Location Privacy

Mark van Cuijk and Barry Weymes

December 5, 2010

Abstract

As more individuals carry mobile devices that have the capability to determine their current location and communicate this information to a global network, location based services which provide the user with personalized information emerge. Several researchers have attempted to formalize the privacy impact of such services and the level of detailed knowledge they obtain about users. Several algorithms to cloak the exact location of individuals have been designed, each of them delivering a certain balance between privacy and usability. This paper presents the results of a small-scale interview performed by the authors, summarizes several methods to cloak location data and explains an algorithm for a privacy-aware location query processor.

1 Introduction

As the progression of location based services (LBS) becomes greater and the ability to locate oneself on a map using GPS, the possibility of abuse by others becomes greater also. LBS will have a significant role to play in the future. Uses such as finding the nearest shop in unfamiliar terrain or detecting traffic patterns are already being utilized more and more these days.

While these new technologies may have benefits, they also carry the possibility of violating ones privacy. For example, a simple location based query to find the nearest train to your location could be used to track ones movements if it is queried sufficiency frequently enough. Tracking movements of a user may allow an adversary to deduce additional information, e.g. someone frequently visiting a cancer treatment facility probably has cancer; someone that unexpectedly visits a competitor of his employer might be shopping for a new job.

There are many privacy enhancing technologies (PET) that can help protect ones privacy. We will outline some of the different PET that are being researched in the field of location privacy. The main idea in most of the PETs is to be anonymous within a group of others at your location, therefore preserving ones privacy. k-anonymous is the general term to describe how one subject cannot be identifiable from $k-1$ other subjects at the same location at a given time.

This paper starts with a small-scale interview research we conducted, in order to find out whether students of the Radboud University in Nijmegen are using LBS and whether they're concerned about their privacy when using such services. Section 3 introduces the concept of k-anonymity and demonstrates how this notion can be applied to location information. Several algorithms are discussed that combine location data from several users in order to present anonymized data to an LBS. For an LBS to use anonymized data, it needs modified search algorithms, one of which is presented in section 4: find a candidate set of nearest objects, given a cloaked user location. Section 5 discusses three location PETs in a social setting: it allows two users of the system to determine the physical distance between each other, without revealing exact location information in the first place. Finally, a wrap up is done in section 6.

2 Interview

As part of this project we perform a small interview. This section enlists the questions we posed, a description of the subjects we used for the interview and some results of the interview.

2.1 Questions

We wanted to find out whether people are already using location based services and to see what their level of understanding of location privacy is.

In question section 1, we ask the students about their usage of location based services to get them in the correct frame of mind. In question section 2a and 2b we ask about the students current usage to gauge their experiences with the technology. Question section 3 is designed to query the students impressions of location privacy after determining their

usage levels. Finally in question section 4, we ask the students about scenarios that may change the way they think about the subject. By comparing the answers to the questions in sections 3 and 4, we want to determine whether the concerns students acknowledge agree with how they respond to specific scenarios.

Section 1 Do you use location based services on:

- Q1. Your smart phone
- Q2. In-car navigation system
- Q3. Laptop or desktop computer

Section 2a Have you ever used location based services to find:

- Q1. An ATM
- Q2. A gas station
- Q3. A specific shop
- Q4. Currently nearby friends

Section 2a Have you ever used:

- Q1. Trackr![1]
- Q2. FourSquare[2]
- Q3. Twitter location tags[3]
- Q4. Photo camera with GPS

Section 3 Do you have location privacy concerns with:

- Q1. Google Maps
- Q2. Photo camera with GPS
- Q3. FourSquare

Section 4 Are you concerned about the following scenario:

- Q1. "While on vacation in Cuba you make known to the public that you visited there. Some years later while entering the US a customs officer asks you what you were doing in Cuba."
- Q2. "You forgot to tell your boyfriend or girlfriend that your going to someone else's house to study. He or she finds out because of LBS and asks for an explanation."
- Q3. "For your job, you visit a customer. On the way back to the office you make a lengthy detour. You employer finds out about the detour and asks for an explanation."

2.2 Results

We questioned 69 students of the Radboud University, with a slight bias towards students following a technically-oriented educational program. These results are therefore not representative for society, but we expect this target group to resemble higher educated individuals of the younger generation.

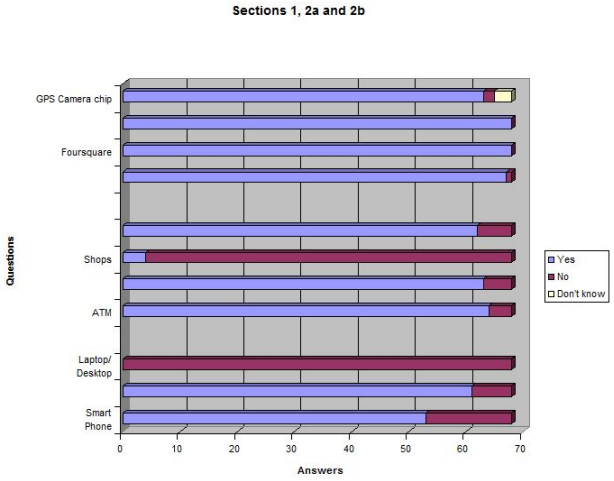

Figure 1: Depicts the answers from the students on the questions in sections 1, 2a and 2b. Each column represents a question.

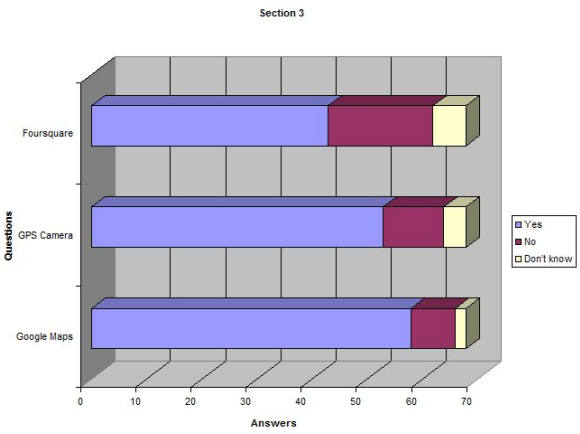

Figure 2: This figure shows the answers to the questions of section 3. Each column represents a privacy concern.

[1] trackr.nl allows users to track the location of a device and display the movements on a map in real-time

[2] FourSquare is a service that allows users to check-in into a location in order to share this information in social networks

[3] Twitter is a micro-blogging service, allowing users to post 140 character messages to a stream. Certain twitter clients can report the location of the user at the time the message is composed.

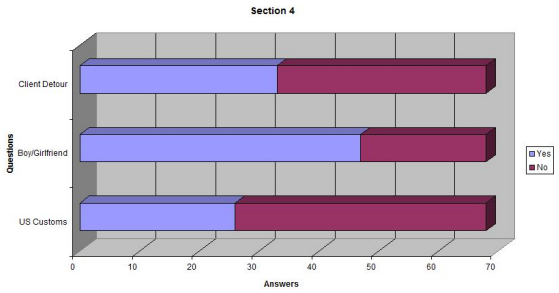

Figure 3: This figure shows the results of the questions of section 4.

2.3 Analysis

Section 1 show that very few people currently use smart phones and in-car navigation systems, while all students use location based services – like Google Maps or Bing Maps – on their personal computer. We expect that smart phone usage is low because of the high prices of these devices compared to normal phones and the fact that our target group is students, which have a relatively low budget. Also car usage low among students, so this probably explains why in-car navigation systems are not much used in our target group.

