Collusion-Resistant Query Anonymization for Location-Based Services

Qin Zhang and Loukas Lazos
Dept. of Electrical and Computer Engineering, University of Arizona, Tucson, Arizona
Email: {qinzhang, llazos}@email.arizona.edu

***Abstract*—We address the problem of anonymizing user queries when accessing location-based services. We design a novel location and query anonymization protocol called MAZE that preserves the user privacy without relying on trusted parties. MAZE guarantees the user's anonymity and privacy in a decentralized manner using P2P groups. Compared to prior works, MAZE enables individual user authentication for the purpose of implementing a pay-per-use or membership subscription model and is resistant to collusion of the P2P users. We extend MAZE to L-MAZE, a multi-stage protocol that is resistant to collusion of the P2P users with the LBS, at the expense of higher communication overhead.**

I. Introduction

Sharing one's location with service providers has enabled a wealth of personalized services typically referred to as, *Location-Based Services*, including the discovery of points of interest, localized traffic updates, and social networking. In a location-based service scenario, the user consents to disclose his location in exchange for receiving location-relevant information. However, oftentimes a query reveals personal information of sensitive nature. For instance, a user querying about specialized hospitals implicitly reveals a possible medical illness.

To ease privacy concerns raised by authenticated information retrieval, several location-based services are provisioned anonymously. Intuitively, protecting user privacy by excluding his identity from his interactions with a location-based server (LBS) should disassociate his identity from his queries. However, knowledge of one's location may be sufficient to link the user to his identity and eventually, his queries. For example, a user submitting a query from his private residence may become uniquely identifiable. On the other hand, location-based services that implement a subscription or "pay-per-use" model require implicit or explicit user authentication. Anonymity and authentication requirements are seemingly antithetic goals. The former requires the concealment of one's identity, while the latter requires its disclosure. *In this article, we address the problem of preserving the user location privacy and anonymity when accessing authenticated location-based services.*

Previously proposed solutions for preserving location privacy primarily rely on a trusted third party known as the location anonymizer server (LAS) [1], [7], [9], [12]. Using an intermediate trusted party for query anonymization has several drawbacks. First, the LAS constitutes a single point of failure. A breach of the LAS exposes users' privacy. Second, the anonymity problem is moved from the LBS to the LAS. The users' location privacy is not preserved with respect to the LAS, even if it is assumed to be trusted.

These shortcomings are addressed by decentralized schemes which eliminate the third trusted party by anonymizing queries in a peer-to-peer (P2P) fashion [4], [6], [11]. However, users may need to disclose their locations to the P2P group members, to perform peer discovery and compute the cloaking region. Users' location privacy is not preserved among the P2P group members. Moreover, existing methods do not address the problems of collusion among the participating parties and of provision of explicit user authentication at the LBS.

Main Contributions: We design a novel query anonymization scheme called MAZE that is resistant to collusion. Specifically, MAZE provides the following attractive features:

(a) Location privacy is achieved in a decentralized manner using P2P groups, without the need for a trusted LAS.
(b) User queries are k-anonymized.
(c) The LBS can authenticate the users submitting queries.
(d) MAZE is resistant to collusion of any number of users.

Finally, we extend MAZE to a multi-stage protocol called L-MAZE, which is resistant to collusion of up to $(L-1)$ users with the LBS, at the expense of higher communication cost.

The remainder of the paper is organized as follows. In Section II, we highlight related work. Our model assumptions are presented in Section III. The details of the MAZE protocol are described in Section IV. In Section V, we analyze the privacy of MAZE. In Section VI, we extend MAZE to the L-MAZE protocol. The communication and network overhead analysis is presented in Section VII. We conclude in Section VIII.

II. Related Work

State-of-the-art methods anonymize location information by reducing the temporal and spatial resolution of the user's position, a technique termed as *spatial cloaking*. The basic idea of spatial cloaking is to "blur" the user's exact location into a cloaked area where at least $(k-1)$ other users are present. This concept is termed as k-anonymity and was first introduced for protecting published medical data [17]. Several works have adopted spatial cloaking and k-anonymity in mobile environments. These works can be classified to centralized schemes [1], [7], [9], [12], and decentralized ones [4], [6], [10], [11]. We focus our attention to the latter category.

Chow et al. proposed a P2P k-anonymous group formation method where the spatial cloaking region is collaboratively computed based on the users' exact locations [4]. One of the P2P group participants is randomly selected to act as an agent, responsible for forwarding the group's queries to the LBS and distributing the responses to individual users. Since the exact

user locations are needed to compute the spatial cloaking region, this P2P method does not preserve location privacy within the P2P group. Magkos et al. proposed a distributed privacy-preserving scheme where users submit queries via a set of peers that act as information mixes and re-encrypt messages before sending them to the LBS provider. The authors also discuss the idea of cryptographic homomorphism for the purpose of aggregating multiple queries into single messages. Compared to MAZE, the works in [4], [6], [11] do not allow for user authentication at the LBS. Only the id of the agents forwarding the queries become known to the LBS.

Query anonymization and user authentication was jointly achieved via the use of group signatures [3], ring signatures [15], and the theoretical concept of private information retrieval (PIR) [5]. Group signature techniques rely on a central authority for setting up and managing the group membership and associated pseudonyms, thus adding considerable communication overhead. Ring signatures allow users to use a ring of pseudonyms for creating their own privacy cloaks. PIR-based methods incur high communication and computation overhead while large portions of the LBS database is exposed to users.

III. Problem Statement and Model Assumptions

Problem Statement: We address the problem of disassociating the user identity u_i submitting a query q_i to an LBS, from the contents of q_i and user location ℓ_i. To quantify the anonymity level, we adopt the k-anonymity metric [13]:

Definition 1: k-anonymity: A dataset is said to be k-anonymized, if each record is indistinguishable from at least $(k-1)$ other records. In our context, a q_i submitted by a user u_i is said to be k-anonymized, if q_i can be attributed to at least $(k-1)$ other users with equal probability.

