Mobile Systems Location Privacy: "MobiPriv" A Robust K Anonymous System.

Leon Stenneth
Dept. of Computer Science
University of Illinois, Chicago
Chicago, USA
lstenn2@uic.edu

Phillip S. Yu
Dept. of Computer Science
University of Illinois, Chicago
Chicago, USA
psyu@cs.uic.edu

Ouri Wolfson
Dept. of Computer Science
University of Illinois, Chicago
Chicago, USA
wolfson@cs.uic.edu

ABSTRACT

With the rapid advancement of positioning and tracking capabilities (mobile phones, on-board navigation systems) location based services are rapidly increasing. Privacy in location based systems is addressed in many papers. Our work is focused on the trusted third party privacy framework that utilizes the concept of k-anonymity with or without l-diversity. In previous anonymization models k may be defined as a personalization parameter of the mobile user or as uniform system parameter for all mobile users . Clearly, k other users may not be available at the time of request in these systems. These requests are discarded because the quality of service (QoS) they require cannot be satisfied. In this paper we introduce a novel suite of algorithms called MobiPriv that guarantees a 100% success rate of processing a mobile request using k-anonymity with diversity considerations. We evaluated our suite of algorithms experimentally against previously proposed anonymization algorithms using real world traffic volume data, real world road network and mobile users generated realistically by a mobile object generator.

Keywords – Location Based Services; k-anonymity; l-Diversity; Privacy; Mobility

1. INTRODUCTION

Location based services (LBS) are applications that take the geographic location into consideration. LBS is enhanced by the rapid improvement of the mobile phone capabilities such as GPS and multimedia. Example of location based services are TransitGenie[20], NextBus [22],Google Latitude[5].

In general a user submit a request to some database or location based server and receives a response. A typical request from a user include some location criteria and may be in the the form of *<id,time, location(x,y),query> Example: "What is the fastest path from my current location (latitude,longitude) to Navy Pier Chicago, Illinois (latitude,longitude)*".

With untrusted servers the privacy and security of an individual may be leaked to adversaries. Several reports are available where GPS devices were used to stalk user locations [23,24]. A knowledge of location may reveal a person's political, religious or medical affiliations. A knowledge of location may also lead to tracking or unwanted advertisements sent to your mobile device.

There are several architectures that are considered for privacy aware location based services. These architectures are *client-server, trusted third party,* and *peer based.*

In the strict client server architecture clients communicate directly with the LBS by submitting a request, the LBS then returns a response directly to the client [11,12,13,14]. In the peer based model clients communicate directly with each other [21]. The intention of the clients is to cloak with each other in order to satisfy the k-anonymous principle.

The trusted third party model utilizes the concept of a middle-ware between the mobile user and the LBS. We sometimes refer to the middle-ware as *anonymization server* or *AS*. Mobile requests are first sent to the middle-ware,the request is then cloaked into a region with the spatial and temporal tolerance, we refer to this as a *region request*. The request is then *cloaked* with other users region request in close proximity, we refer to it as an *aggregate region request*. These anonymization (middle-ware) systems are already deployed publicly [25]. Our work is focused on the trusted third party architecture.

Location privacy in location based system is to prevent adversaries from learning a mobile user past or current locations and the time the locations where visited. We used the concept of *cloaking* for location protection.

We also define *request linking protection* as preventing an adversary from knowing the mobile user that submitted a request. We used the concept of k-anonymity [9,8] and l-diversity [7,26] to prevent *request linking*.

Mobile users in these frameworks are considered k-anonymous if a mobile user cannot be distinguished from at least k-1 other mobile users in the same region request. The concept of l-diversity ensures that the queries in region request are not homogenous. Recall that a *region request* consist of a cloaked area containing multiple mobile requests sent from the middle-ware to the LBS.

The remainder of the paper is organized as follows. Section 2 addresses the preliminaries of our framework. Section 3 presents an overview of MobiPriv and Section 4 highlights the experimental evaluations. Finally, section 5 and section 6 discusses the related work and the conclusion respectively.

2. PRELIMINARIES

In this section we discuss our motivation, architecture,

attack model and main concepts.

2.1 Motivation

In this paper we tackle privacy in location based services by preventing adversaries from being aware of the current or past mobile user location as well as linking a specific request or response to a mobile user.

Current k-anonymous techniques that uses the AS concept [7,8,15,19] will under-perform if k-1 other users are not available at the time of request. These AS systems allow the user to define their own personal level of privacy by specifying the *k* in k-anonymity.

The anonymization engine has two options if at the time of request *k-1* users are not present: (1)Wait until *k-1* other requests are made in the same region [8]. (2) Expand the region in search for *k-1* other requests [19,15] or continue to divide a large region until the region contains at most k-1 other users [15].

If after waiting for *k-1* other users or region expansion to find *k-1* other requests, if the AS still cannot satisfy k then the request is culminated.

Clearly dropping user request because *k-1* other users are not around will appraise the system as unreliable. Also, some LBS systems cannot tolerate temporal delay and region expansion compromises the accuracy of the response.

2.2 Architecture

Our work is focused on the trusted third party architecture and is the first work to consider realistic diverse dummy generation on this type of architecture.

The users first send the requests to *AS,* the AS then remove or pseudo-generate the identification field and also cloak the client location point into a region containing *k-1* other users. One novelty about our approach is if *k-1* other mobile user request cannot be found we generate realistic diverse dummies instead of dropping the query. The AS then forwards the aggregate region request to the LBS.

The LBS processes the query and send a response. This response is generic and should be filtered to get precise results. Filtering can be done on the AS or by the mobile client. A diagram depicting our architecture is shown below in Figure 1.

