Future Generation Computer Systems ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

The flexible and privacy-preserving proximity detection in mobile social network

Ayong Ye *, Qiuling Chen, Li Xu, Wei Wu

Key Lab of Network Security and Cryptology, Fujian Normal University, Fuzhou, 350007, China

HIGHLIGHTS

- A proximity detection method based on the transfer of neighbor relation is proposed.
- The method takes beacon signals as a reference of neighbor discovery.
- The users whose nearby reference lists have a common item are neighbors.
- Two ways of transmitting signals of beacon nodes are proposed.
- The energy loss and the rate of signal coverage should be well controlled.

ARTICLE INFO

Article history:
Received 28 August 2016
Received in revised form
30 October 2016
Accepted 9 December 2016
Available online xxxx

Keywords:
Mobile social network
Proximity detection
Location privacy

ABSTRACT

With the popularity of mobile social network, proximity detection has become a fundamental service. For the traditional proximity detection methods, users need to upload their locations to a location server so that they can find their neighbors by calculating relative distances among them. It will cause significant privacy concerns when the location server is an untrusted one. To solve this problem, we propose a proximity detection method which is based on the transfer of neighbor relation. Specifically, each request user in our paper only needs to submit a nearby reference list to the social network server (SNS). After that, the SNS searches the neighbors of the request user by judging whether their nearby reference lists have a common item. Moreover, we present two mechanisms to determine the ways of transmitting signals of beacon nodes, i.e., Beacon Node Rotating Mechanism and Beacon Node Competition Mechanism, respectively. Besides, we experimentally demonstrate the effectiveness and feasibility of the proposed method.

1. Introduction

Driven by the rapid development of mobile Internet and social networks [1], the representatives of the social network sites like Facebook [2] and Twitter [3,4] grow fast. And the number of users is growing at an alarming rate. Nowadays, more and more users use the proximity detection service such as "people in the vicinity", "nearby restaurants", which make the proximity detection becomes a basic service in mobile social network. The traditional proximity detection methods require users to submit their location information to the location server in order to find their neighbors. This approach is unsafe when the location server is an untrusted one. For instance, if the location server has security vulnerability or the insider abuses users' location information, the location privacy of users will be disclosed.

In order to solve the problems of traditional proximity detection methods, we propose a novel privacy-preserving proximity detection method. Specifically, instead of using users' locations to find their neighbors, the proposed method takes beacon signals as a reference of neighbor discovery. More specifically, users can find their neighbors by determining whether users' nearby reference lists have a common item. In addition, we also present two distributed mechanisms – Beacon Node Rotating Mechanism and Beacon Node Competition Mechanism – to determine the ways of transmitting signals of beacon nodes. Specifically, in the presented mechanisms, each mobile user upgrades and services as a beacon node. For facilitating other users to find their neighbors, these beacon nodes send beacon signals (i.e., the references) periodically.

The main contributions of this paper can be summarized as follows:

* Corresponding author.
E-mail addresses: yay@fjnu.edu.cn (A. Ye), 695568779@qq.com (Q. Chen), xuli@fjnu.edu.cn (L. Xu), weiwu81@gmail.com (W. Wu).

http://dx.doi.org/10.1016/j.future.2016.12.012
0167-739X/

(1) We propose a novel proximity detection method which is based on the transfer of neighbor relation. The SNS searches the neighbors of request user by judging whether their nearby reference lists have a common item.
(2) We present two mechanisms, i.e., Beacon Node Rotating Mechanism and Beacon Node Competition Mechanism, to determine the ways of transmitting signals of beacon nodes. Specifically, each mobile user in these mechanisms acts as a beacon node and periodically sends out a beacon signal as a neighbor discovery reference to other users.

The rest of this paper is organized as follows. Related works are reviewed in the next section. In Section 3, we present the overall scheme and the principle of the new proximity detection method. Besides, we also introduce the detailed process of algorithm to quickly match nearby reference lists. Moreover, the Beacon Rotating Mechanism and Beacon Node Competition Mechanism are the two methods for mobile users to send a beacon signal. We also introduce the process of these two mechanisms, respectively. The security analysis and discussions are presented in Section 4. The experimental results are presented in Section 5. Section 6 concludes this paper and presents the future work.

2. Related work

2.1. Location privacy protection method in LBS

In recent years, the main techniques to preserve location privacy in Location Based Services (LBS) are as follows.

Beresford in [5] proposed a conception of "Mix Zone" to hide all users' locations. Specifically, users in the mix zone have no need to communicate with each other. More specifically, when a user enters a mix zone, he changes his pseudonym and then gets out the mix zone. It is difficult for LBS to know who the user is when the user obtains a new pseudonym and goes out of the mix zone.

Gruteser et al. [6] first introduced k-anonymity into location privacy. It meets location k-anonymity when the location of a mobile user cannot be distinguished from other $k-1$ users. This method protects the location privacy by sending a minimum cloaking region which contains at least k users to the LBS instead of sending a single user's exact location. In this method, a trusted third-party is employed to generate these minimum cloaking regions through collecting the locations of different mobile users.

Spatial cloaking method can be employed to reduce the accuracy of location information and protect location privacy. For example, Ardagna presented a typical spatial cloaking method in [7]: the user sends a circle region to LBS instead of an exact location. However, due to the reason that the LBS cannot obtain the exact locations of users, its service quality will decrease.

The above methods are ineffective in the mobile social network. For instance, an attacker can obtain the location information or non-location information of users through the variety ways. Taking advantage of this information, the attacker can directly or indirectly reconstruct the location privacy of those users.

2.2. The proximity detection method

In order to provide proximity detection service on mobile social network without exposing user's location information, researchers have proposed numerous solutions. To the best of our knowledge, they can be categorized in four main classes: grid dividing, location tags, location anonymity and encryption method.

(1) The proximity detection based on grid dividing

In 2009, Šikšnys et al. in [8] proposed a method "grid-and-hashing", which divides space into a uniform grid unit (cell), and the ID of unit that each user located in has irreversibly hashed before sending them to the location server. In this manner, the server cannot get any location information about users when it is detecting proximity. However, due to the division of grid is pre-specified, this method exists false negative results, namely although two mobile users are close to each other, they are divided into different cells. As a result, they generate different hash values and submit the values to the server. For this reason, a "grid overlay" technology was designed in [9]; each user has a corresponding hash vector by using a series of interlocking grids. It indicates that two users are neighbors if they have a same hash result in a particular vector dimension. In this method, the layout and placing of grid will have a significant impact on reducing false negative rate. Vicinity Locator, is addressed in [10], makes users specify their areas of interest named "vicinity region". This region can be flexibly changed on the fly. One user's friends are the user's neighbors only when they are in the region. In [11], this "vicinity region" is centered on the false location generated by using differential privacy techniques, which can avoid location exposure.

Zhuo et al. [12] presented another proximity detection based on grid dividing. Each user can define his privacy region and two users are neighbors only when their privacy regions have an intersection. Moreover, this method supports users to verify the correctness of proximity test results from the server.

(2) The proximity detection based on location tags

Refs. [13–15] proposed proximity detection methods based on location tags. The spatial–temporal location tags, which are constructed from radio signals captured in a device's surrounding environment, such as WiFi and LTE signals. An attacker cannot forge a location tag if she is not at the corresponding location and time, due to the high freshness (entropy) and spatial variety of environmental signals. The location tags can resist location cheating for malicious users, but there exist some difficulties to construct a suitable location tag.

(3) The proximity detection based on location anonymity

Location anonymity is a location privacy protection method of most proximity detections adopted. Ruppel et al. [16] applied a distance-preserving mapping to transform the user's location q into a location q'. After the transformation, a centralized proximity detection method is presented to detect the proximity among the transformed locations. Nevertheless, Liu et al. in [17] pointed out that such distance preserving mapping is not secure. An outside attacker can easily derive the secret mapping function and recover the original location of users. Hide & Crypt is a privacy-preserving method to achieve proximity detection by taking a filter-and-refine two-phase procedure [18]. In the first phase of [18], all users cloak their locations before sending them to the server. In the second phase, the server computes the minimum and maximum distances between these cloaking regions. The server classifies users' friends being in, not-in, or possibly-in proximity according to the specified thresholds and the computed distances.

(4) The proximity detection based on encryption method

Currently, many privacy-preserving proximity detections based on encryption have been presented. For example, Refs. [19,20] presented a proximity detection method based on homomorphic encryption, respectively. According to the characteristics of homomorphic encryption, the user sends location ciphertexts to the request user. The distance between them can be computed homomorphically without exposing the location of the user.

The above methods can protect the users' location privacy, but there exists some problem, for example, the implementation of encryption method is expensive and the construction of location tag is difficult. Unlike these methods, our new method is simple and effective. It only needs to match the nearby reference lists of request users. Namely, users in our method can find their neighbors without using the location information of them.

