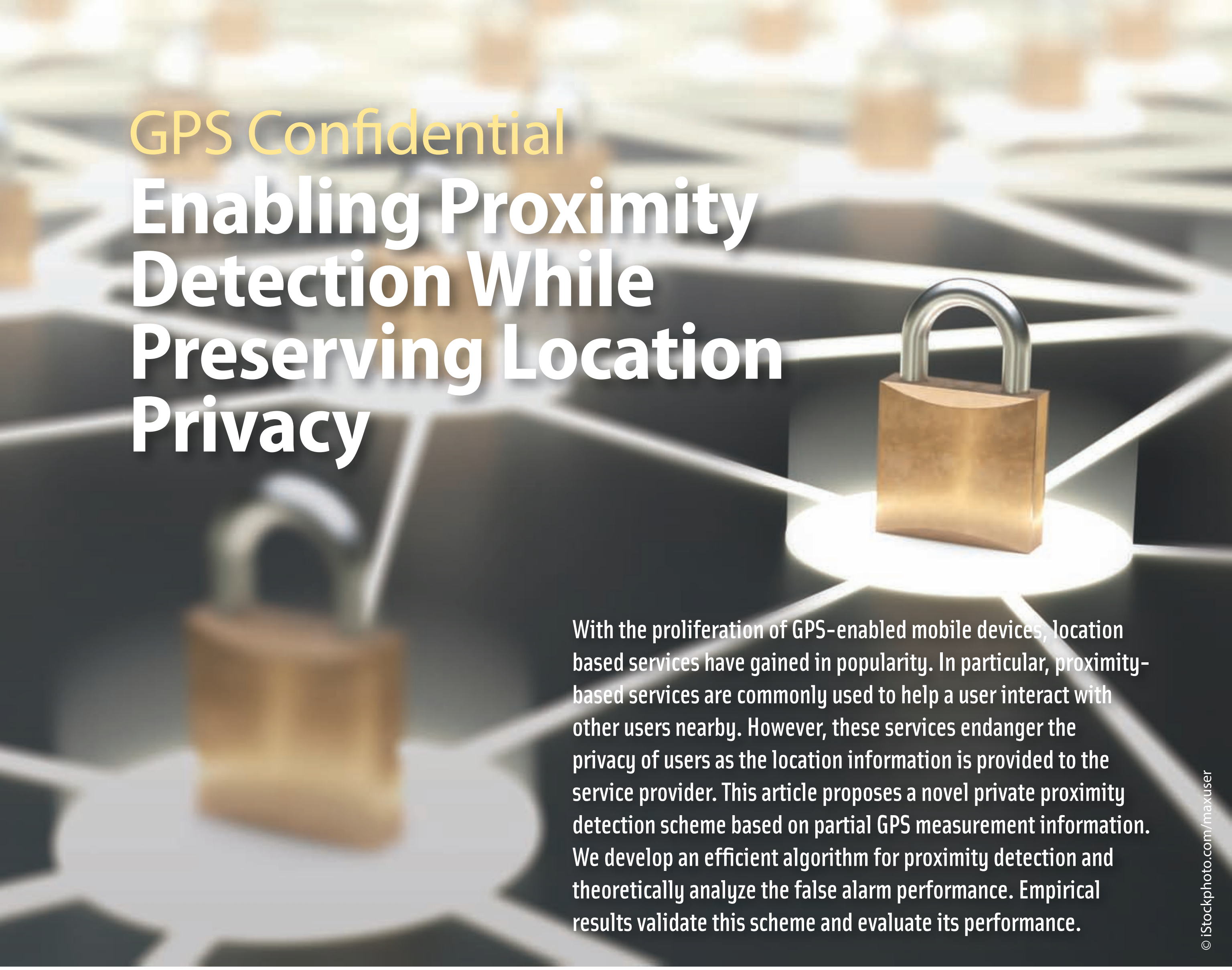

© iStockphoto.com/maxuser

GPS Confidential

Enabling Proximity Detection While Preserving Location Privacy

With the proliferation of GPS-enabled mobile devices, location based services have gained in popularity. In particular, proximity-based services are commonly used to help a user interact with other users nearby. However, these services endanger the privacy of users as the location information is provided to the service provider. This article proposes a novel private proximity detection scheme based on partial GPS measurement information. We develop an efficient algorithm for proximity detection and theoretically analyze the false alarm performance. Empirical results validate this scheme and evaluate its performance.