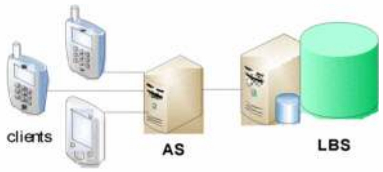


Figure 1 – LBS with Anonymization Server (AS)

2.3 Attack Model and Assumptions

We assume an environment containing mobile users with positioning capabilities, location based services (LBS), wireless networks and our algorithms running on a middle-ware that we called an anonymization server (AS). Mobile users submit a request incorporating positioning information such as current location (latitude,longitude) as a parameter of the request to the AS.

We assume that adversaries loiter between the anonymization server and the LBS or adversaries directly aims for the LBS. We also assume that adversaries know the exact location of some mobile users along with the time they submitted the request. This assumption is realistic as adversaries may have background knowledge of the victim in real life. Adversaries are also aware of the hashed identification of all the mobile users in a *region request.*

The first level of protection we provide is securely hashing the user identification and message identification before submitting the region request. We refer to this as *weak anonymization* since it is not enough to deter adversaries. Even if a user does not disclose their identity at a location, an adversary may still learn this information through spatial and temporal inferences.

Consider a transit itinerary system such as TransitGenie [20] or NextBus[22] where a user may query for current buses in their vicinity. By sending this request a user location is revealed to the LBS. Knowing a sender's location may lead to spam advertisements send to your device also, religious affiliations and hotel visits may be revealed. Furthermore, if continuous queries are sent a user may be tracked.

Location privacy is the prevention of adversaries from knowing a user's current or past locations. To protect location privacy we perturb a point request submitted by the user into a region request. In this way an adversary cannot know where exactly in the region the sender of the request is located.

Clearly, if the sender is the only person in the perturbed region the adversary may infer with high confidence that the request belongs to the sender. We called this a *request linking attack.* In order to prevent linking a request to the user the concept of aggregating *k-1* other users is the same region is used. We refer to this as *location k-anonymity* [10].

Observe also that if all the *k* users in the region submit request to the same symbolic address such as the same movie theater, k-anonymity fails because the adversary may infer that some persons are interested in that symbolic address. This kind of attack is referred to as the *homogeneity attack*. For this reason the concept of *l*-diversity is considered [26]. L diversity adds another dimension to the privacy level and ensures that our query location and query contents are diversified. For example, request should span across different postal addresses or different buildings. We generate diverse dummies and users are not allowed to define their own value of *l*.

A *corollary history attack* is possible when the adversary is aware of all the hashed identification in a region request sent by the AS to the LBS and also a knowledge of the regions that a user may send requests from. Existing methods using k-anonymity on the AS generate one cloaking region per location. User *u* may be distinct in

some regions. By doing a user intersection of all the regions u may be identified.

Let Rx be a region called x and u_y be a user y. Let $Rx = (u_w\ u_x\ u_y\ u_z)$ be a query sent from users w,x,y,z from Region x.

If $R1 = (u_1,u_2,u_3,u_4)$, $R2 = (u_1,u_{12},u_{13},u_6)$, $R3 = (u_1,u_8,u_0,u_{14})$. If an adversary has background knowledge that *Alice* queried in all three regions then by taking the intersection $(R1 \wedge R2 \wedge R3)$ then u_1 is revealed. The adversary may claim with high confidence that u_1 is Alice with the real identification weakly anonymized.

To the best of our knowledge our system is the only system that protects against the corollary history attack in an AS architecture that uses the k-anonymous technique.

2.4 Dummies

We define the term dummy to be a fake user U_a automatically and realistically generated by the system.

Dummies should be generated in a way that an adversary cannot differentiate a dummy from a real user U_r. In LBS systems such as road navigation systems users send continuous position queries. If dummies are generated randomly then adversaries can easily find the difference between the real user and the dummies. In our algorithm the dummies are generated relative to the temporal and spatial property of the real user request. The identification number of the realistic dummies are taken from the dummy profile. Our dummies request are also *diverse* and is different from the mobile user request.

Our work is not the first work to use dummies as a concept to increase privacy in location based systems. However, it is the first work to include dummy user generation on the anonymization server (AS). *Kido et al* [12] introduced the concept of dummy location generation in location based systems. In [12] the client sends the true position along with the dummy positions to the LBS. The LBS then respond with answers to both the true position and the dummy position. Our work is different as [12] did not consider the anonymization server instead only considered the client server architecture. We also considered dummy user generation and not dummy location generation as described in [12]. The disadvantages of the approach in [12] is the high processing cost on the mobile device to filter the results that were returned by the LBS. The excess filtering cost on the mobile client will negatively affect the short battery life and limited processing power of hand-held mobile devices. Also, there is a very high communication cost between the LBS and the client. If an adversary has a knowledge of history then a user may be re-identified in [12] by taking the intersection of all the regions that an adversary is sure that the single user sent a request from.

The work done by *You et al* in [13] considered generating dummy trajectories to prevent real trajectories from being identified. They concentrated on dummy trajectories and our focus is dummy mobile users with diverse request. Also, the work in [13] focused on the client-server based systems where the anonymization server is not present.

Dummy profile: The dummy profile is a file containing a list of all mobile users in the system along with corresponding dummy user identification numbers. We define *profileCount* to be the number of dummies associated with a real mobile user on the dummy profile. We initialize *profileCount* to be the maximum k value. We build the dummy profile as follows:

For each possible real user U_r in the system we associate a set of dummies with the real user id. The amount of dummies associated with each real user is specified by profile count. When we need to generate a dummy for a particular user we consult the dummy profile and take the dummies in topological order.

Our work is the first to consider realistic dummy user generation on the anonymization server in-order to guarantee accuracy and real time results to the user. We present our dummy generation algorithm.

