative approach is the simplest approach but is vulnerable to several attacks. The user hides his/her location using his/her own capabilities and knowledge. To do this, the user uses hiding techniques such as pseudonymity, false dummies and landmark objects [3].

In centralized trusted third party (TTP) approach, users rely on a trusted third party that anonymizes the location, requests the service using the anonymized location information and returns the result to the user. TTP-based models are very common because they are easy to understand and in general they offer a reasonable trade-off between efficiency, accuracy and privacy. K-anonymity is a very common way to hide the real location of the users from the LBS provider. K-anonymity was suggested by Samarati and Sweeney for applications in databases by the Statistical Disclosure Control (SDC) community [6], [7]. K-anonymity has been adapted for LBS privacy: the location of a user is k-anonymous if it is indistinguishable from the location of k users. The basic idea behind k-anonymity is to replace the real location of a user by cloaking areas in which at least $k-1$ other users are located.

Several k-anonymity approaches have been proposed in the literature [8], [9], [11], [10]. *Gedik et al.* have proposed an anonymizer that allows a user to define his/her personal privacy requirements such as the number k of users amongst which he/she wants to be anonymized, the maximum delay, and location perturbation he/she is willing to accept [11]. This is an extension of their previous anonymizer [9]. The proposed anonymizer is resilient against identification attacks such as restricted space identification (RSI) attack and the observation identification (OI) [8]. However, it has some shortcomings: the user must trust the platform mediating him/her and the LBS provider; LBS providers are assumed to be not malicious but semi-honest, which is not always the case; and the architecture is centralized, which makes it a point of failure. *Bamba et al.* have proposed a similar method called PrivacyGrid. Although the proposed anonymizer by *Gedik et al.* and the PrivacyGrid approach are very similar, the latter seems to be more efficient because it uses the cloaking techniques based on grids. Furthermore, PrivacyGrid improves privacy of LBS users by considering the l-diversity property as well as the already considered k-anonymity property. Although PrivacyGrid seems to improve the anonymizer proposed by *Gedik et al.*, it suffers from the same drawbacks.

In the cooperative approach, in order to achieve privacy, users cooperate to hide their location information in a distributed manner. *Ferrer* has proposed a collaborative TTP-free algorithm for location privacy in LBS [13]. First, the user adds zero-mean Gaussian noise to his/her location to perturb it. Then the user broadcasts his/her perturbed location and requests neighbors to return their perturbed locations. The user selects $k-1$ neighbors such that the group formed by these neighbors and his/her own perturbed location spans an area satisfying a privacy parameter and an accuracy parameter. Finally, the user sends the centroid of the group of k perturbed locations to the LBS. The users do not need to trust each other because they only exchange perturbed locations. This method does not achieve k-anonymity because the centroid is only used by a single user to identify himself/herself. In addition, if users are static the noise will be canceled, so users cannot use this method many times without changing their location. *Chow et al.* have proposed another peer-to-peer scheme for location privacy similar to the one proposed by *Ferrer* [14]. The main idea is to generate cloaking area: users find other users in their cover range and share their

Digital Object Identifier: 10.4108/ICST.COLLABORATECOM2009.8374
http://dx.doi.org/10.4108/ICST.COLLABORATECOM2009.8374

location information. Then users can send their queries to LBS providers using the cloaking area instead of their real locations. The main drawback of this proposal is that users must trust other users because they exchange their real locations. *Solanas et al.* propose a method to compute a fake location that is shared by k users based on Gaussian noise addition [15]. All k users use the same fake location, so that their locations become k-anonymous. The authors have also extended the method to support non-centralized communications [16]. They propose a modular approach that progressively increases the privacy achieved by users. The basic module is equivalent to the method proposed by *Ferrer* [13] where users trust each other to share their location and compute a centroid that they use as their fake location. The second module allows users to exchange their location without trusting other peers. This module adds Gaussian noise with zero mean to the real location of the users. However, due to the noise cancelation effect, if users do not change their location and repeat this procedure several times when their location is static, their real location could be revealed. In order to prevent this, a third module is added that uses privacy homomorphisms to guarantee that users cannot see the real locations of other users whilst still being able to compute the centroid [17]. Finally, in order to avoid the denial of service attacks to the central user, a module that distributes users in a chain is added. At the end of the protocol users become k-anonymous and their location privacy is secured. *Zhong et al.* have proposed another k-anonymity approach that assumes that there are multiple servers, each deployed by a different organization [2]. They have location brokers that keep track of the current location of a user. Also there are secure comparison servers that interact with a user to let the user learn whether there are at least k users who have registered the user's current cell as their location across all location brokers. They have also implemented their protocol to show its efficiency.