LIANG HENG,
ATHINDRAN RAMESH KUMAR,
GRACE XINGXIN GAO
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Ubiquitous location-aware mobile devices, mainly GPS-enabled smartphones, have led to a boom in location-based services (LBS), which have been revolutionizing businesses and lifestyles. Common uses of LBSs include asset tracking, location-based advertising, emergency roadside service, turn-by-turn navigation, and real-time traffic & road information sharing.

A major category in LBSs is proximity-based services, which allow users to search for friends or other points of interest around them. Examples of proximity-based mobile social networking include Apple's "Find My Friends," Facebook's "Nearby Friends," Tencent's "WeChat," Momo, and Nearby.

In Find My Friends, a user can see the locations of his friends and get notified when his friends are nearby. In WeChat, a user can find nearby users by the following two ways:

- Shake — A user shakes the phone, and the app will find other WeChat users who are also shaking at the moment locally and around the world. Then the user has an opportunity to message them and make new friends.
- Look Around — Look Around is like Shake without the shaking. The app simply finds other WeChat users who have been recently in the user's vicinity.

To use an LBS, a user usually has to send his exact location to the service provider, sporadically or frequently. The contextual information attached to user locations may, however, also reveal the users' habits, interests, activities, health status, and political and religious affiliations. The high level of intrusion and privacy threats associated has made many users reluctant to opt into LBSs. So, designing practical and effective privacy-preserving proximity-detection schemes would reassure users who have concerns about maintaining their privacy.

Past work on location privacy has explored quite a few approaches, such as anonymization, obfuscation, and adding dummies. A common concept underlying most of these approaches is to "degrade information in a controlled way before releasing it," as summarized in the paper by B. Hoh and M. Gruteser listed in the Additional Resources section near the end of this article.

This article proposes a new approach to preserving location privacy in proximity-based services. Our approach makes use of the location information inside a GPS receiver. The key difference from previous work is that rather than obtaining accurate location information and then degrading it, we extract privacy-preserving location information directly from an intermediate step in GPS location estimation.

Our private proximity detection scheme presented in this article is designed for location-based "friending" applications in a global social network. A user shares his untagged range measurement, which is derived from the user's GPS range measurements, with the server. The server can efficiently detect if any two users are within a threshold distance of each other. However, it is computationally intensive for the server to infer each user's exact location from the untagged range measurement. Our approach can be used independently or together with other approaches, such as obfuscation, to provide a higher level of privacy protection.

This article describes how untagged range measurements preserve location privacy and a very efficient matching algorithm for proximity detection. It also evaluates the proximity detection performance through a theoretical analysis and field experiments. The evaluation results demonstrate the efficacy and robustness of our scheme.

Previous Work in the Field

When a large number of people are using a "friending" app, a centralized server that detects proximity between each pair of users greatly reduces the communication costs. In this scenario, the goal would be to enable the server to perform proximity detection without leaking user locations to the server. The server should be able to infer as little information about user location as possible from the data it receives.

A possible approach is privacy-preserving test described in the article by A. Narayanan *et alia*, which is based on the *location tag* initially studied by the publications by D. Qiu *et alia* listed in Additional Resources. With proper location tags, location proximity can be reduced to measuring the similarity between two sets of tags. Narayanan *et alia* suggested deriving tags from surrounding environment including WiFi traffic and access point identifiers, GSM signals, and GPS signals.

The untagged range measurement proposed in this article is closely related to the location tag. A major difference is that the location tag still requires certain types of cryptography to work. The ElGamal encryption suggested by A. Narayan *et alia* requires much more computational resources on the user end and server side than does our method proposed in this article.