To prevent the association of q_i with u_i, we adopt a spatial cloaking technique that obfuscates the user's exact location to the *k-anonymizing spatial region* (k-ASR) [6], [7]. The k-ASR is defined as a disk $C(\ell_c, \alpha)$ with radius α, centered at ℓ_c, which contains at least $(k-1)$ other users. To preserve the location privacy in high user density areas, we also adopt the metric of *privacy resolution tolerance* (PRT), defined as follows:

Definition 2: Privacy resolution tolerance: The privacy resolution tolerance ϕ_i is defined as the radius of a disk $C(\ell_c, \phi_i)$ that determines all candidate locations of a user u_i submitting a q_i to an LBS. Candidate locations are equally probable.

We further define the user privacy profile as follows:

Definition 3: User Privacy Profile: The privacy profile of a user u_i is defined as $P_i : \langle k_i, \phi_i \rangle$, and can take values from a *finite ordered* set of privacy levels $\mathcal{P}=\{P^a, P^b, \ldots, P^w\}$, corresponding to privacy values $\{\langle k^a, \phi^a \rangle, \langle k^b, \phi^b \rangle, \ldots, \langle k^w, \phi^w \rangle\}$. For P^a and P^b with $a < b$, it holds that $k^a < k^b$ and $\phi^a \leq \phi^b$. A user u_i that has selected P_i^a is willing to accept any higher level profile of a user u_j. This flexibility allows the participation of users with different privacy profiles to the same P2P group.

The user privacy profile allows the user to control his privacy level (similar suggestions have been made in [10]). Because the user is likely unfamiliar with the technical parameters k, ϕ, the different privacy levels may be abstracted to a fuzzy set. For example, privacy levels $\{P^a, P^b, P^c\}$ can be translated to fuzzy set $\{LOW, MED, HIGH\}$.

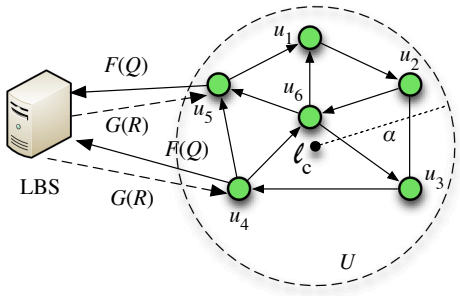


Fig. 1. The MAZE protocol architecture.

We design MAZE to satisfy the following requirements: (a) the queries submitted by any user u_i must be k_i-anonymous, (b) the user location cannot be determined beyond the PRT level ϕ_i, (c) the LBS can authenticate all queries originating from legitimate users (subscribers), and (d) the LBS can charge a user u_i for obtaining a response r_i to q_i.

Model Assumptions: Users form an overlay P2P network, facilitated by an existing infrastructure such as a set of base-stations, or by ad-hoc communications. The confidentiality and authenticity of P2P and peer-to-LBS communications is guaranteed via cryptographic methods such as symmetric or asymmetric cryptography. Any node pair can establish pairwise symmetric keys, when necessary. This can be facilitated via a public key infrastructure or symmetric key pre-deployment. Users are assumed to be *quasi-static* relative to their PRT area and are expected to be in the same group for several query rounds.

Collusion Model: We assume that none of the network participants is trusted. Peers may collude with other peers or with the LBS to compromise the privacy of a targeted user. However, network participants are *honest but curious*. Despite their curious nature, all peers honestly participate in the system and do not launch any active denial-of-service attacks such as dropping, modifying, or misrouting packets. Their goal is to associate query q_i with the identity u_i of that issuing user.

IV. The MAZE Protocol

A. MAZE Architecture

MAZE consists of three phases: a group formation phase, a query anonymization phase, and a query service phase. In the group formation phase, a set of users $\mathcal{U} = \{u_1, u_2, u_3, \cdots, u_k\}$ forms a P2P group that satisfies the privacy profile of each participating user. In the query anonymization phase, members of $\mathcal{U}$ anonymize their queries $\mathcal{Q} = \{q_1, q_2, \cdots, q_k\}$ by applying a transformation $\mathcal{F}(\mathcal{Q})$. The transformed set $\mathcal{F}(\mathcal{Q})$ is collectively submitted to the LBS for service. In the query service phase, the LBS authenticates each member of $\mathcal{U}$. It then obtains $\mathcal{Q}$ by applying the inverse transformation $\mathcal{F}^{-1}$ and prepares a response set $\mathcal{R} = \{r_1, r_2, \cdots, r_k\}$, without being able to associate any query to its originator. The LBS encrypts the response set R by applying an encryption transformation $\mathcal{G} = \{g_1, g_2, \ldots, g_k\}$. Encrypted responses $\mathcal{G}(\mathcal{R})$ are sent to

all members of $\mathcal{U}$. Each member extracts r_i from $\mathcal{G}(\mathcal{R})$ by applying an inverse transformation (decryption) g_i^{-1}.

Fig. 1 shows the MAZE protocol. A group of six users forms a P2P group $\mathcal{U}$ that satisfies the privacy profile of each member. Users in $\mathcal{U}$ send $\mathcal{F}(\mathcal{Q})$ to the LBS. The LBS responds with the $\mathcal{G}(\mathcal{R})$. Each member extracts their individual response r_i. We now describe the three phases of MAZE in detail.

