On the Impact of Location Errors on Localization Attacks in Location-Based Social Network Services

Hanni Cheng[1], Shiling Mao[1], Minhui Xue[2,3], and Xiaojun Hei[1](✉)

[1] Huazhong University of Science and Technology, Wuhan 430074, China
heixj@hust.edu.cn
[2] East China Normal University, Shanghai 200062, China
[3] NYU Shanghai, Shanghai 200122, China

Abstract. Location-based Social Network (LBSN) services, such as *People Nearby* in WeChat, enable users to discover users within the geographic proximity. Though contemporary LBSN services have adopted various obfuscation techniques to blur the location information, recent research has shown that based on the number theory, one can still accurately pinpoint user locations by strategically placing multiple virtual probes. In this paper, we conducted a comprehensive simulation study to examine the impact of location errors on localization attacks to track target users based on the number theory by using the LBSN services provided by WeChat. Our simulation experiments include four location error models including the exponential model, the Gaussian model, the uniform model, and the Rayleigh model. We improve the one-dimensional and two-dimensional localization algorithms where the location errors exit. Our simulation results demonstrate that the number theory based localization attacks remain effective and efficient in that target users can still be pinpointed with high accuracy.

Keywords: Privacy leakage · Localization attack · Error analysis · Location-based social network · WeChat

1 Introduction

The proliferation of smart phones has spawned the development of many popular pervasive location-based services (LBSs). In recent years, Location-Based Social Network (LBSN) services, enable users to discover their geographic neighbors and then communicate. For these services to function properly, the location of users have to be provided by the system; nevertheless, the integrity of user location privacy must be preserved. Otherwise, potential location privacy leakage may arise and users' location information may be misused by malicious attackers [1–4]. Some research has been conducted to study this new type of localization attacks and the defense schemes for mobile users against localization attacks [5,6].

Among all the well-known LBSN applications, we focus our study on WeChat. With over 600 million registered users, WeChat has become the largest user group that provides an instant messaging service for intelligent terminals. WeChat also provides LBSN services by sharing instant data such as *"People Nearby," "Shake," "Circle of Friends," and "Drift Bottles"* [7,8]. In [5], Xue et al. first conducted a theoretical study on the privacy leakage problem of online social applications and then proposed an effective approach to track target users in a simple one-dimensional case based on the number theory. In [9], Xue et al. extended a localization attack to a more general two-dimensional case. The theoretical analysis shows its effectiveness without considering the location errors. Peng et al. [10] developed a new two-dimensional algorithm in spite of location errors. In this paper, we first validate that the fundamental algorithm in [10] can work correctly; then we introduce four error models considering real-world location errors and improve the one-dimensional (1-D) algorithm and the two-dimensional (2-D) algorithm under the four error models.

The rest of the paper is organized as follows. In Sect. 2, we introduce the *"People Nearby"* service in Wechat and discuss the privacy-leakage problem due to this LBSN service. We also present the basic 1-D algorithm and the 2-D algorithm to determine the locations of users in practice. We then introduce four types of the error models in Sect. 3. Furthermore, we propose an improved 1-D algorithm in Sect. 4 and an improved 2-D algorithm in Sect. 5. Finally, we conclude the paper in Sect. 6.

2 Problem Statement

With the popularity of location-based services, improper use of user location information may bring privacy breaches. To defend against the trilateration-based localization attacks, contemporary LBSN applications have applied various obfuscation techniques to blur the location information. In the *"People Nearby"* service provided by WeChat, one obtains a list of user names and relative distances of people nearby when using *"People Nearby."* However, these relative distances are not so accurate; instead, WeChat only reports the relative distance in bands of 100 m or 1000 m. For example, two users, Alice and Bob, are using the *"People Nearby"* service. When WeChat shows to Bob that Alice is 800 m away from him, it means that Alice is located in a band centered at Bob's location with the radius ranging from 700 m to 800 m. Such a band-based approach to report a rough relative distance of nearby users, so that users are not able to obtain the accurate coordinates of target users directly. However, using the number theory, Xue et al. proved that one can pinpoint the location of a target user within a circle of radius no greater than 1 m theoretically [5]. We first examine two basic location attacks as follows.