Private Proximity Testing Using Untagged GPS Range Measurements

Traditionally, GPS range measurements are tagged with satellite pseudo-random noise (PRN) codes, in order to identify the specific GPS satellites from which the ranges are measured. The novelty of our approach is based on untagged range measurements, a vector of GPS range measurements without pseudo-random noise (PRN) code designations. Suppose user *i* shares his untagged range measurements

$$\boldsymbol{r}_i = \left[r_i^{(1)}, r_i^{(2)}, \ldots, r_i^{(K_i)}\right]^{\mathrm{T}}, \qquad (2)$$

where K_i is the number of satellites visible to this user. The range measurement made to the satellite *k*, $r_i^{(k)}$ does not include the receiver clock bias, for all $k = 1, \ldots, K_i$, as the receiver clock bias can be easily calculated and removed beforehand. Furthermore, we require r_i to be a *sorted vector* in ascending order, i.e., $r_i^{(1)} \leq r_i^{(2)} \leq \cdots \leq r_i^{(K_i)}$.

Location Privacy Protection. When the range measurements are not designated with PRN numbers, if the adversary has no knowledge of satellite orbits, the *K* range measurements can be seen as an ordered selection from the *L* satellites in the whole constellation. Therefore, the search space is *K*-permutations of *L*.

Usually, we have $L \approx 30$ and $K \approx 10$, and thus the size of search space is $L!/(L - K)! \approx 1.1 \times 10^{14}$. Even though a server may use the knowledge of satellite orbits to reduce the search space, a large number of permutations still remain to be searched. Intensive computation discourages an adversary from inferring a user's actual location from untagged range measurements, especially in a large social network.

The untagged range measurements can also confuse an adversary. First, untagged range measurements seen at two or more distant locations may happen to be similar. These events are categorized as *distant false alarms* in a later section on "Proximity Detection Performance Analysis." Second, a user can add dummy measurements to his sorted vector so that multiple locations exist at which the untagged range measurements will be a subset of the sorted vector. In this article, we assume no dummy measurements added to sorted vectors.

Furthermore, the untagged range measurements are ephemeral. The satellite-to-user range is decreasing when the satellite is approaching and is increasing when the satellite is leaving. The range change rate varies with satellite elevation. For satellites at the zenith, the range rate is close to zero. For low-elevation satellites, the range rate can be as high as ±930 meters per second. Therefore, untagged range measurements are valid for

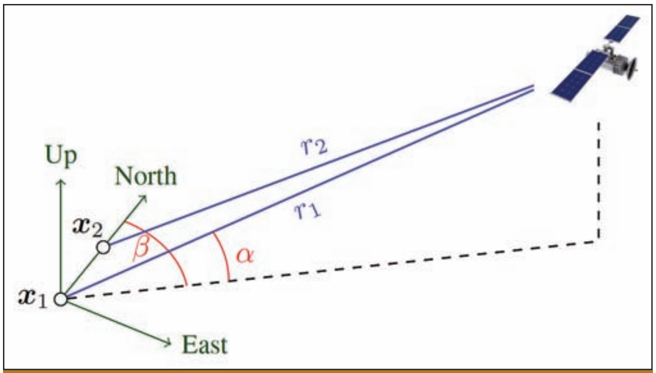

FIGURE 1 Two receivers make range measurements to the same satellite. Eq. (4) shows how the distance between the two receivers is related to the difference between the two range measurements.

$$\tau = t \Big/ \max_{1 \leq k \leq K} \{|\dot{r}_k|\}, \tag{3}$$

where t is the threshold distance, and $\dot{r}_k$ is the change rate of range k. If we choose t = 10 kilometers, then untagged range measurements are valid for approximately 10 seconds.

Proximity Detection. Consider two users at locations x_1 and x_2, as shown in Figure 1. Without loss of generality, assume user 2 is to the north of user 1. Suppose a GPS satellite is visible to both users, and the elevation and azimuth of the GPS satellite seen by user 1 are α and β. When the two users are nearby, the distance between the two users is much shorter than the distance to the satellite. Thus, we have the approximation

$$|r_2 - r_1| \approx \|\boldsymbol{x}_2 - \boldsymbol{x}_1\|_2 |\cos\alpha \cos\beta|. \tag{4}$$

Define a threshold distance $t > 0$. The two users are deemed "nearby" if $\|\boldsymbol{x}_2 - \boldsymbol{x}_1\|_2 \leq t$.. Therefore, a necessary condition for two users to be nearby is

$$|r_2 - r_1| \leq t|\cos\alpha \cos\beta| \leq t. \tag{5}$$

In our scheme, the server has to do blind matching of range measurements because the users do not designate PRN numbers to range measurements in order to protect their privacy. We formulate this "blind matching" problem in an optimization framework and propose an algorithm to solve it.

