Mutually Private Location Proximity Detection with Access Control

Michael G. Solomon, Vaidy Sunderam, Li Xiong, Ming Li

▶ **To cite this version:**

Michael G. Solomon, Vaidy Sunderam, Li Xiong, Ming Li. Mutually Private Location Proximity Detection with Access Control. 31th IFIP Annual Conference on Data and Applications Security and Privacy (DBSEC), Jul 2017, Philadelphia, PA, United States. pp.164-184, 10.1007/978-3-319-61176-1_9 . hal-01684358

HAL Id: hal-01684358

https://inria.hal.science/hal-01684358v1

Submitted on 15 Jan 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

Mutually private location proximity detection with access control

Michael G. Solomon[1], Vaidy Sunderam[1], Li Xiong[1], and Ming Li[2]

[1] Department of Mathematics & Computer Science, Emory University
Email: {msolo01, vss, lxiong}@emory.edu
[2] Department of ECE, University of Arizona, Email: ming.li@arizona.edu

Abstract. Mobile application users want to consume location-based services without disclosing their locations and data owners (DO) want to provide different levels of service based on consumer classifications, sometimes without disclosing areas of interest (AOI) locations to all users. Both actors want to leverage location-based services utility without sacrificing privacy. We propose a protocol that supports queries from different classifications of users, such as subscribers/non-subscribers, or internal/external personnel, and imposes embedded fine-grained access control without disclosing user or DO location information. We use Ciphertext Policy Attribute-Based Encryption (CP-ABE) and Hidden Vector Encryption (HVE) to provide flexible access control and mutually private proximity detection (MPPD). Our protocol minimizes expensive cryptographic operations through the use of location mapping with compressed Gray codes, each representing multiple locations. Our protocol encrypts AOI locations using HVE, and then encrypts AOI information using CP-ABE with an expressive access policy. Our protocol's use of these two encryption methods allows DOs to define a single set of AOIs that can be accessed by sets of users, each with potentially different access permissions. A separate service provider (SP) processes queries without divulging location information of the user or any DO provided AOI.

1 Introduction

Mobile applications increasingly incorporate location awareness into services they provide. Many such services rely on location to provide context-sensitive results, such as proximity alerts when a consumer approaches some defined area of interest. Today's smart phones, tablets, and other mobile devices make accurate location-sensing a commonly used feature. However, consumers of such features are becoming increasingly concerned over the loss of privacy resulting from ongoing location disclosure. Consumers want to use location-based services without sacrificing their privacy. Likewise, Data providers or data owners (DO) of areas

Research supported by AFOSR DDDAS grant FA9550-17-1-0006 and NSF TWC grant CNS-1618932

of interest (AOI) want to provide differing levels of service based on consumer classifications, without disclosing the locations of all AOIs to all consumers. For example, DOs may provide services of greater value to paid subscribers than to users who consume services for free. Alternatively, DOs may keep specific AOI locations private from some users in the interest public safety or to avoid public panic, as in the case of areas of active criminal activity or ongoing health hazards. DOs may want to restrict access to, and even awareness of, AOIs to consumers based on access permissions and location.

We propose a protocol that allows DOs to define a single set of AOIs with consumer access rules that limits AOI proximity alerts to specific groups of consumers. The locations of defined AOIs are kept private and not disclosed to consumers until they are in proximity to individual AOIs. The protocol fundamentally provides proximity detection based on a consumer's current location, without disclosing any consumer's location. Our protocol supports mutual privacy (privacy for the data provider and the consumer), along with embedded access control that allows DOs to control proximity alerts by consumer type.

1.1 Motivation

Consider Mary, a guest at the "Fun Times" amusement park. Mary wants to make the most of her day and desires to minimize time spent waiting in line for attractions or shows. Mary has subscribed to the "Fun Times InTheKnow app" premium service that sends information to her smart phone about nearby attractions and shows with short wait times. Bob is also in the "Fun Times" park and has the "InTheKnow" app, but Bob did not subscribe to the premium service. Bob only receives general information about attraction wait times for attractions that are somewhat close to his current location. Both Mary and Bob want to receive helpful information without disclosing their locations to "Fun Times." On the other hand, "Fun Times" wants to provide location-sensitive services to Mary and Bob without publishing all of the "Fun Times" AOIs. If "Fun Times" published all of their AOIs, no subscribers would need to pay for the subscription to the premium service. "Fun Times" protects their revenue stream by keeping their AOIs private. "Fun Times" can limit the information they provide to subscribers, and even define different subscriber levels based on subscription fees paid. "Fun Times" can also use this service to direct their employees to areas of the park that need cleaning, servicing, or even crowd management. Other use cases for such a location privacy preserving framework could include providing epidemiological related alerts or criminal activity investigation proximity with precision granularity based on clearance and/or "need to know".

1.2 Existing and Potential Solutions

A naive approach to location privacy could be to simply stop the sensing or release of physical location. However, this results in the loss of utility (e.g. navigation, lost device tracking, etc.) Alternatively, applications could introduce novel ways of exchanging and processing location information that protects the location owner's privacy without requiring user action. Application layer location privacy efforts generally focus on one of four main areas, each with its own

drawbacks: location perturbation (limited accuracy), access control (limited privacy/utility trade-off), private information retrieval (PIR) (lack of DO privacy), and encryption (performance cost). Each type of approach attempts to provide the ability for users to consume location based services without divulging their exact locations to any untrusted service provider (SP) or any DO, and in some cases, allow the DO to keep its AOI location data private as well. Existing approaches assume equal access to proximity detection for all AOIs. Traditional access controls require a trusted authority to make decisions at run time.

Existing MPPD solutions largely focus on location as the only criteria for determining proximity to a defined area. In some cases, additional attributes should be considered, such as security level or subscriber status, before providing proximity responses. A useful MPPD protocol should allow a single set of AOIs to service a large group of diverse users. Such a protocol would only provide proximity alerts for users that are both near a defined AOI and meet requirements set by the data owner. This is normally accomplished by separating the MPPD functionality from the authorization phase. This requires a third party to examine each user and AOI attributes to determine if a proximity alert is allowed, based on data owner policy. In this scenario, the third party possesses substantial information about users and AOIs.