2.1 1-D Algorithm

We first consider a special linear case. Assume that an attacker places multiple virtual probes on the line of a target user, while the probes can obtain the relative distances of the target using the *"People Nearby"* service. We summarize the notation introduced throughout this section in Table 1.

Table 1. Summary of notations for the 1-D algorithm

Symbol	Meaning
K	The length of band
x	The distance between one probe and any adjacent probe
r	The reported error generated by WeChat
d_i	The actual distance between the probe P_i and the target point
W_{p_i}	The reported distance between the probe P_i and the target point
D_{p_1}	The estimated distance between the probe P_1 and the target point
$OneDim$	One-dimensional function
Z	The set of integers
$\gcd(\cdot,\cdot)$	The greatest common divisor

In Fig. 1, assuming that T is the target, a number of isometric probes are placed on the line. We can obtain the return values of the probes W_{p_i}. The relation between the reported relative distance W_{p_i} and the actual relative distance d_i can be determined following the basic 1-D algorithm in [10].

$$W_{p_i} = \left(\left\lfloor \frac{d_i}{K} \right\rfloor + 1 \right) \times K, \tag{1}$$

where

$$K = \begin{cases} 100, & 0 \le d < 1000, \\ 1000, & d \ge 1000. \end{cases} \tag{2}$$

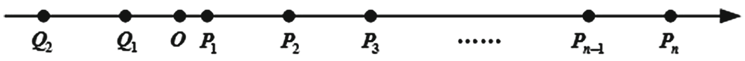


Fig. 1. Basic idea of the 1-D algorithm

In order to verify the correctness of the basic 1-D algorithm, we conducted simulation experiments using Matlab. Assume that the location error is 0, and the distances between probes are 11 m. Some representative simulation results of the 1-D algorithm are shown in Table 2.

By inspecting Table 2, we find that some errors are more than 1 m, which is unexpected. A deep examination shows for the point whose original location is already large enough since the coordinate of some probes will be exceeding 1,000 m. Based on the relationship between the actual distance and the reported distance by *"People Nearby,"* the reported distance will be 2,000 m (the band

Table 2. Some representative results of the 1-D algorithm

Actual distance (m)	Predicted distance (m)	Error (m)
322.43	322.50	0.07
555.16	555.50	0.34
804.52	833.50	8.97
24.41	24.50	0.09
371.51	371.50	0.01
491.87	491.50	0.37
466.05	466.50	0.45
41.71	41.50	0.21
617.00	622.50	5.50
578.03	578.50	0.47

will be 1,000 m). When the coordinate is too large, the 1-D algorithm is not able to work correctly. In Sect. 4, we will discuss how to reduce the errors by changing the location of the first probe. Although one is not able to locate the target within a circle of radius no greater than 1 m, over 90 % of the errors are less than 10 m. Therefore, the 1-D algorithm is sufficiently accurate for practice.

2.2 2-D Algorithm

Xue et al. developed a 2-D algorithm which can locate the target user very precisely for a triangle area [9]. Assume that the target user is in a triangle whose side length is X, as shown in Fig. 2.

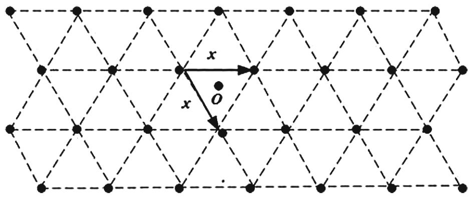


Fig. 2. Two-dimensional Lattice

The bar area is divided into several equilateral triangle whose side length is X. Xue et al. proved that one can pinpoint the target user with error no more than 1 m using the 2-D localization algorithm on all the sides of the triangle, as shown in Fig. 3.

