Hao Wu and Yih-Chun Hu

Location Privacy with Randomness Consistency

Abstract: Location-Based Social Network (LBSN) applications that support geo-location-based posting and queries to provide location-relevant information to mobile users are increasingly popular, but pose a location-privacy risk to posts. We investigated existing LBSNs and location privacy mechanisms, and found a powerful potential attack that can accurately locate users with relatively few queries, even when location data is well secured and location noise is applied. Our technique defeats previously proposed solutions including fake-location detection and query rate limits.

To protect systems from this attack, we propose a simple, scalable, yet effective defense that quantizes the map into squares using hierarchical subdivision, consistently returns the same random result to multiple queries from the same square for posts from the same user, and responds to queries with different distance thresholds in a correlated manner, limiting the information gained by attackers, and ensuring that an attacker can never accurately know the quantized square containing a user. Finally, we verify the performance of our defense and analyze the trade-offs through comprehensive simulation in realistic settings. Surprisingly, our results show that in many environments, privacy level and user accuracy can be tuned using two independent parameters; in the remaining environments, a single parameter adjusts the tradeoff between privacy level and user accuracy. We also thoroughly explore the parameter space to provide guidance for actual deployments.

Keywords: Location based social network, Location privacy, Randomness consistency

DOI 10.1515/popets-2016-0029
Received 2016-02-29; revised 2016-06-02; accepted 2016-06-02.

1 Introduction

Because of the increasing popularity of smartphones worldwide, Location-Based Social Network (LBSN) [31] mobile applications are becoming increasingly popular. Such applications use location services to make mobile applications more location-aware by providing knowledge about the surroundings of a mobile user, e.g., by retrieving posts or finding people around the user. For example, users are able to socialize with people nearby using a rendezvous social network such as PeopleNet [33], online dating applications which connect nearby people [4, 5, 7], friend recommendation networks for shopping together [26], and collaborative networks which gather people who work together [13, 41]. Not only do some regular social networks like Facebook widely use the user's location to tag a user's post, many popular anonymous social network applications are relying on geo-location to help users to filter and retrieve more related posts around users. Examples of such applications include Whisper [8] and Yik Yak [10].

With the increasing number of LBSN users, location privacy in such environments is becoming more important. Abuse and unauthorized use of users' location can threaten users physically [23], financially [39], and even legally [20]. For example, thieves can locate and rob a victim's home through Facebook, resulting in both physical injuries and financial losses [11]. Because an attacker can learn a user's identity through anonymized GPS traces [22, 29], the vulnerabilities are exacerbated if the location of a user is exposed from an anonymous social application, in which the user's identity is supposed to be hidden so that they are able to speak without fear of bullying and abuse [12]. For applications with geo-distance-based retrieval (e.g. Whisper filters posts by the distance to the query), attackers can easily deduce the location of the post owner by *triangulation* [42] or the *space partition attack* [30, 36], and even adding noise and offset in the location data fails to resist attackers [36, 43].

Existing approaches to location privacy can be divided into three categories: first, those that use trusted servers to anonymize the user's location data [27, 28, 32]; second, those that apply cryptographic or private information retrieval (PIR) to location data [21, 34, 35, 37, 38, 45]; third, those that add noise or obfuscation to location data [15, 19, 24, 30, 32, 43]. The first two approaches focus on protecting location data from unsecured data transmission or data storage, and the third sacrifices user experience for reduced attacker

Hao Wu: University of Illinois at Urbana-Champaign, E-mail: haowu11@illinois.edu
Yih-Chun Hu: University of Illinois at Urbana-Champaign, E-mail: yihchun@illinois.edu

precision. In this work, we examine a new powerful entropy-minimization-based attack, which is able to expose victims' location in applications that support geo-location-based information retrieval (e.g. Tinder [7], Yik Yak [10], Whisper [8], etc.), even when these three previous defense categories are performed perfectly (i.e. the location data is secured and the location noise is large enough while providing a reasonable user experience).

In this paper, we consider a geo-location-based query response system which receives a query with the geo-location coordinate of the requestor and returns a set of posts within a specified distance threshold, so that each requestor can retrieve posts around itself. The content of a "post" varies by application, and may include user-generated content, or even the users themselves. We develop an effective entropy-minimization-based attack algorithm that can efficiently deduce the location of a post using only a small number of queries, even when 1) the threshold distance is not adjustable by the requestor (an extreme case of coarse-grained threshold values [43]), 2) noise and offset is added to the location data, and 3) the system is secured against direct access to location information. This attack algorithm relies only on knowledge of the location offset and the probability distribution of the random noise, which can be deduced by queries on posts with pre-known location [36, 43]. In each round, our attack algorithm starts with a probability distribution on a particular victim post. The algorithm then considers all possible query locations and greedily chooses the one that minimizes the expected entropy. (Specifically, for each possible query location, we can compute the resulting probability distribution conditional on whether the system returns the post, and weights those probabilities based on the attacker's current probability distribution, to compute the expected post-query entropy). Our experiments show that to accurately locate the target post in a 10.25 km × 10.25 km map with 30 meter error in around 100 queries (rounds). This outperforms the RANDUDP attack [36], which needs 1400 queries to achieve an error of 37.4 meters against the Gaussian noise of the same variance 0.039 miles), by a factor of 10. Even against stronger noise, our attack can achieve 100-meter error in hundreds to thousands of queries, depending on the variance introduced by random noise. When no location noise is involved, the space partition attack [30, 36], has the same performance as our attack, but the space partition attack is limited to systems that do not add noise.

Although existing work has proposed faked-GPS-coordinate detection and query rate limitation as countermeasures [36, 43], attackers can simply create a few Sybil accounts [44] to improve their total query rate and avoid faked GPS coordinate detection which is mainly based on detecting unrealistic movement patterns in an account [43]. Because only a few queries are needed for our attack, attackers only need tens or hundreds of Sybil accounts, so such accounts can be maintained by attackers to mimic real accounts and avoid Sybil account detection, which usually aims to detect a larger number of such accounts, e.g., thousands of accounts [17].

To mitigate privacy compromise attacks, we propose a simple and effective defense using a quantized map and consistent random response, which can always prevent an attacker from gaining precise knowledge of the quantized square containing the user, even against a strong attacker who is able to make unlimited queries and knows the probability distribution of the noise (but does not know the random values themselves). The system responds to each query based on the center coordinates of the quantized squares containing the post and the query, and provides an identical response to all queries from the same square for posts by the same user. When queries can be made with different distance thresholds, the system responds to those queries in a consistent manner. Against our defense, the attacker can at most learn the quantized square in which the victim is located, and can only make a finite number of different queries in a size-limited map, because the number of quantized squares is finite. The attacker cannot cancel noise by making multiple queries at the same location, so the information available to the attacker is limited.

Moreover, we make this defense scalable over a map of any size by a hierarchical subdivision of this square-quantized map, while limiting the attacker information gain with an upper bound which is customizable by tuning the hierarchical subdivision. This approach trades location accuracy of posts that are far away from the query requestor for scalability; however, experiments indicate that it has minimal impact on application usability. Moreover, users usually don't care whether the post is 99 miles or 101 miles away from them, and existing applications such as Whisper also set more coarse-grained resolution for far away posts than for nearby posts [8].

Surprisingly, the evaluation shows that our defense mechanism allows us to easily tune system parameters to maximize privacy while limiting average user error to an acceptable level. Broadly speaking, noise distributions can be categorized based on whether the user error is bounded in world size (with a constant density of posts). In bounded distributions, user error is bounded regardless of the map scale, so to trade-off between privacy and user error, we need only consider the size of

the quantized squares. In unbounded distributions, privacy is primarily driven by square size, while user error is primarily driven by map scale, so we can maximize privacy for a given distance threshold, while bounding map size to provide acceptable user error and usability; we discuss such tradeoffs in Section 6.

Our main contributions are as follows:

- We analyzed today's LBSN applications and found a potential threat which can expose a user's location quickly and effectively through an entropy-minimization algorithm that bypasses existing defense and detection mechanisms.
- To limit the capability of attackers, we propose a simple and scalable defense based on map quantization, randomness-consistent response, and a novel hierarchical subdivision over a quantized map. In our defense, privacy and accuracy can be easily tuned.
- We comprehensively evaluated our scheme's performance by simulating different levels of noise, scales of map, and sizes of quantized squares. The results support the effectiveness of our defense and provide a thorough guide for parameter tuning.

2 Background and Related Work

2.1 Location-Based Social Network

Location-Based Social Networks (LBSNs) [31] are mobile applications that allow users to retrieve nearby users [2, 4, 7] or posts [8, 10] through geo-location-based queries, which are usually based on the distance from the query requestor to other users or posts. For example, the "Nearby Friends" function in the Facebook mobile application [2] lists each nearby friend and their distance to you; Tinder [7] (a mobile dating application) and Whisper [8] (an anonymous social network application) allow the user to set the distance threshold and retrieve all the results within this threshold.

We classify LBSN vulnerabilities into two categories. (1) *Location data leakage*. Puttaswamy and Zhao [38] show that some LBSN applications run on untrusted servers that store location coordinates in plain text and may leak location data through software bugs or plaintext transmission. (2) *Location information leakage*. As Wang et al. [43], Li et al. [30], Polakis et al. [36], and our paper show, attackers can infer the location of nearby users or posts by exploiting information leaked from query results. Such locations can be easily learned by our attack algorithm, or even simpler ones, e.g. triangulation [42] and the space partition attack [30, 36].

2.2 Information-Leakage Location Attacks

Polakis et al. [36] and Li et al. [30] both propose the space partition attack which can achieve $O(\log(\frac{d}{\epsilon}))$ query count complexity for a noise-free LSBN system (d is the query threshold and ϵ is the distance error of the attack). Polakis et al. [36] also propose an attack, RANDUDP, based on Maximum Likelihood Estimation (MLE) for LSBN with Gaussian noise. In our paper, we propose a more generic entropy-minimization-based attack which works against any kind of noise distribution, and outperforms RANDUDP by approximately 10x against the Gaussian noise of Skout [36].

2.3 Location Privacy Techniques

We have classified existing location-privacy-enhancement techniques in three categories, as in Puttaswamy et al. [37].

Anonymization by trusted servers. Some techniques rely on trusted servers to remove the identities of users and send anonymized location data to service providers, so that the service providers cannot know the owner of the location data. Mokbel et al. [32] and Kalnis et al. [28] both use an anonymizer to remove the user identity before transmitting data to a service provider. Kalnis et al. [28] also achieves K-anonymity that hides the query requestor by mixing with $K-1$ other users around it. However, these anonymization techniques cannot be certain to prevent an attacker from identifying the user, because anonymized GPS traces can be used to infer users' home, office, and even real identities [22, 29]. Even if anonymization techniques can prevent attackers from knowing the identity of leaked location data, they still cannot prevent the location leakage from the query results.

Cryptographic and Private Information Retrieval (PIR). These two techniques are widely used in keeping location data secure in the transmission and storage. Zhong et al. [45], Narayanan et al. [34], and Puttaswamy et al. [37, 38] use cryptographic techniques to ensure that only nearby users can successfully decrypt location data. Again, these techniques focus on securing the location data, but do not protect location privacy from leaking through query results. Although

Puttaswamy et al.'s approach [37] allows only a user's social group to see the user's location information in kNN-query-based LBSN applications, it is not applicable in applications which rely entirely on distance-based queries, such as anonymous applications in which users do not have friends or social groups, e.g. Whisper [8] and Yik Yak [10]. Ghinita et al. [21] and Papadopoulos et al. [35] use Private Information Retreival to allow users to retrieve the nearest neighbor or k-nearest neighbors (kNN) without revealing the location of the query requestor; however, they focus on the location privacy of query requestor but not that of the "post" in this paper.

Low resolution and obfuscation. Gruteser et al. [24], Gedik et al. [19], Mokbel et al. [32], Li [30], and Polakis et al. [36] improve location privacy by limiting the resolution of the location data or replacing the location data with a region based on privacy level, so that neither attackers nor users can acquire location data beyond this maximum resolution. These low resolution approaches equally hurt both users and attackers. Ardagna et al. [15] introduce obfuscation-based techniques that randomly adjusts the user's location to a nearby point. Wang et al. [43] indicated that Whisper [8] is also using random noise and distance offset to increase the location uncertainty. However, Wang et al. [43], Polakis et al. [36] and this paper show that simply adding offset and random noise cannot stop attackers from learning the location of victims. We even demonstrate that limiting the query rate cannot mitigate this potential threat.

Directly storing noisy location data into the database can create error for the attacker (as in Andrés et al. [14]). We consider such strategies to be orthogonal to our defense mechanisms, which operate on the query response. Such approaches permanently introduce a bias in each post, and some posts will experience high location noise (even though the average noise across all posts can be smaller), which hurts the experience of posters who sent those unlucky high-noise posts (because the poster expects that nearby people can see its post).