Algorithm 1: Realistic-Dummy-Generation

```
1. precondition: mobileUserId!=null, totalDummies>0
2. input: mobileUserId, totalDummies,C /* C is query*/
3.method:
4.   profileCount ←50, count ← 0 ,offset
5.   /* reading the dummy profile*/
6.   profile = read_dummy_profile(mobileUserId,totalDummies)
7.    if(totalDummies <= profileCount)
8.      while(count<totalDummies)
9.            t =  getRequest().getT()+Math.random()+offset
10.           x = getRequest().getX()+Math.random()+offset
11.           y = getRequest().getY()+Math.random()+offset
12.           id =  profile.nextID()
13            C' = diversify(C) /* diversifying the dummy query */
14.           newDummy = createDummy(id,x,y,t,C')
15.           dummyMap.put(newDummy)
16.           count++
17.     end while
18. end if
19. else
20.         return
21. end else
end
```

In line 4 we declared the dummy profile to be 50, likewise in the experimental evaluation. In line 6 we read the dummy profile for the mobile user by user passing the mobile user identification and the total number of dummies required as parameters. In line 7- 18 we are guaranteed that the total number of required dummies is less than the profile count and we generate all the dummy identification from the dummy profile (line 12). Likewise the temporal and spatial properties of the dummies are related to the mobile user (line 9,10,11). In line 13 we *diversify* the query to prevent the *homogeneity attack* as defined in the attack model. We ensure that the dummies query point is not the same as the real mobile user query point instead a different building or symbolic address in the region is utilized. The new query is C'. We then create the new dummy and record it in memory (line 14).

2.5 Corollary History Attack Protection

Current anonymization techniques that involve a trusted third party AS[8,15,19,26] cannot protect against the corollary history attack. In these models when a user u

submits a query to the AS there is a search for k-1 other users in the same cloaking region. If the k-1 other users are not found immediately then the CR is expanded with the intention to locate these users. There may also be a delay until k-1 other users submit requests from the same CR.

After the discovery of *k-1* other users the AS sends a region request to the LBS. If *u* moves to another location or region and submits another query *k-1* other users have to be rediscovered again. However, the *k-1* users in the latter query may be a totally different set from the former *k-1* users. Taking the intersection of the latter and the former region enables the identification of *u* in such systems. Furthermore, rediscovery of the same users at all request time reduces the QoS of the results.

Our approach

Let U be the set of real users in the system. For each real user $U_r \in U$ we maintain a dummy profile. This dummy profile associates U_r with a set of *profileCount* dummy users A_r.

When a user submits the required privacy level then we associate this privacy level with a value of k in k-anonymity. If k < *profileCount* we generate k-1 dummies such that the k-1 dummies $\subset A_r$. We discuss an example below:

Assume a user U_r submitted a query to the AS with a privacy requirement corresponding to k=5. Let the privacy profile (dummy profile) of U_r be the set $A_r = \{U_{a1}, U_{a2}, U_{a3}, U_{a4}, U_{a5}, U_{a6}, U_{a7}, U_{a8}\}$ where *profileCount*=8. In the first cloaking region CR_1 we maintain the set $\{U_r, U_{a1}, U_{a2}, U_{a3}, U_{a4}\}$.

Assume U_r submits another query in CR_2 with a privacy requirement corresponding to k=6 we maintain the set $\{U_r, U_{a1}, U_{a2}, U_{a3}, U_{a4}, U_{a5}\}$. If U_r submits another query in CR_3 with a privacy requirement corresponding to k=7 we maintain the set $\{U_r, U_{a1}, U_{a2}, U_{a3}, U_{a4}, U_{a5}, U_{a6}\}$.

Observe that $CR_1 \cap CR_2 \cap CR_3 = \{U_r, U_{a1}, U_{a2}, U_{a3}, U_{a4}\}$. This means the user has a 1/5 chance of been identified. Clearly, there is a correspondence between the lowest value of k specified by the user and the chance of been identified in a corollary history attack.

Protection against the corollary history attack via dummy profile increases the cloaking time for the algorithm. We evaluated the effect on cloaking time in Figure 7(a).

3. MOBIPRIV: SYSTEM OVERVIEW

MobiPriv promotes a three tier model similar to the trusted third party mechanism discussed in section 2.2. Our suite of algorithms runs on the anonymization server. The first step is a *request submission* phase where the user submits a request. The request contains the percentage privacy level required by the user. Next is the *transformation* phase where the user percentage of privacy is converted to some value of *k* by a *mapping function.*

Other phases include the *perturb* phase whereby we form a region based on the spatial resolution from the request. The real user is then placed in the region. Next we have two options depending on the algorithm. One option is *CloakLessK* and the other is *CloakedK*.

In *CloakLessK* once we get a mobile user request we quickly generate k-1 dummies in the perturbed region and send the request to the LBS.

In *CloakedK* once we receive a request before we generate dummies we verify if any other request are close by and can be cloaked together. Finally if after cloaking with neighbors k-1 real mobile users are still not in the region we add realistic dummies as the remaining users and send the aggregate request to the LBS.

The LBS then processes the query and sends back a response. This response should be filtered in order to get a precise results. Filtering may be done on the AS or on the mobile client. Processing the results on the AS then sending the results to the clients may be more suitable in a real time mobile environment.

Our anonymizer maintains the following contributions:

1. All queries are given a response. In previous models some queries may not be given an answer because *k-1* users are not available to meet the QoS required or are not available in the system. If query request cannot be satisfied they are dropped from the system. In our experiments s*uccess rate* is used as a matrices to measure reliability.
2. Elimination of the temporal cloaking problem present in [8,15,19] whereby we wait for k-1 other users to be available.
3. Elimination of the spatial cloaking problem whereby we extend the region to find k-1 other users.
4. Communication Cost Reduction – We present two models *CloakLessK* and *CloakedK*. *CloakLessK* has a fixed communication cost regardless of the number of mobile users submitting request. *CloakedK* has a much lower communication cost, and the communication cost of *CloakedK* improves much more than *CloakLessK* as the number of request increases.
5. Corollary history attack protection.

Request

In *MobiPriv* the user submits a *request* in the form *<user_id,msg_num, {t,x,y},{dx,dy,dt},P,C>*. We define *user_id* to be a unique identification of the user and *msg_num* to be the message identification. The combination of *user_id* and *msg_id* is unique for all messages. Also, *{t,x,y}* is the temporal and spatial property of the request, *dx,dy,dt* is the spatial and temporal resolution demanded by the mobile user. *P* is the percentage of privacy desired and *C* is the request content. Large spatial tolerances produces less accurate responses and high temporal tolerances will result in longer message delays.