Section 2a shows that not many students use location based services to find anything other than the location of certain shops. In an additional ad-hoc questionnaire, we discovered very few students are aware that they can find ATMs using an LBS like Google Maps. As with section 1, we note that many students don't own a car and therefore probably never need to find a gas station. Many students also don't use the functionality of being able to detect if a friend is close proximity to them. The most common use for location based services was finding a specific shop, e.g. finding the closest GAP shop: 64 of the students have used this type of service.

In section 2b, we asked the student which location based services they use, other then finding the location of a certain object. The results is that most students don't use or don't know about many of these services. Only one person indicated that they use Trackr! to plot his movements on an interactive map and only 2 people use a camera with GPS functionality, like the iPhone 4 that automatically adds the GPS coordinates to each picture. Some students responded that they don't know whether their photo camera tracks GPS coordinates.

In section 3, we ask about the students concerns about location privacy. 85% of the students report that they are concerned about Google recording their locations when using Google Maps. Less people are concerned with the privacy issues of having a GPS-equipped camera and the website FourSquare.

Finally in section 4, students report they most concerned that their partner would use an LBS to track discover about their movements to a friends house. Regarding questions 1 and 3, some noted that when deciding to visit the US gives nation officials the right to interrogate you as a foreigner and that an employer is entitled to request explanation of what an employee does during work hours.

We expected students to show less concern on questions in section 3 and expected students who discover about potential dangers that researchers talk about – by hearing about the scenarios in section 4 – to raise more concern. However, the results show the contrary: in section 3 students claim they are rather concerned about their privacy, while the answers in section 4 show they accept government officials or employers to question their behavior. Although probably a more psychological topic, it can be interesting to do further research given these results.

3 Location k-Anonymity

In 2002, Sweeney describes a model to allow queries on a database to obtain valuable information from a database containing personal information, while at the same time retaining a certain level of privacy. This section described the original k-anonymity model by Sweeney, explains how Gruteser and Grunwald apply this model to location information and describes two algorithms that are built upon the k-anonymity concept: CliqueCloak by Gedik and Liu and a peer-to-peer cloaking algorithm by Chow, Mokbel and Liu.

3.1 k-Anonymity in general

To do business, organizations need to keep personal information. For example, a bank needs to maintain financial data on an individual and a hospital needs to maintain medical records. Often, this data is useful to conduct research, which is often done by a third party, but law and privacy policies enforce organizations to be careful with revealing such information about individuals.

In [1], Sweeney describes a common practice for organizations to remove explicit identifiers from person-specific data, like name, address and telephone number. The assumption that is made is that the lack of explicit identifiers make the data secure with respect to the ability to identify persons. Sweeney shows how 87% of the population in the United States can likely be uniquely identified based only on {5-digit ZIP, gender, date of birth}

[2]. Combining data from two sources – medical data made available to researchers [3] and a voters list that Sweeney purchased for twenty dollars [4] – it seemed possible to link diagnosis, procedures and medications to particularly named individuals.

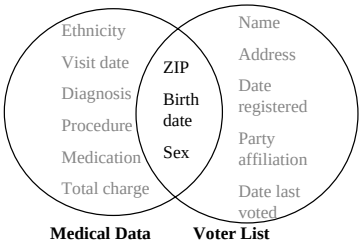

Figure 4: The left circle contains part of the information Sweeney obtained from [3]; the right circle contains the information Sweeney obtained from [4]. The three attributes in the intersection allows the two data sets to be successfully combined.

As it was shown that just removing explicit identifiers is not sufficient to properly anonymize a certain data set, Sweeney proposes a model that allows organization to publish or sell information from their database, while making sure that the identity of individuals cannot be deduced by combining the information from this data set with any other source. To get to such a strict goal, the k-anonymity model is proposed to ensure that any release of information about a single individual cannot be distinguished from the information about at least $k-1$ other individuals. A formal definition is given in [1]:

Attributes Let B$(A_1, \ldots, A_n)$ be a *table* with a finite number of tuples. The finite set of *attributes* of B are $A_1, \ldots, A_n$.

Quasi-identifier Given a population of entities U, an entity-specific table T$(A_1, \ldots, A_n)$, f_c: U $\rightarrow$ T and f_g: T $\rightarrow$ U', where U $\subseteq$ U'. A quasi-identifier of T, written Q_T, is a set of attributes $A_i, \ldots, A_j \subseteq A_1, \ldots, A_n$ where: $\exists p_i \in$ U such that $f_g(f_c(p_i)[Q_T]) = p_i$.

k-anonymity Let RT$(A_1, \ldots, A_n)$ be a table and QI_{RT} be the quasi-identifier associated with it. RT is said to satisfy k-anonymity if and only if each sequence of values in RT$[QI_{RT}]$ appears with at least k occurrences in RT$[QI_{RT}]$.

3.2 Applied to location data

Gruteser et al [5] looks at a way to apply Sweeney et al's method [1] of anonymizing data in general, to the field of location data. This algorithm seeks the provide anonymity regardless of the population density by setting a minimum amount of possible nodes that any communication could have originated from. The number used to describe this is called k_{min}.

The system model contains an anonymity server that receives location information from nodes that communicate with it. This anonymity server acts as a middle man for all services that the nodes wish to use. Once a node requests a service the anonymity server perturbs the location information and forwards the message to the external service.

The location of the node can be represented by the tuple $([x_1, x_2], [y_1, y_2][t_1, t_2])$ where $[x_1, x_2]$ and $[y_1, y_2]$ describe an two dimensional area that the subject was present in and $[t_1, t_2]$ represents the time range at which the subject was in the area. This tuple can be considered anonymous when it describes the location of other subjects at the same time.

The key approach to this algorithm is that the anonymity server monitors the locations of its nodes. The nodes request for a service will be delayed until the required k_{min} nodes have visited the area. Then the request will be anonymous. The time t_2 is set to the current time and t_1 is set the the time of the request minus a random cloaking value.

The location tuple can be considered k-anonymous once an adversary cannot uniquely identify a subject through observation, since the tuple also matches $k-1$ other subjects.

3.2.1 Analysis

However in some circumstances an adversary can reveal a nodes location because of a weakness in the algorithm.

Consider the following location tuples:

1. $([0, 1], [0, 1][t_1, t_2])$
2. $([1, 2], [0, 1][t_1, t_2])$
3. $([0, 1], [1, 2][t_1, t_2])$
4. $([0, 2], [0, 2][t_1, t_2])$

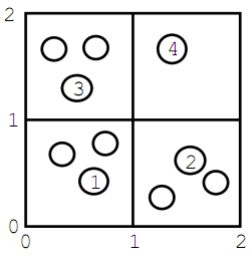

Figure 5: This figure displays a weakness in the algorithm, obtained from [5]; If node 4 requests a service it can be singled out because the algorithm must widen the quadrant, if k_{min} is greater than 1 - which is must be to preserve privacy

All four tuples share the same time attributes. Let us take k_{min} to be 3, for example. In figure 5, the node 4 can be singled out in communications as all other quadrants have sufficient possible nodes to prevent a privacy violation. If each numbered node requests a service at the same time node 4 could be identified. This simple example shows that the privacy of the subjects cannot be guaranteed.

3.3 CliqueClock

While the algorithm by Gruteser and Grunwald (section 3.2) allows the client to specify the acceptable spatial and temporal resolution for each individual message, the minimum anonymity set size k_{min} is a global parameter and is therefore the same for all messages.