3 Problem Definition

This paper examines the location privacy issue in LBSN applications that support geo-location-based query, and to design a defense to limit an attacker's ability to learn user locations. In this section, we define the system model and adversary model, formulate the location privacy problem, and summarize our goals.

3.1 System Model

An LBSN application retrieves posts by queries based on geo-location. Although the query response algorithms can vary, we generalize the system model as follows.

Post from users. A post includes the location of the user at the posting time. When a server receives a post, it stores the post with its location so that the server can retrieve the post in response to a query from a nearby location. The content of a "post" varies from application to application, and may be a user, a text post, an image, or any other geographically-bound content.

Geo-location-based query In an LBSN, posts are returned in response to geo-location-based queries. Each query $\mathcal{Q}$ includes the sender's location X_{query}, and a set of parameters d. For example, d can specify the radius of the search. For each post $\mathcal{P}$ with location X_{post}, the server adds random noise to X_{post}, and decides whether or not to return the post based on X_{query} and X_{post}. For example, the server uses a response probability function $\mathsf{P}(X_{query}, X_{post}, d)$ to calculate the probability (between $[0, 1]$) of returning post $\mathcal{P}$ in response to query $\mathcal{Q}$, then generates a random number R from $[0, 1]$ to decide whether to include $\mathcal{P}$ in the response:

- If $R \leq \mathsf{P}(X_{query}, X_{post}, d)$, then include post $\mathcal{P}$ in the response.
- If $R > \mathsf{P}(X_{query}, X_{post}, d)$, then don't include post $\mathcal{P}$ in the response.

In our evaluation, we choose a typical setting. We set a distance threshold d. A querier expects to receive a post if and only if that post lies in a circle $\mathcal{C}_d$, with center X_{query} and radius d. By choosing different $\mathsf{P}(X_{query}, X_{post}, d)$, this generic model can cover all three models (Disk UDP, Round UDP, and Randomized UDP) discussed by Polakis et al. [36].

Secured data transmission and storage. We assume that the system secures both data transmission and storage, so that the only source of location information is through queries. Existing techniques, such as location data anonymization [27, 28, 32] and location data encryption [21, 34, 35, 38, 45], can provide such security.

3.2 Location Privacy Problem

The attacker's goal is to maximize certainty about post location, while the system's goal is to minimize the attacker's certainty subject to acceptable application ac-

curacy. We define two metrics that quantify the attacker and application errors:

Privacy Level: mean absolute error (MAE) for attackers. According to Shokri et al., the MAE (called "correctness" in [40]) is the difference between the attacker's knowledge and the actual location, which measures user privacy. When the attacker learns a probability distribution $Pr_{post}(X)$ for a post at X_{post}, the MAE MAE_{att} is:

$$MAE_{att} = \sum_X |X - X_{post}| \cdot Pr_{post}(X) \quad (1)$$

where $|X_1 - X_2|$ is the geometric distance between X_1 and X_2.

Accuracy Level: per-post absolute error (PAE) for legitimate users. When the system introduces noise or other error sources into its reply, we can categorize responses as true positives (responses that are in the desired area and are returned), false positives (responses that are not in the desired area but are returned), true negatives (responses that are not in the desired area and are not returned), and false negatives (responses that are in the desired area but are not returned). We calculate the per-post absolute error by calculating the error represented by the false positives and false negatives (that is, the distance by which the response is in error) divided by the number of posts that are either returned, or that should have been returned (that is, are in $\mathcal{C}_d$). Then the PAE is defined as:

$$PAE_{user} = \frac{FPE + FNE}{E(|FP|) + |P_A|} \quad (2)$$

where

$$FNE = \sum_{X \in X(\mathcal{P}) \cap \mathcal{C}_d} [(d - |X - X_{query}|) \cdot (1 - \mathsf{P}(X_{query}, X, d))] \quad (3)$$

$$FPE = \sum_{X \in X(\mathcal{P}) \setminus \mathcal{C}_d} [(|X - X_{query}| - d) \cdot \mathsf{P}(X_{query}, X, d)] \quad (4)$$

$$E(|FP|) = \sum_{X \in X(\mathcal{P}) \setminus \mathcal{C}_d} \mathsf{P}(X_{query}, X, d) \quad (5)$$

$$|P_A| = |X(\mathcal{P}) \cap \mathcal{C}_d| \quad (6)$$

and $X(\mathcal{P})$ is the set of coordinates of all posts.

FNE is the total false negative error, which is the distance error represented in the unreturned posts, and *FPE* is false positive error, which is the distance error represented in the incorrectly returned posts. When calculating the *FNE* and *FPE*, we give more weight to posts far from the threshold and less weight to posts near the threshold, because smaller location errors impact user experience less than large location errors. $E(|FP|)$ is the expected number of false positive posts, and $|P_A|$ is the number of posts in the desired area. We normalize the error by the number of posts that the user cares about $E(|FP|) + |P_A|$, which is the number of returned posts plus the number of false negatives, since these posts drive the user experience (as compared to true negatives). Moreover, normalized error is a better measure than absolute error. For example, if in *Case 1*, the user sees 10 posts but 5 of them are wrong, and in *Case 2*, the user sees 100 posts, but only 10 of them are wrong, the user experience in *Case 2* is better, though the absolute error of *Case 1* is larger.

Tradeoff between privacy level and accuracy level. Because high attacker error may cause high user error, we aim to tune the system to maximize the privacy-to-user-error ratio metric MAE_{att}/PAE_{user}, to maximize privacy level per unit loss of accuracy level.

3.3 Adversary Model

Because location authentication is difficult in the Internet, especially when the attacker only makes a small number of queries from each Sybil account, we assume the attacker can choose arbitrary coordinates for each query, and send queries with any X_{query} it wants. Our defense assumes that the attacker can send an unlimited number of queries; however, in our evaluation of previously proposed schemes, we show how an attacker can gain extensive information even with a limited number of queries.

We further assume the attacker knows all system settings, such as the response probability function $\mathsf{P}(X_{query}, X_{post}, d)$. This assumption is reasonable because the attacker can make some posts and use multiple probe queries with attacker-selected coordinates to estimate the noise and distance offset from the responses; such an attack could determine [36, 43] the noise and distance offset added in Skout [6] and Whisper [8]. However, the attacker lacks direct access to the random seed for noise generation and to location data for any post.

3.4 Goals

In this paper, we aim to evaluate the ability of an attacker, and develop a defense mechanism.

Evaluate attacker ability. When evaluating an attacker's ability, we assume that the location data is well-secured and can only be inferred from queries, that noise is added into each response, and the distance threshold d of each query is fixed. Under this system setting, many previously proposed attacks including triangula-

tion and space partition [30] do not work, because they do not expect noise, and because triangulation requires an adjustable threshold (to do binary search to learn the distance from a query to a victim). Even the attack of Wang et al. [43] will be thwarted, because it depends an adjustable threshold. Even under these strong constraints, we show that the attacker can learn nearly perfect information about the location of each post.

Defense mechanism. Because we show that an attacker can learn strong information, we develop a defense that limits the attacker's ability to learn location information. Specifically, we want a mechanism that maximizes privacy level for a given level of accuracy.

4 Attack Algorithm

In this section, we present a powerful attack algorithm (Algorithm 1) based on the adversary model defined in Section 3. The attacker updates its estimated probability distribution for the victim's location based on the query result in each round, and chooses the next query location to maximize the expected information gain. With this algorithm, the entropy of the probability distribution converges to zero.

4.1 Algorithm Construction

Let $\mathcal{M}$ be the region of possible victim locations, and let $|\mathcal{M}|$ be the area of the map. The attacker initially assumes the victim's location falls in a prior probability distribution $Pr_0(X)$, where $X \in \mathcal{M}$. Following the principle of maximum entropy [25], when an attacker knows nothing, it assumes $Pr_0(X)$ is uniformly distributed across $\mathcal{M}$:

$$Pr_0(X) = \frac{1}{|\mathcal{M}|} \tag{7}$$

Update probability distribution. In round n, we compute an updated victim location probability distribution $Pr_n(X)$ based on the previous probability distribution $Pr_{n-1}(X)$. Before the attacker can learn the location of a post $\mathcal{P}$, he must first see post $\mathcal{P}$, so in our model, the first round is the first time the attacker sees post $\mathcal{P}$. Then

$$Pr_1(X) = Pr_1(X|\mathbf{Rx}, \mathbf{X_{Q,1}}) \tag{8}$$

where $\mathbf{Rx}$ denotes that post $\mathcal{P}$ was returned in response to the query made in that round, and $\overline{\mathbf{Rx}}$ denotes that $\mathcal{P}$ was not returned. $\mathbf{X_{Q,n}}$ denotes the event that the nth-round query is made at location $X_{Q,n}$.

Algorithm 1 Attack Algorithm

1: Initialization: Init() (Lines 12–13)
2: **for** n from 1 to N **do**
3: **if** n = 1 **then**
4: $Pr(X)$:= UpdateWithRx$(X_{Q,n}, Pr(X))$ (Lines 14–15)
5: **else**
6: **if** Rx **then**
7: $Pr(X)$:= UpdateWithRx$(X_{Q,n}, Pr(X))$ (Lines 14–15)
8: **else**
9: $Pr(X)$:= UpdateWithoutRx$(X_{Q,n}, Pr(X))$ (Lines 16–17)
10: $X_{Q,n+1}$:= NextQueryLocation$(Pr(X))$ (Lines 18–21)
11: Return $Pr(X)$

12: **Initialization**, Init()
13: Initialize $Pr(X)$ to a flat distribution $Pr_0(X)$

14: **Update distribution when victim post $\mathcal{P}$ is in reply to query at** $X_{Q,n}$, UpdateWithRx$(X_{Q,n}, Pr(X))$
15: $Pr(X) = Pr(X|\mathbf{Rx}, \mathbf{X_{Q,n}})$, (Follow Formula 10)

16: **Update distribution when victim post $\mathcal{P}$ is not in reply to query at** $X_{Q,n}$, UpdateWithoutRx$(X_{Q,n}, Pr(X))$
17: $Pr(X) = Pr(X|\overline{\mathbf{Rx}}, \mathbf{X_{Q,n}})$, (Follow Formula 11)

18: **Pick next query location**, NextQueryLocation$(Pr(X))$
19: **for** each possible X_Q in map $\mathcal{M}$ **do**
20: Use Formula 16 to calculate the expected entropy $E(H(X|\mathbf{X_Q}))$ if our next query is at X_Q
21: Return the X_Q with the smallest $E(H(X|\mathbf{X_Q}))$

After the 1st round ($n \geq 2$), the attacker may or may not receive post $\mathcal{P}$ in its query response. Then

$$Pr_n(X) = \begin{cases} Pr_n(X|\mathbf{Rx}, \mathbf{X_{Q,n}}) & \text{if post received} \\ Pr_n(X|\overline{\mathbf{Rx}}, \mathbf{X_{Q,n}}) & \text{otherwise} \end{cases} \tag{9}$$

To calculate $Pr_n(X|\mathbf{Rx}, \mathbf{X_{Q,n}})$ and $Pr_n(X|\overline{\mathbf{Rx}}, \mathbf{X_{Q,n}})$, we apply Bayes' rule to get

$$Pr_n(X|\mathbf{Rx}, \mathbf{X_{Q,n}}) = \frac{Pr(\mathbf{Rx}|X, \mathbf{X_{Q,n}}) Pr_{n-1}(X)}{Pr(\mathbf{Rx}|\mathbf{X_{Q,n}})} \tag{10}$$

$$Pr_n(X|\overline{\mathbf{Rx}}, \mathbf{X_{Q,n}}) = \frac{Pr(\overline{\mathbf{Rx}}|X, \mathbf{X_{Q,n}}) Pr_{n-1}(X)}{Pr(\overline{\mathbf{Rx}}|\mathbf{X_{Q,n}})} \tag{11}$$

where $Pr(\mathbf{Rx}|X, \mathbf{X_{Q,n}})$ is response probability function $\mathsf{P}(X_{query}, X_{post}, d)$ (chosen by the service) with inputs $X_{Q,n}$ and X. Thus,

$$Pr(\mathbf{Rx}|X, \mathbf{X_{Q,n}}) = \mathsf{P}(X_{Q,n}, X, d) \tag{12}$$

$$Pr(\overline{\mathbf{Rx}}|X, \mathbf{X_{Q,n}}) = 1 - \mathsf{P}(X_{Q,n}, X, d) \tag{13}$$

$Pr(\mathbf{Rx}|\mathbf{X_{Q,n}})$ is the probability that we receive post $\mathcal{P}$ in response to a query at $X_{Q,n}$. This probability is calculated using the attacker's probability distribution from the previous round $Pr_{n-1}(X)$:

$$Pr(\mathbf{Rx}|\mathbf{X_{Q,n}}) = \sum_{X \in \mathcal{M}} Pr(\mathbf{Rx}|X, \mathbf{X_{Q,n}}) Pr_{n-1}(X) \tag{14}$$

$$Pr(\overline{\mathbf{Rx}}|\mathbf{X_{Q,n}}) = 1 - Pr(\mathbf{Rx}|\mathbf{X_{Q,n}}) \quad (15)$$