The point request above is converted to a region request of the form *<user_id',msg_num',{x1,x2},{y1,y2},{t1,t2}C>* where *user_id',msg_num'* are hashed versions of *user_id, and msg_num* respectively. The parameters *x1,x2* are the request region *x* coordinates, *y1,y2* are the request region *y* coordinates, *t1* and *t2* represents the request region *z*

coordinates. The three coordinates are used to form the cloaking box. *C* is the request content that should always be preserved.
Request example:
<user_101,msg_num_004, {11:15am,-87.653,41.85}, {50m, 50m, 5s}, 90% Privacy,"Shortest route from current location(-87.653,41.85) to Navy Pier (87.6215,41.210)" >

Transformation and Mapping function
We give users the opportunity to define a percentage value of accuracy they desire and use mapping function to determine a suitable value of *k*. We refer to this as *transformation*. AS administrators are not limited to one mapping function instead they may define there own mapping function. A mapping function may reflect the nature of the underlying location based system.
For our algorithm we consider a mapping such that percentage privacy is related to the certainty with which an association between a user and a message can be ascertained.
Let *n* be the total number of users that should be in the cloaking region to guarantee *P%* privacy.
Let *P* be the percentage of privacy desired by a real user U_r.:We define *k= ceiling (100/(100-P))*.

3.1 Algorithms

Below we discuss our two algorithms *CloakLessK* and *CloakedK*.

CloakLessK
First, the mobile user submits a request to the anonymization server (line 2). In line 4 we hash the user identification and the message identification. Each request contains as one of its parameters a privacy percentage that is converted to some *k*. We refer to this step as the transformation and mapping phase (line 5). We then create regions depending on *dx, dy, dt* (spatial and temporal tolerance) of the user request. The total number of users (real user and dummy users) in the region is determined by *k* from the *transformation* phase.
The real user is randomly placed in the grid (line 7) and then we generate *k-1* dummies and add them to the region (line 8). The dummies generated can be realistic and diverse or unrealistic dummies. Realistic diverse dummies are used to protect against the corollary history attack. However, this protection comes with a cost (see Figure 7a). Finally, we send the cloaked region request to the LBS for processing in line 9. Our experiments reveal that this algorithm has higher communication cost than *CloakedK*.

Algorithm 2: CloakLessK
1. **pre-condition**: *request!null*
2. **input:** *request <u_id,msg_num {t,x,y},{dx,dy,dt},P,C>*
3. **method**:
4. *hash (u_id,,msg_num)*
5. *transformation (P)*
6. *createGrid(request.dx,request.dy,request.dt)*
7. *insertRealUser()*
8. *insertDummies (request.u_id,equest.K-1)*
9. *sendRegionRequestToLBS()*
10. **end**

We next discuss *CloakedK*, an algorithm that reduces the communication cost.

CloakedK
The principal difference between *CloakLessK* and *CloakedK* is the addition of line 8 in the algorithm shown below. At line 8 of the *CloakedK* algorithm we perform cloaking in-order to reduce the communication cost. Instead of sending one real mobile user along with dummies in a region request transferred to the LBS the AS now aggregate multiple real users in the same cloaking region before sending to the LBS.
We now discuss the cloaking methodology of line 8. Once a user submit a request to the LBS we generate a perturbed box as discussed in section 3.1. The size of the box is dependent on *dx, dy, dt*. Then we place the user in the perturbed box. Before inserting any dummies inside the box, we query if this box is intersectable with other request boxes currently in the system. Two boxes are intersectable if and only if they overlap to include the request points. We then take a constraint on box size. We continue until the request for *k* is satisfied, if the request for *k* is still not satisfied we then generate dummies as the remaining mobile users (line 9) . Finally we send the request to the LBS. The selection of the box size is not discussed due to space constraints.

Algorithm 3: CloakedK
1. **pre-condition**: *request!null*
2. **input:** *request <u_id,msg_num {t,x,y},{dx,dy},P,C>*
3. **method**:
4. *hash (u_id,,msg_num)*
5. *transformation(P)*
6. *createGrid(request.K,request.dx,request.dy)*
7. *insertRealUser()*
8. *intersectAndMerge() /* cloak */*
9. *insertDummies()*
10. *sendRegionRequestToLBS()*
11. **end**

4. EVALUATION

We performed experimental evaluations of our algorithms (*CloakLessK* and *CloakedK*) against the three PrivacyGrid approaches (*Bottom Up, Top Down, Hybrid*) [15] and also against the *pyramid* based approach such as Casper [19].
First, we briefly discuss the three PrivacyGrid techniques [15]. In the *bottom-up* cloaking the mobile user cell is expanded to meet the k-anonymity and l-diversity requirement. Mobile users are considered for k-anonymity count and static objects eg. gas stations, supermarkets are used for the l-diversity count. For a given cell it may expand to its immediate neighbor east,west, north or south. The next cell to be chosen is the cell with the highest mobile user count. This cell will be included in cloaking box. The *top-down* approach selects the largest possible cloaking box first that satisfies the users QoS requirement. If *k* users cannot be found in this cloaking box the query cannot be satisfied. If *k* users are found then the algorithm will try to prune the cloaking box to see if *k* can still be satisfied. The *hybrid* approach makes a decision to use the

bottom up or top down depending on the value of k and the QoS. Hybrid combines the strength of both approaches.
We now discuss briefly the pyramid based scheme[19]. For anonymization and cloaking a pyramid structure is maintained. The cells in the region contain the number of mobile users present in the cell. If the current region/cell of user cannot satisfy the value of k then neighboring cells are considered. A cell is considered a neighbor if they have the same parent. If expansion to neighbors does not satisfy the user requirements then the parent expansion is considered. One can envision a balance quad-tree where the users are the leaves in the system.

