PRIVACY FOR LOCATION-BASED SERVICES ON SMARTPHONES

by

JAMES NATHANIEL SPARGO

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

THE UNIVERSITY OF TEXAS AT ARLINGTON

May 2011

To my father Dr. Bill Spargo whose encouragement helped push me to limits I never thought I would attain. To my friends and family: c'mon, let's conquer the world.

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my supervising professor Dr. Matthew Wright for his patience, guidance, encouragement and for putting up with of my shenanigans in general. His careful guidance and hands off approach helped me to think about things in different perspectives that I had initially taken. It was this guidance that helped to open my mind to different possibilities and considerations. I will keep and use these experiences for the rest of my life, and for that I shall be eternally grateful.

I would like to thank Dr. Vassilis Athitsos and Dr. Dongang Liu for serving on my thesis committee as well as their interest in my research. I am grateful for the help of Aniket Pingley for his guidance in getting my application working and its implementation. Additionally, I would like to thank Duy Nguyen for bringing to my attention Pingley's work on CAP and the importance of location privacy.

I would be remiss if I did not thank Dr. Carter Tiernan and Mr. Tom Rethard. My conversations (and arguments) with them during my final undergraduate years are some of the largest reasons as to why I decided to continue with graduate school. I should also thank everyone involved with the Information Security Lab. The feedback, suggestions and encouragement they offered helped me to continue on in even my most frustrating times.

Finally, I would like to thank my family and friends. Were it not for their support throughout my academic career, I wouldn't have made it as far as I have. I will never forget it.

April 18, 2011

ABSTRACT

PRIVACY FOR LOCATION-BASED SERVICES ON SMARTPHONES

James Nathaniel Spargo, M.S.

The University of Texas at Arlington, 2011

Supervising Professor: Matthew Wright

In recent years, mobile devices and smartphones enabled with GPS and Internet access have become extremely common. People use these devices as they would a personal computer for easy access to information. Location Based Services (LBS) provide customized information based on a user's geographic location that has been retrieved from a dedicated spatial database such as Google Places, Yahoo's Local Search Web Services and Yelp.com. This information can include nearby hotels and restaurants, gas stations, banks or other Points of Interest (POIs). Since most search engines and databases are known to store previous queries in order to improve future search results and other data analysis on previous search queries, many researchers have expressed concerns and proposed solutions to protect a user's location privacy. Research has shown that a significant amount of information, such as medical conditions, political or religious affiliations and more can be inferred based on a person's previous location tracks. Many methods proposed by researchers rely on the use of trusted third parties such as Anonymizing Servers, other nearby mobile devices, or the LBS itself. CAP (Context-Aware Privacy), introduced in 2008 by A. Pingley et. al., is a system that was designed to protect a user's location without having to rely on a trusted third party or interfere with the operation of the LBS. A desktop prototype was made, yet it was never implemented on a mobile device or smartphone until now. Preliminary tests of CAP with lower privacy settings proved to be effective, although when the privacy settings were increased, the results seemed to deteriorate. Closer examinations of the algorithm indicate that it is effective when compressing contextual map data for use by a mobile device, as well as effective perturbation of the user's location. The POI results returned from the LBS tell a different story. While the POI results at low privacy levels seemed to be accurate (i.e. the POIs returned from the LBS are in fact the POIs that are closest to the user), when the privacy settings were increased, the results would degrade (i.e. the POIs returned were, as expected, further away from the user's actual location). This is effective in the sense that the user's actual location is not able to be divulged to an adversary, but is not very effective in terms of usability and convenience for the user. In this thesis, we review CAP and several other proposed methods of location privacy intended for use with mobile devices. We have implemented CAP on a smartphone in its proposed method and evaluate its results, followed by modifications in order to gain more accurate POI results from publicly available LBS.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

In recent years, the capabilities of handheld cellular devices and smartphones have increased dramatically. Processor speeds, wireless transmission speeds, and location determination, combined with lower hardware costs have made these devices both desireable and attainable to the general public. These devices have made it possible for people to obtain, process and transmit large quantities of information with relative ease more than ever before. One area that has had increasing attention paid to it is the availablity of a device's location and the services that are available to it based on that location. These Location Based Services (LBS) have wide areas of application, from informational, e.g. nearby hotels, gas stations and banks, to entertainment, e.g. location of nearby friends or entertainment, to commercial such as location-based advertising, and many more

1.1 Motivation

Currently, there are many services available to provide content to users. Search engines, social networks, advertisers and more all collect information from users when accessing these services. Many times, the information that is collected by these services are either poorly protected or abused. In early 2010, Ron Bowes wrote a webcrawler program and harvested 2.8 GB of data from public facebook profiles [1] and distributed the data via torrent [2]. He collected the information of over 171 million facebook users. While the information he collected is public, his technique sidestepped a facebook privacy setting that prevents a user from being searchable [3].

If a user had this setting enabled, yet was on the 'friend list' of someone with a public searchable profile, that user's information is now exposed. This can be a major problem for organizations or individuals that don't want their detailed personal or ogranizational information easily available [4].

1.2 Contribution of Thesis

In this thesis, we take a look at numerous proposals for protecting a users location privacy when querying Location Based Servers with a mobile device. We examine the positives and negatives of each method, the amount of location privacy they provide for the user and their feasibility of being implemented in a real world scenario. Finally, we take a more focused look at CAP, introduced by Pingley, et al. [5] which provides algorithms for protecting a users location when querying an LBS via location perturbation and communication anonymity. CAP is unique because it does not depend on a trusted third party to provide location privacy. It provides two methods for protecting the users location, location perturbation to mask the users actual GPS coordinates and anonymous routing to prevent location discovery through IP tracing. Most handheld mobile devices available today use cellular connections to connect to the internet. As explained in Chapter 3, cellular network providers always know the location of a mobile device accessing their network to within 300 meters, making the anonymous routing component of CAP unnecessary, therefore, we will only be looking at the location perturbation component of CAP. We will be taking a closer look at its implementation, its utility in maintaining users privacy, and its performance characteristics on a midrange smartphone.

1.3 Organization of this Thesis

This thesis is organized as follows: Chapter 2 describes previous research and gives detailed explanations of LBS, Context Awareness, Location Privacy and packet routing on cellular networks. Chapter 3 gives a detailed description of the implementation of CAP, its current abilities, how it handles context with relation to map recreation, and the reading of the map files that are produced. It also describes how we set up our experiments, the efficiency of the algorithms and introduces a modification of CAP that creates a cloaking region similar to many of the other location privacy preserving methods proposed by other researchers. Chapter 4 describes the hardware and software implementations of that we used to test using readily available components. Chapter 5 describes the results the accuracy results that we were able obtain how we measured them and what they mean.. Finally, the conclusion found in Chapter 6 provides a summary of what we did, why we did it and what the results were.

CHAPTER 2

BACKGROUND

In this chapter, we review previous research and provide a general overview of Location Based Services, Context Awareness in the terms of mobile devices and their location and the importance of location privacy.

2.1 Location Based Services

Location Based Services (LBS) deliver content to or from devices based on location of the device in its current environment. While not limited to use by mobile devices, most are currently implemented with the intent of being used by mobile devices that have the ability to determine their geographic location either through the use of GPS or radio telemetry triangulation, particularly with the use of cellular towers [6, 7]. Services such Rave Mobile Safety [8] and e911 [9] provide location, identification information and more to emergency personal in the event that the user is unaware of or unable to communicate their location. Some of the more popular LBS [10, 11, 12, 13] allow users to 'check in' to a location, alerting approved users and "friends" to their current whereabouts.

The LBS that this thesis focuses on are the ones that have the ability to implement search criteria to locate nearby Points of Interest (POI). POIs are places such as restaurants, coffee shops, hotels, retailers, etc. that are near the mobile device's current geographic location. Examples include Google Places [14], Yahoo Local Search [15], and Yelp.com [16]. These types of LBS require a minimum input of geographic location and search criteria. They have the ability to include things such as area of interest, the number of results returned, categories (i.e. restaurant, lodging, entertainment), user ratings, etc, based on the implementation of the LBS . In practice, the user sends a GET HTTP request, along with their current location and search criteria to the LBS. The LBS then searches its database and returns the list of POI results to the user. These results are usually returned in a limited set initally to save bandwidth and limit the amount of data that is required to be transmitted.

LBS are usually implemented as spatial databases. Spatial databases are full-fledged database systems with additional functionality for handling spatial, or geographic data and connecting it to non-spatial, or alphanumeric data [17]. The discussion of backend implementation and optimization techniques of LBS and spatial databases, while important, is beyond the scope of this work.

2.2 Location Tracks

In [18], Kang et al. explain how an adversary can determine a user's habits, places of frequent visitation and more based solely on a user's previous locations, or *tracks*. They were able to identify a user's significant places when they determine that a number of consecutive location measurements are clustered together. They examined the location tracks of different users for periods that ranged from a few hours to a few days. In their study, they used an application called The Place Lab [19], which allows for a WiFi-enabled client to determine its location by listening to access points and identifiying significant locations by clustered areas of location samples. These clients collected these data points as their subjects worked through out the day on the campus of the University of Washington. In their algorithm, they used variable time and distance parameters for recording locations throughout the user's daily routine. These variable parameters were then used to filter out noise created by samples that were taken too close together in either time or distance when the user was traveling from one location to another. They were able to determine where a user spent most of his time throughout the day with a precision of 84% when the time between location recording was set to 30 meters and time set to 300 seconds. They show that specific areas, where a user spends a significant amount of his time, can be identified more easily when the time is increased to at least 500 seconds and the distance threshold is increased to 50 meters. In spite of the accuracy of their findings when samples were taken further apart, when the sample rates were low, i.e. time less than 100 seconds and distance less than 20 meters, the number of clusters they detected were high, many times over 30. They were able to show that when there is more noise in a set of sampled locations, the sizes of detected clusters is larger, giving the attacker a larger area to search when trying to determine a user's significant location.

2.3 Location Privacy

In recent years, there have been numerous studies regarding location privacy and protecting a user's significant locations, such as a home, place of employment, or places of frequent visitation. Some studies [18, 20, 21] have shown several successful methods for determining a user's significant location based on a user's previous location data, while [5, 22, 23, 24, 25, 26] have demonstrated with varied success, methods of obscuring a user's location data. A user's significant locations are the ones where a user spends a significant amount of their time, such as their home or work place, but can also include particularly sensitive locations such as a doctors office, a religious establishment or a place of socially alternative entertainment.

Liu defines location privacy as the ability to prevent other parties from learning a user's current or past locations [27]. It is shown that an adversary can learn a great deal of information about a person from their previous location tracks. An adversary can make inferences about a user's medical conditions, alternative lifestyle or political views which can be insinuated from visits to clinics, entertainment areas or political rallys [27, 22]. Exposing this information can lead to targeted spamming, fraud, stalking and even physical harm such as domestice violence [28]. For example, in late 2002, a man was arrested and charged with stalking his ex-girlfriend after investigators found a GPS device installed under the hood of her car that was allegedly installed by the accused [28, 29]. This case suggests the potential harm that can be done by a malicious individual when a person's GPS coordinates have been obtained. Liu also points out that a k-anonymity algorithm, a popular method for location cloakingi, is not very effective when the k-1 other users are all in the same location [27]. K-Anonymity, as defined by Domingo-Ferrer et. al. in [30] states: A protected data set can be said to satisfy k-anonymity for $k > 1$ if, there are at least k data sets that contains the same combination of key attributes.

2.3.1 Location Privacy Using K-Anonymity

Recall from 2.3, that k-anonymity is satisfied when a data set containing k objects have the same combination of key attributes, rendering them indistinguishable from each other. Key attributes can be any attributes that, when used in combination with each other and linked to external information, can be used to re-identify an element of the protected data set. Some researchers [26, 24, 5, 22, 31] use k-anonymity with respect to location privacy in order to perform spatial cloaking around a user's actual location. Using this method of spatial cloaking, they are attempting to increase the area where a user might possibly be.

