2012

Novel techniques for location-cloaked applications

Patricio A. Galdames-Sepulveda
Iowa State University

Novel techniques for location-cloaked applications

by

Patricio A. Galdames-Sepúlveda

A dissertation submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Major: Computer Science

Program of Study Committee:

Ying Cai, Major Professor

Leslie Miller

Shashi Gadia

Wensheng Zhang

Daji Qiao

Iowa State University

Ames, Iowa

2012

DEDICATION

To my mom Isidora, my dad Humberto, my wife Lilian and my son Agustín. Without their love and support I would not have been able to complete this work

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

I thank God for all the opportunities, trials and strength that have been showered on me to complete not only my thesis, but also the whole process of obtaining a Ph.D. degree. I feel that I have learned so much from this process, not only the academic experience, but also a lot about life. I am extremely fortunate that I received support, encouragement, and inspiration from many people. Without them this work would not have been possible.

My deep gratitude goes to my wife, Lilian Torres, my loving, patience, funny, talented and devoted life partner and my parents, Isidora Sepúlveda and Humberto Galdames. You are the sources of my greatest joy and my most profound sorrow, but my life would be barren without the teaching, love and experiences of each of them.

My greatest gratitude goes to my advisor Dr. Ying Cai for his guidance, enormous patience and consistent support. His knowledgeable and wise discussions have inspired the ideas of all the work in this dissertation. I have been fortunate that he has been always there supporting me to find the right direction. From him I have learned not only how to do research in Computer Science but also many aspects about life. I will be always in debt to him.

I am grateful to the other members of my committee, Dr. Shashi Gadia, Dr. Leslie Miller, Dr. Daji Qiao and Dr. Wensheng Zhang for their time and input. I would like to thank them for their insightful comments which helped me clarify my ideas and highlight my contributions in this dissertation.

I would like to express my gratitude for the generous research assistantship that I have received from the Professor of Animal Science, Philip Spike. His understanding, flexibility and generosity expressed during the time I worked on FLIP allowed me to finish successfully this thesis. Also I would like to thank both the Fulbright Commission and the Consejo Nacional de Ciencia y Tecnologia de Chile, CONICYT, for giving me the opportunity to come to the USA to pursue this Ph.D. and to fund the initial years of studies at Iowa State University.

I am greatly thankful to my lab mates, Kihwan Kim, Ashwin S. Natarajan, Toby Xu and Guolei

Yang for their collaboration during this research and contributions to this dissertation. I wish to thank my friends Sammarth Shetty, Girish Lingappa, Mercedes Silva, Jim Merideth, Al Cox, Jesse Larsson, Lorena Zuñiga, Arnoldo Padrón, Roger Toledo, Ana Maria Salazar, Bessie Fick, Hannah McCulloh, Mark Heilman, Tom Gust, professors George & Agatha Burnet, Mary Gillette and many friends of the Memorial Lutheran Church for their support and assistance during my studies.

Finally, I thank all unnamed people who have helped me in many ways to make my life happier in Ames.

ABSTRACT

Location cloaking has been shown to be cost-effective in mitigating location privacy and safety risks. This strategy, however, has significant impact on the applications that rely on location information. They may suffer efficiency loss; some may not even work with reduced location resolution. This research investigates two problems. 1) How to process location-cloaked queries. Processing such queries incurs significant more workload for both server and client. While the server needs to retrieve more query results and transmit them to the client, the client downloading these results wastes its battery power because most of them are useless. To address these problems, we propose a suite of noval techniques including query decomposition, scheduling, and personalized air indexing. These techniques are integrated into a single unified platform that is capable of handling various types of queries. 2) How a node V can verify whether or not another node P indeed locates in a cloaking region it claims. This problem is challenging due to the fact that the process of location verification may allow V to refine P's location within the region. We identify two types of attacks, transmission coverage attack and distance bounding attack. In the former, V refines a cloaking region by adjusting its transmission range to partially overlap with the region, whereas in the latter, by measuring the round trip time of its communication with P. We present two corresponding counter strategies, and built on top of them, propose a novel technique that allows P to participate in location verification while providing a certain level of guarantee that its cloaking region will not be refined during the process.

CHAPTER 1. Introduction

Many applications and protocols today rely on location information. For examples, location-based services (LBS) (e.g., [42]) need to know users' location in order to provide them the information that is specific to their location; location-aware routing protocols (e.g., [48], [9], [46]) in mobile ad hoc networks use nodes' location for path discovery and construction. Unfortunately, disclosing location information raises significant privacy and safety concerns. For example, frequent visits at places such as medical clinics or bars may be linked to a person's health condition and lifestyle. In digital battlefields, a wireless device may be located and destroyed if its location is known to an adversary.

The potential risks arising from location disclosure have motivated significant research effort. Among a number of proposed strategies, the most practical one appears to be *location cloaking*. When a node needs to disclose its location, it reports a cloaking region instead of its precise position. The cloaking region needs to cover the node's current position and satisfy other conditions, depending on the risks of concerned:

- *Anonymous service uses*: The techniques in ([31, 25, 56, 15, 19, 79], etc.) require a cloaking region to contain at least K users. This constraint is there to support anonymous uses of LBS. An adversary will not know who requests the service even if he manages to identify all these users by matching the cloaking region with restricted spaces such as houses and offices or having a direction observation over the cloaking region.

- *Location privacy protection*: The techniques in ([80, 81]) ensure that each cloaking region has been visited by at least K different users. Since these users visit the region at different times, it prevents an adversary from identifying the user who was there at the time when the cloaking region is disclosed, thus protecting the user's location privacy.

- *Location safety protection*: The techniques in ([70, 82]) provide safety protection to nodes by

ensuring that the location information they disclose cannot be used to identify any spatial region with a node density that exceeds a pre-defined threshold, say θ. A spatial region having denser nodes is more attractive for an adversary to locate the nodes and destroy them. As such, setting a larger θ results in a higher level of protection. This is opposite to privacy protection, wherein accompanying a subject with more others makes it harder to distinguish it out.

Despite their differences in purposes and cloaking requirements, all these techniques let nodes reduce their location resolution to achieve a desired level of protection. This, unfortunately, has a significant impact on the applications that rely on location information. They may suffer efficiency loss; some may not even work. Our research is to investigate this impact. Specifically, we consider two problems:

- *How to process location-cloaked queries (LCQ)*: When requesting an LBS, a user's query is now associated with a cloaking region. Since the user could be anywhere inside the cloaking region, the server would have to retrieve the query results for each position in the cloaking region. This workload is many times more when compared to handling a query that is associated with a precise location. In addition to more server workload in terms of CPU and disk I/O costs, the server needs to transmit all query results. A client downloading these results will waste its battery power because most of this data can be useless. This is especially problematic to users with a large cloaking region (e.g., requesting a high level of protection), because wireless communication is usually a dominant factor in mobile battery consumption.

- *How to verify whether or not a node is inside a cloaking region it discloses*: Location verification is a process in which a node called a verifier attests if another node called a prover is in a location it claims. Existing techniques are designed under the assumptions that the prover is willing to disclose its exact location and allows it to be localized as precisely as possible. These techniques can no longer work when a node reports a cloaking region and is willing to be verified only when there is a certain level of guarantee that its location within the cloaking region will not be refined during the process.

Our research is aimed at addressing the above problems. We summarize our contribution as follows:

- For efficient processing of LCQs, we propose a generic model that can handle various types of queries, such as ranges queries and K-nearest neighbor (KNN) queries, within a single unified platform. Our key observation is that queries may overlap in their cloaking regions and thus share some query results. In light of this, we propose to process queries as a batch instead of one by one independently. Our solution consists of three components. 1) *Query Decomposition*. We propose to decompose queries into subqueries based on their interested region. Since the subqueries with a common region need to be processed only once, the server workload is minimized. 2) *Scheduling*. The proposed technique addresses the dilemma between minimizing server latency and ensuring good fairness in query processing. 3) *Personalized Air Indexing*. This technique allows a client to filter out and download only the needed query results, thus avoiding the waste of energy in downloading irrelevant data.

- For secure verification of cloaked location, we identify two location refinement attacks and propose corresponding counter strategies. 1) *Transmission Coverage Attack* (TCA). The verifier adjusts its transmission range to partially overlap the prover's cloaking region. Preventing this attack is challenging because the verify may not disclose its transmission range and location. Even if it does, such information may not be trustworthy. We propose a solution that allows the prover to decide whether or not to participate the verification process based on the received signal strength. 2) *Distance Bounding Attack* (DBA). Here the verifier measures the round trip time of its communication with the prover to estimate their physical distance. An intuitive solution to this attack is to have the prover delay some time period before replying a challenge. The question is how much this delay should be. If the delay is too short, it leaves room for location refinement. If it is too large, it introduces unnecessary uncertainty in location verification. We propose a random delay solution that allows the prover to set a delay based on the minimum probability that it wants to prevent its location from being refined.

The rest of this thesis is organized as follows. We discuss the background and related work in Chapter 2. We present our research for efficient processing of LCQs and location verification for location-cloaked applications in Chapter 3 and Chapter 4 respectively. Finally, we conclude this thesis in Chapter 5.

CHAPTER 2. Related Work

2.1 Location-based Services (LBS)

LBS refer to services that allow users to retrieve information such as local yellow pages, local events, hotels and restaurants, and so on [89]. One type of work in this area aims at real-time delivery of location-dependent information, in which a message is sent to a user as soon as the user moves close to the location where the message is associated. Existing implementations for such services include ActiveCampus [30], GeoNotes [61], and ePost-it [13]. The other type of research is handling location-dependent queries (LDQs) such as range query (e.g., "retrieve the hotels within one mile") and K-Nearest Neighbors (KNN) search (e.g., "find me three nearest hotels"). At the core of location-based applications, efficient processing of LDQs has been investigated intensively in the context of cellular networks, where one or more stationary servers are used as information repositories. In general, an LDQ can be processed on demand or by broadcast, and client caching can be applied in each mode to improve performance [51]:

- In on-demand access, a mobile client submits a request, associated with a location, to a server. To minimize disk I/O incurred in query processing, the server usually indexes the information using spatial data structures such as R-tree [33].

- In broadcast mode, the server broadcasts the information periodically without explicit requests from clients. A client handles its own query by tuning into the broadcast channel and filtering out the data according to the query. A major challenge here is to minimize client access latency and their battery consumption. A number of techniques (e.g., [77], [89], [88], [52]) have been proposed for this purpose. These schemes extended the early concept of air indexing [39, 38] to support location-dependent queries in broadcast environment.

- Caching location-dependent data at mobile clients for future LDQs was investigated in [65, 85, 84, 78, 54, 37] and more recently in [50, 49]. While other schemes assume a client caches data just for its own use, the techniques in [50, 49] allow clients to share their caching results. That is, a client can check its nearby clients for query results before submitting its query to a server.

An LDQ is a continuous query if it stays active for some time period. A continuous LDQ is often associated with a mobile user's current position. As the user moves, the query results keep changing and require update. Two indexing techniques were proposed in [69] and [71] for on-demand processing of continuous range queries and nearest neighbor search, respectively. Continuous search of nearest neighbor in broadcast environment was investigated in [86, 87].

2.2 Location Cloaking

LBS offer a wide range of opportunities, but raise some significant privacy issues. One threat is the potential exposure of service uses. Just like regular Internet access, a user may not want to be known for using some service, especially when the service is sensitive. Another threat is location privacy. Users need to disclose their location, many times periodically, to the service provider, but the provider may not be trustworthy in keeping such data in confidential. This problem is of particular concern because a person's whereabouts may imply additional sensitive information such as health condition and lifestyle.

For self-protection, a user needs to choose a pseudonym in service uses. But simply using a pseudonym is not sufficient because the location data itself may reveal a user's real-world identity. If a location belongs to a house, then the subject is most likely the house owner. A single location sample may not be linked directly to a particular user, but the accumulation of a sequence of time-series location samples is highly likely to do so. This problem, known as restricted space identification, has motivated a series of research effort on location cloaking. The proposed techniques can be classified into two categories, according to the risks they are designed to mitigate:

Anonymous uses of LBS: This problem was first investigated in [31]. The proposed solution reduces the accuracy of location information along spatial and/or temporal dimensions. When a client requests a service, the proposed scheme computes a cloaking box that contains the client and at least $K - 1$ others, and then uses this cloaking box as the client's location to request the service. If the

resolution is too coarse for quality services, temporal cloaking is applied, i.e., delaying a user's service request. When more mobile nodes come near to the user, a smaller cloaking area can then be computed. This basic concept has since been improved by a series of work. The work in [25] considers allowing users to specify their own value of K and minimizing the size of cloaking boxes, a factor critical for the quality of LBS. The techniques proposed in [56, 44, 15, 83] address the challenges of processing location-dependent queries with reduced location resolution. Preventing an adversary from identifying a subject based on her moving pattern was considered in [10] and [40]. The proposed techniques cloak a client's position using the neighbors that have been close to the client for some time period. In all these techniques, a central server is used as a trusted middleware between mobile nodes and service providers. The server tracks the movement of mobile nodes and computes cloaking boxes upon requests.

Anonymous uses of LBS in fully distributed mobile peer-to-peer environments has also been investigated [19, 28, 29, 18, 36]. The techniques in [19, 28] let mobile nodes exchange their location and collaborate in computing cloaking boxes. The work in [29, 18] assumes that users' actual positions are publicly known. Each cloaking box generated by these schemes not only contains at least K users, but is also used by at least K of these users for service requests. They all assume mobile nodes trust each other and require nodes to disclose exact location to their neighbors. In contrast, the technique [36] allows nodes to collaborate in computing cloaking boxes without having to reveal their exact location.

Location Privacy Protection: The above techniques are designed to preserve users' anonymity in service uses, but not their location privacy. Each cloaking box contains a set of users who are currently in the area. By correlating with restricted spaces such as home and office, an adversary has the potential to identify all these users. The adversary may not know which of them requests the service, but knows they are all in the area at the time when the service is requested, thus violating their location privacy. When compared to a single user's location, revealing the presence of a group of people together in a small area is actually even more threatening – it is well said that "where you are and whom you are with are closely correlated with what you are doing" [53].

The work [80, 81] considers the problem of location privacy protection. The proposed techniques use footprints for location cloaking. A footprint is a user's location sample collected at some historical time point. By ensuring that a cloaking region contains a certain number of different footprints, this strategy can provide a certain level of guarantee that the location cannot be correlated with restricted

spaces for subject identification. A spatial region R with K different footprints means it has been visited by K different people. Knowing that there is a service request originating from R at time t, the adversary knows that one person, out of the K people, was there at time t. But as far as who was this person, the adversary does not know.

Ghinita et al. [26] study the leak of private location refinement through a linkage attack. This attack consists in pinpointing a user position within a cloaking region through the correlation of cloaking regions obtained at multiple timestamps. In this work, authors assume the attacker has a prior knowledge about the maximum user velocity and it has information about the sensitive locations. Authors propose spatial and temporal cloaking transformations to preserve user privacy.

2.3 Non-location Exposure Uses of LBS

The technique proposed in [27] lets a user download location-based information from a server without having to report location. It applies the theory of Private Information Retrieval (PIR) [17] to prevent an adversary from deriving the user's location based on the downloaded data. This approach protects a user's location privacy to its maximum extent, but has two major problems. First, a user has to download a large amount of data. For each query, a user needs to download the square root of the total number of data items stored at the server. Second, it is impossible to avoid location update completely in some applications. For example, in spatial messaging (e.g., [30, 61, 13]), the delivery of a message is triggered when the movement of mobile nodes with respect to a user-defined location satisfies some conditions (e.g., at least two nodes are within one mile). Here mobile nodes must report their location in order for the server to compute their spatial relationship.

Applying transformation-based matching for data retrieval was investigated in [83]. This scheme, called SpaceTwist, allows a user to retrieve data from the server incrementally. The process starts with an anchor, a location different from that of the user, and it proceeds until an accurate query result is reported. Although it does not require a client to disclose an accurate location, an adversary can still derive a region where the user is located based on the points-of-interest being retrieved. It also shares similar problems with the aforementioned PIR-based approach. It requires a number of iterations to processing a query and cannot be used in applications where users have to report their location.

2.4 Location-cloaked Queries

Unlike tranditional location-dependent queries, an LCQ is associated with a cloaking region instead of a precise location. The problem of LCQ processing has motivated a series of work [15, 56, 44]. In [15], a probabilistic model is proposed to process the queries with cloaked location data. It generates imprecise answers to the user. Each answer is a tuple of (S, P), where S is the retrieved information and P is the probability that this information will satisfy the corresponding query. Several metrics are defined to evaluate the quality of a service based on the imprecise answers. These metrics allow a user to decide whether or a finer cloaking region should be reported.

The work in [56] considers the problem of handling location cloaked nearest neighbor (NN) query. A grid-based algorithm is proposed to find the minimum set of candidates for an NN query. The main idea of the algorithm is to initially select a set of filter objects that can be used to prune the search over the whole set of object. With the filter objects, the algorithm can identify the spatial region which covers all potential answers to an NN query regardless of the exact location of objects in their cloaking boxes.

The work in [44] focuses on the processing of location cloaked K Nearest Neighbor (KNN) query. An algorithm called CkNN-Circ is proposed to compute the candidate list of query results. Specifically, the algorithm partitions the circumference of the circular cloaking region into disjoint arcs, and associates to each arc the data objects nearest to it. In addition, it shows that compared with query processing on rectangular cloaking boxes, CkNN-Circ has a higher overhead but it can reduce the number of candidates, which means that using circular cloaking box is preferable in the situation when communication cost is more important than processing cost.