4.1 Evaluation Criteria

In this section we discuss the evaluation criteria that we used to measure the efficiency of the algorithms.
One of the most important evaluation criteria is *success rate*. The main goal of any anonymization server is to maximize the number of messages that can be perturbed successfully within the restricted quality of service. We measure the *success rate* as the ratio of the number of anonymized request by the total number of individual mobile request. A success rate of 100% implies that all the requests that are sent by the mobile clients are anonymized.
Another evaluation criterion is the *cloaking time*. The cloaking time of an algorithm is the time taken to perturbed the requests. An algorithm with a lower cloaking time does better because the cloaking time is a measure of the temporal complexity. Cloaking time is a performance measure.
The third evaluation considered is the communication cost, which is the algorithms cloaking ability. Communication cost may be seen as the number of region request sent to the LBS by the AS for a fixed amount of individual request. Some other QoS evaluation variables considered are *spatial tolerance* and *anonymity level*. Spatial tolerance is the user defined spatial resolution that should be respected when satisfying the k-anonymity requirement. The anonymity level is the user defined k-anonymity requirement.

4.2 Experimental Setup and Road Network

The experiments where conducted on a HP Notebook PC running Windows Vista containing a P8400 Intel DUO 2.27 GHz processor with 4GB RAM. The algorithms (Bottom up PrivacyGrid, Top down PrivacyGrid, Hybrid PrivacyGrid, Casper Pyramid approach, MobiPriv CloakLessK, MobiPriv CloakedK) where implemented using Java and the development environment was Eclipse Platform version: 3.4.1. We refer to these algorithms as *B*, *T*, *H*, *Py*, *CLK*,*CK* respectively in the experiments. The mobile object generator that we used is an extension of mobile object generator used in [8,15]. We used a map of Chamblee in the state of Georgia, USA for our experiments. The map covers a region of 160 km^2 see Figure 2.We generate traces based on real world traffic volume data extracted from [10] for 10,000 cars on the road graph. Three types of roads are considered in the simulation; expressway (black), arterial (red) and collector roads (gray) see Figure 2. Cars are placed randomly on the road network initially and continues to move along a road trajectory making a decision at each intersection. The property of each road is shown in Table 1.
Each car (mobile user) generates multiple requests to the anonymization server.

TABLE 1

Properties	**Road categories**		
	Expressway	***Arterial***	***Collector***
Mean speed (km/h)	90	60	50
Std. Dev (km/h)	20	15	10
Traffic volume (cars/h)	2916.6	916.6	250

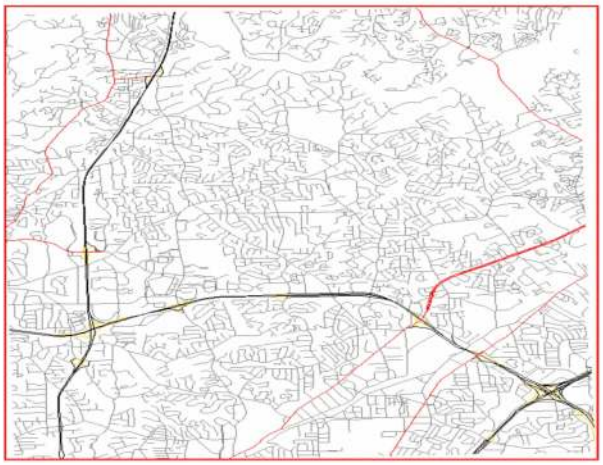
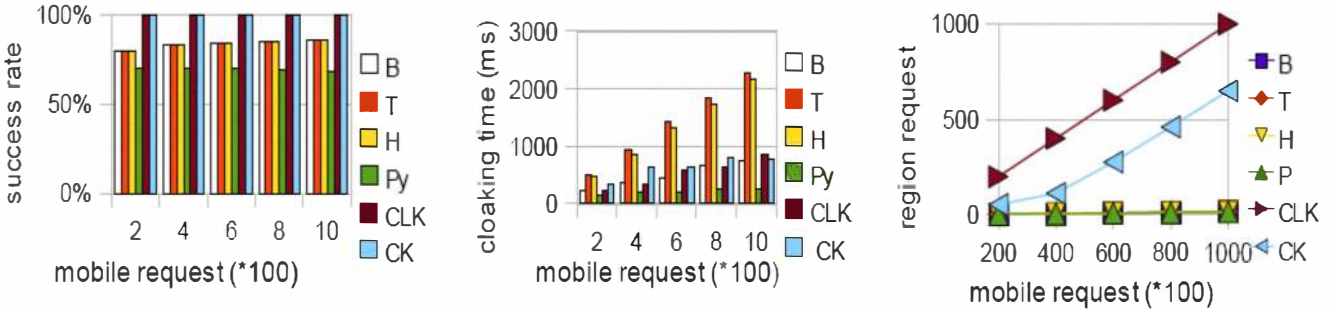


Figure 2 : Evaluation area (Map of Chamblee Region, Georgia, USA)