Chow et. al. [24] propose a method that combines peer-to-peer communication with LBS servers that implement a privacy-aware query processor. Where existing LBS servers require an exact location with their queries and return exact results,

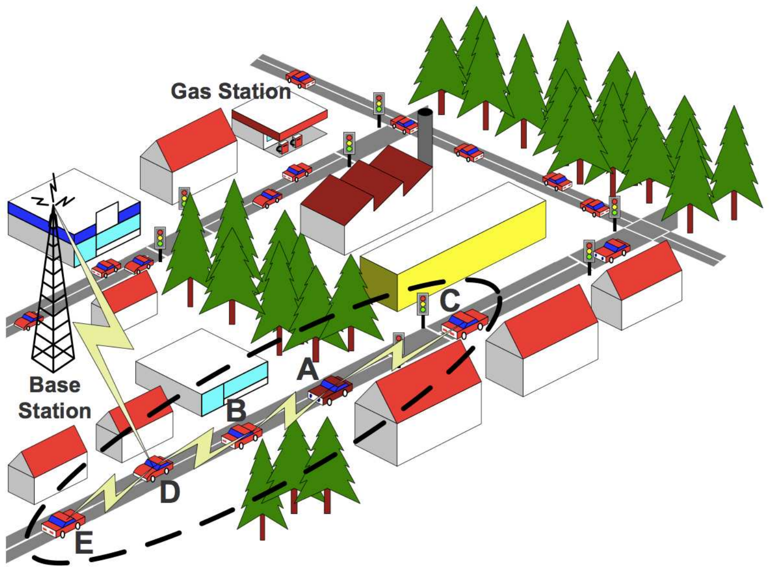

Figure 2.1. An example of a mobile device communicating with other nearby mobile devices to establish a cloaking area of k-1 other users.

Privacy-aware query processors [26, 25] require a cloaked region with the query, and return a list of *candidate* answers. In the peer-to-peer phase, the user specifies the privacy settings k and A_{min}, where k is the minimum level of *k-anonymity* and A_{min} is the minimum area to be used as the cloaking region. The user's mobile device communicates with other nearby mobile devices, either directly or via other nearby peers, attempting to find k-1 other devices. These k-1 other peers return their actual location to the user's device which are then used to determine the initial size of the spatial cloaking region (See figure 2.1. Image source: [24]) If the device is unable to find k-1 other peers in two rounds of rebroadcasts, it determines that a peer group cannot be formed and the user must relax their privacy settings. If the cloaking region is smaller than A_{min}, the cloaking region is increased to the size of A_{min}. The user's device then selects one of the k-1 peer devices to act as a query agent between the device and the LBS. The cloaking area is then sent with the query to the agent which then forwards it on to the LBS that contains a privacy-aware query processor. The LBS then returns the list of *candidate* answers to the query agent who then forwards them to the user's device. The list of candidate answers is then filtered to remove any false positives.

2.3.2 Location Privacy Using Location Anonymizers

Other researchers have proposed location privacy solutions using a combination of k-anonymity and spatial cloaking through the use of location anonymizers [25, 26, 24]. Location Anonymizers are systems designed to mask a users identity when requesting information from an outside service. It does this by keeping track of all the user registered with the system and assigns masking identifiers to them so that an adversary outside or the system are unable to determine who exactly is requesting the information Mokbel describes two methods of creating a spatial cloaking region using Location Anonymizers; data-dependent and space-dependent [25]. Each method has a requirement of providing the user wtih a minimum k for k-anonymity and specific areas A_{min} and A_{max} for a spatial region A that satisfies $A_{min} < A < A_{max}$. In the data-dependent method, the cloaking region is formed using several point locations, made up of other users or devices, around the user's actual location to form a Minimum Bounding Rectangle (MBR). With the space-dependent method, the cloaking region is formed from a prepartioned space, such as a fixed size grid or through the use of a Quad-tree. In Quad-tree partitioning, the anonymizer, tries to determine a cloaking area A that satisfies A_{min} and A_{max}. It starts by looking at the whole space, then recursively partitions the space into 4 quadrants until it comes to a quadrant that does not satisfy the user's requirements. This latest quadrant is what is returned as the spatial cloaking region. The fixed sized grid partitioning method partitions the space into equal sized cells, then locates the grid cell that the user's

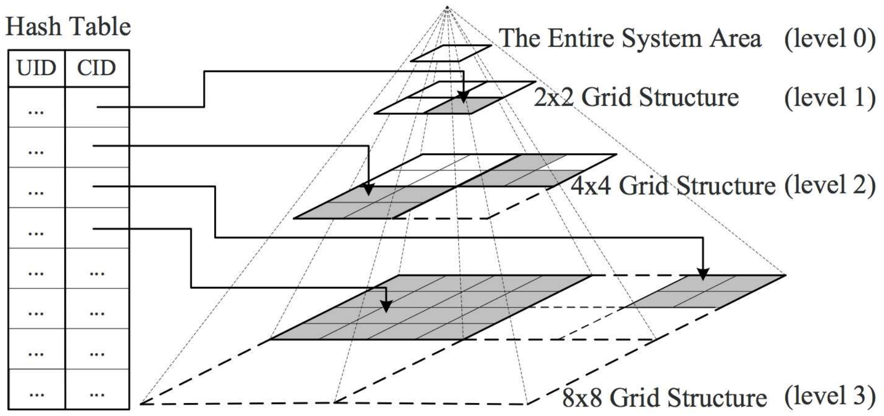
Hash Table
UID
CID
The Entire System Area (level 0)
2x2 Grid Structure (level 1)
4x4 Grid Structure (level 2)
8x8 Grid Structure (level 3)

Figure 2.2. The graphical representation of the incomplete pyramid used by The New Casper for spatial cloaking.

actual location lies in. If the size of this cell does not meet the requirements for k and A_{min}, the anonymizer adds adjacent grid cells until it does so.

The New Casper [26] implements spatial cloaking by the use of a Location Anonymizer based on the Quad-tree partitioning method mentioned above. The Location Anonymizer breaks down a geographic region into a pyramid shaped grid data structure that decomposes the area heirarchically into into H levels, where the height level of h has 4^h grid cells. The root of the pyramid is at height 0 and consists of the whole space that the anonymizer services. The user defines a privacy profile that contains values for k and A_{min}. The user's device continually reports its location to the Location Anonymizer. The Location Anonymizer maintains a hashtable consisting of touples made up of (*cid, N*), where *cid* is the cell identifier and N is the number of mobile users within the cell boundaries. It also maintains a hashtable of users, with each user having an entry that consists of a tuple made up of (*uid, profile, cid*), where *cid* is the cell identifier in which the user is located, *profile* is the user's privacy profile, and *uid* is the mobile user identifier.

In an effort to reduce the processing time and cost of maintaining the entire pyramid, The New Casper uses an Adaptive Location Optimizer (See figure 2.2. Image Source: [26]). In general, it maintains an incomplete pyramid structure that contains only those grid cells that can possibly be used as cloaking regions for the mobile users. It does this by maintaing only the cells at the highest level, which is determined by the user with the most relaxed privacy settings. This results in the Location Anonymizer only needing to maintain an incomplete pyramid, with the smallest grid cells being maintained for the user with the most relaxed privacy settings.

They compared the effects produced by varying the number of users and k ranges on the cloaking times and update cost of maintaining the two different types of pyramids; the basic pyramid which maintains all of the cells at the highest level, and the pyramid maintained by the Adaptive Location Optimizer. The results when varying the number of users maintained by the Location Anonymizer is as expected, with the efficiency of the Adaptive Location Optimizer being greater than the basic pyramid when more users were added to the system. However, the cloaking time and update cost of the Adaptive Location Optimizer, when varying the k ranges, had a significant increase of efficiency over the basic pyramid when the values of k were increased. This follows along with their conclusion that this method of implementing a Location Anonymizer provideing sufficient location privacy can meet the requirements of efficiency, accuracy, quality and flexibility, yet still be scalable enough to handle large amounts of users.

2.3.3 Location Privacy with The VHC-Map

The two methods discussed in this subsection make use of a technique called *V*arious-size-grid *H*ilbert *C*urve Mapping (VHC-Mapping). We begin by describing the main concept behind the VHC-Mapping, followed by its general implementation details. The first method we discuss, *C*ontext *A*ware *P*rivacy (CAP) [5], proposed by Pingley et. al., applies VHC-Mapping to road density in a geographical region due to its strong (approximately) linear correllation to population density [32]. The second method we discuss, proposed by Olumofin et. al. [33], applies VHC-Mapping to the POIs contained in the LBS with respect to their density in a geographical region. In the case of CAP, the VHC-Mapping is created by the user and is used by the user's mobile device. In the case of [33], the VHC-Mapping is agreed upon by both the user's device and the LBS to retreive all POIs within a given geographic area from the LBS. The mobile device then chooses which POIs are of interest to the user.

2.3.3.1 Hilbert Curve

The Hilbert curve is a continuous fractal, space filling curve. It is used for mapping points in a multi-dimensional space to a linear single dimension [34]. In most cases, it is used to map a 2 dimensional (2-d) space into a 1 dimensional (1-d) space. The Hilbert Curve is used because of its well known locality-preserving properties, which basically means that two points close to each other in the 2-d space are likely to be close to each other in the 1-d space. Due to these well known locality-preserving properties, it has been applied to mapping applications [5], databases [33], image compression [35] and more. The Hilbert curve is designed in such a way that after enough recursive iterations, it will pass through every point in a two dimensional plane. Figure 2.3 shows a series of images that display the first three orders of the hilbert curve [1].

[1]Image Source: http://en.wikipedia.org/wiki/Hilbert_curve

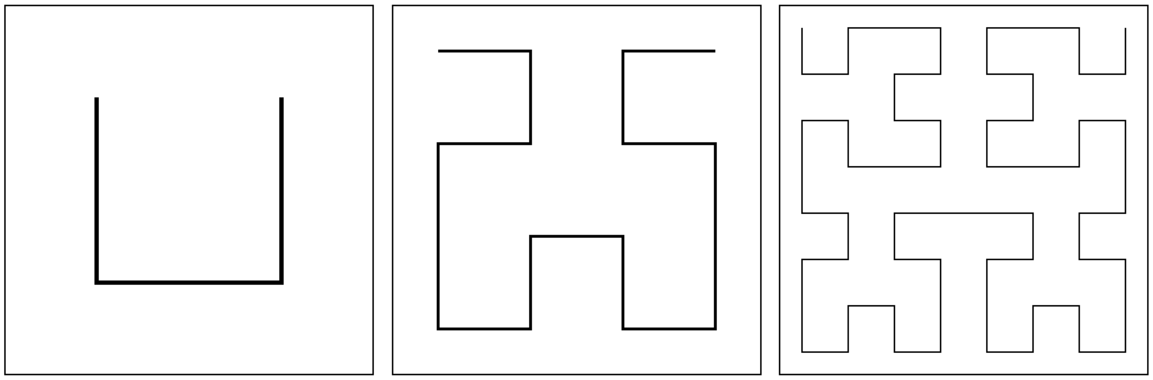

Figure 2.3. The first three orders of the Hilbert Curve.