In an attempt to overcome this limitation, Gedik and Liu designed the *CliqueCloak* algorithm that can handle messages that each have individual spatial and temporal resolution requirements, but also have individual privacy constraints by setting a k_{min} on each message [6].

3.3.1 Concept of CliqueCloak

The algorithm takes as input a set of messages $m_s \in S : \langle u_{id}, r_{no}, \{t, x, y\}, k, \{d_t, d_x, d_y\}, C\rangle$ from mobile nodes and transforms these into a set of transformed messages $m_t \in T : \langle u_{id}, r_{no}, \{X : [x_s, x_e], Y : [y_s, y_e], I : [t_s, t_e]\}, C\rangle$. The message with sequence number r_{no} encodes the location (x, y) of user u_{id} at timestamp t, the requested minimum anonymity set size k, required spatio-temporal resolution (d_x, d_y, d_t) and a user message C. The box defined by the ranges $[x - d_x, x + d_x]$, $[y - d_y, y + d_y]$ and $[t - d_t, t + d_t]$ is called the *anonymity constraint box*. In the output message, the spatio-temporal information is transformed into a *cloaking box* defined by a range for both spatial dimensions X and Y and also for the temporal dimension I.

Each output message m_t relates to exactly one input message m_s; each input message m_s relates to at most one output message m_t. This means that the algorithm cannot "come up" with new message, but is able to drop input message if it is unable to construct an output message with the requested spatio-temporal resolution and privacy requirements.

Gedik and Liu introduce a couple of properties that must hold for all messages:

Spatial and Temporal Containment state that the cloaking box contains the exact location of the user: $m_s.x \in m_t.X$, $m_s.y \in m_t.Y$ and $m_s.t \in m_t.I$

Spatial and Temporal Resolution state that the anonymity constraint box contains the cloaking box entirely: $m_t.X \subset [m_s.x - m_s.d_x, m_s.x + m_s.d_x]$, $m_t.Y \subset [m_s.y - m_s.d_y, m_s.y + m_s.d_y]$ and $m_t.I \subset [m_s.t - m_s.d_t, m_s.t + m_s.d_t]$

Content Preservation states that the user message is passed on unmodified: $m_s.C = m_t.C$

Location k-anonymity states that there are at least $k - 1$ messages, each from a different node, that are mapped to the same spatio-temporal cloaking box: $\exists T' \subset T$, such that $m_t \in T'$, $|T'| \geq m_s.k$, $\forall_{\{m_{t,i}, m_{t,j}\} \in T'} : m_{t,i}.u_{id} \neq m_{t,j}.u_{id}$ and $\forall_{m_{t,i} \in T'} : m_t.X = m_{t,i}.X \wedge m_t.Y = m_{t,i}.Y \wedge m_t.I = m_{t,i}$

3.3.2 Algorithm description

The idea of the *CliqueCloak* algorithm is to define an undirected graph, such that each vertex represents a message $m_s \in S$ and an edge between two vertices $m_{s,1}$ and $m_{s,2}$ means that the messages $m_{s,1}$ and $m_{s,2}$ can be transformed into the same cloaking box and respecting all requested resolution and privacy constraints.

In a graph $G(S, E)$, S being the set of vertices and E the set of edges, an edge $e = (m_{s,i}, m_{s,j}) \in E$ exists if and only if $\mathrm{P}(m_{s,i}) \in \mathrm{B_{cn}}(m_{s,j}))$, $\mathrm{P}(m_{s,j}) \in \mathrm{B_{cn}}(m_{s,i}))$ and $m_{s,i}.u_{id} \neq m_{s,j}.u_{id}$, where $\mathrm{P}(m_s)$ is the (x, y, t) coordinate of message m_s and $\mathrm{B_{cn}}(m_s)$ denotes the anonymity constraint box of message m_s. This means that the two messages must be sent by two different users and that both locations must lie within each others anonymity constraint box.

Having this graph, it is straightforward to translate the problem of finding a set of messages that can be transformed to the same cloaking box into the problem of finding cliques in the graph $G(S, E)$ of the size that is at least equal to the largest k_{min} of the individual messages.

Each time a new message is received, the algorithm creates a vertex in the graph, determines which messages can share a cloaking box with the new message and creates the corresponding edges. Since only vertices that are within the constraint

box of the new vertex can form edges, the *CliqueCloak* algorithm can be optimized to collect all vertices that are within the constraint box and then test the reverse requirement on each of the vertices in this set. To implement this efficiently, the vertices can be indexed in a multi-dimensional index. Pseudo-code for this algorithm is given in Algorithm 1 of [6].

3.3.3 Analysis

Several methods to form cliques have been tested, each giving different results. The best performing method is called *nbr-k* and in this summarized overview we'll only look into the results by this method. We mention some of the interesting results; a more comprehensive overview of the results of these tests can be found in section 6 of [6], including a description of the test methodology.

In general, messages with a lower value for k can be anonymized easier than messages with a higher value for k. For the *nbr-k* method, success rate decreases from around 80% for messages with $k = 2$ to around 60% for messages with $k = 5$. Success means that a message has not been dropped before it expired.

One of the properties of the *CliqueCloak* algorithm is that each message can have a different value for k. This means that messages with different k can be combined in a single cloaking box. A measurement *relative anonymity* for an individual message has been introduced, which equals the ratio between the number of messages that are in the cloaking box and the value of k for this message, e.g. a relative anonymity value of 2 means that the number of messages in the cloaking box is twice the value of k. The location k-anonymity property ensures that the relative anonymity is at least 1 for each message.

For the *nbr-k* method, the average relative anonymity ranges from 1.7 for messages with $k = 2$ to 1.2 for messages with $k = 4$. For messages with $k = 5$, the relative anonymity is always 1, since the algorithm never searches for cliques larger than what is required by the highest k value in a subset and the tests didn't include messages with values for k above 5.

3.4 Peer-to-peer

In the previous algorithms, an intermediate server – or proxy – must be used to anonymize the location of users. In [7], Chow, et al. propose a peer to peer spatial cloaking algorithm, that doesn't require such a server to be in place. The idea is that before sending location information to the service provider, the requester will form a group of nodes and mask its location within this group. It uses the group to request information, therefore taking the responsibly of cloaking its location away form the usual central server system to the nodes themselves.

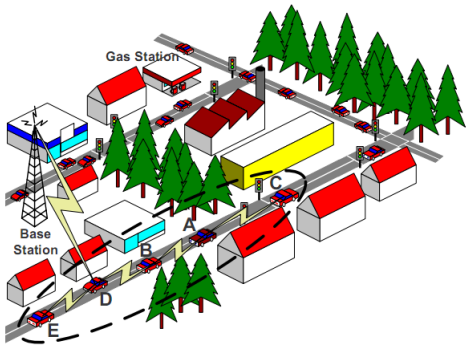

Figure 6: An example of how peer to peer spatial cloaking works

Lets start with a simple example. In figure 6, we can see 5 cars on the road. A wishes to find out where the nearest gas station is. A looks for other peers and finds B, C, D and E. A will then cloak its exact location by randomly selecting a peer, e.g. D, to communicate for the group; D is called the agent. This agent takes $A's$ request and forwards it onto the location-based database servers. Once the information is computed the agent D receives the information and forwards the reply to A.

This algorithm has two modes of use: *on-demand* and *procative*. In the *on-demand* mode, the cloaking algorithm is only used when information is required by one of the nodes, while *procative* mode means the nodes periodically seek other nodes to form a group with, just in case it or others wish to request information.