Blind Matching As an Optimization Problem. Let $\subset$ denote the subset relation between two sorted vectors. We write $\boldsymbol{x} \subset \boldsymbol{y}$ if each element in $\boldsymbol{x}$ also belongs to $\boldsymbol{y}$. Let card(x) denote the cardinality, i.e., number of elements, of a vector (or a set) x. The proximity detection problem can be formulated as the following optimization problem:

$$\begin{aligned} &\text{maximize} \quad c, \\ &\text{subject to} \quad c = \operatorname{card}(\boldsymbol{q}_1) = \operatorname{card}(\boldsymbol{q}_2), \\ &\qquad\qquad\quad \boldsymbol{q}_1 \in \boldsymbol{r}_1, \\ &\qquad\qquad\quad \boldsymbol{q}_2 \in \boldsymbol{r}_2, \\ &\qquad\qquad\quad \|\boldsymbol{q}_1 - \boldsymbol{q}_2\|_\infty \leq t, \end{aligned} \tag{6}$$

where the infinity norm

$$\|[u_1, \ldots, u_n]^{\mathrm{T}}\|_\infty = \max\{|u_1|, \ldots, |u_n|\}.$$

The optimization problem maximizes c, the number of matched range measurements. The decision whether the two users are nearby depends on c, card($\boldsymbol{r}_1$), and card($\boldsymbol{r}_2$). In this article, we use a very simple criterion: two users are decided to be nearby if the *match ratio*

$$m = \frac{c}{\min\{\operatorname{card}(\boldsymbol{r}_1), \operatorname{card}(\boldsymbol{r}_2)\}} \geq \zeta, \tag{7}$$

where the decision threshold ζ, $0 \leq \zeta \leq 1$, is selected to achieve certain detection error performance.

Efficient Blind Matching Algorithm. The optimization problem (6) is similar to the longest common subsequence (LCS) problem. Dynamic programming is often used to solve the LCS problem efficiently. Here we borrow this idea to solve the optimization problem.

Let $r^{(k)}$ denote the kth element in the sorted vector $\boldsymbol{r}$, and let $\boldsymbol{r}[n]$ denote the vector of the first n elements, i.e., $\boldsymbol{r}[n] = [r^{(1)}, r^{(2)}, \ldots, r^{(n)}]^{\mathrm{T}}$. Let $c(\boldsymbol{r}_1[k_1], \boldsymbol{r}_2[k_2])$ denote the maximum number of matched range measurements between $\boldsymbol{r}_1[k_1]$ and $\boldsymbol{r}_2[k_2]$. We then have the following recursive property:

$$c(\boldsymbol{r}_1[k_1], \boldsymbol{r}_2[k_2]) = \begin{cases} 0 & \text{if } k_1 = 0 \text{ or } k_2 = 0; \\ c(\boldsymbol{r}_1[k_1 - 1], \boldsymbol{r}_2[k_2 - 1]) + 1 & \text{if } |r_1^{(k_1)} - r_2^{(k_2)}| \leq t; \\ c(\boldsymbol{r}_1[k_1 - 1], \boldsymbol{r}_2[k_2]) & \text{if } r_1^{(k_1)} > r_2^{(k_2)} + t; \\ c(\boldsymbol{r}_1[k_1], \boldsymbol{r}_2[k_2 - 1]) & \text{if } r_1^{(k_1)} < r_2^{(k_2)} - t. \end{cases} \tag{8}$$