Please cite this article in press as: A. Ye, et al., The flexible and privacy-preserving proximity detection in mobile social network, Future Generation Computer Systems (2016), http://dx.doi.org/10.1016/j.future.2016.12.012

Table 1
Symbol definition.

Symbol	Definition
hop	The farthest neighbor hop
uid	The user's identify
N_i	The i-hop neighbor list of user
NBs	The user's nearby reference list
NBs (U)	The nearby reference list of user U
USERs(*Be*)	The nearby user list of the beacon
N_i (U)	The i-hop neighbor list of user U

3. The privacy-preserving proximity detection method

3.1. Preliminaries

Definition 1 (*String Hash Function BKDRhash*)**.** BKDRhash function is a hash function for string handling which is the most prominent in practical effects and coding implementations. In this paper, we adopt BKDRhash function and the chain addressing hash to construct the hash table. The pseudo-code of BKDRhash function is as follows:

Function BKDRhash(key)

Input: the keyword key;
Output: hash value hash;
1) seed ←131;
2) hash← 0;
3) **While**(*key != 0)
4) hash← hash * seed + (*key++);
5) **End while**
6) **Return** (hash% (Tab.length))

In BKDRhash function, each *key* represents a string of MAC address, which serves as a keyword to create hash table.

Definition 2 (*Symbol Definition*)**.** Some symbol definitions are shown in Table 1.

If U is the set of users, N_i (U) represents the i-hop neighbor lists of all users in U.

3.2. The model of privacy and threat

In traditional proximity detection methods, users need to upload their locations to a location server so that they can find their neighbors by calculating relative distances among them. However, the locations of mobile users not only contain massive personal privacy information, but also can infer other sensitive information, such as interests, hobbies and health condition. It will cause serious privacy threaten if an adversary obtains locations of users.

Therefore, if the SNS is untrusted one, it can be compromised and controlled by an adversary who desires to steal users' locations. Moreover, we think mobile users are "honest but curious". They will follow the proposed protocol and want to know more information about other users' locations.

Our privacy security goal is to prevent the SNS from obtaining locations of mobile users. Furthermore, users can only get information about nearby users, not other extra information.

3.3. The principle of proposed method

We assume that mobile users can search for nearby beacon signals in the mobile social network. Each beacon can be identified by its ID. Motivated by this assumption, we propose a proximity detection method based on the transfer of neighbor relation. The method takes beacon signals as the references of neighbor discovery for mobile users, and users can find neighbors by determining whether the users' nearby reference lists have a common item. As show in Fig. 1, both U_1 and U_2 can receive beacon signals sent by A. Therefore, both of them have A's ID in their nearby reference lists. Since U_1 and U_2 have a common beacon item, they can be labeled as neighbors.

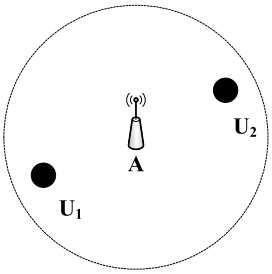


Fig. 1. Principle of proximity detection method.

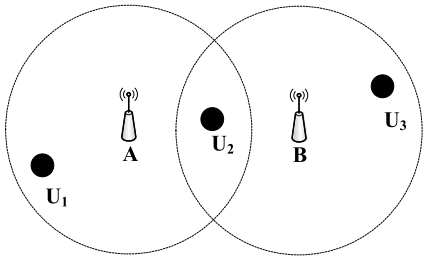


Fig. 2. The multi-hop mechanism.

In general, the nearby reference lists of users have a common item when users are in the same beacon's signal coverage, so they are neighbors. However, since the signal coverage of a single beacon node is limited, most of mobile nodes cannot find farther neighbors outside the coverage area.

Thus, we propose a multi-hop mechanism to expand the search scope of neighbors. As shown in Fig. 2, the U_1 and U_2 are neighbors because their nearby reference lists exist intersection, namely U_1 and U_2 are one-hop neighbors with each other; With the help of U_2 who has other neighbor U_3 in beacon node B's signal coverage, U_1 can search out his two-hop neighbor U_3. Through this way, we can easily expand the search scope of neighbors.

The principle of proximity detection algorithm can be formally described as follow:

Definition 3. Let *NBs* (U) = $\{Be_1, Be_2, \ldots, Be_n\}$; then through the iterative way, we can get the *x*-hop neighbors by using (1).

$$N_x\ (U) = \cup_{Be_x \in NBs(N_{x-1}(U))}\ USERs(Be_x). \tag{1}$$

Proof. Let *NBs* (U) = $\{Be_1, Be_2, \ldots, Be_n\}$; the each hop neighbors of user U is show follow:

$N_0(U) = U$;
$N_1(U) = \cup_{Be_1 \in NBs(N_0(U))}\ USERs(Be_1) = \cup_{Be_1 \in NBs\ (U)}\ USERs(Be_1)$;
.....
$N_i\ (U) = \cup_{Be_i \in NBs(N_{i-1}(U))}\ USERs(Be_i)$;
Therefore, we can get the *n*-hop neighbors as:

$$N_n\ (U) = \cup_{Be_n \in NBs(N_{n-1}(U))}\ USERs(Be_n).$$

3.4. System framework of the proposed approach

The system framework is shown in Fig. 3. When a user requests proximity detection services, the user's nearby reference lists will also be sent to the SNS. After receiving the request, the SNS matches this nearby reference list with other request users' nearby reference lists. If there is a common item in two users' nearby reference lists, these two users are neighbors. The SNS finds out all the neighbors in this manner, and the final results are returned to users.

Please cite this article in press as: A. Ye, et al., The flexible and privacy-preserving proximity detection in mobile social network, Future Generation Computer Systems (2016), http://dx.doi.org/10.1016/j.future.2016.12.012

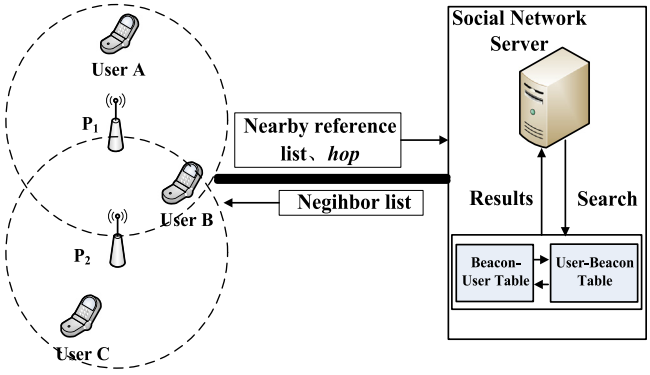


Fig. 3. System framework.

3.5. Neighbors quick search method

(1) **The overview of matching algorithm**

For the SNS, the simplest method to find neighbors is to compare nearby reference lists one by one. If nearby reference lists of two users have a common item, they will be neighbors. But with the number of users n increases, the matching number $n(n-1)/2$ will be increasing, which will leads to the degradation of service quality.

To address this problem, this paper presents a dynamic hash matching algorithm, which is shown in Fig. 4. First, the SNS needs to create two hash table, namely User–Beacon table and Beacon–User table. Both of them store users' nearby reference lists and beacons' users, respectively.

When the SNS receives the user's query message, it first traversals nearby reference list term by term and uses the beacon's ID to generate corresponding hash value *Key* through the function BKDRhash; then it accesses the Beacon–User table according to the hash value *Key*; if the result is null, it can prove that this beacon is a new neighbor discovery reference and need to insert to the Beacon–User table; otherwise, the result will be joined to the neighbor list. After that, the SNS needs to judge whether the farthest neighbor hop *hop* is greater than one; if not, it stops searching and returns the neighbor list to user; if it is true, the SNS accesses the User–Beacon table by using the previous hop neighbors' IDs to generate corresponding hash values, which can get nearby reference lists of previous hop neighbors. After that, the SNS recalculates the hash values *Keys* according to these nearby reference lists and accesses the Beacon–User table again. The results except previous hop neighbors will be joined to the neighbor list, which are the user's next hop neighbors. At the same time, the value of *hop* minus one. The above iterative process will be continuous until the *hop* is less one.

(2) **The storage structure of hash tables**

The SNS utilizes hash table to store users' nearby reference lists, named User–Beacon table. By combining each user's ID and BKDRhash, the SNS can generate a hash value *Key*. Then, the SNS stores each user's nearby reference list in corresponding storage space depends on the value of *Key*. The storage structure of User–Beacon table is shown in Fig. 5.

The Beacon–User Table is also created by hash table. Each beacon's ID in user's nearby reference list is a keyword to generate a hash value *Key*. The SNS stores each beacon's nearby users (i.e., the users who can receive this beacon's signal) in corresponding storage space according to the value of *Key*. The storage structure of Beacon–User table is shown in Fig. 5. If the beacon item has already in Beacon–User table and the current user does not exist under this beacon item, the user will be added into this item; if the beacon item is not in the hash table, the server create a new beacon item in the hash table. The users under each beacon item are neighbors.

