2012 IEEE 28th International Conference on Data Engineering

Privacy-Preserving and Content-Protecting Location Based Queries

Russell Paulet*, Md. Golam Kaosar*, Xun Yi*, Elisa Bertino†
*School of Engineering and Science, Victoria University, Melbourne, VIC 8001, Australia
†Department of Computer Sciences, Purdue University, West Lafayette, IN 47907, USA

***Abstract*—In this paper we present a solution to one of the location-based query problems. This problem is defined as follows: (i) a user wants to query a database of location data, known as Points Of Interest (POI), and does not want to reveal his/her location to the server due to privacy concerns; (ii) the owner of the location data, that is, the location server, does not want to simply distribute its data to all users. The location server desires to have some control over its data, since the data is its asset. Previous solutions have used a trusted anonymiser to address privacy, but introduced the impracticality of trusting a third party. More recent solutions have used homomorphic encryption to remove this weakness. Briefly, the user submits his/her encrypted coordinates to the server and the server would determine the user's location homomorphically, and then the user would acquire the corresponding record using Private Information Retrieval techniques. We propose a major enhancement upon this result by introducing a similar two stage approach, where the homomorphic comparison step is replaced with Oblivious Transfer to achieve a more secure solution for both parties. The solution we present is efficient and practical in many scenarios. We also include the results of a working prototype to illustrate the efficiency of our protocol.**

I. Introduction

A location based service (LBS) is an information, entertainment and utility service generally accessible by mobile devices such as, mobile phones, GPS devices, pocket PCs, and operates through a mobile network. A LBS can offer many services to the users based on the geographical position of their mobile device. The services provided by a LBS are typically based on a point of interest database. By retrieving the Points Of Interest (POIs) from the database server, the user can get answers to various location based queries, which include but are not limited to - discovering the nearest ATM machine, gas station, hospital, or police station. In recent years there has been a dramatic increase in the number of mobile devices querying location servers for information about POIs. Among many challenging barriers to the wide deployment of such application, privacy assurance is a major issue. For instance, users may feel reluctant to disclose their locations to the LBS, because it may be possible for a location server to learn who is making a certain query by linking these locations with a residential phone book database, since users are likely to perform many queries from home.

The Location Server (LS), which offers some LBS, spends its resources to compile information about various interesting POIs. Hence, it is expected that the LS would not disclose any information without fees. Therefore the LBS has to ensure that LS's data is not accessed by any unauthorized user. During the process of transmission the users should not be allowed to discover any information for which they have not paid. It is thus crucial that solutions be devised that address the privacy of the users issuing queries, but also prevent users from accessing content to which they do not have authorization.

A. Related Work

The first solution to the problem was proposed by Beresford [3], in which the privacy of the user is maintained by constantly changing the user's name or pseudonym within some mix-zone. It can be shown that, due to the nature of the data being exchanged between the user and the server, the frequent changing of the user's name provides little protection for the user's privacy. A more recent investigation of the mix-zone approach has been applied to road networks [25]. They investigated the required number of users to satisfy the unlinkability property when there are repeated queries over an interval. This requires careful control of how many users are contained within the mix-zone, which is difficult to achieve in practice.

A complementary technique to the mix-zone approach is based on k-anonymity [14], [8], [4]. The concept of k-anonymity was introduced as a method for preserving privacy when releasing sensitive records [28]. This is achieved by generalisation and suppression algorithms to ensure that a record could not be distinguished from $(k-1)$ other records. The solutions for LBS use a trusted anonymiser to provide anonymity for the location data, such that the location data of a user cannot be distinguished from $(k-1)$ other users.

An enhanced trusted anonymiser approach has also been proposed, which allows the users to set their level of privacy based on the value of k [22], [21]. This means that, given the overhead of the anonymiser, a small value of k could be used to increase the efficiency. Conversely, a large value of k could be chosen to improve the privacy, if the users felt that their position data could be used maliciously.

Methods have also been proposed to confuse and distort the location data, which include path and position confusion. Path confusion was presented by Hoh and Gruteser [16]. The basic idea is to add uncertainty to the location data of the users at the points the paths of the users cross, making it hard to trace users based on raw location data that was k-anonymised. Position confusion has also been proposed as an approach to provide privacy [22], [17]. The idea is for the

1084-4627/12 $26.00 © 2012 IEEE
DOI 10.1109/ICDE.2012.95

trusted anonymiser to group the users according to a cloaking region (CR), thus making it harder for the LS to identify an individual.

As solutions based on the use of a central anonymiser are not practical, Hashem and Kulik presented a scheme whereby a group of trusted users construct an ad-hoc network and the task of querying the LS is delegated to a single user [15]. This idea improves on the previous work by the fact that there is no single point of failure. If a user that is querying the LS suddenly goes offline, then another candidate can be easily found. However, generating a trusted ad-hoc network in a real world scenario is not always possible.

Another method for avoiding the use of a trusted anonymiser is to use 'dummy' locations [18], [6]. The basic idea is to confuse the location of the user by sending many random other locations to the server, such that the server cannot distinguish the actual location from the fake locations. This incurs both processing and communication overhead for the user device. The user has to randomly choose a set of fake locations as well as transmitting them over a network, wasting bandwidth. We refer the interested reader to Krumm [19], for a more detailed survey in this area.

Most of the previously discussed issues are solved with the introduction of a private information retrieval (PIR) location based server scheme [12]. Their basic idea is to employ PIR to enable the user to query the location database without compromising the privacy of the query. Generally speaking, PIR schemes allow a user to retrieve data (bit or block) from a database, without disclosing the index to the database server [5]. Ghinita et al. used a variant of PIR which is based on the quadratic residuosity problem [20]. Basically the quadratic residuosity problem states that is computationally hard to determine whether a number is a quadratic residue of some composite modulus n ($x^2 = q \ (mod \ n)$), where the factorisation of n is unknown.

This idea was extended to provide database protection [10], [11]. This protocol consists of two stages. In the first stage, the user and server use homomorphic encryption to allow the user to privately determine whether his/her location is contained within a cell, without disclosing his/her coordinates to the server. In the second stage, PIR is used to retrieve the data contained within the appropriate cell.

The homomorphic encryption scheme used to privately compare two integers is the Paillier encryption scheme [24]. The Paillier encryption scheme is known to be additively homomorphic and multiplicatively-by-a-constant homomorphic. This means that we can add or scale numbers even when all numbers are encrypted. Both features are used to determine the sign (most significant bit) of $(a - b)$, and hence the user is able to determine the cell that he/she is located, without disclosing their location.