1.3 Our Contributions

We propose a novel method of providing Mutually Private Proximity Detection (MPPD) with embedded fine-grained access control. Our protocol provides fine-grained access control that is embedded in the encryption technique. That is, decryption is only successful when attempted with an authorized user's key. Our method does not disclose location data to any user, DO, or SP. Our protocol introduces a novel use of two existing techniques, Ciphertext Policy Attribute-Based Encryption (CP-ABE) to provide fine-grained access control based on descriptive consumer attributes, and Hidden Vector Encryption (HVE) to efficiently determine user proximity to AOIs. Neither CP-ABE, nor HVE, alone solve the problem of MPPD, but when both are integrated into a new protocol, work together to efficiently provide MPPD. To the best of our knowledge, there is only one other proposed protocol [24] that controls access to AOIs based on access rules the data owner supplies, along with MPPD. However, this protocol requires distance calculations for determining proximity to each AOI, which can be costly. Our protocol substantially reduces computational overhead and increases scalability by using HVE with compressed AOI location tokens to determine if a user's location overlaps one or more AOIs, along with CP-ABE to provide flexible fine-grained access control.

Our proposed protocol is novel in that it provides fine-grained flexible consumer access control, minimizes computational load on devices with limited processing power (eg. mobile devices), and also provides a high level of privacy guarantees for both users and DOs. The protocol supports these services and reduces the DO administrative workload by incorporating two state of the art partial solutions, drawing on the existing strengths of each approach (CP-ABE and HVE). Using this new protocol, DOs can create a single set of AOIs, along

with access policies for each AOI, that restricts user consumption of proximity alerts based on DO defined descriptive attributes. Users (consumers) must possess the attributes necessary to satisfy a DO defined access policy for each AOI to receive proximity alerts for that AOI. All of this functionality is provided without requiring any third party to access unencrypted location information to make access authorization decisions. Using CP-ABE, DOs can define the granularity of user proximity alert consumption based on its own defined access policies. The use of HVE provides the actual proximity determination of a user location to an AOI without disclosing either location to any party. The combination of CP-ABE and HVE provides a flexible and scalable approach to implementing MPPD with controlled access based on user classifications.

2 Related Work

Existing private proximity detection solutions generally fall into the following four areas, each with its own drawbacks: location perturbation (limited accuracy), access control (limited privacy/utility trade-off), private information retrieval (PIR) (lack of DO privacy), and encryption (performance cost). Current private proximity detection solutions, with the exception of [24], provide results based on location alone, and do not consider other attributes. Results filtering is left to a trusted third party that can examine attributes and learn user locations.

2.1 Location Perturbation and Transformation

Location perturbation schemes use obfuscated or perturbed user location data. K-anonymity is the most common technique to limit the ability to determine any user location. Kim [23, 21] proposed two different approaches to cloaking locations. One weakness of such schemes is that attackers can use the same cloaking techniques as authorized users to generate and analyze shared cloaked locations. Another weakness of these schemes is that the Location Based Server (LBS) only responds with answers of the same, or lower, precision of the obfuscated user location. Yiu [30] proposed a collection of schemes to strengthen the shared knowledge weakness by partitioning data using different criteria, but these techniques are still vulnerable to attacks based on a priori knowledge and brute-force attacks. Hossain [18] proposed shear-based spatial perturbation schemes. Recent work in cloaking [3, 29] extends differential privacy [11] and provides greater semantic privacy protection. However, all perturbation schemes reduce location data precision.

2.2 Access Control

Another approach is via structured access control techniques. Bugiel [7] proposed FlaskDroid, a fine-grained Mandatory Access Control (MAC) approach for Android devices. Li [24] proposed Privacy-preserving Location Query Protocol (PLQP). PLQP allows queries to access location information while still upholding user privacy and is efficient enough to operate on mobile devices. Lu [25] proposed Secure and Privacy-preserving Opportunistic Computing (SPOC) framework for health care data, which requires close proximity of a patient and

medical personnel to grant PHI access. Fawaz [13] proposed more fine-grained access control for sharing location data with third-party apps. Access control methods only provide binary access control with limited granularity for privacy protection, i.e. users can either grant or block access, and these approaches require a trusted third party to access locations to make access decisions.

2.3 Private Information Retrieval

PIR [10] allows users to issue DO queries without the DO learning the query's content. These techniques build private spatial indexes that utilize PIR operations, which can provide efficient spatial query processing while the underlying PIR scheme provides privacy. Hengartner [17] uses PIR to hide location information from an untrusted server and uses trusted computing to guarantee that the PIR algorithms are only performing the intended operations. Khoshgozaran [20] proposed two location privacy approaches that eliminate the need for an LBS anonymizer. Ghinita [14] uses PIR techniques to support Nearest-Neighbor (NN) queries without requiring a trusted third party. This approach uses cryptography to protect user locations and increases performance with data mining techniques to eliminate redundant calculations. PIR techniques can be efficient and provide a high level of privacy guarantees, but they assume that the AOI data is public, and provide no privacy for DOs.

2.4 Encryption

Other approaches use encryption for location data to provide privacy, requiring varying degrees of communication and computation in the encrypted space to determine proximity. Encryption techniques can be viewed as symmetric PIR, that is, PIR plus the restriction of AOI privacy, which is the main focus of our research. Although encryption introduces overhead, techniques that use it can be more resistant to attackers, even those with prior knowledge. Proposed encryption techniques can loosely be grouped into three general categories: spatial data encryption, novel encoding/notation, and homomorphic encryption schemes.

Spatial data encryption approaches include Khoshgozaran [19], who proposed a grid-based scheme that uses group shared symmetric keys, allowing users to query nearby cell contents and then locally decrypt location details for items encrypted with their group shared key. Wang's [28] approach performs a geometric range search on encrypted spatial data.