Choosing the next query location. At the end of the nth round, we choose the next query location $X_{Q,n+1}$ to be the one with the largest expected information gain (or, equivalently, the smallest expected entropy of $Pr_{n+1}(X)$ computed with a weighted average across the two cases where the post is, and is not, received). For each possible $X_{Q,n+1}$, we calculate the expected entropy:

$$\begin{aligned} E(H(X|\mathbf{X_{Q,n+1}})) =& H(X|\mathbf{Rx}, \mathbf{X_{Q,n+1}}) Pr(\mathbf{Rx}|\mathbf{X_{Q,n+1}}) \\ &+ H(X|\overline{\mathbf{Rx}}, \mathbf{X_{Q,n+1}}) Pr(\overline{\mathbf{Rx}}|\mathbf{X_{Q,n+1}}) \end{aligned} \quad (16)$$

where,

$$\begin{aligned} &H(X|\mathbf{Rx}, \mathbf{X_{Q,n+1}}) \\ &= \sum_{X \in \mathcal{M}} -Pr(X|\mathbf{Rx}, \mathbf{X_{Q,n+1}}) \log_2 Pr(X|\mathbf{Rx}, \mathbf{X_{Q,n+1}}) \end{aligned} \quad (17)$$

$$\begin{aligned} &H(X|\overline{\mathbf{Rx}}, \mathbf{X_{Q,n+1}}) \\ &= \sum_{X \in \mathcal{M}} -Pr(X|\overline{\mathbf{Rx}}, \mathbf{X_{Q,n+1}}) \log_2 Pr(X|\overline{\mathbf{Rx}}, \mathbf{X_{Q,n+1}}) \end{aligned} \quad (18)$$

We choose the $X_{Q,n+1}$ that produces the smallest $E(H(X|\mathbf{X_{Q,n+1}}))$ and use it for the next round. We run the algorithm for N rounds until the entropy converges.

4.2 Engineering the Attack Algorithm

Map quantization. To efficiently determine the point of each query, we quantize the map $\mathcal{M}$ into a finite number of squares, and make each query at the center of such a square, so the algorithm can calculate the discrete probability that the victim is in each square. The attacker should choose a square size comparable to the accuracy level desired, because our attack algorithm tends to converge the probability distribution to a single square after several iterations.

Gradually shrinking quantized squares. Because smaller square sizes give the attacker a greater level of accuracy at the cost of greater attacker computational complexity ($O(N^2)$ time to choose the next query position, where N is number of squares), an attacker can start with relatively large square sizes, identify the post's larger square, then refine the search with progressively smaller square sizes.

4.3 Examples and Evaluation

In this section, we present some attack examples over different response probability functions $\mathsf{P}(x, d)$ to show the effectiveness of our attack algorithm, where x is the distance between a post and a query $|X_{query} - X_{post}|$.

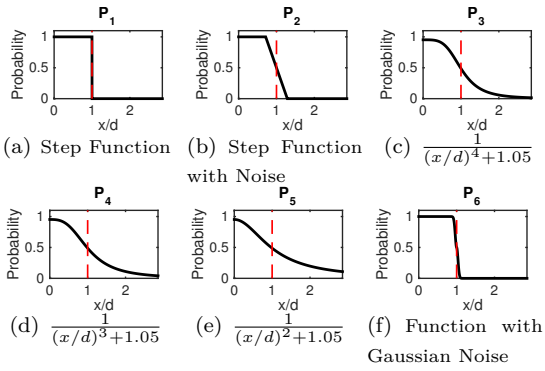


Fig. 1. Sample response probability functions.

Experiment setting. We pick a 10.25 km × 10.25 km area of North Chicago as $\mathcal{M}$ for evaluation, and we quantize the map $\mathcal{M}$ into a grid of 41 × 41 squares. The square size is 0.25 km × 0.25 km, and each square has center x- and y-coordinates from −5 to 5 km. The victim post $\mathcal{P}$ is randomly sampled within a fixed distance threshold $d = 1.75$ km (which is a typical threshold used in Tinder [7] and Yelp [9]), and consider the following 6 response probability functions:

- Step function (Figure 1(a)). $\mathsf{P_1}(x, d) = 1$ when $x \leq d$ and $\mathsf{P_1}(x, d) = 0$ when $x > d$. This function introduces no error, but provides no attacker uncertainty.
- Step function with random noise (Figure 1(b)). We add noise around the distance threshold d, so that $\mathsf{P_2}(x, d) = \mathsf{P_1}(x, d) + Noise(x, d)$. However, the noise is bounded and the attacker can still shrink the potential region of the victim by repeated queries.
- Non-step functions (Figures 1(c)–1(e)). $\mathsf{P_3}(x, d) = \frac{1}{(x/d)^4+1.05}$, $\mathsf{P_4}(x, d) = \frac{1}{(x/d)^3+1.05}$, and $\mathsf{P_5}(x, d) = \frac{1}{(x/d)^2+1.05}$, so no distance has probability of 0 or 1. The attacker can never completely exclude any location from the map. $\mathsf{P_3}(x, d)$ to $\mathsf{P_5}(x, d)$ decrease differently as x/d increases, with $\mathsf{P_3}(x, d)$ being the least uncertain (most like the step function) and $\mathsf{P_5}(x, d)$ being the most uncertain (least like the step function).
- Gaussian-noise functions (Figure 1(f)). $\mathsf{P_6}(x, d) = 0.5 - \mathsf{sgn}(1 - x/d)\frac{\mathsf{G}(0) - \mathsf{G}(x/d-1)}{2\mathsf{G}(0)}$, where $\mathsf{G}(x)$ is a Gaussian function with $\mu = 0, \sigma = 0.036$. Because our threshold $d = 1.75$ km, the variance of the noise is $d \cdot \sigma = 0.039$ miles, which is the same noise variance used in the evaluation of RANDUDP [36].

Thus, we can compare the performance of our attack against RANDUDP.

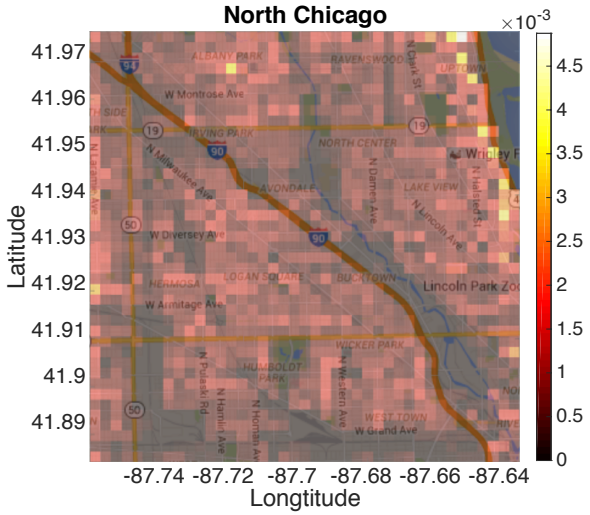


Fig. 2. North Chicago population probability distribution map.

We combine the 2010 block population data of Cook County, IL from the U.S. Census Bureau [3] with the borders of those blocks from the City of Chicago [1] to generate the population probability distribution map (Figure 2) and use this map as the attacker's prior probability to make the evaluation more realistic. We also evaluate the attack algorithm without the prior knowledge of population distribution, to demonstrate a harder case for the attacker.

Furthermore, we demonstrate an example of how to gradually shrink quantized squares (Section 4.2). After the attacker has located the 0.25 km × 0.25 km square (level-1 square) containing the victim, it sub-divides the square into 11 × 11 level-2 squares (each 23 m × 23 m), and locates the victim within its level-2 square.

Experimental results. We run our attack algorithm using these six probability response functions. Figure 3 shows the attacker's probability distribution functions on a victim post $\mathcal{P}$ for various numbers of rounds, which gives us a sense of the power of the attacker algorithm. Figures 17(a)–17(e) in Appendix B show the same attack against other response probability functions.

Figures 4 represent the attacker's certainty, measured using entropy and MAE_{att}, plotted against the number of rounds. Each point represents the average over 100 runs using different victim post positions. We include MAE_{att} values to present the actual error in the attacker's estimate.

No-noise and bounded-random-noise response probability functions provide no privacy to the victim. When the system uses $\mathsf{P}_1(x, d)$, very few queries (in this example, 8) are needed to perfectly locate the victim post. Though the attacker cannot adjust d for triangulation, it can still shrink the potential locations of the victim by a factor of two each round, just as in the space partition attack [30, 36]. Thus, existing defenses based on a coarse-grained d cannot by themselves provide privacy. When the system uses $\mathsf{P}_2(x, d)$, the attacker can also quickly locate the victim with a high degree of accuracy even when bounded uncertainty is added to the step function. In this example, the attacker lowers the MAE_{att} to 0.0211 km within 50 queries. Thus, simply adding bounded random noise to the response also cannot preserve location privacy.

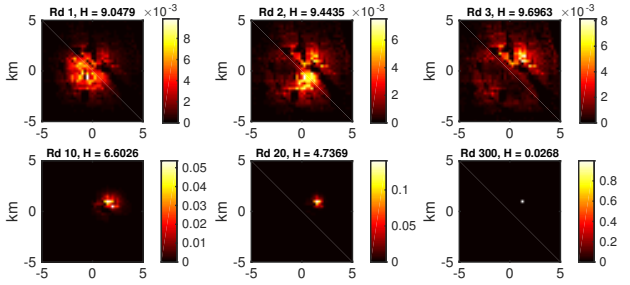


Fig. 3. Probability distribution map vs round number. Case of the response probability function $\mathsf{P}_3(x, d) = \frac{1}{(x/d)^4+1.05}$. The post is at coordinate $(1.25, 0.75)$ km.

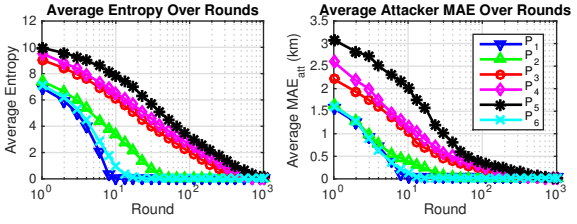


Fig. 4. Ave. entropy and MAE over 100 runs.

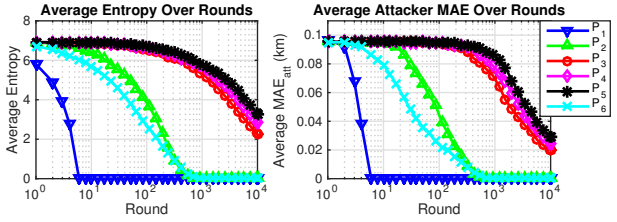


Fig. 5. Ave. entropy and MAE over 100 runs. (Gradual shrink)

Gaussian noise with low variance, such as $\mathsf{P}_6(x, d)$, provides privacy approximately equal to bounded-random noise, and cannot prevent attacker locating the victim.

Against response probability functions with more noise (e.g. $\mathsf{P}_3(x,d)$, $\mathsf{P}_4(x,d)$, and $\mathsf{P}_5(x,d)$), the attacker's performance somewhat slows. The attacker needs hundreds of queries to be confident in its location estimate; for example, with $\mathsf{P}_5(x,d)$ the attacker needs 500 queries to reduce the MAE_{att} to 0.103 km.

Figure 5, shows that gradually shrinking squares can help the attacker improve accuracy with reduced computation overhead. For $\mathsf{P}_1(x,d)$, due to the lack of noise, the attacker can accurately locate the level-2 square in a few rounds. For $\mathsf{P}_2(x,d)$ to $\mathsf{P}_6(x,d)$, because of the noise, to locate the level-2 square, the attacker needs hundreds to thousands of rounds, depending on the noise level. We show only the first 10000 rounds to show the trend.

Figures 4 and 5 show that our attack can locate a victim against $\mathsf{P}_6(x,d)$ to an accuracy of 30 meters using about 100 queries, while the authors of RANDUDP [36] show that RANDUDP needs 1400 queries to achieve error of 37.4 meters against the system with same noise $\mathsf{P}_6(x,d)$, demonstrating the increased power of our attack algorithm.

Evaluations for attackers without prior knowledge of population distribution also show the similar results to those presented above, which suggests that the attack algorithm does not rely significantly on prior knowledge.

Table 1. User Per-post Absolute Error (PAE_{user}).

$\mathsf{P}(x,d)$	P_1	P_2	P_3	P_4	P_5
PAE_{user}**(km)**	**0**	**0.042**	**0.508**	**0.804**	**1.224**

Our results confirm Wang et al. [43]'s observation that increasing uncertainty in the response probability cannot provide strong privacy. Moreover, introducing additional uncertainty in the response probability increases user error PAE_{user}, as shown in Table 1, which is calculated using Formulas 2–6.