B. Our Contributions

In this paper, we propose a novel protocol for location based queries that has major performance improvements with respect to the approach by Ghinita at el. [10] and [11]. Like such protocol, our protocol is organized according to two stages. In the first stage, the user privately determines his/her location within a public grid, using oblivious transfer. This data contains both the ID and associated symmetric key for the block of data in the private grid. In the second stage, the user executes a communicational efficient PIR [9], to retrieve the appropriate block in the private grid. This block is decrypted using the symmetric key obtained in the previous stage. Our protocol thus provides protection for both the user and the server. The user is protected because the server is unable to determine his/her location. Similarly, the server's data is protected since a malicious user can only decrypt the block of data obtained by PIR with the encryption key acquired in the previous stage. In other words, users cannot gain any more data than what they have paid for. We also provide results from a working prototype showing the efficiency of our approach.

C. Paper Organisation

The rest of the paper is organised as follows. Section II presents the protocol model and other preliminaries. Section III presents and describes our proposed protocol. Section IV analyses the security of the protocol. Section V analyses the performance and efficiency of the protocol. Section VI reports the performance results of the working prototype and discusses feasibility. Section VII summarises the key contributions of this paper and future directions.

II. Protocol Model

Before describing our protocol we introduce the system model, which defines the major entities and their roles. The description of the protocol model begins with the notations and system parameters of our solution.

A. Preliminaries

Let $x \leftarrow y$ be the assignment of the value of variable y to variable x and $E \Leftarrow v$ be the transfer of the variable v to entity E. Denote the ElGamal [7] encryption of message m as $E(m) = \boldsymbol{A} = (A_1, A_2) = (g^r, g^m y^r)$, where g is a generator of group G, y is the public key of the form $y = g^x$, and r is chosen at random. Note that $\boldsymbol{A}$ is a vector, while A_1, A_2 are elements of the vector. The cyclic group G is a multiplicative subgroup of the finite field F_p, where p is a large prime number and q is a prime that divides $(p-1)$. Let g be a generator of group G, with order q and $\langle |g| \rangle$ denote the order of generator g. We denote by $|p|$ the bit length of p, $a||b$ the concatenation of a and b, and $\oplus$ the exclusive OR operator.

We require, for security reasons, that $|p| = 1024$ and $|q| = 160$. We also require that the parameters G, g, p, q be fixed for the duration of a round of our protocol and be made publicly accessible to every entity in our protocol.

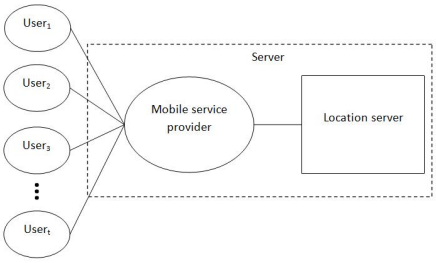

Fig. 1. System model

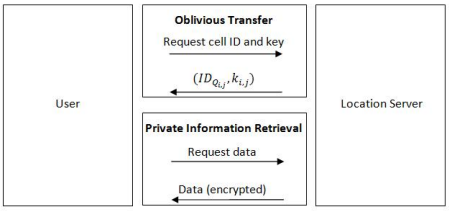

Fig. 2. High level overview of the protocol

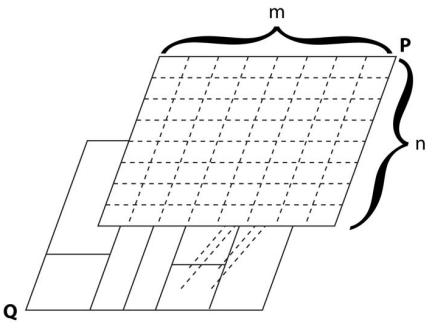

Fig. 3. The public grid superimposed over the private grid

B. System Model

The system model consists of three types of entities (see Figure 1): the set of users[1] who wish to access location data U, a mobile service provider SP, and a location server LS. From the point of view of a user, the SP and LS will compose a server, which will serve both functions. The user does not need to be concerned with the specifics of the communication.

The users in our model use some location-based service provided by the location server LS. For example, what is the nearest ATM or restaurant? The purpose of the mobile service provider SP is to establish and maintain the communication between the location server and the user. The location server LS owns a set of POI records r_i for $1 \leq r_i \leq \rho$. Each record describes a POI, giving GPS coordinates to its location (x_{gps}, y_{gps}), and a description or name about what is at the location.

We assume that the mobile service provider SP does not interfere with the communications between the user and the location server. This means that the mobile service provider does not collude with the location server to attack the privacy of the user. As a consequence of this assumption, the user is able to either use GPS (Global Positioning System) or the mobile service provider to acquire his/her coordinates.

Since we are assuming that the mobile service provider SP is trusted to maintain the connection, we consider only two possible adversaries. One for each communication direction. We consider the case in which the user is the adversary and tries to obtain more than he/she is allowed. Next we consider the case in which the location server LS is the adversary, and tries to uniquely associate a user with a grid coordinate.

III. Protocol Description

We now describe our protocol. We first give a protocol summary to contextualise the proposed solution and then describe the solution's protocol in more detail.

A. Protocol Summary

The ultimate goal of our protocol is to obtain a set (block) of POI records from the LS, which are close to the user's position, without compromising the privacy of the user or the server. We achieve this by applying a two stage approach, which is shown in Figure 2. The first stage is based on a two-dimensional oblivious transfer [23] and the second stage is based on a communicationally efficient PIR [9]. The oblivious transfer based protocol is used by the user to obtain the cell ID, where the user is located, and the corresponding symmetric key. The knowledge of the cell ID and the symmetric key is then used in the PIR based protocol to obtain and decrypt the location data.

The user determines his/her location within a publicly generated grid P by using his/her GPS coordinates and forms an oblivious transfer query[2]. The minimum dimensions of the public grid are defined by the server and are made available to all users of the system. This public grid superimposes over the privately partitioned grid generated by the location server's POI records, such that there is at least one $P_{i,j}$ cell within the server's partition $Q_{i,j}$. This is illustrated in Figure 3.

Since PIR does not require that a user is constrained to obtain only one bit/block, the location server needs to implement some protection for its records. This is achieved by encrypting each record in the POI database with a key using a symmetric key algorithm, where the key for encryption is the same key used for decryption. This key is augmented with the cell info data retrieved by the oblivious transfer query. Hence, even if the user uses PIR to obtain more than one record, the data will be meaningless resulting in improved security for the server's database. Before we describe the protocol in detail, we describe some initialisation performed by both parties.

[1]In this paper we use the term "user" to refer to the entity issuing queries and retrieving query results. In most cases, such user is a client software executing on behalf of a human user.