Most existing techniques process queries one by one independently. One exception is [58]. In the context of road networks, the authors propose a new location obfuscation technique and a framework for efficient processing of anonymous queries. In this scheme, the server processes the queries periodically, where the queries arriving in the previous cycle are processed at the same time. This approach does not support real-time query processing because earlier queries are made to wait for later ones, even when the server is idling. Moreover, this paper does not consider the problems of fairness in query processing and transmission of query results. It assumes that all clients submit their queries through a central anonymization server. This anonymizer is also responsible for filtering out the necessary query results

for clients. In reality, such a server may not exist and a client may have to submit its query directly. In this case, the client downloading all query results would waste a significant amount of resources because most of these results are usually useless due to location cloaking.

The problem of privacy leak during the processing of LCQs over a mobile peer-to-peer networks was analyzed by Hashem et al. [35] only for GNN queries. A group nearest neighbor (GNN) query returns the location of a meeting place that minimizes the aggregate distance from a spread out group of users. In this approach, users provide their locations as cloaking regions instead of exact locations and they perform a collaborative query processing. Authors show that a distance intersection attack can refine a group member's location. To avoid this problem, the authors develop a private filter that determines the actual group nearest neighbor from the retrieved candidate answers without revealing user locations to any involved party.

2.5 Scheduling Techniques

The problem of query scheduling to balance system throughput and fairness has been well studied in literature. In [21], Dan et al considered how to select a video from a batching queue to multicast and proposed a number of scheduling techniques, including first-in-first-out (FIFO), maximum-queue-length (MQL), and maximum-factored-queue-length (MFQL). Among them, MFQL schedules the video for multicast which has the largest product between the number of new requests for this video and the time elapse since the last time it was multicast. This scheme was showed to have the advantages of both FIFO and MQL: having FIFO's performance in fairness and MQL's performance in system throughput.

Scheduling techniques for wireless data broadcast can be classified into three categories: push-based, pull-based, and hybrid. In the "push-based" category, the server periodically broadcasts a schedule which is computed based on the user access history [23, 20, 67]. For instance Franklin et al. [23] assume queries requests a single data item. On the contrary other works assume queries requests multiple data items [20, 67]. Chung et al. [20] propose an scheduling approach that assumes the frequency of the queries are known and schedules the data items based on a measure named Query Distance. This measure represents the degree of coherence for the data set accessed by a query. Shih et al. [67] propose a technique to schedule dependent data items that allows to clients wait for their requests in a equal time

interval. They measure the fairness of the broadcast using the variance of the latency. The data items are represented by a weighted directed acyclic graph, where a weight represents the probability the data item is accessed.

All approaches in this category assume access probabilities do not change often and known. Also they do not take into account the current data access pattern. In the second category named as "pull-based" or commonly referred to on-demand broadcast system, users explicitly request for data items from the server, which subsequently processes, schedules and broadcasts only the requested data items [22, 75, 4, 14]. FCFS (First Come First Serve), MRF (Most Requested First) and LWF (Longest Wait First) were studied in [22, 75] In FCFS, pages are broadcast in the order they were requested. In MRF, the page with the maximum pending request is broadcast first. MRFL is similar to MRF, but ties are solved in favor of the page with the lowest request access. In LWF, for each page is computed the sum of the waiting time of all pending requests for this page. Then the page with largest sum of waiting time is broadcast first. A well-known scheduling technique is $R \times W$ proposed by Aksoy et al in [4]. Here "R" means the number of outstanding requests for a data item and "W" is the time of the oldest outstanding request for that page. The basic idea is to select the data item with highest product of R and W to broadcast first. While this scheme assumes each query is for a single data item, Chen et al. [14] present an approach for scheduling of multi-item requests with network coding. The authors assume each request is associated with a deadline.

In environments where data items have different sizes, Acharya et al. [3] propose the concept of Stretch, which is defined as the ratio between the response time of its request to the service time. Based on this metric the authors propose an scheduling algorithm called longest total stretch first (LTSF). This metrics is considered to be more fair with those queries requesting less data items than with those requesting a large number. Wu et al. [76] show this algorithm is not suitable for large databases, and they proposed an improved version.

All previous technique assume the scheduling decision is made when there is available bandwidth to transmit a data item., i.e, at each broadcast tick. On the contrary, The work by Prabhu et al. [62] assumes queries request single data items and propose an approach that makes scheduling decisions at periodic interval for on demand broadcast systems. In this way indexing can be interleaved with the data items and for this purpose the authors adapt the (1,m) indexing proposed by Imielinski et al. [39].

In the third category, consists of hybrid approaches between periodic and on-demand data broadcasting [2, 32, 47]. Server partitions data items into hot and cold items. Hot items are broadcast periodically based on their access probabilities. Cold items are scheduled based on the waiting time and the number of outstanding requests. Acharya et al. [2] assume queries requests single data items and analyzes the access preferences of each item. Data items are scheduled based on their popularity. Guo et al. [32] show that it is possible to obtain significant performance improvement if an optimal cut-off point between push-based and pull-based technique is found. Kim et al. [47] found an efficient way to obtain this optimal cut-off point between hot and cold items.

Although our technique falls into the push-based approach, all aforementioned scheduling techniques assume the server knows exactly what clients request. In other words, a client needs all the data retrieved in response to its query. *In contrast, when dealing with an LCQ, the server knows only a geographic region where a client is located, but not its exact location*. To handle such ambiguity, we have to consider the probability that a client is located inside a specific geographic region (i.e., a subquery's cloaking region).

2.6 Air Indexing

Air indexing refers to those indexes broadcasted in wireless environments to address scalability issues and power saving on mobile devices [38]. Without indexing, if a user wants to retrieve data, it has to continuously monitor the broadcast channel until the desired data arrives. This operation will waste a lot of energy since user must remain active while listening the channel. To overcome this issue, indexing information of the arrival times of the data is provided earlier on the broadcast channel. With this extra information, user can predict when the needed data will be available on the air and they can remain in sleep mode during the waiting time and become active only when the needed data is close to become available.

Air indexing was initially proposed by Imilienski et al [39]. These authors proposed two indexing methods named as flexible indexing and hash-based indexing. The first one requires data items to be sorted first in ascending or descending order of the search key values and then the data items are divided into p-segments. The first bucket in each data segment contains a control index, which is a binary index

mapping a given key value to the segment containing that key and the local index, which is a m-entry index mapping a given key to buckets within the current segment . Authors claim that by tuning *p* and *m*, mobile users can achieve good tuning time or good access latency. The second method is a hash-based indexing in which each data item is mapped to a bucket defined by the broadcast schedule. Since multiple items can be hashed to the same bucket, these authors decide to solve collisions by pushing overflow items into the succeeding slots and pushing forward the items originally hashed to these buckets.

The same authors [38] introduced later the $(1, m)$ indexing as an index allocation method, according to which the index information is broadcasted m times during each bcast. The main drawback of this technique is the replication of the entire index structure m times, which prolongs the broadcast cycle and consequently the average access time. The same paper, proposes a tree-based indexing technique, called distributed indexing. In this technique, the data file is associated with a $B+$ tree which is linearized with a pre-order traversal to be broadcast in a wireless channel. Additionally the first k levels of the index structure are partially replicated in the broadcast, while the remaining levels are not. The distributed indexing has lower access time compared to $(1, m)$ indexing while both tuning time are similar.

The authors in [77] propose the exponential index. This scheme can be loosely seen as a linearizarion of a directed acyclic graph. Each bucket contains a vertex of this graph and a data item, then a client can initiate a binary search operation over the broadcast data from any broadcasted bucket. This indexing shows logarithmic access complexity and it is resilient to errors in the reception of the broadcasted data.

Until now, air indexing techniques are used mainly for periodic broadcasting of data items to a set of users. By downloading the indexing portion, a client knows when the data items of its interest will arrive and therefore can put itself in a sleeping mode until the time of data arrival. While early research focused on indexing one-dimensional data, recent work [78, 88, 41, 57, 52, 55] has extended the concept to broadcast spatial data for queries such as range queries and KNN queries.

2.7 Location Verification

Location verification is a process where a node called verifier verifies whether or not another node called prover is indeed on a location which it claims. This problem was first investigated by Brands and Chaum [11]. The proposed technique allows the verifier to find the upper-bound of its distance to the prover. In this scheme, the verifier sends a series of challenges to the prover. Upon receiving a challenge, the prover sends a response immediately. Here the prover's processing time is ignored. Sastry et. el [66] argue that this processing time is not negligible with respect to the propagation time when RF signal are used. To address this problem, they propose that the verifier uses RF signal to transmit challenges and the prover replies these challenges using ultrasound signal.

Hancke et al. [34] proposed a distance bounding protocol based on ultra-wideband pulse communication that can be implemented on computationally weak RFID devices. Capkun et al. [73] proposed location verification protocols based on simple ranging to avoid the need of fast processing at the prover. Nodes require of an RF and an ultrasonic interfaces. To prevent the possibility of cheating, the verifier is assumed to be a mobile covert station. Later, Rassmussen et. al. [64] show that it is possible to implement the distance bounding protocol using only RF network interfaces. The device they developed is able to receive, process and transmit RF signals in less than 1ns.

To narrow down the prover's location, many techniques based on multiple verifiers working collaborativelly have been proposed [68] [72] [74] [16]. Singelee et al. [68] claimed that protocols based on ultrasonic are vulnerable to RF wormhole attacks and advocated using only RF signals. These authors also showed guidelines on how to modify a distance bound protocol to make it resistant to the terrorist fraud attack. Capkun et al. [72] proposed a multilateration scheme that specifies a δ test and *point-in-triangle* test to check if the provers claimed location is true. In the context of sensor networks, Vora et al.[74] proposed a collaborative approach, where some sensors within the prover's location area are the only ones that can listen to a signal from the prover. Others located outside of the claimed area should not listen a signal. Chiang et al. [16] study the impact of collusion attacks where adversaries share their private keys and also study the jamming attack where attackers inject a high amount of noise to prevent successful challenge and response receptions.

Liu et al.[55] argued existing protocols either require special network interfaces or need the col-

laboration of nearby peers. They proposed a node-to-node verification protocol for sparse networks which is supported by a satellite network. All previous approaches assume a free space scenario, but Abumansoor et al [1] proposed a location verification protocol for Non-Line Of Sight in VANET.

All these techniques assume that the prover is willing to release its exact location and to be localized as precisely as possible.

The problems of privacy leaks caused by location verification was investigated by Rasmussen et. al in [63]. They showed that an external listener can discover the information such as the upper-bound distance between the prover and the verifier by measuring the arrival times of their messages. In their proposed technique, prover establishes with the verifier an stream of random bits and the verifier establishes another similar one with the prover. When verifier wants to sends a challenge, it inserts the challenge into the stream going to the prover. In order that the prover can identify a valid challenge, the verifier prefixes a hidden mark to the challenge. This hidden mark is only known by the verifier and the prover. When the prover identifies a challenge, it inmediately inserts a hidden prefix into the stream going to the verifier and it begins inserting the bits for the response. With this approach authors claim an external listener cannot determine the arrival times of a challenge and a response, since all bits look completely random. This proposed technique prevents the leak of location information to an external attacker but it assumes the prover and the verifier trust each other and the prover is willing to share its exact location to the verifier.

CHAPTER 3. Efficient Processing of Location-cloaked Queries

Traditionally, a client submitting a location-dependent query (e.g., finding my nearest gas station/hotel) discloses an exact position. Because of privacy concerns, the client now discloses a cloaking region. Since the user could be on any position in region, the server needs to retrieve all possible query results in order to guarantee the information required by the user is included. This will increase server workload dramatically in terms CPU and disk I/O costs. When the server returns these query results, the client downloading these result will waste its resources because most of them are useless.

This part of research addresses the above problems. We propose a generic model that can handle various types of queries, such as ranges queries and K-nearest neighbor (KNN) queries, within a single unified platform. The proposed system supports real-time query processing in the sense that queries are processed immediately as long as the server resource is available. When the server is underloaded, a query is processed upon its arrival without any latency. But when the server is overloaded, the incoming queries have to be queued. Our research focuses on how to process the queries pending in the queue when the server resource becomes available. Our key observation is that these queries may overlap in their cloaking regions and thus share some query results. In light of this, we propose to process the pending queries as a batch instead of one by one independently. Our main technical contributions are as follows:

- *Query Decomposition*. We propose to decompose queries into subqueries based on their interested region. Since the subqueries with a common region need to be processed only once, the server workload is minimized.

- *Scheduling*. Given a queue of pending queries, the order of their processing has significant impact on the overall system performance. We propose a novel scheduling technique that addresses the dilemma between minimizing server latency and ensuring good fairness in query processing.

- *Personalized Air Indexing*. When the server returns the query result to a user, we apply air indexing to allow the user to filter out and download only the needed portion, thus saving their energy consumption. Until now, air indexing is typically used in applications wherein data are broadcast to a large group of users.

The rest of this chapter is organized as follows. We first give a system over in Chapter 3.1 and then present the proposed techniques in detail in Chapter 3.2. In Chapter 3.3, we examine the performance of these techniques.

3.1 System Overview

Without loss of generality, we assume that a single server is used to manage all point-of-interests (POIs). Each POI is represented by a point location and a data structure that describes the interest related to the location. The POIs are indexed based on their location. A simple indexing approach is to partition the network domain into a set of subdomains. If the number of POIs located inside a subdomain exceeds some threshold, the subdomain is recursively partitioned. A subdomain that is no longer split is called a *cell*. This index structure is illustrated in Figure 3.1. The server is assumed to be able to keep in the main memory the indexing nodes, which store the domain partitioning hierarchy. On the other hand, all data nodes, each keeping a list of POIs, are stored on disk and retrieved when needed. In this paper, we will estimate the cost of processing an LCQ using the number of data nodes that the server needs to retrieve.

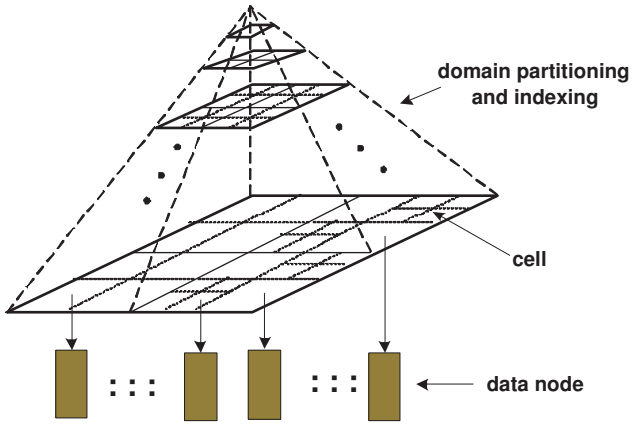


Figure 3.1 POI indexing

We say a cell is *relevant* to an LCQ if the POIs in the cell must be retrieved in order to answer the query. A range query retrieves the POIs within a user-defined window, so its relevant cells are simply those which overlap with the query's cloaking region. But for other types of query, determining the relevant cells may be nontrivial. For example, when dealing with a location cloaked KNN query, innovative techniques will be needed in order to identify the relevant cells. Some efficient techniques (e.g., [56, 6, 58, 8]) have been developed for this purpose, so we will assume that given a query, the server can find its relevant cells, without concerning ourselves how this is actually done. To ease our discussion, when loading the POIs within a cell into memory memory, we will simply say retrieving the cell.

The server maintains a query queue $\mathbb{Q}$. All incoming queries are first placed in the queue and are processed as soon as the server resources are available. Under this model, when the server is underloaded, a query is processed upon its arrival without any latency. The problem arises when the server is overloaded, in which case the queries will be queued. Our research focuses on how to process the queries pending in the queue when the server resources become available. Let Q_1, Q_2, ..., and Q_n be the set of queries pending in $\mathbb{Q}$, we have the following definitions:

- We define *server workload* W to be the total number of data nodes retrieved by the server to answer these queries.

- Given a query Q_i ($1 \leq i \leq n$), $|Q_i|$ denotes the *query size*, which is defined to be the total number of cells relevant to Q_i. We now define Q_i's *client waiting* and *server latency*, denoted as $W(Q_i)$ and $L(Q_i)$, respectively. The former is defined to be the time period starting from the time when Q_i was put in the queue to the time when the cells needed by the requesting client are retrieved. The latter, on the other hand, is the duration from the time when Q_i was put in the queue to the time when the last cell relevant to Q_i is retrieved. Note that due to location cloaking, the server may have already retrieved all information needed by a client before it finishes retrieving all relevant cells.

- Let $\bar{L}$ denote the *average server latency* of these queries, where $\bar{L} = \frac{\sum_{i=1}^{n} L(Q_i)}{n}$. We define the *unfairness* F of processing these queries based on the average variance in their processing

latency, i.e.,

$$F = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{L(Q_i)}{|Q_i|} - \overline{L}\right)^2.$$

Note that this definition of unfairness takes into consideration the size of queries, and a lower value of F indicates a higher fairness.

- Let d be the total amount of data which a client downloads from the server and d' be the amount of data that is actually needed for its interest. We define the client's *download usefulness* U as $\frac{d'}{d}$. A higher U means a less waste of client battery power in receiving irrelevant data (due to location cloaking).

Our system design has the following goals:

- For the server, we want to minimize server workload, server latency, and unfairness.
- For each client, we want to maximize its download usefulness.

3.2 Proposed Techniques

Let Q_1, Q_2, ..., and Q_n be the queries pending in the queue. As mentioned earlier, these queries may have their cloaking regions overlapped and thus share some common relevant cells. In light of this, we propose to process them as a batch, instead of one by one independently. Our techniques consist of three elements.