The system model is similar to the other algorithms above. A node or mobile client sends its location and its query to the location based Database servers via the base station. Each node has its own privacy profile. Each one of them can set its desired level of privacy. To do this 2 variables are set: k and A_{min}. k describes the level of k-anonymity it desires. A high k means a high privacy requirement. A_{min} is the minimum resolution of the cloaking spatial region. The higher A_{min}, the higher success of finding an appropriate group, but the lower the level of quality of the returned answer.

Each node has two communication devices attached to it. One is used to communicate with other peers and the other is used to communicate with the location based database servers. The method of communication can be varied, for example using Bluetooth, GSM or wireless LAN.

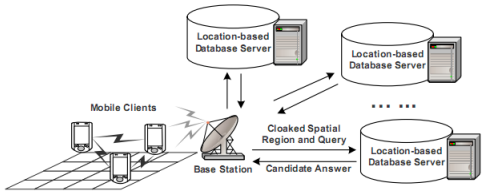

Figure 7: System model

3.4.1 Analysis

In experimental studies it has been shown that *proactive* mode outperforms the *on-demand* mode in terms of response time, since a group has already been formed at the moment the user initiates a request. However the higher overhead of using *proactive* mode when the system is not used means that there is a higher idle power consumption, which may not be acceptable for mobile devices.

4 Query processing

In section 3, several techniques were presented that can be used to hide the identification of individuals to a privacy-aware LBS and – more important for the topic of this section – to cloak the exact location of an individual. This means that the query processor – the component of an LBS that has the task to process queries presented by clients – has to cope with location uncertainty.

There are several kinds of queries that can be performed on location data. In [8], Mokbel, Chow and Aref present a method for processing location queries in a privacy-aware setting, that can be implemented in conjunction with existing geographic information systems.

The kind of query that is being processed by the explored algorithm is of the kind "Please, find me the location of the nearest object of type T," where T can be an object with a fixed location – like an ATM machine or a gas station – or a moving object – like a known friend or a police officer. It is even possible that the location of this object is stored as a privacy-aware region, as described in section 4.2. This kind of query is called a *nearest-neighbor query*.

Since the query processor is unaware of the exact location of the client, it can intuitively been determined that it might not always be possible to return a single correct answer. Given cloaked location data for the client, section 4.1 introduces an algorithm that returns a minimal set of candidate results of objects with a known exact location. Section 4.2 extends the algorithm to cope with locating the nearest neighbor among objects whose locations are also cloaked.

4.1 Private queries over public data

Section 5.1 of [8] describes an algorithm that can handle queries in this situation. When dealing with public data for objects of type T, the query processor has access to the full list of objects of that specific type, which are probably stored in a database: $t \in obj_T : \langle id, \{x, y\}\rangle$, identified by an object identifier id and a position (x, y).

Besides this list, the query processor obtains the (x_l, x_r, y_u, y_l) bounds of a rectangle that contains the position of the client the performs the request as input. Every point in this rectangle has an equal probability of being the location of the client. The output of the algorithm is the set of objects that can be the nearest neighbor for any client located inside the input rectangle. Formally, this can be described as the set of all objects $t \in obj_T$ for which $\exists(x, y)$ with $x_l \leq x \leq x_r \wedge y_l \leq y \leq y_u$ such that $\nexists t' \in obj_T : \mathrm{D}(\mathrm{P}(x, y), t') < \mathrm{D}(\mathrm{P}(x, y), t)$, with $\mathrm{D}(p_1, p_2)$ being the distance between points p_1 and p_2.

The algorithm consists of four steps to obtain the set of output objects:

Step 1: filter objects The cloaked region that contains the client is rectangle-shaped and therefore has four corners for which the algorithm knows the exact locations: $v_1 : (x_r, y_l)$, $v_2 : (x_l, y_l)$, $v_3 : (x_l, y_u)$ and $v_4 : (x_r, y_u)$. For each corner, the algorithm select the database object of type T that is closest to the corner. A set of at most four *filter objects* is constructed in this way.

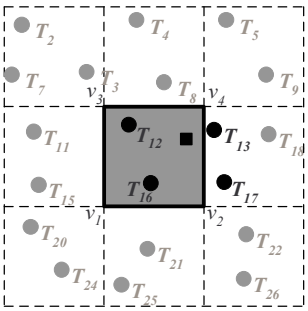

Figure 8: Result of step 1. T_{12}, T_{13}, T_{16} and T_{17} are the selected filter objects, as they are closest to the corner points of the cloaked region.

Step 2: middle points For each pair of corners that share a border of the cloaked region, the

algorithm picks the intersection of the perpendicular bisector of the associated filter objects with the border between the two region corners. If both corners have the same filter object, then the perpendicular bisector does not exist. This isn't a problem, as the single filter object will be the closest object for each point on the border.

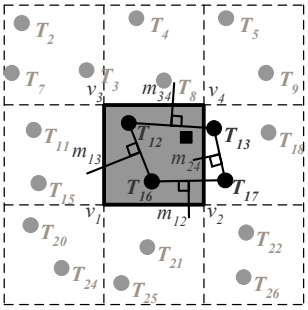

Figure 9: Result of step 2. m_{12}, m_{13}, m_{34} and m_{24} are the selected middle points.

Step 3: extended area For each border, the algorithm determines the distance from both corners to their associated filter objects and the distance from its midpoint object to the filter object that is closest to that point. In case the border share a single filter object, the last value is non-existent and therefore not taken into account. Among these two or three distances, the algorithm selects the maximum and extends the search area by moving the border outwards over this distance, forming a rectangle-shaped extended area A_{EXT}.

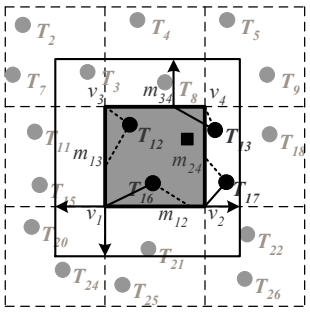

Figure 10: Result of step 3. The new border is the border of the extended area. The arrows that are drawn indicate the position of the point contributed to the distance of the extended area border.

Step 4: candidate list Given the extended area A_{EXT} – which by the proof in section 5.1.2 of [8] contains the nearest neighbor – the query processor selects all points the are contained in this area and returns that set to the client.

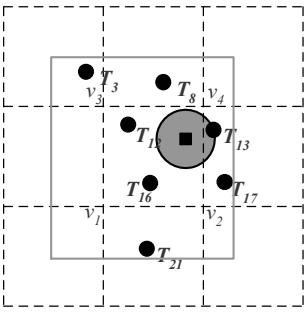

Figure 11: Result of step 4. All points that are within the extended area are selected and returned to the client. Since the client knows its exact location, it can select the nearest neighbor from this set.

4.2 Private queries over private data

One of the more interesting things that Mokbel, Chow and Aref present in [8] is an algorithm that allows not only the client that issues a query to present privacy-aware location data, but also allow the location of target objects in the database to have a cloaked location. Instead of a single point that represents the exact location of a target object, its location is stored as a rectangle-shaped area, with any point in that area having an equal probability of being the exact location of the object.

The algorithm for processing queries over private data follows the same steps as the one described in section 4.1 and is, in fact, a generalized version. For each step, one of the corners of the cloaked region is used as the input for the algorithm:

Step 1: filter objects When determining the nearest target object T for each of the corners $c_{u,1}$ to $c_{u,4}$ of the cloaked location of user u, the algorithm uses the corner $c_{T,i}$ of T that has the largest distance to $c_{u,i}$.