Wang et al. [43] proposed to cap the number of daily queries from each user. For example, if an attacker can at most make 100 queries per day, then he can only lower the MAE_{att} to 0.5 km for $\mathsf{P}_5(x,d)$. However, the attacker can use a distributed attack by multiple accounts (i.e. Sybil accounts [44]) to make more queries. Although techniques exist to detect dummy users [17], such techniques are more equipped to handle thousands of such accounts; however, our attack only needs dozens of accounts, which is hard to detect using existing schemes. In the examples above, 10 accounts are sufficient to locate a victim within a 0.25 km × 0.25 km square from a 10.25 km × 10.25 km square, and at most 100 accounts are enough to locate it to hundreds of square meters. Thus, limited query rate cannot guarantee privacy against a reasonable adversary.

5 Defense Approach

In this section, we propose a simple and effective approach that uses consistent randomness to limit attacker effectiveness. We evaluate this approach experimentally in Section 6.

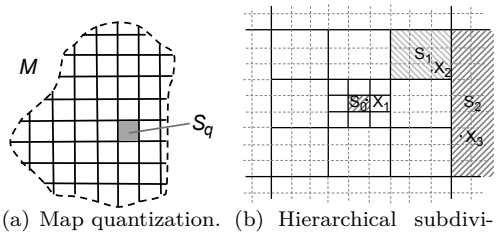
Adversary model. We extend our adversary model from Section 3.3 to admit an attacker that can make unlimited queries. Such an attacker reflects a worst case scenario, so a protocol that can resist such an adversary is also secure against a more limited adversary.

5.1 Approach Description

Our goal is to limit the information that can be gained by an attacker with an unlimited number of queries by limiting the number of queries that give unique information through three techniques: map quantization, consistent response, and hierarchical subdivision.

Map quantization. First, as in [36], we quantize the map $\mathcal{M}$ into squares of equal size, as illustrated in Figure 6(a). Any queries or posts made within a quantized square S_q are assigned a location equal to the center of that square. When the server response to a query $\mathcal{Q}$, it computes the distance between the centers of the squares of the query $\mathcal{Q}$ and post $\mathcal{P}$, instead of the distance between the query and the post. This technique limits the granularity of the attacker's knowledge: the attacker can at most learn the square in which the victim is located; the attacker cannot learn the location within that square. Moreover, this technique also limits the number of distinct queries that the attacker can make; previously, each point in space is a distinct query, but now all queries within a square are identical.

Consistent response. The technique of consistent response works together with the limited number of distinct queries to limit the total information leakage. In consistent response, if two queries (by possibly different nodes) are made within the same square, the same set of posts are returned for those queries. With this technique, an attacker gains information only from the first query in each square. Consistent response is imple-



(a) Map quantization. (b) Hierarchical subdivision (example of $n = 3$).

Fig. 6. Map quantization examples.

mented using a random oracle [16] to return a random response according to the response probability function for the first query at this square, but to return the same result for future queries at this square. If a map $\mathcal{M}$ has $|\mathcal{M}|$ squares, an attacker gets a maximum of $|\mathcal{M}|$ independent responses, no matter how many queries the attacker makes. Moreover, because the response is consistent between queriers, additional attackers do not provide additional information.

Consistent response for multi-post system. In practice, a victim may make multiple posts at the same location and approximately the same time; an attacker may use internal and external information to infer that the same user is responsible for each such post. The attacker can then join information it learns about all posts to better infer victim location. In Section 6, we show that the multi-post can severely compromise location privacy. Such privacy loss in a multi-post environment is common for defenses that use noisy location data (as in Andrés et al. [14]) However, consistent response has the advantage that the system can cluster multiple posts from the same user that are close in time and location, and treat all such posts consistently for each query. We evaluate this approach in Section 6, but leave parameter choices and further development as future work.

Monotonic response for multi-threshold system. The attacker can gain more information from systems with multiple query thresholds than from systems with a single threshold, especially when responses to queries with different thresholds are independent. To limit the information gain, we propose a *monotonic response* that ensures a consistency between queries of different distances. In particular, if a query at threshold a ($< b$) returns $\mathcal{P}$, then a query at threshold b will also return $\mathcal{P}$. Monotonic responses correlate responses for different thresholds, reducing the attacker's potential information gain. A n-threshold system has n response probability functions, $\mathsf{P}^{(\mathsf{i})}(x, d_i)$ ($1 \le i \le n$), for n thresholds (d_i, $1 \le i \le n$). We choose sorted thresholds so $d_i < d_j$ for $i < j$, and then for $1 \le i < j \le n$ and $0 < x$, we require $\mathsf{P}^{(\mathsf{i})}(x, d_i) < \mathsf{P}^{(\mathsf{j})}(x, d_j)$. For an incoming query of d_i, we denote the smallest queried threshold that is larger than d_i as d_b; and denote the largest queried threshold that is smaller than d_i as d_a. Then the system returns a monotonic response as follows:

1. If d_b exists and its response does not include a post $\mathcal{P}$, we do not include $\mathcal{P}$ for d_i either.
2. Otherwise, if d_a exists and its response includes $\mathcal{P}$, we include $\mathcal{P}$ for d_i as well.
3. Otherwise, we determine whether or not to return $\mathcal{P}$ according to the response probability function $\mathsf{P}(x) = \frac{\mathsf{P}^{(\mathsf{i})}(x,d_i) - \mathsf{P}^{(\mathsf{a})}(x,d_a)}{\mathsf{P}^{(\mathsf{b})}(x,d_b) - \mathsf{P}^{(\mathsf{a})}(x,d_a)}$. If d_a does not exist, $\mathsf{P}^{(\mathsf{a})}(x, d_a)$ is replaced with 0; if d_b does not exist, $\mathsf{P}^{(\mathsf{b})}(x, d_b)$ is replaced with 1.

The $\frac{\mathsf{P}^{(\mathsf{i})}(x,d_i) - \mathsf{P}^{(\mathsf{a})}(x,d_a)}{\mathsf{P}^{(\mathsf{b})}(x,d_b) - \mathsf{P}^{(\mathsf{a})}(x,d_a)}$ is $Pr(\mathbf{Rx_i}|x, \overline{\mathbf{Rx_a}}, \mathbf{Rx_b})$, where $\mathbf{Rx_k}$ denotes $\mathcal{P}$ was returned to a query of threshold d_k, and $\overline{\mathbf{Rx_k}}$ denotes that $\mathcal{P}$ was not returned, where k can be i, a, or b.

Hierarchical subdivision. Although map quantization and consistent response can bound the information learned by attackers, the information available to attackers increases as the map area. To reduce the information available in a large map, we introduce hierarchical subdivision, which uses the structure of an n-by-n base map $\mathcal{M}_B$ to hierarchically quantize the map $\mathcal{M}$. We divide our map $\mathcal{M}_B$ into squares S_0 as before, and bundle adjacent groups of $n \times n$ squares into a higher-level square S_1. We repeat the joining of $n \times n$ squares of S_i to get higher-level squares S_{i+1}, and stop when S_m covers $\mathcal{M}$. Figure 6(b) shows an example where the $\mathcal{M}_B$ is a 3-by-3 map.

With hierarchical subdivision, our responses are consistent over potentially larger squares. Specifically, when we receive a query $\mathcal{Q}$ at X_{query} and consider whether to return a post $\mathcal{P}$ at X_{post}, we first determine the hierarchical squares $S_{i,q}$ and $S_{i,p}$ that contain the query and the post respectively. Then we find the highest-level square in which the post and query are in different squares (that is, the maximum i for which $S_{i,p} \neq S_{i,q}$ and $S_{i+1,p} = S_{i+1,q}$). We then use the center of those squares to determine the response probability; that is, we compute the response probability based on the distance between the centers of $X_{S_{i,p}}$ and $X_{S_{i,q}}$.

Figure 6(b) shows a 3-by-3 base map, with a query at X_1 and two posts at X_2 and X_3. Because X_2 and X_1 are not in the same S_1, but in the same S_2, hierarchical subdivision calculates the response probability function using the center of X_1's S_1 and the center of X_2's S_1. Likewise, X_1 and X_3 are not in the same S_2, but in the

same S_3; therefore, the center of X_1's S_2 and the center of X_3's S_2 are used to calculate the response probability.

Theorem 1. *An attacker's accuracy on the whole map $\mathcal{M}$ under hierarchical subdivision is no greater than the attacker's accuracy on the base map $\mathcal{M}_B$ without hierarchical subdivision. Specifically, given an attacker probability distribution on the entire map $Pr(S_{0,p})$, and given an attacker probability on the base map $Pr(S_{0,p})_{B,max}$, $Pr(S_{0,p}) \leq Pr(S_{0,p})_{B,max}$ for all $S_{0,p}$.*

Proof sketch: In $\mathcal{M}$, for any post $\mathcal{P}$ in $S_{0,p}$,

$$Pr(S_{0,p}) = Pr(S_{m,p}) \prod_{0 \leq i \leq m-1} Pr(S_{i,p}|S_{i+1,p}) \tag{19}$$

In hierarchical subdivision, any query made outside $S_{1,p}$ provides no information about sub-squares within $S_{1,p}$, because we respond to such queries using the center point of $S_{1,p}$, regardless of the actual $S_{0,p}$. Thus only queries inside $S_{1,p}$ provide any information on $Pr(S_{0,p}|S_{1,p})$. But S_1 is the base map $\mathcal{M}_B$, so $Pr(S_{0,p}|S_{1,p}) \leq Pr(S_{0,p})_{B,max}$,

$$\begin{aligned} Pr(S_{0,p}) &\leq Pr(S_{m,p}) \cdot Pr(S_{0,p})_{B,max} \prod_{1 \leq i \leq m-1} Pr(S_{i,p}|S_{i+1,p}) \\ &\leq Pr(S_{0,p})_{B,max} \end{aligned} \tag{20}$$

which completes the proof ■

Theorem 1 shows that hierarchical subdivision can expand the world size while retaining an equal level of location privacy. Moreover, hierarchical subdivision makes thresholds larger than the base map size useless to attackers, thus reducing the number of thresholds available to attackers.

Scalability. To implement response consistency, we can either store the consistent query responses for each square in the map $\mathcal{M}$, or use a hash function to compute the response for each post for each query. Hierarchical subdivision allows us to optimize memory usage in the former case, where each response is stored.

To avoid visiting every post for each query, we precompute the result for each S_i at which a distinct query may be made. In particular, our data structure is a quadtree [18], where the root node represents the whole map $\mathcal{M}$, each leaf node represents a distinct S_0 square, and a node at height i represents a distinct middle-level node S_i. Each node's children are the subsquares at the next lower S_i level. Each node in the tree contains a list of posts that should be returned for each query within that square. Thus, when servicing a query for a particular S_0 square, the system returns all posts in the leaf node corresponding to the query location, and in each ancestor of that leaf node.

When we process a post $\mathcal{P}$, we consider each square S_i at which a distinct query may be made, and determine the set of squares at which $\mathcal{P}$ is returned. We then place the key of $\mathcal{P}$ at the node corresponding to each square S_i for which $\mathcal{P}$ is to be returned.

The quadtree takes $O(N+D \cdot P)$ storage, and inserting each post takes $O(D)$ time, where P is the number of posts, N is number of S_0 squares in the map $\mathcal{M}$, and D is the unique number of queries possible for each post; D is $O(s \log_s N)$ (where s is number S_0 squares in the base map $\mathcal{M}_B$). For example, when $s = 25$ ($\mathcal{M}_B$ is a 5×5 base map) and $N = 390625$ (if S_0's area is 0.25 km $\times$ 0.25 km, then the area of the whole map $\mathcal{M}$ is 156.25 km $\times$ 156.25 km, which is much larger than Chicago), D is only 100. Because D is small, our defense is scalable to system with a large map and a large number of posts. The above data structure demonstrates that our scheme is efficiently implemented; further improvements are beyond the scope of this paper.

5.2 Evaluation Metrics

Privacy level metrics. Because we quantized a map $\mathcal{M}$ into squares S_q, attackers can only localize any post $\mathcal{P}$ to a quantized square S_q, and cannot know the precise location of $\mathcal{P}$ in S_q. If an attacker can determine that post $\mathcal{P}$ is in the square with center $\hat{X}$ with probability $Pr_{post}(\hat{X})$, and the post $\mathcal{P}$ is uniformly distributed in a square, then we calculate MAE_{att} for $\mathcal{P}$ as follows:

$$MAE_{att} = \sum_{\hat{X} \in \mathcal{M}} \frac{1}{L^2} \iint\limits_{\substack{x_{min} \leq x \leq x_{max} \\ y_{min} \leq y \leq y_{max}}} |\hat{X} - (x,y)| \cdot Pr_{post}(\hat{X}) \, \mathrm{d}x \, \mathrm{d}y \tag{21}$$

where (x, y) is the coordinate of the victim post $\mathcal{P}$, L is the length of one side of the quantized square S_q, x_{min} and y_{min} are the minimum x and y in the square of post $\mathcal{P}$ respectively, and x_{max} and y_{max} are the maximum x and y in the square of post $\mathcal{P}$ respectively. The derivation is in Appendix A.