3.2.1 Query Decomposition

Let Q_i ($1 \leq i \leq n$) be a query in the queue. We decompose it into a set of subqueries as follows. The server follows the index tree to determine the cells that overlap with Q_i's cloaking region (denoted as $Q_i.R$ hereafter). Suppose $Q_i.R$ overlaps with k cells, c_1, c_2, ..., and c_k. For each cell c_j ($1 \leq j \leq k$), the server creates a subquery q_j with a cloaking region $c_j \cap Q_i.R$ (the region common to c_j and $Q_i.R$). This decomposition is illustrated in Figure 3.2.

For each subquery, the server determines its relevant cells. We will denote the set of cells relevant to a query q as $C(q)$. Recall that a cell is relevant to a query if the data node linked by the cell must be retrieved in order to answer the query. Given a query Q_i with a set of subqueries $\{q_1, ..., q_k\}$,

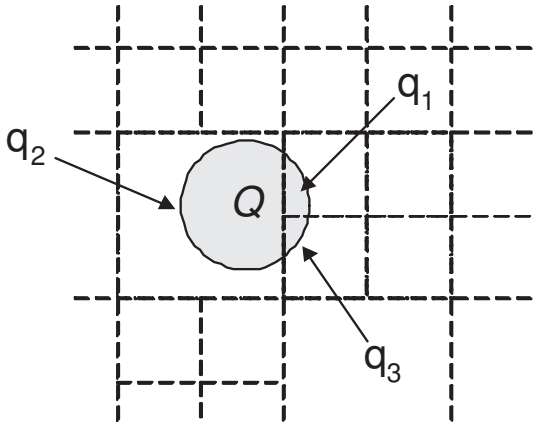


Figure 3.2 Query Q is decomposed into q_1, q_2, and q_3

the set of its relevant cells is equal to the union of the cells relevant to its subqueries, i.e., $C(Q_i) = C(q_1) \cup C(q_2) \cup ... \cup C(q_k)$.

For each query in the queue, the server follows the above procedure to decompose it into a set of subqueries and then determine their relevant cells. Without loss of generality, let the union of all these relevant cells be $\mathbb{C} = \{c_1, c_2, ..., c_m\}$. Clearly, the cost of processing the queries in the queue is to retrieve these cells, i.e., retrieve from the disk the data nodes linked by these cells. When a cell is relevant to more than one query, the server needs to retrieve the cell only once, thus minimizing its workload.

3.2.2 Scheduling

We now consider the order of retrieving relevant cells. We refer to a particular order of retrieval as a *schedule*. For example, c_1, c_2, ..., c_m is a schedule if these cells are loaded sequentially in that order. Different schedules will result in vastly different latency and fairness in query processing. Indeed, minimizing latency and maximizing fairness are two seemly conflicting goals.

For example, we can arrange a schedule using *first-in-first-out* (FIFO). In this scheme, the queries are sorted according to their arrival time and the cells relevant to an earlier query are retrieved at an earlier time. While this strategy maximize the fairness, it can lead to poor server latency in query processing.

Alternatively, we can schedule the retrieval of cells just to minimize the average server latency,

which we will refer to as *maximum completeness first* (MCF). Before presenting this scheme, we have the following definitions. Let S be a set of cells. We define the completeness of S towards answering a subquery q_j to be $M(S, q_j) = \frac{|C(q_j) \cap S|}{|C(q_j)|}$. This metric measures the percentage of a subquery's relevant cells being contained by S. If no cell inside S is relevant to q_j, then $M(S, q_j) = 0$. On the other hand, if S has all cells relevant to q_j, we have $M(S, q_j) = 1$.

Given an LCQ Q_i with a set of subqueries $\{q_1, ..., q_k\}$, we define the completeness of S towards answering Q_i as $M(S, Q_i) = \sum_{j=1}^{k} M(S, q_j) \cdot \frac{A(q_j.R)}{A(Q.R)}$. Note that this formula takes into consideration where the client is actually located. If S contains all cells relevant to q_j, then Q_i would be answered if the client is in q_j's cloaking region. Since the server does not know the client's exact position, we measure the probability that it is located in a subquery's cloaking region as the proportion of its area with respect to the area of the client's entire cloaking region.

Finally, we define the completeness of S towards answering all queries pending in $\mathbb{Q}$ as $M(S, \mathbb{Q}) = \frac{\sum_{i=1}^{n} M(S,Q_i)}{n}$.

Given the above definitions of the completeness of S towards answering a subquery, a query, and all queries in queue, we are now ready to introduce MCF. Let S be an empty set initially and the entire set of relevant cells be $\mathbb{C} = \{c_1, c_2, ..., c_m\}$. For each cell c in $\mathbb{C}$, we compute $M(S \cup \{c\}, \mathbb{Q})$. The cell with the largest value of $M(S \cup \{c\}, \mathbb{Q})$ is then inserted to S and removed from $\mathbb{C}$. This process is repeated until all cells are moved into S. The order of inserting these cells into S is the schedule.

The schedule generated by the above MCF approach will have the smallest average server latency. However, it tends to be unfair to the queries that arrive earlier but do not have much overlap with other queries. To alleviate this problem, we revise MCF to have the following approach. Let S be an empty set initially. For each cell c in $\mathbb{C}$, we compute $M(S \cup \{c\}, \mathbb{Q}) \times T(c)$, where $T(c)$ is the waiting time of the earliest query in $\mathbb{Q}$ to which c is relevant. The cell with the largest value of $M(S \cup \{c\}, \mathbb{Q}) \times T(c)$ is then inserted to S and removed from $\mathbb{C}$. This process is repeated until all cells are moved into S. The order of inserting these cells into S is the schedule. Note that this scheduling algorithm takes into consideration the time that a query has been waiting in the queue and is therefore more fair than MCF.

3.2.3 Personalized Air Indexing

We now consider how to send the query results to a client. Because of location cloaking, most query results would be useless. Although the server needs to transmit all of them to the client, the client should be allowed to download only the data it truly needs, because it knows its own location. Essentially there are two approaches for a client to submit an LCQ. One is through a central anonymization server, which functions as a trustworthy middleware to request an LBS on behalf of the client. When the anonymization server receives the query results, it can filter out only the needed portion and send it to the client [56, 45]. In the other approach, a client cloaks its own location (e.g., collaborating with its neighboring users)[19, 7] and submits its query directly. In this case, the client will receive query results directly from the server. To avoid downloading irrelevant data, we propose the following air indexing technique.

Let Q_i be a query and suppose this query is decomposed into subqueries $\{q_1, q_2, ..., q_m\}$ and its relevant cells are $\{c_1, c_2, ..., c_k\}$. Once the schedule is determined, the server will know when the retrieval of a cell will be finished. As such, given a subquery q_j and its relevant cells, the server can compute when these cells will be retrieved and the time when the POIs in a cell will be sent to the client. So we index the query results with the data structures illustrated in Figure 3.3. It has three segments. The first segment is simply a 4-byte field that records the number of subqueries. The second segment records the subqueries and the time when the POIs in their relevant cells will be delivered. A subquery q_j is identified by $q_j.R$ and the number of its relevant cells and then followed by an array of the delivery times of its relevant cells. The third segment stores the POIs in the cells, which are ordered according to the schedule of their retrieval.

To download the query results, a client listens to its channel for the first 4 bytes (the first segment). Based on the number, it downloads the index of subqueries (the second segment). According its own position, the client knows which subqueries and their relevant cells are needed, so it just needs to be active in data receiving only when the delivery of the POIs in these relevant cells starts.

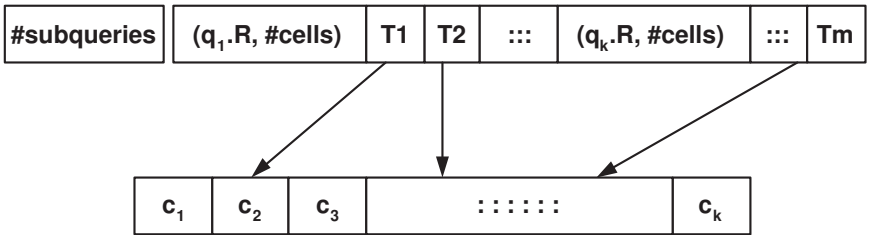


Figure 3.3 Personalized air indexing

3.3 Performance Evaluation

We evaluate the performance of our proposed techniques using simulation. Four performance metrics are used, including server workload, server latency, unfairness, and download usefulness. The definitions of these metrics are given in Section 3.1. The metric of server workload measures the effectiveness of our query decomposition in reducing server disk I/Os (i.e., preventing the server from same cells more than once during one batch of processing). The metrics of server latency and unfairness together measure the performance of scheduling techniques. Finally, the metric of download usefulness measures the percentage of the amount of data downloaded by a client that is truly useful. A higher usefulness means less waste of client battery power.

For performance comparison, we implemented two processing techniques, Independent and Batch. The former processes queries one by one independently, whereas the latter processes queries in batch through query decomposition. For Batch, we implemented three scheduling techniques, namely, *FIFO* (first-in-first-out), *MCF* (maximum completeness first), and *CW* (completeness $\times$ waiting time). CW is what we propose. The parameters used in our simulation are summarized in Table 3.1. Largely we simulate a location-based information system for a medium-sized city. The server is assumed to be capable of loading 100 cells per time unit from disk. In each round of simulation, we generate a number of POIs for a network domain of $100,000 \times 100,000$. These POIs are indexed through domain decomposition discussed in Section 3.1. Each cell is allowed to contain no more than 100 POIs. We then generate a number of KNN queries. The value of K ranges from 10 to 100. We are mainly interested in how the system performances are impacted by three parameters: query distribution skewness, query arrival rate, and cloaking radius.

Table 3.1 Parameters - Processing LCQs

Parameter	Default	Variation
Network domain	$100,000 \times 10,000$	N/A
Number of Cells	10,000	N/A
Number of POIs per Cell	100	N/A
Server processing capability	100 [cells/time unit]	N/A
K value in KNN	50	10 - 100
Query Arrival Rate	50 [queries/time unit]	10 - 100 [queries/time unit]
Cloaking Radius	500	100 - 1000
Skewness	5	1 (uniform) - 10

3.3.1 Impact of Query Distribution Skewness

In this study, we set the query rate at 50 queries/time unit, the K value in KNN to be 50, and the average cloaking radius to be 500. We varied the skewness of query distribution from 1 to 10. When the skewness is 1, the queries are uniformly distributed within the network domain. When the skewness increases, the queries are distributed more densely toward the upper-left corner of the network domain. The results are plotted in Figure 3.4.

Given a set of queries, the number of their relevant cells changes only slightly when only the location of these queries changes. As such, the workload under Independent is nearly flat. However, when query distribution skewness increases, there will be more overlapping among queries. Through query decomposition and batch processing, this would reduce the server workload and the average client waiting time. These expectations are confirmed in Figure 3.4(a) and (b), respectively. The study shows that the reduction on the server workload is significant, more than 100% when the skewness is more than 8. It also shows that under all scheduling schemes, the average client waiting time decreases, but FIFO always has the longest waiting time. On the other hand, MCF has the smallest client waiting, but is the most unfair among the three techniques, as shown in Figure 3.4(c). As for CW, the average client waiting time is almost the same as that achieved by MCF. However, CW is considerably more fair than MCF under all simulation settings.

Figure 3.4(d) shows that query distribution skewness has little impact on client download usefulness. However, without using air indexing, less than 60% of the total amount of data transmitted from the

server to a client as query results is useful. In contrast, with indexing, the usefulness is nearly 100%, where the only overhead is the indexing portion.

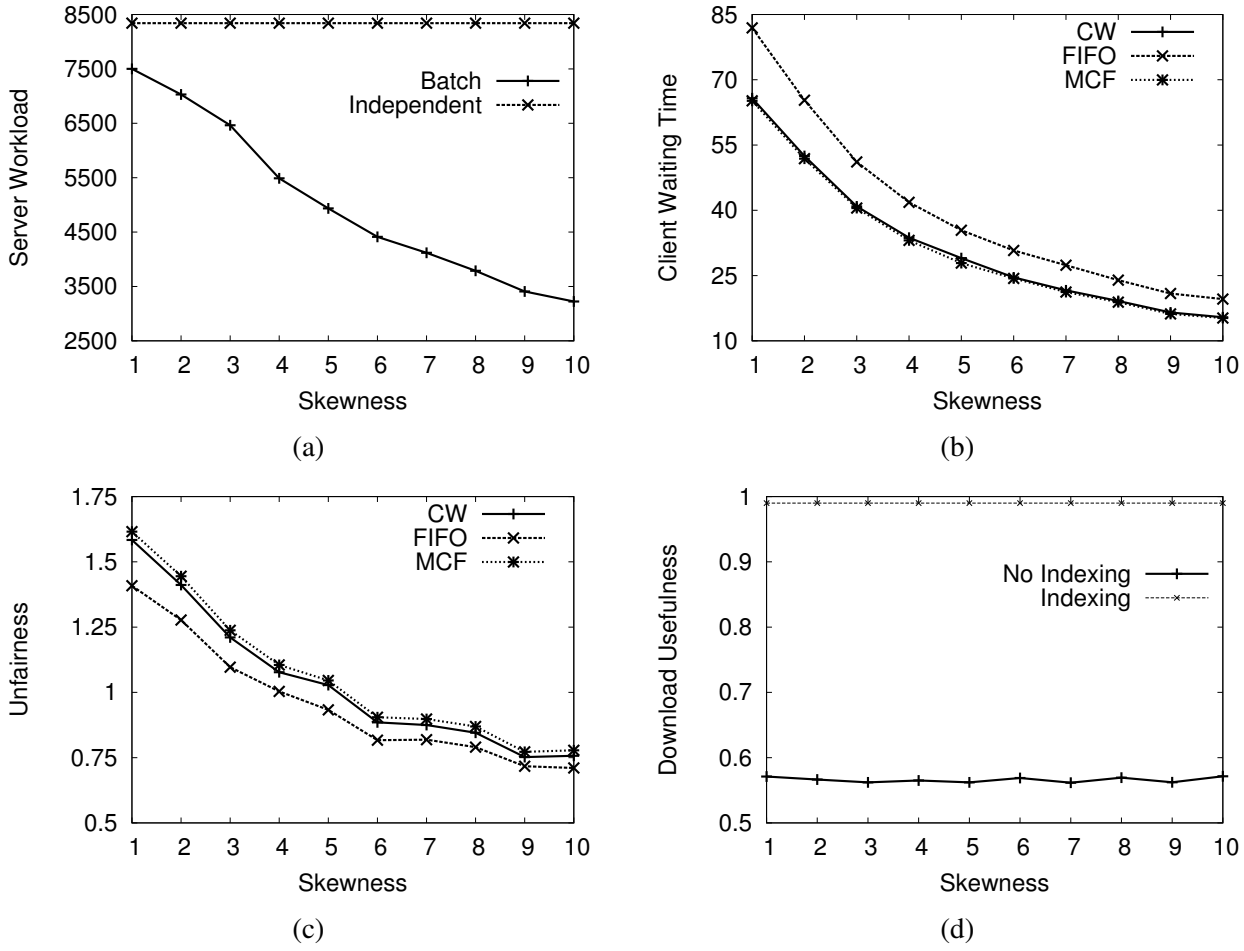


Figure 3.4 Impact of query distribution skewness

3.3.2 Effect of Query Arrival Rate

In this study, We fixed query distribution skewness at 5 and cloaking radius at 500. We varied the query arrival rate from 10 to 100 queries per time unit. The results are plotted in Figure 3.5.

Figure 3.5(a) shows that as query arrival rate increases, the server workload increases under both Independent and Batch. However, the increase from Independent is much sharper than Batch. This study shows that the effectiveness of Batch in handling LCQs when the server is overloaded. For

Independent, each query is processed independently, but for Batch, when a new query shares its query results with other earlier queries, its processing does not introduce additional server workload.

Because the server workload increases, the average client waiting time increases, as illustrated by Figure 3.5(b). Again, FIFO has the worst performance while CW and MCF have similar results. Nevertheless, Figure 3.5(c) shows that CW consistently outperforms MCF in terms of unfairness. When the query arrival rate increases from 10 to 40, the unfairness under all techniques increases, but as the query arrival rate continues to increase, the unfairness starts to reduce and eventually stabilizes. We examined the raw data and found that after the rate exceeds 40 queries per time unit, most queries arriving lately overlap with the queries that are pending in the queue. Thus, the cells relevant to the earlier queries are also relevant to the late queries. This phenomenon reduces the average variance of query processing time and thus reduces unfairness.

For each client, its download usefulness is determined by the total amount of data it receives and the amount of the data it actually needs. This is not affected by the query arrival rate. Figure 3.5(a) confirms that this parameter does not have much impact on the average usefulness. The slight variance of usefulness, in a range of 0.03, was due to the randomness in generating query location.

3.3.3 Effect of Cloaking Radius

This study investigated the impact of cloaking radius. The query arrival rate was fixed at 50 queries per time unit and the query distribution skewness at 5. We varied the cloaking radius from 100 to 1000. The results are plotted in Figure 3.6.

Figure 3.6(a) shows that the server workload increases as the cloaking radius increases. This is not surprising because the server needs to retrieve more relevant cells for the enlarged cloaking regions. However, Batch outperforms Independent significantly; in particular, its workload increase is much slower. This is due to the fact that larger cloaking regions result in more overlapping. Figure 3.6(b) shows that the average client waiting time increases under all schemes, but FIFO is the one that performs the worst while CW and MCF have similar performance. As for the metric unfairness, it increases when the cloaking radius increases from 100 to 500, but then starts to decrease and then stabilizes. This phenomenon is due to the heavy overload, which we explained in the previous subsection.