Duckham et al. have proposed an approach based on obfuscation that is the process of degrading the quality of information about user's location to protect his/her privacy [18]. The authors have also presented an obfuscation method based on imprecision that models the space as a graph where vertices are locations and edges indicate adjacency [19]. The user sends a set of vertices instead of the single vertex in which he/she is located. So, the LBS provider cannot distinguish which of the vertices is the real one. Their negotiation algorithms allow users to increase the QoS whilst maintaining their privacy. The drawback of this approach is that users and providers must share the graph modeling the space. *Ardagna et al.* have recently proposed an obfuscation method where the real location of LBS users is replaced by circular areas of variable center and radius [21]. SpaceTwist is another obfuscation method that generates an anchor used to retrieve information on the k nearest points of interest from the LBS provider [22]. SpaceTwist can determine the closest point of interest to the real location whilst the LBS provider is not able to obtain the real location of the user. In this approach no TTP and no collaboration are needed. The closest point of interest is always found, and the real location of the user is hidden in a controlled area. However, due to the lack of collaboration, this method does not achieve the k-anonymity nor the l-diversity properties. Our proposed approach is a distributed cooperative k-anonymity approach that does not rely on the use of a trusted third party. Users (or location providers) do not need to trust each other. It also integrates well with the existing infrastructures. It is a modular approach, so users can use the protocols that they need depending on their current requirements and as we have shown it is robust against the collusion among user and location providers.

VI. Conclusion

We have proposed a distributed cooperative approach for location privacy in LBS based on k-anonymity. Our presented approach needs neither a trusted third party nor users to trust each other. By using cryptographic schemes, user can learn whether there are at least k users including himself/herself in the query area. In order to do this, user cooperates with location providers to determine whether k-anonymity property is satisfied. No party can learn information about user's location other than his/her location provider. Our approach integrates well with the existing infrastructure. We have proposed a set of protocols to deal with different situations that maybe encountered. Furthermore, we have analyzed possible threat scenarios and shown how our proposed approach defends against these threats.

References

[1] Solanas A. and Martinez-Balleste A., "A TTP-free protocol for location privacy in location-based services", Journal of Computer Communications, Vol. 31, No. 12, 2008.

[2] Ge Zhong and Urs Hengartner, "Toward a Distributed k-Anonymity Protocol for Location Privacy", In Proc. of the Workshop on Privacy in the Electronic Society (WPES'08), pages 33-37, USA, 2008.

[3] A. Mohaisen, D. Hong, and D. Nyang, "Privacy in Location Based Services: Primitives toward the Solution", In Proc. of the Fourth International Conference on Networked Computing and Advanced Information Management, pages 572-579, 2008.

[4] Marius O. Gheorghita, Agusti Solanas, and Jordi Forn, "Location Privacy in Chain-Based Protocols for Location-Based Services", In Proc. of the Third International Conference on Digital Telecommunications, pages 64-69, 2008.

[5] Urs Hengartner, "Location Privacy based on Trusted Computing and Secure Logging", In Proc. of the 4th International Conference on Security and Privacy in Communication Networks (SecureComm 2008), 2008.

[6] P. Samarati, "Protecting respondents identities in microdata release", IEEE Transactions on Knowledge and Data Engineering, Vol. 13, No. 6, pages 1010-1027, 2001.

[7] P. Samarati, L. Sweeney, "Protecting privacy when disclosing information: kanonymity and its enforcement through generalization and suppression", Technical report, SRI International, 1998.

[8] M. Gruteser, D. Grunwald, "Anonymous usage of location-based services through spatial and temporal cloaking", In Proc. of the First International Conference on Mobile Systems, Applications, and Services (MobiSys 2003), pages 31-42, USA, 2003.

[9] B. Gedik, L. Liu, "A customizable k-anonymity model for protecting location privacy", In Proc. of the IEEE International conference on Distributed Computing Systems (ICDS'05), pages 620-629, 2005.