[2]An oblivious transfer query is where a server cannot learn the user's query, while the user cannot gain more than they are entitled. This is similar to PIR, but oblivious transfer requires protection for the user and server. PIR only requires that the user is protected.

B. Global Initialisation

A user u from the set of users U initiates the protocol process by deciding a suitable square cloaking region CR, which contains his/her location. All user queries will be with respect to this cloaking region. The user also decides on the accuracy of this cloaking region by how many cells are contained within it, which is at least the minimum size defined by the server. This information is combined to form the public grid P and submitted to the location server, which partitions its records or superimposes it over pre-partitioned records (see Figure 3). This partition is denoted Q (note that the cells don't necessarily need to be the same size as the cells of P). Each cell in the partition Q must have the same number r_{max} of POI records. Any variation in this number could lead to the server identifying the user. If this constraint cannot be satisfied, then dummy records can be used to make sure each cell has the same amount of data. We assume that the LS does not populate the private grid with misleading or incorrect data, since such action would result in the loss of business under a payment model.

Next, the server encrypts each record r_i within each cell of Q, $Q_{i,j}$, with an associated symmetric key $k_{i,j}$. The encryption keys are stored in a small (virtual) database table that associates each cell in the public grid P, $P_{i,j}$, with both a cell in the private grid $Q_{i,j}$ and corresponding symmetric key $k_{i,j}$. This is shown by Figure 4.

The server then processes the encrypted records within each cell $Q_{i,j}$ such that the user can use an efficient PIR [9], to query the records. Using the private partition Q, the server represents each associated (encrypted) data as an integer C_i, with respect to the cloaking region. For each C_i, the server chooses a set of unique prime powers $\pi_i = p_i^{c_i}$, such that $C_i < \pi_i$. We note that the c_i in the exponent must be small for the protocol to work efficiently. We also stipulate that the unique prime powers π_i follow a predictable pattern. Finally, the server uses the Chinese Remainder Theorem to find the smallest integer e such that $e = C_i \ (mod\ \pi_i)$ for all C_i. The integer e effectively represents the database. Once the initialisation is complete, the user can proceed to query the location server for POI records.

C. Oblivious Transfer Based Protocol

The purpose of this protocol is for the user to obtain one and only one record from the cell in the public grid P, shown in Figure 4. We achieve this by constructing a 2-dimensional oblivious transfer, based on the ElGamal oblivious transfer [2], using adaptive oblivious transfer[3] proposed by Naor et al. [23].

The public grid P, known by both parties, has m columns and n rows. Each cell in P contains a symmetric key $k_{i,j}$ and a cell id in grid Q i.e., $(ID_{Q_{i,j}}, k_{i,j})$, which can be represented by a stream of bits $X_{i,j}$. The user determines his/her i, j coordinates in the public grid which is used to acquire the data from the cell within the grid. The protocol is initialised by the server by generating $m \times n$ keys of the form $g^{R_i}||g^{C_i}$. This initialisation is presented in Algorithm 1.

[3]Appendix A presents a simple example of adaptive oblivious transfer.

Algorithm 1 $Initialisation$

Input: $X_{1,1}, ..., X_{m,n}$, where $X_{i,j} = ID_{Q_{i,j}}||k_{i,j}$
Output: $Y_{1,1}, ..., Y_{m,n}$

1: $K_{i,j} \leftarrow K_{i,j} = g^{R_i}||g^{C_j}$, for $1 \leq i \leq n$ and $1 \leq j \leq m$, where R_i and C_j are randomly chosen
2: $Y_{i,j} \leftarrow X_{i,j} \oplus H(K_{i,j})$, for $1 \leq i \leq n$ and $1 \leq j \leq m$, where H is a fast secure hash function
3: **return** $Y_{1,1}, ..., Y_{m,n}$ {Encryptions of $X_{1,1}, ..., X_{m,n}$ using $K_{i,j}$}

Algorithm 1 is executed once and the output $Y_{1,1}, ..., Y_{m,n}$ is sent to the user. At which point, the user can query this information using the indices i, and j, as input. This protocol is presented in Algorithm 2.

Algorithm 2 $Transfer$

Input: User:i, j
Output: User:$(ID_{Q_{i,j}}, k_{i,j})$

1: **User**
2: $y \leftarrow g^x$, where y is the public key of the user and x is chosen at random
3: $\boldsymbol{C_1} \leftarrow (A_1, B_1) = (g^{r_1}, g^{-i}y^{r_1})$
4: $\boldsymbol{C_2} \leftarrow (A_2, B_2) = (g^{r_2}, g^{-j}y^{r_2})$
5: $Server \Leftarrow \boldsymbol{C_1}, \boldsymbol{C_2}$
6: **Server**
7: $\boldsymbol{C'_{1,\alpha}} \leftarrow (A_1^{r'_\alpha}, g^{R_\alpha}(g^\alpha B_1)^{r'_\alpha})$ for $1 \leq \alpha \leq n$
8: $\boldsymbol{C'_{2,\beta}} \leftarrow (A_2^{r'_\beta}, g^{C_\beta}(g^\beta B_2)^{r'_\beta})$ for $1 \leq \beta \leq m$
9: $User \Leftarrow \boldsymbol{C'_{1,1}}, ..., \boldsymbol{C'_{1,n}}, \boldsymbol{C'_{2,1}}, ..., \boldsymbol{C'_{2,m}}$
10: **User**
11: Let $(U_{1,i}, V_{1,i}) = \boldsymbol{C'_{1,i}}$ and $(U_{2,j}, V_{2,j}) = \boldsymbol{C'_{1,j}}$
12: $W_1 \leftarrow V_{1,i}/(U_{1,i})^x$
13: $W_2 \leftarrow V_{2,j}/(U_{2,j})^x$
14: $K'_{i,j} \leftarrow W_1||W_2$
15: $X'_{i,j} \leftarrow Y_{i,j} \oplus H(K'_{i,j})$
16: Reconstruct $(ID_{Q_{i,j}}, k_{i,j})$ from $X'_{i,j}$
17: **return** $(ID_{Q_{i,j}}, k_{i,j})$ {Cell id of grid Q, with associated cell key}

At the conclusion of the protocol presented by Algorithm 2, the user has the information to query the location server for the associated block.

Theorem (Correctness) 1. *Assume that the user and server follow Algorithms 1 and 2 correctly, then* $X'_{i,j} = Y_{i,j} \oplus H(K_{i,j})$.