B. Phase I: Group Formation Phase

The group formation phase is initiated by any user that does not belong to a group. We denote the group initiator by $\hat{u}_i$. Based on his privacy profile $P_i =< k_i, \phi_i >$, user $\hat{u}_i$ selects k_i, and the k-ASR radius $\alpha \geq \phi_i$. User $\hat{u}_i$ also selects the k-ASR center ℓ_c at random within $C(\ell_i, \alpha)$. Point ℓ_c is uniformly selected within $C(\ell_i, \alpha)$ to prevent the *center-of-ASR* attack [8]. Disk $C(\ell_c, \alpha)$ becomes the k-ASR for $\mathcal{U}$ initiated by $\hat{u}_i$. We note that selection of the k-ASR is a well-studied topic (e.g., [4], [6], [7]) and is not considered a main contribution of our work. Any of the k-ASR selection methods can be adopted for our purposes. We include a mechanism adapted to MAZE for completion. To form $\mathcal{U}$, the following steps are executed.

Step 1: $\hat{u}_i$ broadcasts a group formation message

$$m_f : g_{id} \parallel (\ell_c, x\alpha) \parallel \hat{u}_i \parallel x \parallel P_i,$$

where g_{id} is a random group id and x is a message counter for m_f's with a fixed g_{id}. The value of x is initially set to one.

Step 2: A user u_j located within k-ASR $C(\ell_c, x\alpha)$ replies with a join message

$$m_j : g_{id} \parallel join \parallel u_j,$$

if $P_i \geq P_j$ (i.e. $k_i \geq k_j$ and $x\alpha \geq \phi_j$).

Step 3: If the number of join messages received by $\hat{u}_i$ is less than $(k-1)$, $\hat{u}_i$ increases x by one, and repeats Steps 1 and 2. To avoid duplicate join requests, users within $C(\ell_c, (x-1)\alpha)$ do not respond to requests with the same g_{id}.

Step 4: The group initiator $\hat{u}_i$ randomly selects $(k-1)$ users that replied with a join message and forms group $\mathcal{U} = \{u_1, u_2, \ldots, u_k\}$ identified by g_{id}. It then notifies all group members of $\mathcal{U}$, by broadcasting an accept message

$$m_a : \{u_1, u_2, \cdots, u_k\} \parallel g_{id} \parallel \hat{u}_i \parallel accept.$$

Step 2 of the group formation phase can be adjusted to take into account available mobility information. In a mobile scenario, a user can utilize knowledge of his direction and velocity to estimate the period of time that he will remain within the k-ASR and decide whether to respond to a group formation message. This decision is independently made by each user without revealing mobility information to other users.

The mechanism for disseminating messages for the formation of $\mathcal{U}$ depends on the underlying network architecture. In infrastructure-based networks, all communication among peers is realized via base stations. For an ad hoc deployment, we employ a controlled flooding mechanism. In Step 1, the group initiator $\hat{u}_i$ broadcasts m_f. Any user within the k-ASR that receives m_f for the first time re-broadcasts it. Users can identify duplicate requests based on the unique g_{id} and the message counter x. Users within the k-ASR re-broadcast m_f. The propagation of m_f terminates at the boundaries of the k-ASR, since nodes outside that region ignore m_f. Any user u_j that has received an m_f which satisfies his privacy profile, unicasts message m_j to $\hat{u}_i$. The group initiator randomly selects $(k-1)$ replies and forms group $\mathcal{U}$ by sending the accept message m_a to the group members. At this stage, group members form an overlay network using a broadcast routing algorithm for ad hoc networks. When the insufficient number of users are located within $C(\ell_c, \alpha)$ (Step 3), users within the extended k-ASR, flood m_f similarly to the flooding of the original request. However, only users that did not reply in all previous requests, unicast m_j to $\hat{u}_i$. Once $\hat{u}_i$ obtains at least $(k-1)$ replies, it broadcasts m_a to the corresponding users.

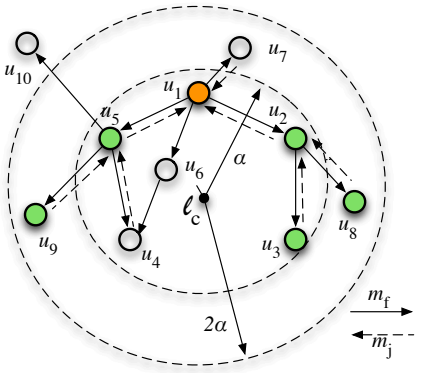


Fig. 2. Group formation phase in an ad hoc network.

The group formation phase for an ad hoc network of 10 users is shown in Fig. 2. In this example, u_1 is the group initiator, with a privacy profile P_1:$\langle 6, \alpha \rangle$. User u_1 broadcasts m_f, indicating the k-ASR boundaries $C(l_c, \alpha)$ and P_1. Users u_2, u_5, and u_6 re-broadcast m_f, since they are located within the k-ASR. Similarly, users u_3 and u_4 re-broadcast m_f, after receiving it from u_2 and u_6, respectively. On the other hand, users u_7, u_8, u_9, and u_{10} do not propagate m_f, since they are outside the k-ASR. Out of the five users within the k-ASR, only four reply with a join message. User u_6 does not join the P2P group due to its high privacy requirement. Because at least six nodes are needed to satisfy the anonymity value $k = 6$, user u_1 expands the k-ASR to $C(l_c, 2\alpha)$ and re-broadcasts m_f. Under the new k-ASR, users u_7, u_8, and u_9 reply with a join message. User u_1 randomly picks the five members of $\mathcal{U}$. The group formation phase terminates with the notification of $\mathcal{U} = \{u_2, u_3, u_5, u_8, u_9\}$ with an accept message m_a.

The group formation phase culminates in a P2P group of k users satisfying their privacy profiles. Note that users do not reveal their location while joining a group. The only information that is provided is that every user is within the k-ASR.

C. Phase II: Query Anonymization Phase

In this phase, each user anonymizes his query by applying an All-Or-Nothing (AONT) transformation [14], [16].