Other approaches depend on novel location encoding or notation. Ghinita [15] proposed a method based on HVE to protect user locations. Calderoni [8, 26] proposed a new compact data structure, Spatial Bloom Filter (SBF), to privately store AOI locations, and the Paillier cryptosystem to protect AOIs from disclosure while still allowing comparison with encoded (but unencrypted) user locations. Kim [22] proposed Hilbert Curve Transformation (HCT), which encodes locations using a Hilbert Curve and then encrypts the result using AES.

And finally, other approaches depend on homomorphic encryption to provide location protection. Elmehdwi [12] proposed Secure k-Nearest Neighbors (SkNN), which uses the Paillier cryptosystem and protocols to support private

k-NN queries. Choi [9] also uses Paillier cryptosystem, along with Order Preserving Encryption (OPE) [1] to detect proximity between proximity zones. Sedenka [27] uses the Paillier cryptosystem as well, but incorporates garbled circuits to define three protocols to privately calculate distances between any two points using different coordinate systems on a spherical surface.

None of these encryption-based techniques provides access control They only address private proximity detection. To the best of our knowledge, only one other proposed technique addresses MPPD and access control. Li [24] proposes a MPPD solution that does include fine-grained access control by using CP-ABE [4], along with Paillier cryptosystem. Li's use of Pailler is similar to the secure distance calculations of Elmehdwi's approach, and is the most similar to our approach. We compare our framework to Li's in the experiments section.

Encryption techniques are promising and can offer good privacy guarantees for both users and DOs, but at a cost. Encryption requires computation overhead which can be a drawback for devices with limited processing power.

3 Problem Setting and Preliminaries

In this section, we define our problem setting and privacy model, then introduce the two cryptographic primitives that we use in our protocol.

3.1 Framework Model

Our location-based proximity detection model is based on users traveling through real space represented by a two-dimensional grid. At any one point in time, a user occupies exactly 1 grid cell. On demand, users can query an encrypted database of AOIs, using their own encrypted locations, to determine if their current location overlaps any AOI. The DO defines multiple AOIs, encrypts them, and supplies them to the SP. The SP determines access authorization using encrypted data, and then carries out the calculations on the encrypted data if access is authorized to determine if the user's current location is in proximity to one or more AOIs. The SP responds to the user with a list of AOIs that overlap the cell that encloses the user's location. Cells can represent any physical size. The only restriction is that the DO and all users must use the same cell granularity when converting AOI or user locations into grid coordinates.

Our protocol uses an architecture of four distinct entities. They are:

- Data Owner/Provider (DO) - Defines/encrypts AOI locations/access policies
- Key Generator (KG) - Generates CP-ABE and HVE keys and tokens.
- Service provider (SP) - Provides computation services for users to determine user proximity to any AOI
- User - user with ability to determine current location (GPS or other means)

The function of each step of the protocol is explained in a later section.

3.2 Privacy Model

Our protocol provides the following privacy guarantees:

- DOs learn nothing

– Users learn nothing except when in proximity to an AOI
– SPs learn nothing

All DOs learn nothing about user locations and users learn nothing about AOI locations (defined by a DO), except in the case when their current location overlaps one or more AOIs. Even when a user learns that he or she overlaps an AOI, the complete dimensions of the AOI are undisclosed. A user could travel around the grid in a structured attempt to learn AOI locations by remembering cells with AOI overlaps. Our protocol does not protect eventual AOI location disclosure from such an attack. SPs only learn that a user's location overlaps one or more AOIs, but do not know any actual user or AOI location, and thus cannot infer any physical location information. SPs can learn that two or more users are in proximity to one another, when those users overlap the same AOIs, but again without any AOI or user location information, cannot determine any physical location information.

Protocol extensions to protect AOI locations from deliberate user brute-force attacks are left for future work, but could be implemented at the KG by having the KG detect patterns of such attacks and responding to the user accordingly. The KG does know the location of a user when that user requests an HVE key, but the KG is trusted, and thus does not share user location with any other entity and does not have access to any AOI information.

The KG does not return the generated HVE encrypted user location to the user. Instead, the KG sends the encrypted user location to the SP. The SP only has access to encrypted AOI and user locations. It can only learn that a user's location overlaps one or more AOIs, but does not know what location the overlap represents in the grid. The SP could infer user to user proximity when multiple users are in proximity to the same AOI within a short period of time. But the SP still would not know the location of any user or AOI. Although HVE is a public key cryptography scheme and would provide the possibility for a malicious service provider to attempt to build a fake encrypted AOI list, our framework limits the distribution of the HVE public keys. This practice limits the ability to encrypt AOI locations to only trusted entities, thereby removing the ability to use dictionary type attacks to generate imposter AOI lists.

3.3 Ciphertext Policy Attribute Based Encryption

A CP-ABE scheme provides fine-grained access control over data [4]. CP-ABE associates a user with a descriptive attribute set to generate a secret key, SK. Only users whose attributes match the encryption access policy can decrypt the data. CP-ABE provides the ability for a set of users with identical attributes to share data (i.e. able to decrypt the same ciphertext) without having to share keys or have any awareness of the existence of other users with the same attributes. Further, any user's attributes may satisfy the requirements of many different ciphertext access policies. CP-ABE expressive access policies free ciphertext from being bound to a static set of attributes. Users that possess different attributes may be able to decrypt a ciphertext, as long as each user's attributes satisfies the DO supplied access policy. To encrypt a message M using CP-ABE, the encryptor

provides an access policy, expressed as a boolean expression containing selected attributes and values for M. Figure 1 shows an access policy in a tree structure. M is then encrypted based on the access structure, T. Decryptors generate SK based on their attributes. A decryptor is only able to decrypt ciphertext, CT, when her SK satisfies the access policy used to encrypt M. Unauthorized users cannot decrypt CT even if they collude and combine disjoint attributes.