(3) **One-hop matching algorithm**

When the SNS searches one-hop neighbors, it accesses the Beacon–User table to obtain the neighbors by calculating the hash values according to the request user's nearby reference list. The specific neighbor discovery process is shown in Algorithm 1.

Algorithm 1 **OneHopNearbySearch**(uid, NBs)
Input: the nearby reference list of user NBs, user ID uid;
Output: one-hop neighbor list N_1.
1) **for** i←1 **to** NBs.length
2) Key←BKDRhash(NBs[i])
3) **if** (hash[Key] != null) then
4) N_1←hash[Key]
5) **end if**
6) **if** (HashNotExist(uid))
7) hash[Key] ←uid
8) **else** insertHash(hash, Key, uid)
9) **end if**
10) **end for**
11) **Return** (N_1)

As shown in Fig. 5, the user U_1 can search for nearby P_1, therefore, he or she uses P_1 as a neighbor discovery reference to send a query message to the SNS; the server calculates the hash value Key (P_1) and finds P_1's neighbor list is $\{U_1, U_2, U_3\}$ according to the Beacon–User table, and then takes $\{U_2, U_3\}$ as a result back to U_1.

(4) **Multi-hop matching algorithm**

When the value of *hop* is greater than one, the SNS needs to search multi-hop neighbors. At the beginning, the server needs to find nearby reference lists of the previous hop neighbors from User–Beacon table. Then, according to these nearby reference lists, the server accesses the Beacon–User table to get the *hop*-hop neighbors. The specific process is shown in Algorithm 2:

Algorithm 2 **MultiHopNearbySearch**(hop, N_1)
Input: one-hop neighbor list of request user N_1, the farthest neighbor hop hop;
Output: multi-hop neighbor list $N_{m\text{-}h}$.
1) **for** h←1 to (hop-1)
2) **for** i←1 to N_h.length
3) NB←N_h[i].NBs
4) **for** j ←1 to NB.length
5) Key←BKDRhash (NB[j])
6) **if** (hash[Key] != null) then
7) N_{h+1}←hash[Key]
8) **end if**
9) **end for**
10) **end for**
11) **end for**
12) $N_{m\text{-}h}$←{$N_1 \cup N_2 \ldots \cup N_{hop}$}
13) **Return** ($N_{m\text{-}h}$)

As shown in Fig. 5, assume that the farthest neighbor hop *hop* of request user U_1 is 2, then the server obtains the one-hop neighbor list $\{U_2, U_3\}$ by the one-hop matching algorithm. After that, the server gets the one-hop neighbors' nearby reference list $\{P_1, P_2\}$ finding both P_1 and P_2's one-hop neighbor list $\{U_1, U_2, U_3, U_4, U_5\}$ in Beacon–User table according to the hash value Key (P_1) and Key (P_2); Finally, the server returns $\{U_2, U_3, U_4, U_5\}$ as the result to U_1.

(5) **The calculating of neighbor distance**

In the above steps, we can calculate the Hop distance among mobile users. However, the Euclidean distance between each two users is necessary to calculate in the proximity detection. Therefore, here we mainly discuss how to transform the Hop distance to Euclidean distance. In the presented method, the distance between two users can be divided into the addition of multiple Euclidean distances, which is shown in Fig. 6.

Please cite this article in press as: A. Ye, et al., The flexible and privacy-preserving proximity detection in mobile social network, Future Generation Computer Systems (2016), http://dx.doi.org/10.1016/j.future.2016.12.012

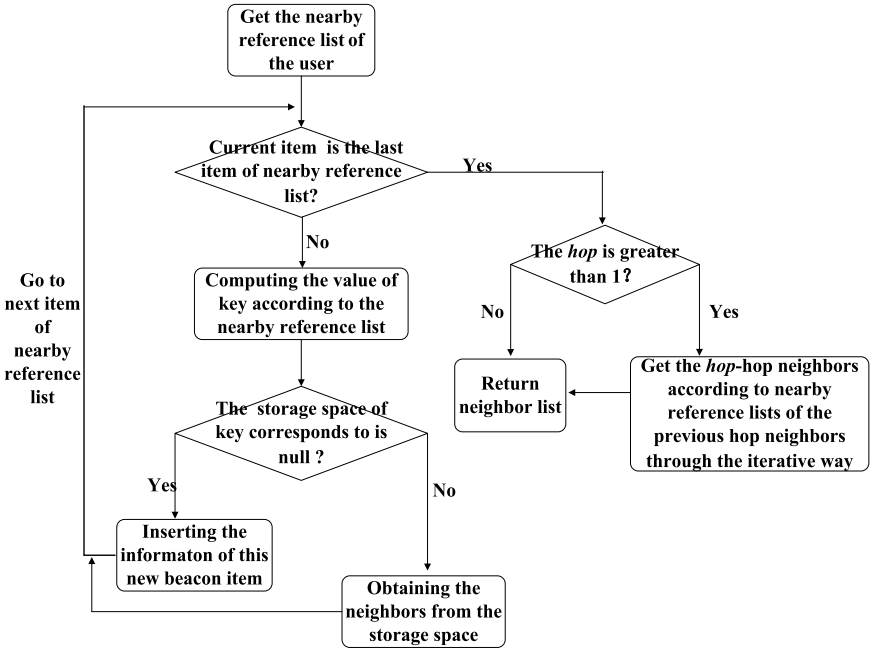


Fig. 4. The flow chart of matching algorithm.

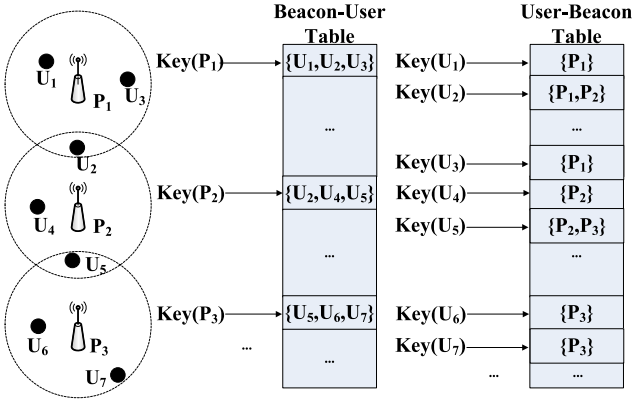


Fig. 5. The storage structure of hash tables.

For example, the distance between user U_1 and his neighbors U_2 is consisting of ud_1, ud_2 and the addition of pd_1 to pd_{n-1}, namely the distance of them can be computed by formula (2).

$$dis = ud_1 + \sum_{i=1}^{n-1} pd_i + ud_2. \tag{2}$$

(1) The calculating of the User–Beacon distance

Since each user can receive surrounding beacon signals, the distance between the user and beacon can be calculated by the wireless signal propagation attenuation model. The stronger of the received signal, the smaller of the distance between the user and the beacon, and the greater of the distance conversely. Therefore, we use the Shannon-logarithmic model to calculate the distance, which is shown in formula (3):

$$\mathrm{PL}\,(\mathrm{dB}) = PL\,(d_0)\,(\mathrm{dB}) + 10\gamma \log_{10}\left(\frac{d}{d_0}\right) + X_\sigma\,(\mathrm{dB}) \tag{3}$$

where, the d represents the distance between the receiver (i.e. the user) and transmitter (i.e. the beacon), the d_0 is the reference distance, which is fixed as a constant 1 m, the PL indicates the receive signal power of the receiver, the X_σ is the normal random variable of the standard deviation, the γ is the path loss exponent and represents the rate of path loss with the growth of distance, which depends on the ambient and building types. In open environment, the value of γ is 2. Therefore, the distance between the user and beacon can be calculated by using formula (4):

$$d = d_0 10^{\frac{\mathrm{PL(dB)} - \mathrm{PL}(d_0)}{10\gamma}}. \tag{4}$$

(2) The calculating of the Beacon–Beacon distance

As shown in Fig. 7, if the signal intersection area of two beacons is larger, the distance between two beacons will be smaller. Suppose the mobile users span evenly in the signal coverage area of a beacon, and then both the area and the number of mobile users covered by a beacon have a linear relationship. If the signal intersection area of two beacons is larger, the number of users in this area will also be larger. Therefore, the distance of two beacons

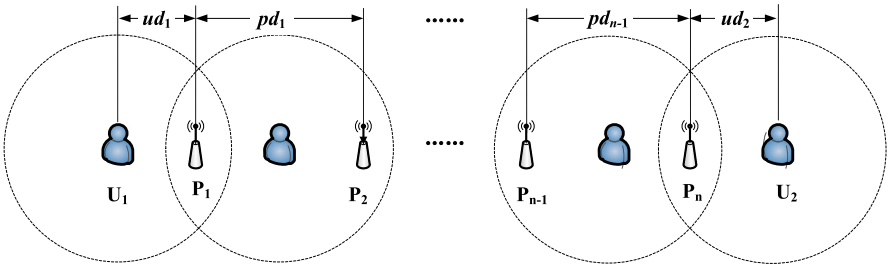


Fig. 6. The calculating of neighbor distance.