Definition 4: AONT: A transformation f:$\{\mathbb{F}_u\}^n \rightarrow \{\mathbb{F}_u\}^{n'}$, mapping a message $y = \{y^1, y^2, \ldots, y^n\}$ to pseudo-messages $s = \{s^1, s^2, \ldots, s^{n'}\}$ is an AONT if: (a) f is a bijection, (b) it is infeasible to obtain any part of y, if one of the s^i's is unknown, and (c) f and its inverse f^{-1} are efficiently computable.

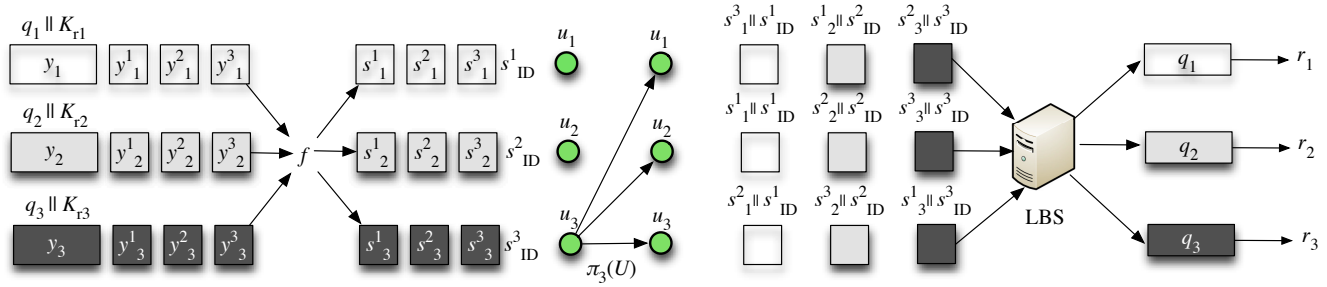


Fig. 3. Query anonymization phase: each query and random key is split to k=3 messages that are transformed to three pseudo-messages by AONT f. The pseudo-messages are permuted among the k members using a random permutation. Each member sends one pseudo-message from each query to the LBS.

In this definition, $\mathbb{F}_u$ denotes the alphabet of message blocks y^i, s^i, and n' denotes the number of output pseudo-messages. The main idea of an AONT is to prevent the recovery of any part of y, if any pseudo-message s^i is not known. In essence, AONTs can be considered to be an (n', n')-threshold scheme [14]. To anonymize $\mathcal{Q}$, group $\mathcal{U}$ executes the following steps.

Step 1: Each user generates a random key K_{r_i}. This key is used to encrypt/decrypt the response provided by the LBS.

Step 2: Each user transforms $y_i : q_i \| K_{r_i}$ to pseudo-messages $s_i = \{s_i^1, s_i^2, \ldots, s_i^k\}$ by applying an AONT such as the package transform [14] or the linear transform [16].

Step 3: Each user selects a random permutation $\pi_i(\mathcal{U})$ of the user set $\mathcal{U}$ and sends pseudo-message $s_i^j \| s_{ID}^i$ to user $\pi_i(\mathcal{U})(j)$. Here, s_{ID}^i denotes a random unique identifier for message s_i, so that pseudo-messages belonging to the same s_i can be correlated at the LBS. Pseudo-messages along with the message identifier $s_i^j \| s_{ID}^i$ are encrypted with the symmetric key shared between user u_i and user $\pi_i(\mathcal{U})(j)$.

Step 4: Each user u_i encrypts all received pseudo-messages with the pairwise key shared between u_i and the LBS. It then sends to the LBS signed message

$$m_i : E_{K_{u_i,LBS}} \Big(s_1^{\pi_1^{-1}(\mathcal{U})(i)} \parallel s_{ID}^1, s_2^{\pi_2^{-1}(\mathcal{U})(i)} \parallel s_{ID}^2, \ldots, s_k^{\pi_k^{-1}(\mathcal{U})(i)} \parallel s_{ID}^k \parallel u_i \parallel (l_c, \alpha) \Big) \parallel u_i \parallel g_{id} \parallel sig_{u_i}(m_i).$$

The four steps of the anonymization phase are shown in Fig. 3. In this example, the P2P group consists of three users. Queries $q_i \| K_{r_i}$ are split to three pseudo-messages. The recipient of each pseudo-message is selected according to the random permutation $\pi_i(\mathcal{U})$ generated by each user. Each user forwards one pseudo-message from every query to the LBS.

D. Phase III: Query Service Phase

In this phase, the LBS individually authenticates each user in $\mathcal{U}$ and serves the queries in $\mathcal{Q}$ via the following steps.

Step 1: The LBS verifies $sig_{u_i}(m_i)$ and decrypts m_i using $K_{u_i,LBS}$. It then groups all messages carrying the same g_{id}.

Step 2: The LBS recovers $s_i, \forall u_i \in \mathcal{U}$, using the message ids s_{ID}^i. It then reconstructs q_i, $\forall u_i \in \mathcal{U}$ by applying the inverse AONT f^{-1} on $f(\mathcal{Q})$.

Step 3: The LBS prepares response set $\mathcal{R} = \{r_1, r_2, \ldots, r_k\}$. Response r_i serves query q_i.

Step 4: The LBS sends $g(\mathcal{R}) = \{E_{K_{r_1}}(r_1), E_{K_{r_2}}(r_2), \ldots, E_{K_{r_k}}(r_k)\}$ to all users in $\mathcal{U}$.

Step 5: Each u_i obtains r_i by decrypting $E_{K_{r_i}}(r_i)$ with K_{r_i}.

At the termination of this phase, each user in $\mathcal{U}$ has received its individual response r_i. Moreover, the LBS has authenticated every user participating in $\mathcal{U}$, individually.