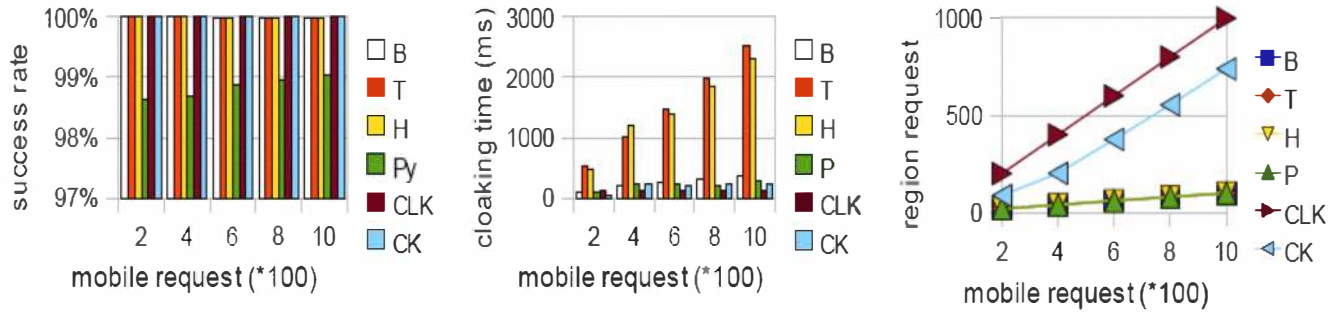
4.3 Results

In Figure 3 (a,b,c) we evaluated the success rate, cloaking time and communication cost with *high spatial tolerance* (700m*700m) and *high anonymity level* (k=50).
Figure 3(a) plots the success rate with varying number of mobile requests sent to the anonymization server. We saw that *MobiPriv* algorithms achieved 100% success rate, this means that all requests sent can be safely anonymized with the *MobiPriv* algorithms. The PrivacyGrid (top down, bottom up, hybrid) and Pyramid approaches where only able to anonymize 70%-80% of the requests. PrivacyGrid and Pyramid schemes will drop some of the requests because the QoS demanded by the request cannot be satisfied.
Figure 3(b) shows the cloaking time with different number of requests sent to the AS. *MobiPriv* algorithms achieved fast cloaking time. In particular *CloakLessK* has a lower cloaking time than *CloakedK*. The fastest cloaking time under these settings is the pyramid based. Also, MobiPriv algorithm's cloaking time are hardly affected by an increase in the number of requests .



(a) (b) (c)

Figure 3 – Results with high anonymity level (k=50) and high spatial tolerance (700m*700m)



(a) (b) (c)

Figure 4 - Results with low anonymity level (k=10) and high spatial tolerance (700m*700m)

In Figure 3(c) depicts the communication cost against the number of requests. We saw that MobiPriv algorithms have a higher communication cost. CloakedK does better than CloakLessK in terms of communication cost and continues to do much better as the number of request increases.

We conclude our discussion of high anonymity level and high spatial tolerance by claiming that MobiPriv algorithms guarantee that all messages can be anonymized safely at a relatively fast speed for high anonymity level and high spatial tolerance. MobiPriv algorithms however have a higher communication cost. *MobiPriv CloakedK* has a better communication cost than *CloakLessK*. The other anonymizing schemes such as PrivacyGrid and pyramid based dropped 20%-30% of the user requests.

We now evaluate the algorithms success rate, cloaking time and communication cost with *low average anonymity level* (k=10) and *high maximal spatial resolution* (700m*700m) as shown in Figure 4 (a,b,c).

Figure 4(a) plots the success rate with different number of user requests. In general all cloaking algorithms should have a higher success rate with a reduction in the average anonymity level. *MobiPriv* algorithms still anonymize all the requests. PrivacyGrid and Pyramid based shows an improvement in the success rate. This improvement is stimulated by the fact that it is fairly easy to find a small number of users to anonymize inside a large region. Pyramid based scheme only anonymizes 98.5% of the total request received.

In Figure 4(b) the graph shows the cloaking time with increasing number of request. The *MobiPriv* algorithms has the best cloaking time. In particular *CloakLessK* has an exceptionally fast cloaking time for low anonymity level since it can quickly generate dummies.

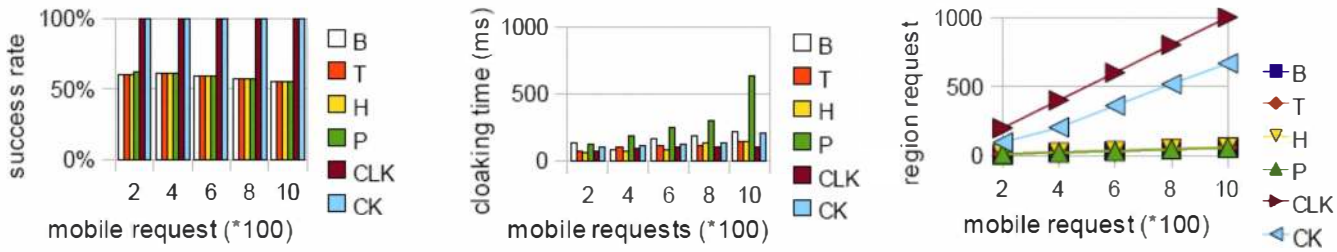
Figure 4(c) shows that all the cloaking algorithms has an increase in communication cost. *CloakLessK* communication cost is not affected by a reduction in anonymity level.

We end our discussion of the algorithms under low anonymity level and high spatial tolerance by concurring that MobiPriv algorithms still maintain a 100% success rate and also the fastest cloaking time. We however pay a price on the communication cost. In general we observed that all the cloaking algorithms showed an increase in communication cost.

We now evaluate the algorithms under *low average anonymity* level (k=10) and *low spatial resolution* (100m * 100m) in Figure 5(a,b,c).

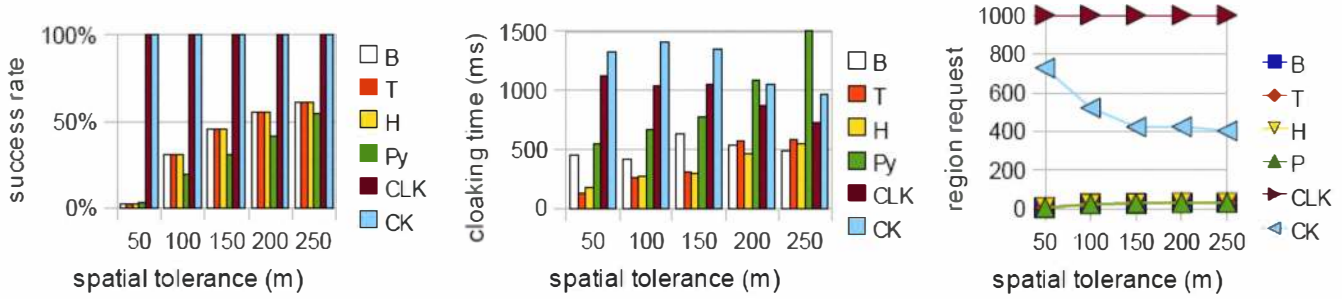
Figure 5(a) shows the success rate with different number of request. We observe two things. First the the success rate of the PrivacyGrid and Pyramid approaches is less than 60%. Secondly *MobiPriv* algorithms still maintains a 100% success rate even under such tight QoS requirements.