Figure 3.6(c) shows that usefulness decreases significantly when no indexing is used. When a client

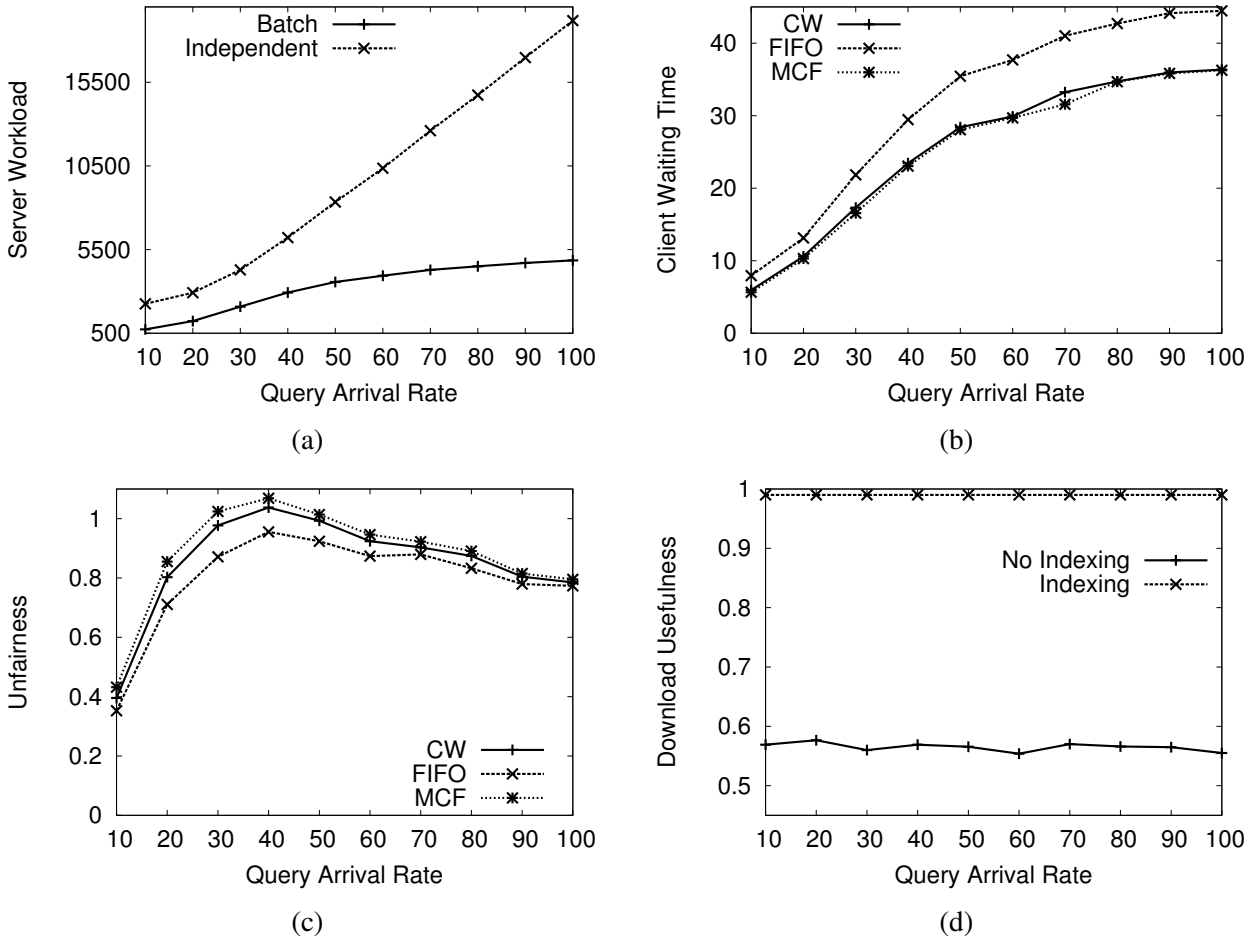


Figure 3.5 Impact of query arrival rate

uses a larger cloaking region (for a higher level of privacy or safety protection), the server needs to retrieve and transmit more data. Clearly, without air indexing, the client will have to download more irrelevant data and thus wastes more energy consumption.

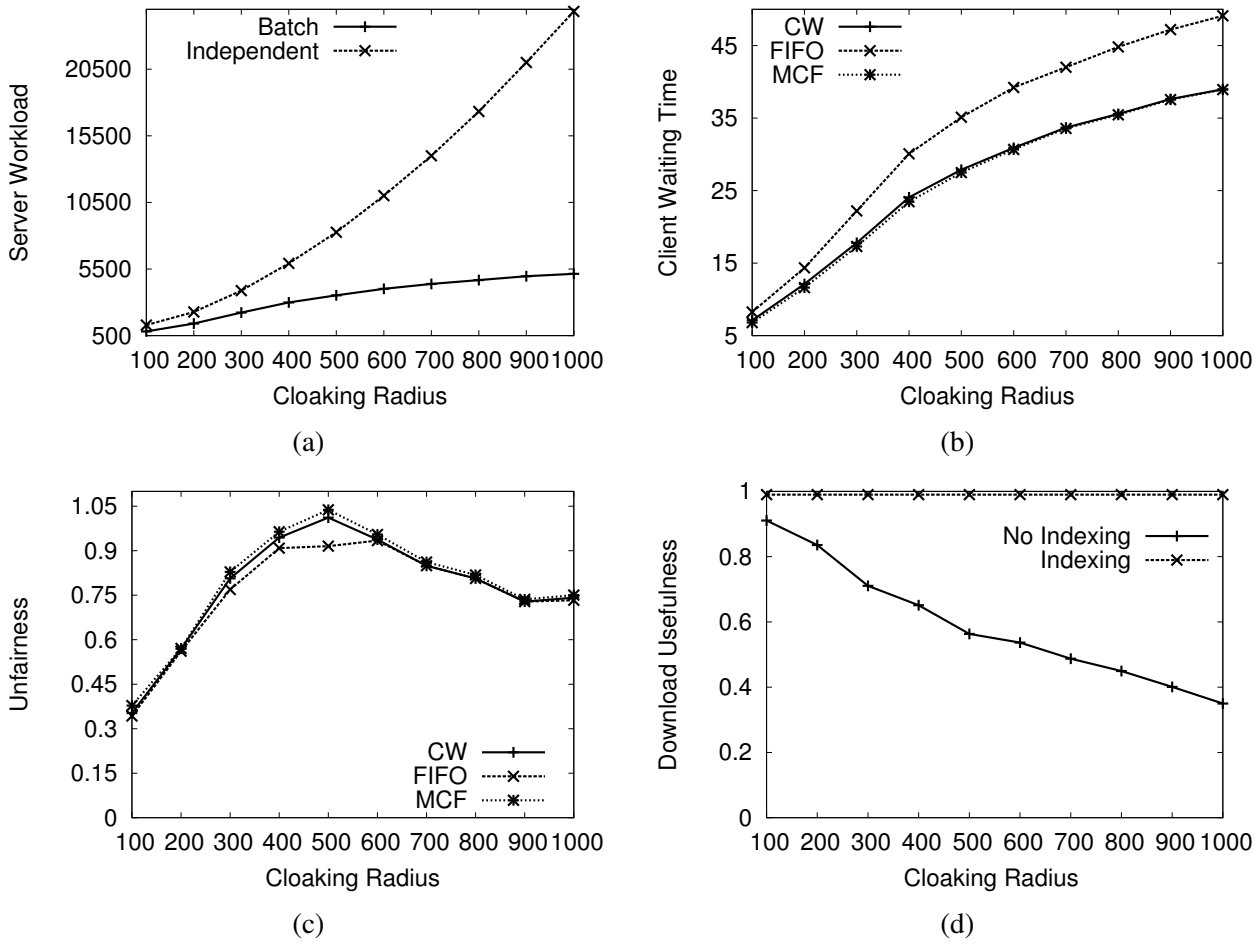


Figure 3.6 Impact of cloaking radius

CHAPTER 4. Secure Location Verification of Cloaked Locations

The problem of location verification is how an entity called a verifier can verify whether or not another entity called a prover is indeed in the location it claims. There are important uses of location verification. In ad hoc networking, location-based routing protocols (e.g, LAR [48], DREAM [9], GPSR [46]) are showed to outperform early topology-based counterparts (e.g., FSR [59], AODV [60], DSR [43]) to a large extent, yet this is true only when participating nodes are honest in disclosing their location. As many other applications (e.g., location-based services and access controls) today are also designed to take advantage of location information, it is crucial to have robust and efficient location verification in place.

The problem of location verification was initially investigated by Brands and Chaum [11]. In the proposed *distance bounding protocol*, the verifier and the prover exchange a series of challenges and responses; based on the RF propagation delay, the verifier computes a tight circular region that contains the prover's position. This work has since inspired a number of more advanced techniques (e.g., [66, 68, 74, 73, 16]). Existing techniques, however, are designed under the assumptions the prover is willing to 1) disclose its exact location and 2) allow the verifier to localize its position as precisely as possible. These assumptions make them infeasible for location-cloaked applications.

In this part of research, we consider the problem of secure location verification in the context of location cloaking. Specifically, we investigate how to verify if a prover is inside a cloaking region while providing a certain level of guarantee that the cloaking region will not be refined during the verification process. A cloaking region is refined if an adversary (e.g., the verifier) can conclude some part of the region where the prover cannot locate. To our knowledge, secure valification of cloaked location claim has not been studied in literature. We summarize our main contributions as follows:

- We present two types of location refinement attacks, namely *Transmission Coverage Attack*

(TCA) and *Distance Bounding Attack* (DBA). In the former, the verifier refines a cloaking region by adjusting its transmission range to partially overlap with the region, whereas in the latter, by measuring the round trip time of its communication with the prover.

- For each of these attacks, we propose a corresponding solution. For TCA, our technique allows the prover to decide whether or not to respond to the verifier's challenge based on the signal strength it receives. For DBA, we propose the prover to delay its response and analyze how much this delay should be based on the minimum probability that it wants to prevent the attack.

- Built on top of these solutions, we present a secure location verification technique that support location cloaking.

The remainder of this chapter is organized as follows. We discuss the two types of attacks in details and present their solutions in Chapter 4.1. The proposed location verification technique is presented in Chapter 4.2 and its security is analyzed in Chapter 4.3. Finally, we evaluate the performance of the proposed technique in Chapter 4.4.

4.1 Location Refinement Attacks

Consider two nodes, prover P and verifier V. P claims it is inside a cloaking region R. Without loss of generality, we assume R is a circular region centered on position O with a radius r. V wants to verify whether or not P is indeed in R. We say *P's location is refined* if during the verification process, V can conclude that there exists a sub-region within R where P cannot be there.

Both P and V are assumed to send and receive signals through an RF interface with an omnidirectional antenna. In other words, neither P nor V is able to determine the direction of a signal. Moreover, we follow the assumption in existing work (e.g., [11, 68, 74, 73, 16]) that a node's processing time is negligible with respect to the signal propagation time. This assumption is considered practical because light travels about 30 cm per nanosecond while today's electronics can easily handle timings of a few nanoseconds [11]. Rassmussen et. al., for example, has recently implemented a device that can receive, process and transmit RF signals in less than 1ns [64].

In the following subsections, we explain the two attacks and present the basic idea of our solutions.

4.1.1 Transmission Coverage Attack

This attack is illustrated in Figure 4.1. It shows P, V, P's cloaking region R and V's transmission coverage C, where R and C partially overlap. If V sends a challenge and P answers the challenge, V can conclude that P must be in the region intersected by R and C, thus refining P's location.

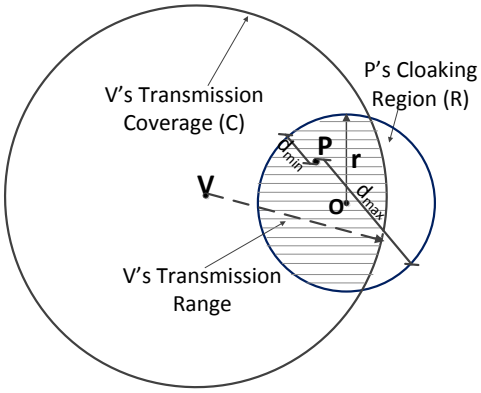


Figure 4.1 Transmission Coverage Attack

To prevent TCA, P must not respond to V's challenge unless its cloaking region is completely covered by V's transmission range. However, V may not disclose such information to P; even if V does, the information may not be trustworthy.

Here we propose using the received signal strength to estimate V's transmission range. Let W_r be the strength of the signal received by P. According to the Free Space Model [24], P can compute the distance that this signal can be further propagated as $r_c = (k\frac{W_r}{W_{min}})^{1/2}$, where k is a constant and W_{min} the minimum signal strength that is detectable to a networking interface card. We will refer to the circular region that centers on P's position with the radius of r_c as V's *minimal transmission coverage* (MTC).

P can avoid TCA by simply checking if V's MTC contains R, i.e., $d_{max} \leq r_c$, where d_{max} is the maximum distance from P to R's perimeter. When this condition is true, V's transmission range is guaranteed to cover the entire R, so P can answer V's challenge. Otherwise it drops the challenge. This strategy, however, assumes the worst scenario. Even if $MTC(P)$ just overlaps with R, V's transmission coverage may still cover the entire R, depending on V's position. Let $Prob_{TCA}$ be the probability

that V's transmission coverage covers R completely and d_{min} the minimum distance from P to R's perimeter. We have the following formula:

Theorem 1.

$$Prob_{TCA} = \begin{cases} 0 & r_c \geq d_{max} \\ 1 & r_c < d_{min} \\ 1-q & d_{min} < r_c < d_{max} \textit{ and } r > r_c \\ q & d_{min} < r_c < d_{max} \textit{ and } r < r_c \\ \frac{1}{2} & d_{min} < r_c < d_{max} \textit{ and } r_c = r \end{cases}$$

where $q = \frac{1}{\pi}\arctan(\frac{\sqrt{(d_{max}-r_c)(r_c-d_{min})}}{|r_c-r|})$.

If R is completely covered by V's MTC (i.e., $r_c \geq d_{max}$, see Figure 4.2(a)), there is no chance for V to have a TCA, so $Prob_{TCA} = 0$. On the other hand, if R completely covers V's MTC (i.e., $r_c \leq d_{min}$, see Figure 4.2(b)), we have $Prob_{TCA} = 100\%$. This is due to the fact that regardless V's position, there is no way that V's transmission coverage can cover R. The value of $Prob_{TCA}$ under these two extreme cases is easy to derive, but when R partially overlaps with V's MTC (i.e., $d_{min} < r_c < d_{max}$, see Figure 4.2(c)), the problem is more complicate. Our analysis shows that $Prob_{TCA}$ under this scenario has to do with the relationship between r and r_c. We will present this analysis in detail later.

Let S_t be the minimum probability that P wants to prevent the TCA. By deriving the reverse function of the above formula, we can compute the minimum r_c, denoted as $r_{c_{min}}$, for a given S_t:

$$r_{c_{min}} = r - (d_{max} - r) \cdot \cos(\pi \cdot S_t) \quad \text{for all } 0 \leq S_t \leq 1 \quad ‘ \tag{4.1}$$

Note that $r_{c_{min}} = d_{min}$ when $S_t = 0$ and $r_{c_{min}} = d_{max}$ when $S_t = 1$.

Given $r_{c_{min}}$, the corresponding minimum received signal strength, denoted as $W_{r_{min}}$, can be computed as $W_{r_{min}} = W_{min} \cdot \frac{r_{c_{min}}^2}{k}$. As such, given S_t, P computes $W_{r_{min}}$. When receiving a challenge from V, P checks the received signal strength W. If $W < W_{r_{min}}$, P does not reply. Otherwise, it sends its answer.

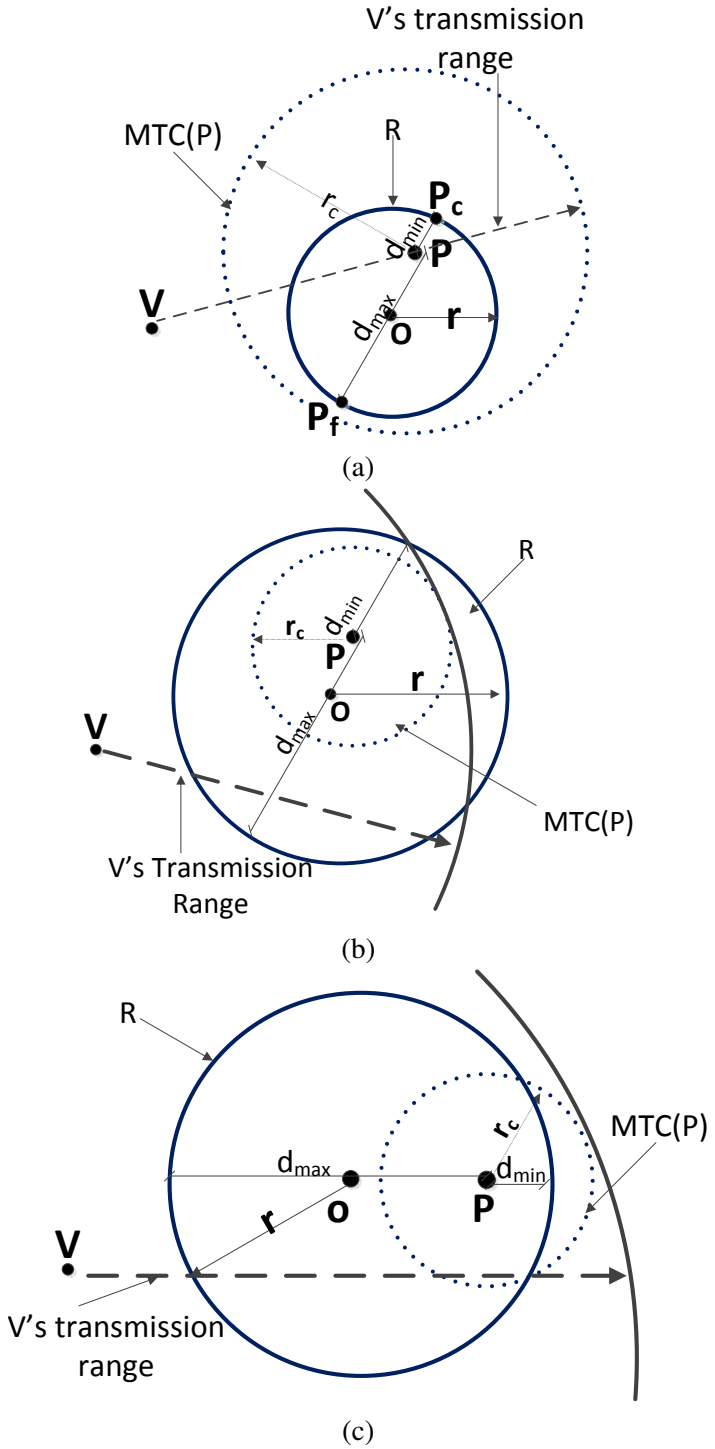


Figure 4.2 Three possible scenarios for TCA

4.1.2 Distance Bounding Attack

Here V measures the RTT of its communication with P and uses this time period to estimate its distance to P. Let t be the time duration from the time when V sends out a challenge to P to the time when V receives P's response. The physical distance ℓ between V and P must be no greater than $\Delta = t \cdot c/2$, where c is the speed of light. In other words, P must locate inside the circle that centers on V's position with a radius of Δ. We will refer to this circle as P's distance bounding circle (DBC) and Δ as P's upper bounding distance. If the DBC does not contain P's cloaking region completely, P's location is refined.