Step 2: middle points The method to define a middle point $m_{i,j}$ on the edge between corners $c_{u,i}$ and $c_{u,j}$ is the same a in the original algorithm, with the corners of the target object regions selected, such that the length of the line $L_{i,j}$ connecting the two points is maximal.

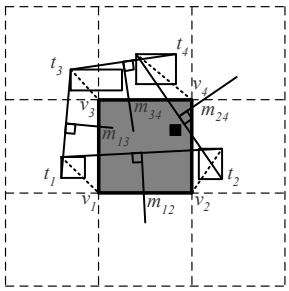

Figure 12: Result of steps 1 and 2. t_1 to t_4 are the location regions of the selected filter objects. m_{12}, m_{13}, m_{34} and m_{24} are the selected middle points.

Step 3: extended area To determine the distance that is used to move the borders to create the extended area, the maximum of two or three distances is used. For the distance between a corner $c_{u,i}$ and its filter object, the largest distance between the corner $c_{u,i}$ and any of the corners $c_{T,j}$ is used. The third distance used, is the one between the middle point $m_{i,j}$ and any of the endpoints of line $L_{i,j}$ – which give an equal distance, as point $m_{i,j}$ is on the perpendicular bisector between these points.

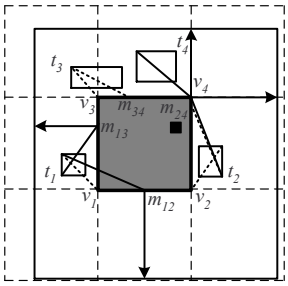

Figure 13: Result of step 3. The bold border is the border of the extended area.

Step 4: candidate list Given the extended area, all target objects T that have a cloaked area that overlaps with the extended area, have a probability of more than 0 of being the nearest-neighbor and can be returned to the user u as part of the candidate list.

4.3 Correctness

For each of the algorithms described in sections 4.1 and 4.2, two theorems are given in [8]:

Completeness Given a cloaked area A for a user u located anywhere within A, the algorithm returns a candidate list that includes the exact nearest target to u.

Minimum number of results Given a cloaked area A for a user u and a set of filter target objects t_1 to t_4, the algorithm issues the minimum possible range query to get the candidate list.

These two theorems can be written more formally as, where it should be noted that the phrasing "minimum possible range query" can be interpreted as requiring the A_{EXT} area to be rectangle-shaped, such that it can be expressed using bounds (x_l, x_r, y_u, y_l):

Completeness $\forall (x,y) \in A : (\exists t \in A_{EXT} : (\nexists t' \in obj_T : \mathrm{D}(t', \mathrm{P}(x,y)) < \mathrm{D}(t, \mathrm{P}(x,y))))$

Minimum number of results $\forall t \in A_{EXT} : (\exists (x,y) \in A : (\nexists t' \in obj_T : \mathrm{D}(t', (x,y)) < \mathrm{D}(t, \mathrm{P}(x,y))))$

Although [8] contains a proof for both theorems, which are correct for the interpreted requirement on the shape of A_{EXT}, we have actually determined that the proof for the *minimum result* theorem is incorrect if no restrictions on the shape of A_{EXT} apply. We can even provide a counter-example using the figures in section 4.1 to demonstrate that the theorem itself isn't valid without this restriction. Note that this restriction hasn't explicitly been stated in [8]. Although this counter-example invalidates the theorem, it's impact is only small and the presented algorithm is very useful nevertheless.

The example figures used in sections 4.1 already contain a target object T_{21} that can be used as a counter-example. The object lies within the extended area ($T_{21} \in A_{EXT}$), while it can never be the nearest object, since for all points (x,y) between v_1 and v_2, and therefore all points (x,y) in the dark area A, the distance to T_{16} is smaller than the distance to T_{21}: $\forall (x,y) \in A : \mathrm{D}(\mathrm{P}(x,y), T_{16}) < \mathrm{D}(\mathrm{P}(x,y), T_{21})$.

5 Social location privacy

The privacy enhancing technologies presented in the previous sections describe situations where a user wants to query a database containing location information about objects of a specific kind, i.e. the nearest neighbor query. In all cases – except for the peer-to-peer solution described in section 3.4 – there is an intermediate system that knows the

exact location of users and has the task of transforming messages in such a way that they can be relayed to an LBS provider without revealing user identification and exact location information.

Zhong, et al., take a different approach in [9], where they describe three protocols that are capable of conditionally revealing the exact location of two users of a system to each other, without revealing this information to a third party, like a social network provider. The three protocols take a different approach and therefore result in a different kind of information to be revealed, but have a common property that specific location information is only revealed when the distance between both users is within a specified range.

5.1 The Louis Protocol

The Louis protocol is described in section 4 of [9] and consists of two phases. In the first phase, Alice and Bob are able to learn whether the distance between them is within a certain range, while an optional run of phase two allows them to reveal their exact locations. The protocol makes use of a third party, which will not learn the locations of either Alice or Bob.

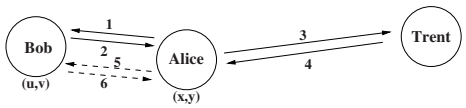

Figure 14: System model of the Louis protocol. Alice and Bob want to share location data, while Trent is a third party that won't learn anything about the location of Alice and Bob. The solid arrows indicate protocol messages for the first phase, the dashed arrows indicate message for the second phase.

Define (x, y) as the location of Alice and (u, v) as the location of Bob. The distance between both individuals is then given by $\sqrt{(x-u)^2 + (y-v)^2}$. Testing that this is below a certain maximum range r can be done efficiently by checking whether $d = (x-u)^2 + (y-v)^2 - r^2 < 0$. The goal of the first phase of the protocol is to perform this check, without revealing both (x, y) and (u, v) to a single party.

In the protocol description, $\mathcal{P}_{\mathrm{A}}(\cdot)$ is the Paillier encryption function [10] using the public key of Alice, $\mathcal{R}_{\mathrm{T}}(\cdot)$ is a public-key encryption function (say, RSA) using the public key of Trent, $\mathcal{H}(\cdot)$ is a cryptographic hash function, $\mathrm{sig}_{\mathrm{A}}(m)$ is a signature by Alice on message m and similarly $\mathrm{sig}_{\mathrm{T}}(m)$ is a signature by Trent. Note that the Paillier encryption function is non-deterministic and that is has the following homomorphic property: $\mathcal{P}_{\mathrm{A}}(m_1 + m_2) = \mathcal{P}_{\mathrm{A}}(m_1) \cdot \mathcal{P}_{\mathrm{A}}(m_1)$.

The messages below form the protocol; the first four message form phase one and the two remaining messages form phase two. Both the connection between Alice and Bob and the one between Alice and Trent are assumed to be confidential.