Figure 5(b) plots the cloaking time with varying number of mobile requests. The most obvious observation is that the pyramid scheme now has a very high cloaking time for low spatial resolutions. The *MobiPriv* algorithms has a very fast cloaking time in particular *CloakLessK* does slightly better than *CloakedK* .



(a) (b) (c)

Figure 5 – Results with low average anonymity level (k=10) and low spatial tolerance (100m*100m)



(a) (b) (c)

Figure 6 – Results showing the effect of spatial tolerance (k=50, number of request = 1000)

Figure 5(c) displays the communication cost against the number of request. We see that *CloakedK* has a better communication cost than C*loakLessK*. *MobiPriv* algorithms has a higher communication cost but it also ensures that all the messages are anonymized. The other algorithms only manages to anonymize 60% of the messages.

We end our discussion of low average anonymity level and low average spatial resolutions by claiming that MobiPriv algorithms maintains a 100% success rate while the PrivacyGrid and Pyramid based approaches achieved 60% success rate. We also saw that MobiPriv algorithms has a fast cloaking time that is comparable to any of the algorithms that we studied. The *MobiPriv CloakedK* does better in communication cost than *CloakLessK*.

In Figure 6 (a,b,c) we evaluated the effect of spatial resolution on the success rate, cloaking time and communication cost. We set the average anonymity level to 50 and the number of requests submitted to 1000.

Figure 6(a) plots the success rate with the spatial tolerance. *MobiPriv* algorithm achieved 100% success rate for any spatial resolution. The *PrivacyGrid* and *Pyramid* approaches all have very low success rate for low spatial resolution. For a spatial resolution of 50m*50m the *PrivacyGrid* (top down, bottom up, hybrid) and pyramid based (Casper) drops over 98% of the requests. An increase in spatial resolution to 100m*100m and PrivacyGrid and Pyramid based dropped 70% of the messages.

Fig 6(b) shows the cloaking time with different spatial tolerances. The *MobiPriv algorithms* all have a higher cloaking time because all the request are anonymized, unlike the other algorithms that only anonymizes a small fraction of the requests. The cloaking time of *MobiPriv* schemes decreases as the spatial tolerance increases.

Fig 6(c) highlights the chart of communication cost with different spatial tolerances. It is obviuos that the spatial tolerances does not affect the communication of *CloakLessK*. As the spatial tolerance increases the communication cost of *CloakedK* decreases because more real mobile users can be included in a *region request.* The other schemes PrivacyGrid and Pyramid all have very low communication cost because most of the requests that they received are discarded.

We conclude our discussion of the effect of low spatial tolerance on success rate, cloaking time and communication cost. We have seen that under low spatial tolerances (eg. 50m*50m) PrivacyGrid and Pyramid schemes discard most of the request. An example of a low spatial requirement in real life location based system is a transit itinerary system where the user of the system may request the shortest or fastest route from an origin to a destination and does not want to walk more than 50m.

4.4 CloakLessK vs CloakedK

We introduced the *MobiPriv* suite of algorithms. Both algorithms *CloakLessK* and *CloakedK* are able to achieve the maximum success rate under any anonymity level or spatial resolution.

The cloaking time of both algorithms in the *MobiPriv* collection are approximately the same with *CloakLessK* doing slightly better when the average anonymity level is small. With regards to communication cost *CloakedK* outperforms *CloakLessK*.

We found that the communication cost of CloakLessK is not affected by spatial resolution unlike *CloakedK*. The communication cost of *CloakedK* improves as the anonymity level increases or if the spatial tolerance increases. Also *CloakedK* does much better than *CloakLessK* as the number of request sent to our middle-ware increases.

4.5 Corollary history Attack and Cloaking Time

As discussed before, *MobiPriv* can handle corollary history attack. However, this is achieved at the cost of higher cloaking time.

Below in Figure 7(a) we discuss the relationship between the cloaking time and our technique used to reduce the corollary history attack. We set the average anonymity level to 5 and the spatial resolution to 700m * 700m. We observe a sharp increase in cloaking time for our two algorithms when we consider the corollary history attack prevention. This is due to the fact that for each request we read the dummy profile and get a set of realistic dummy properties. The time taken to read the dummy profile is added to the cloaking time. An optimization is possible if we consider a more efficient searching technique when consulting the dummy profile. In our experiments we used the simple linear search.

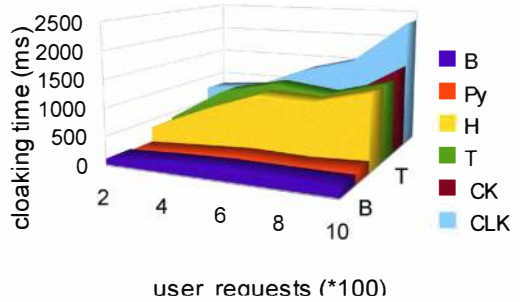


Figure 7 (a) – Corollary attack effect on cloaking time

5. RELATED WORK

The concept of k-anonymity was originally used within the realm of relational databases [9]. The idea of location k-anonymity was first used in [10] where *k* was static and is the same value for all users in the system. The framework of a personalized value of k was first introduced in [8]. To add another level of privacy *l-diversity* is considered. L-diversity adds more protection to mobile user and guarantees protection from the homogeneity attack. The concept of l-diversity was coined in [26].