Please cite this article in press as: A. Ye, et al., The flexible and privacy-preserving proximity detection in mobile social network, Future Generation Computer Systems (2016), http://dx.doi.org/10.1016/j.future.2016.12.012

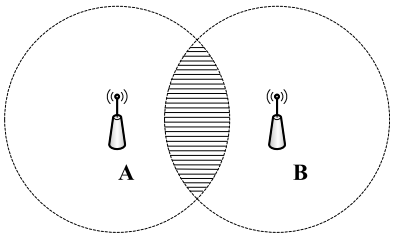


Fig. 7. The calculating of Beacon–Beacon distance.

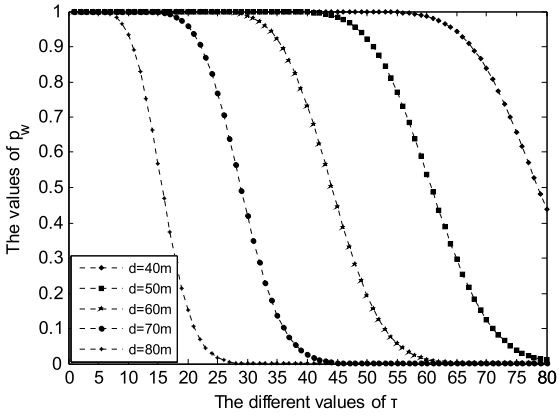


Fig. 8. The function figure of the Poisson formula.

can be divided into three levels according to the number of the user n, which is shown as follow:

$$\begin{cases} n < \tau_1 : & 90\text{ m} \\ \tau_1 < n < \tau_2 : & 60\text{ m} \\ n > \tau_2 : & 30\text{ m} \end{cases}$$

where, the τ_1 and τ_2 are two thresholds of n and they meet $0 < \tau_1 < \tau_2$. When $n < \tau_1$, the distance of two beacons is 90 m; when $\tau_1 < n < \tau_2$, the distance is 60 m and the distance is 30 m when $n > \tau_2$.

For the suitable values of τ_1 and τ_2, we use the Poisson formula (5) (6) to analyze these two parameters by changing the distance of two beacons under the different density of mobile users. The Poisson formulas are shown as follows:

$$p_w = p\left(|n| > \tau\right) = 1 - \sum_{i=0}^{\tau} \frac{(p_b \times A)^i}{i!} e^{-(p_b \times A)} \tag{5}$$

$$A = 2\left[r^2 \arccos\left(\frac{d}{r}\right) - d\sqrt{(r^2 - d^2)}\right] \tag{6}$$

where, the r is the radius of a beacon's signal coverage, the p_b is the density of the mobile user, A is the intersection area of two beacons' signal coverage, the p_w is the probability that n (the number of the user in A) greater than τ.

The Fig. 8 is the function figure of the Poisson formula. From the figure, we can see that when the value of d is increasing, the p_w increases and the p_w decreases when the value of τ is increasing. Therefore, we can get the desired value of τ by setting the proper value of p_w and d. For instance, we think the $p_w \geq 0.9$ will meet our needs, then when $d = 50$ m, we can set $\tau = 55$ or when $d = 70$ m, the $\tau = 27$.

Here, we set the p_b equals 0.005 and 0.01, respectively and the value of d equals 30 m, 60 m and 90 m, respectively. Moreover, we think when $p_w \geq 0.9$, the value of τ is the best one. The final results are shown in Fig. 9. From the Fig. 9(a), we can find that when $p_w \geq 0.9$, we can set $\tau_1 = 2$ and $\tau_2 = 35$ while from the Fig. 9(b), we can see that when $p_w \geq 0.9$, we can set $\tau_1 = 10$ and $\tau_2 = 75$. Thus, we can get the conclusion that the values of τ_1 and τ_2 are different when the density of mobile users changes.

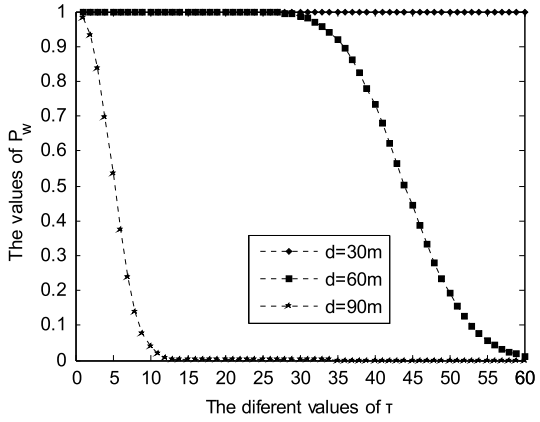
(6) **The value of *TTL***

Since the user has the mobility (i.e., its positional relationship with the beacon is dynamic), all users' neighbor relationship is dynamic. Therefore, each user in Beacon–User table needs to have a survival time—*TTL*. When the user's survival time *TTL* changes into zero, the server automatically removes the user out the Beacon–User table, and also deletes the user's nearby reference list from the User–Beacon table.

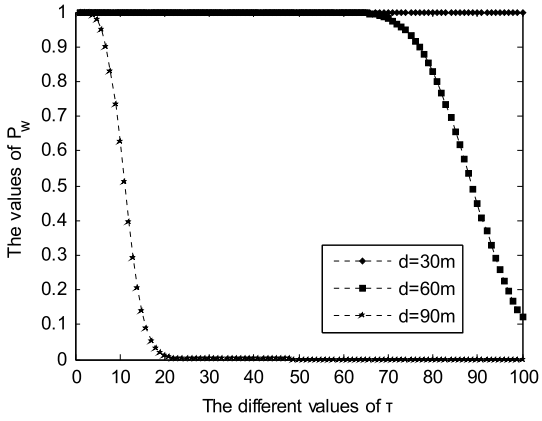
However, the user's survival time *TTL* will make the correct rate of searching neighbors decreased. For example, the user A has already moved out the beacon node Q's signal coverage before his or her survival time come to zero. Due to the *TTL*, A is considered still in the Q's signal coverage. In the meanwhile, if the new user B's nearby reference list contains the item of beacon node Q, A will be the B's neighbor.

In this paper, the correct rate of searching neighbors can be calculated by the formula $P_a = \frac{\text{Nearby}_{\text{true}}}{\text{Nearby}_{\text{total}}} \times 100\%$, where the $\text{Nearby}_{\text{true}}$ represents the number of each user's true neighbors, the $\text{Nearby}_{\text{total}}$ represents the number of searched neighbors. Therefore, the $1 - P_a$ is the error rate of searching neighbors.

In order to solve this problem, we need to select a suitable value for the user's survival time *TTL*. There are two factors can affect the value of *TTL*: the signal coverage of the beacon and the speed of the user. In generally, if the range of the beacon's signal coverage is larger, the value of *TTL* is also bigger. However, this factor not only depends on the beacon itself, but also can be affected by the obstacle, so we only consider the other factor in this paper. If the speed of the user is faster, the time a user stay in a beacon's signal



(a) The values of p_w with different values of d when $P_b = 0.005$.



(b) The values of p_w with different values of d when $P_b = 0.01$.

Fig. 9. The best values of τ_1 and τ_2.

Please cite this article in press as: A. Ye, et al., The flexible and privacy-preserving proximity detection in mobile social network, Future Generation Computer Systems (2016), http://dx.doi.org/10.1016/j.future.2016.12.012

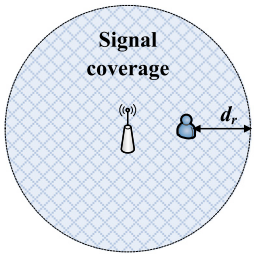


Fig. 10. The range of *TTL* value.

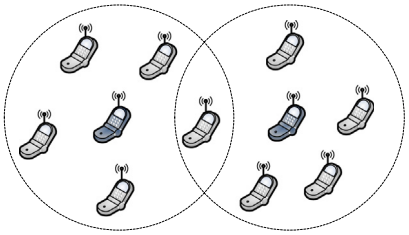


Fig. 11. The principle of signal transmission mechanisms of beacon nodes.

coverage is shorter, namely the value of *TTL* is smaller. Therefore, the value of *TTL* can be calculated theoretically by using (7):

$$TTL = \frac{d_r}{v_{\max}} \tag{7}$$

where, the d_r is the distance between the user and signal boundary of the beacon, like the Fig. 10 shown; the $V_{\max}$ is the maximum speed of user.

3.6. The signal transmission mechanisms of beacon nodes

In our proximity detection method, the mobile device (mobile user) can act as a beacon node, as shown in Fig. 11. Each mobile user upgrades to be a beacon node (the wireless network adapter has three modes: master, monitor and managed. The master mode can make the mobile device become a wireless access point, namely it can act as a beacon node and the MAC address of the mobile device is its ID) and periodically sends a beacon signal. Then other users can find neighbors by making use of the beacon signals (i.e., the references). The SNS discovers neighbor relations among users by determining whether their nearby reference lists (the mobile device of each request user is also a beacon node, so every list should include the beacon node's ID of request user himself) have a common item.