Using the foregoing property and the fact that both $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ are already sorted, we have the following algorithm.

```
Require: two sorted vectors
    $\boldsymbol{r}_{\mathrm{i}} = [r_{\mathrm{i}}^{(1)}, r_{\mathrm{i}}^{(2)}, \ldots, r_{\mathrm{i}}^{(K_i)}]^{\mathrm{T}}, i \in 1, 2\}$
Require: threshold distance $t > 0$
    $k_i \leftarrow 1, i \in \{1, 2\}$
    $c \leftarrow 0$
    while $k_1 \leq \operatorname{card}(\boldsymbol{r}_1)$ and $k_2 \leq \operatorname{card}(\boldsymbol{r}_2)$ do
        if $|r_1^{(k_1)} - r_2^{(k_2)}| < t$ then
            $c \leftarrow c + 1$
            $k_1 \leftarrow k_1 + 1$
            $k_2 \leftarrow k_2 + 1$
        else if $r_1^{(k_1)} < r_2^{(k_2)}$ then
            $k_1 \leftarrow k_1 + 1$
        else
            $k_2 \leftarrow k_2 + 1$
    return $c$
```

This algorithm achieves the worst-case time complexity of $O(\operatorname{card}(\boldsymbol{r}_1) + \operatorname{card}(\boldsymbol{r}_2))$ and the space complexity of $O(1)$. The algorithm only involves addition and comparison, two of the fastest operations on most CPUs. Therefore, our proximity detection algorithm is very efficient.

Once we obtain c using the previously described algorithm, we then use Equation (7) to determine if the two users are in proximity.

Proximity Detection Performance Analysis

As a statistical hypothesis test, private proximity detection has a probability of making two types of errors: *false alarm* and *missed detection*. Suppose there are N users and let S denote the set of all pairs of users, card(S) = $\binom{N}{2}$. Let X be the set of pairs of users who are within a threshold distance t. Let Y be the set of pairs of users who are detected to be close to each other. We define the following performance measures:

- Probability of false alarm (P_{FA})

$$P_{FA} = \frac{\text{card}(Y \setminus X)}{\text{card}(S \setminus X)}, \tag{9}$$

where the set difference

$$Y \setminus X = \{z \in Y | z \notin X\};$$

- Probability of missed detection (P_{MD})

$$P_{MD} = \frac{\text{card}(X \setminus Y)}{\text{card}(X)}; \tag{10}$$

- Probability of detection error (P_{DE})

$$P_{DE} = \frac{\text{card}(Y \setminus X) + \text{card}(X \setminus Y)}{\text{card}(S)}. \tag{11}$$

We focus our theoretical performance analysis on PFA for two reasons. First, PDE is dominated by PFA, as demonstrated by

$$P_{DE} = P_{FA}\frac{\text{card}(S \setminus X)}{\text{card}(S)} + P_{MD}\frac{\text{card}(X)}{\text{card}(S)}, \tag{12}$$

where in general we have card($S \setminus X$) >> card(X). Second, the inequality (5) always holds if two users are within the threshold. Therefore, missed detection mainly results from users accidentally losing track of several satellites, which can happen indoors, in an urban canyon, or in other GPS-challenged environments.

We should note the existence of two types of false alarm:

- *Nearby false alarm:* A pair of users are incorrectly detected to be nearby; their actual distance is greater than t, but still close to t, and they may see the same set of GPS satellites.
- *Distant false alarm:* A pair of users are incorrectly detected to be nearby; their actual distance is much greater than t, and they may see totally different sets of GPS satellites.

In this article, nearby false alarm is not our major concern because "proximity" itself is a fuzzy concept in social networking. For example, if two users within a distance t are always deemed nearby, it is acceptable that two users within a larger distance (e.g., 1.5t) are detected to be nearby with a certain probability. The following analysis is about distant false alarm.

Probabilistic Model of Ranges. Suppose we randomly choose a location on the Earth. At a random epoch the range to an arbitrary GPS satellite observed at this location is a random variable r. Let $\text{pdf}_r(x) = \frac{d}{dx}\text{Prob}(r \le x)$ be its probability density function.

Here we use a uniform distribution to approximate the actual distribution of ranges, i.e, $r \sim U(r_{min}, r_{max})$. When the satellite elevation mask angle is set to 10 degrees, $r_{min} \approx 20{,}189$ kilometers and $r_{max} \approx 24{,}619$ kilometers. Let the spread of range measurements $\lambda = r_{max} - r_{min}$, and we have $\text{pdf}_r(x) = 1/\lambda$.

A fundamental assumption of this analysis is that ranges to different satellites are independent and identically distributed (i.i.d.). The validity and efficacy of this assumption has been demonstrated in our previous work (L. Heng *et alia*).