The notation in the 2-D algorithm are shown in Table 3. We then introduce the basic principle of the 2-D algorithm. Take $\overrightarrow{X}$ direction in Fig. 3 as an example,

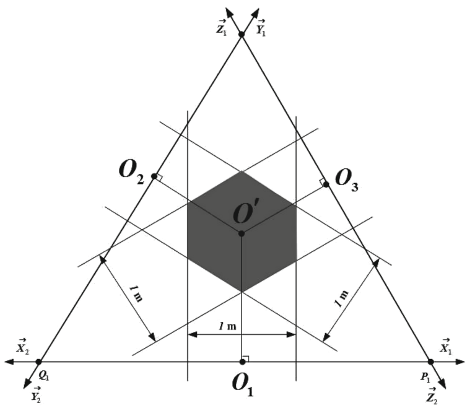


Fig. 3. An Illustration of the overlapping area

we place probe P_1 and Q_1 on the vertices of the equilateral triangle, and place p_i and q_i on the extension cord so that we can obtain the reported distance w_{p_i} and w_{q_i}, as shown in Fig. 4.

Table 3. Summary of notations for the 2-D algorithm

Symbol	Meaning
O'	The target point (the location of the user)
O_1	The first projection of the target point to a line of virtual probes
$d_{p_i} = \|O_1 P_i\|$	The actual distance from O_1 to the probe P_i
$d_{q_i} = \|O_1 Q_i\|$	The actual distance from O_1 to the probe Q_i
D_{p_i}	The estimated distance from O' to the probe P_i
D_{q_i}	The estimated distance from O' to the probe Q_i
w_{p_i}	The reported distance from O' to the probe P_i
w_{q_i}	The reported distance from O' to the probe Q_i

For the reported distance w_{p_i} and w_{q_i}, we run the 2-D algorithm to obtain the output D_{p_1} and D_{q_1}. The step $N = (\lfloor \frac{x}{K} \rfloor + 1) \times K + T \cdot s \pmod{K} + 1$ should be carefully selected; hence, we need more probes to ensure the precision if X is larger.

We conducted simulation experiments to evaluate the 2-D algorithm. If the algorithm works well in the given direction $\overrightarrow{X}$, the 2-D algorithm can pinpoint the target precisely. We place a few probes on the extension line of the triangle side in Fig. 4 and then we obtain the coordinates of the probes w_{p_i} and w_{q_i}. The parameter X is set as 99; hence, both the triangle side length and the interval of probes are 99. What we expect is that the target can be located quite precisely

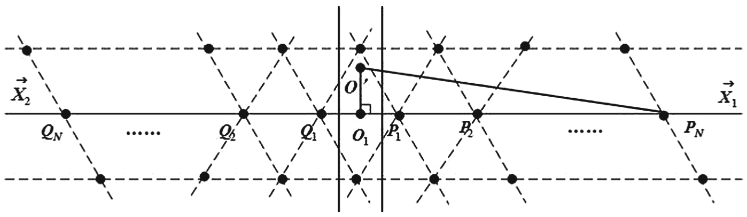


Fig. 4. Basic idea of the 2-D algorithm

in the equilateral triangle. We generate 1,000 random targets for the tracking experiments, and the average tracking error of these 1,000 targets is 25.15 m. Some representative simulation results are shown in Table 4.

Table 4. Some representative results of the 2-D algorithm

Actual distance (m)	Predicted distance (m)	Error (m)
42.37	49.50	7.13
84.43	49.50	34.92
22.40	49.50	27.10
63.22	49.50	13.72
9.00	49.50	40.50
77.59	49.50	28.10
69.25	49.50	19.75
66.66	49.50	17.16
46.15	49.50	3.34
27.14	49.50	22.36

These results show that most errors are much larger than what we expected. In the 2-D algorithm, if $X = 99$, we need 200 probes. As a result, most reported distances of the probes are more than 1,000 m, so that K is 1,000 m instead of 100 m which does not meet the assumptions of the 2-D algorithm. However, if we set X smaller, then the location range is too small that our algorithm is not practical. Therefore, we conclude that the basic 2-D algorithm should be improved. After we examine the 4 error models, we propose an improved 2-D algorithm in Sect. 5.