Moreover, we have proposed two novel mechanisms – Beacon Node Rotating Mechanism and Beacon Node Competition Mechanism – to determine the manners of transmitting signals of beacon nodes (mobile users).

(1) **Beacon node rotating algorithm**

In this algorithm, the time of each beacon node transmits signal is fixed. However, the value of waiting time is calculated by the follow algorithm:

(1) Determine the surrounding node degree K. Assume the number of nodes located near one beacon node is Nu, then the surrounding node degree of this beacon node needs to satisfy $K = \lceil \log\_2(Nu + 1) \rceil$.
(2) The beacon node selects an integer r randomly, where $r \in \{0, 1 \ldots 2^K - 1\}$.
(3) According to the value of r, one can calculate waiting time $T = r * t$ (t is the minimum waiting time).

Nevertheless, the waiting time of transmitting signal depends on the surrounding node density varies. If the density of surrounding nodes is higher, the value of K is bigger. Then the value range of the integer r will be larger, which makes each beacon node has a higher probability to choose the bigger one and the waiting time of each beacon node will become longer. Finally, the time for each beacon node to transmit signal will be reduced. Namely the number of beacon nodes will be smaller.

In Beacon Node Rotating algorithm, each beacon node initially has two states:

(1) Each beacon node monitors whether there exists signals from other beacon nodes; if it is true, the beacon node changes into the waiting state. Until the waiting time is 0, the beacon node comes to transmit signal state;
(2) If the beacon node cannot receive signal sent from other beacon nodes, then it immediately transmits signals when the time of transmitting signal becomes zero, the beacon node changes into waiting state and calculates the waiting time according to the above algorithm.

After the initial state is determined, the state of each beacon node is transformed between the transmitting state and waiting state, alternately.

(2) **Beacon node competition mechanism**

This mechanism is based on the P-CSMA [21] algorithm. The P-CSMA algorithm is mainly focus on avoiding "collision", but our goal is to decline the energy loss by reducing the number of beacon nodes as much as possible when we detect proximity. Therefore, the mechanism is as follows:

(1) Each beacon node monitors whether there exists signals from other beacon nodes; If not, each beacon node transmits signal with a probability of P (to wait a unit time t_1 with a probability $1 - P$), and the time of each beacon node transmits signal is t_2;
(2) After waiting a unit time, the beacon node comes back to the step 1.
(3) If it is true, the beacon node keeps monitoring until it does not receive any signals from other beacon nodes.

Furthermore, the probability P is computed as follows:

$$P = \frac{E_{final}}{L \times C} \times \varepsilon \tag{8}$$

where, the E_{final} represents the final energy of the beacon node; the L represents the number of surrounding beacon nodes; the C represents the times of transmitting signals; the ε is a weighted coefficient, which ensures the value of P meets $0 < P < 1$.

When a beacon node does not receive any signals, it calculates the value of P by using (8); if P is greater than the threshold Th, the beacon node can transmit signals, which is used as a reference for others to find neighbors. From (8), we also can conclude: if the value of E_{final} is greater and both of the L and C are less, the probability of transmitting signals is greater. Therefore, this method can save energy resources and be effective to achieve the load balance of system energy. However, compared with the beacon rotating algorithm, the signal coverage of this algorithm is lower.

4. Security analysis and discussions

4.1. Security analysis

The top priority of our design is to prevent the SNS from obtaining locations of mobile users. Furthermore, users can only get information about nearby users without extra information. As a result, the security analysis is discussed based on this priority.

Please cite this article in press as: A. Ye, et al., The flexible and privacy-preserving proximity detection in mobile social network, Future Generation Computer Systems (2016), http://dx.doi.org/10.1016/j.future.2016.12.012

Table 2
Privacy-preserving proximity detection methods.

Solutions	Security principle	System structure	Threat object	Security level	System overhead
Friend-locator [8] Ref. [9] Vicinity locator [10] Ref. [11]	Grid dividing	Centralization	Server	1. Protect location privacy of users from server; 2. Cannot resist collusion attack.	Large communication and computation overhead;
InnerCircle [18]	Homomorphic encryption	Distribution	Users	Protect location privacy among users;	Expensive cost of encryption method;
Ref. [12]	Locationtag	Distribution	Users	1. Protect location privacy among users; 2. Can resist location cheating.	Large communication overhead;
Ref. [13] Ref. [14]		Centralization	Server, Users	1. Protect location privacy among users; 2. Resist location cheating; 3. Cannot resist collusion attack.	
Hide & Crypt [17]	Location cloaking	Centralization	Server	Protect location privacy of users from server;	
Ref. [15]	Coordinate transformation			1. Protect location privacy of users from server; 2. Cannot resist collusion attack.	Large computation overhead;
Our method	The transfer of neighbor relation	Centralization	Server	1. Protect location privacy of users from server; 2. Cannot resist collusion attack.	

Here, when users search neighbors, they send nearby reference lists instead of their locations to the SNS. Moreover, since users have the mobility, namely their positional relationship with the beacon is dynamic. Therefore, all users' neighbor relationship is dynamic. Although the SNS can obtain the ID of each beacon, even get some extra information about beacons from other ways, it still cannot infer location information of users from these information.

Nowadays, many Location-based services adopt Wifi (AP) to achieve locating, which make the malicious user can cheat locations by forging corresponding AP [22]. In our method, due to the mobility of users, mobile devices act as beacon nodes also have the mobility, namely beacon nodes cannot be mapped into the specific environment. Therefore, the malicious user cannot achieve location cheating. For users, they only can get information about nearby users from their mobile devices without other information.

Unfortunately, our method cannot resist collusion attack. If a malicious user colludes with the SNS, he will send his location and nearby reference list to SNS. It will cause location exposure if there exist some users whose nearby reference lists have a common item with malicious user's one. However, this collusion attack only exposes locations of users who are nearby with malicious user. The SNS still cannot obtain all users' locations. What is more, according to the above analysis, we can know positional relationships of users with beacons are dynamic, so the SNS may not be able to get the correct location information of users.

4.2. Discussions

As shown in Table 2, recently, researchers have proposed many privacy-preserving proximity detection methods. According to the system structure, these methods can be divided into centralized and distributed structure.

For methods of centralized structure, their mainly threat object is server, namely the server cannot get locations of users. From the table, we can find that most of methods about centralized structure, including our method, cannot resist collusion attack. For example, for the methods adopted grid dividing, they request all users to share a common grid. Once a malicious user collude with the server, the server can know the way of dividing grid and maps each cell ID of grid into the location of each user. There is a common problem with the methods of coordinate transformation. All users need to share a common secret key and coordinate transformation function. By taking the collusion attack, the server can obtain locations of all users by using the secret key and coordinate transformation function. For these methods, the influence of the collusion is global. However, compared with these methods, though our method cannot resist the collusion attack either, its influence is regional. In other words, only those users who are close to the malicious user will expose their locations. Moreover, since the positional relationships of users with beacons are dynamic, the server may not be able to get the correct location information of users. Therefore, under the collusion attack, our method is safer than others.

For methods of distributed structure, their privacy guarantee mostly depends on encryption schemes. However, most encryption schemes cost expensively and their implements are difficult and complicated.

Therefore, our method is simple and effective compared with existing methods. For users, they only need to send nearby reference lists to the SNS; for the SNS, it can find neighbors by judging whether these lists have a common item. Moreover, the mobility of users has immensely improved the security of our method, so our method can protect location privacy of users.

5. Experiment

We have implemented an experimental system based on the presented method in Section 3. Specifically, the client is implemented in JAVA on an ANDROID 3.0 smartphone. The SNS is deployed on a third-party cloud hosting services, provided by the AliCloud. The interface of APP succeeds to find neighbors is shown in Fig. 12. Moreover, we also experimentally analyze some important parameters in our method, which can guarantee the effectiveness and feasibility of the proposed method.

5.1. Experimental analysis of TTL value

In order to obtain the suitable value of *TTL*, we have already measured that one beacon's signal coverage is about 100 m. During the experiment, we research the correct rate of searching

Please cite this article in press as: A. Ye, et al., The flexible and privacy-preserving proximity detection in mobile social network, Future Generation Computer Systems (2016), http://dx.doi.org/10.1016/j.future.2016.12.012

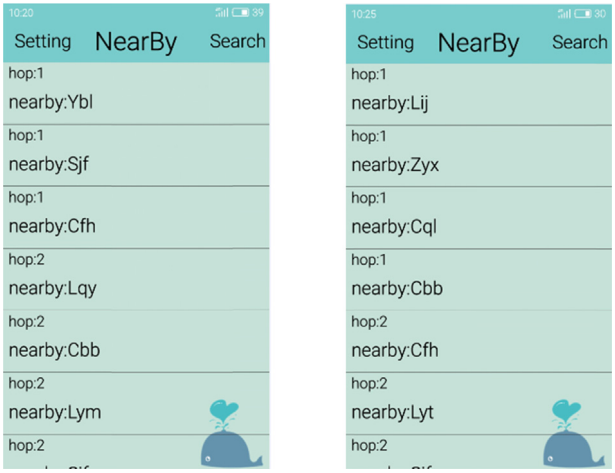


Fig. 12. The example of neighbor discovery on APP.

neighbors by changing the value of *TTL* under the different ranges of mobile speed. The result is shown in Fig. 13 and we can find that with the increasing of *TTL*, the correct rate of searching neighbors is declined and the error rate is raised. When both of rates have an intersection point, this point is the proper value of *TLL*.