V. Privacy Analysis

Location Privacy in $\mathcal{U}$–We first show that MAZE preserves the location and query privacy of all members in $\mathcal{U}$.

Proposition 1: For every user $u_i \in \mathcal{U}$, with profile $P_i = \langle k_i, \phi_i \rangle$, no user $u_j \in \mathcal{U}$, $j \neq i$ can determine u_i's position ℓ_i, at an accuracy greater than ϕ_i.

Proof: During the group formation phase, the group initiator $\hat{u}_i$ determines the k-ASR $C(\ell_c, \alpha)$ by uniformly selecting ℓ_c within disk $C(\ell_i, \alpha)$. Therefore, all points of $C(\ell_i, \alpha)$ are candidate centers of the k-ASR with equal probability. The group formation message m_f contains only $C(\ell_c, \alpha)$, where α is selected such that $\alpha \geq \phi_i$. Thus, any user receiving m_f cannot localize $\hat{u}_i$ with accuracy greater than α, which satisfies $\hat{u}_i$'s location privacy requirement ϕ_i.

Any user u_j replying to m_f with m_a, acknowledges to be within $C(\ell_c, \alpha)$, without revealing his exact location. Since u_j replies to m_f only if $P_i \geq P_j$, it follows that $\alpha \geq \phi_j$. Moreover, users can only belong to a single group. This requirement prevents a k-ASR intersection attack, in which the location of a user is computed as the intersection of several k-ASRs that include the same user. Furthermore, we note that $\alpha \geq \phi_j$ if Step 3 of the group formation phase becomes necessary. This is because the k-ASR radius is increased by α in consecutive executions of Step 3 and hence, users in $C(\ell_c, (x+1)\alpha)$ - $C(\ell_c, x\alpha)$ achieve a PRT equal to α. ■

Query Privacy in $\mathcal{U}$–For query q_i of $u_i \in \mathcal{U}$ it holds:

Proposition 2: Collusion of up to $(k-1)$ users in $\mathcal{U}$ does not reveal the query of the k^{th} user, or the corresponding response.

Proof: This is a direct consequence of the use of AONTs. During the anonymization phase, a query q_i is partitioned to k pseudo-messages, $(k-1)$ of which are distributed to $(k-1)$ other users. The collusion of $(k-1)$ users can recover $(k-1)$ pseudo-messages. According to the definition of an AONT, the combination of any $(k-1)$ pseudo-messages does not reveal any information about $q_i \parallel K_{r_i}$. Similarly, the collusion of up to $(k-1)$ users cannot recover K_{r_i}. Therefore, no user but u_i can decrypt the query response and obtain r_i. ■

Location Privacy at the LBS–The reconstructed query set at the LBS specifies the k-ASR via the inclusion of $C(l_c, x\alpha)$.

Because a user u_i participates in $\mathcal{U}$ only if $\alpha \geq \phi_i$, the LBS cannot learn u_i's location at an accuracy greater than ϕ_i.

Query Anonymization at the LBS–We now show that the LBS cannot associate a q_i with an identity u_i.

Proposition 3: MAZE preserves k-anonymity.

Proof: In Step 4 of the query anonymization phase, every $u_i \in \mathcal{U}$ sends to the LBS: (a) a message that contains one pseudo-message from *every query* in $\mathcal{Q}$ and (b) the associated query ids s^i_{ID} corresponding to each query q_i. Using s^i_{ID}, the LBS can reconstruct $\mathcal{Q}$, but it cannot link q_i to a u_i, since the s^i_{ID} for each query is not related to u_i (s^i_{ID} is a nonce changing with every query). Hence, all users in $\mathcal{U}$ are equally likely to have generated q_i, and k-anonymity is preserved. ∎

User authentication at the LBS–It is straightforward to verify that MAZE allows for the authentication of each user in $\mathcal{U}$. Using the message signature included in Step 4 of the query anonymization phase the LBS can authenticate the user identity submitting a query as part of the group $\mathcal{U}$.

VI. The L-MAZE Protocol

MAZE does not preserve the query anonymity when the LBS colludes with one of the users in $\mathcal{U}$. This is because members of $\mathcal{U}$ can associate a query id s^i_{ID} with u_i that generated it. Moreover, the LBS can associate q_i with s^i_{ID}. Thus, collusion of the LBS with a user leads to the association of u_i with q_i.

In this section, we develop L-MAZE, an anonymization protocol that is resistant to collusion of up to $(L-1)$ users with the LBS, at the expense of a higher communication cost. To preserve anonymity, we employ an L-stage decryption mixnet [2]. Mixnets are cryptographic systems that implement an anonymous channel between two parties, a sender and a receiver [3]. These systems involve the combination of encryption/decryption operations and shuffling and/or permutation operations that prevent message tracing. In our context, the role of the sender is assumed by a user u_i that wants to anonymize query q_i. The role of the receiver is assumed by the set of colluding users and the LBS. The disassociation of the user's identity u_i from the message s^i_{ID} effectively addresses the collusion problem. In L-MAZE, the group formation phase and query service phase remain identical to those of MAZE. The query anonymization phase is modified as follows.

A. L-MAZE: Query Anonymization Phase

Step 1: Each user generates a random key K_{r_i}.

Step 2: Each user transforms y_i:$q_i \| K_{r_i}$=$\{y^1_i, y^2_i, \ldots, y^{k'}_i\}$ to pseudo-messages $s_i = \{s^1_i, s^2_i, \ldots, s^k_i\}$ by applying an AONT.

Step 3: Each user u_i generates a mixnet matrix $M^i_{k\times L}$. Every column of $M^i_{k\times L}$ is a permutation of $\mathcal{U}$. Every row of $M^i_{k\times L}$ is a sub-permutation of $\mathcal{U}$. The initial column permutation is selected at random. The remaining column permutations are also selected at random from the set of possible permutations that satisfy the row sub-permutation requirement. One such $M^i_{k\times L}$ for $k = 5$ and $L = 3$ is shown in Fig. 4(a).