Our defense increases MAE_{att} in two ways: (1) map quantization makes the location of users coarse-grained; (2) consistent response limits the information gain for attackers so that they cannot accurately estimate the victim's square. To isolate the contributions of each factor, we define quantized mean absolute error $QMAE_{att}$

to represent the error in an attacker's knowledge of the square of post $\mathcal{P}$:

$$QMAE_{att} = \sum_{\hat{X} \in \mathcal{M}} |\hat{X} - \mathsf{C}(X_{post})| \cdot Pr_{post}(\hat{X}) \qquad (22)$$

where $\mathsf{C}(X)$ is the center coordinate of X's square.

Accuracy level metrics. We also change the calculation of per-post absolute error PAE_{user} for legitimate users. For a post $\mathcal{P}$ at X_{post} and a query at X_{query} in the quantized map $\mathcal{M}$, the service provider decides whether to return $\mathcal{P}$ based on the distance between $\mathsf{C}(X_{post})$ and $\mathsf{C}(X_{query})$ and the probability response function: $\mathsf{P}(\mathsf{C}(X_{query}), \mathsf{C}(X_{post}), d)$. Therefore, we replace $\mathsf{P}(X_{query}, X, d)$ in Formulas 2–6. The derivation is in Appendix A.

6 Defense Evaluation

We experimentally evaluate our defense using the North Chicago map (Figure 2), across different scales, square sizes, and response probability functions.

6.1 Experiment Settings

Response probability functions. In Section 4.3, we showed that the step response probability function $\mathsf{P}_1(x, d)$ and its noise-added variants $\mathsf{P}_2(x, d)$ and $\mathsf{P}_6(x, d)$ cannot defend against our new attack, so in our evaluation we focus on $\mathsf{P}_3(x, d)$, $\mathsf{P}_4(x, d)$, and $\mathsf{P}_5(x, d)$ (Figures 1(c)–1(e)) to evaluate our defense.

Map scenarios. We evaluate our defense with and without hierarchical subdivision. In experiments without hierarchical subdivision, discussed in Section 6.3, the scale of map $\mathcal{M}$ and the size of quantized squares are key factors in defense performance. Therefore, we consider square maps with width ranging from 1.25 km to 10.25 km. We put the origin at the center of the map, so an $n \times n$ map has x- and y-axis values from $[-\frac{n}{2}, \frac{n}{2}]$. For each map, we perform experiments with the following quantized square sizes L: 0.08325, 0.1, 0.125, 0.1665, 0.25, 0.5, and 0.75 km. We fix $d = 0.375$ km as the distance threshold. In experiments with hierarchical subdivision, we use the same settings of square sizes, threshold, and response probability functions, but we fix the map to 10.25 km wide, and vary the base map size from 1.25 km to 10.25 km.

We use real population density as the attacker's prior knowledge to make the evaluation of defense performance more realistic, and we use flat population density to show more generic trends of user error and privacy level as map scale and square size vary.

User per-post absolute error (PAE_{user}). To calculate the PAE_{user}, we assume all posts are uniformly distributed across the map $\mathcal{M}$. We chose a density of 100 posts per square-unit of map area (more posts in a map results in more accurate computations of error due to the law of large numbers), consider a query at $X_{query} = (0, 0)$, and use Formula 26 to calculate PAE_{user} for each map scale and quantized square size. We also calculate PAE_{user} under hierarchical subdivision for a 10.25 km-wide map using various base map scales to evaluate how hierarchical subdivision influences PAE_{user} on a fixed-size map. WLOG, we use threshold $d = 0.375$ km here.

Evaluation for privacy level. We randomly sample 10 victim posts within the smallest test map with size of 1.25 km × 1.25 km, and calculate the average across these 10 posts. For each map scale and quantized square size, we run our attack algorithm (described in Section 4) to locate the victim post $\mathcal{P}$, where queries are returned using our defense without hierarchical subdivision. We also evaluate our defense with hierarchical subdivision, fixing the map size at 10.25 km wide and varying the size of the base map. We also compare with a control that omits consistent response, so the attacker can gain information through multiple queries from the same location. Because of quantization, an attacker can at best learn the square in which the post was made. We calculate the MAE_{att} and $QMAE_{att}$ using Formulas 24–25. We run the experiments and the control tests 100 times and plot the averages.

Evaluation against multiple posts. We place multiple posts of the same victim at $X_{post} = (0.5, 0.5)$ km. The attacker queries at multiple locations for posts within a fixed threshold, and the system returns each relevant post according to the probability response function. Our evaluation uses the 5.25 km × 5.25 km North Chicago map and its population density as the attacker's prior knowledge.

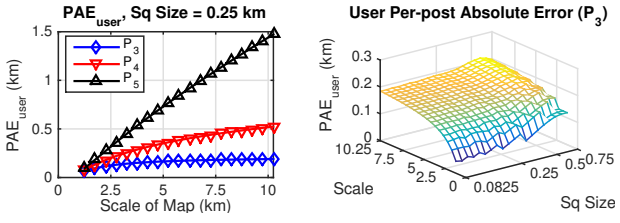
Evaluation against multiple thresholds. We evaluate both non-monotonic- and monotonic-response defense systems in the 5.25 km wide North Chicago map, and show their performance under different thresholds from 0.375 to 2.25 km. We choose different thresholds because hierarchical subdivision limits the number of thresholds available to the attacker. We still place the victim at $X_{post} = (0.5, 0.5)$ km.

6.2 Experiment Results

Figures 7(a)–7(d) show the user per-post absolute error (PAE_{user}) in our defense without hierarchical subdivision. To show generic trends, we assume a flat population density. As Figures 7(b)–7(d) show, for a given response probability function $\mathsf{P}(x,d)$ PAE_{user} is mainly determined by the map scale, but not the square size L. For a fixed map scale, PAE_{user} is mostly constant as square size varies. When the square size starts to reach or exceed the threshold d, PAE_{user} increases, and edge effects cause PAE_{user} to fluctuate. Thus, the threshold loosely bounds the square size. Figure 7(a) shows PAE_{user} when $L = 0.25$ km. This figure suggests that the response probability function $\mathsf{P}(x,d)$ is also a key factor in PAE_{user}. In our experiments, each of our probability response functions decreases as $O(\frac{1}{(x/d)^k})$; when such functions have a long tail (such as $k = 2$), PAE_{user} is unbounded with increasing map scale. Formulas 26–30 imply that FNE and $|P_A|$ are constant in k, and $E(|FP|)$ and FPE are bounded (with respect to map scale) if $k > 2$ and $k > 3$ respectively. Therefore, when $k > 3$, PAE_{user} converges; when $k \in (2, 3]$, PAE_{user} is sublinear in map scale; when $k \leq 2$, PAE_{user} increases linearly in map scale. Moreover, when PAE_{user} is bounded, it is primarily influenced by map size, and secondarily (and to a much smaller extent) by square size. At larger map sizes, as PAE_{user} approaches its bound, increasing map size has little impact, and square size becomes the only factor. Although PAE_{user} may vary when $X_{query} \neq (0,0)$, our results still show the trend of PAE_{user} across different parameter choices.

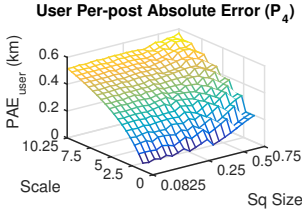
Figure 8 shows PAE_{user} for a 10.25 km-wide square map in our defense with hierarchical subdivision that for varying sizes of base map. For any given probability response function, PAE_{user} is approximately equal to PAE_{user} of a 10.25 km-wide square map regardl of base map size (c.f. Figure 7(a)). Due to space cc straints, we show only results for base maps widths 0.75 km, 1.25 km and 3.25 km; the results for other siz base maps are similar. Our results demonstrate that any given response probability function, PAE_{user} is p marily determined by the scale of the whole map $\mathcal{M}$, regardless of the use of hierarchical subdivision. Thus, further experiments examine PAE_{user} without hierarchical subdivision.

Figures 9(a) and 9(b) present attacker entropy and quantized MAE ($QMAE_{att}$) in maps with different scales, using a fixed square size $L = 0.25$ km. The attacker has prior knowledge of the real population density. We show only $QMAE_{att}$ as a function of round number for response probability function $\mathsf{P}_5(x,d)$ here, though the results are similar for $\mathsf{P}_3(x,d)$ and $\mathsf{P}_4(x,d)$ (Figures 16(a) and 16(b) in Appendix B). These results show that our defense provides significant privacy improvement. Figure 9(b) shows that without our defense, the attacker can accurately learn the quantized square containing the victim post with nearly zero entropy and $QMAE_{att}$, whereas with our defense, both the entropy and $QMAE_{att}$ are always much larger than zero in the experiments with our defense, which means that with our defense, the attacker can never accurately locate the square containing the victim.

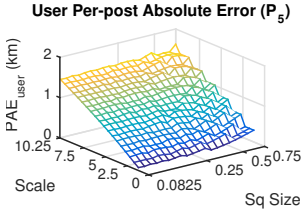


(a) PAE_{user} (0.25 km sq size).

(b) PAE_{user} of $\mathsf{P}_3(x,d)$.



(c) PAE_{user} of $\mathsf{P}_4(x,d)$.



(d) PAE_{user} of $\mathsf{P}_5(x,d)$.

Fig. 7. Per-post user absolute error (PAE_{user}) in maps with various scales and quantized square sizes. The "scale" and "sq size" denote the length of the side of map and square respectively (the unit is km). These experiments do not include hierarchical subdivision.

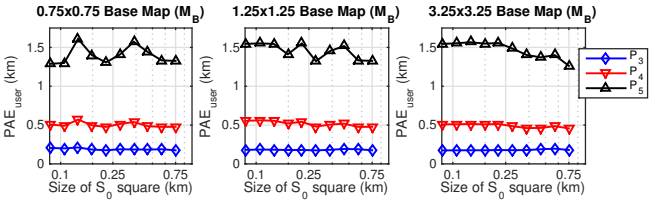
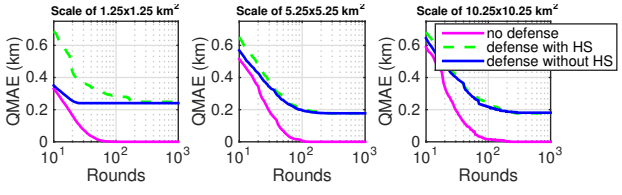


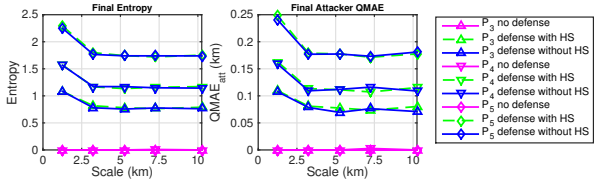
Fig. 8. Per-post user absolute error under our defense with hierarchical subdivision. The map size is 10.25 km × 10.25 km, and we vary the size of the base map and quantized squares.

Figures 9(a) shows how the $QMAE_{att}$ decreases across rounds. Without our defense, the attacker can decrease the entropy to zero in at most a few hundreds of rounds (queries). In contrast, against our defense, the

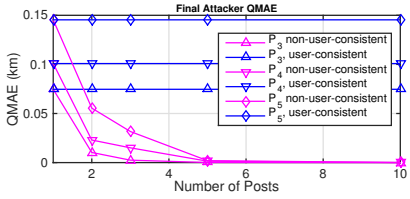
attacker cannot gain more information to lower the entropy after a few rounds (queries). In this experiment, we show how soon the attacker can locate the square of the victim if our defense is not applied, but in practice, the attacker can easily locate victim's location more accurately as described in Section 4.2. Furthermore, these figures show that the entropy and $QMAE_{att}$ of experiments with hierarchical subdivision are similar to those without hierarchical subdivision, as Theorem 1 predicts. The results also show that smaller base maps result in increased privacy under hierarchical subdivision, with the best improvements shown when the scale of the base map is smaller than the threshold, as shown by the Figures 18(a)– 18(b) in Appendix B. Because of Theorem 1, we consider only the defense without hierarchical subdivision.



(a) Case of $\mathsf{P}_5(x,d) = \frac{1}{(x/d)^2+1.05}$.



(b) Average final entropy and QMAE (single-post case).

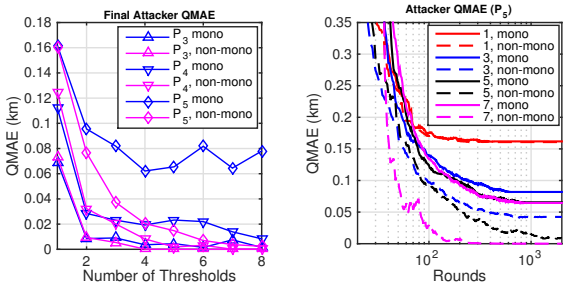


(c) Average final QMAE (multi-post case).