Through this experiment method, we can obtain proper values of *TTL* on different ranges of speed. At the same time, we have compared this result with the result calculated by using formula (7), like the Fig. 14 shown. The experiment result is close to the theoretical one.

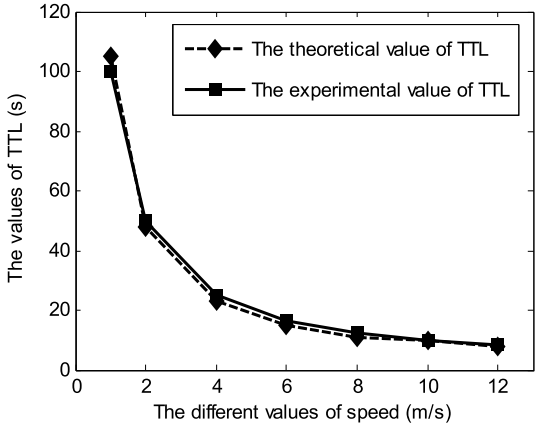
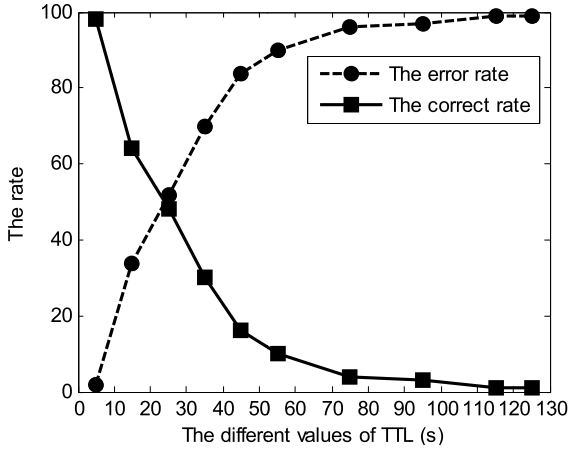


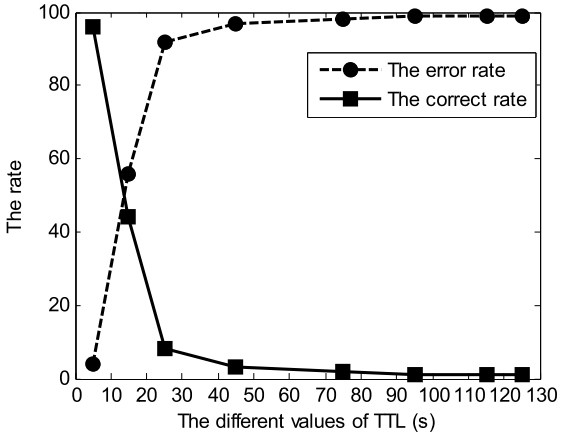
Fig. 14. The value of *TTL* in experiment or theoretical result.

5.2. Beacon nodes rotating algorithm

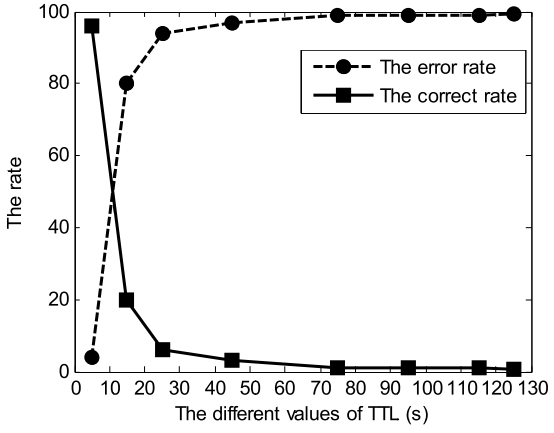
As we know, the more beacon nodes exist, the more signals mobile users can receive. However, if the number of beacon nodes increases, more energies will be consumed by this algorithm. Therefore, in order to get the suitable time of transmitting signals (i.e., the number of beacon nodes), we research the rate of signal



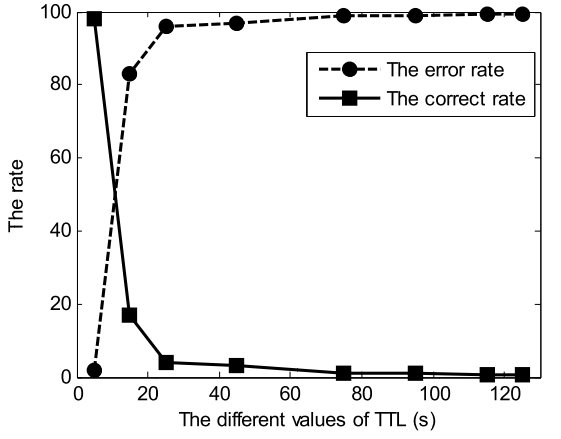
(a) The range of mobile speed is 2–4 m/s.



(b) The range of mobile speed is 4–6 m/s.



(c) The range of mobile speed is 6–8 m/s.



(d) The range of mobile speed is 8–10 m/s.

Fig. 13. The suitable values of *TTL* on different ranges of mobile speed.

Please cite this article in press as: A. Ye, et al., The flexible and privacy-preserving proximity detection in mobile social network, Future Generation Computer Systems (2016), http://dx.doi.org/10.1016/j.future.2016.12.012

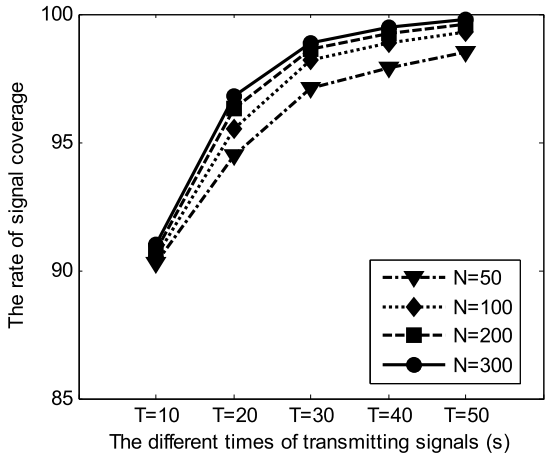


Fig. 15. The rate of signal coverage with different values of mobile node.

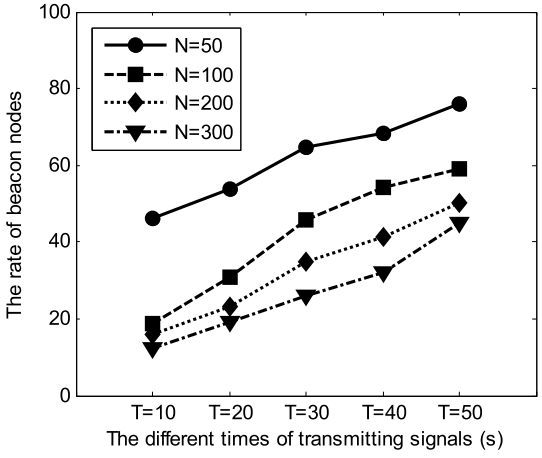


Fig. 16. The number of beacon nodes with different values of mobile node.

coverage by changing the time of transmitting signals, which under the different number of mobile users. The result is shown in Fig. 15. When the time of transmitting signals T is greater than 30 s, the rate of signal coverage is close to 100%, but the number of beacon nodes is still increasing, as shown in Fig. 16. The rate of beacon nodes can be calculated by the formula $P_b = \frac{N_{Beacon}}{MN_{total}}$ where the N_{Beacon} represents the number of beacon nodes, the MN_{total} represent the all of the mobile users. Therefore, the best value range for the time of transmitting signals is 30–40 s.

For the purpose of verifying the validity of the above experimental result, the rate of signal coverage by changing the number of mobile nodes is researched again. This time, we choose the time of transmitting signals is 30 s. Fig. 17 shows the final result. We can discover that the rate of signal coverage has remained above 96%, and with the increase of the number of mobile users, the number of beacon nodes is less. So, the 30–40 s is the suitable value range for the time of transmitting signals.