3 Error Models

In the previous section, we assume that the relationship between reported distance and the actual distance is determined based on Eq. (1). Nevertheless, the actual distance and the location distance reported by WeChat may not be the same. Note that the localization methods may incur errors during the positioning process. Besides, WeChat may intentionally introduce errors to the location distance in order to protect user's privacy. These location errors may differ significantly. There have been no reported studies on the system localization errors in WeChat. In this section, we assume some commonly-used error models to simulate the gap between the positioning distance and the actual distance. Based on the analysis of these error models, we aim to better understand the impact of localization errors on localization attacks in LBSN services.

3.1 Error Measurement

In order to understand the relationship between the actual relative distance and the distance reported by WeChat, we measured the total 42 sets of data points in the field. In the measurement process, we found that even at the same location, the measured distances at different time instants are not the same. Based on the analysis of these data, we conjecture that the errors follow two empirical rules:

1. The location error is relatively small comparing with the actual distance;
2. The location errors are roughly proportional to the actual distance.

3.2 Model Settings

We studied 4 commonly-used error models including the exponential distribution model, the Gauss distribution model, the uniform distribution model, and the Rayleigh distribution model. The parameters of these four models are configured following the observed empirical rules. We conducted simulate experiments with these 4 error models to evaluate the 1-D algorithm. Then, we further improve the 1-D and 2-D algorithms in spite of the location errors. To make the fair comparison, we ensure the same mean of each error model.

Exponential Error Model. Assume the exponential distribution model with the following parameter settings, in which the *exprnd* is an exponential distribution function, and the parameter is the mean value.

$$r = \begin{cases} 0, & d \leq 100, \\ \text{exprnd}(5), & 100 < d \leq 200, \\ \text{exprnd}(10), & 200 < d \leq 400, \\ \text{exprnd}(50), & 400 < d \leq 800, \\ \text{exprnd}(100), & 800 < d \leq 1200, \\ \text{exprnd}(150), & \text{otherwise.} \end{cases}$$

Gaussian Error Model. Assume the Gaussian distribution model with the following parameter settings, where *normrnd* is the Gauss distribution function, the first parameter is the mean, and the second parameter is the variance. Since the probability of points in range $(\mu - 3\sigma, \mu + 3\sigma)$ is 99.7 %, we set $\sigma = \mu/3$ to ensure that the error is positive to match other error models.

$$r = \begin{cases} 0, & d \leq 100, \\ \text{normrnd(5,5/3)}, & 100 < d \leq 200, \\ \text{normrnd(10,10/3)}, & 200 < d \leq 400, \\ \text{normrnd(50,50/3)}, & 400 < d \leq 800, \\ \text{normrnd(100,100/3)}, & 800 < d \leq 1200, \\ \text{normrnd(150,150/3)}, & \text{otherwise.} \end{cases}$$

Uniform Error Model. Assume the uniform distribution model is configured with the following parameter settings, where $unifrnd$ is the uniform distribution function, the first parameter error is the maximum, and the second parameter is the minimum error.

$$r = \begin{cases} 0, & d \leq 100, \\ \text{unifrnd(0,10)}, & 100 < d \leq 200, \\ \text{unifrnd(0,20)}, & 200 < d \leq 400, \\ \text{unifrnd(0,100)}, & 400 < d \leq 800, \\ \text{unifrnd(0,200)}, & 800 < d \leq 1200, \\ \text{unifrnd(0,300)}, & \text{otherwise.} \end{cases}$$

Rayleigh Error Model. Note that the Rayleigh distribution meets the condition that $\mu(X) = \sigma\sqrt{\frac{\pi}{2}} \approx 1.253\sigma$. The parameter of the $raylrnd$ function in Matlab is the variance. In order to keep in line with the previous error models, the parameters are divided by the coefficient of 1.253.

$$r = \begin{cases} 0, & d \leq 100, \\ \text{raylrnd(5/1.253)}, & 100 < d \leq 200, \\ \text{raylrnd(10/1.253)}, & 200 < d \leq 400, \\ \text{raylrnd(50/1.253)}, & 400 < d \leq 800, \\ \text{raylrnd(100/1.253)}, & 800 < d \leq 1200, \\ \text{raylrnd(150/1.253)}, & \text{otherwise.} \end{cases}$$

3.3 Simulation Results

With the introduced location errors in the experiments, the simulation results show the degraded performance. Nevertheless, the accuracy of the location attacks still remain high as shown in Fig. 5. The performance trends with these four error model are similar, though. We will further improve the 1-D and 2-D algorithms in the next two sections.