2.3.3.2 Various Sized-Grid Hilbert Curve

The *V*arious-sized-grid *H*ilbert *C*urve based Mapping (VHC-Mapping) is constructed based on context information such as population or road density. Similar to the Hilbert Curve, it is designed to map an original 2-d space to a locality preserving 1-d space. VHC-Mapping goes a step further in that it divides the 2-d space based on some type of context which maintains a constant density in each partition. It does this by relying using the *Minimum-Density Rule* which is defined as follows:

Minimum-Density Rule: *Partition a cell into 4 square sub-cells iff the total number of X is greater than Y. Where X is some context metric determined by the application of the VHC-Mapping and Y is a pre-determined granularity ratio.*

Figure 2.4 shows Andrews County, Texas after being paritioned by a VHC map based on road density as the context parameter, X. Olumofin et. al. [33] use the number of POIs in the given cell as the metric for X, and the preset measure of 50 for the granularity level in their example. In the case of [5], Pingley et. al. use road density, or the number of miles of road in the given cell, as the metric for X, and a formula that takes into account the edge length of the cell and a predetermined granularity level. The details of each method are given later in the chapter.

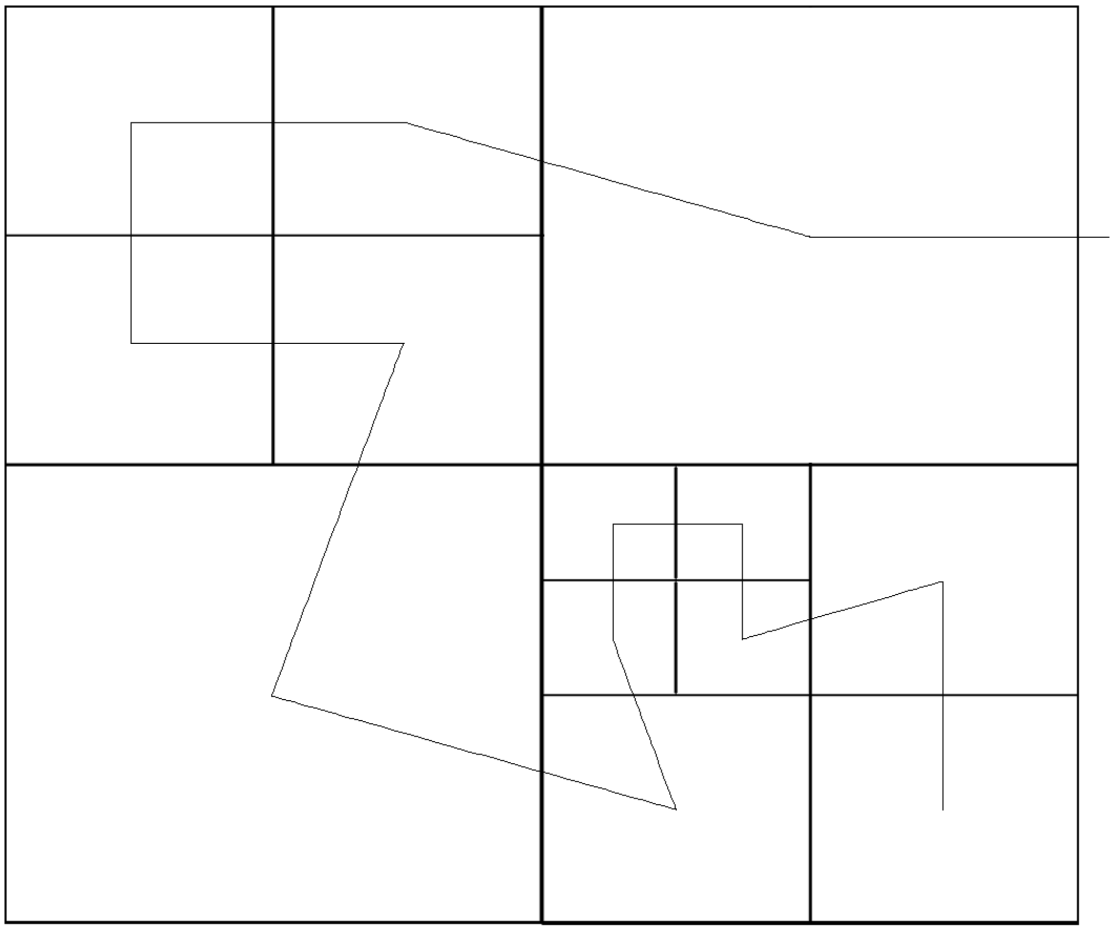

Figure 2.4. Andrews County, Texas divided by a VHC Map including the Hilbert Curve.

2.3.3.3 Context Aware Privacy

*C*ontext *A*ware *P*rivacy (CAP) [5], proposed by Pingley et. al., is a location privacy system that attempts to hide the user's actual location by using a combination of location perturbation and anonymous routing. The method they propose is effective for hiding the user's location from an attack directed at either the LBS or eavsdropping. The location perturbation component of CAP presents the LBS with a different location than where the user actually is. They use the anonymous routing componenet in order to mask its location from attackers attempting to determine the users location based on their IP address [5].

In the location perturbation component of CAP, they use VHC-Mapping to allow the mobile device to determine the population of its immediate area, or context, before determining the amount of perturbation required for the users desired privacy settings. The VHC-Mappaing is implemented in two stages; an offline stage that creates the VHC map and serializes it for transfer to the mobile device, and an online stage that reconstructs the map on the mobile device and performs the actual perturbation. Recall from 2.3.3.2, the VHC-Mapping is a precomputed projection of the original 2-d space of latitude and longitude, into a 1-d space. In the construction of the VHC-Mapping, they used road density data as input because economic studies have shown that road density is strongly correlated with population density [32] and because the data is readily available from the U.S. Census Bureau [36].

The offline portion of the location perturbation component starts by creating the VHC mapping. It takes a square area and recursively partitions it into various sized square cells based on the context (road density) information. During each iteration of the partioning, each cell is divided into 4 smaller cells, or not, based on the Minimum Density Rule. After the space has been partitioned, they map the 1-d space using a variation of the Hilbert curve to connect all of the various sized cells in the 2-d space. Because each partitioned cell in the VHC map is partitioned into 4 equal sized cells, it can easily be represented as a quad-tree with each node.

This quad-tree is then able to be serialized into a binary array in depth-first order with a 0 representing nodes that have been partitioned and a 1 representing a node that has not been partitioned. Figure 2.5 shows a quad tree representation and binary serialization of the VHC Map of Andrews County, Texas. The size of the serialized binary file is orders of magnitude smaller than the original map containing all of the context information. This allows for the precomputed map to be stored and transferred to the mobile device efficiently.

The online portion of the location pertubation components starts by loading the binary VHC file and bounding coordinates into memory It uses these two files to recreate the VHC mapping in memory. When the user is ready to send a query to the LBS, they first specify the privacy level, query terms and sets the location. This actual location is then used as the input to the location perturbation component. The location perturbation component then determines which VHC cell it is located within and maps it to its corresponding 1-d VHC map value. They then add random noise to the 1-d value, based on the users privacy setting. This 1-d value is then mapped back to the corresponding 2-d VHC cell. The system then chooses a random point within this VHC cell to be used as the coordinates in the query to be sent to the LBS.

After the location perturbation had been performed, the query is assembled sent to the LBS. The second component of CAP is the Anonymous Routing Component. The Anonymous Routing component is used to prevent an attacker from determining the user's actual location based on their IP address. CAP uses Tor[2] for Anonymous Routing due to practicality and ability to perform anonymous routing over the Internet. Tor is an overlay network made up of Onion Routers that use random path selection algorithms to preserve the anonymous routing aspect. Many Tor routers use donated bandwidth whos donors may limit its availability and in turn, limit the end-to-end throughput of the overall query and response time. The challenge then becomes tuning up the Tor system in order to optimize the *Q*uality *o*f *S*ervice (QoS) given.

[2]https://www.torproject.org/

To overcome this challenge and improve the QoS, CAP proposes a method of differential QoS within the Tor network. They do this by partitioning the Tor network into classes of low and high bandwidth using the class of high bandwidth routers for the requests and response to the LBS. After creating the partition, they then create a path, or circuit, of Tor nodes to be used with the differential routing. These circuits can then be chosen based on a particular flow requests priority level. This proposal only partially guarantees the the upper bound of the available throughput of the Tor nodes, mainly due to traffic congestion along the path. To get around this problem, they propose a method of proactive circuit creation which proactively measures the path throughput and creates a circuit. It continues to take these measurements until the path throughput requirement is met and then builds the circuit. Using this scheme, they keep track of the Tor nodes used in previous queries to avoid using those nodes again. They have shown that this use of Tor still creates a lag in communication time, but by classifying the paths within the Tor network, they are able to get this lag down to acceptable levels.

To test CAP, Pingley et. al. created their own LBS using 800 randomly selected POIs to test against and set up numerous experiments. Some of the results examined were true positive rate, LBS accuracy and extra miles traveled due to the LBS returning POIs based on the perturbed location. To measure true positive rate, they measured the probablity that a tuple in the result set is indeed a true top-k tuple. For LBS accuracy, they measured the average rank distance of the tuples returned from the LBS query made on the perturbed position compared to the results returned for the users actual location. The 'extra miles traveled' results are based on the distance the user will have to go because the POIs returned from the LBS are not actually the POIs closest to the user. While the results from these experiments seem to be within acceptable limits, their desktop prototype of CAP give different results.

When using higher privacy settings, i.e. higher noise paramerter, the POIs returned from the LBS are significantly further away than the POIs returned when the lowest privacy settings are used.

2.3.3.4 Query Privacy with VHC-Mapping and Private Information Retrieval

In the approach proposed by Olumofin et. al. in [33], VHC-Mapping is applied in a manner differing from the previously mention work. They propose a method to preserve a users privacy by combining location privacy with query privacy through the use of a cloaking region and an LBS that implements *P*rivate *I*nformation *R*etrieval (PIR) protocols. This method differs from CAP in the sense that they use VHC-Mapping on the LBS with respect to the spatial density of the POIs registered with the LBS as opposed to popultion or road density. They divide a 2-d geographic region in such a way that each partition has equal POI density. For example, suppose a POI database covering the U.S. and Canada contains 6 million POIs. If each cell contaians the same amount of POIs, e.g. 60, they will have a VHC-Map that contains 100,000 cells. Now suppose the lowest density cell with 60 POIs has an area of 40,000 km^2. This implies that the largest VHC cell will have an area of 40,000 km^2, or 200 km x 200 km. This geographic area overlying the U.S. and Canada is then divided into grid cells of 200 km in length, although each cell may contain a number of smaller VHC cells in regions with a greater POI density. The user then determines their desired privacy level which dictates the number of VHC cells to use as the cloaking region. This cloaking region is then sent to the LBS in the form of bounding coordinates, i.e. the top left and bottom right lattitude and longitude of the area. This allows the server to determine which VHC cells belong to the cloaking region and which portion of the database to be read. Additionally, the client also encodes which VHC cell contains the area of interest inside the PIR query. The server then returns to the user, all of the POIs within the requested VHC cell and the user's device is then able parse off the unwanted POIs.

Private Information Retreival (PIR) is a protocol that allows a user to retreive information from a database without revealing the particular information the user is retrieving. Unfortunately, PIR retrieval is computationally expensive due to the fact that the protocol typically must processes every item, or POI, when executing a query in order to defeat the servers ability to narrow down the search space. This gives a linear computational complexity on the order of n, where n is the number of POIs in the database. They state that this is one of the main drawbacks of a practical PIR deployment.

In order to reduce this computational complexity, they propose a hybrid method using a trade off of privacy in exchange for better query perforamce. Returning to the example from above, if the user lowers their privacy level, or reduces the size of their cloaking region, the PIR query will execute faster than it would with a larger cloaking region.

Their solution preserves the privacy of the user's location regardless of the number of other users in the area sending POI queries from the same location. They point out that tens of thousands to millions of users can inhabit an area of 200 km^2, which they claim is excellent privacy. It also protects the content of the query with the use of PIR retrieval to futher reduce the ability of an attacker to determine what types of POIs the user is interested in.