*P*roof: We begin this proof by showing that $K_{i,j} = K'_{i,j}$. In the initialisation Algorithm 1 $K_{i,j}$ is calculated as $K_{i,j} = g^{R_i}||g^{C_j}$. At the end of the transfer protocol, the user computes $K'_{i,j}$ as $W_1||W_2$. We now need to prove that W_1 and W_2 equal g^{R_i} and g^{C_j} respectively. W_1 is computed as $V_{1,i}/(U_{1,i})^x$, where $U_{1,i} = A_1^{r'_\alpha} = (g^{r_1})^{r'_\alpha} = g^{r_1 r'_\alpha}$ and

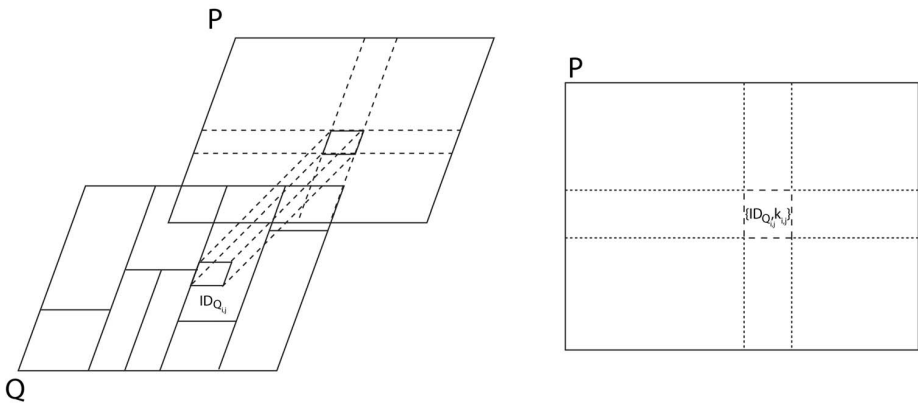

Fig. 4. Association between the public and private grids

$V_{1,i} = g^{R_\alpha}(g^\alpha B_1)^{r'_\alpha} = g^{R_\alpha}(g^\alpha g^{-i} y^{r_1})^{r'_\alpha}$, for $1 \leq i \leq n$. When $\alpha = i$ then $V_{1,i} = g^{R_i}(y^{r_1})^{r'_i} = g^{R_i} y^{r_1 r'_i}$. Raising $U_{1,i}$ to the power x gives $(U_{1,i})^x = (g^{r_1 r'_i})^x = g^{x r_1 r'_i} = y^{r_1 r'_i}$. Therefore, $W_1 = V_{1,i}/(U_{1,i})^x = g^{R_i}$. By similar means we can prove that $W_2 = V_{2,j}/(U_{2,j})^x = g^{C_j}$. Since $W_1 || W_2 = g^{R_i} || g^{C_j}$, then $K_{i,j} = K'_{i,j}$. Since $\oplus$ is self inverse and given that $Y_{i,j} = X_{i,j} \oplus H(K_{i,j})$, it follows that $X_{i,j} = Y_{i,j} \oplus H(K_{i,j})$. Using knowledge of $K_{i,j}$, the user can compute $X_{i,j}$ as desired. This completes the proof. □

D. Private Information Retrieval Based Protocol

With the knowledge about which cells are contained in the private grid, and the knowledge of the key that encrypts the data in the cell, the user can initiate a private information retrieval protocol[4] with the location server to acquire the encrypted POI data. Assuming the server has initialised the integer e, the user u_i and LS can engage in the following private information retrieval protocol using the $ID_{Q_{i,j}}$, obtained from the execution of the previous protocol, as input. The $ID_{Q_{i,j}}$ allows the user to choose the associated prime number power π_i, which in turn allows the user to query the server. The protocol is presented in Algorithm 3.

Theorem (Correctness) 2. *Assume that the user and the server follow the protocol correctly, then the user successfully acquires C_i for his/her chosen prime index.*

*P*roof: It is easy to see that $C_i = e \ (mod \ \pi_i)$ and $h_e = g_e^{|\langle g \rangle|/\pi_i}$. Then C_i is the discrete logarithm of h_e to the base h, since $g_e^{|\langle g \rangle|/\pi_i} = g^{e|\langle g \rangle|/\pi_i} = g^{e_{\pi_i}|\langle g \rangle|/\pi_i} = h^{e_{\pi_i}}$, where e_{π_i} stands for $e \ (mod \ \pi_i)$. This completes the proof. □

At the conclusion of the protocol, the user has successfully acquired the block that contain the encrypted POI records. With the knowledge of the cell key $k_{i,j}$, the user can decrypt C_i and obtain the requested data, thus concluding one round of the protocol. Using the same set-up, the user can execute several more rounds very efficiently and effectively without compromising his/her privacy. Similarly, the server's data remains protected based on the fact the user can only acquire one key per round. The security is analysed in more detail next.

[4]Appendix B presents a simple example of Gentry's private information retrieval scheme.

Algorithm 3 $PIRProtocol$

Input: User:$ID_{Q_{i,j}}$
Output: User:C_i

1: **User**
2: $\pi_0 \leftarrow \pi_i$, where π_i is chosen based on the value of $ID_{Q_{i,j}}$
3: Generate random group G and group element g, such that π_0 divides the order of g
4: $q \leftarrow |\langle g \rangle|/\pi_0$
5: $h \leftarrow g^q$
6: $Server \Leftarrow G, g$
7: **Server**
8: $g_e \leftarrow g^e$
9: $User \Leftarrow g_e$
10: **User**
11: $h_e \leftarrow g_e^q$
12: $C_i \leftarrow log_h h_e$, where log_h is the discrete log base h
13: **return** C_i {The requested (encrypted) data}

IV. Security Analysis

In this section, we analyse the security of the user and LS. While the user does not want to give up the privacy of his/her location, the server does not want to simply transfer all records to the user. This would not make much business sense in a variety of applications.

A. User's security

Fundamentally, the user does not want to disclose the cell $P_{i,j}$ which contains his/her location to the server. Two assumptions must be maintained in order to effectively render location private. The server must not be able to determine which cell the user is querying in the oblivious transfer protocol, and the server must not be able to determine which

cell the user is querying in the private information retrieval protocol.

The oblivious transfer assumption is based on the discrete logarithm assumption. This essentially means that given $g^x \ (mod\ p)$, where p is a large prime and g is a generator of some cyclic group, it is computationally infeasible to determine x. In our case, if the user supplies $(g^{r_1}, g^{-i}y^{r_1})$ and $(g^{r_2}, g^{-j}y^{r_2})$ to the server, then the server is unable to determine i and j. If the discrete logarithm assumption holds, then we claim this is secure.