4 Improving 1-D Algorithm

In this section, we propose to further reduce the errors by fine-tuning the position of the first probe.

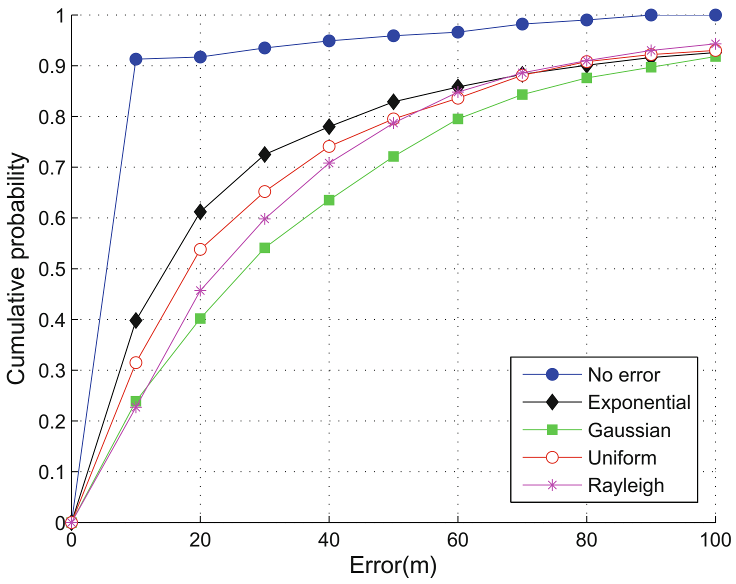


Fig. 5. Performance comparison by different error models

4.1 Basic Ideas

In Sect. 3, we assume that positioning error is proportional to the relative distance between two points. Therefore, we make preliminary positioning for the initialization of the 1-D algorithm. In the basic 1-D algorithm, probes are placed following Fig. 6. After determining the scope of 1,000 m, the first probe is placed on the origin point in the algorithm, and the rest probes are placed on the left side of the origin point in order. Eventually, the distance between the first probe and the target is used as the coordinate of the target. If the target point is far away from the origin point, the reported distance of all probes will also be very large. Hence, the error will be larger if the relative distance is large; besides, the error will increase greatly if the relative is even larger than 1,000 m.

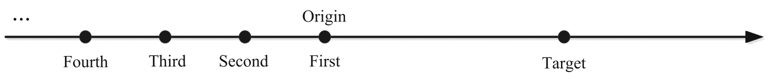


Fig. 6. Probe positioning in the 1-D algorithm

Since the distance band reported by *"People Nearby"* in WeChat is 100 m if the relative distance is less than 1,000 m, we can place 10 probes every 100 m to roughly determine the interval of the target point. Specifically, the preliminary positioning is as shown in Fig. 7.

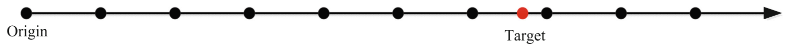


Fig. 7. Preliminary probe positioning in the 1-D algorithm

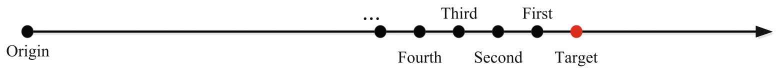


Fig. 8. Probe positioning in the improved 1-D algorithm

First of all, we place probes every 100 m within the determined 1,000 m, and we place 10 probes in total. Then, we can obtain the reported distances of the 10 probes w_{p_i},$(1 \leq i \leq 10)$. At last we can obtain the i value corresponding to the minimum w_{p_i}. We may obtain two of i which replace the closest probes to the target point. In Fig. 6, i is 6 or 7. We want to choose the smaller one so that we set $i = 6$.