Some of the drawbacks of this implementation are similar to the ones found in Casper [26] and the Peer to Peer methods [24] in that they rely on a trusted third party. Both the client and the server are in agreement when it comes to the layout of the geographic cells. Additionally, when a cloaking region is used in combination with PIR protocols on the LBS, the larger the cloaking region specified by the user, the longer the processing time of the PIR retrieval, and therefore the response from the LBS, will take.

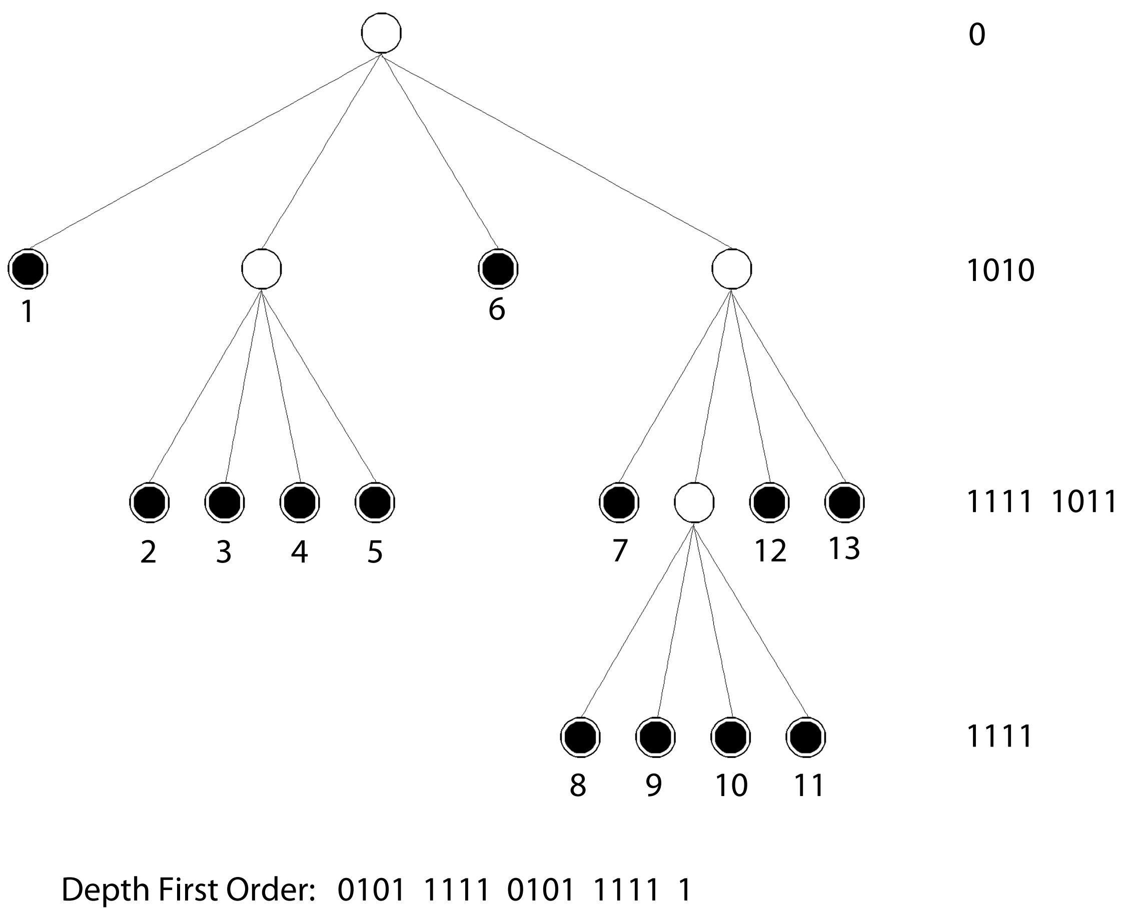

Figure 2.5. The quad-tree and binary representation of the Andrews County, Texas VHC Map.

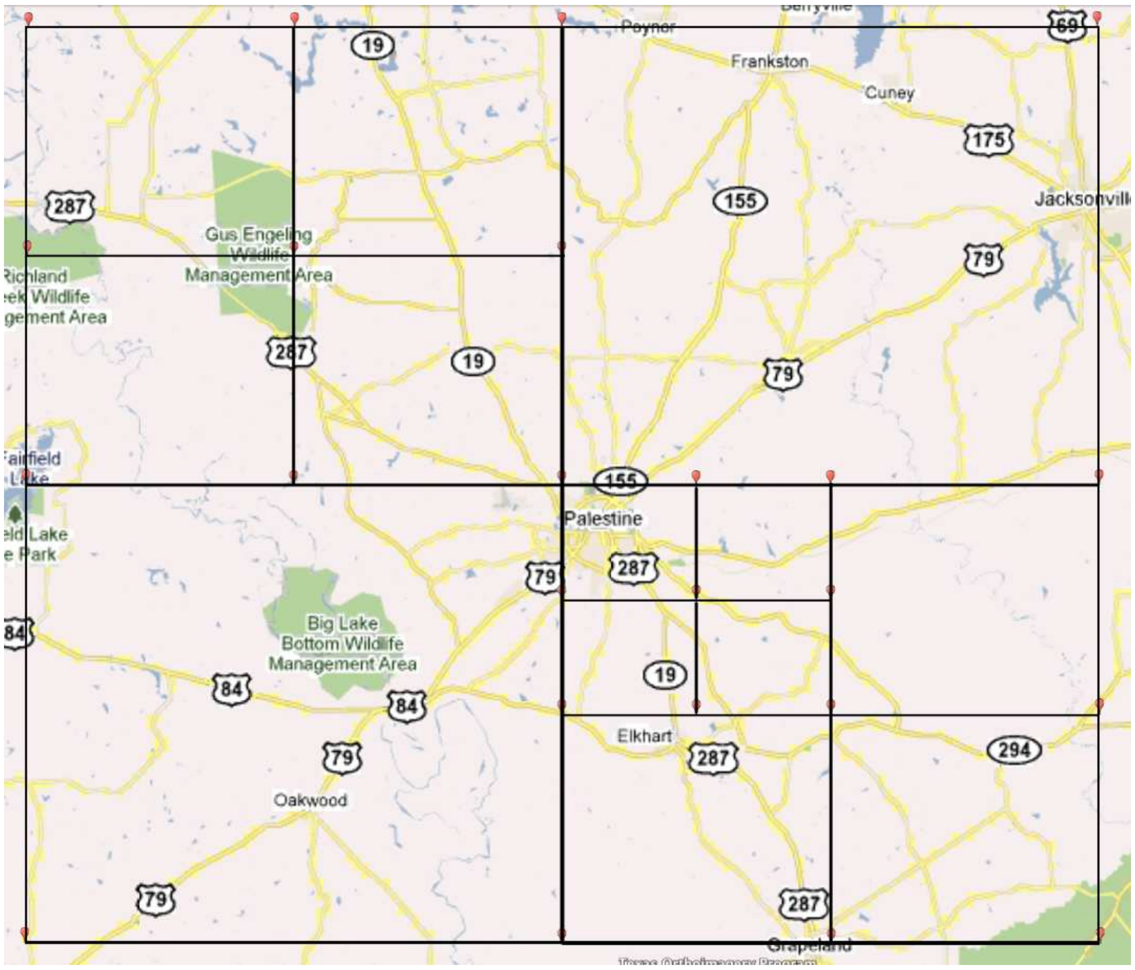
Poynor
Frankston
Cuney
Jacksonville
Gus Engeling Wildlife Management Area
Richland Creek Wildlife Management Area
Fairfield Lake
Fairfield Lake State Park
Palestine
Big Lake Bottom Wildlife Management Area
Oakwood
Elkhart
Grapeland
19
287
155
175
79
69
84
294

Figure 2.6. The Andrews County VHC Map laid over the geographic map.

CHAPTER 3

EXPERIMENT SETUP

All of the previously discussed methods provide adequate privacy and anonymity when a mobile device is accessing an LBS. Most of those methods, except CAP, rely on a trusted third party, such as an anonymizer [26, 25], a relay agent [24], the LBS itself [33, 25], or any combination of the three. CAP does all of its context calculations offline using a desktop or laptop computer with plentiful resources, then transfers the compressed map data to the mobile device so that it can recreate the map and perform location perturbation on demand. It allows a user to send a 'fake' location to the LBS without the LBS being able to determine that it is being lied to. The calculation and storage analysis from CAP [5, 37] suggest that it is efficient enough to implement and experiment on modern devices without the need to rely a trusted third party, reducing the number of places an attacker can harvest information. The POI and LBS accuracy analysis show to be accurate for lower privacy levels, but when the privacy levels are increased, the results worsen to the point that a user may find it too inconvenient to use. Users have to make a trade-off between privacy and convenience. The goal then becomes, to minimize the cost of inconvenience.

We chose to implement and evaluate CAP for a number of reasons. As stated previously, it does not rely on a trusted third party, is efficient enough to run on a modern smartphone with constrained resources, and the data used for testing is freely and easily available from the U.S. Census Bureau [36]. Recent improvements in mobile and cellular technology have allowed many people to own smartphones and other mobile devices that have Internet access and are capable of determining their current location. Most importantly, we feel that the POI results from the LBS at the higher privacy levels can be improved.

In our experiment, we implement and test the location perturbation component of CAP. In its current form, CAP protects against information disclosure in two ways; location perturbation and anonymous routing. We chose not to experiment with the anonymous routing component based on the assumption that most smartphone users access the Internet using a cellular connection as opposed to a wireless hotspot with direct Internet access. When using a modern cellular connection, the cellular provider is always aware of the device's location to within 300 meters as required by the FCC [6]. If the user would like to use a wireless Internet connection, or hotspot, they can install the appropriate porting of Tor for their mobile device, but its implementation is beyond the scope of this thesis.

3.1 Assumptions

Before we describe the the experiment setup, there are a number of assumptions that need to be made. To begin, we assume that the LBS and the cellular provider are not colluding, and the cellular provider will not inform the LBS that the user is using a 'fake' location. We further assume that the user is only contacting the LBS by using the cellular Internet connection. Smartphones and other cellular devices are obviously highly mobile and have the potential to be constantly moving, frequently switching between celluar towers that the device is communicating with. In order to maintain this mobility as these devices change towers, cellular providers utilize a type of address translation similar to DNAT [1] when these devices access the Internet [6, 38]. The cellular provider issues an IP address (device address) to the mobile device that is internal to the cellular network and can potentially change as the device switches

[1]Dynamic Network Address Translation

between cellular towers. The provider also creates an external gateway address for the mobile device which does not change throughout the connection. During this time, the cellular provider does dynamic address translation between the dynamic device address and the gateway address. Previous examinations of apache log files that have been accessed by mobile devices show an IP address from different geographic regions, such as Milwaukee, WI when the device is actually located in Dallas, TX. This leads us to the assumption that as long as access to the LBS is made using cellular networks and the cellular provider is not colluding with the LBS, the anonymous routing component is not needed. The last assumption that we make is that the user does not care if the attacker is able to determine which city the user is in. For example, the users general area, i.e. Arlington, TX, is not sensitive to disclosure, but the user wants to hide their precise location from the LBS, i.e. 416 Yates St #600, Arlington, TX.