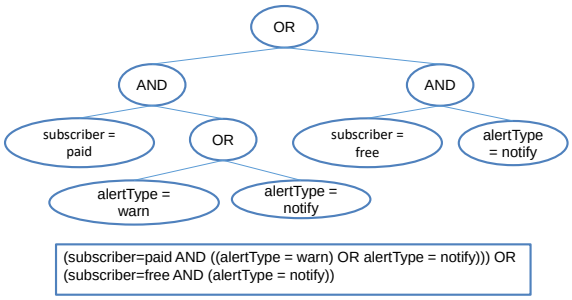

Fig. 1. CP-ABE Access Policy Tree

CP-ABE defines the following four essential functions:

1. Setup(): Input security parameter, output public key (PK), for encryption, and master key (MK), to generate user secret keys.
2. Encrypt: Input message M, access structure T, public key PK, output ciphertext CT.
3. KeyGen: Input set of user's attributes SX and MK, output secret key SK for SX.
4. Decrypt: Input CT, SK. If SK satisfies access structure in CT, return M, else return NULL.

3.4 Hidden Vector Encryption

Hidden Vector Encryption (HVE) [6] is an extension of an anonymous identity based encryption (IBE) [5] scheme. With IBE, the keys used for encryption and decryption are based on identities and attributes. HVE allows an attribute string that is associated with the ciphertext or the user secret key to contain wildcards. Thus, HVE provides a searchable encryption scheme that supports conjunctive equality, range and subset queries. An HVE attribute is represented as a vector of elements with a value of 0, 1, or a wildcard (represented as "*" and often referred to as a "don't care" value). The wildcard in an HVE attribute matches the values 0 or 1 in comparison operations. An HVE comparison of a search predicate S and a ciphertext C evaluates as True if the attribute vector I used to encrypt C contains the same values as S for all positions that are not "*".

HVE defines the following four essential functions:

1. Setup(): Input security parameter, output public key (PK), for encryption, and master key (MK).
2. KeyGen: Input MK and a string y in 0, 1, *, output secret key SK for y.
3. Encrypt: Input public key PK, message M, attribute string x in 0, 1, output ciphertext CT.
4. Query: Input CT, SK. If SK satisfies attribute string x in CT, return M, else return NULL.

4 Protocol Description

In this section, we present our proposed protocol for MPPD with access control by combining HVE and CP-ABE. We call the protocol PrivProxABE, which stands for Private Proximity detection using ABE. We first define AOI types and user attributes which will be used to enforce access control and then present the details for each of the steps of the protocol.

4.1 AOI and User Attributes

Suppose "Fun Times" defines four types of AOIs, each with a different color designator. Table 1 lists the AOI types and what each one represents.

Table 1. AOI Types

Color	**Who can access**	**Alert type**
Red	Paid subscribers	Warn
Blue	Paid subscribers	Notify
Green	Paid subscribers	Approach
Yellow	Free users	Notify

These AOI types are simply examples of what our protocol can represent. AOI types can be of any type and any number. The AOI type definition is left up to the specific implementation definition, as long as each AOI is uniquely identified with a character label. AOIs can overlap by sharing one or more cells. In such cases, users would receive a response to a proximity query indicating that their current location places them within more than one AOI. Using our example, "Fun Times" wants to provide some value to users who use their mobile app, but reserve the more detailed information for premium subscribers to their service. Users of the "InTheKnow" app that have paid for the premium service can receive "warn", "notify", and "approach" alerts. The first two alert types could be used to provide guidance for areas to avoid, while the third alert type could inform users of areas that would be beneficial to visit. Users that have not paid for the premium subscription will only receive "notify" alerts. Notice that there are two different colors for the "notify" alert. Premium subscribers will receive more specific information about "notify" alerts, while free users will only receive general messages. This approach allows "Fun Times" to define different classifications of users, each of which receives different proximity alerts.

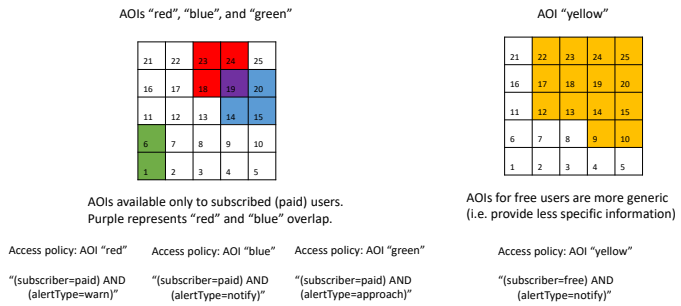

Fig. 2. Paid Subscriber and Free User AOIs

Figure 2 shows how each AOI accessible by both paid and free subscribers is defined on a grid and the access policy associated with each AOI. Our example includes four users to demonstrate the protocol's flexibility. Table 2 shows each user and their associated descriptive attributes used to generate each user's secret key. A user's attributes must satisfy (match) an AOI's access policy to decrypt and access the AOI.

Table 2. Users and Descriptive Attributes

User	**Subscriber**	**AOI Types**	**Can decrypt AOIs**
Alice	free	warn, notify	yellow
Bart	free	approach	none
Mary	paid	approach, notify	blue, green
Daniel	paid	warn, notify	red, blue

To determine proximity to an AOI, a user sends the current encrypted location to a service provider, which then determines if that user is within any cell defined as an AOI, all without ever learning any location information from the user or data provider. Users request a secret key from a key generator based on descriptive attributes. The attribute-based key provides the ability for the service provider to assess accessibility for each AOI. To continue our example, four users request proximity alerts for the amusement park defined AOIs.

We call the technique PrivProxABE (MPPD using ABE). The protocol is made up of four basic phases:

I Setup - DO Initializes protocol state
II InitAOIs - DO Encrypts AOIs with access policy
III InitUserLoc - User encrypts current user location
IV Query - User initiates location proximity query

4.2 Setup

Algorithm 1 shows the steps in the "Setup" phase and refers to figure 3. In the "Setup" phase, the DO calls the ABEsetup() method on the KG to generate

the ABE public key, PK_{ABE}. The KG sends PK_{ABE} to the DO and the SP. The DO also calls the HVEsetup() function on the KG to generate the HVE public key, PK_{HVE}, and returns PK_{HVE} to the DO. The DO keeps PK_{ABE} and PK_{HVE} secret, (i.e. the DO does not share either key with any other entity.)