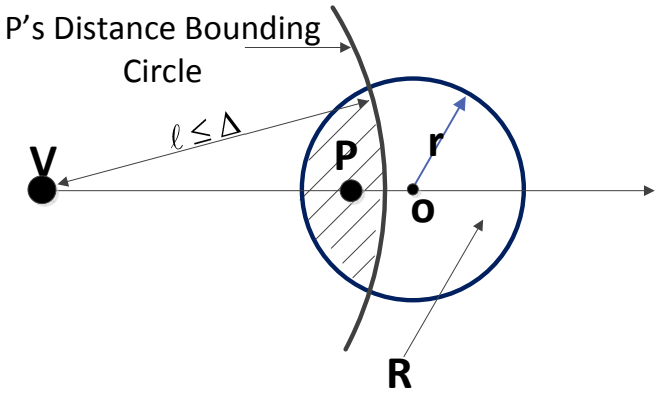


Figure 4.3 Distance Bounding Attack

To thwart such attack, P can delay some time period before replying a challenge. The question is how much this delay should be. It may first appear that P can just set the delay to be $2 \cdot d_{max}/c$ time units, where c is the speed of light. Since d_{max} is the maximum distance from P to the perimeter of its cloaking region, the DBC computed by V based on the measured Δ will cover P's cloaking region completely.

This approach, however, allows V to refine P's location. Since V knows that the delay is set to be $2 \cdot d_{max}/c$ time units and d_{max} must be at least r, it can always subtract r from the measured Δ. P's location is refined if the DBC based on the adjusted Δ does not cover R completely. Indeed, based on the measured Δ, V can refine P's location even further. Suppose V is on R's perimeter, in which case V knows that Δ must be no less than $2r$ but no greater than $4r$, if P is in R. Let R_1 and R_2 be R's two subregions divided by the circle that is centered on V with a radius of r (See 4.4). If Δ measured by V is greater than $3r$, then V can conclude that P cannot be in R_1. This is due to the fact that if P is in R_1,

its distance to V is at most r and d_{max} is at most $2r$, i.e., $\Delta = r + d_{max} \leq 3r$.

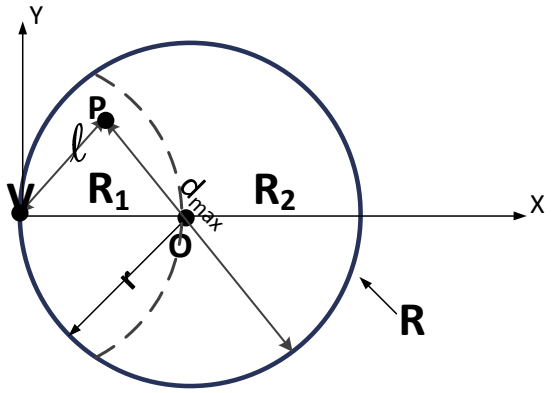


Figure 4.4 Location refinement allowed by the simple delay protocol

To circumcise the above problem, we propose that P obtains a random number δ from a uniform distribution defined over $[d_{min}, \Lambda]$, and delays its response by $\frac{2 \cdot \delta}{c}$ time units. The setting of Λ has to do with the minimum probability that P wants to prevent the DBA. Let this probability be S_d. The value of Λ can be computed using the following formula, the details of which will be discussed later:

$$\Lambda = \frac{0.87 \cdot r}{1 - S_d} \tag{4.2}$$

Note that S_d and Λ are related, so P must keep secret these two parameters. Choosing a random delay prevents V from subtracting some fixed value (e.g., r) from Δ it measures. If $\delta \geq d_{max}$, P can be sure that its cloaking region will not be refined, because Δ measured by V will be at least equals to $2r$. But if $\delta < d_{max}$, there is a risk that P's location gets refined. In this case, P can either assume the risk and transmit the response or obtain a new δ. Note that V must not know what choice P takes. If V knows that P always chooses the second option, V can substract r for sure. In our solution, we let P obtains δ only once. We will present our analysis on the probability that this strategy can prevent location refinement in Section 4.3.2.

4.2 Proposed: Location Privacy-aware Location Verification (LPLV)

Before participating in location verification, P sets S_t and S_d, the minimum probability that it wants to prevent TCA and DBA, respectively. Given a pair of values (S_t, S_d), P's safety level can be computed as $S_t \times S_d$. Given the values of S_t and S_d, P computes the corresponding minimum signal strength $W_{r_{min}}$ and minimum delay δ using the formula presented earlier.

We now adapt the classic distance bounding protocol (DBP) [11], which was originally designed to prevent mafia fraud attack and false distance attack. The proposed verification process consists of three phases, as illustrated in Figure 4.5. We explain as follows.

Phase 1: Initialization phase for V and P

- P generates a k-bit random string $(m_1|...|m_k)$, where m_i $(1 \leq i \leq k)$ is either 0 or 1. This string is then sent to V using a secure commitment scheme [12]. This allows P to commit the string to V while keeping it hidden before revealing it at the last step of the verification process. P computes the minimum signal strength required to prevent TCA, $W_0(P) = \frac{W_{min}}{k} \cdot (2r + d_{max})^2$ and it chooses at random a value δ from the interval $[d_{min}, \Lambda]$.

- V also generates a k-bit random string (α_1, ..., α_k), where α_i $(1 \leq i \leq k)$ is either 0 or 1.

Phase 2: V and P exchange k-bit challenges and responses.

- V starts this phase by sending P the first challenge, α_1. It records the time of sending, t_1, and then waits for P's response for at most T_{max} time units. If V does not receive the response during this time period, it stops the verification process and rejects P's location claim. Otherwise, V computes the first upper-bound distance $\Delta_1 = \frac{t_2 - t_1}{2} \cdot c$, where t_2 is the time when it receives the response. Let Δ_{min} be the minimum distance from V to P's cloaking region. If $\Delta_1 \leq \Delta_{min}$, V rejects P's location claim. Otherwise, it proceeds to send the second challenge α_2 and compute Δ_2, and so on so forth, until α_k is sent and Δ_k computed.

- When P receives a challenge, say α_i $(1 \leq i \leq k)$, it first determines whether or not to respond. This is done by checking if the signal strength of the received challenge, W_r, is greater or equals to $W_{r_{min}}$. If W_r is less than $W_{r_{min}}$, P drops the challenge. Otherwise, it waits $\frac{2\delta}{c}$ time units, and then sends its response $\beta_i = m_i \oplus \alpha_i$.

Phase 3: This is the phase when V and P verify the k challenges-responses were successfully exchanged.

- P opens its commitments of the k-bit string $(m_1|...|m_k)$, which it committed securely to V in Phase 1. P concatenates the $2k$ bits of the challenges and corresponding responses (i.e., $\alpha_1|\beta_1|\ldots|\alpha_k|\beta_k$). Let m be this string. P signs m with its secret key and sends the resulting signature to V.

- After receiving the above information, V checks if $\alpha_i \oplus \beta_i$ is equal to m_i for all i from 1 to k. If this is true, V computes m in the same way as P did and verifies whether the signature it received is indeed a correct signature of P on m. If this is true, V computes the upper-bound of its distance to P, (Δ_u), as the maximum of all Δ_i $(1 \leq i \leq k)$. V accepts P's location claim if $\Delta_u \leq \Delta_{accept}$, where Δ_{accept} is the maximum distance V allows in order to accept P's location claim.

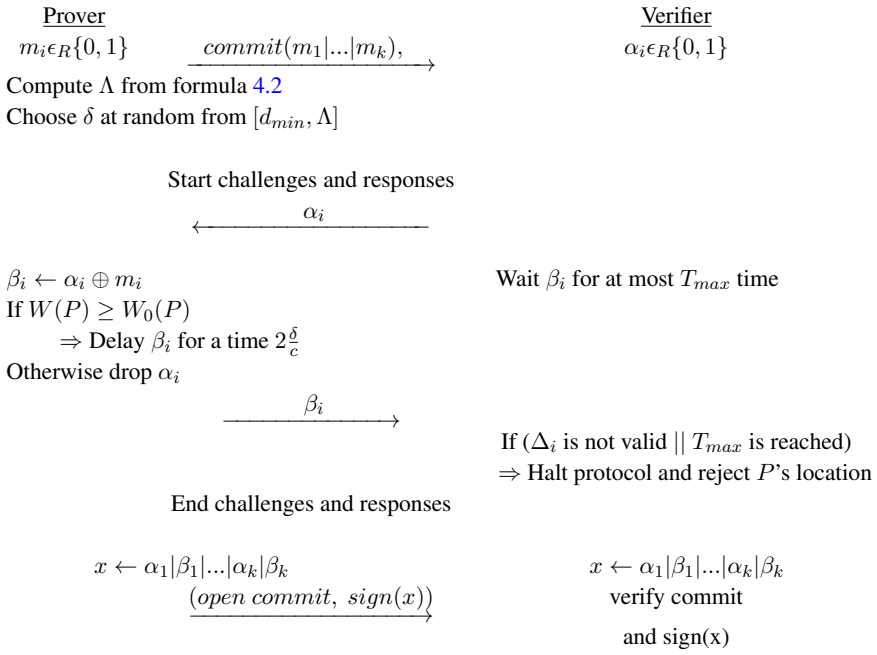


Figure 4.5 LPLV Protocol for a stationary prover and verifier

4.3 Security Analysis

In this section, we analyze the probability that P can prevent the two location refinement attacks.

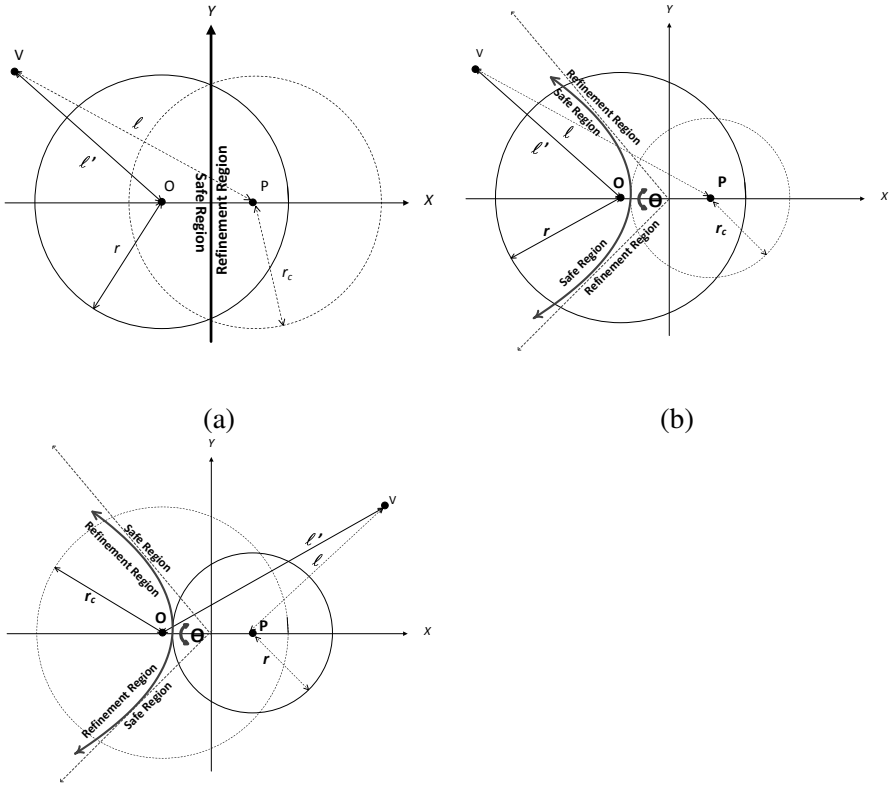


Figure 4.6 Calculating the probability V can refine P's cloaking region

4.3.1 Probability of Preventing TCA: S_t

If V's MTC covers P's entire cloaking region R, $Prob_{TCA}$ is 0. If V's entire MTC is inside R, $Prob_{TCA}$ is 100%. We now consider the scenario when the MTC and R partially overlap each other. We use the following notations, where are illustrated in Figure 4.6.

- ℓ: the distance from V to P;
- ℓ': the distance from V to O (i.e., the center of P's cloaking region);

- r: the radius of P's cloaking region;
- r_c: the radius of V's MTC.

We have the following observation: if $\ell' + r \leq \ell + r_c$, the entire cloaking region is within V's transmission coverage. In other words, whether or not P can avoid the attack has to do with V's position. The problem is, P does not know V's location. Our idea to solve this problem is to identify the regions where V cannot refine P's location if it is inside, and then compute $Prob_{TCA}$ based on the probability of V being in those regions.

Let's set a reference frame X-Y at the middle point between P and O, as illustrated in Figure 4.7. P and O are both on the X-axis, where O positions at $(-\frac{d_{max}-r}{2}, 0)$ and P at $(+\frac{d_{max}-r}{2}, 0)$. Let (x, y) be V's coordinate in the reference frame. We consider three possible scenarios.

Case 1: $r = r_c$. Here V's MTC has the same radius as R. If $x \leq 0$ (i.e., V is in left side of Y-axis), V cannot refine P's location. This is due to the fact that V's distance to O is always no greater than its distance to P. On the other hand, if $x >= 0$, V's distance to O is always no less than its distance to P, in which case P's location is refined. Since V could be anywhere, P can compute $Prob_{TCA}$ as 50%.

Case 2: $r > r_c$ (see Figure 4.7). If $x >= 0$, ℓ is always less than ℓ'. Since $r > r_c$, we have $\ell + r_c < \ell' + r$. Thus, if V is in the right side of Y-axis, P's location will not be refined. We now consider the case when $x < 0$. We observe that the equation $\ell - \ell' = r - r_c$ satisfies the definition of an hyperbola. This curve is "the locus of a point in a plane so that the difference between its undirected distances from two fixed points is a non-zero constant [5]." In our case, the undirected distances are l' and ℓ', the fixed points are P's location and O, and the non-zero constant is $r - r_c$. This hyperbola separates the entire region into two regions, R_s (Safe Region) and R_u (Refinement Region), as illustrated in Figure 4.7. If V is in R_s, P's location cannot be refined. Otherwise, it is refined.

Let $a = \frac{|r-r_c|}{2}$, $b = \frac{d_{max}-r}{2}$, and $c^2 = b^2 - a^2$, where $c \geq 0$. We can rewrite the inequality $\ell - \ell' \geq r - r_c$ as $\frac{x^2}{a^2} - \frac{y^2}{c^2} \geq 1$. Solving this inequality for y, we have $-\sqrt{\frac{c^2x^2}{a^2} - 1} \leq y \leq +\sqrt{\frac{c^2x^2}{a^2} - 1}$. Since $\frac{c^2x^2}{a^2} \geq 1$, we have $x \geq \frac{a}{c}$ or $x \leq -\frac{a}{c}$. Because $x < 0$, R_s is the region where $x \leq -\frac{a}{c}$ and $|y| \leq +\sqrt{\frac{c^2x^2}{a^2} - 1}$ and it can be seen as the hatched region "enclosed" by the hyperbola as illustrated in Figure 4.7.

Let C_a be a circle centered at location (0,0) and radius $r_a \geq a$. $Prob_{TCA}$ can then be computed as $\lim_{r_a \to +\infty} \frac{Area(C_a \cap R_s)}{Area(C_a)}$. Let ($x_0$>0,$y_0$>0) be the point of intersection between C_a and the hyperbola, where $x_0 = \frac{a \cdot \sqrt{{r_a}^2 + c^2}}{\sqrt{c^2 + a^2}}$ and $y_0 = \frac{c \cdot \sqrt{{r_a}^2 - a^2}}{\sqrt{c^2 + a^2}}$. We have

$$\lim_{r_a \to +\infty} 2 \cdot \frac{\int_0^{y_0} (\sqrt{{r_a}^2 - y^2} - \frac{a}{c} \cdot \sqrt{y^2 + c^2}) dy}{\pi \cdot {r_a}^2}$$

Solving this integral, we have $Prob_{TCA} = 1 - \frac{1}{\pi} \arctan(\frac{\sqrt{(d_{max} - r_c)(r_c - d_{min})}}{(r_c - r)})$.