3.2 Privacy Measures

The traditional models of k-anonymity are not directly applicable to CAP because it does not utilize a trusted third party anonymizer that maintains a global view of active users. This means that we must estimate the values of k, although in areas of high population, fluctuations that differ from the estimate will still provide adequate k-anonymity levels. CAP, as described in [5] and [37] use road density data obtained from the U.S. Census Bureau in the form of Topically Integrated Geographic Encoding and Referencing (TIGER) files [36]. The U.S. Census Bureau determines the population of an area by taking surveys every 10 years and updates the TIGER files annually. In 2008 they changed the format of the TIGER files from ASCII text to shapefiles for use with Geographic Information System (GIS) software, making the data for 2006 the latest year the data is available in text format. As of 2008, the

U.S. Census Bureau estimates that the population of Dallas county, TX is approximately 2.4 million [39] and has an area of approximately 908 square miles [36]. If we combine the population information with the total road density for Dallas county at approximately 12,941 miles, this gives us a approximately 183 people per mile of road distance. While still not an exact measure of population, we feel that it is a close enough approximation for our uses based on the assumption that road density and popoulation have an approximately linear correlation based on the findings of [32]. Obviously, the population of Dallas County is not uniformly distributed and several areas will be known to fluctuate over a small periods of time (i.e. Downtown). In this case, the user can make observations about their current surroundings and adjust their privacy settings as needed.

Using definition 3.2.1, originally defined by Pingley et. al. in [5], the measure is defined over the whole population in the area where the user is hidden. This definition implies that a privacy-preserving scheme will satisfy N-confidentiality iff an adversary cannot determine the users actual location between two locations in a region of population N.

Definition 3.2.1 *A privacy-preserving scheme which perturbs userPos from x to R(x) satisfies N-camouflage iff there exists a region C of population at least N, such that for any subregion C' ⊆ C,* $\Pr\{x \in C' \mid R(x)\} = |C'|/|C|$, *where* $|\cdot|$ *is the area of the region.*

3.3 Efficiency

Recall from Chapter 2.3.3.3, the location perturbation component of CAP is executed in two stages, an offline stage and an online stage. The offline stage does most of the resource intensive calculations, creates the VHC-map, then compresses it for transfer to the mobile device. The online stage then recreates the VHC-map in memory in order to do efficient location perturbation on demand. In this section, we review the algorithms and their efficiency when used by CAP while creating, storing and finally recreating the VHC-map

The first algorithm we review is the construction of the VHC-map in the offline stage.

Algorithm 1: Offline Construction of VHC-Mapping

```
Input:  $Map$ consisting of road coordinates,
        $C$ as the rectangular boundary of the $Map$
Output: $VHCMap$ a serialized binary file representing the VHC-map
1  Store $C \| $BuildTree($C$) as the Hilbert Curve representaion of $Map$.
2  function BuildTree($C$)
3    if total road length in $C \geq \mu \cdot$edge length of $C$ then
4        Partition $C$ equally into $C_{nw}$, $C_{ne}$, $C_{sw}$, $C_{se}$
5        for $i = C_{nw}, C_{ne}, C_{sw}, C_{se}$ do
6            return $0 \|$BuildTree($C_i$)
7        end
8    else
9        return 1
10   end
```

In Algorithm 1, $\|$ is used to represent concatenation. The original map is partitioned (Line 3) using the Minimum Density Rule described in section 2.3.3.2 and stores the quad-tree as a synchronized binary file to be transferred to the mobile device (Lines 6 and 9). The computational complexity of the VHC-map creation is

$O(n)$, with n being defined as the amount of road coordinates contained in the input file.

Algorithm 2: Online Location Perturbation

```
  Input:  Pre-computed VHC-mapping file $VHCMap$
          $X$ as the user's actual 2-d location
  Output: Perturbed 2-d point
1 Load a quad-tree $T$ of the partition from $VHCMap$
2 Build Hilbert curve that connects all leaf nodes in $T$ and assign 1-d value
  range for each leaf cell.
3 Wait to recieve an actual 2-d location
4     Map $X$ to 1-d $F(X)$ based on the Hilbert curve
5     Generate random noise $r$ based on a uniform distribution in the range of
      $[-\sigma, \sigma]$ and add it to the 1-d value.
6     Map the new 1-d value back a 2-d cell.
7     Choose a random point within the new 2-d cell.
8     Output $R(X) = F^{-1}(F(X) + r)$.
9 Goto 3.
```

Algorithm 2 depicts the process of reading the binary file $VHCMap$, recreating the VHC-map into memory on the mobile device, and perturbing the user's actual location. Reading of the binary file and recreating $VHCMap$ in memory (Lines 1 and 2) has a computational complexity of $O(n)$, but only needs to be done once when recreating the VHC-map. The perturbation of each location has a complexity of $O(\log n)$. In the case of reading the binary file and recreating the VHC map, n is defined as the number of nodes in the quad-tree of the VHC-map, or the number of bits in the binary file.

The location pertubation algorithm takes the user's actual location as input to map it to the 1-d Hilbert curve value with $F(X)$. The algorithm then adds random noise r based on the uniform distribution and privacy levels to obtain the new 1-d value with $F(X) + r$. This new 1-d value is then mapped back to a VHC cell corresponding to the 1-d Hilbert curve value. A random point is then chosen within the new VHC cell and returned as the perturbed location (Lines 4 through 8).

3.4 Cloaking Region

One of the most noticable difference between CAP and the other location privacy methods discussed in this thesis is that CAP obscures the user's location to another single point, while the other methods obscure the users location by creating a cloaking region around the users actual location. Noticing this, we modified the online component of CAP to create a cloaking region around or near the users actual location. The algorithm for creating the cloaking region is described in Algorithm 3. Creation of the cloaking region starts out in the same manner as determining the perturbed location up to step 4. The user's actual 2-d location is mapped to the 1-d value on the Hilbert curve. The algorithm then generates two random numbers of opposite signs r_1, and r_2. It then adds each r_1 and r_2 to the original 1-d value, to get two new 1-d values, $F(X) + r_1$ and $F(x) + r_2$ (keep in mind that r_1 and r_2 have opposite signs). These two new 1-d values map to two different VHC nodes, of which, the maximum and minimum latitudes and longitudes between the two are kept and used to create the new cloaking region. This cloaking region is then used for the LBS query.

Due to the nature of the Hilbert curve, and only using the two nodes corresponding to the two perturbed VHC nodes, there is a possiblity of the user's actual location being located outside of the generated cloaking region. This fact can potentially lead the LBS to provide poorer results (i.e. POIs with a higher rank distance) since most publicly available LBS only return the POIs that are located within the cloaking region used with the queries. If an attacker is interested in narrowing down the user's potential actual location and has information of the creation of the VHC map that is being used, they will still end up with a larger area to search due to the Hilbert curve leaving the cells used by the cloaking region. This cloaking area is still not as large as the potential cloaking area created by the two extremes of $R_1(X)$ and $R_2(X)$ because the attacker is still able to determine the range of 1-d values from the interior edges of the VHC cells used to create the cloaking region. In spite of this fact, even the smallest possible cloaking area that the attacker would be able to calculate is still no smaller than the region created by the extremes of the cloaking region formed by the VHC cells mapped to by $R_1(X)$ and $R_2(X)$.

3.5 Evaluation Methods

To evaluate CAP, we examine the accuracy of the POIs returned from the LBS, the distance of the perturbed locations from the actual location compared to the different privacy levels (i.e. σ), and the speed of location perturbation on a modern midrange smartphone [2]. In [5], they point out that an LBS query is similar to a top-k query over a spatial database. Based on that similarity, they propose a number of different methods to evaluate the results obtained from the LBS, including rank

[2]By midrange, we mean a non-top of the line smartphone capable of determining its GPS location and accessing the Internet. The specifications of the device used for testing are given in Chapter 4

Algorithm 3: Online Cloaking Region Generation

```
Input:  Pre-computed VHC-mapping file $VHCMap$
        $X$ as the user's actual 2-d location
Output:  maximum latitude,
         minimum latitude,
         maximum longitude,
         minimum longitude
1 Load a quad-tree $T$ of the partition from $VHCMap$
2 Build Hilbert curve that connects all leaf nodes in $T$ and assign 1-d value
  range for each leaf cell.
3 Wait to recieve an actual 2-d location
4     Map $X$ to 1-d $F(X)$ based on the Hilbert curve
5     Generate random noise $r_1$ and $r_2$ of opposite signs based on a uniform
      distribution in the range of $[-\sigma, \sigma]$
6     Add each random value to the 1-d value $F(X)$ to get $R_1(X)$ and $R_2(X)$.
      Map each new 1-d value back to two new 2-d cells.
7     Find maximum latitude, minimum latitude, maximum longitude,
      minimum longitude of the two cells.
8     Output maximum latitude, minimum latitude, maximum longitude,
      minimum longitude.
9 Goto 3.
```

distance [3], true positive rate [4] and more. In their experimental tests, they used rank distance and true positive rate compared to privacy levels and *road density index*. *Road density index* is defined in [5] as the level of the leaf node that contains the location being referenced, with the root having level 1 (e.g. in Figure 2.5, node 9 has a *road density index* of 4).

In CAP [5], the results they obtained when measuring LBS accuracy seem to indicate, as expected, decent results with the lower privacy levels, yet they degrade with the higher privacy levels (i.e. higher noise levels). We downloaded their prototype in the spring of 2010 and began manual testing and visual inspection of the the location perturbation component of CAP. Their implementation accesses an LBS they created, populated with 800 POIs downloaded from *http://www.gps-data-team.com/poi/*. Using a location in Washington D.C. and various query terms, we tested each of the privacy levels that were available in the user interface. Upon visual inspection of their prototype, the accuracy of the returned POIs seem to be lower than what their results seem to indicate. In most cases, when the privacy settings were high, the POIs returned from the LBS were many times completely disjoint from the POIs returned from the LBS when low or no privacy settings were used. In their calculations determining LBS accuracy, they use rank distance of the POI tuples returned from the LBS based on Definition 3.5.1.

Definition 3.5.1 *The average rank distance of a privacy-preserving scheme that perturbs a user's position from x to R(x) is:*

[3]the rank difference between the returned tuple and its true rank based on a query with the user's actual location.

[4]The probability that a returned tuple based on the pertubed location is in fact a true top-k tuple.

$$l_r(x) = AVG_{t \in q(x)}(\mid rank(t, q(x)) - rank(t, R(q(x))) \mid).$$

where $AVG(\cdot)$ represents the average value, q(x) is the LBS query answer when the user's position = x, and rank(t, q(x)) is The rank of tuple t in the returned answer q(x).

CHAPTER 4

CAP IMPLEMENTATION

As stated numerous times throughout this thesis, CAP is implemented in two stages. This chapter discusses the hardware and software we used for our implementation of the two stages.

4.1 Hardware Environment

The offline creation of the VHC-map was done on a desktop computer using an Intel© Core™2 Duo e6600 rated at 2.4 GHz with a 1066 MHz bus speed and 4 GB of memory. The mobile device used in the experiment is the G1 smartphone from T-Mobile running Google's Android 1.6 Operating System. The smartphone used in the experiments contains a Qualcomm®ARM processor that is rated at 528 Mhz, includes 192 MB of DDR SDRAM, and 1 GB of flash memory. The phone was originally released to the public in October 2008. The current pace of techonology improvements for modern smartphones makes this device suitable for testing since it is not a bottom of the line device, yet is by far not the most advanced.

4.2 Software

The Android operating system is Java based so it made natural sense to implement the offline stage in Java as well, although other languages are suitable for use as long as they provide the output in the appropriate format. Both the offline component and the Android application were developed using Eclipse. The offline component was executed using the Java Virtual Machine (JVM) with the default settings which has a maximum available setting of 256 MB. The offline stage takes as input road coordinates obtained from 2006 TIGER files published by the U.S. Census Bureau [36]. The 2006 TIGER files were the last ones published in text format before the U.S. Census Bureau started publishing them as shape files for use with GIS software. The shape files still contain the same context information, but the new format makes extracting the information more conplex. In spite of the fact that the data is a few years out of date, due to the slow pace of new road construction we feel this data is still suitable for our purposes. Addionally, these files are the same ones used in the original prototype implementation of CAP [5] allowing for a more direct comparison of the CAP prototype in a controlled environment and our practical implementation using real world components.