Algorithm 1 PrivProxABE Protocol - Setup

1. DO calls ABEsetup() on KG. KG sends PK_{ABE} to DO and SP.
2. DO calls HVEsetup() on KG. KG sends PK_{HVE} to DO.

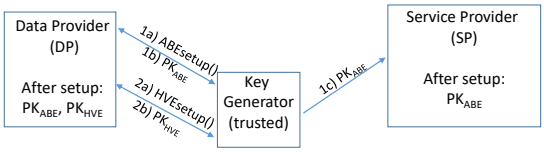

Fig. 3. PrivProxABE Setup phase

4.3 Encrypting AOIs with access policy

Algorithm 2 shows the steps in the "InitAOIs" phase and refers to Figure 4. Our evaluation assumes a static set of AOIs. If AOIs change frequently, this phase would be revisited to initialize any new or changes AOIs. Initializing individual AOIs has a small impact on performance, and would only impact overall performance in relation to the number of frequently update AOIs.

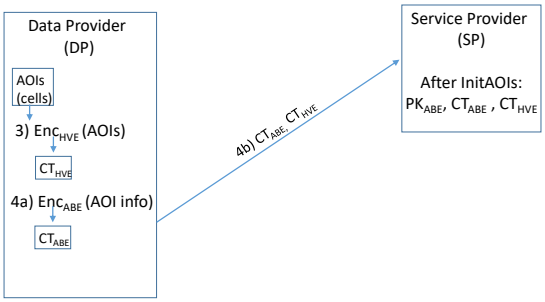

Fig. 4. PrivProxABE InitAOIs phase

Encode AOI Locations In the "InitAOIs" phase, the DO maps cell IDs to coordinates, *(x,y)*, and generates Gray codes for the x and y values. The complete Gray code for a cell ID is the concatenation of the ordered pair's *GrayCode(x)* and *GrayCode(y)*. Table 3 shows the Gray codes for the "red", "blue", "green", and "yellow" AOIs presented in Figure 2.

Table 3. AOI Location Encoding - Step 1

AOI color	Cell ID	Cell Coord	Gray code
red	18	(3,3)	0001000110
red	19	(4,3)	0011000110
red	23	(3,4)	0001000111
red	24	(4,4)	0011000111
blue	14	(4,3)	0011000010
blue	15	(5,3)	0011100010
blue	19	(4,4)	0011000110
blue	20	(5,4)	0011100110
green	1	(1,1)	0000100001
green	6	(1,2)	0000100011
yellow	9	(4,2)	0011000011
yellow	10	(5,2)	0011100011
yellow	12	(2,3)	0001100010
yellow	13	(3,3)	0001000010
yellow	14	(4,3)	0011000010
yellow	15	(5,3)	0011100010
yellow	17	(2,4)	0001100110
yellow	18	(3,4)	0001000110
yellow	19	(4,4)	0011000110
yellow	20	(5,4)	0011100110
yellow	22	(2,5)	0001100111
yellow	23	(3,5)	0001000111
yellow	24	(4,5)	0011000111
yellow	25	(5,5)	0011100111

We use Gray encoding to reduce the number of stored values to represent AOIs. In many cases, using Gray codes dramatically reduces the number of distinct values necessary to represent groups of cells that comprise AOIs. To reduce the number of required comparisons to determine AOI proximity in user queries, we compress the Gray codes into search tokens that contain wildcards, as proposed in [15] . If two Gray code values differ by only a single digit, we replace the digit with a wildcard, "*", allowing the new single search token to represent two individual Gray codes, or Cell IDs. The iterative compression process continues until no two remaining search tokens differ by a single digit, potentially resulting in a small number of search tokens for each AOI.

For example, the Gray codes for cell 18 (0001000110) and cell 19 (0011000110) only differ by the value in position 2. Therefore, we can combine the two Gray codes into a single token, replacing the value in position 2 with a wildcard "*", (00*1000110). The wildcard represents a "don't care" value, since the token matches any value in position 2. The Gray codes for cell 23 and 24 also differ by only a single digit and can be represented with the token (00*1000111). Notice that the two resulting tokens differ by only a single digit and can also be combined into a single token (00*100011*). In this way, a single token can represent four cells. Table 4 shows the result of the compression process for the "red", "blue", "green", and "yellow" AOIs presented in Figure 2.

Encrypt Encoded AOIs After compressing each of the Gray encoded AOI locations, each AOI is represented by one or more compressed tokens. The DO encrypts each compressed token with HVE using PK_{HVE}, returning CT_{HVE}. The DO then encrypts the AOI label and any additional AOI information for a single AOI with CP-ABE using PK_{ABE}, returning CT_{ABE}. The DO then sends CT_{ABE} and CT_{HVE} for all AOIs to the SP.

Table 4. AOI Location Encoding - Step 2

AOI color	**Compressed Gray code token(s)**
red	00*100011*
blue	0011*00*10
green	00001000*1
yellow	0011*00*1*, 0001*00*10, 0001*00111

Algorithm 2 PrivProxABE Protocol - InitAOIs

3. DO encrypts encoded AOI cells (encoded using Gray encoding) and keeps CT_{HVE}.
4a. DO encrypts AOI label and other descriptive AOI info using access policy to get CT_{ABE}.
4b. DO sends CT_{ABE} and CT_{HVE} to SP.

4.4 Encrypting user location

Algorithm 3 shows the steps in the "InitUserLoc" phase and refers to Figure 5. If the user is new (eg. not a user known to the KG), the KG authenticates the user and records the authorized user attributes in the KG user table. The KG uses the user attributes to generate a secret key (SK_{ABE}), and returns SK_{ABE} to the user. The user then calls HVEencrypt(currentLocation) on the KG to encrypt the user's current location using HVE, generating Loc_{HVE}. The KG returns Loc_{HVE} to the user. The user then generates a randomIdentifier for use in the query phase. The randomIdentifier detaches queries from user identities.