Step 4: Every user u_i with pseudo-messages $s_i = \{s^1_i, s^2_i, \ldots, s^k_i\}$ encrypts each pseudo-message $s^j_i \| s^i_{ID}$, L times using the sequence of keys of the set of users denoted by row j of $M^i_{k\times L}$ (for simplicity, we denote $M^i_{k\times L}$ as M):

$$E_{k_{i,M(j,1)}}\left(\ldots E_{k_{i,M(j,L)}}(s^j_i \| s^i_{ID} \| u_{M(j,L)}) \| u_{M(j,1)}\right),$$

where $k_{i,j}$ is a pairwise key shared between u_i and u_j.

Step 5: Encrypted pseudo-messages are broadcasted to $\mathcal{U}$.

Step 6: At each stage, an intended receiver removes one encryption layer from each pseudo-message it receives. In total, a user recovers $(k-1)$ pseudo-messages per stage. Steps 5 and 6 are repeated $(L-1)$ times (L stages in total).

Step 7: After all layers encryption layers are removed, each user u_i encrypts all received pseudo-messages and u_i with $K_{u_i,LBS}$ and signs it. It then sends all encrypted pseudo-messages to the LBS.

B. Privacy Analysis

1) Correctness: We first show that in L-MAZE, the LBS can reconstruct all user queries, using the following proposition.

Proposition 4: At every mixing stage, every user in $\mathcal{U}$ holds exactly one pseudo-message from each query.

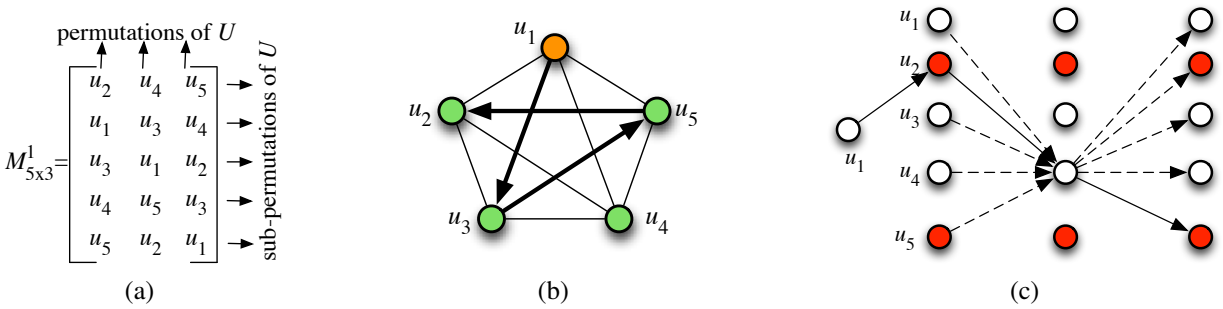
Proof: Consider y_i=$q_i \| K_{r_i}$ transformed to k pseudo-messages s_i=$\{s^1_i, s^2_i, \ldots, s^k_i\}$ via an AONT. At every mixnet stage, the encrypted s^j_i's are sent to a permutation of $\mathcal{U}$ (columns of $M^i_{k\times L}$ are permutations of $\mathcal{U}$). Hence, a user receives exactly one encrypted s^j_i from each other user. ∎

According to Proposition 4, at the end of the L^{th} stage, each user holds one pseudo-message from each query (k total). In Step 7, all pseudo-messages are forwarded to the LBS. The LBS reconstructs the s_i's using each s^i_{ID} and recovers all q_i's by applying the inverse AONT. To serve the q_i's, the LBS constructs and encrypts each response r_i with the corresponding key K_{r_i}. Each user decrypts his own r_i by using K_{r_i}.

2) Query anonymization under collusion: We now show that L-MAZE is resistant to the LBS collusion with up to $(L-1)$ users. Our purpose is to show when the LBS colludes with any $z \leq (L-1)$ users, the queries submitted by the $(k-z)$ non-colluding users are indistinguishable. This is the best case scenario since the colluding users already reveal their queries to the LBS. To illustrate the collusion resistance property, we model the mixnet represented by a matrix $M^i_{k\times L}$ as a set of paths on a complete graph G_k, in which the vertex corresponds to set $\mathcal{U}$. An example of such a graph for the mixnet of Fig. 4(a) is shown in Fig. 4(b). Using this graph model, the mixnet operations applied to a pseudo-message s^j_i are represented as a path $P^j_i = \{i, M(j,1), M(j,2), \ldots, M(j,L)\}$. This path has the following properties.

Proposition 5: Each path $P^j_i, j = 1, \ldots, k$ corresponding to the mixnet operation applied to pseudo-message $s^j_i, j = 1, \ldots, k$ contains at least one non-colluding user.

Proof: A path P^j_i consists of exactly L vertices corresponding to the L mixnet stages. These vertices are denoted by each row of $M^i_{k\times L}$. By construction, each row j of $M^i_{k\times L}$ is an L size sub-permutation of $\mathcal{U}$. Therefore, all L vertices of P^j_i are distinct. Given that at most $(L-1)$ users collude, every P^j_i contains at least one non-colluding user. ∎



(a) (b) (c)

Fig. 4. (a) MIXNET matrix $M^1_{5\times 3}$, for a set of five users providing resistance to the collusion of the LBS with up to two members of $\mathcal{U}$, (b) a complete graph G_5. Any path originating from u_1 that visits nodes of G_5 according to the rows of the mixnet $M^1_{5\times 3}$, traverses at least one non-colluding user, (c) in 3-MAZE, when users u_2 and u_5 collude, pseudo-messages sent from u_1 are mixed at u_4 during the second stage of the mixnet, thus preventing the tracing of s^1_1.