After creating the VHC-map using Algorithm 1, the offline component outputs the binary $VHCMap$ file in a string representation. String representation of the binary Map does indeed use more memory than a raw binary representation, although this amount is overshadowed by the memory requirements of the recreated VHC-map on the mobile device. The recreated VHC-map needs to store the minimum and maximum 2-d coordinates for each node, the 1-d Hilbert curve range for each node, references to its parent and children nodes (if it has any), its sibling order (i.e. which child it is) and other minor administration variables. These other variables that each node must keep track of have a much higher memory requirement, making the method of storage of the binary array a moot concern.

The online stage of location perturbation, as stated previously, is deployed as an Android application on T-mobile's G1. It provides an interface that allows the user to input the needed data to make queries against an LBS. It allows the user to input search terms, privacy level, location acquisition methods (GPS, triangulation, or manual), search radius and includes functionality for the user to see the perturbed location before sending the request to the LBS.

Behind the interface, the system rebuilds the VHC-map in the same manner and uses the same data structures used by the offline stage. It determines which VHC-map to use based on the user's *actual location*, comparing it to the bounding coordinates of the pre-computed VHC-map files. It loads and recreates the appropriate VHC-map in the memory of the mobile device. The application then calculates the perturbed location, with the amount of perturbation based on the user's requested privacy level. Upon return of the perturbed location, the application allows the user to see the perturbed location so that they can make sure the perturbed location is far enough away from the user's actual location. We added this functionality because the noise generated by the random number generator still has a probability of being very small, putting the perturbed location very near to the user's actual location. Furthermore, in [27], Liu points out that a k-anonymity privacy scheme is not very effective when the $k-1$ other users are in the same location and while the application and the user know that the perturbed location is randomized based on a uniform distribution, an adversary might be unaware of this and assume that the perturbed location is the users actual location. After obtaining a suitable perturbed location, the user then presses a button to send the query to the LBS. The application receives the POI response from the LBS and sorts the POIs in increasing distance from the users actual location before presenting them to the user.

CHAPTER 5

METRICS AND RESULTS

To determine the effectiveness of CAP, we ran a number of tests on our implementation. Since the main goal of querying an LBS is to find the most relevent POIs that are closest to the user, this is where we focused our tests. We examined the usable sizes of areas with different road densities, the rank distance and true positive rates of the POIs returned from different LBS and finally introduce a slight modification to the original algorithm to create a cloaking region similar to other methods of previously proposed location privacy methods [24, 25, 26, 33]. As expected, most results follow along the original implementation of CAP.

5.1 Metrics Used

To provide the application with the most granular control over the area as possible, we used VHC maps for each county at the lowest granularity possible that would fit within the memory constraints of the smartphone (i.e. largest bit count and tree height). We chose three counties to test on so that we could obtain a broad set of results. We chose Dallas county because of its high population and road density, Wise county because of its low population and road density, and finally we chose Middlesex County, Massachusettes so that we could have a comparison against the results in [5].

For our accuracy measurements, we tested numerous publicly available LBS in an attempt to find one that would consistently provide us with enough POI results for an objective comparison of tests based in different areas. Our results are based on queries against Yelp.com and MapQuest.com since they were both able to provide enough POIs in every county for us to be able to draw some reasonable conclusions. We used rank distance as defined in Definition 3.2.1 and True Positive Rate for our POI accuracy measurements, which are the same ones used in [5]

5.1.1 Map Creation and Memory Usage

For each county tested, we created VHC mappings of decreasing granularities (μ) starting at 40 and decreasing by 5 until the JVM crashed due to Out of Memory errors. At this point, we determined the lowest granularity of each county that would successfully create a VHC map. These values and their corresponding tree heights for each county are displayed in Figure 5.1. The corresponding usable bit counts for Dallas, Wise and Middlesex counties are displayed in Figures 5.2, 5.3, 5.4 respectively. As expected and stated previously in [5], these values all increase exponentially. This is not very well indicated for Dallas county in Figure 5.2, because the next lowest granularity of 16 produces a node count of 50,5116 which consumes more memory than is available. Figures 5.2, 5.3, 5.4 show the usable tree heights for each county that were able to work under the tight memory constraints of the smartphone. Wise, Middlesex and Dallas counties were able to use a tree height of 17, 18, and 8 at offline creation granularities of 9, 15, and 17, respectively, before running out of memory.

5.2 Location Perturbation Distance and Cloaking Region Areas

In this section we discuss the POI accuracy of querying an LBS with a single perturbed location, and a cloaking region generated using the modified CAP algorithm. For the single perturbed location tests, we compared Dallas, Wise and Middlesex counties, in both high and low road density areas. For the cloaking region tests, we look at Dallas County in an area of High density (i.e. Downtown Dallas) because of

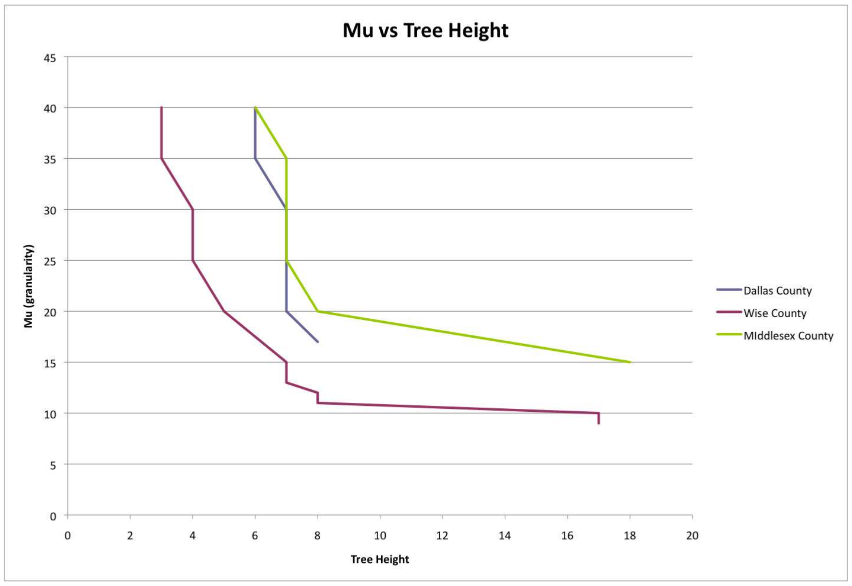

Figure 5.1. Mu vs Tree Height for Dallas, Wise and Middlesex County.

the large number of restaurants in the area, yet we compare the results returned from both Yelp.com and MapQuest.com.

5.2.1 Single Point Location Perturbation Distance

To evaluate the average distances of the perturbed locations in each type of area, we executed 100 tests in each area at each privacy level. These results can be viewed in Figure 5.5. The perturbation distances for Dallas and Wise counties follow a gentle linear increase, while the results for Middlesex, MA seem to have a sharp increase between the privacy levels of 0.5 and 1. This can be attributed to the differences in the layout of the different counties which can be noticed by visual inspection of a geographic road map. Dallas and Wise each only contain one area of high road density, near the city that it contains, and the densities decrease the further out from the city. Middlesex county has a more evenly dispersed road density throughout the county, yet still has small pockets of low road densities.

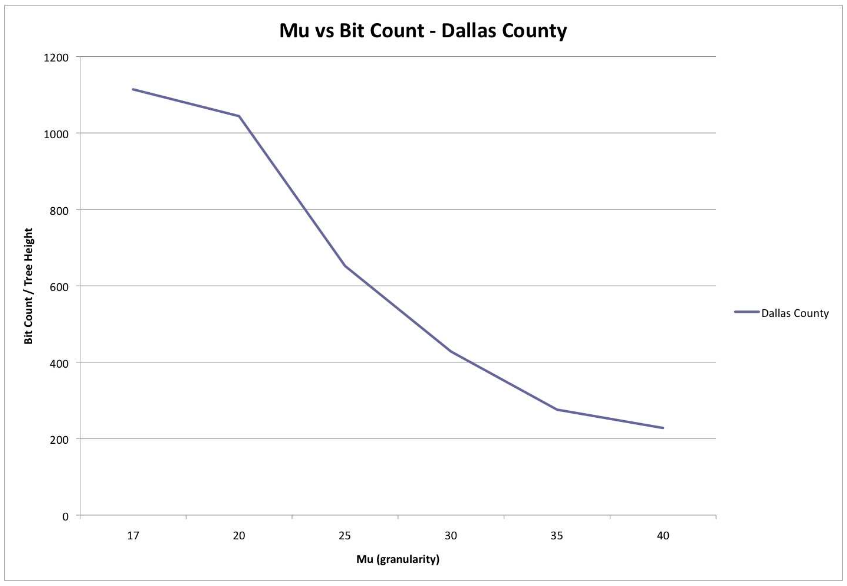

Figure 5.2. Mu vs Bit Count for Dallas County.

5.2.2 Cloaking Region Areas

So that we may have an understanding of the size of the areas being searched when querying the LBS with a cloaking region, we examine the areas of the cloaking region used when querying (i.e. constructed of the two nodes at the end-points of the 1-d perturbed 1-d values) as well as the total area of the possible cloaking area (i.e. the area created by every node along the 1-d range). Figures 5.6 and 5.7 show that the algorithm follows the same exponential curve, although the total possible cloaking areas show to be much larger. This is somewhat expected.

5.3 Results For Point of Interest Accuracy

For our POI tests, we executed 10 queries, per privacy level, under the same conditions as the distance and cloaking region analysis described in [5] For the LBS, we used MapQuest.com and Yelp.com because they seemed to provide adequate POI results and were available for testing. We used the query term of ”Restaurant” in

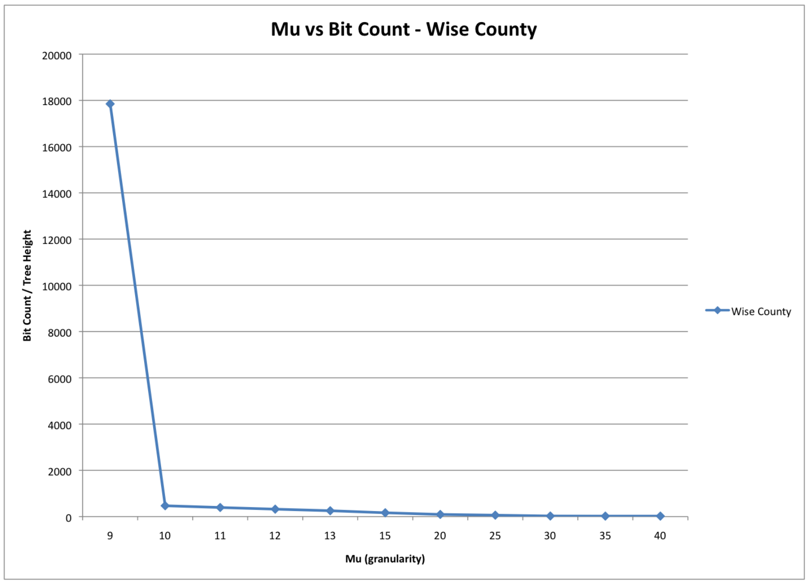

Figure 5.3. Mu vs Tree Height for Wise County.

each test because of their ubiquitousness in most areas and because of the high POI count returned from the two LBS.

5.3.1 Single Point Location Perturbation

The rank distances of the POI results for the single point location perturbation are displayed in Figure 5.8. The results shown are much higher than the rank distances found in [5]. The true positive rate of the POI results for the single point location perturbation are displayed in Figure 5.9. Again, these results are much less desireable than those found in [5]. The results found in these tests coincide with the results that were obtained by doing manual tests on the CAP prototype from the authors of [5] and seem to indicate lesser results than found in the paper.