Algorithm 3 PrivProxABE Protocol - InitUserLoc

5. User calls ABEkeyGen(userID) on KG. The KG authenticates new users and generates a CP-ABE secret key (SK_{ABE}) based on the user's attributes, and returns SK_{ABE} to the user.
6. User calls HVEencrypt(currentLocation) on KG. KG sends the encrypted location (Loc_{HVE}) to the user. The user generates a randomIdentifier.

4.5 Querying proximity to AOIs

Algorithm 4 shows the steps in the "Query" phase and refers to Figure 6. The HVEdecrypt(Loc_{HVE}) function, run on the SP, iterates through each AOI and attempts to decrypt CT_{HVE}. If HVE decryption is successful, that means that the user's location shares 1 cell defined by the AOI. The SP returns a list of all successfully decrypted HVE ciphertexts to the user. The user runs ABEdecrypt(SK_{ABE},CT_{ABE}) to attempt to decrypt each AOI returned from the SP. Successful ABE decryption means the user's attributes satisfy the AOI's access policy for an AOI that overlaps the current user's location. The service provider returns a list of encrypted AOIs that overlap the user's location. The user then attempts to decrypt each AOI using CP-ABE. Any successfully decrypted AOI means that the user's location overlaps the AOI and the user is authorized to consume the AOI's information.

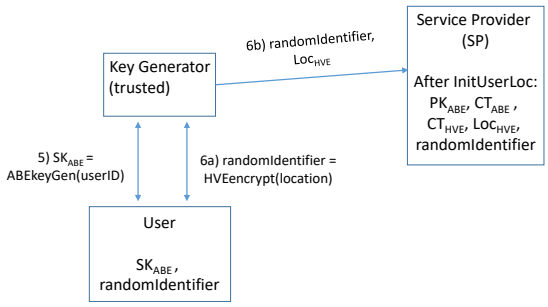

Fig. 5. PrivProxABE InitUserLoc phase

Algorithm 4 PrivProxABE Protocol - Query

7. User calls HVEdecrypt() on SP. SP attempts to decrypt all AOIs using user's Loc_{HVE}.
8. SP returns a list of all successfully decrypted (HVE) AOIs.
9. User calls ABEdecrypt(SK_{ABE},CT_{ABE}) for each AOI returned from the SP to determine if the user is authorized to consume proximity alerts for each AOI.

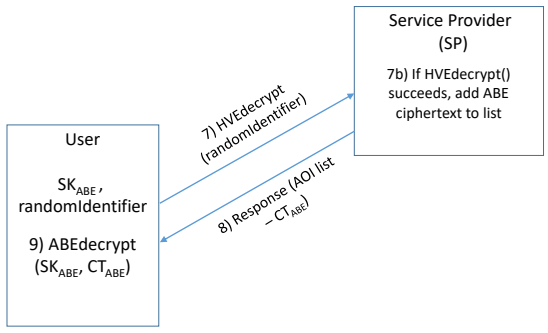

Fig. 6. PrivProxABE Query phase

5 Security and Privacy

In this section, we analyze the security and privacy guarantees of our proposed protocol. One of the primary requirements of this protocol is to maintain the privacy of all actors. The proposed protocol relies on the security guarantees of CP-ABE and HVE to provide the overall security of our approach. In each scheme, the SK is necessary to decrypt data. Our protocol ensures that only trusted entities, users and the KG, have access to any SK. The following list explains the privacy guarantees with respect to each architectural entity.

- Data Owner/Provider (DO) - The DO is the only entity with access to the unencrypted AOI data. It does not share unencrypted AOI locations with any other entity to protect AOI locations. The DO does not have access to any other information generated or stored by the KG or any user, and thus cannot learn any information about user locations.
- Key Generator (KG) - The KG (trusted entity) generates SKs, but never has access to AOIs and cannot decrypt any AOI data without colluding with another entity. The KG does learn user locations as it generates Loc_{HVE} for a user's current location, but never shares this data with any entity other than the user and does not have access to any information, such as AOI locations, with which to correlate user locations. The KG could remember user locations to build user trajectories, but our protocol defines the KG as a trusted entity and we expect that the KG will not exceed its authority.
- Service provider (SP) - The SP never has access to unencrypted AOI or user locations, and cannot access the decryption keys to decrypt any ciphertext. The SP never learns any information about AOI or user locations. The SP does learn that a user is in proximity to one or more AOIs when the locations overlap, but does not have any information to imply actual user/AOI locations. The SP can determine when users are close to one another when they are in proximity to the same AOI(s). The service provider cannot attempt a dictionary type attack by building a fake AOI list since it lacks PK_{HVE} which is necessary to encrypt AOI locations.
- User - The user only divulges actual location to the KG. Since the SP carries out all calculations on encrypted data, the SP never learns the user's location, and as stated previously, the DO never has access to any user location information. The user never has access to any AOI location and only learns general AOI information when the SP discloses that the user is in proximity to one or more AOIs. The user learns that the current location overlaps the reported AOIs, but does not immediately learn the dimensions of any AOI. Through a process of strategically visiting multiple cells, a user could build a map of AOI locations based on proximity alerts. However, a user could only construct meaningful map entries from AOI data the user is authorized to decrypt. One method to restrict malicious users from building even a partial AOI map in a short period of time could be to set a threshold of maximum speed at which a user could realistically travel from cell to cell. If a user attempts to exceed this threshold, he would be deemed malicious. Hallgren, et. al. [16] propose a protocol, MaxPace, based on the concept of using travel

speed thresholds to detect malicious users. Implementing protection from this type of attack is left for future work.

Our protocol protects both AOI and user locations from disclosure, and only allows users to eventually build an AOI map after determined actions resulting in recording history of cells visited and proximity alerts.