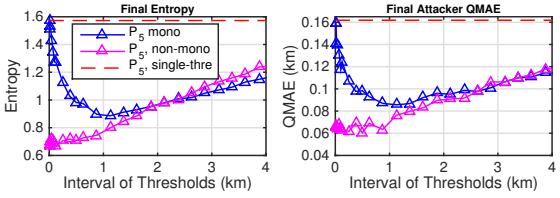
Fig. 9. The average attacker entropy and QMAE ($QMAE_{att}$) over North Chicago maps with different scales, using a fixed square size $L = 0.25$ km. We choose a threshold $d = 0.375$ km, and average the $QMAE_{att}$ across 10 victims randomly sampled from the map with scale of 1.25 km × 1.25 km. In the multi-post case, we set the victim at $X_{post} = (0.5, 0.5)$ km. HS denotes runs on a 10.25 km-wide square map with Hierarchical Subdivision; the map scale reflects the size of the base map. Each point represents the average of 100 runs.

Figure 9(c) shows final $QMAE_{att}$ for increasing numbers of same-location posts from one user. These results show that an attacker can quickly reduce its error when the system treats each such post independently; at 5 posts, the user has no location privacy. Consistently responding to each query provides identical privacy to the one-post scenario.

Figure 10(a) and 10(b) show the $QMAE_{att}$ against a system with multiple thresholds. Without monotonic response, the final $QMAE_{att}$ drops quickly in systems with more thresholds. However, with monotonic response, the attacker's improvement in error is not as dramatic, and almost no marginal improvement is found for the fourth and subsequent thresholds. Monotonic defense has a greater effect with noisier response probability functions.



(a) Ave. final $QMAE_{att}$ (b) $QMAE_{att}$ of P_5.

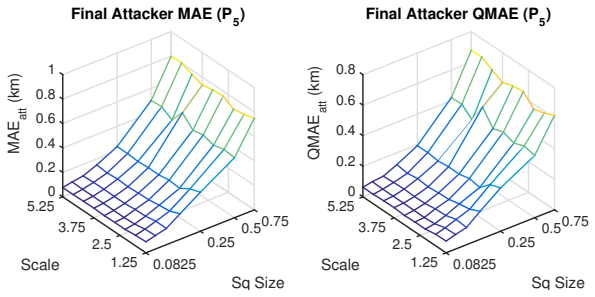


(c) Ave. final $QMAE_{att}$ and entropy in two-threshold sys.

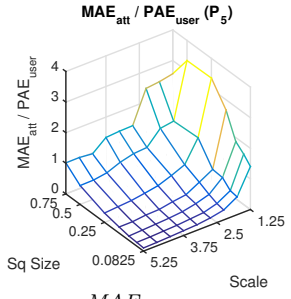
Fig. 10. Average attacker QMAE and entropy (over 1000 runs) against monotonic-response and non-monotonic-response system with multiple thresholds.

Figure 10(c) further shows how the interval between thresholds affects attacker information gain and final accuracy. The system has thresholds d_1 and d_2, and the x-axis shows $d_2 - d_1$, where d_1 is fixed at 0.375 km. In the monotonic-response system, the attacker $QMAE_{att}$ and entropy is high when two thresholds are close, because two close thresholds have high response correlation. On the contrary, in the non-monotonic-response system, closer thresholds reduce attacker error, because each independent query gains more information, but farther thresholds reduce the information available from the second query (intuitively, if $d_2 = \infty$ and returns all posts, then that second query provides no addi-

tional information even when drawn independently). At greater $d_2 - d_1$, the correlation between the two queries in the monotonic-response system gets weak, causing the monotonic-response curve to converge to the non-monotonic-response curve, and both increase in $d_2 - d_1$. The reduced correlation also explains why additional thresholds in the range $[0.375, 2.25]$ km from the monotonic-response system of Figure 10(a) does not help attackers to get lower $QMAE_{att}$, because more thresholds in a fixed range (limited by hierarchical subdivision) results in higher correlation between them.



(a) MAE_{att} of P_5. (b) $QMAE_{att}$ of P_5.



(c) $\frac{MAE_{att}}{PAE_{user}}$ of P_5.

Fig. 11. The average attacker MAE (MAE_{att}), quantized MAE ($QMAE_{att}$), and privacy-to-user-error ratio (MAE_{att} / PAE_{user}) over varying map scale (no hierarchical subdivision) and quantized square size. Each point represents the average of 100 runs with flat population density.

To explore the privacy level attained under different parameterizations of our defense, we performed extensive experiments under flat population density, and plot the results of $\mathsf{P}_5(x, d)$ in Figures 11(a)–11(c). The results of other response probability functions show similar trends. Figures 11(a) and 11(b) show MAE_{att} and $QMAE_{att}$ in maps of different scales and quantized square size (labeled as "Sq Size" in the figures). $QMAE_{att}$ has a trend similar to that of MAE_{att}, showing that the attacker error is mostly attributable to our defense, rather than the map quantization. Moreover, MAE_{att} and $QMAE_{att}$ are mainly determined by two factors: (1) the square size and (2) the uncertainty of the $\mathsf{P}(x, d)$. Attacker error increases with larger square sizes and with more uncertain probability response functions. On the other hand, the scale of map does not significantly influence attacker error. However, if the scale of the map is too large, and hierarchical subdivision is not used, privacy level will also be low, because Figures 9(b) and 11(a)–11(b) show that privacy level decreases with increasing map size. Thus, we need hierarchical subdivision to construct a map with large scale from a small base map while keeping the privacy level the same as that in the small base map. Figure 11(c) shows our privacy-to-user-error ratio (MAE_{att}/PAE_{user}). A large ratio means the system can achieve more privacy level (MAE_{att}) per unit of lost accuracy. This figure shows that as the quantized square size increases, and as the map scale decreases, the ratio increases.

6.3 Tuning Our Defense

Response probability function. A less certain response probability function increases privacy level, but at the cost of user error. Applications requiring a relatively large map should use a probability response function that decreases as $O(\frac{1}{(x/d)^k})$ for $k > 3$, so that user error will be bounded regardless of map size.

Square size and map scale. Section 6.2 shows that for a given response probability function where PAE_{user} is not bounded, privacy level (MAE_{att}) is most strongly correlated with the quantized square size, whereas the user error (PAE_{user}) is most strongly correlated with map scale; the relative independence of map scale and square size allows the system designer to maximize the privacy level for a given threshold d (because the threshold loosely bounds the choice of usable square sizes), while choosing a map scale that keeps an acceptable user error and user experience (because increasing a user's map scale potentially increases the usefulness of the system, but also increases the user's average error). For a given response probability function where PAE_{user} is bounded, user error is only slightly affected by square size, while privacy is strongly affected by square size; in this case, the system designer must consider the tradeoff between MAE_{att} and PAE_{user} to choose a proper square size. The threshold d of a real system may be adjustable by user, so square size should be smaller than the granularity of the threshold d. For example, Whisper [8] uses 1 km, 5 km, 15 km, and so forth as choices for threshold,

so Whisper would likely not want to use squares larger than 1 km^2.

Hierarchical subdivision. Both our theorem and experimental results show that hierarchical subdivision makes our defense scalable in that it extends the privacy protection of a small base map to an arbitrarily sized map while having minimal impact on user experience. Nonetheless, the map size impacts user accuracy, and an excessively large base map size could result in a low privacy level.

Multiple thresholds. If the LSBN system requires multiple thresholds, we recommend the designer to have fewer thresholds that are smaller than the scale of the base map. For a typical multi-threshold setting of 0.3, 1, 5, and 20 miles (from Yelp [9]), we can set the base map to 2 miles wide so that only the first two thresholds are useful for attackers. For thresholds less than base map scale, privacy is improved with closer thresholds.

Periodically updating responses. Consistent response limits the information gained by attackers, but it introduces a usability problem for users that do not move between such squares. If we never update our consistent response, then a user that does not move may miss some nearby posts. We therefore allow a system to periodically update its responses. Naturally there is a tradeoff between such a period and the privacy level. Because attackers gain new information each time the random oracle updates the response, the shorter the period, the more quickly attackers can locate users. Also, the lifetime of a post affects the number of times an attacker's query can be used to locate it, so systems with short-lived posts can have shorter periods than systems with long-lived posts. For example, Yik Yak [10] appears to return posts for only 12 hours, and can update their responses every 4 hours with minimal privacy loss, since each location will only gain information on the location of the post 3 times before post expiration.

Population density: sparse vs. dense. To keep an equivalent privacy level, we should choose a larger square and threshold in a sparse-population map, and use small square size and threshold in a denser map. For example, the population density is 500/km^2 in Chicago suburbs, but 50,000/km^2 in Chicago downtown. Then, 0.1×0.1 km^2 squares in Chicago downtown can guarantee averagely 500 people in a square, and achieve "500-anonymity" However, we need larger squares of 1×1 km^2 to keep the same "500-anonymity" privacy level in Chicago suburbs.

7 Conclusion

LBSN applications face strong threats to location privacy. We investigated current LBSNs and found a powerful attack which can quickly locate a victim using relatively few queries, defeating proposed defense and detection mechanisms. To limit the ability of such attackers, we proposed a novel randomness-consistent defense that quantizes the map into squares and consistently responds to queries from the same quantized square. Our defense prevents an attacker from accurately learning users' location, and can be scaled to a large map using hierarchical subdivision without loss of privacy level or significant detriment to user experience.

The results of our comprehensive simulations show the effectiveness of our strategy, and suggests that in our system, privacy and accuracy can be easily tuned. We can bound the user error (PAE_{user}) when using a response probability function without a long tail, independent of map size, and can then tune the quantized square size to manage the tradeoff between privacy level (MAE_{att}) and user error (PAE_{user}) for a given threshold d (because the threshold loosely bounds the choice of usable square sizes). We found that the privacy level is primarily influenced by quantized square size, while the user error is not. Although the user error is proportional to the scale of the map when using a response probability function with a long tail, the privacy level and the user error can be almost independently tuned by the quantized square size and the scale of the map respectively, we can maximize privacy for any given distance threshold, and choose a map scale that provides the desired user error. Though randomness consistency prevents a user from seeing some posts if that user stays in the same square, we believe this bias is acceptable in applications that do not require users to see all nearby posts. Moreover, the service provider can update its consistent response periodically to trade privacy for usability.

8 Acknowledgments

We thank the reviewers for their input. This research was supported in part by NSF under grant CNS-0953600. We also thank Hui Shi for helping us find and process the data of the borders of census blocks from the City of Chicago [1].

References

[1] Boundaries - Census Blocks (Chicago, IL). http://www.cityofchicago.org/city/en/depts/doit/dataset/boundaries_-_censusblocks.html.

[2] Facebook. Online socail mobile application. https://itunes.apple.com/us/app/facebook/id284882215?mt=8.

[3] Fact Finder. http://factfinder.census.gov/faces/nav/jsf/pages/index.xhtml.

[4] Hinge. Online dating mobile application. http://hinge.co.

[5] Match. Online dating mobile application. http://www.match.com.

[6] Skout. http://www.skout.com.

[7] Tinder. Online dating mobile application. https://www.gotinder.com.

[8] Whisper. Anonymous mobile social network. https://whisper.sh.

[9] Yelp. http://www.yelp.com.

[10] Yik Yak. Anonymous mobile social network. http://www.yikyakapp.com.

[11] Police: Thieves robbed homes based on Facebook, social media sites. http://www.wmur.com/Police-Thieves-Robbed-Homes-Based-On-Facebook-Social-Media-Sites/11861116, 2010.

[12] New social App has juicy posts, all anonymous. http://www.nytimes.com/2014/03/19/technology/new-social-app-has-juicy-posts-but-no-names.html?_r=0, 2014.

[13] G. Ananthanarayanan, V. N. Padmanabhan, L. Ravindranath, and C. A. Thekkath. Combine: leveraging the power of wireless peers through collaborative downloading. In *Proceedings of the 5th international conference on Mobile systems, applications and services*, pages 286–298. ACM, 2007.

[14] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi. Geo-indistinguishability: Differential privacy for location-based systems. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pages 901–914. ACM, 2013.

[15] C. A. Ardagna, M. Cremonini, E. Damiani, S. D. C. Di Vimercati, and P. Samarati. Location privacy protection through obfuscation-based techniques. In *Data and Applications Security XXI*, pages 47–60. Springer, 2007.

[16] M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM conference on Computer and communications security*, pages 62–73. ACM, 1993.

[17] G. Danezis and P. Mittal. Sybilinfer: Detecting sybil nodes using social networks. In *NDSS*. San Diego, CA, 2009.

[18] R. A. Finkel and J. L. Bentley. Quad trees a data structure for retrieval on composite keys. *Acta informatica*, 4(1):1–9, 1974.