Case 3: $r > r_c$. This case is similar to when $r < r_c$. Following the same procedure, we have $Prob_{TCA} = \frac{1}{\pi} \arctan(\frac{\sqrt{(d_{max} - r_c)(r_c - d_{min})}}{(r - r_c)})$

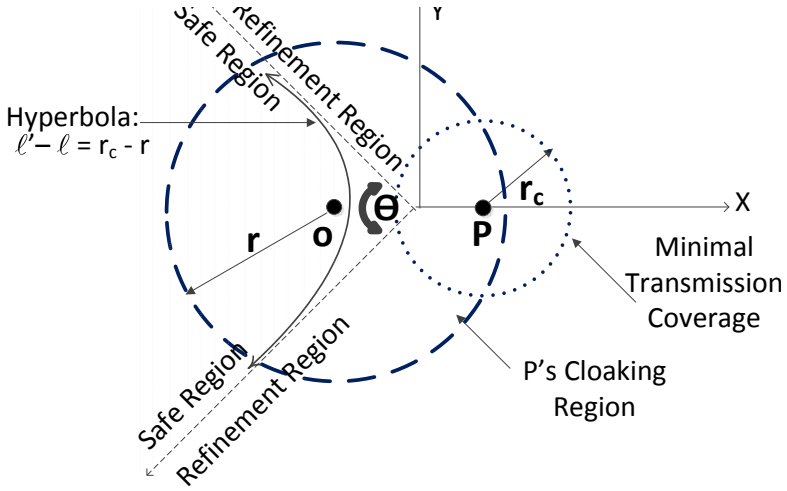


Figure 4.7 Scenario where the success of TCA depends on V's location

4.3.2 Probability of Preventing DBA: S_d

In our solution that prevents DBA, P delays $\frac{2\cdot\delta}{c}$ time units. Here δ is a random value selected from the interval $[d_{min}, \Lambda]$, where $\Lambda \geq d_{max}$ is determined by S_d, the minimum probability that it wants to prevent DBA. We now analyze the relationship between Λ and S_d, from which we can compute Λ for a given S_d. We first assume V is on the perimeter of P's cloaking region and then show that Λ computed for this special case is enough to achieve S_d when V is in other regions.

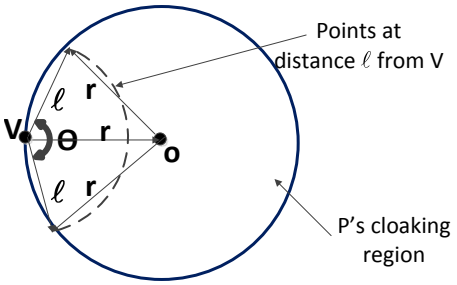


Figure 4.8 Computing the p.d.f of ℓ when V is in the border of R

Let ℓ be the distance from V to P (see Figure 4.8). Since any position (x, y) in R has an equal probability to be P's position, the probability density function (p.d.f) for ℓ, denoted as $f(\ell)$, can be derived as follows.

First, the total number of points whose distance to V is equal to ℓ is $\pi(\ell + d\ell)^2 - \pi\ell^2 = 2\pi \cdot \ell d\ell$. Among them, there are $\frac{\theta}{2\pi} \cdot 2\pi \cdot \ell \cdot d\ell$ inside the cloaking region (see Figure 4.8 for θ). Thus, the p.d.f of the points inside R having a distance ℓ is $\frac{\theta}{\pi \cdot r^2} \cdot \ell \cdot d\ell$. Using the law of cosines to compute $\frac{\theta}{2}$, we have $r^2 = \ell^2 + r^2 - 2\ell \cdot r \cdot \cos(\frac{\theta}{2})$. Since $\theta = 2 \cdot \arccos(\frac{\ell}{2r})$, we have $f(\ell) = 2 \cdot \arccos(\frac{\ell}{2r}) \frac{\ell \cdot d\ell}{\pi \cdot r^2}$

Since δ is randomly chosen from $[d_{min}, \Lambda]$, the p.d.f for δ, denoted as $f(\delta)$, is:

$$f(\delta) = \begin{cases} \frac{1}{\Lambda - d_{min}} & \text{for } d_{min} \leq \delta \leq \Lambda \\ 0 & \text{elsewhere} \end{cases}$$

Because ℓ and δ are independent to each other, we have the p.d.f. $f(\ell, \delta) = f(\ell) \cdot f(\delta)$. Since $\Delta = \ell + \delta$, we have the following the p.d.f. $f(\Delta,\delta)$:

$$f(\Delta,\delta) \quad = \quad \begin{cases} \frac{2(\Delta-\delta)}{\pi r^2(\Lambda-d_{min})} \cdot \arccos(\frac{\Delta-\delta}{2r}) & \text{for } 0 \leq \Delta-\delta \leq 2r \\ & \quad d_{min} \leq \delta \leq \Lambda \\ 0 & \text{elsewhere} \end{cases}$$

Note that P's location is refined if the upper bound distance computed by V is less than $2r$. So S_d can be computed as the probability that P obtains a δ in $[d_{min}, d_{max}]$. Since $0 \leq \Delta - \delta \leq 2r$, then $\delta \leq \Delta \leq 2r$. In the worst case scenario $d_{min} = 0$ and $d_{max} = 2r$, since $d_{max} + d_{min} = 2r$, R has the highest risk to be refined.

$$\begin{aligned} 1 - S_d &= \int_{d_{min}}^{d_{max}} \int_{\delta}^{2r} f(\Delta,\delta) d\Delta d\delta \\ &\leq \int_{0}^{2r} \int_{\delta}^{2r} f(\Delta,\delta) d\Delta d\delta \\ &= 2 \cdot \frac{(9 \cdot \pi - 16)}{9 \cdot \pi} \frac{r}{\Lambda} \\ &= \frac{0.87 \cdot r}{\Lambda} \\ \Lambda &\leq \frac{0.87 \cdot r}{1 - S_d} \end{aligned} \tag{4.3}$$

When $d_{min} = r$, it means P is located at O and P only needs to delay r to prevent DBA. Then in this case the probability that R is refined is zero.

We have computed the exact formula to determine the probability of refinement for location of P. To simplify its presentation, we have divided this formula into four auxiliary functions f_1, f_2, f_3 and f_4.

$$\int_{d_{min}}^{d_{max}} \int_{\delta}^{2r} f(\Delta,\delta) d\Delta d\delta \quad = \quad f_1(d_{min}, r) + f_2(d_{min}, r) + f_3(d_{min}, r) + f_4(d_{min}, r)$$

$$f_1(d_{min}, r) = \frac{1}{12} \left(r^3 \left(\frac{d_{min}(-d_{min} + 4r)}{r^2} \right)^{3/2} - 2r \sqrt{1 - \frac{d_{min}^2}{4r^2}} \left(-d_{min}^2 + 4r^2 \right) \right)$$

$$f_2(d_{min}, r) = \frac{1}{18}\left(2\sqrt{1-\frac{d_{min}^2}{4r^2}}\cdot r\left(d_{min}^2+8r^2\right) - r\sqrt{\frac{d_{min}(-d_{min}+4r)}{r^2}}\left(d_{min}^2-4d_{min}r+12r^2\right)\right)$$

$$f_3(d_{min}, r) = \frac{1}{18}\left(3(-d_{min}+2r)^3 \text{ArcCos}\left[1-\frac{d_{min}}{2r}\right] - 3d_{min}^3 \text{ArcCos}\left[\frac{d_{min}}{2r}\right]\right)$$

$$f_4(d_{min}, r) = -2r^3\left(\sqrt{1-\frac{d_{min}^2}{4r^2}} - \frac{1}{2}\sqrt{\frac{d_{min}(-d_{min}+4r)}{r^2}}\right)$$
$$-2r^3\left(-\left(1-\frac{d_{min}}{2r}\right)\text{ArcSin}\left[1-\frac{d_{min}}{2r}\right] + \frac{d_{min}}{2r}\text{ArcSin}\left[\frac{d_{min}}{2r}\right]\right)$$

Note that if $\Delta \geq 2r$, there is no risk of location refinement, because for each possible value for $0 \leq \ell \leq 2r$, there exists a value for $0 \leq \delta \leq \Lambda$ that satisfies equation: $\Delta = \ell + \delta$. If $\ell = 0$, then $\delta = \Delta > 0$. If $\ell = 2r$, then $\delta = \Delta - 2r \geq 0$. Then we can conclude that each location within P's cloaking region is equiprobable to be P's location.

In the above analysis, we assume V is on the perimeter of P's cloaking region. We now consider the cases when V is in other locations and we determine an estimation of the probability that V can successfully refine P's cloaking region when P moves away from the center of P's cloaking region, O. Let P_r denote the probability that V refines R and let ε denote V's distance to O and P chooses $\delta \in [d_{min}, \Lambda]$, and suppose $\Lambda \geq d_{max}$.

Case 1: $\varepsilon = 0$. Here V is located at 0 and since P delays at least d_{min}, it always can prevent location refinement, then $P_r = 0$

Case 2: $0 < \varepsilon \leq r$. Here the minimum δ that P can obtain is $r - \varepsilon$ and the minimum δ needed to insure location refinement is prevented is $\varepsilon + r$. Then P_r can be estimated as $\frac{2\varepsilon}{\Lambda - r + \varepsilon}$. Note we have estimated an upper bound for P_r, since it is possible that even tough $\delta < r + \varepsilon$ V is still not able to refine R due to the fact the distance from P to V compensates the delay needed to prevent DBA. As we expected $P_r = 0$ when V is in O and $P_r = \frac{2r}{\Lambda} > \frac{0.87r}{\Lambda}$ when V is R's perimeter.

Case 3: $\varepsilon \geq r$. Here V is always located out of R and assuming P remains within R, then the requirements to prevent DBA are the same as case 2 when $\varepsilon = r$, then $P_r < \frac{2r}{\Lambda}$.

Since V does not know where P is located, it needs to find a place where the P_r can achieve its largest value. We discard position O, since always $P_r = 0$, but when $\varepsilon \geq r$, there exists the possibility V has the highest chance to refine R ($\frac{2r}{\Lambda}$). Then we choose to locate V in the perimeter of R.

4.4 Performance Evaluation

We have designed a simulator to evaluate our proposed technique and validate our theoretical models. We have evaluated how effective our protocol is to prevent DBA and TCA from happening. In our results, our technique is either identified as "LPLV" or by the safety level used to prevent either TCA or DBA. As a baseline approach we have measured the performance of a technique that does not delay any response or does not verify the signal strength. This latter technique is labeled as "Baseline" in our results. To evaluate the proposed approaches the following three metrics are used.

- *Ave. Hit Ratio*: This metric is defined as the average ratio between the number of successful refinement attacks and the total number of attacks. A value of 1 means P's cloaking region is being refined in every attack. On the contrary a value of 0 means P's cloaking circle is never refined.

- *Ave. Refinement Ratio*: Given a transmission range or a distance bound computed from V's location, this metric is defined as the average area intersected by a distance bounding circle centered at V's location and P's cloaking region divided by the area of P's cloaking region. A value of 1 means P's cloaking region has been fully protected from the location refinement attacks. On the contrary a value of 0 means P's location has been identified with accuracy.

- *Ave. Tightness Ratio*: This metric is measured when V cannot refine R and it is defined as the average upper-bound distance from V to P divided by the radius of P's cloaking region.

The parameters of our simulations are described by table 4.1.

4.4.1 Evaluation of LPLV against DBA

Initially we evaluate how our technique performed against DBA when S_t is kept secret and V assumes all possible S_t are equally probable to be one used by P. Then P's delay is always chosen as the minimum time that satisfies P's safety level. Node P is located in a random position within its cloaking region and node V is located at any random position in the perimeter of R. We assume V's transmission range is at least $2r$ in order that V is able to reach any point in R. We also fixed the safety level at various ranges as described in table 4.1. We repeated this simulation until the Hit Ratio becomes

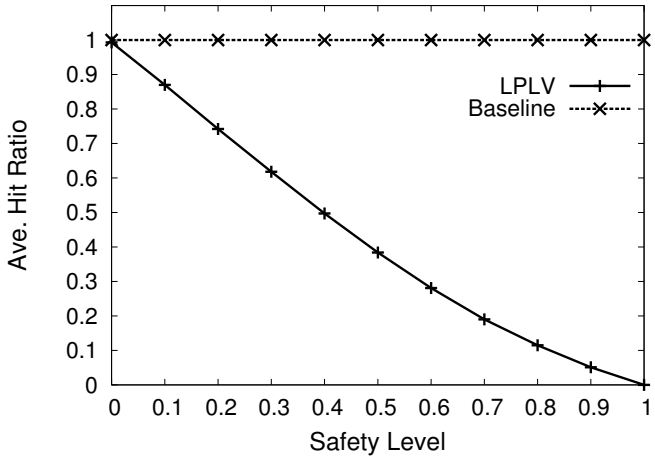
Table 4.1 Parameters - LPLV Protocol

Parameter	Default	Variation
Network domain	1,000 x 1,000	N/A
Safety Level DBA (S_d)	0.5	[0.0, 1.0]
Safety Level TCA (S_t)	0.5	[0.0, 1.0]
Radius P's cloaking region	100	N/A
Distance from V to P	100	[0, 200]
Transmission Range	200	[10,200]
Number of challenges	10	[10, 50]

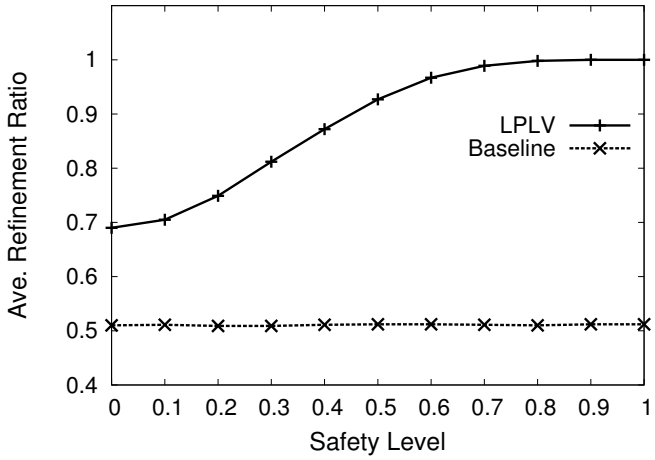
stabilized. In Figure 4.9(a) we observe the Hit Ratio of our technique decreases when S_d is increased until 1.0. This result was expected since the higher the safety level is, the longer the delay is to transmit a response. Figure 4.9(a) confirms our theoretical model since the Hit Ratio is close to the theoretical estimation (1-S_d). When the baseline technique is used, we observe that R is refined in all cases.

These later results are emphasized by Figure 4.9(b). We can observe when the higher the safety level is the closer the Refinement Ratio is to one, which means almost no refinement is achieved. When $S_d = 0$, there is a gap between our approach and the baseline because our technique at least delays a response for $\frac{d_{min}}{c}$ time units. Since the baseline approach does not delay its response, we can observe the cloaking region R is refined to about 50% on average.

We evaluate how our technique performed against a protocol attack (This attack is explained at Figure 4.4). To prevent this attack, node P chooses only once a random number from interval $[d_{min}, \Lambda]$ which is then used to delay any response. We set node P in a random position within its cloaking region and the distance from V to the center of R (ℓ') varies from 0 to 200. We fixed the safety level at various ranges as described in table 4.1. We repeated this simulation until the Hit Ratio becomes stabilized. In Figure 4.10(a) we observed the Hit Ratio of our technique becomes larger when the distance from V to the center of P's cloaking region is increased. This result was expected because when V is in the center of R, P only needs to delay d_{min} to prevent location refinement. However when V is deployed away from the center of R, P needs to increase this delay to almost $2r$ in the worst case to achieve the same goal. As we expected, this ratio becomes constant when the distance between R and V is greater than the radius of R. As we expected the greater the safety factor is the lesser the probability to refine R is.



(a)



(b)

Figure 4.9 Performance of our protocol against DBA when S_d is secret

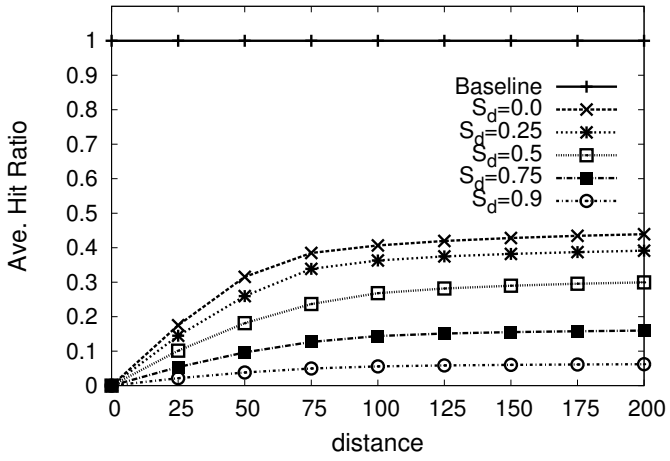
When "Baseline" is intentionally used, node V achieves 100% of success in refining R.

These later results are emphasized by Figure 4.10(b). From this figure, we observed for our technique that when the safety level is higher, the Refinement Ratio is closer to one, which means almost no refinement is achieved. However, when "Baseline" is intentionally used, the cloaking region R is refined to about 50% on average. In Figure 4.10(c) we can observe the impact of delaying a response on the upper-bound distance. As we expected the *Tightness Ratio* becomes larger when the distance between V and R is increased.