The PIR assumption is based on the phi-hiding (ϕ-hiding) assumption. The phi-hiding assumption concerns the difficulty of factorising $\phi(N)$, where the factorisation of N is unknown. The ϕ function is known as Euler's totient function, and is defined as the cardinality of numbers that are less than N, and relatively prime to N (i.e. $gcd(N, a_i) = 1$). When N is composed of two distinct primes p and q, the totient function is defined as $\phi(N) = (p-1)(q-1)$. However, when the factorisation of N is unknown, this is believed to be difficult to compute and hence $\phi(N)$ is hard to factor.

The PIR protocol [9] uses this assumption to hide a secret prime power π_i within some multiplicative group with order $\phi(N)$, where the factorisation of N is known only to the user. The server is thus unable to determine the prime power π_i which divides $\phi(N)$, and hence is unable to determine the user's query. If both the oblivious transfer and PIR assumptions are true, then our solution is secure for the user.

B. Server's security

The server's security is based on keeping the boundaries of its records private. Since disclosing this information may enable the user to infer more information about the database than he/she is allowed. In our solution this information is protected by the oblivious transfer protocol.

The user is forced to retrieve one and only one record from the public grid $P_{i,j}$. This is because of the two sets of random values r'_α and r'_β. Only when $i = \alpha$ and $j = \beta$, the user can decrypt to find g^{R_i} and g^{C_i}. All other times, the result will be indistinguishable from random. Under the discrete logarithm problem assumption, it is computationally intractable to determine any exponent from the ciphertext. Hence, the user is only able to determine one and only one result.

V. Performance Analysis

We now analyse the performance of our solution and show that it is very practical. The performance analysis consists of the computation analysis and the communication analysis. We supplement this analysis with a comparison with the protocol by Ghinita et al. [12], [11].

A. Computation

Since the most expensive operation in our protocol is the modular exponentiation, we focus on minimising the number of times it is required. We assume that some components can be precomputed, and hence we only consider the computations needed at runtime. Furthermore, we reduce the number of exponentiations required by the PIR protocol to the number of multiplications that are required. This will make the computational comparison between our solution and the solution of Ghinita et al. easier to describe.

The transfer protocol is initiated by the user, who chooses indices i and j. According to our protocol the user needs to compute $(A_1, B_1) = (g^{r_1}, g^{-i}y^{r_1})$ and $(A_2, B_2) = (g^{r_2}, g^{-j}y^{r_2})$. Since the user knows the discrete logarithm of y (i.e. x), the user can compute (A_1, B_1) and (A_2, B_2) as $(A_1, B_1) = (g^{r_1}, g^{-i+xr_1})$ and $(A_2, B_2) = (g^{r_2}, g^{-j+xr_2})$ respectively. Hence, the user has to compute 4 exponentiations to generate his/her query.

Upon receiving the user's query, the server needs to compute $((A_1)^{r'_\alpha}, g^{R_\alpha}(g^\alpha(A_2))^{r'_\alpha})$ for $1 \leq \alpha \leq n$ and $((B_1)^{r'_\beta}, g^{C_\beta}(g^\beta(B_2))^{r'_\beta})$ for $1 \leq \beta \leq m$. Since g^α and g^β can be precomputed, the server has to compute $3n + 3m$ exponentiations.

The user requires an additional 2 more exponentiations to compute $(U_{1,i})^x$ and $(U_{2,j})^x$ to determine $K_{i,j}$. After the user has determined $K_{i,j}$, he/she can determine $X_{i,j}$ and proceed with the PIR protocol. This protocol requires 3 more exponentiations, 2 performed by the user and 1 performed by the server. In terms of multiplications, the user has to perform $2|N|$ operations and the server has to perform $|e|$ operations. The user also has to compute the discrete logarithm base h, log_h, of h_e. This process can be expedited by using the Pohlig-Hellman discrete logarithm algorithm [26]. The running time of the Pohlig-Hellman algorithm is proportional to the factorisation of the group order $O(\sum_{i=1}^{r} c_i(\lg n + \sqrt{p_i}))$, where r is the number of unique prime factors and n is the order of the group. In our case, the order of the group is $\pi_i = p_i^{c_i}$ and the number of unique factors is $r = 1$, resulting in running time $O(c(\lg p^c + \sqrt{p}))$.

When we compare our approach with the one by Ghinita et al. we find that our approach is computationally more efficient. Their protocol uses the homomorphic properties of the Paillier encryption scheme [24] in order to test whether a user is located in a cell or not. This requires the user to perform 4 exponentiations to compute the ciphertext of his/her coordinates, x and y. The server then has to compute $(4 \times (n \times m))$. The user has to decrypt at most all these ciphertexts $(4 \times (n \times m))$.

Once the user has determined his/her cell index he/she can proceed with the PIR protocol (described in [12]) to retrieve the data. The PIR is based on the Quadratic Residuosity Problem [20], which allows the user to privately query the database. Let t be the total number of bits in the database, where there are a rows and b columns. The user and server have to compute $2(\sqrt{a \times b}) \times \frac{|N|}{2}$ and $a \times b$ multiplications respectively. We remark that multiplying the whole database by a string of numbers, which is required by the PIR protocol based on the quadratic residuosity problem, is equivalent to computing g^e in our PIR protocol.

	Computation			Communication
	User	Server	Total	
Our Solution	6	$3n + 3m$	$6 + 3n + 3m$	$4L + 2(m + n)L$
Ghinita et al.	$4 + 4(n \times m)$	$4(n \times m)$	$4 + 4(n \times m) + 4(n \times m)$	$4L + 4(m \times n)2L$

TABLE I
STAGE 1 PERFORMANCE ANALYSIS SUMMARY

	Computation			Communication
	User	Server	Total	
Our Solution	$O(c(\lg p^c + \sqrt{p})) + 2\|N\|$	$\|e\|$	$O(c(\lg p^c + \sqrt{p})) + 2\|N\| + \|e\|$	$2L$
Ghinita et al.	$2(\sqrt{a \times b}) \times \frac{\|N\|}{2}$	$a \times b$	$2(\sqrt{a \times b}) \times \frac{\|N\|}{2} + a \times b$	$\sqrt{a \times b}L$

TABLE II
STAGE 2 PERFORMANCE ANALYSIS SUMMARY

B. Communication

Since we require the discrete logarithm to be intractable for security reasons, we set the modulus p to be 1024 bits in size. Hence, one ElGamal encryption is 2048 bits. Let L be the length of an element in the ElGamal ciphertext, 1024. In our proposed solution, the user needs $4L$ communication, while the server requires $2(m + n)L$ communication in the oblivious transfer protocol. In the PIR protocol, the user and server exchange one group element each.