[19] B. Gedik and L. Liu. Location privacy in mobile systems: A personalized anonymization model. In *Distributed Computing Systems, 2005. ICDCS 2005. Proceedings. 25th IEEE International Conference on*, pages 620–629. IEEE, 2005.

[20] A. Gendar and A. Lisberg. How cell phone helped cops nail key murder suspect. Secret 'pings' that gave bouncer away. http://www.nydailynews.com/archives/news/cell-phone-helped-cops-nail-key-murder-suspect-secret-pings-gave-bouncer-article-1.599672, 2006.

[21] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K.-L. Tan. Private queries in location based services: anonymizers are not necessary. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 121–132. ACM, 2008.

[22] P. Golle and K. Partridge. On the anonymity of home/work location pairs. In *Pervasive computing*, pages 390–397. Springer, 2009.

[23] F. Grace. Stalker victims should check for GPS. http://www.cbsnews.com/news/stalker-victims-should-check-for-gps/, 2003.

[24] M. Gruteser and D. Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *Proceedings of the 1st international conference on Mobile systems, applications and services*, pages 31–42. ACM, 2003.

[25] P. Harremoës and F. Topsøe. Maximum entropy fundamentals. *Entropy*, 3(3):191–226, 2001.

[26] M. Hendrickson. The state of location-based social networking on the iPhone. http://techcrunch.com/2008/09/28/the-state-of-location-based-social-networking-on-the-iphone, 2008.

[27] T. Jiang, H. J. Wang, and Y.-C. Hu. Preserving location privacy in wireless lans. In *Proceedings of the 5th international conference on Mobile systems, applications and services*, pages 246–257. ACM, 2007.

[28] P. Kalnis, G. Ghinita, K. Mouratidis, and D. Papadias. Preventing location-based identity inference in anonymous spatial queries. *Knowledge and Data Engineering, IEEE Transactions on*, 19(12):1719–1733, 2007.

[29] J. Krumm. Inference attacks on location tracks. In *Pervasive Computing*, pages 127–143. Springer, 2007.

[30] M. Li, H. Zhu, Z. Gao, S. Chen, L. Yu, S. Hu, and K. Ren. All your location are belong to us: Breaking mobile social networks for automated user location tracking. In *Proceedings of the 15th ACM international symposium on Mobile ad hoc networking and computing*, pages 43–52. ACM, 2014.

[31] N. Li and G. Chen. Analysis of a location-based social network. In *Computational Science and Engineering, 2009. CSE'09. International Conference on*, volume 4, pages 263–270. Ieee, 2009.

[32] M. F. Mokbel, C.-Y. Chow, and W. G. Aref. The new casper: A privacy-aware location-based database server. In *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*, pages 1499–1500. IEEE, 2007.

[33] M. Motani, V. Srinivasan, and P. S. Nuggehalli. Peoplenet: engineering a wireless virtual social network. In *Proceedings of the 11th annual international conference on Mobile computing and networking*, pages 243–257. ACM, 2005.

[34] A. Narayanan, N. Thiagarajan, M. Lakhani, M. Hamburg, and D. Boneh. Location privacy via private proximity testing. In *NDSS*. Citeseer, 2011.

[35] S. Papadopoulos, S. Bakiras, and D. Papadias. Nearest neighbor search with strong location privacy. *Proceedings of the VLDB Endowment*, 3(1-2):619–629, 2010.

[36] I. Polakis, G. Argyros, T. Petsios, S. Sivakorn, and A. D. Keromytis. Where's wally?: Precise user discovery attacks in location proximity services. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications*

Security, pages 817–828. ACM, 2015.

[37] K. P. Puttaswamy, S. Wang, T. Steinbauer, D. Agrawal, A. El Abbadi, C. Kruegel, and B. Y. Zhao. Preserving location privacy in geosocial applications. *Mobile Computing, IEEE Transactions on*, 13(1):159–173, 2014.

[38] K. P. Puttaswamy and B. Y. Zhao. Preserving privacy in location-based mobile social applications. In *Proceedings of the Eleventh Workshop on Mobile Computing Systems & Applications*, pages 1–6. ACM, 2010.

[39] B. Schilit, J. Hong, and M. Gruteser. Wireless location privacy protection. *Computer*, 36(12):135–137, 2003.

[40] R. Shokri, G. Theodorakopoulos, J.-Y. Le Boudec, and J.-P. Hubaux. Quantifying location privacy. In *Security and privacy (sp), 2011 ieee symposium on*, pages 247–262. IEEE, 2011.

[41] M. Siegler. Foodspotting is a location-based game that will make your mouth water. http://techcrunch.com/2010/03/04/foodspotting.

[42] M. Veytsman. How I was able to track the location of any Tinder user. http://blog.includesecurity.com/2014/02/how-i-was-able-to-track-location-of-any.html.

[43] G. Wang, B. Wang, T. Wang, A. Nika, H. Zheng, and B. Y. Zhao. Whispers in the dark: Analysis of an anonymous social network. In *IMC '14 Proceedings of the 2014 Conference on Internet Measurement Conference*, pages 137–150, New York, USA, 2014. ACM.

[44] Z. Yang, C. Wilson, X. Wang, T. Gao, B. Y. Zhao, and Y. Dai. Uncovering social network sybils in the wild. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 8(1):2, 2014.

[45] G. Zhong, I. Goldberg, and U. Hengartner. Louis, lester and pierre: Three protocols for location privacy. In *Privacy Enhancing Technologies*, pages 62–76. Springer, 2007.

A Defense Evaluation Metrics

Privacy level metrics. Because we quantized a map $\mathcal{M}$ into squares S_q, attackers can only localize any post $\mathcal{P}$ to a quantized square S_q, and cannot know the precise location of $\mathcal{P}$ in S_q. If an attacker can determine that post $\mathcal{P}$ is in the square with center $\hat{X}$ with probability $Pr_{post}(\hat{X})$, then we modify the calculation for mean absolute error (Formula 1) as follows:

$$MAE_{att} = \sum_{\hat{X} \in \mathcal{M}} |\hat{X} - X_{post}| \cdot Pr_{post}(\hat{X}) \tag{23}$$

where X_{post} is the coordinate of the victim post $\mathcal{P}$.

If post $\mathcal{P}$ is uniformly distributed in a square, then the average MAE_{att} for $\mathcal{P}$ is:

$$MAE_{att} = \sum_{\hat{X} \in \mathcal{M}} \frac{1}{L^2} \iint\limits_{\substack{x_{min} \le x \le x_{max} \\ y_{min} \le y \le y_{max}}} |\hat{X} - (x, y)| \cdot Pr_{post}(\hat{X}) \, \mathrm{d}x \, \mathrm{d}y \tag{24}$$

where the L is the length of one side of the quantized square S_q, x_{min} (y_{min}) is the minimum x (y) in the square of post $\mathcal{P}$, and x_{max} (y_{max}) is the maximum x (y) in the square of post $\mathcal{P}$.

Our defense increases MAE_{att} in two ways: (1) map quantization makes the location of users coarse-grained; (2) consistent response limits the information gain for attackers so that they cannot accurately estimate the victim's square. To isolate the contributions of each factor, we define quantized mean absolute error $QMAE_{att}$ to represent the error in an attacker's knowledge of the square of post $\mathcal{P}$:

$$QMAE_{att} = \sum_{\hat{X} \in \mathcal{M}} |\hat{X} - \mathsf{C}(X_{post})| \cdot Pr_{post}(\hat{X}) \tag{25}$$

where $\mathsf{C}(X)$ is the center coordinate of X's square.

Accuracy level metrics. We also change the calculation of per-post absolute error PAE_{user} for legitimate users. For a post $\mathcal{P}$ at X_{post} and a query at X_{query} in the quantized map $\mathcal{M}$, the service provider decides whether to return $\mathcal{P}$ based on the distance between $\mathsf{C}(X_{post})$ and $\mathsf{C}(X_{query})$ and the probability response function: $\mathsf{P}(\mathsf{C}(X_{query}), \mathsf{C}(X_{post}), d)$. We use Formulas 2–6 to calculate PAE_{user} as follows:

$$PAE_{user} = \frac{FPE + FNE}{E(|FP|) + |P_A|} \tag{26}$$

where,

$$FNE = \sum_{X \in X(\mathcal{P}) \cap \mathcal{C}_d} [(d - |X - X_{query}|) \cdot (1 - \mathsf{P}(\mathsf{C}(X_{query}), \mathsf{C}(X), d))] \tag{27}$$

$$FPE = \sum_{X \in X(\mathcal{P}) \setminus \mathcal{C}_d} [(|X - X_{query}| - d) \cdot \mathsf{P}(\mathsf{C}(X_{query}), \mathsf{C}(X), d)] \tag{28}$$

$$E(|FP|) = \sum_{X \in X(\mathcal{P}) \setminus \mathcal{C}_d} \mathsf{P}(\mathsf{C}(X_{query}), \mathsf{C}(X), d) \tag{29}$$

$$|P_A| = |X(\mathcal{P}) \cap \mathcal{C}_d| \tag{30}$$

The $X(\mathcal{P})$ is the set of locations of all posts in the system. The distribution of $X(\mathcal{P})$ may vary across environments, so one system setting can produce different metrics.

B Additional Figures

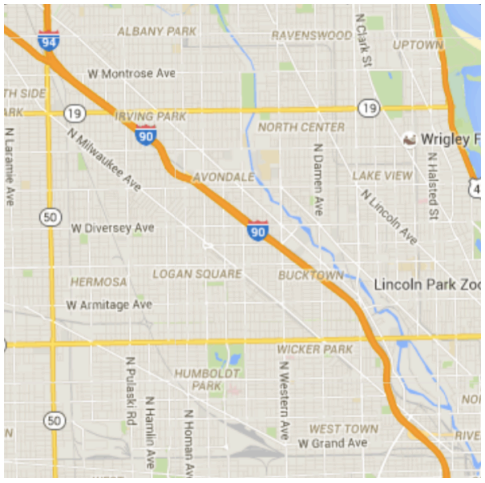
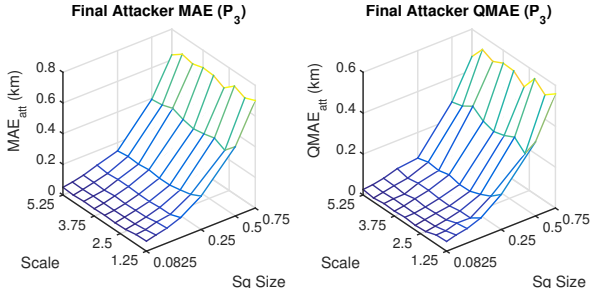
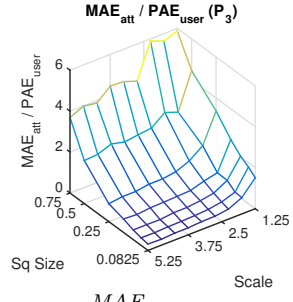


Fig. 12. North Chicago geographical map.

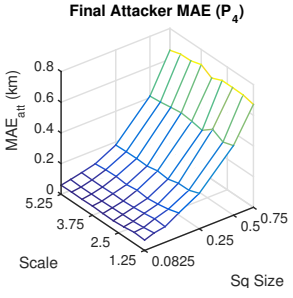


(a) MAE_{att} of P_3.

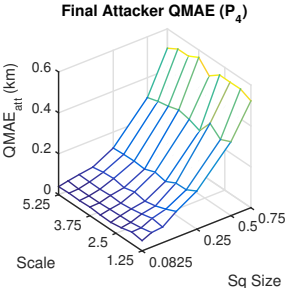
(b) $QMAE_{att}$ of P_3.



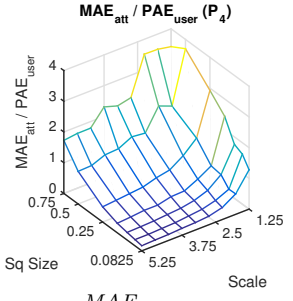
(c) $\frac{MAE_{att}}{PAE_{user}}$ of P_3.



(d) MAE_{att} of P_4.

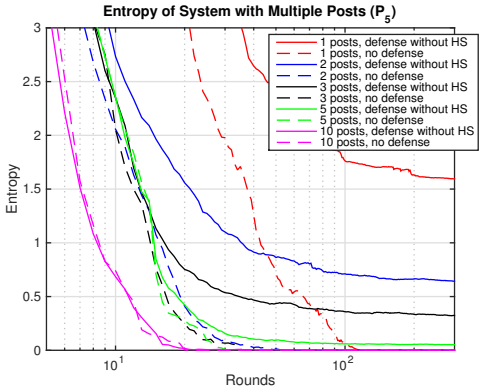


(e) $QMAE_{att}$ of P_4.