$A \to B$ $\mathcal{P}_{\mathrm{A}}(x^2 + y^2)$, $\mathcal{P}_{\mathrm{A}}(2x)$, $\mathcal{P}_{\mathrm{A}}(2y)$, r, $\mathcal{H}(x \| y \| s_A)$

$B \to A$ $\mathcal{P}_{\mathrm{A}}(d+k)$, $\mathcal{R}_{\mathrm{T}}(k)$, $\mathcal{H}(u \| v \| s_B)$, $\mathcal{H}(k)$

$A \to T$ $d+k$, $\mathcal{R}_{\mathrm{T}}(k)$, $\mathrm{sig}_{\mathrm{A}}(d+k)$, $\mathrm{sig}_{\mathrm{A}}(\mathcal{R}_{\mathrm{T}}(k))$

$T \to A$ answer, $\mathrm{sig}_{\mathrm{T}}(\text{answer} \,\|\, \mathrm{sig}_{\mathrm{A}}(d+k) \| \,\mathrm{sig}_{\mathrm{A}}(\mathcal{R}_{\mathrm{T}}(k)))$

$A \to B$ answer, $d+k$, $\mathrm{sig}_{\mathrm{A}}(d+k)$, $\mathrm{sig}_{\mathrm{A}}(\mathcal{R}_{\mathrm{T}}(k))$, $\mathrm{sig}_{\mathrm{T}}(\text{answer} \,\|\, \mathrm{sig}_{\mathrm{A}}(d+k) \| \,\mathrm{sig}_{\mathrm{A}}(\mathcal{R}_{\mathrm{T}}(k)))$, x, y, s_A

$B \to A$ u, v, s_B, k

s_A is a random salt chosen by Alice and s_B is one chosen by Bob; r is the desired maximum radius chosen by Alice; k is a random value selected by Bob. After receiving the first message, Bob can decide to abort the protocol if he doesn't like the value of r. Note that Bob is able to compute $\mathcal{P}_{\mathrm{A}}(d+k)$ using the homomorphic property of the Paillier encryption function, using the values $\mathcal{P}_{\mathrm{A}}(x^2+y^2)$, $\mathcal{P}_{\mathrm{A}}(2x)$ and $\mathcal{P}_{\mathrm{A}}(2y)$ he received from Alice and the values $\mathcal{P}_{\mathrm{A}}(u^2+v^2)$, $\mathcal{P}_{\mathrm{A}}(2u)$, $\mathcal{P}_{\mathrm{A}}(2v)$, $\mathcal{P}_{\mathrm{A}}(k)$ and $\mathcal{P}_{\mathrm{A}}(r^2)$ he computes himself:

$$\begin{aligned}
&\mathcal{P}_{\mathrm{A}}(d+k) \\
&= \mathcal{P}_{\mathrm{A}}((x-u)^2 + (y-v)^2 - r^2 + k) \\
&= \mathcal{P}_{\mathrm{A}}(x^2 + u^2 - 2xu + y^2 + v^2 - 2yv - r^2 + k) \\
&= \frac{\mathcal{P}_{\mathrm{A}}(x^2+y^2) \cdot \mathcal{P}_{\mathrm{A}}(u^2+v^2) \cdot \mathcal{P}_{\mathrm{A}}(k)}{(\mathcal{P}_{\mathrm{A}}(2x))^u \cdot (\mathcal{P}_{\mathrm{A}}(2y))^v \cdot \mathcal{P}_{\mathrm{A}}(r^2)}
\end{aligned}$$

After receiving the third message, Trent can decrypt $\mathcal{R}_{\mathrm{T}}(k)$ and subtract it from $d+k$ to obtain d. If $d < 0$, Trent sets the answer to 'YES'. Trent sets the answer to 'NO' otherwise. After receiving the forth message, Alice knows whether the distance to Bob is smaller than r and can decide whether or not to proceed with the second phase of the protocol.

In the second phase, Alice reveals her location to Bob and sends him all information that he needs to verify her location and the answer from Trent. Bob can now decide to reveal his location to Alice. By including his salt and random k, Alice can verify the protocol run.

5.1.1 Louis protocol analysis

Assuming an honest protocol run, after the first phase neither party will not learn the exact location of any of the other parties involved. Trent will learn nothing about the locations of Alice or Bob and he won't even learn about the distance between them. The only thing he will learn is the difference in distance and radius of the circle that is recognized as "nearby". Although Alice nor Bob will learn this difference in value, Trent does sent the sign of this value to Alice, so she'll learn whether the distance to Bob is above or below the radius r that she came up with. After the second phase, both Alice and Bob will learn each others exact locations, while Trent will not learn any new information.

When one or more of the parties involved in the protocol are dishonest, some interesting conclusions can be drawn. The hash values used in the message exchange ensure that both parties can verify that the locations revealed in the second phase are the same as those committed to in the first phase. However, the locations that are reported by Alice and Bob can in no way be verified by this protocol. Therefore, the easiest attacks possible by both Alice and Bob is to use incorrect location data during the entire protocol run. Such behavior defeats the purpose of this protocol and may damage the social relation between both parties.

One interesting attack Alice can perform is to execute the first phase of the protocol several times, using different faked values for her location. This way, she can probe Bob for being at a set of likely locations. Bob can prevent such an attack by refusing to take part of the protocol when it is invoked multiple times within a short timeframe.

The protocol has no way to prevent or detect Alice or Bob to collude with Trent. When Alice and Trent collude, they are able to determine the distance between Alice and Bob: $\sqrt{d+k-k-r^2}$. When Bob and Trent collude, they can in the same way determine the distance between Alice and Bob, but it's even more powerful to have Trent always return 'YES', in order to persuade Alice to reveal her exact location to Bob, after which Bob can abort the protocol.

5.2 The Lester Protocol

The Louis protocol described in section 5.1 requires a semi-trusted third party. Although this third party learns only the difference between the distance between Alice and Bob and the requested threshold radius, Alice and Bob may be interested in using a protocol that doesn't have a dependency on a third party. The Lester protocol presented in section 5 of [9] removes the need for the third party, at the cost of only disclosing the distance between the two users – instead of the exact location(s) – to Alice and disclosing nothing to Bob. It is noted, however, that the protocol can be run a second time with the roles reversed to perform a mutual exchange of information.

In the protocol description, $a \in \mathbb{Z}_q$ is the private key of Alice, b is the private key of Bob and $\mathcal{C}_\mathrm{A}(m) = (c_a, c_b) = (g^r \pmod p, A^{r+m} \pmod p)$ is the CGS97 encryption function [11], using a generator $g \in \mathbb{Z}_p$, a random value $r \in_R \mathbb{Z}_q$ and the public key of Alice $A = g^a$. Alice and Bob can both compute $C = A^b = B^a$, like Diffie-Hellman key exchange. Like the Paillier encryption function, the CGS97 encryption function is both non-deterministic and homomorphic: $\mathcal{C}_\mathrm{A}(m_1 + m_2) = (c_{1,a} \cdot c_{2,a}, c_{1,b} \cdot c_{2,b})$. CGS97 also has the property that decryption involves computing a discrete logarithm and therefore takes $O(\sqrt{M})$, where M is the number of possible plaintext messages, when using the Pollard lambda method described in [12].