6 Experiments

In this section, we evaluate our protocol in terms of computational performance. We do not evaluate communication overhead in this analysis, as it is expected that communication overhead is similar between the approaches we used in out evaluation. In future work we will expand the evaluation to measure communication overhead. We implemented a prototype of our framework and Li's [24] similar framework and ran multiple tests to evaluate performance as input data size varied. We ran each test by first defining access policies, half of which are satisfied by test user attributes. We then define a set of AOIs, each with one of the test access policies. The test user then issues location proximity queries, each with a random user location. The times reported in the results represent the average time to resolve a single location proximity query. All tests were run on a single computer with 16GB memory and an Intel Core i7 2.4GHz CPU running Ubuntu Linux 16.04. We chose to run all tests on a single computer to focus on computation load. Tests were run for varying numbers of AOIs, ranging from 10 to 1500, using grid sizes from 100x100, up to 1500x1500, and various AOI shapes. Figure 7 shows the 11 different AOI shapes used in the performance evaluation. We defined 9 basic shapes, 1 irregular shape, and a standard symmetric AOI represented as a square when using grid cells and as a circle for circle-based AOIs. For Li's Paillier based implementation, we constructed each AOI using multiple circles to approximate the shapes built from grid cells. The only AOI type that can be generally approximated with a single circle is the square AOI. All other AOI shape definitions require multiple circles, which results in more computation.

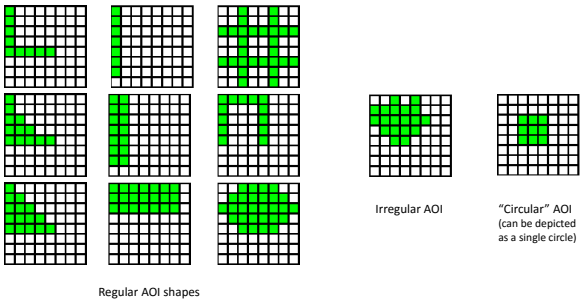

Fig. 7. AOI shapes used in tests

Both frameworks were implemented in Python 2.7, using the Charm [2] framework for rapid cryptosystem prototyping. The primary purpose of our prototype implementations is to demonstrate the viability and advantages of our proposed protocol and to compare its relative performance to similar proposed protocol that is less scalable, as opposed to providing a deployment-ready solution. Our experiments showed that varying the grid sizes only had a minor affect on performance - much less than varying the number of AOIs. For this reason, we chose to present results of tests all run using a grid size of 250x250. The unit size can represent any physical distance. In this way, the association of a physical measurement with the unit size defines the precision of the approach. That is, a smaller unit size results in higher precision.

Figure 8 shows the performance time of our scheme compared against Paillier-based protocol [24] with varying number of AOIs of different shapes. Our tests show that the choice of HVE for AOI encoding and encryption provides superior scalability over Paillier cryptosystem distance calculations. Li's CP-ABE/Paillier approach works efficiently for circular AOIs, but lacks scalability for more complex AOI shapes. In our evaluations, we represented Li's circular AOIs as square regions. Although squares only approximate circular regions, they provide the most compact structure when using Gray code compression. Figure 8(a) shows that CP-ABE/Paillier performs better than our CP-ABE/HVE framework when all AOIs are represented as circles only. This is because our HVE approach requires a collection of cells to define each AOI. Although using Gray codes with token compression to combine multiple tokens, the Paillier distance calculation for a single circle is still more efficient than an HVE token match.

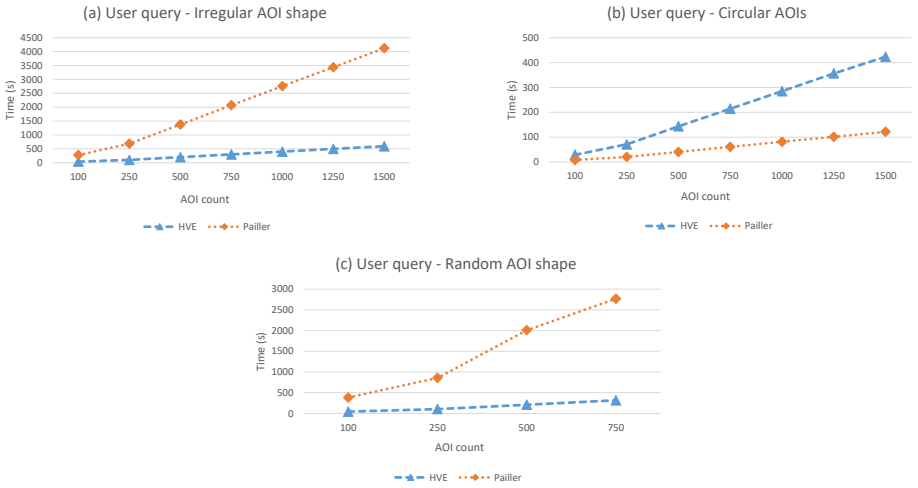

Fig. 8. Experimental Results

However, when AOI shapes are not circular, as is desirable to represent distinct regions in the real world of various shapes, the CP-ABE/Paillier approach which requires multiple circles to represent a single AOI exhibits slower performance due to the additional overhead. Our approach can efficiently represent irregular AOI shapes and still provide good performance. Figure 8(b) shows the performance when using AOIs with irregular shapes. The last set of test we ran randomly selected AOI shapes from 10 pre-defined shapes, including circular AOIs. Even when selecting some regular shapes, our approach using CP-ABE and HVE performs better than the P-ABE/Paillier approach. Figure 8(c) shows the performance for randomly shaped AOIs.

7 Conclusion

We propose a novel framework and protocol based on CP-ABE and HVE that provides MPPD with embedded access control for different classifications of users. With our framework, DOs can define and maintain a single set of AOIs and grant access to AOI information based on user attributes. We implemented our framework and protocol, along with another approach that uses CP-ABE and Paillier cryptosystem, and showed that our approach is more scalable when using AOIs defined as non-circular shapes. Our framework provides a basis for implementers to develop scalable MPPD services that minimizes workload on DOs or users. This approach can provide flexible MPPD services to meet a wide variety of client needs, without requiring a trusted third party to examine user locations to determine proximity.

Future work toward refining our MPPD with access control solution include hardening the protocol against attack, optimizing the HVE token compression algorithm, and reducing the communication overhead between framework components. Each of these enhancements will make our protocol more secure and scalable, and easier to deploy to mobile devices with limited resources.