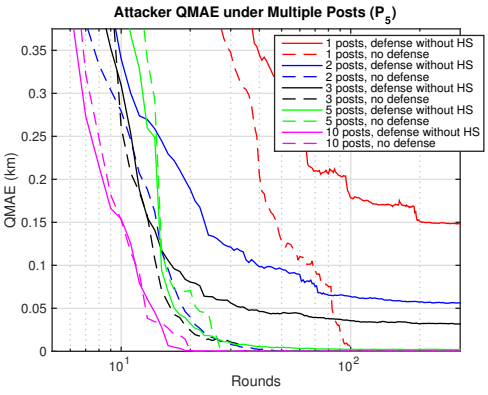


(f) $\frac{MAE_{att}}{PAE_{user}}$ of P_4.

Fig. 13. The average attacker MAE (MAE_{att}), quantized MAE ($QMAE_{att}$), and privacy-to-user-error ratio (MAE_{att} / PAE_{user}) of $\mathsf{P}_3(x, d)$ and $\mathsf{P}_4(x, d)$ over varying map scale (no hierarchical subdivision) and quantized square size. Each point represents the average of 100 runs.

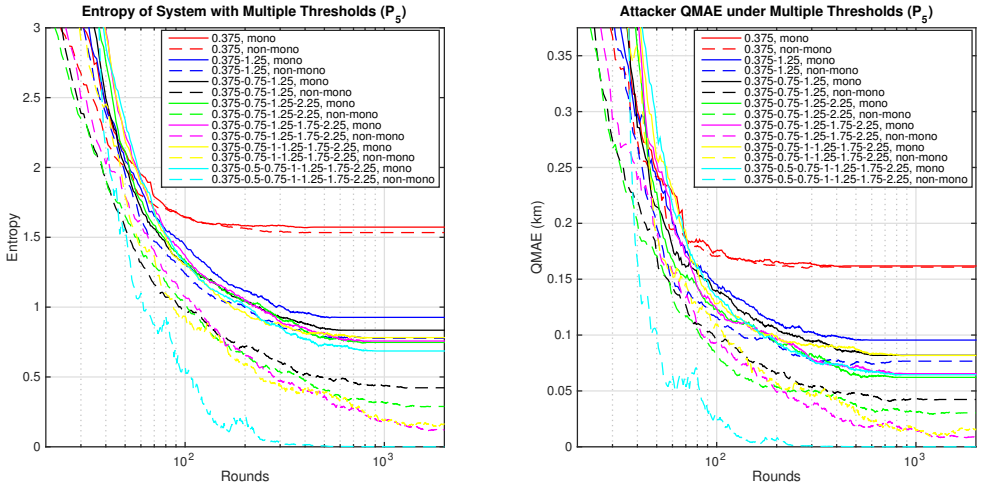


(a) Entropy of P_5 vs. rounds.



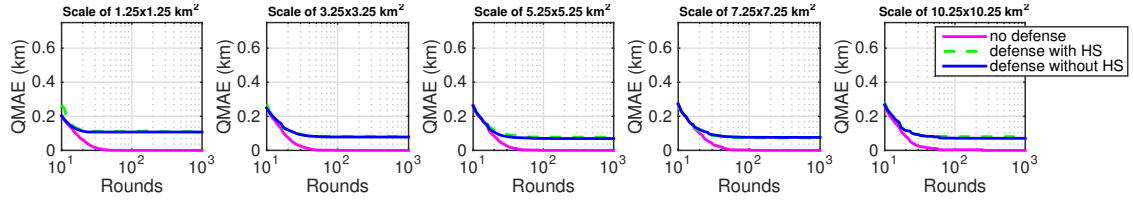
(b) $QMAE_{att}$ of P_5 vs. rounds.

Fig. 14. Non-user-consistent defense privacy loss rate. Average attacker entropy and QMAE (over 100 runs) against system with multiple same-location posts from one user in the 5.25 km $\times$ 5.25 km North Chicago map. The square size is 0.25 km $\times$ 0.25 km. The victim is at $X_{post} = (0.5, 0.5)$ km and the threshold $d = 0.375$ km.

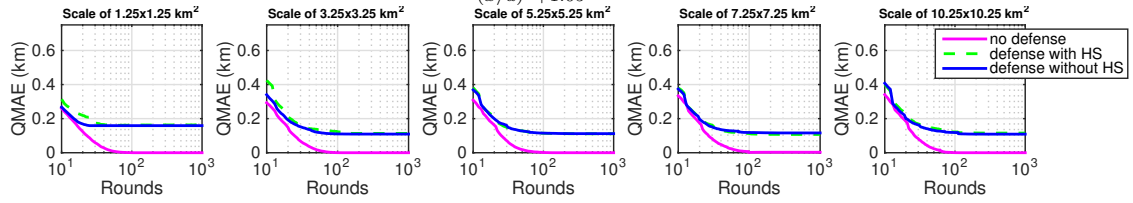


(a) Entropy of P_5 vs. rounds. (b) $QMAE_{att}$ of P_5 vs. rounds.

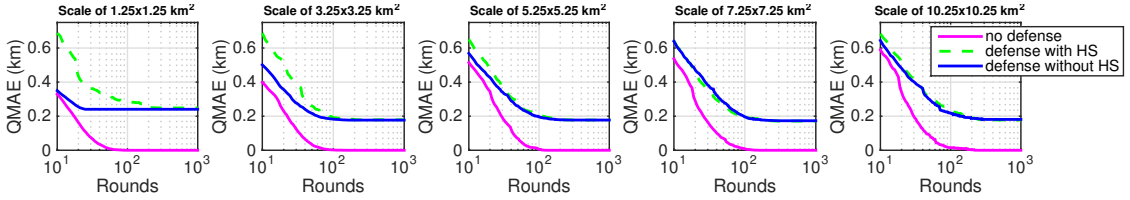
Fig. 15. Average attacker entropy and QMAE (over 100 runs) against non-monotonic-response and monotonic-response systems with multiple thresholds in the 5.25 km × 5.25 km North Chicago map. The square size is 0.25 km × 0.25 km. The victim is at $X_{post} = (0.5, 0.5)$ km and the threshold $d = 0.375$ km. The threshold settings are presented in the figure legend. For example, "0.375-1.25" means thresholds are 0.375 and 1.25 km.



(a) Case of $\mathsf{P}_3(x, d) = \frac{1}{(x/d)^4+1.05}$ response probability function.

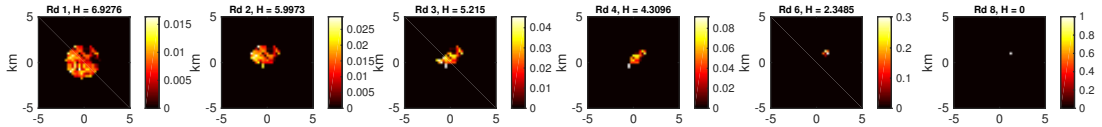


(b) Case of $\mathsf{P}_4(x, d) = \frac{1}{(x/d)^3+1.05}$ response probability function.

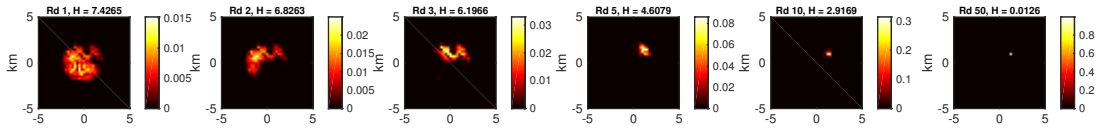


(c) Case of $\mathsf{P}_5(x, d) = \frac{1}{(x/d)^2+1.05}$ response probability function.

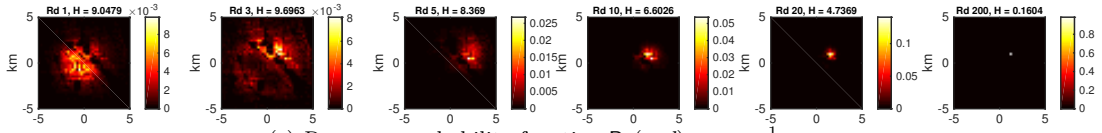
Fig. 16. The average attacker entropy and QMAE ($QMAE_{att}$) over North Chicago maps with different scales, using a fixed square size $L = 0.25$ km. We choose a threshold $d = 0.375$ km, and average the $QMAE_{att}$ across 10 victims randomly sampled from the map with scale of 1.25 km × 1.25 km. HS denotes runs on a 10.25 km × 10.25 km map with Hierarchical Subdivision; the map scale reflects the size of the base map. Each point represents the average of 100 runs.



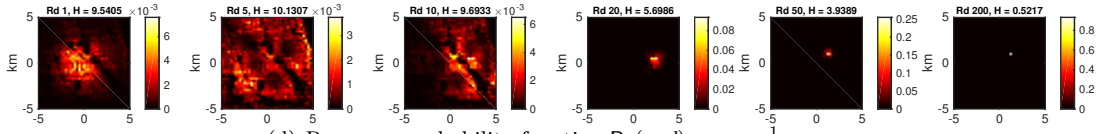
(a) Response probability function $\mathsf{P}_1(x, d)$



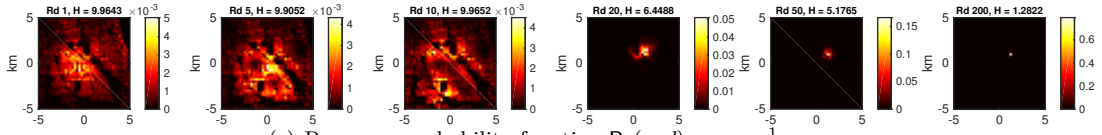
(b) Response probability function $\mathsf{P}_2(x, d)$



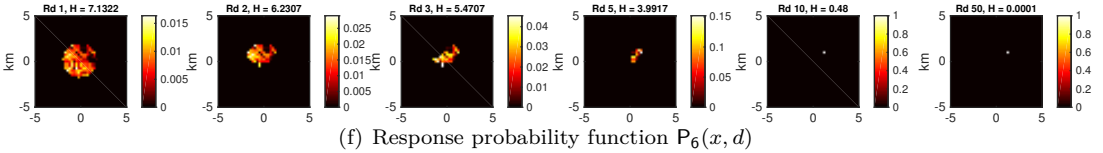
(c) Response probability function $\mathsf{P}_3(x, d) = \frac{1}{(x/d)^4+1.05}$



(d) Response probability function $\mathsf{P}_4(x, d) = \frac{1}{(x/d)^3+1.05}$

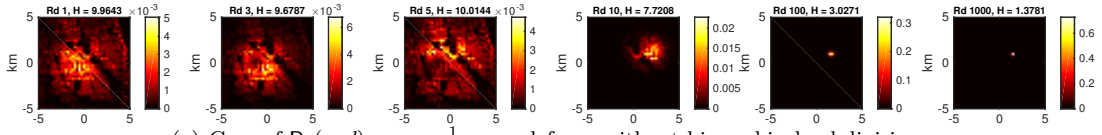


(e) Response probability function $\mathsf{P}_5(x, d) = \frac{1}{(x/d)^2+1.05}$

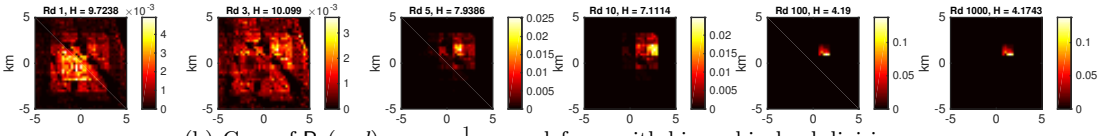


(f) Response probability function $\mathsf{P}_6(x, d)$

Fig. 17. No defense privacy loss rate. Probability distribution map vs round number (attacker starts with population distribution) under no response-consistent defense. The map scale is 10.25 km × 10.25 km, the square size is 0.25 km × 0.25 km, the victim post is at $(1.25, 0.75)$ km, and the threshold distance is 1.75 km. The number of rounds between each image varies between the functions to show the effectiveness of each probability response function.



(a) Case of $\mathsf{P}_5(x, d) = \frac{1}{(x/d)^2+1.05}$, defense without hierarchical subdivision.



(b) Case of $\mathsf{P}_5(x, d) = \frac{1}{(x/d)^2+1.05}$, defense with hierarchical subdivision.

Fig. 18. Hierarchical-subdivision defense privacy loss rate. Change of probability distribution map by rounds. We run the system with three different response probability functions ($\mathsf{P}_3(x, d)$, $\mathsf{P}_4(x, d)$, and $\mathsf{P}_5(x, d)$) under the defense without hierarchical subdivision and the defense with hierarchical subdivision. Due to space limit, we just show the result of $\mathsf{P}_5(x, d)$ here, and other results are similar. The scale of the whole map is 10.25 km× 10.25 km, the victim post X_{post} is at $(1.25, 0.75)$ km, the threshold $d = 1.75$ km, and the quantized square size $L = 0.25$ km. In the defense with hierarchical subdivision, the scale of the base map is 1.25 km× 1.25 km, and we expend it to the map 10.25 km× 10.25 km. Attacker knows the population density in advance.