[10] R. Cheng, Y. Zhang, E. Bertino, S. Prabhakar, "Preserving user location privacy in mobile data management infrastructures", In Proc. of the 6th Workshop on Privacy Enhancing Technologies (PET), pages 393-412, 2006.

Digital Object Identifier: 10.4108/ICST.COLLABORATECOM2009.8374
http://dx.doi.org/10.4108/ICST.COLLABORATECOM2009.8374

[11] B. Gedik, L. Liu, "Protecting location privacy with personalized k-anonymity: Architecture and algorithms", IEEE Transactions on Mobile Computing, Vol. 7, No. 1, pages 1-18, 2008.
[12] B. Bamba, L. Liu, P. Pesti, T. Wang, "Supporting anonymous location queries in mobile environments with privacygrid", In Proc. of the International World Wide Web Conference, pages 237-246, 2008.
[13] J. Domingo-Ferrer, "Microaggregation for database and location privacy", In O. Etzion, T. Kuflik, A. Motro, eds.: Next Generation Information Technologies and Systems-NGITS, pages 106-116, 2006.
[14] C. Chow, M. F. Mokbel, X. Liu, "A peer-to-peer spatial cloaking algorithm for anonymous location-based services", In Proc. of the 14th annual ACM international symposium on Advances in geographic information systems, pages 171-178, Virginia, USA, 2006.
[15] A. Solanas, A. Martinez-Balleste, "Privacy protection in location-based services through a public-key privacy homomorphism", In Proc. of the 4th European PKI Workshop: theory and practice, pages 362-368, 2007.
[16] A. Solanas, A. Martinez-Balleste, "Location privacy in location-based services: Beyond TTP-based Schemes", In Proc. of the 1st International Workshop on Privacy in Location-Based Applications (PiLBA 2008), Spain, 2008.
[17] T. Okamoto, S. Uchiyama, "A new public-key cryptosystem as secure as factoring", In Proc. of the Advances in Cryptology EUROCRYPT'98, 1998.
[18] M. Duckham, L. Kulit, "Location Privacy and Location-Aware Computing", In Dynamic and Mobile GIS: Investigating Changes in Space and Time, CRC Press, pages 35-52, 2007.
[19] M. Duckham, L. Kulit, "A formal model of obfuscation and negotiation for location privacy", In Pervasive Computing, pages 152-170, 2005.
[20] M. Duckham, K. Mason, J. Stell, M. Worboys, "A formal approach to imperfection in geographic information", Computers, Environment and Urban Systems, Vol. 25, No. 1, pages 89-103, 2001.
[21] C. A. Ardagna, M. Cremonini, E. Damiani, S. De Capitani di Vimercati, P. Samarati, "Location privacy protection through obfuscation-based techniques", In S. Baker, G. Ahn, eds.: Data and Applications Security, pages 47-60, 2007.
[22] M. L. Yiu, S. S. Jensen, X. Huang, H. Lu, "Spacetwist: Managing the trade-offs among location privacy, query performance, and query accuracy in mobile services", In Proc. of the IEEE 24th International Conference on Data Engineering (ICDE'08), pages 366-375, 2008.
[23] G. Ghinita, P. Kalnis, A. Khoshgozaran, C Shahabi, K. Tan, "Private queries in location based services: Anonymizers are not necessary", In Proc. of the 2008 ACM SIGMOD international conference on Management of data, pages 121-132, Canada, 2008.
[24] L. Pareschi, D. Riboni, C. Bettini, "Protecting users' anonymity in pervasive computing environments", In Proc. of the 6th Annual IEEE International Conference on Pervasive Computing and Communication (PERCOM'08), pages 11-19, 2008.
[25] J. Benaloh, "Dense probabilistic encryption", In Selected Areas of Cryptography (SAC 1994), pages 120-128, 1994.
[26] D. Boneh, E. J. Goh, and K. Nissim, "Evaluating 2-DNF formulas on ciphertexts", In Joe Kilian, editor, TCC 2005, volume 3378 of Lecture Notes in Computer Science, pages 325-341, 2005.

Digital Object Identifier: 10.4108/ICST.COLLABORATECOM2009.8374
http://dx.doi.org/10.4108/ICST.COLLABORATECOM2009.8374