In [11,12, 13,14], a client-server approach is considered instead of using anonymization servers. Also, the work in [21] is based on the peer to peer approach. Our work is different and we concentrated on the trusted third party architecture.

Work on the trusted third party architecture includes *Interval Cloaking* [4,10], *Clique-Cloaking* [8], *Casper* (Pyramid Based) [19], *PrivacyGrid* [15]. These model all suffer from the request dropping problem and did not consider generating dummies on the middle-ware. These models are also more susceptible to the corollary history attack. In [19] the authors maintain an anonymizer and a privacy aware query processor. The same authors of [19] also introduced *TinyCasper* [18]. TinyCasper is solely for wireless sensor networks. The approach in [15] is dynamic and produces a smaller cloaking box than [19].

Our work is the first to guarantee that a user's personalized request of privacy and quality of service can be satisfied using the trusted third party architecture. Also, the first to consider protection against the corollary history attack by utilizing realistic diverse dummies.

6. CONCLUSION

We introduced our suite of MobiPriv algorithms which allows mobile users to specify any value of *K* to guarantee a very high level of privacy. Our experimental evaluation proved that our algorithms can guarantee a 100% success rate for any personalized QoS requirement. Our results showed that *MobiPriv* algorithms have a fast cloaking time and higher communication cost. The communication cost of *MobiPriv CloakedK* improves relative to *CloakLessK* when the number of requests increases. Also, *MobiPriv* guarantees resilience against the corollary history attack

7. FUTURE WORK

We intend to deploy *MobiPriv* system to be used as the middle-ware for TransitGenie[20]. TransitGenie is a context aware transit itinerary planner for the city of Chicago, Illinois USA.

8. ACKNOWLEDGEMENTS

This work is supported in part by NSF through grants IIS-0914934, IIS-0905215, and DBI-0960443. Also DGE-0549489, IIS-0957394, and IIS-0847680.

9. REFERENCES

[1] The cellular telecommunication and internet association, ctia Website . http://www.wow-com.com/.

[2] R. Agrawal and E. Wimmers. *A Framework for Expressing and Combining Preferences. In Proceedings of the ACM International Conference on Management of Data, SIGMOD, 2000.*

[3] Federal Communications Commission http://www.fcc.gov/cgb/consumerfacts/wireless911srvc.html

[4]S. Mascetti, C. Bettini, D. Freni, X. Wang. *Spatial Generalization Algorithms for LBS Privacy Preservation. Journal of Location Based Services ISSN 1748-9725/ISSN 1748-9733, 2007*

[5] Google Latitude Website - http://www.google.com/latitude/intro.html

[6]L. Lui, *From Data Privacy to Location Privacy: Models*

and Algorithms. VLDB, 2007
[7]F. Liu, K. Hua, Y. Cai. *Query l-Diveristy in Location-Based Services. International Conference On Mobile Data Management,2009*
[8]B. Gedik, L. Lui *Location Privacy in Mobile Systems: A Personalized Anonymization Model. ICDS, 2005*
[9] P. Samarathi and L Sweeney. *Protecting Privacy when disclosing Information: k-Anonymity and its Enforcement through Generalization and Suppression. SRI-CSL-98-04*
[10] M. Grusteser and D. Grunwald. *Anonymous usage of location based services through spatial and temporal cloaking. ACM/USENIX MobiSys, 2003*
[11] M. Yiu, C. Jensen, X. Huang, H. Lu. *SpaceTwist: Managing the Trade-Offs Among Location Privacy, Query Performance, and Query Accuracy in Mobile Services. 24th International Conference on Data Engineering , 2008*
[12] H. Kido, Y. Yanagisawa, T. Satoh. *An Anonymous Communication Technique using Dummies for Location Based Services. Second International Conference on Pervasive Services, 2005.*
[13]T. You, W. Peng, and W.-C. Lee, *Protecting Moving Trajectories Using Dummies. International Workshop on Privacy-Aware Location-Based Mobile Services, 2007.*
[14] G. Ghinita, P Kalnis, A. Khoshgozaran. C. Shahabi, K. Tan. *Private queries in Location Based Services: Anonymizers are not Necessary. SIGMOD, 2008*
[15] B. Bamba, L. Liu, Peter Pesti, T.Wang. *Supporting Anonymous Location Queries in Mobile Environments with PrivacyGrid . World Wide Web, 2008*
[16] B. Chor, O. Goldreich, E. Kushilevitz, M. Sudan. *Private Information Retrieval. IEEE Symposium on Foundations of Computer Science, 1995*
[17] E. Kushilevitz and R. Ostrovsky. *Replication is NOT needed: Single Database, Computationally-Private Information Retrieval. IEEE Symposium on Foundations of Computer Science, 1999*
[18] C. Chow, M. Mokbel, T. He. *Tiny Casper: A Privacy-Preserving Aggregate Location Monitoring System in Wireless Sensor Networks. SIGMOD, 2008*
[19] M. Mokbel, C. Chow, W.G. Aref . *The New Casper: Query Processing for Location based Services without Compromising Privacy. 32nd International Conference on VLDB, Sep 2006*
[20] TransitGenie Website- *Your Personal Transit Navigator* (November 2009) www.transitgenie.com
[21] C. Chow, M. Mokbel, X. Liu . *A peer to Peer Spatial Cloaking Algortihm for Anonymous Location Based Services. ACM GIS, 2006*
[22]NextBus Website- http://www.nextbus.com/predictor/agencySelector.jsp (2009)
[23] USAToday. Authorities: GPS system used to stalk woman.http://www.usatoday.com/tech/news/2002-12-30-gpsstalker x.htm.
[24] Foxs News. Man Accused of Stalking Girlfriend With GPS. http://www.foxnews.com/story/0,2933,131487,00.html.
[25] Anonymous surfing. http://www.anonymizer.com.
[26] A. Machanavajjhala, J Gehrke, D. Kifer and M Venkitasubramaniam, *"L-diversity": Privacy beyond k anonymity. ICDE, 2006*