Since the solution by Ghinita et al. uses the Paillier encryption scheme, which has equivalent ciphertext size as ElGamal ciphertext, then the size of one size of one ciphertext in their scheme is $2L$. Based on this parameter, the user has to submit $4L$ bits to the server as their encrypted location. Then the server has to send $4 \times n \times m \times 2L$, for the user to determine his/her location. For the PIR based on the QRA, the user and server have to send $\sqrt{a \times b} \times L$. The performance analysis for stage 1 (user location test) and stage 2 (private information retrieval) are summarised in Tables I and II respectively, where the computation in Table I is in terms of exponentiation and the computation in Table II is in terms of multiplication.

When we analyse the difference in performance between our solution and the one by Ghinita et al., we find that our solution is more efficient. The performance of the first stage of each protocol is about the same, except our solution requires $O(m + n)$ operations while the solution by Ghinita et al. requires $O(m \times n)$. In the second stage, our protocol is far more communicationally efficient, requiring the transmission of only 2 group elements whereas the Ghinita et al. solution requires the exchange of an $a \times b$ matrix.

VI. EXPERIMENTAL EVALUATION

We implemented a prototype of our location based query solution using the C++ programming language. We measured the required time for the oblivious transfer and private information retrieval protocols separately to test the performance of each protocol and the relative performance between the two protocols. The prototype was created on a machine with a Intel Core 2 Duo E8200 2.66GHz processor and 2GB of RAM. The prototype was written using Visual C++ under the Windows XP operating system. We used the Number Theory Library (NTL) [27] for computations requiring large integers and OpenSSL [1] to compute the SHA-1 hash. The whole solution was executed for 100 trials, where the time taken (in seconds) for each major component was recorded and the average time was calculated.

Component	*Average Time (s)*
Initialisation	0.28829
Query	0.00484
Response	0.11495
Decode	0.00031

TABLE III
AVERAGE TIME REQUIRED FOR OBLIVIOUS TRANSFER PROTOCOL

A. Oblivious Transfer Protocol

In our implementation experiment for the oblivious transfer protocol, we generated a modified ElGamal instance with $|p| = 1024$ and $|q| = 160$, where $q|(p-1)$. We also found a generator a, and set $g = a^q$ (g has order q). We set the public matrix P to be a 25×25 matrix of key and index information.

We first measured the time required to generate a matrix of keys according to Algorithm 1. This procedure only needs to be executed once for the lifetime of the data. There is a requirement that each hash value of the concatenation $g^{R_i}||g^{C_j}$ is unique. We use the SHA-1 to compute the hash $H(\cdot)$, and we assume that there is negligible probability that a number will repeat in the matrix.

The major three components of the oblivious transfer protocol are the user's query, server's response, and user's decode. Table III displays the average time required for each component of the protocol. The magnitude of the numbers in Table III demonstrates that our protocol is efficient at runtime.

B. Private Information Retrieval Protocol

In the PIR we fixed a 15×15 private matrix, which contains the data owned by the server. We chose the prime set to be the first 225 primes, starting at 3. The powers for the primes were chosen to allow for at least a block size of 1024 bits ($3^{647}, 5^{442}, ..., 1429^{98}$). Random values were chosen for each prime power $e = C_i \ (mod\ \pi_i)$, and the Chinese Remainder

Component	*Average Time (s)*
Query	9.64984
Response	4.57127
Decode	0.25451

TABLE IV
AVERAGE TIME REQUIRED FOR PRIVATE INFORMATION RETRIEVAL PROTOCOL

Theorem was used to determine the smallest possible e satisfying this system of congruences.

Once the database has been initialised, the user can initiate the protocol by issuing the server his/her query. The query consists of finding a suitable group whose order is divisible by one of the prime powers π_i. We achieve this in a similar manner to Gentry and Ramzan [9]. We choose primes q_0 and q_1 and compute "semi-safe" primes $Q_0 = 2q_0\pi_i + 1$ and $Q_1 = 2q_1 + 1$. We set the modulus as $N = Q_0Q_1$ and group order as $\phi(N) = \phi(Q_0Q_1) = (Q_0 - 1)(Q_1 - 1)$. Hence, the order $\phi(N)$ has π_i as a factor. We set g to be a quasi-generator, such that the order of g also contains π_i. In our experiment, we set $|q_0| = |q_1| = 128$. This results in a modulus N which is roughly 1024 bits in length, which is equivalent to an RSA modulus.

As in the oblivious transfer based protocol there are 3 major steps: the user's query, the server's response, and the user decoding. The average time required for each of these major components are presented in Table IV.

Based on these experimental results, most of time is taken by the generation of the user's query. This is due to the primality testing of Q_0 and Q_1. This requirement must be satisfied, otherwise we would not be able to compute the order as $\phi(N) = (Q_0 - 1)(Q_1 - 1)$, and the factorisation of the order would not contain π_i. The average of the response time and the decoding time are much smaller in comparison. We assume that the server has much more computational power at its disposal. Hence, if there are many users, the server can use parallel processing to increase the throughput of the protocol. The main concern is keeping the query time for the user as low as possible, and on average the user query time is reasonable, given the amount of data that is exchanged in one round of the protocol.

VII. CONCLUSION

In this paper we have presented a location based query solution that employs two protocols that enables a user to privately determine and acquire location data. The first step is for a user to privately determine his/her location using oblivious transfer on a public grid. The second step involves a private information retrieval interaction that retrieves the record with high communication efficiency.

We analysed the performance of our protocol and found it to be both computationally and communicationally more efficient than the solution by Ghinita et al., which is the most recent solution. We also implemented a software prototype that highlighted the efficiency of our protocol. The software prototype was run on a contemporary desktop machine and the execution time was recorded.

Future work will involve porting the software to a mobile device to test the actual feasibility of our proposed protocol. Additionally, the problem concerning the LS supplying misleading data to the client is also interesting. Privacy preserving reputation techniques seem a suitable approach to address such problem. A possible solution could integrate methods from [13]. Once suitable strong solutions exist for the general case, they can be easily integrated into our approach.

ACKNOWLEDGMENT

This work was supported by ARC Discovery Project (DP0988411) "Private Data Warehouse Query" and NSF award (1016722) "TC: Small: Collaborative: Protocols for Privacy-Preserving Scalable Record Matching and Ontology Alignment". Also, we thank the anonymous reviewers for providing thoughtful and detailed feedback on this paper.