We place the first probe at coordinate $(i - 1) \times 100$, the following probes will be placed on the left side of the first probe with an equal interval as shown in Fig. 8. In this manner, the distance between all probes and the target will be smaller so that the error will decrease. The output of the 1-D algorithm is the distance between target and the first probe, so that the coordinate of the target is the output d_{p_i} added by $(i - 1) \times 100$. In summary, this improved 1-D algorithm is more accurate by reducing the distances between all the probes and the target. However, the time cost increases because the new 1-D algorithm is more complicated.

4.2 Simulation Results

We evaluate the new 1-D algorithm with the parameter $X = 11$ with the 4 error models. In Fig. 9, OneDim $v1$ is the original 1-D algorithm and OneDim $v2$ is the newly proposed 1-D algorithm. In Fig. 9, the results in four figures are very similar; the errors in OneDim $v2$ are much smaller than that in OneDim $v1$. Almost all the errors are less than 40 m in our new 1-D algorithm.

4.3 Summary

In this section, we evaluate the 1-D algorithm with 4 error models. The optimized 1-D algorithm makes tracking more accurate by preliminary positioning with these 4 error models. It shows that even under the different error models, the target can be located quite accurately using the new version, which indicates the location of users can still be tracked with high accuracy. Though the errors can be reduced, the complexity of this new 1-D algorithm increases.

5 Improving 2-D Algorithm

We have discussed that the tracking area should bounded in an equilateral triangle whose side length is upper-bounded. The 2-D algorithm cannot pinpoint

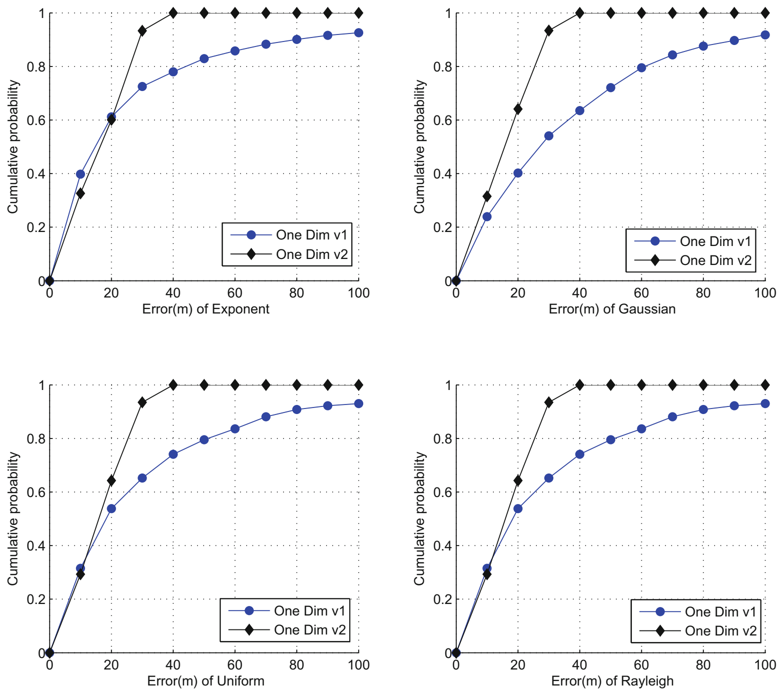


Fig. 9. Improving the 1-D algorithm to reduce errors

the target accurately if K is not 100 or extra errors exist. However, positioning errors are difficult to be neglected in practice. The tracking performance of the original 2-D algorithm is degraded with the inherent localization errors of the system. Another 2-D algorithm that can work well when extra error is added is proposed in [10]. The target user can be located precisely in an square area of $1,000 \times 1,000$. 1-D algorithm is carried out on both X axis and Y axis so that the coordinates of the target can be determined. In this section, we improve the 2-D algorithm in [10] and report our simulation results.

5.1 Basic Ideas

In Sect. 4, we notice that if the relative distances between the target and the probes are too large, the tracking results will be of less accurate. Thus, we follow the preliminary positioning idea to estimate a better initial location of the target before we use the 1-D algorithm to obtain more accurate coordinates.