As before, (x, y) defines the location of Alice and (u, v) the location of Bob. $D = (x-u)^2+(y-v)^2$ is the squared distance between Alice and Bob. The protocol requires two messages to be exchanged:

$$A \rightarrow B \;\; \mathcal{C}_\mathrm{A}(x^2+y^2), \mathcal{C}_\mathrm{A}(2x), \mathcal{C}_\mathrm{A}(2y)$$

$$B \rightarrow A \;\; t, \mathcal{C}_\mathrm{A}(b \cdot (D \cdot 2^t + s))$$

In the second message, Bob introduces a random salt s of length t. The length t determines the amount of work Alice has to do to successfully decrypt the message. Note that Bob is able to compute $\mathcal{C}_\mathrm{A}(b \cdot (D \cdot 2^t + s))$ using the homomorphic property of CGS97, using $\mathcal{C}_\mathrm{A}(2^t b \cdot D)$ and $\mathcal{C}_\mathrm{A}(b \cdot s)$ as input ciphertexts. Bob can compute $\mathcal{C}_\mathrm{A}(2^t b \cdot D)$ using the values $\mathcal{C}_\mathrm{A}(x^2+y^2)$, $\mathcal{C}_\mathrm{A}(2x)$ and $\mathcal{C}_\mathrm{A}(2y)$ he received from Alice and the values $\mathcal{C}_\mathrm{A}(u^2+v^2)$, $\mathcal{C}_\mathrm{A}(2u)$ and $\mathcal{C}_\mathrm{A}(2v)$ he computes himself. In the derivation below, the $\cdot$ operator is overloaded as pairwise multiplication operator when used with two CGS97 ciphertext as operands:

$$\begin{aligned} & \mathcal{C}_\mathrm{A}(2^t b \cdot ((x-u)^2 + (y-v)^2)) \\ = \mathcal{C}_\mathrm{A}(2^t b \cdot (x^2 + u^2 - 2xu + y^2 + v^2 - 2yv)) & \\ = \mathcal{C}_\mathrm{A}(x^2 + u^2 - 2xu + y^2 + v^2 - 2yv)^{2^t b} & \\ = \left(\frac{\mathcal{C}_\mathrm{A}(x^2+y^2) \cdot \mathcal{C}_\mathrm{A}(u^2+v^2)}{(\mathcal{C}_\mathrm{A}(2x))^u \cdot (\mathcal{C}_\mathrm{A}(2y))^v} \right)^{2^t b} & \end{aligned}$$

Using $\mathcal{C}_\mathrm{A}(b \cdot (D \cdot 2^t + s)) = (g^r, A^{r+(b \cdot (D \cdot 2^t + s))}) = (g^r, A^r A^{b \cdot (D 2^t + s)})$ and $A^r = g^{ar}$, Alice can compute $A^{b \cdot (D \cdot 2^t + s)} = C^{D \cdot 2^t + s}$ from the received message.

If Alice wants to determine whether the distance to Bob is less than some threshold r, she tries to solve the discrete logarithm by finding $D \cdot 2^t + s$ in the range $[0, r^2 2^t]$. Using the Pollard lambda method [12], this takes $O(\sqrt{r^2 2^t}) = O(2^{t/2} r)$ time. Notice that this is linear in the radius r chosen by Alice and exponential in the work factor t chosen by Bob.

If Alice succeeds in solving the discrete logarithm, she can obtain D from $D \cdot 2^t + s$ by a shifting the value by t bits (or performing an integer division by 2^t) and she can compute the distance $\hat{r} = \sqrt{D}$ to Bob. If Alice fails to solve the discrete logarithm, she can conclude that the distance $\hat{r}$ to Bob is larger than r.

5.2.1 Lester protocol analysis

In the Lester protocol, only Alice and Bob participate, so there is no third party Trent that can learn anything about the locations of Alice or Bob or can collude with any of them. Like the Louis protocol, the Lester protocol cannot enforce Alice and Bob to use their real location in the message exchange, but neglecting to do so may damage the social relation between them.

Regarding the protocol, it is interesting to note that it is difficult for Bob to determine a correct value for t. A value too high may prevent Alice from successfully learning the distance between her and Bob, while a value too low will allow Alice to determine the distance in a situation where it is larger than Bob is comfortable to reveal.

Since the time to solve the discrete logarithm is linear in the probed radius, Alice can easily double her effort in an attempt to double her search radius. Another effect is that Bob must know what kind of equipment Alice is using in order to determine a suitable value for t and failure to do so introduces a mismatch between his privacy requirements and the ability for Alice to find his location. For example, if Bob chooses a t that allows Alice to find him with a mobile device within a certain range, she can probably find him within the same timeframe in a much larger area, using her personal computer. We believe this critical property makes the Lester protocol unsuitable for real-world scenarios.

5.3 The Pierre Protocol

One disadvantage of the Lester protocol described in section 5.2 is that Alice can decide to do more work, to discover the distance between her and Bob for a longer distance. For example, if Bob decides on a certain factor that allows Alice to determine the distance within 500 meters, Alice can choose to do twice the amount of work and determine the distance between them, even when the distance is between 500 meters and one kilometer.

In section 6 of [9], the Pierre protocol is described that doesn't have this property and can therefore give Bob more confidence in his privacy. The downside is that even less information is disclosed using this protocol. Instead of a continuous coordinate system, the Pierre protocol assumes a grid. Therefore, the coordinates of Alice will become $(x_r, y_r) = (\lfloor \frac{x}{r} \rfloor, \lfloor \frac{y}{r} \rfloor)$, with r being the size of the grid cells.

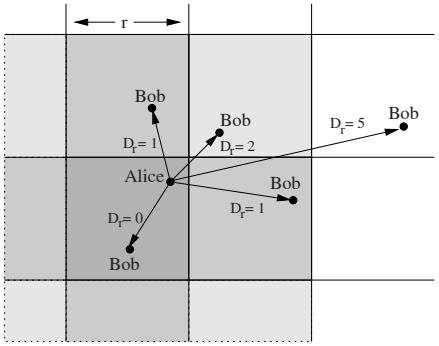

Figure 15: Grid coordinates used as in the Pierre protocol. Notice that the squared distance D_r equals 0 when Alice and Bob are in the same cell, equals 1 when they're in adjacent cells and equals 2 when they are in diagonally touching cells.

The protocol again relies on the use of a homomorphic encryption function and the authors of [9] state that both Paillier [10] and CGS97 [11] can be used. In the description below, the notation from section 5.2 – and therefore the CGS97 function – is used.

$$A \rightarrow B \;\; r, \mathcal{C}_{\mathrm{A}}(x_r^2 + y_r^2), \mathcal{C}_{\mathrm{A}}(2x_r), \mathcal{C}_{\mathrm{A}}(2y_r)$$

$$B \rightarrow A \;\; \mathcal{C}_{\mathrm{A}}(\rho_0 \cdot D_r), \mathcal{C}_{\mathrm{A}}(\rho_1 \cdot (D_r - 1)), \mathcal{C}_{\mathrm{A}}(\rho_2 \cdot (D_r - 2))$$

The values ρ_0, ρ_1 and ρ_2 are random elements of $\mathbb{Z}_p^*$ and $D_r = (x_r - u_r)^2 + (y_r - v_r)^2$ is the squared distance between Alice and Bob. Bob can compute the messages he must send, because the protocol uses a homomorphic encryption function, he learned $\mathcal{C}_{\mathrm{A}}(x_r^2 + y_r^2)$, $\mathcal{C}_{\mathrm{A}}(2x_r)$ and $\mathcal{C}_{\mathrm{A}}(2y_r)$ from Alice and can compute $\mathcal{C}_{\mathrm{A}}(u_r^2 + v_r^2)$, $\mathcal{C}_{\mathrm{A}}(2u)$ and $\mathcal{C}_{\mathrm{A}}(2v)$ himself:

$$\begin{aligned} & \mathcal{C}_{\mathrm{A}}(\rho_i \cdot ((x_r - u_r)^2 + (y_r - v_r)^2) - i)) \\ = & \frac{\mathcal{C}_{\mathrm{A}}(x_r^2 + u_r^2 - 2x_r u_r + y_r^2 + v_r^2 - 2y_r v_r)^{\rho_i}}{\mathcal{C}_{\mathrm{A}}(i\rho_i)} \\ = & \frac{\left(\frac{\mathcal{C}_{\mathrm{A}}(x_r^2 + y_r^2) \cdot \mathcal{C}_{\mathrm{A}}(u_r^2 + v_r^2)}{(\mathcal{C}_{\mathrm{A}}(2x_r))^{u_r} \cdot (\mathcal{C}_{\mathrm{A}}(2y_r))^{v_r}} \right)^{\rho_i}}{\mathcal{C}_{\mathrm{A}}(i\rho_i)} \end{aligned}$$

Alice can now check whether one of the three decrypted values equals zero: if Alice and Bob are in the same grid cell, $D_r = 0$ and therefore $\rho_0 \cdot D_r = 0$, if Alice and Bob are in adjacent cells, $D_r = 1$ and therefore $\rho_1 \cdot (D_r - 1) = 0$ and if Alice and Bob are in diagonally touching cells, $D_r = 2$ and therefore $\rho_2 \cdot (D_r - 2) = 0$. The authors note that checking whether the plaintext of $(c_1, c_2) = (g^r, A^{r+m})$ equals zero using CGS97 doesn't require decryption, since Alice can just check whether $c_1^a = c_2$.