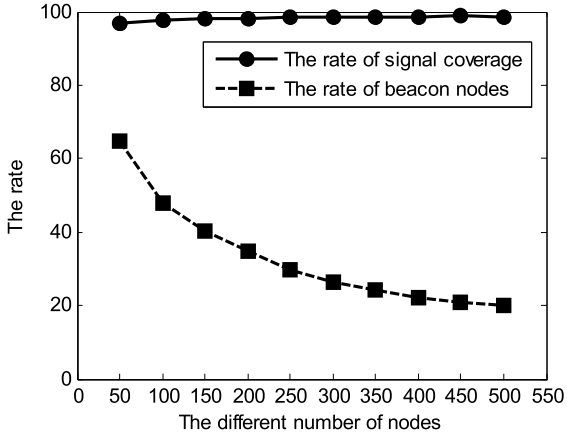
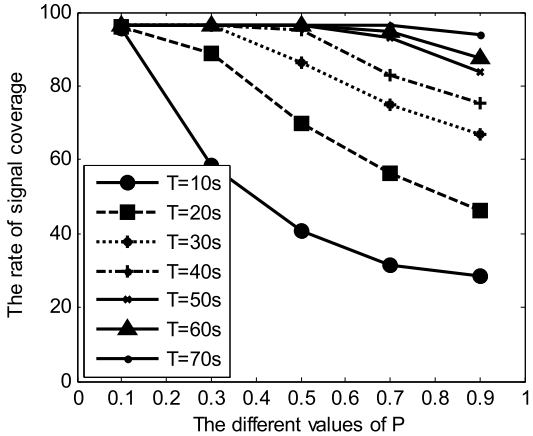


Fig. 17. The experiment result with the time of transmitting signals is 30 s.

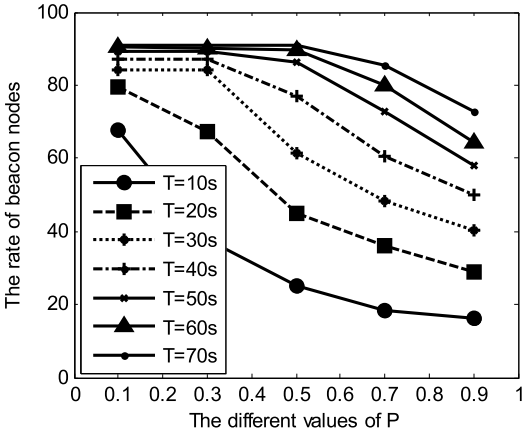
5.3. Beacon node competition algorithm

The same to the Beacon Node Rotating method, this method needs to analyze these two important factors: the signal coverage and energy loss. But compared with Beacon Node Rotating method, this method not only needs to get a suitable time for transmitting signals, but also needs to obtain a proper value of P (threshold Th). So, we research the rate of signal coverage by changing the time of transmitting signals and the value of P respectively, which under the different number of mobile users. The result is shown in Figs. 18–19. From the result, we can conclude that when the value of P is between 0.1–0.5 and the time of transmitting signals T is 40 s or more, the rate of signal coverage comes to a large value. At this time, the number of beacon node should be as small as possible. Therefore, we think that the best value range for the time of transmitting signals and the P are 30–40 s and 0.4–0.5, respectively.

In order to test the correctness of the above experimental results, we research the rate of signal coverage again by changing



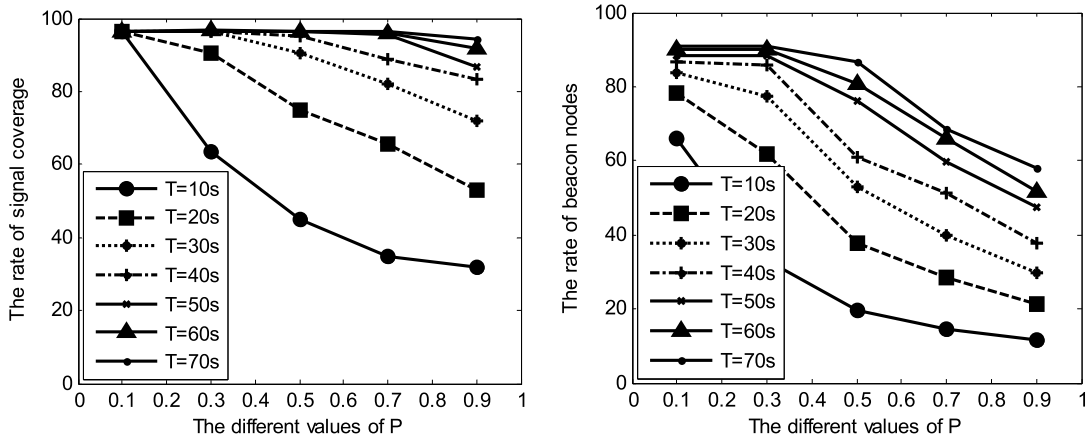
(a) The rate of signal coverage in different value of P and T.



(b) The rate of beacon node in different value of P and T.

Fig. 18. The result on the number of mobile node $N = 50$.

Please cite this article in press as: A. Ye, et al., The flexible and privacy-preserving proximity detection in mobile social network, Future Generation Computer Systems (2016), http://dx.doi.org/10.1016/j.future.2016.12.012



(a) The rate of signal coverage in different value of P and T.

(b) The rate of beacon nodes in different value of P and T.

Fig. 19. The result on the number of mobile node $N = 100$.

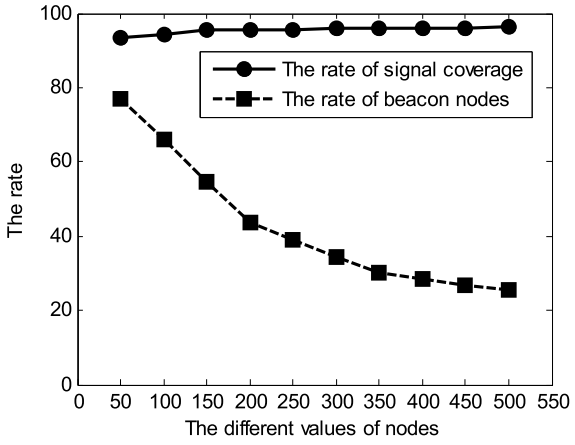


Fig. 20. The rate of signal coverage when $T = 40$ s and $P = 0.5$.

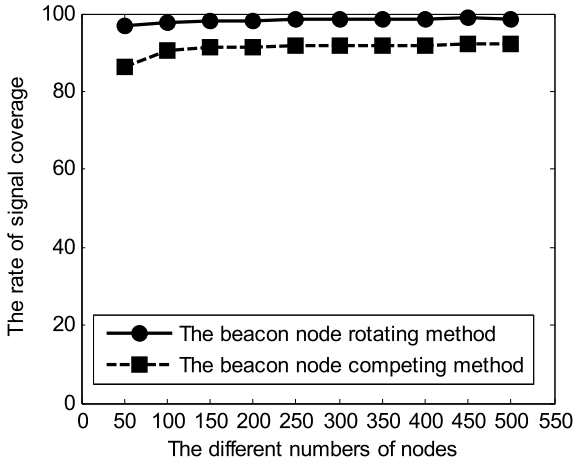


Fig. 21. The rates of signal coverage of two mechanisms when $T = 30$ s.

the number of mobile users. In this scenario, we choose the time of transmitting signals is 40 s and the value of P is 0.5. Fig. 20 shows the final result. We can find that the rate of signal coverage has remained above 95%, and with the increase of the number of mobile users, the number of beacon nodes is less. So, both the 30–40 s and 0.4–0.5 are the suitable value ranges for the time of transmitting signals and the P, respectively.

5.4. The comparative analysis of two mechanisms

We have compared these two mechanisms in both signal coverage and energy loss. The Fig. 21 is the rate of signal coverage of these two mechanisms under the different number of mobile nodes. In this experiment, both of them set the time of transmitting signals are 30 s. From the figure, we can conclude that compared with the Beacon Node Rotating mechanism, the beacon node competing mechanism has a lower rate of signal coverage when the energy losses of them are the same.