First of all, we make preliminary positioning on both abscissa and ordinate as shown in Fig. 10. There is a target point in a square area of $1,000 \times 1,000$. We take the lower left corner of the square as the origin point, and place probes along the X axis and the Y axis every d m, then we obtain the reported distance w_{p_x} and w_{p_y}. P_x and P_y are the corresponding coordinates of the probe corresponding

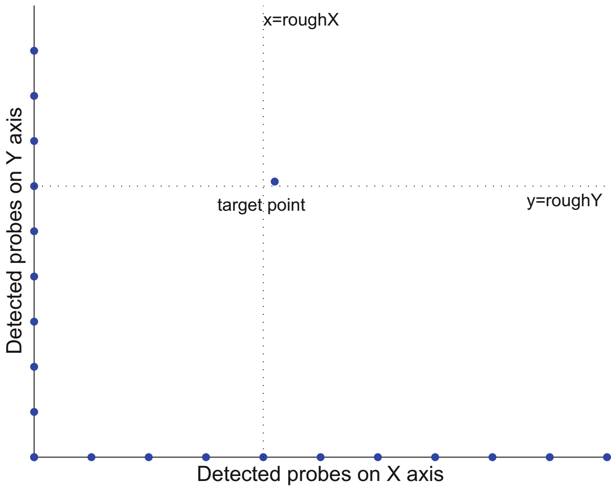


Fig. 10. Preliminary probe positioning in the 2-D algorithm

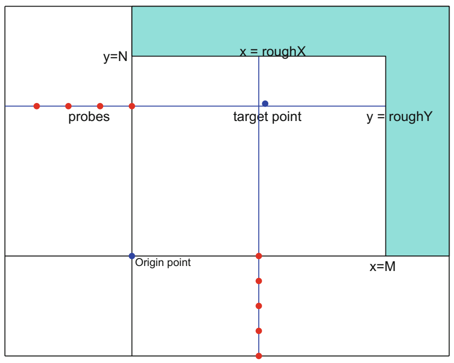
with the minimum of w_{p_x} and w_{p_y}. Afterwards, we have two lines: $x = \text{rough}X$ and $y = \text{rough}Y$ as shown in Fig. 10. In the next positioning step, we use the 1-D algorithm on $y = \text{rough}Y$ and $x = \text{rough}X$ to obtain the accurate coordinates of x and y. Then, the probes will be placed on the two lines as shown in Fig. 10.

We conduct some extra steps before by applying the 1-D algorithm. The analysis in Sects. 3 and 4 shows that larger relative distances between the probes cause larger tracking errors. As a result, if the target locates at the top right part of the square area far from the origin point, the positioning error may increase dramatically. The abnormal error may have a strong impact on the performance of the localization algorithm.

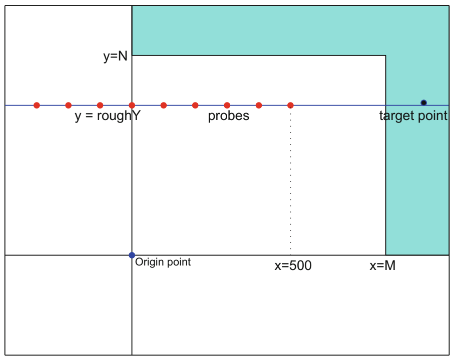
In order to reduce these above-mentioned abnormal errors, we attempt to partition the target area according to the distribution of localization errors as shown in Fig. 11. For the points on the right side of $x = M$, we still use the 1-D algorithm on $x = \text{rough}X$ and $y = \text{rough}Y$, but the first probe should be placed on the crossing point at $x = 500$ and $y = \text{rough}Y$ if we want to obtain the coordinate x. Similarly, for the points on the top side if $y = N$, the first probe should be put on the crossing of $y = 500$ and $x = \text{rough}X$ if we want to obtain the coordinate y. Figure 11 shows the detailed steps.

5.2 Simulation Results

At first we determine the values of M and N. Since the area is a square, M and N are considered to be the same. We expect that the error can be reduced to the minimum after M and N are carefully selected. The measurement of the performance is the number of points whose positioning error is more than 100 m, smaller n indicates better performance. For the 4 error models in Sect. 3, we start iterations from 501 to $1,000$ to find the best M and N for each error model.