Bibliography

[1] Agrawal, R., Kiernan, J., Srikant, R., Xu, Y.: Order preserving encryption for numeric data. In: Proceedings of the 2004 ACM SIGMOD International Conference on Management of data. pp. 563–574. ACM (2004)

[2] Akinyele, J.A., Garman, C., Miers, I., Pagano, M.W., Rushanan, M., Green, M., Rubin, A.D.: Charm: a framework for rapidly prototyping cryptosystems. Journal of Cryptographic Engineering 3(2), 111–128 (2013)

[3] Andrés, M.E., Bordenabe, N.E., Chatzikokolakis, K., Palamidessi, C.: Geo-indistinguishability: Differential privacy for location-based systems. In: Proceedings of the 2013 ACM SIGSAC Conference on Computer & communications security. pp. 901–914. ACM (2013)

[4] Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: 2007 IEEE symposium on security and privacy (SP'07). pp. 321–334. IEEE (2007)

[5] Boneh, D., Di Crescenzo, G., Ostrovsky, R., Persiano, G.: Public key encryption with keyword search. In: International Conference on the Theory and Applications of Cryptographic Techniques. pp. 506–522. Springer (2004)

[6] Boneh, D., Waters, B.: Conjunctive, subset, and range queries on encrypted data. In: Theory of Cryptography Conference. pp. 535–554. Springer (2007)

[7] Bugiel, S., Heuser, S., Sadeghi, A.R.: Flexible and fine-grained mandatory access control on android for diverse security and privacy policies. In: Usenix security. pp. 131–146 (2013)

[8] Calderoni, L., Palmieri, P., Maio, D.: Location privacy without mutual trust: The spatial bloom filter. Computer Communications 68, 4–16 (2015)

[9] Choi, S., Ghinita, G., Bertino, E.: Secure mutual proximity zone enclosure evaluation. In: Proceedings of the 22Nd ACM SIGSPATIAL Intl Conf on Advances in Geographic Information Systems. pp. 133–142 (2014)

[10] Chor, B., Kushilevitz, E., Goldreich, O., Sudan, M.: Private information retrieval. Journal of the ACM (JACM) 45(6), 965–981 (1998)

[11] Dwork, C.: Differential privacy: A survey of results. In: Intl Conf on Theory and Applications of Models of Computation. pp. 1–19. Springer (2008)

[12] Elmehdwi, Y., Samanthula, B.K., Jiang, W.: Secure k-nearest neighbor query over encrypted data in outsourced environments. In: 2014 IEEE 30th International Conference on Data Engineering. pp. 664–675. IEEE (2014)

[13] Fawaz, K., Shin, K.G.: Location privacy protection for smartphone users. In: Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security. pp. 239–250. ACM (2014)

[14] Ghinita, G., Kalnis, P., Khoshgozaran, A., Shahabi, C., Tan, K.L.: Private queries in location based services: anonymizers are not necessary. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of data. pp. 121–132. ACM (2008)

[15] Ghinita, G., Rughinis, R.: A privacy-preserving location-based alert system. In: Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems. pp. 432–435. ACM (2013)
[16] Hallgren, P., Ochoa, M., Sabelfeld, A.: Maxpace: Speed-constrained location queries. In: Communications and Network Security (CNS), 2016 IEEE Conference on. pp. 136–144. IEEE (2016)
[17] Hengartner, U.: Hiding location information from location-based services. In: Mobile Data Management, 2007 Intl Conf on. pp. 268–272. IEEE (2007)
[18] Hossain, A.A., Lee, S.J., Huh, E.N.: Shear-based spatial transformation to protect proximity attack in outsourced database. In: Trust, Security and Privacy in Computing and Communications (TrustCom), 2013 12th IEEE International Conference on. pp. 1633–1638. IEEE (2013)
[19] Khoshgozaran, A., Shahabi, C.: Private buddy search: Enabling private spatial queries in social networks. In: Computational Science and Engineering, 2009. CSE'09. International Conference on. vol. 4, pp. 166–173. IEEE (2009)
[20] Khoshgozaran, A., Shahabi, C.: Private information retrieval techniques for enabling location privacy in location-based services. In: Privacy in Location-Based Applications, pp. 59–83. Springer (2009)
[21] Kim, H.I., Chang, J.W.: k-nearest neighbor query processing algorithms for a query region in road networks. Journal of Computer Science and Technology 28(4), 585–596 (2013)
[22] Kim, H.I., Hong, S., Chang, J.W.: Hilbert curve-based cryptographic transformation scheme for spatial query processing on outsourced private data. Data & Knowledge Engineering (2015)
[23] Kim, H.I., Kim, Y.K., Chang, J.W.: A grid-based cloaking area creation scheme for continuous lbs queries in distributed systems. Journal of Convergence 4(1), 23–30 (2013)
[24] Li, X.Y., Jung, T.: Search me if you can: privacy-preserving location query service. In: INFOCOM, 2013 Proceedings IEEE. pp. 2760–2768 (2013)
[25] Lu, R., Lin, X., Shen, X.: Spoc: A secure and privacy-preserving opportunistic computing framework for mobile-healthcare emergency. Parallel and Distributed Systems, IEEE Transactions on 24(3), 614–624 (2013)
[26] Palmieri, P., Calderoni, L., Maio, D.: Spatial bloom filters: enabling privacy in location-aware applications. In: International Conference on Information Security and Cryptology. pp. 16–36. Springer (2014)
[27] Šeděnka, J., Gasti, P.: Privacy-preserving distance computation and proximity testing on earth, done right. In: Proceedings of the 9th ACM Symposium on Info, Computer and Comm Security. pp. 99–110. ASIA CCS '14 (2014)
[28] Wang, B., Li, M., Wang, H.: Geometric range search on encrypted spatial data. IEEE Trans on Info Forensics and Security 11(4), 704–719 (2016)
[29] Xiao, Y., Xiong, L.: Protecting locations with differential privacy under temporal correlations. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security. pp. 1298–1309. ACM (2015)
[30] Yiu, M.L., Ghinita, G., Jensen, C.S., Kalnis, P.: Enabling search services on outsourced private spatial data. The VLDB Journal 19(3), 363–384 (2010)