5.3.2 Cloaking Region Area

The rank distances of the POI results for the cloaking regions are displayed in Figure 5.8 and the True Positive rate is displayed in Figure 5.11. The rank distances of

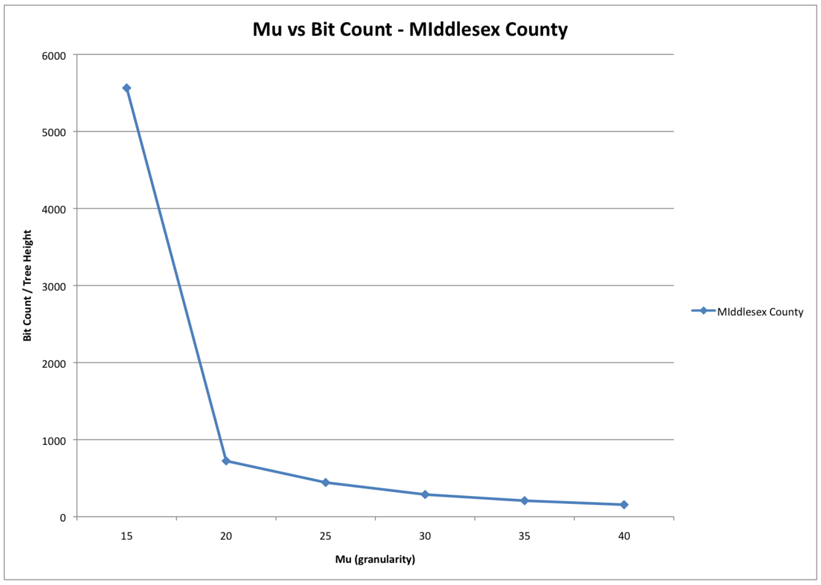

Figure 5.4. Mu vs Tree Height for Middlesex County.

results when a cloaking region appear to be magnitudes lower than the rank distances for the single perturbed location. The True Positive Rate of the results for a cloaking region are much greater than the single perturbed location up until the privacy levels between 1 and 4. At this point, the True positive rate appears to drop off sharply.

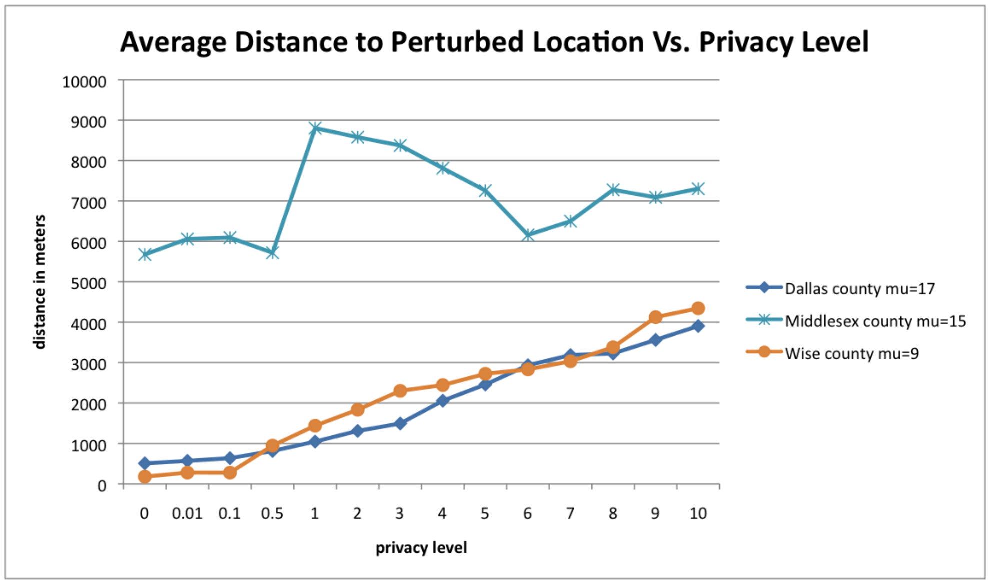

Figure 5.5. Average distance from the user's actual location to the perturbed location for each privacy level.

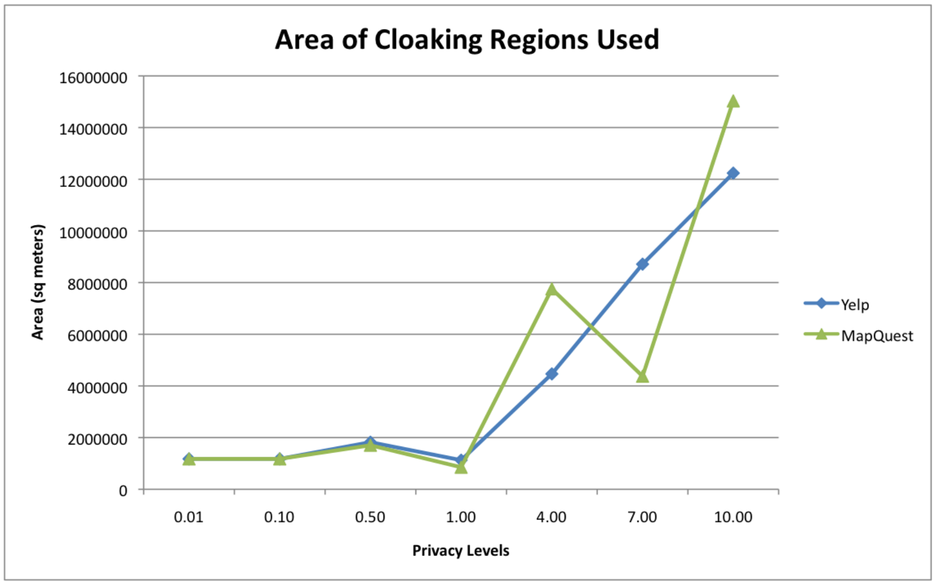

Figure 5.6. The average area of the cloaking regions used with the LBS for each privacy level.

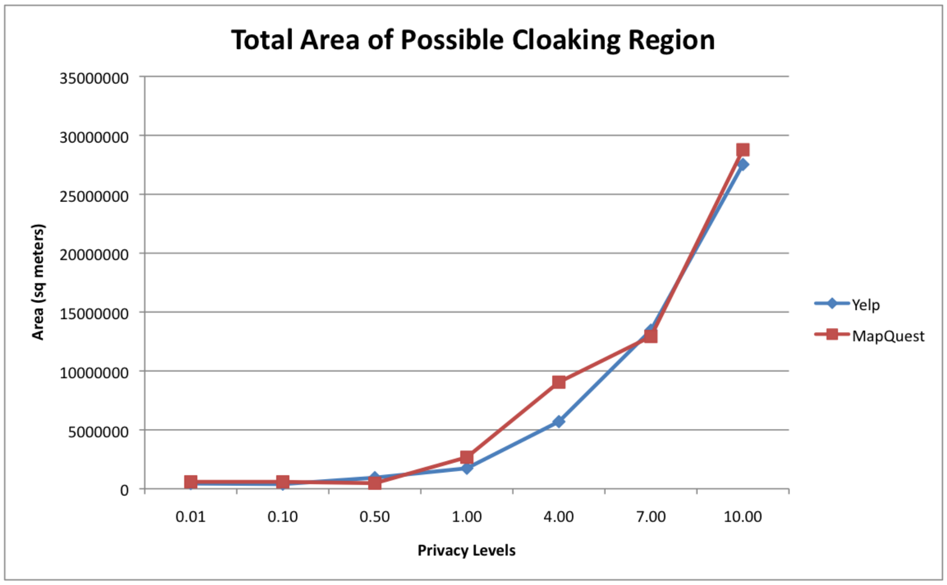

Figure 5.7. The total possible area of the cloaking regions used with the LBS for each privacy level.

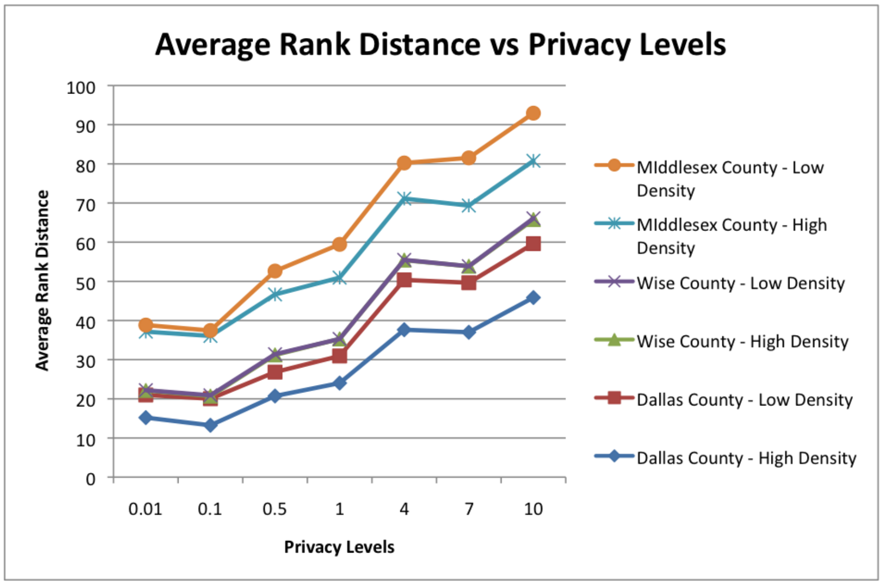

Figure 5.8. The average rank distances for the POIs for each privacy level for a single perturbed location.

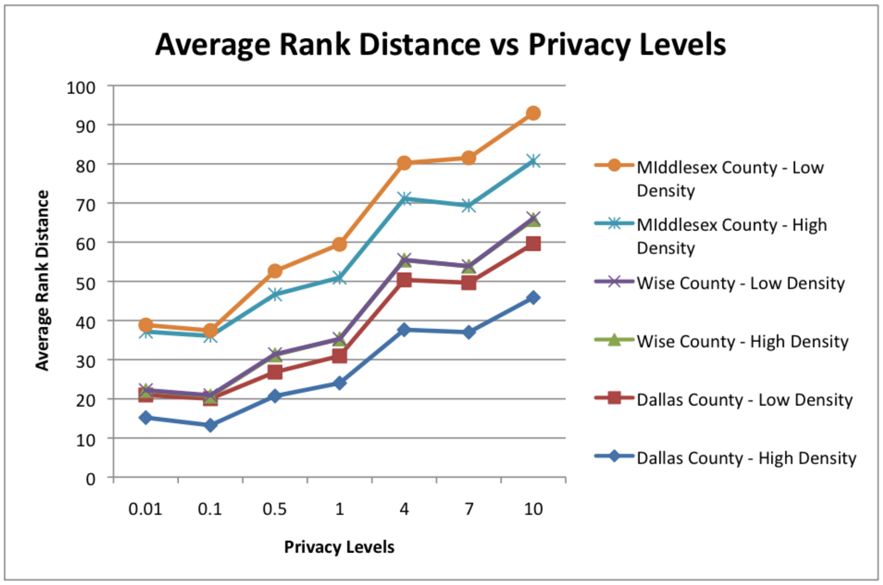

Figure 5.9. The True Positive Rates for the POIs at each privacy level for a single perturbed location.