(a) 2-D probe positioning in white area



(b) 2-D probe positioning in blue area

Fig. 11. Localization steps in the 2-D algorithm (Color figure online)

The best M (or N) of the exponential model, the Gaussian model, the uniform model, and the Rayleigh Model are 512, 549, 503, and 520.

5.3 Summary

In this section, we examine the 2-D algorithm proposed in [10] with 4 error models. We can first determine the approximate location of the target user by preliminary positioning, then obtain the accurate coordinates by applying the 1-D algorithm on both X axis and Y axis. The simulation results show that the improved 2-D algorithm can still locate a target user quite accurately. Since the 2-D algorithm is more complicated, more time is needed for the tracking process. We emphasize again that the tradeoff between the accuracy and the overhead should be considered for effective attacks.

6 Conclusion

Contemporary LBSN applications have adopted the band-based approach to report distances of nearby users. In this paper, we show how the location-based feature of WeChat can be exploited to determine the user's location with great accuracy in any city from any location in the world. We examined the location algorithms developed in our previous work with 4 location error models to better evaluate the performance of the real-world attacks. Simulation results show that the improved 1-D algorithm and the 2-D algorithm still achieve good performance even under 4 error models. Our research may bring this serious privacy pertinent issue into the spotlight based on comprehensive experiment results and hopefully motivate better privacy-preserving LBSN designs.

Acknowledgments. This work was supported in part by the National Natural Science Foundation of China (No. 61370231), and in part by the Fundamental Research Funds for the Central Universities (No. HUST:2016YXMS303).

References

1. Ding, Y., Peddinti, S.T., Ross, K.W.: Beijing stalking from timbuktu: a generic measurement approach for exploiting location-based social discovery. In: ACM Workshop on Security and Privacy in Smartphones and Mobile Devices, pp. 75–80 (2014)
2. Li, M., Zhu, H., Gao, Z., Chen, S., Le, Y., Shangqian, H., Ren, K.: All your location are belong to us: breaking mobile social networks for automated user location tracking. In: ACM Mobihoc, pp. 43–52 (2014)
3. Polakis, I., Argyros, G., Petsios, T., Sivakorn, S., Keromytis, A.D.: Where's wally? Precise user discovery attacks in location proximity services. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, pp. 817–828. ACM (2015)
4. Xue, M., Ballard, C.L., Liu, K., Nemelka, C.L., Yanqiu, W., Ross, K.W., Qian, H.: You can yak but you can't hide: localizing anonymous social network users. In: Proceedings of the 2016 Conference on Internet Measurement Conference. ACM (2016)
5. Xue, M., Liu, Y., Ross, K.W., Qian, H.: I know where you are: thwarting privacy protection in location-based social discovery services. In: 2015 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), pp. 179–184 (2015)
6. Shokri, R., Theodorakopoulos, G., Papadimitratos, P., Kazemi, E., Hubaux, J.P.: Hiding in the mobile crowd: location privacy through collaboration. IEEE Trans. Dependable Secure Comput. **11**(3), 266–279 (2014)
7. Wang, R., Xue, M., Liu, K., Qian, H.: Data-driven privacy analytics: a WeChat case study in location-based social networks. In: Xu, K., Zhu, H. (eds.) WASA 2015. LNCS, vol. 9204, pp. 561–570. Springer, Heidelberg (2015). doi:10.1007/978-3-319-21837-3_55
8. Xue, M., Yang, L., Ross, K.W. et al.: Characterizing user behaviors in location-based find-and-flirt services: anonymity and demographics. Peer-to-Peer Netw. Appl. 1–11 (2016). doi:10.1007/s12083-016-0444-5

9. Xue, M., Liu, Y., Ross, K.W., Qian, H.: Thwarting location privacy protection in location-based social discovery services. Secur. Comm. Networks. **9**, 1496–1508 (2016). doi:10.1002/sec.1438
10. Peng, J., Meng, Y., Xue, M., Hei, X., Ross, K.W.: Attacks, defenses in location-based social networks: a heuristic number theory approach. In: International Symposium on Security and Privacy in Social Networks and Big Data (SocialSec), pp. 64–71, November 2015. doi:10.1109/SocialSec2015.19