REFERENCES

[1] "Openssl," http://www.openssl.org/, 2011, [Online; accessed 7-July-2011].
[2] M. Bellare and S. Micali, "Non-interactive oblivious transfer and applications," Proc. *CRYPTO'89*. 1990, pp. 547 - 557.
[3] A. Beresford and F. Stajano, "Location privacy in pervasive computing," *IEEE Pervasive Computing*, vol. 2, no. 1, pp. 46 - 55, 2003.
[4] C. Bettini, X. Wang, and S. Jajodia, "Protecting privacy against location-based personal identification," Proc. *Secure Data Management*, Lecture Notes in Computer Science, W. Jonker and M. Petkovic, Eds., 2005, vol. 3674, pp. 185 - 199.
[5] B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan, "Private information retrieval," *J. ACM*, vol. 45, no. 6, pp. 965 - 981, 1998.
[6] M. Duckham and L. Kulik, "A formal model of obfuscation and negotiation for location privacy," Proc. *Pervasive Computing*, Lecture Notes in Computer Science, H. Gellersen, R. Want, and A. Schmidt, Eds., 2005, vol. 3468, pp. 243 - 251.
[7] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms,", *IEEE T. Information Theory*, vol 31, no. 4, pp. 469 - 472, 1985.
[8] B. Gedik and L. Liu, "Location privacy in mobile systems: A personalized anonymization model," Proc. *ICDCS'05*, 2005, pp. 620 - 629.
[9] C. Gentry and Z. Ramzan, "Single-database private information retrieval with constant communication rate," Proc. *Automata, Languages and Programming*, Lecture Notes in Computer Science, L. Caires, G. Italiano, L. Monteiro, C. Palamidessi, and M. Yung, Eds., 2005, vol. 3580, pp. 803 - 815.
[10] G. Ghinita, P. Kalnis, M. Kantarcioglu, and E. Bertino, "A hybrid technique for private location-based queries with database protection," Proc. *Advances in Spatial and Temporal Databases*, Lecture Notes in Computer Science, N. Mamoulis, T. Seidl, T. Pedersen, K. Torp, and I. Assent, Eds., 2009, vol. 5644, pp. 98 - 116.
[11] G. Ghinita, P. Kalnis, M. Kantarcioglu, and E. Bertino, "Approximate and exact hybrid algorithms for private nearest-neighbor queries with database protection," *GeoInformatica*, pp. 1 - 28, 2010.
[12] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K.-L. Tan, "Private queries in location based services: anonymizers are not necessary," Proc. *SIGMOD'08.*, 2008, pp. 121 - 132.
[13] G. Ghinita, C. R. Vicente, N. Shang, and E. Bertino, "Privacy-preserving matching of spatial datasets with protection against background knowledge," Proc. *GIS '10*, 2010, pp. 3 - 12.
[14] M. Gruteser and D. Grunwald, "Anonymous usage of location-based services through spatial and temporal cloaking," Proc. *1st international conference on Mobile systems, applications and services*, 2003, pp. 31 - 42.
[15] T. Hashem and L. Kulik, "Safeguarding location privacy in wireless ad-hoc networks," Proc. *UbiComp'07*, 2007, pp. 372 - 390.
[16] B. Hoh and M. Gruteser, "Protecting location privacy through path confusion," Proc. *SecureComm'05*, 2005, pp. 194 - 205.

[17] P. Kalnis, G. Ghinita, K. Mouratidis, and D. Papadias, "Preventing location-based identity inference in anonymous spatial queries," *IEEE T Knowledge and Data Engineering*, vol. 19, no. 12, pp. 1719 - 1733, 2007.

[18] H. Kido, Y. Yanagisawa, and T. Satoh, "An anonymous communication technique using dummies for location-based services," Proc. *ICPS'05*, 2005, pp. 88 - 97.

[19] J. Krumm, "A survey of computational location privacy," *Personal and Ubiquitous Computing*, vol. 13, pp. 391 - 399, 2009.

[20] E. Kushilevitz and R. Ostrovsky, "Replication is not needed: single database, computationally-private information retrieval," Proc. *Foundations Computer Science*, 1997, pp. 364 - 373.

[21] S. Mascetti and C. Bettini, "A comparison of spatial generalization algorithms for lbs privacy preservation," Proc. *2007 International Conference on Mobile Data Management*, 2007, pp. 258 - 262.

[22] M. F. Mokbel, C.-Y. Chow, and W. G. Aref, "The new casper: query processing for location services without compromising privacy," Proc. *VLDB'06*, 2006, pp. 763 - 774.

[23] M. Naor and B. Pinkas, "Oblivious transfer with adaptive queries," Proc. *CRYPTO'99*, 1999, vol. 1666, pp. 791 - 791.

[24] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," Proc. *EUROCRYPT'99*, 1999, vol. 1592, pp. 223 - 238.

[25] B. Palanisamy and L. Liu, "Mobimix: Protecting location privacy with mix-zones over road networks," Proc. *ICDE'11*, 2011, pp. 494 - 505.

[26] S. Pohlig and M. Hellman, "An improved algorithm for computing logarithms over gf(p) and its cryptographic significance (corresp.)," *IEEE T Information Theory*, vol. 24, no. 1, pp. 106 - 110, 1978.

[27] V. Shoup, "Number thoery library," http://www.shoup.net/ntl/, 2009, [Online; accessed 7-July-2011].

[28] L. Sweeney, "k-anonymity: a model for protecting privacy," *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, vol. 10, pp. 557 - 570, 2002.

APPENDIX A
AN EXAMPLE OF ADAPTIVE OBLIVIOUS TRANSFER

The purpose of this appendix is to demonstrate the operation of the Adaptive Oblivious Transfer [23] protocol with an example. The example we present is simplified to improve the clarity. Let us first create an ElGamal instance. Let $p = 1031$ be the prime modulus and $g = 14$, be the generator of the multiplicative group with order $\phi(p) = p - 1 = 1030$.

The oblivious transfer protocol is then initialised by creating a matrix $K_{i,j}$ from arrays R_i and C_j as $K_{i,j} = g^{R_i}||g^{C_j}$. In this example, we will set $R_i = [7, 33, 51, 27]$ and $C_j = [21, 10, 24, 37]$. Once this database of keys have been initialised, the user can construct their query.

The user constructs his/her query by first creating a private key $x = 49$, and setting the public key as $y = g^x = 247$. Next the user sets $i = 2$ and $j = 3$ and selects two random numbers r_1 and r_2. In this example we will use $r_1 = 24$ and $r_2 = 14$. Then the user computes $\boldsymbol{C_1} \leftarrow (A_1, B_1) = (g^{r_1}, g^{-i}y^{r_1}) = (373, 685)$ and $\boldsymbol{C_2} \leftarrow (A_2, B_2) = (g^{r_2}, g^{-j}y^{r_2}) = (507, 183)$. The user then sends this query $(\boldsymbol{C_1}, \boldsymbol{C_2})$ to the server.