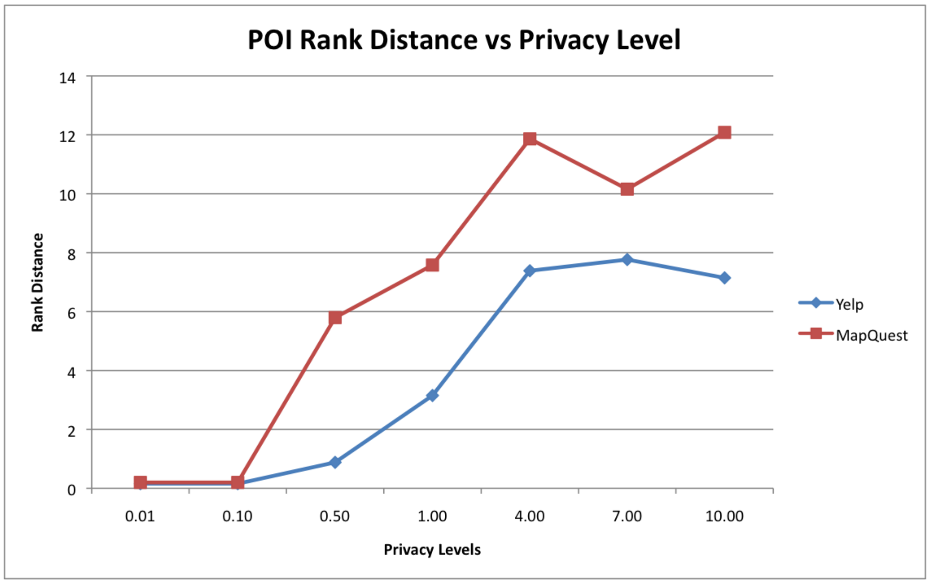

Figure 5.10. The average rank distances for the POIs for each privacy level when a cloaking region is used with an LBS.

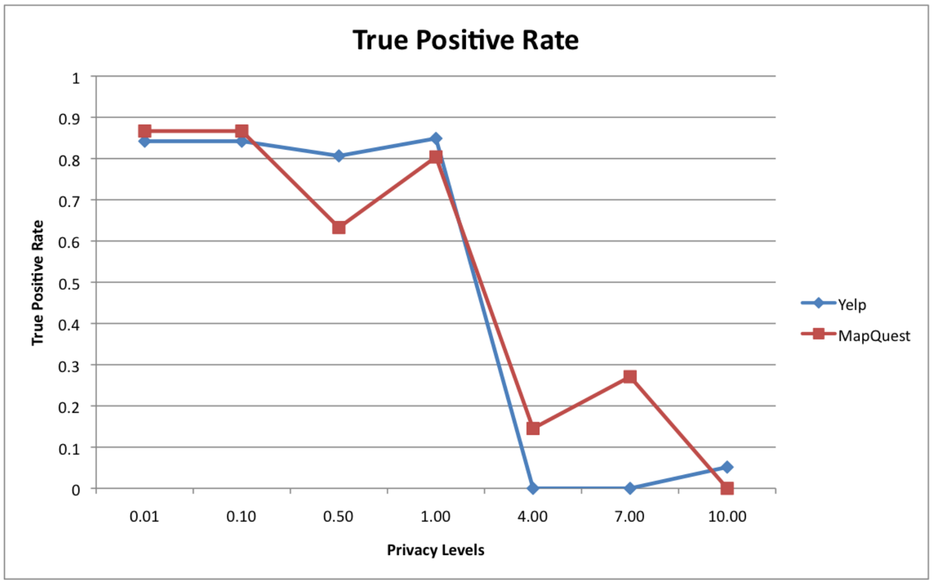

Figure 5.11. The True Positive Rates for the POIs at each privacy level when a claoking region is used when an LBS.

CHAPTER 6

CONCLUSION

In this thesis, we have taken a survey of several location privacy methods intended for use on mobile devices. While most of the proposed methods rely on a trusted third party to protect a user's location privacy, CAP does not. Trusted third parties allow for another point for an attacker to exploit, so reducing as many as possible gives an attacker less points of exploit. CAP reduces the number of third parties and demonstrated satisfactory results in a fully enclosed controlled environment (i.e. homegrown LBS). The current availability of real world components and LBS make this a prime time to test this method in a real world envronment. We implemented CAP on a modern smartphone and tested the location perturbation against publicly available LBS and made a slight modification to make it act more like the other proposed methods of location privacy, yet were still able to maintain the reduced the trusted third party.

To measure our results, we used the same accuracy measurements as [5] and found that they were actually a bit less desireablei, although still effecitve at the lower privacy levels. We found that implementing CAP as a cloaking region generating method of location privacy still has the added benefit of the reduced trusted third party and efficiency of CAP, but yeilds more desireable results. In addition to accuracy measurements, we took a look at the memory usages of CAP on a commercically available smartphone. [5] claims to be the first end-to-end location privacy and lays the foundation for other ongoing studies of location privacy preserving methods. In spite of less desireable results than indicated, we have shown that it still produces satisfactory results in a real world scenario, and have taken it a step further to adapt CAP so that it acts more similar to other location privacy methods by creating a cloaking region to be sent to the LBS. The use of this cloaking region appears to have increased the accuracy of CAP significantly and still provides the same level of privacy as both the original proposal as well as the other methods.

REFERENCES

[1] J. Nixon. (2010, July) 100 million facebook pages leaked on torrent site. [Online]. Available: http://www.thinq.co.uk/2010/7/28/100-million-facebook-pages-leaked-torrent-site/

[2] E. Price. (2010, July) 100m facebook profiles now available for download. [Online]. Available: http://www.pcworld.com/article/202126/100m_facebook_profiles_now_available_for_download.html

[3] I. Paul. (2010, July) The facebook data torrent debacle. [Online]. Available: http://www.pcworld.com/article/202167/the_facebook_data_torrent_debacle_qanda.html

[4] D. Goldman. (2010, October) Rapleaf is selling your identity. [Online]. Available: http://money.cnn.com/2010/10/21/technology/rapleaf/

[5] A. Pingley, W. Yu, N. Zhang, X. Fu, and W. Zhao, "Cap: A context-aware privacy protection system for location-based services," in *Distributed Computing Systems, 2009. ICDCS '09. 29th IEEE International Conference on*, june 2009, pp. 49 –57.

[6] Y. K. Kim and R. Prasad, *4G Roadmap and Emerging Communication Technologies.* Artech House, 2005.

[7] R. Rogers, J. Lombardo, Z. Mednieks, and B. Meike, *Android Application Development.* O'Reilly, 2009.

[8] (2011) Rave guardian. rave mobile safety. [Online]. Available: www.ravewireless.com/products/raveguardian

[9] (2011) Enhanced 9-1-1 - wireless services. [Online]. Available: www.fcc.gov/pshs/services/911-services/enhanced911/

[10] (2011) Facebook places. [Online]. Available: www.facebook.com/places/

[11] (2011) Foursquare. [Online]. Available: foursquare.com/

[12] (2011) Gowalla. [Online]. Available: gowalla.com/

[13] (2011) Zhiing. [Online]. Available: www.zhiing.com/

[14] (2011) Google places. [Online]. Available: www.google.com/mobile/places/

[15] (2011) Yahoo local search web services. [Online]. Available: http://developer.yahoo.com/search/local/V3/localSearch.html

[16] (2011) Yelp. [Online]. Available: www.yelp.com

[17] R. H. Gting, "An introduction to spatial database systems," *The VLDB Journal*, vol. 3, pp. 357–399, 1994.

[18] J. H. Kang, W. Welbourne, B. Stewart, and G. Borriello, "Extracting places from traces of locations," in *Proceedings of the 2nd ACM international workshop on Wireless mobile applications and services on WLAN hotspots*, ser. WMASH '04. New York, NY, USA: ACM, 2004, pp. 110–118.

[19] The place lab. [Online]. Available: http://www.placelab.org

[20] J. Krumm, "Inference attacks on location tracks," in *In Proceedings of the Fifth International Conference on Pervasive Computing (Pervasive), volume 4480 of LNCS.* Springer-Verlag, 2007, pp. 127–143.

[21] L. Liao, D. Fox, and H. Kautz, "Location-based activity recognition using relational Markov networks," in *IJCAI'05: Proceedings of the 19th international joint conference on Artificial intelligence.* Morgan Kaufmann Publishers Inc., 2005, pp. 773–778.

[22] M. Gruteser, D. Grunwalddepartment, and C. Science, "Anonymous usage of location-based services through spatial and temporal cloaking," 2003, pp. 31–42.

[23] U. Hengartner, "Hiding location information from location-based services," in *Mobile Data Management, 2007 International Conference on*, May 2007, pp. 268 –272.

[24] C.-Y. Chow, M. F. Mokbel, and X. Liu, "A peer-to-peer spatial cloaking algorithm for anonymous location-based service," in *Proceedings of the 14th annual ACM international symposium on Advances in geographic information systems*, ser. GIS '06. New York, NY, USA: ACM, 2006, pp. 171–178.

[25] M. Mokbel, "Towards privacy-aware location-based database servers," in *Data Engineering Workshops, 2006. Proceedings. 22nd International Conference on*, 2006, p. 93.

[26] M. F. Mokbel, C.-Y. Chow, and W. G. Aref, "The new casper: query processing for location services without compromising privacy," in *Proceedings of the 32nd international conference on Very large data bases*, ser. VLDB '06. VLDB Endowment, 2006, pp. 763–774.

[27] L. Liu, "From data privacy to location privacy: models and algorithms," in *Proceedings of the 33rd international conference on Very large data bases*, ser. VLDB '07. VLDB Endowment, 2007, pp. 1429–1430.

[28] F. Grace. (2003, February) Stalker victims should check for gps. [Online]. Available: http://www.cbsnews.com/stories/2003/02/06/tech/main539596.shtml

[29] A. Press. (2002, December) Authorities: Gps system used to stalk woman. [Online]. Available: http://www.usatoday.com/tech/news/2002-12-30-gps-stalker_x.htm

[30] J. Domingo-Ferrer and V. Torra, "A critique of k-anonymity and some of its enhancements," in *Proceedings of the 2008 Third International Conference on Availability, Reliability and Security.* IEEE Computer Society, 2008, pp. 990–993.

[31] K. LeFevre, D. DeWitt, and R. Ramakrishnan, "Mondrian multidimensional k-anonymity," in *Data Engineering, 2006. ICDE '06. Proceedings of the 22nd International Conference on*, 2006, p. 25.

[32] D. R. Glover and J. L. Simon, "The effect of population density on infrastructure: The case of road building," *Economic Development and Cultural Change*, vol. 23, no. 3, pp. 453–68, April 1975.

[33] F. Olumofin, P. Tysowski, I. Goldberg, and U. Hengartner, "Achieving efficient query privacy for location based services," in *Privacy Enhancing Technologies*, ser. Lecture Notes in Computer Science, M. Atallah and N. Hopper, Eds. Springer Berlin / Heidelberg, 2010, vol. 6205, pp. 93–110.

[34] H. V. Jagadish, "Analysis of the hilbert curve for representing two-dimensional space," *Information Processing Letters*, vol. 62, no. 1, pp. 17 – 22, 1997.

[35] Z. Song and N. Roussopoulos, "Using hilbert curve in image storing and retrieving," *Information Systems*, vol. 27, no. 8, pp. 523 – 536, 2002.

[36] U. C. Bureau. (2006) 2006 second edition tiger/line files. [Online]. Available: http://www.census.gov/geo/www/tiger/tiger2006se/tgr2006se.html

[37] A. Pingley, "Cap: A context-aware privacy protection system for location-based services," Master's thesis, University of Texas at Arlington, 2008.

[38] T. Janevski, *Traffic Analysis and Design of Wireless IP Networks.* Artech House, 2003.

[39] U. C. Bureau. (2010) Population estimates. [Online]. Available: http://www.census.gov/popest/estimates.html

BIOGRAPHICAL STATEMENT

James Spargo was born on Spetember 4, 1976 at a private location in the state of Maryland. He received his Bachelors of Science in Computer Science and Engineering from the University of Texas at Arlington in May 2007. In 2011, he received his Masters of Science in Computer Science with a Thesis focused on Location Privacy with smartphones from the University of Texas at Arlington. His research interests include Information Security and Location Privacy for mobile devices. His other interests include operating systems and networks, mobile application development, artificial intelligence, robotics, internet technologies, aviation, music and art.