We can now show the following proposition.

Proposition 6: L-MAZE preserves the query anonymity of the $(k-z)$ non-colluding users when $z \leq (L-1)$ users collude with the LBS.

Proof: According to Proposition 4, at each L-MAZE stage, every user obtains exactly one pseudo-message from the k queries generated in $\mathcal{U}$. Moreover, according to Proposition 5, the pseudo-messages of a transformed query s_i follow paths that contain at least one non-colluding user. Consider a non-colluding user u_{nc} receiving k messages from $\mathcal{U}$ (one of these messages originates from u_{nc} himself). Out of the total k encrypted pseudo-messages received by u_{nc}, z of them are known to the LBS since they belong to the colluding users. The remaining $(k-z)$ messages were generated by non colluding users. These unknown $(k-z)$ pseudo-messages are encrypted with keys shared between u_{nc} and the non-colluding users. Therefore, colluding users cannot obtain those pseudo-messages.

When one layer of encryption is removed at u_{nc}, the incoming $(k-z)$ encrypted pseudo-messages become uncorrelated to the $(k-z)$ outgoing ones. Since the colluding users cannot correlate the $(k-z)$ incoming unknown pseudo-messages with the $(k-z)$ outgoing ones when mixed at u_{nc}, the linking of an s^i_{ID} to u_i becomes a random guess with probability of success equal to $1/(k-z)$. We emphasize that in Step 5 of L-MAZE, the encrypted pseudo-messages are broadcasted to the entire group $\mathcal{U}$. This is because a recipient of an encrypted pseudo-message is not aware of which user is supposed to receive the pseudo-message at the next stage of the mixnet ■

To illustrate Proposition 6 consider the mixnet shown in Fig. 4(c). Assume that u_2 and u_5 attempt to correlate a pseudo-message originating from u_1 with s^1_{ID}. In stage 1, u_2 identifies that the encrypted pseudo-message received from u_1, since the first transmission of any user must contain pseudo-messages generated by himself. User u_2 removes one encryption layer and forwards the pseudo-message to the next mixnet stage. Since u_4 is not colluding, u_2 and u_5 cannot identify which of u_1, u_3, and u_4 received the pseudo-message. Moreover, u_4 receives four more pseudo-messages, only two of which are known to u_2 and u_5. User u_4 remove another layer of encryption and forward all five received pseudo-messages to the next mixnet stage. Out of the five pseudo-messages forwarded to stage three, the three messages that belong to u_1, u_3, and u_4 are indistinguishable. Therefore, u_2 and u_5 can correlate those messages to their originators with probability $1/3$.

VII. Overhead Analysis

We now analyze the communication and network overhead of MAZE. The former is defined as the number of messages exchanged by the users in $\mathcal{U}$ via a one-hop overlay network. The network overhead is defined as the number of messages transmitted by nodes organized in a multi-hop ad hoc network.

A. Communication Overhead of MAZE

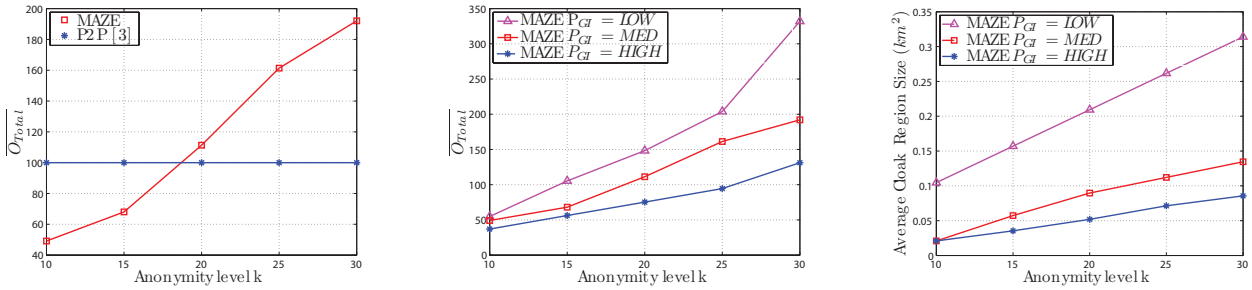
Let O_{gfp}, O_{qap}, and O_{qsp} denote the communication cost of the group formation, query anonymization, and query service phases, respectively. For the group formation phase, $O_{gfp} = k + x$. This accounts for the x messages m_f broadcasted by $\hat{u}_i$ until $(k-1)$ users are discovered within the k-ASR, the $(k-1)$ messages m_j unicasted back to the $\hat{u}_i$, and the group acceptance message m_a broadcasted by $\hat{u}_i$ to all $(k-1)$ users.

In the query anonymization phase, every user transforms his query to a set of k pseudo-messages of size $1/k$ relative to the query size, and transmits $(k-1)$ pseudo-messages to other users. After the query anonymization, every user sends one message to the LBS. Hence, the total number of messages transmitted during this phase is equal to $O_{qap} = ((k-1)\frac{1}{k}+1)k = (2k-1)$. For L-MAZE, the communication overhead of the query anonymization phase increases by $L-1$ times, due to the L-stage pseudo-message exchange. Finally, during the query service phase, the LBS sends one reply for every query it receives, accounting for a total of $O_{qsp} = k$ messages. Summing the communication overhead of all three phases yields a total communication overhead of $O_{Total} = 4k + x - 1$.

B. Network Overhead of MAZE

We further experimentally evaluated the network overhead of MAZE by randomly deploying n users within an area of $1,000\text{m}\times 1,000\text{m}$. Users formed a multi-hop ad hoc network based on their limited communication range which was set to 250m. For unicast transmissions, messages were routed using a minimum-hop path. Broadcast transmissions were routed using a broadcast tree spanning all users in $\mathcal{U}$. Results were averaged over 40 random network topologies.