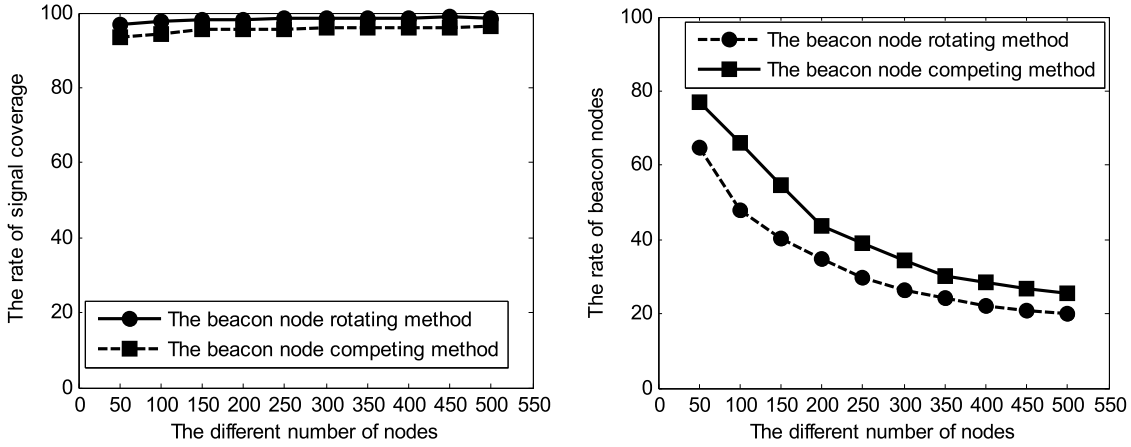
For our proximity detection can be feasible, there must have a high rate of signal coverage. So, the beacon node competing mechanism needs to increase the time of transmitting signals to ensure high signal coverage. The compared results of these two mechanisms are shown in Fig. 22 (when they get a suitable value of transmitting signals or P). We can discover that the beacon node competing mechanism has more energy loss than the Beacon Node Rotating mechanism when they have a close rate of signal coverage.

6. Conclusions

In order to overcome the disadvantages of traditional proximity detection method, we propose a novel privacy-preserving proximity detection method which is based on the transfer of neighbor relation. The SNS discovers neighbor relationships among users by computing users' nearby reference lists (i.e., by checking whether they have a common item). In addition, we also propose another two methods—Beacon Node Rotating Mechanism and Beacon Node Competition Mechanism to determine the ways of transmitting signals of mobile nodes. Specifically, in the Beacon Node Rotating Mechanism, each mobile node acts as a beacon node and periodically sends out a beacon signal as a neighbor discovery reference to other nodes. However, in the Beacon Node Competition Mechanism, each mobile node competes to become a beacon node and transmits signal with a probability of P. The proximity detection mechanism we have proposed without using the location information of users, so it well improves the security of location privacy.

In our future work, we plan to implement our methods in broader scenarios. For example, the beacon nodes can transmit signals based on the Bluetooth, RFID. Moreover, we plan to research and design a more effective proximity detection mechanism, which cannot only decline the energy loss as much as possible, but also improve the rate of the signal coverage.

Please cite this article in press as: A. Ye, et al., The flexible and privacy-preserving proximity detection in mobile social network, Future Generation Computer Systems (2016), http://dx.doi.org/10.1016/j.future.2016.12.012



(a) The rates of signal coverage of two mechanisms.

(b) The rates of beacon node of two mechanisms.

Fig. 22. The compare results of two mechanisms.

Acknowledgments

This work is supported partially by National Natural Science Foundation of China (61202452); the Joint Funds of the National Natural Science Foundation of China (U1405255); the 2015 Science and Technology Projects of Fuzhou (2015-G-51); and the key project of Fujian Provincial Department of Science and Technology (2014H0018).

References

[1] M. Hennig, S.P. Borgatti, L. Krempel, et al. Studying Social Networks: A Guide to Empirical Research, Studying social networks: a guide to empirical research. Campus, 2013.
[2] B. Viswanath, A. Mislove, M. Cha, et al. On the evolution of user interaction in Facebook, in: Proceedings of the ACM Workshop on Online Social Networks, 39, (4), 2009, pp. 37–42.
[3] H. Kwak, C. Lee, H. Park, et al., What is Twitter, a social network or a news media? in: Proceedings of the 19th International Conference on World Wide Web, ACM, 2010, pp. 591–600.
[4] L. Atzori, A. Iera, G. Morabito, The Internet of things: A survey, Comput. Netw. 54 (15) (2010) 2787–2805.
[5] A.R. Beresford, F. Stajano, Mix zones: User privacy in location-aware services, in: Proceedings of the Second IEEE Conference on Pervasive Computing and Communications Workshops, 2004, IEEE, 2004, p. 127.
[6] M. Gruteser, D. Grunwald, Anonymous usage of location-based services through spatial and temporal cloaking, in: International Conference on Mobile Systems, Applications, and Services, 2003, pp. 31–42.
[7] C.A. Ardagna, M. Cremonini, E. Damiani, et al., Location privacy protection through obfuscation-based techniques, in: Data and Applications Security XXI, Springer, Berlin, Heidelberg, 2007, pp. 47–60.
[8] Laurynas Šikšnys, J.R. Thomsen, Simonas Šaltenis, et al., A location privacy aware friend locator, in: Proceedings of the 11th International Symposium on Advances in Spatial and Temporal Databases, Springer-Verlag, 2009, pp. 405–410.
[9] H.P. Li, H. Hu, J. Xu, Nearby friend alert: Location anonymity in mobile geosocial networks, IEEE Pervasive Comput. 12 (4) (2013) 62–70.
[10] Laurynas Šikšnys, Private and flexible proximity detection in mobile social networks, in: Eleventh International Conference on Mobile Data Management, IEEE, 2010, pp. 52–62.
[11] C. Huang, R. Lu, H. Zhu, J. Shao, A. Alamer, X. Lin, EPPD: Efficient and privacy-preserving proximity testing with differential privacy techniques, in: 2016 IEEE International Conference on Communications, ICC, Kuala Lumpur, 2016, pp. 1–6.
[12] G. Zhuo, Q. Jia, L. Guo, et al. Privacy-preserving verifiable proximity test for location-based services, in: GLOBECOM 2015–2015 IEEE Global Communications Conference, 2014.
[13] Y. Zheng, M. Li, W. Lou, T. Hou, Location based handshake and private proximity test with location tags, IEEE Trans. Dependable Secure Comput. PP (99) (2015) 1.
[14] Y. Zheng, M. Li, W. Lou, et al. SHARP: Private proximity test and secure handshake with cheat-proof location tags, in: ESORICS, 2012, pp. 361–378.
[15] R. Rajadurai, R. Suganyaa, V. Santhakumari, et al., Proximity based security for location based services, in: International Conference on Advanced Research in Computer Science Engineering & Technology, ACM, 2015, pp. 1–5.
[16] P. Ruppel, G. Treu, A. Küpper, et al. Anonymous user tracking for location-based community services, in: Location- and Context-Awareness, Second International Workshop, LoCA 2006, Dublin, Ireland, May 10–11, 2006, Proceedings, 2006, pp. 116–133.
[17] C.G. Kun Liu, H. Kargupta, An attacker's view of distance preserving maps for privacy preserving data mining, J. Biol. Chem. 270 (18) (1995) 10658–10663.
[18] S. Mascetti, C. Bettini, D. Freni, et al. Privacy-aware proximity based services, in: MDM 2009, Tenth International Conference on Mobile Data Management, Taipei, Taiwan, 18–20 May, 2009, pp. 31–40.
[19] P. Hallgren, M. Ochoa, A. Sabelfeld, InnerCircle: A parallelizable decentralized privacy-preserving location proximity protocol, IEEE Priv. Secur. Trust (2015) 1–6.
[20] Peizhao Hu, T. Mukherjee, A. Valliappan, S. Radziszowski, Homomorphic proximity computation in geosocial networks, in: 2016 IEEE Conference on Computer Communications Workshops, INFOCOM WKSHPS, San Francisco, CA, 2016, pp. 616–621.
[21] R. Bruno, M. Conti, E. Gregori, Optimal capacity of p-persistent CSMA protocols, IEEE Commun. Lett. 7 (3) (2003) 139–141.
[22] N.O. Tippenhauer, K.B. Rasmussen, C. Pöpper, et al. Attacks on public WLAN-based positioning systems, in: International Conference on Mobile Systems, Applications, and Services, 2009, pp. 29–40.

Ayong Ye Ph.D., Associate Professor; Member of China Computer Federation etc. He got Ph.D. degree in computer software and communications engineering from Xidian University (Xi'an) in 2009. Now he is conducting information privacy and security research in the Key Lab of Network Security and Cryptology, Fujian Normal University, China.

Qiuling Chen Now she is pursuing the B.S. degree in Key Lab of Network Security and Cryptology, Fujian Normal University, China.

Li Xu Ph.D., tutor for Ph.D. students, he received his B.S. and M.S. degrees from Fujian Normal University in 1992 and 2001, respectively. He received his Ph.D. degree from Nanjing University of Posts and Telecommunications, in 2004. He is a Professor and Doctoral Supervisor at the School of Mathematics and Computer Science at Fujian Normal University. Presently, he is the Vice Dean of the School of Mathematics and Computer Science and the Director of the Key Lab of Network Security and Cryptography in Fujian Province.

Wei Wu Ph.D., she received her Ph.D. degree from the School of Computer Science and Software Engineering, University of Wollongong, Australia, in 2011. She is currently an Associate Professor at the Fujian Provincial Key Laboratory of Network Security and Cryptology, School of Mathematics and Computer Science, Fujian Normal University, China.

Please cite this article in press as: A. Ye, et al., The flexible and privacy-preserving proximity detection in mobile social network, Future Generation Computer Systems (2016), http://dx.doi.org/10.1016/j.future.2016.12.012