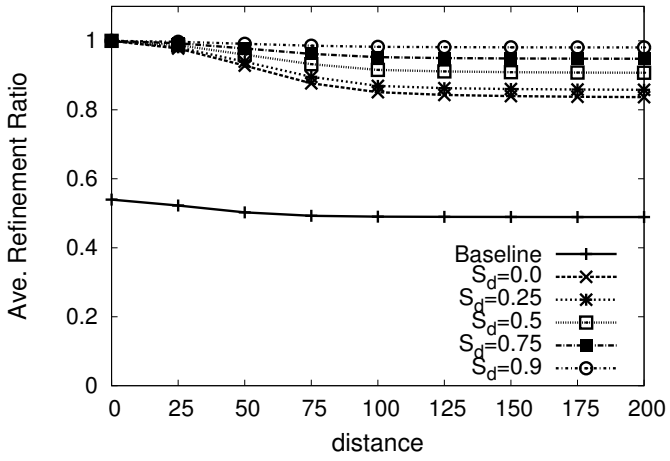
In figures 4.11(a), 4.11(b) and 4.11(c), we illustrated the results obtained when the distance from R to V is fixed and we varied the safety level. We can observe from figure 4.11(a), the probability of location refinement remains constant when V is located either in the perimeter of R (Dist =100) or out of this region (Dist=200). When V is in the center of R, P needs simply to delay d_{mim} to prevent location refinement and that is the reason there is no refinement for all ranges of the safety level. In figure 4.11(b), we observe how the Refinement Ratio gets closer to one when the safety level is increased. However, this is achieved at the expense of the Tightness Ratio 4.11(c).

4.4.2 Evaluation of LPLV against TCA

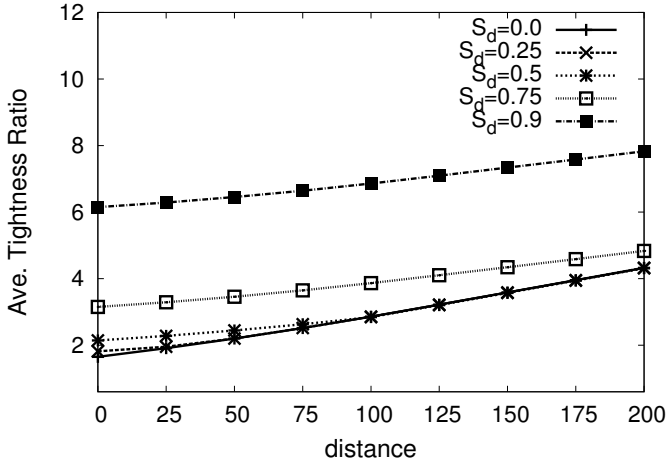
Here we evaluate how our technique performed against TCA when the safety level S_t is secret and V assumes any value for S_t is equally likely to be the one chosen by P. Here node P decides whether or not a challenge is responded or dropped, if the probability that R is refined is greater than its safety factor (See formula 1). We set node P in a random position in the perimeter of P's cloaking region. Since the $r = 100$, we varied V's transmission range from 10 to 200. We also fixed the safety level at various ranges as described in table 4.1. We repeated this simulation until the Hit Ratio becomes stabilized. In Figure 4.12(a) we observed that the Hit Ratio of our technique becomes higher when node V's transmission range is increased. Initially if the transmission range is not enough to reach node P, then the probability that V can refine P's cloaking region is low. However, when V's transmission range is increased, V is able to reach many points in R and P is in situation to respond more often V's challenges. However, how often node P replies depends on its safety level. As we expected the higher the safety level, the larger the signal strength that P demands to reply a challenge. Thus, the probability that R is refined becomes smaller when S_t is increased.



(a)

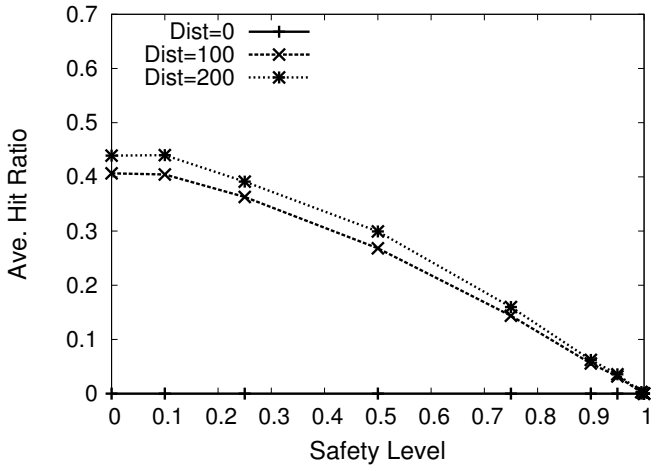


(b)

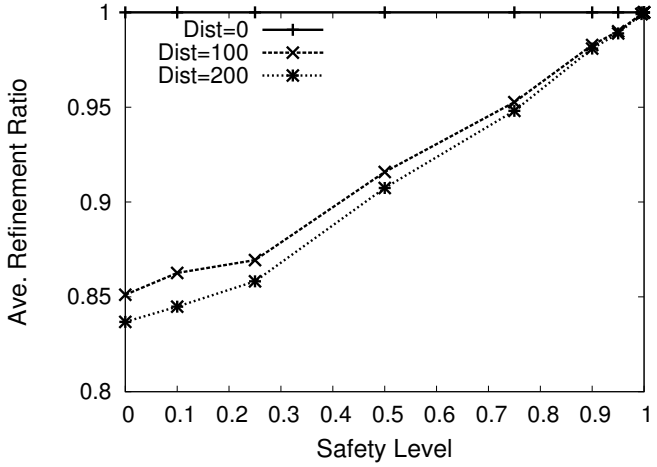


(c)

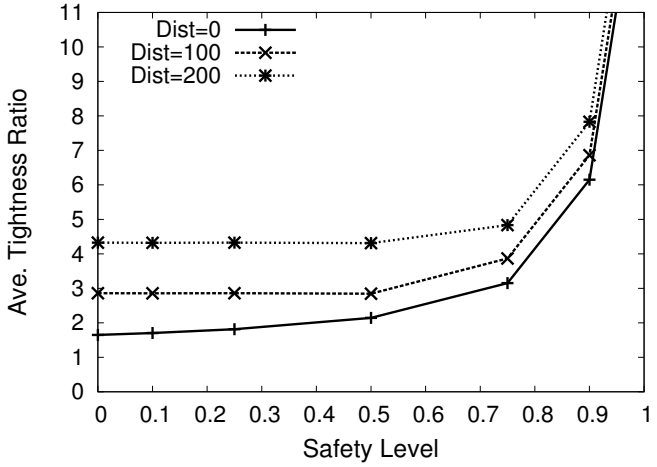
Figure 4.10 Results of LPLV when delay is chosen randomly and the distance from V to R is varied.



(a)



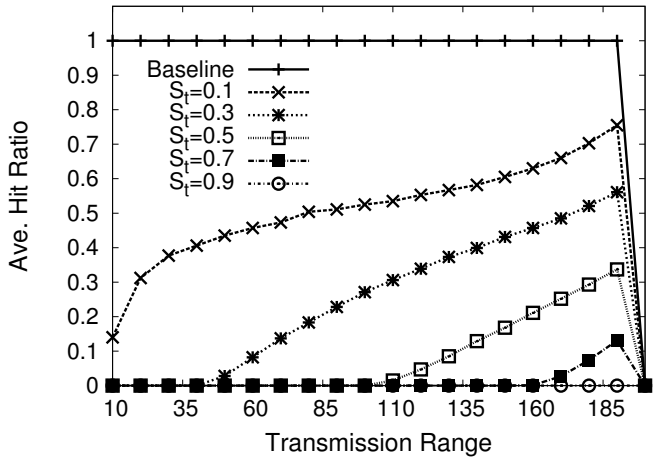
(b)



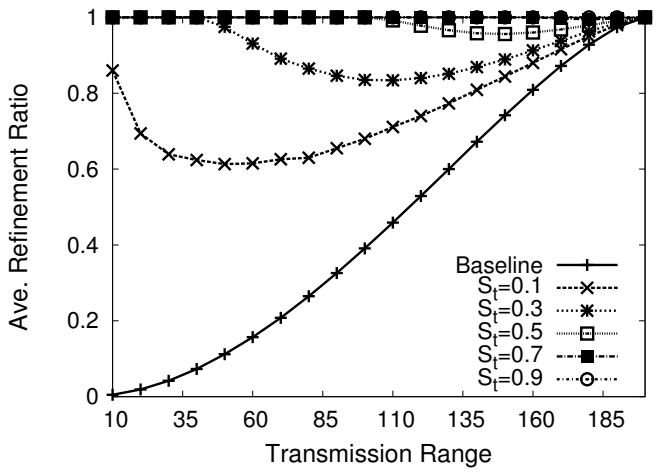
(c)

Figure 4.11 Results of LPLV when delay is chosen randomly and S_d is varied.

The consequences of a location refinement can be better visualized from Figure 4.12(b). Here we observe when the transmission range is short (less than the radius of R) and if P responds a challenge, V is able to identify P's location with high accuracy. When the transmission range becomes larger, the probability that R is refined is higher, but the Refinement Ratio becomes larger, which means although R is refined more often, V is not able to obtain P's location with high accuracy. We also observe when the safety level is increased and gets closer to 1 the Refinement Ratio grows faster to one. When the "Baseline" is intentionally used, node V achieves 100% of success in refining R, but the Refinement Ratio becomes larger due to the transmission range is larger.



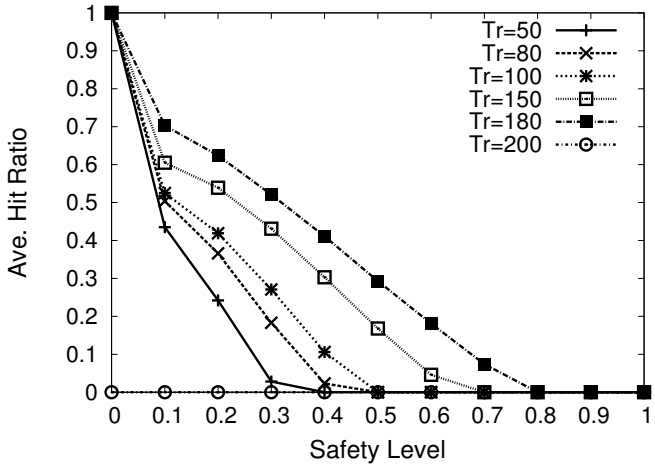
(a)



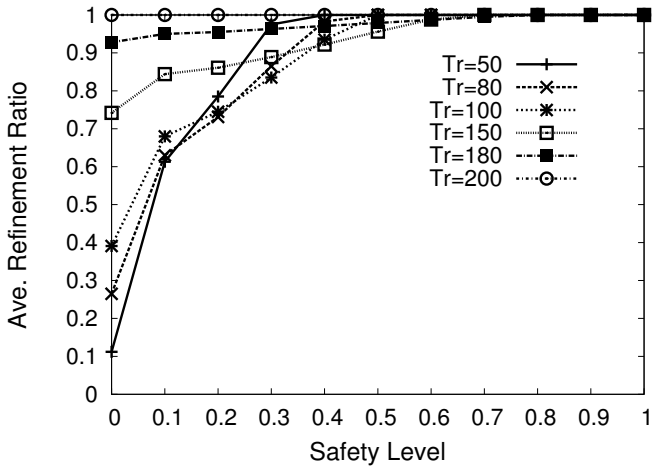
(b)

Figure 4.12 Performance of LPLV against TCA when verifier is located at different distances from R

In figures 4.13(a), 4.13(b) and 4.13(c), we illustrated the results obtained when the safety level is varied and the transmission range is kept constant. From figure 4.13(a) we observe the probability of location refinement decreased when the safety level is increased for all cases because node P demands more signal strength to reply to a challenge. From figure 4.13(b) we observe the Refinement Ratio gets closer to 1 when the safety level is increased.



(a)



(b)

Figure 4.13 Performance of LPLV against TCA when the safety level is varied.

CHAPTER 5. Conclusion and Future Work

This thesis investigates the impact of location cloaking on two applications and makes the following contributions:

- *Query Processing*: Location-cloaked queries present the problems to both server and client. The server needs to retrieve and transmit more information, while the client downloading this information finds most of it is useless. We have addressed these problems with a generic model capable of processing various types of queries such as KNN and range queries within a single unified framework. When the system is under loaded, a query is processed upon its arrival without any delay. When the system is overloaded, the queries pending in the queue are processed in batch, and our model shows its true advantage in this scenario. Since these queries may overlap in their interested regions, our query decomposition ensures that a relevant cell is retrieved only once, thus minimizing the server workload. Given a set of relevant cells, our new scheduling technique orders the retrieval of these cells. This technique is unique in its consideration of the possible position of a client's location in dealing with the dilemma of minimizing average query processing latency and maximizing fairness. When returning a set of query results to a client, we apply air indexing to allow the client to download only the data that it truly needs, thus minimizing battery consumption.

- *Location Verification*: The challenges of verifying whether or not a node indeed locates in a cloaked region come from the fact that the verification process may allow one to refine the node's location within the region. We have identified two such location refinements: transmission coverage attack (TCA) and distance bounding attack (DBA). For TCA, we propose a solution that allows the prover to decide whether or not to response a challenge based on the received signal strength. We show the relationship between the received signal strength and the probability of

preventing the TCA. For DBA, we propose that the prover delays its response by a certain time period. We show how this delay can be computed based on the minimum probability that the prover wants to prevent the DBA. These two solutions are then integrated into a novel location verification technique that allows a prover disclosing a cloaked region to be verified while providing it a certain level of guarantee that its location will not be refined during the verification process. The effectiveness and correctness of our techniques are validated using simulation.

To our knowledge, there is little work toward mitigating the impact of location cloaking on the applications that rely on location information. We plan to extend our work of the two research problems as follows:

- *Query Processing*: In our current work, we assume only location-cloaked queries that retrieve stationary location-based information such as hotel and gas stations. We plan to extend our proposed platform to support efficient processing of queries in moving object database management. Such queries retrieve mobile objects (e.g., the object nearest to a gas station) and the results may keep changing as the objects move continuously.

- *Location Verification*: Our current techniques assume a single verifier and both verifier and prover are stationary. We plan to extend our work to deal with the presence of multiple verifier and the mobility of verifier and prover. Moreover, we will investigate location cloaking in the context of geographic ad hoc routing. Clearly, to prevent TCA, a node needs to avoid forwarding a packet if its cloaking region is not fully covered by the received signal; to prevent DBA, a node needs to delay its forwarding. It is interesting to investigate how these issues impact the performance of existing routing techniques.

BIBLIOGRAPHY

[1] O. Abumansoor and A. Boukerche. A secure cooperative approach for nonline-of-sight location verification in vanet. *IEEE Transactions on Vehicular Technology*, 61:275–285, Jan. 2012.

[2] S. Acharya, R. Alsonso, M. Franklin, and S. Zdonik. Broadcast disks: Data management for asymmetric communication enviroments. In *Proc. of ACM Int'l Conf. on Management of Data (SIGMOD'95)*, pages 199–210, San Jose, CA, USA, April 19-21 1995.

[3] S. Acharya and S. Muthukrishnan. Scheduling on-demand broadcasts: New metrics and algorithms. In *Proc. of IEEE Int'l Conf. on Mobile Computing and Networking (MOBICOM'98)*, pages 43–54, Dallas ,TX ,USA, October 25-30 1998.

[4] D. Aksoy and M. Franklin. RxW: a scheduling approach for large-scale on-demand data broadcast. *ACM/IEEE Trans. on Networking*, 7(6):846–860, 1999.

[5] H.-G. Ayre and R. Stephens. *A First Course in Analytic Geometry*. D. Van Nostrand Company, Inc, 1956.

[6] B. Bamba, L. Liu, P. Pesti, and T. Wang. Supporting anonymous location queries in mobile environments with privacygrid. In *Proc. of the 17th int'l con. on World Wide Web (WWW'08)*, pages 237–246, 2008.

[7] Jie Bao, Haiquan Chen, and Wei-Shinn Ku. Pros: a peer-to-peer system for location privacy protection on road networks. In *Proc. of the 17th ACM SIGSPATIAL Int'l Conf. on Advances in Geographic Information Systems*, GIS '09, pages 552–553, 2009.

[8] Jie Bao, Chi-Yin Chow, Mohamed F. Mokbel, and Wei-Shinn Ku. Efficient evaluation of k-range nearest neighbor queries in road networks. In *Proceedings of the 2010 Eleventh International Conference on Mobile Data Management*, MDM'10, pages 115–124, 2010.

[9] S. Basagni, I. Chlamtac, V. R. Syrotiuk, and B. A. Woodward. A Distance Routing Effect Algorithm for Mobility (DREAM). In *Proc. of the 5th Annual Int'l Conf. on Mobile Computing and Networking (MOBICOM'98)*, pages 76–84, Dallas, Texas, U.S.A, 1998.

[10] C. Bettini, X. Wang, and S. Jajodia. Protecting Privacy Against Location-Based Personal Indentification. In *Proceedings of the 2nd VLDB Workshop on Secure Data Management*, 2005.

[11] S. Brands and D. Chaum. Distance-bounding protocols (extended abstract). In *Workshop on the Theory and Application of Cryptographic Techniques (EUROCRYPT'93)*, pages 344–359, May 23–27 1993.

[12] G. Brassard, D. Chaum, and C. Crepeau. Minimum disclosure proofs of knowledge. *Journal of Computer and System Sciences*, 37:156–189, 1988.

[13] Y. Cai and T. Xu. Design, Analysis, and Implementation of a Large-scale Real-time Location-based Information Sharing System. In *ACM MobiSys'08*, pages 106–117, Breckenridge, Colorado, June 2008.

[14] J. Chen, V. Lee, and C. Zhan. Efficient processing of real-time requests with network coding in on-demand broadcast environmemts. In *Proc. of IEEE Int'l Conf. on Embedded and Real Time Computing Systems and Applications (RTCSA'09)*, pages 119–128, Beijing, China, Aug. 24-26 2009.