Upon receiving the query from the user the server generates 2 sets of random numbers r'_α for $1 \leq \alpha \leq 4$ and r'_β for $1 \leq \beta \leq 4$. In our example we will use $r'_\alpha = [786, 33, 783, 323]$ and $r'_\beta = [382, 897, 806, 449]$. Then the server computes $\boldsymbol{C'_{1,\alpha}} \leftarrow (A_1^{r'_\alpha}, g^{R_\alpha}(g^\alpha B_1)^{r'_\alpha})$ for $1 \leq \alpha \leq 4$ and $\boldsymbol{C'_{2,\beta}} \leftarrow (A_2^{r'_\beta}, g^{C_\beta}(g^\beta B_2)^{r'_\beta})$ for $1 \leq \beta \leq 4$. Using the numbers in our example we have the following ciphertexts.

$$\begin{aligned}\boldsymbol{C'_{1,\alpha}} &= [(184, 679), (46, 62), (661, 845), (271, 597)]\\ \boldsymbol{C'_{2,\beta}} &= [(471, 693), (471, 734), (512, 1012), (357, 119)]\end{aligned}$$

These ciphertexts are transferred back to the user. The only ciphertexts that user can use is when $\alpha = i$ and $\beta = j$, which are $\boldsymbol{C'_{1,2}}$ and $\boldsymbol{C'_{2,3}}$. Let $(U_1, V_1) = \boldsymbol{C'_{1,i}} = (46, 62)$ and $(U_2, V_2) = \boldsymbol{C'_{1,j}} = (512, 1012)$. The user then computes $W_1 = V_1/(U_1)^x = 425$ and $W_2 = V_2/(U_2)^x = 373$. The user can then construct the key $K_{2,3}$ as $W_1||W_2$, since $W_1 = g^{R_2} = 425$ and $W_2 = g^{C_3} = 373$.

APPENDIX B
AN EXAMPLE OF GENTRY'S PRIVATE INFORMATION RETRIEVAL SCHEME

In this appendix, we present an artificially small example of Gentry's Private Information Retrieval scheme [9]. The example begins with the set up of a simple database with each record represented by a single integer. Such database is displayed below.

$$\begin{aligned}e &= 31 \ (mod\ 7^2)\\ e &= 51 \ (mod\ 11^2)\\ e &= 68 \ (mod\ 13^2)\end{aligned}$$

We can use the Chinese Remainder Theorem to find a number e that satisfies this system of congruences. The smallest such value is $e = 17475$. The number e and the list of moduli can alone represent the database. When we want to query this database, we first choose a prime power from the list. In our example we will choose $\pi = 7^2$.

Then we embed this prime power into some group G where the order is divisible by π. More specifically, when $\pi|\phi(N)$, for some composite integer N. This is achieved by computing two "semi-safe" primes $Q_0 = 2q_0\pi + 1$ and $Q_1 = 2dq_1 + 1$, where q_0, q_1 are both prime, and d is taken randomly from some suitable distribution. Primality tests, like the Miller-Rabin primality test or the Fermat primality test, can be used to determine whether Q_0, Q_1 are probably prime in reasonable time.

To construct our query with $\pi|\phi(N)$, we set $q_0 = 17$, $q_1 = 19$, and $d = 8765$. This results in $Q_0 = 1667$ and $Q_1 = 333071$. Now, we compute the product N, as $N = Q_0Q_1 = 555229357$ and the order $\phi(N)$ as $\phi(N) = (Q_0-1)(Q_1-1) = 554894620$. As we can see by the following factorisation of the order $\phi(N)$, this order contains the factor 7^2.

$$\phi(N) = 554894620 = 2^2 \cdot 5^1 \cdot 7^2 \cdot 17^1 \cdot 19^1 \cdot 1753^1$$

We randomly set a "quasi"-generator generator g as $g = 3$, whose order $|\langle g \rangle| = 138723655$, which contains the factor π. The query is composed of (N, g), where the factorisation of

N is kept secret. The integer q, is computed as $q = |\langle g \rangle|/\pi = 2831095$. The user sends the query (N, g) to the server.

The server computes $g_e = g^e$, where e represents the database. In our case $g_e = 127319266$. This is sent back to the user. The user then computes $h_e = g_e^q = 65281217$ and $h = g^q = 474959247$. The user computes the discrete logarithm base h of h_e: $log_h h_e$. By brute force, we can search through all powers of h, until one equals $h_e = 65281917$. The following equations illustrate this method.

$$\begin{aligned} &\vdots \\ x = 30 \quad &\Rightarrow h^x = 138224466 \neq h_e \\ x = 31 \quad &\Rightarrow h^x = 65281917 = h_e \\ x = 32 \quad &\Rightarrow h^x = 274783576 \neq h_e \\ &\vdots \end{aligned}$$

Hence, we have discovered the record corresponding to $\pi = 7^2$. Instead of using brute force, the Pohlig-Hellman discrete logarithm algorithm [26] can be employed to speed up this process. The running time of the Pohlig-Hellman algorithm is dependant on the order of the group. With the current query (N, g), the order of interest is $\pi = 7^2$.

The algorithm begins with the order of $|\langle h \rangle| = 7^2 = 49$. This instance of the discrete logarithm can be written as $h^x = h_e \ (mod\ N)$, where we solve for x. Substituting α, β for h and h_e we have $\alpha^x = \beta \ (mod\ N)$. In our example $\alpha = 474959247$ and $\beta = 65281917$.

In general terms, we want to solve $x_1 = x \ (mod\ 7^2)$. We write $x = c_0 + c_1(7) \ (mod\ 7^2)$, where $c_0, c_1 \in [0, 7)$. We compute $\beta_0 = \beta^{(49)/7} = 466965543$ and $\alpha_1 = \alpha^{(49)/7} = 98589017$. We now calculate all possible powers x of α_1, where $x \in [0, 7)$. These values are displayed in Table V.

x	α_1^x
1	98 589 017
2	230 485 133
3	466 965 543
4	543 238 802
5	127 566 194
6	21 649 616
7	1

TABLE V
ALL POSSIBLE POWERS OF α_1

When we look up β_0, we find that $x = 3$ and hence $c_0 = 3$. We then compute a new β as $\beta_1 = \beta(\alpha^{-3}) = 543238802$. We raise β_1 to the power of $49/49$, we get the same result $\beta_1^{49/49} = 543238802 = \beta_1$. When we look β_1 in Table V, we get $x = 4$ and hence $c_1 = 4$. Substituting c_0, c_1 into $x = c_0 + c_1(7) \ (mod\ 7^2)$, we get $x = 3 + 4(7) = 3 + 28 = 31 \ (mod\ 7^2)$. This gives the same result as the brute force approach, however, this approach is far less computationally expensive.