We first compared the overhead of MAZE with the P2P anonymization protocol in [4] (though [4] does not preserve



(a) (b) (c)

Fig. 5. (a) Average network overhead as a function of the anonymity level k, (b) Comparison of the average network overhead between MAZE and 2-MAZE as a function of the anonymity level k.

location privacy in $\mathcal{U}$, is not resistant to user collusion, and does not provide user authentication). All users were assumed to have the same privacy profile. Fig. 5(a) shows the average network overhead $\overline{O_{Total}}$ as a function of k when the user density is set to $d = 0.0005$ users/m^2. We observe that for small values of k, MAZE yields lower overhead than the scheme in [4]. This overhead increases almost linearly with the anonymity level requirement. On the other hand, the protocol in [4] incurred a fixed network overhead since the search for peers during the group formation phase was limited to broadcasts within one hop (due to the high user density relative to k). Note that nearly all network overhead of the P2P scheme in [4] is incurred during the group formation phase. MAZE, on the other hand, incurs most of its overhead during the query anonymization phase where pseudo-messages are mixed among the P2P users.

In the second set of experiments, we studied the impact of the initial selection of the privacy level by the group initiator. We randomly set the privacy level of users to one of three privacy levels $\{LOW, MED, HIGH\}$ and computed the network overhead of MAZE when the privacy requirement of the group initiator was set to LOW, MED and $HIGH$, respectively. The results are shown in Fig. 5(b). We further show the required k-ASR size for satisfying the privacy profile of the group initiator. MAZE requires a larger cloaking region and incurs higher network overhead when the group initiator's privacy requirement level is set to LOW. This is because users with a MED or $HIGH$ privacy level do not reply to requests with low privacy requirements. On the other hand, when the privacy profile of the group initiator is set to $HIGH$, users of any level respond to u_i's request thus reducing the size of the cloaking region and the corresponding network overhead.

VIII. Conclusions

We addressed the problem of preserving the location privacy and user anonymity when receiving authenticated location-based services. We developed two privacy-preserving communication protocols that allows users to place queries to an LBS without revealing their identity, or current location beyond a desired accuracy. Moreover, our protocols allow the LBS to authenticate and charge any user that receives a location-based service. Our privacy analysis showed that the proposed protocols preserve the location privacy and query anonymity against any of the system participants, and are resistant to collusion.

Acknowledgments

This research was supported in part by the NSF under grants CNS-0844111 and CNS-1016943 and ARO grant W911NF-13-1-0302. Any opinions, findings, conclusions, or recommendations expressed in this paper are those of the author(s) and do not necessarily reflect the views of the NSF.

References

[1] B. Bamba, L. Liu, P. Pesti, and T. Wang. Supporting anonymous location queries in mobile environments with privacygrid. In *Proc. of the 17th International Conference on World Wide Web*, pages 237–246, 2008.

[2] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–90, 1981.

[3] D. Chaum and E. Van Heyst. Group signatures. In *Proc. of the EUROCRYPT Conference*, pages 257–265, 1991.

[4] C. Chow, M. Mokbel, and X. Liu. Spatial cloaking for anonymous location-based services in mobile peer-to-peer environments. *GeoInformatica*, 15(2):351–380, 2011.

[5] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K. Tan. Private queries in location based services: anonymizers are not necessary. In *Proc. of the SIGMOD Conference*, pages 121–132, 2008.

[6] G. Ghinita, P. Kalnis, and S. Skiadopoulos. PRIVE: anonymous location-based queries in distributed mobile systems. In *Proc. of the 16th International Conference on World Wide Web*, pages 371–380, 2007.

[7] M. Gruteser and D. Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *Proc. of the MobiSys Conference*, pages 31–42, 2003.

[8] P. Kalnis, G. Ghinita, K. Mouratidis, and D. Papadias. Preventing location-based identity inference in anonymous spatial queries. *IEEE Trans. on Knowledge and Data Engineering*, pages 1719–1733, 2007.

[9] P. Li, W. Peng, T. Wang, W. Ku, J. Xu, and J. Hamilton Jr. A cloaking algorithm based on spatial networks for location privacy. In *Proc. of the SUTC Conference*, pages 90–97, 2008.

[10] A. Loukas, D. Damopoulos, S. Menesidou, M. Skarkala, G. Kambourakis, and S. Gritzalis. MILC: A secure and privacy-preserving mobile instant locator with chatting. *Information Systems Frontiers*, 14(3):481–497, 2012.

[11] E. Magkos, P. Kotzanikolaou, S. Sioutas, and K. Oikonomou. A distributed privacy-preserving scheme for location-based queries. In *Proc. of the WoWMoM Symposium*, pages 1–6, 2010.

[12] M. Mokbel, C. Chow, and W. Aref. The new Casper: query processing for location services without compromising privacy. In *Proc. of the VLDB Conference*, pages 763–774, 2006.

[13] A. Pfitzmann and M. Kohntopp. Anonymity, unobservability, and pseudonymity: a proposal for terminology. In *Designing Privacy Enhancing Technologies*, pages 1–9, 2001.

[14] R. Rivest. All-or-nothing encryption and the package transform. In *Fast Software Encryption*, pages 210–218, 1997.

[15] R. Rivest, A. Shamir, and Y. Tauman. How to leak a secret. *Advances in Cryptology, ASIACRYPT*, pages 552–565, 2001.

[16] D. Stinson. Something about all or nothing (transforms). *Designs, Codes and Cryptography*, 22(2):133–138, 2001.

[17] L. Sweeney. Achieving k-anonymity privacy protection using generalization and suppression. *International Journal of Uncertainty Fuzziness and Knowledge-Based Systems*, 10(5):571–588, 2002.