[15] R. Cheng, Y. Zhang, E. Bertino, and S. Prabhakar. Preserving User Location Privacy in Mobile Data Management Infrastructures. In *the 6th Workshop on Privacy Enhancing Technologies*, pages 393–412, 2006.

[16] J. Chiang, J. Haas, and "Y.-C." Hu. Secure and precise location verification using distance bounding and simultaneous multilateration. In *Proc. of the 2nd ACM Conf. on Wireless Network Security (WiSec'09)*, pages 181–192, Mar. 16–18 2009.

[17] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan. Private information retrieval. In *IEEE Symposium on Foundations of Computer Science*, 1995.

[18] C.-Y. Chow and M. Mokbel. Enabling private continuous queries for revealed user locations. In *Proc. of the 10th int'l conf. on Advances in spatial and temporal databases (SSTD'07)*, pages 258–273, 2007.

[19] C.-Y. Chow, M. Mokbel, and X. Liu. A peer-to-peer spatial cloaking algorithm for anonymous location-based service. In *Proc. of ACM Int'l Symposium on Advances in Geographic Information Systems (GIS'06)*, pages 171–178, Arlington,VA,USA, November 10 - 11 2006.

[20] Y. Chung and M. Kim. Qem: A scheduling method for wireless broadcast data. In *Proc. of IEEE Int'l Conf. on Database Systems for Advanced Applications (DASFAA'99)*, pages 135–142, Hsinchu, Taiwan, April 19-21 1999.

[21] A. Dan, D. Sitaram, and P. Shahabuddin. Dynamic Batching Policies for an On-Demand Video Server. *Multimedia Systems*, 4(3):112–121, June 1996.

[22] H. Dykeman, M. Ammar, and J. Wong. Scheduling algorithms for videotex systems under broadcast delivery. In *Proc. of IEEE Int'l Conf. on Communications (ICC'86)*, pages 1847–1851, Toronto, Canada, June 22-25 1986.

[23] M. Franklin and S. Zdonik. Dissemination-based information systems. *IEEE Data Engineering Bulletin*, 19(3):20–30, 1996.

[24] H. Friis. A note on a simple transmission formula. *Proc. of the IRE*, 34(5):254–256, May 1946.

[25] B. Gedik and L. Liu. A Customizable k-Anonymity Model for Protecting Location Privacy. In *ICDCS'05*, pages 620–629, 2005.

[26] G. Ghinita, M.-L. Damiani, C. Silvestri, and E. Bertino. Preventing velocity-based linkage attacks in location-aware applications. In *Proc. of the 17th ACM SIGSPATIAL Int'l Conf. on Advances in Geographic Information Systems (GIS'09)*, pages 246–255, 2009.

[27] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K.-L. Tan. Private queries in location-based services: Anonymizers are not necessary. In *Proc. ACM int'l conf. Management of data (SIGMOD'08)*, 2008.

[28] G. Ghinita, P. Kalnis, and S. Skiadopoulos. Mobihide: A mobile peer-to-peer system for anonymous location-based queries. In *Proc. of the Int'l Symposium on Advances in Spatial and Temporal Databases (SSTD'07)*, pages 221–238, July 16-18 2007.

[29] G. Ghinita, P. Kalnis, and S. Skiadopoulos. PRIVE: anonymous location-based queries in distributed mobile systems. In *Proc. of the Int'l Conf. on World Wide Web (WWW 2007)*, pages 371–380, May 8-12 2007.

[30] W. G. Griswold, P. Shanahan, S. W. Brown, R. S. Boyer, M. Ratto, B. R. Shapiro, and T. M. Truong. Activecampus: Experiments in community-oriented ubiquitous computing. *IEEE Computer*, 37(10):73–81, 2004.

[31] M. Gruteser and D. Grunwald. Anonymous Usage of Location-based Services through Spatial and Temporal Cloaking. In *ACM MobiSys'03*, pages 31–42, 2003.

[32] Y. Guo, S. Das, and C. Pinotti. A new hybrid scheduling algorithm for asymmetric communication systems: push and pull data based on optimal cut-off point. In *Proc. of ACM Int'l Symp. on Modeling Analysis and Simulation of Wireless and Mobile Systems (MSWiM'01)*, pages 123–130, Rome, Italy, July 2001.

[33] A. Guttman. R-tree: A Dynamic Index Structure for Spatial Search. In *Proc. of ACM SIGMOD*, pages 47–57, Boston, MA, U.S.A, June 1984.

[34] G. Hancke and M. Kuhn. An rfid distance bounding protocol. In *Proc. of the 1st Int'l Conf. on Security and Privacy for Emerging Areas in Communication Networks (SecureComm'05)*, pages 67–73, Sept. 05-09 2005.

[35] T. Hashem, L. Kulik, and R. Zhang. Privacy preserving group nearest neighbor queries. In *Proc. of the 13th Int'l Conf.on Extending Database Technology (EDBT'10)*, pages 489–500, 2010.

[36] H. Hu and J. Xu. Non-exposure location anonymity. In *Proc. of IEEE Int'l Conf. on Data Engineering (ICDE'09)*, pages 1120–1131, Shangai, China, March 29 - April 2 2009.

[37] H. Hu, J. Xu, W. Wong, B. Zheng, D. Lee, and W.C. Lee. Proactive Caching for Spatial Queries in Mobile Environments. In *IEEE Int'l Conf. on Data Engineering (ICDE'05)*, pages 403–414, Tokyo, Japan, April 2005.

[38] T. Imielinski, S. Viswanathan, and B. Badrinath. Data on air: Organization and access. *IEEE Trans. on Knowledge and Data Engineering*, 9(3):353–372, 1997.

[39] T. Imielinski, S. Viswanathan, and B.R. Badrinath. Power Efficiency Filtering of Data on Air. In *Proc. 4th Intl Conf. Extending Database Technology (EDBT 94)*, pages 245–258, Cambridge, U.K, March 28-31, 1994.

[40] A. Inan and Y. Saygin. Location Anonymity in Horizontally Partitioned Spatial-Temporal Data. In *Master Thesis, Sabanci University, Turkey*, 2006.

[41] H. Jagadish. Linear clustering of objects with multiple attributes. In *Proc. of the ACM Int'l Conf. on Management of Data (SIGMOD'98)*, pages 332–342, Atlantic City,NJ, USA, May 23-25 2003.

[42] C. S. Jensen, A. Friis-Christensen, T. B. Pedersen, D. Pfoser, S. Saltenis, and N. Tryfona. Location-Based Services – A Database Perspective. In *Proc. of the Eighth Scandinavian Research Conf. on Geographical Information Science*, pages 59–68, Norway, June 25-27 2001.

[43] D. Johnson and D. Maltz. Dynamic source routing in ad hoc wireless networks. *Mobile Computing*, 353:153–181, 1996.

[44] P. Kalnis, G. Ghinita, K. Mouratidis, and D. Papadias. Preserving anonymity in location based services. Technical Report TRB6/06, Department of Computer Science. National University of Singapore, Singapore, 2006.

[45] P. Kalnis, G. Ghinita, K. Mouratidis, and D. Papadias. Preventing location-based identity inference in anonymous spatial queries. *IEEE Trans. on Knowl. and Data Eng.*, 19:1719–1733, December 2007.

[46] B. Karp and H. T. Kung. GPSR: Greedy Perimeters Stateless Routing for Wireless Network. In *Proc. of the 6th annual international conference on Mobile computing and networking (MOBICOM'00)*, pages 243–254, New York, NY, USA, August 6-11 2000.

[47] S. Kim and S.-H. Kang. Scheduling data broadcast: An efficient cut-off point between periodic and on-demand data. *IEEE Communications Letters*, 14(12):1176–1178, 2010.

[48] Y. Ko and N. H. Vaidya. Location-Aided Routing (LAR) Mobile Ad Hoc Networks. In *Proc. of ACM Int'l Conf. on Mobile Computing and Networking (MOBICOM98)*, pages 66–75, Dallas, TX, U.S.A, October 25-30 1998.

[49] W.S. Ku and R. Zimmermann. Location-based spatial queries with data sharing in mobile environments. In *Proc. of the 22nd Int'l Conf. on Data Engineering, ICDE 2006*, Atlanta, GA, USA, April 3-8 2006.

[50] W.S. Ku, R. Zimmermann, C.W. Wan, and H. Wang. MAPLE: A Mobile Scalable P2P Nearest Neighbor Query Model for Location-based Services. In *Proc. of the 22nd Int'l Conf. on Data Engineering, (ICDE'06)*, pages 182–222, Atlanta, 2006.

[51] D.L. Lee, W.-C. Lee, J. Xu, and B. Zheng. Data Management in Location-Dependent Information Services: Challenges and Issues. *IEEE Pervasive Computing*, 1(3):65–72, 2002.

[52] W. Lee and B. Zheng. Dsi: A fully distributed spatial index for location-based wireless broadcast services. In *Proc. of IEEE Int'l Conf. on Distributed Computing Systems (ICDCS 2005)*, pages 349–358, Columbus, OH, USA, June 6-10 2005.

[53] U. Leonhardt and J. Magee. Security Considerations for a Distributed Location Services. *Journal of Networks and Systems Management*, 6(1):51–70, March 1998.

[54] B. Liu, W.C. Lee, and D.L. Lee. Distributed Caching of Multi-dimensional Data in Mobile Environments. In *The Sixth Int'l Conf. on Mobile Data Management (MDM'05)*, pages 229–233, Ayia Napa, Cyprus, May 9-13 2005.

[55] F. Liu, G. Hamza-Lup, and K. Hua. Using broadcast to protect user privacy in location-based applications. In *Proc. of IEEE Globecom 2010 Worshop on Web and Pervasive Security (WPS 2010)*, Mami,FL, USA, December 6-10 2010.

[56] M. Mokbel, C. Chow, and W. Aref. The new casper: Query processing for location services without compromising privacy. In *Proc. of ACM Int'l Conf. on Very Large Databases (VLDB'06)*, pages 763–774, Seoul, Korea, September 12-15 2006.

[57] K. Mouratidis, S. Bakiras, and D. Papadias. Continuous monitoring of spatial queries in wireless broadcast enviroments. *IEEE Trans. on Mobile Computing*, 8(10):1297–1311, 2009.

[58] Kyriakos Mouratidis and Man Lung Yiu. Anonymous query processing in road networks. *IEEE Transactions on Knowledge and Data Engineering*, 22:2–15, 2010.

[59] G. Pei, M. Gerla, and T.-W. Chen. Fisheye state routing: A routing scheme for ad hoc wireless networks. In *Proc. of the IEEE Conf. on Communications (ICC'00)*, pages 70–74, June 18-22 2000.

[60] C. Perkins and E. Royer. Ad-hoc on-demand distance vector routing. In *Proc. of the IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'99)*, pages 90–100, Feb. 25-26 1999.

[61] P. Persson and P. Fagerberg. Geonotes: A real-use study of a public location-aware community system. In *Technical Report SICS-T-2002/27-SE, SICS, University of Goteborg*, Sweden, 2002.

[62] N. Prabhu and V. Kumar. Periodic scheduling in on-demand broadcast system. In *Proc. of the 3rd Conf. in Information Systems Technology and its Applications (ISTA'04)*, pages 107–121, Salt Lake City,UT, USA, June 15-17 2004.

[63] K. Rasmussen and S. Čapkun. Location privacy of distance bounding protocols. In *Proc. of the ACM Conf. on Computer and Communications Security (CCS'08)*, pages 149–160, Oct. 27-31 2008.

[64] K. Rasmussen and S. Čapkun. Realization of rf distance bounding. In *Proc. of the 19th USENIX conf. on Security (USENIX Security'10)*, pages 25–25, Aug. 11-13 2010.

[65] Q. Ren and M. H. Dunham. Using Semantic Caching to Manage Location Dependent Data in Mobile computing. In *Proc. of the MOBICOM Conf.*, pages 210–221, Boston, MA, USA, April 2000.

[66] N. Sastry, U. Shankar, and D. Wagner. Secure verification of location claims. In *Proc. of the 2nd ACM workshop on Wireless security (WiSe'03)*, pages 1–10, Sept. 19 2003.

[67] M.-T. Shih and C.-M. Liu. Fair broadcasting schedules on dependent data in wireless environments. In *Proc. of IEEE Int'l Conf. on Sensor Networks, Ubiquitous and Trustworthy Computing (SUTC'08)*, pages 185–192, Taichung, Taiwan, June 11-13 2008.

[68] D. Singelee and B. Preneel. Location verification using secure distance bounding protocols. In *Proc. of the IEEE Int'l Conf. on Mobile Adhoc and Sensor Systems (MASS'05)*, pages 834–840, Nov. 7-10 2005.

[69] Z. Song and N. Roussopoulos. K-nearest Neighbor Search for Moving Query Point. In *Proc. of the 7th Int'l Symposium on Advances in Spatial and Temporal Databases (SSTD'01)*, pages 79–96, London, UK, 2001.

[70] T. Xu and Y. Cai. Location cloaking for safety protection of ad hoc networks. In *Proc. of IEEE Int'l Conf. on Computer Communications (INFOCOM'09)*, pages 1944–1952, Rio de Janeiro, Brazil, April 19-25 2009.

[71] Y. Tao, D. Papadias, and Q. Shen. Continuous Nearest Neighbor Search. In *Proc. of Int'l Conf. on Very Large Data Bases (VLDB'02)*, pages 287–298, Hong Kong, China, August 20-23, 2002.

[72] S. Čapkun and J. Hubaux. Secure positioning in wireless networks. *IEEE journal on Selected Areas in Communications*, 2(24):221–232, 2005.

[73] S. Čapkun, K. Rasmussen, M. Kasper, M. Čagalj, and M. Srivastava. Secure location verification with hidden and mobile base stations. *IEEE Transactions on Mobile Computing*, 7:470–483, Apr. 2008.

[74] A. Vora and M. Nesterenko. Secure location verification using radio broadcast. *IEEE Trans. on Dependable and Secure Computing*, 3:377–385, Oct.-Dec. 2006.

[75] J. Wong. Broadcast delivery. *Proc. of IEEE*, 76(12):1566–1577, December 1988.

[76] Y. Wu and G. Cao. Stretch-optimal scheduling for on-demand data broadcasts. In *Proc. of IEEE Int'l Conf. on Computer Communications and Networks (ICCCN'01)*, pages 500–504, October 15-17 2001.

[77] J. Xu, W. Lee, and X. Tang. Exponential index: A parameterized distributed indexing scheme for data on air. In *Proc. of ACM Int'l Conf. on Mobile Systems, Applications And Services (MobiSYS'04)*, pages 153–164, Boston, MA, USA, June 6-9 2004.

[78] J. Xu, B. Zheng, W.-C. Lee, and D.L. Lee. Energy Efficient Index for Querying Location-Dependent Data in Mobile Broadcast Environments. In *Int'l Conf. on Data Engineering (ICDE'03)*, pages 239–250, Bangalore, India, March 5-8, 2003.

[79] T. Xu and Y. Cai. Location Anonymity in Continuous Location-based Services. In *ACM GIS'07*, pages 300–307, November 2007.

[80] T. Xu and Y. Cai. Exploring Historical Location Data for Anonymity Preservation in Location-based Services. In *IEEE Infocom'08*, pages 547–555, Phoenix, AZ, April 2008.

[81] T. Xu and Y. Cai. Feeling-based Location Privacy Protection for Location-based Services. In *ACM Conference on Computer and Communications Security (CCS'09)*, pages 348–357, November 2009.

[82] T. Xu and Y. Cai. Location safety protection in ad hoc networks. *Ad Hoc Networks*, 7(8):1551–1562, 2009.

[83] M.-L. Yiu, C. Jensen, X. Huang, and H. Lu. Spacetwist: Managing the trade-offs among location privacy, query performance, and query accuracy in mobile services. In *Proc. of the IEEE 24th Int'l Conf. on Data Engineering (ICDE'08)*, pages 366–375, Washington, DC, USA, 2008.

[84] J. Zhang, M. Zhu, D. Papadias, Y. Tao, and D.-L. Lee. Location-based Spatial Queries. In *Proc. of the MOBICOM Conf.*, pages 443–454, San Diego, CA, U.S.A, September 14-19, 2003.

[85] B. Zheng and D. L. Lee. Semantic Caching in Location-dependent Query Processing. In *Proc. of the 7th Int'l Symposium on Spatial and Temporal Databases*, pages 97–116, Redondo beach, CA, U.S.A, July 12-15, 2001.

[86] B. Zheng, W.-C. Lee, and D.L. Lee. Search Continuous Nearest Neighbors on the Air. In *IEEE Int'l Conf. on Pervasive Computing and Communications (PerCom'03)*, pages 297–304, Dallas-Fort Worth,TX, U.S.A, March 23-26, 2003.

[87] B. Zheng, W.-C. Lee, and D.L. Lee. Search continuous nearest neighbors on the air. In *the First Int'l Conf. on Mobile and Ubiquitous Systems: Networking and Services (Mobiquitous'04)*, pages 236–245, Boston, MA, U.S.A, August 22-26 2004.

[88] B. Zheng, J. Xu, W. Lee, and D. Lee. Energy-conserving air indexes for nearest neighbor search. In *Proc. of IEEE Int'l Conf. on Extending Database Technology (EDBT04)*, pages 48–66, Heraklion, Crete, Greece, March 14-18 2004.

[89] B. Zheng, J. Xu, W-C. Lee, and D.L. Lee. Grid-Partition Index: A Hybrid Method for Nearest-Neighbor Queries in Wireless Location-Based Services. *Very Large Data Base (VLDBJ)*, 15(1):21–39, 2006.