5.3.1 Pierre protocol analysis

Like the Lester protocol, the Pierre protocol doesn't require a third party and is unable to verify that the location data used by the participants of the protocol is correct.

Since the three values returned by Bob include a random element, Alice is unable to obtain any information about the location of Bob relative to her own, unless one of the three messages decrypts to zero. There is no information that Bob learns about the location of Alice.

6 Conclusion

Regarding privacy in location based services, four main parts have been looked into in this paper: we did a small-scale interview to gauge the usage of LBS and privacy concerns of students, we discussed several algorithms that can cloak the location of a user using k-anonymity as main privacy constraint, we discussed an algorithm that allows an LBS provider to resolve nearest neighbor queries when the location of the user is cloaked and three protocols have been discussed that allow location information to be exchange in a social setting.

6.1 Interview

In our interview, we questioned 69 students of the Radboud University. The most interesting result of this interview is that students show concerns regarding privacy when using an LBS, but seem to rate this privacy less important than the right of government officials or an employer to ask people about their whereabouts. As this is contrary to what we expected, this can be an interesting topic to do further research. It should be noted that the group we questioned isn't a representation of society and more accurate results can only be obtained by redoing the interview.

For a previous version of this paper, we have been doing a different interview and we can conclude that the questions in this newer version have been more carefully designed. In the previous interview, we were asking people whether they know about the possible consequences of using an LBS, without explaining those. Therefore we were relying on the self-judgement of people to decide how well their knowledge about this topic is. In this newer version, we first used a question to make people agree on the meaning of an LBS by giving explicit devices and services to think about.

After establishing this agreement, the questions of section 3 were used to determine how concerned people are, without actually informing them about the dangers that researchers talk about, while the questions of section 4 were designed to gauge how concerned people really are about their privacy by framing an explicit scenario.

6.2 Discussion of protocols

Using k-anonymity to quantify the level of privacy a user needs, several protocols have been described that allow cloaking of the exact location of a user when using an LBS. We have seen a straightforward approach by Gruteser, et al. [5] which keeps splitting areas to the lowest possible size that meets the k-anonymity constraint. The disadvantage of this method is that a single value for k_{min} is used in the system and the fact that a fundamental flaw has been described that can breach privacy.

One attempt to overcome the limitation of using a single k_{min} value is the CliqueCloak algorithm by Gedik, et al. [6], allowing the sender of a message to specify a different k_{min} value for each individual message. Like the Gruteser algorithm, CliqueCloak relies on a central anonymizing proxy, which is a property that Chow, et al. have tried to overcome in a peer-to-peer algorithm [7].

Since the papers that introduce each algorithm don't use a common method to test the performance and efficiency, it is impossible to quantitatively compare them. However, the flaw in the Gedik algorithm has been replaced by a failure to send a message to the LBS in the other two algorithms. From a privacy point of view we believe this is a better approach, but users probably perceive this as a deficiency of a system implementing such approach.

We believe that the requirement of setting up peer-to-peer communications in the last algorithm may be difficult and cause suboptimal grouping as a result of peer communication that are blocked by obstacles and therefore give a slight preference for the CliqueCloak algorithm. Regarding this preference, we want to note that mobile phone carriers might be a suitable party to implement the anonymizing proxy, as they must already be aware

of the location of the client in order to provide cell routing and are already bound by law to keep this information private. However, proper comparative research should be conducted to make a definite decision.

In section 4 we described the query processing algorithm introduces by Mokbel, et al. [8], which can be used by a LBS provider to construct a list of target objects that can be returned to a client with a cloaked location as a result of a nearest neighbor query. The completeness property of this algorithm proves the usefulness of the algorithm, while the property that it returns the minimum number of result given a rectangular search area makes it suitable for use with mobile clients. Although we identify a minor flaw in this latter property, we believe this algorithm is very useful in an LBS setting.

Finally, we discussed three protocols introduced by Zhong, et al. [9] that can be used in a social setting. The Louis protocol makes use of a semi-trusted third party and includes message hashes that allow the participating parties to verify the committed location values after the protocol has finished. The Lester protocol removes the need for the third party, at the price of revealing less information to the participating parties. We believe the property of this protocol that finding the distance between the participating parties has a complexity linear to the search range to be a very important flaw in the protocol. Finally, the Pierre protocol again reveals less information, but has much stricter control over what information is revealed compared to the Lester protocol, while also not depending on a third party.

References

[1] Latanya Sweeney. k-anonymity: a model for protecting privacy. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 10:557–570, October 2002.

[2] Latanya Sweeney. Uniqueness of Simple Demographics in the U.S. Population. *LIDAP-WP4 Carnegie Mellon University, Laboratory for International Data Privacy, Pittsburgh, PA: 2000*, 1000.

[3] Group Insurance Commission. Testimony before the massachusetts health care committe, 1997. See *Session of the Joint Committee on Health Care*.

[4] City of Cambridge Massachusetts. Cambridge voters list database, Cambridge: February 1997.

[5] Marco Gruteser and Dirk Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *Proceedings of the 1st international conference on Mobile systems, applications and services*, MobiSys '03, pages 31–42, New York, NY, USA, 2003. ACM.

[6] Bugra Gedik and Ling Liu. A customizable k-anonymity model for protecting location privacy. In *In ICDCS*, pages 620–629, 2004.

[7] Chi yin Chow. A peer-to-peer spatial cloaking algorithm for anonymous location-based services. In *In: ACM GIS. (2006)*, pages 171–178. ACM Press, 2006.

[8] Mohamed F. Mokbel, Chi yin Chow, and Walid G. Aref. The new casper: Query processing for location services without compromising privacy. In *In VLDB*, pages 763–774, 2006.

[9] Ge Zhong, Ian Goldberg, and Urs Hengarter. Louis, lester and pierre: Three protocols for location privacy. In *Privacy Enhancing Technologies*, pages 62–76, 2007.

[10] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Advances in Cryptology — EUROCRYPT '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 223–238. Springer Berlin / Heidelberg, 1999.

[11] Ronald Cramer, Rosario Gennaro, and Berry Schoenmakers. A secure and optimally efficient multi-authority election scheme. In *Advances in Cryptology — EUROCRYPT '97*, volume 1233 of *Lecture Notes in Computer Science*, pages 103–118. Springer Berlin / Heidelberg, 1997.

[12] J. M. Pollard. Monte carlo methods for index computation. *Mathematics of Computation*, 32(143):918–924, 1978.