In this analysis, we ignore GPS range measurement errors because such errors are much less than the threshold distance.

Probability of False Alarm. Suppose user 1 reports an untagged range vector $r_1 = [r_1^{(1)}, \ldots, r_1^{(K_1)}]^{T}$ and user 2 reports an untagged range vector $r_2 = [r_2^{(1)}, \ldots, r_2^{(K2)}]^{T}$. Both users are randomly chosen on the Earth so that with a very high probability they are far apart. Let c denote the number of matched range measurements. According to our discussion in the section on "Proximity Detection," false alarm occurs when the match ratio $m = c / \min\{K_1, K_2\}$ is grater than or equal to the threshold ζ.

Let us randomly shuffle r_2. Based on our i.i.d. assumption mentioned earlier, $r_2^{(k)} \sim U(r_{\text{min}}, r_{\text{max}})$ for all $k = 1, \ldots, K_2$. The probability of $r_2^{(1)}$ matching one of the elements of r_1 is given by

$$\begin{aligned} &\text{Prob}\left(r_2^{(1)} \in \bigcup_{k=1}^{K_1}\left[r_1^{(k)} - t, r_1^{(k)} + t\right]\right) \\ &\le \sum_{\mathrm{k}=1}^{K_1}\text{Prob}\left(r_2^{(1)} \in \left[r_1^{(k)} - t, r_1^{(k)} + t\right]\right) \\ &= 2\,tK_1/\lambda\,. \end{aligned} \tag{13}$$

If $r_2^{(1)}$ matches one of the elements of r_1, then the probability of $r_2^{(2)}$ matching one of the remainder elements of r_1 has a similar upper bound $2t(K_1 - 1)/\lambda$. Similarly, the ith match happens with a probability less than or equal to $2t(K_1 + 1 - i)/\lambda$.

Let $\eta = \lceil \zeta \min\{K_1, K_2\} \rceil$ be the required number of matched range measurements, where $\lceil\cdot\rceil$ is the ceiling function. Finally, the probability of at least η matched range measurements has the following upper bound:

$$\begin{aligned} P_{FA} &= \text{Prob}\,\{m \ge \zeta | \text{two randomly chosen users}\} \\ &= \text{Prob}\,\{c \ge \eta | \text{two randomly chosen users}\} \\ &\le \frac{2tK_1}{\lambda}\frac{2t(K_1 - 1)}{\lambda}\cdots\frac{2t(K_1 + 1 - \eta)}{\lambda}\binom{K_2}{\eta} \\ &= \eta!\left(\frac{2t}{\lambda}\right)^{\eta}\binom{K_1}{\eta}\binom{K_2}{\eta}. \end{aligned} \tag{14}$$

Since $P_{FA} \le 1$, we finally have the following upper bound:

$$P_{FA} \le \min\left\{1, \eta!\left(\frac{2t}{\lambda}\right)^{\eta}\binom{K_1}{\eta}\binom{K_2}{\eta}\right\}. \tag{15}$$

The equation shows that increasing λ (equivalent to using a lower mask angle) and/or decreasing threshold distance t will reduce the false alarm rate.

Experiment 1: Real Data from Global GPS Receiver Networks

We first validate our theory and algorithm using real GPS pseudorange measurements collected by the International GNSS Service (IGS) and the University NAVStar Consortium (UNAVCO). The two networks consist of more than 1,000 stations all over the world. Each station has one or multiple GPS receivers continuously generating GPS pseudorange measurement data. We obtained range measurements by removing receiver clock biases from pseudoranges. The experiment provides a more realistic assessment because the receivers occasionally lose GPS signals.

We treated the IGS and UNAVCO stations as nodes to test for proximity using the scheme outlined in the earlier section. These stations are usually very far (at least tens of kilometers) apart. With a proper distance threshold, the receivers at different stations can be seen as distant users, while the receivers at the same stations are nearby users.

We applied our algorithm to the IGS data recorded on January 10, 2014. The pseudorange measurements released by 1,171 stations around the world at the start of the day during one time epoch was used to aggregate the statistics for validation. **Figure 2** shows the variation of probability of false alarm with the threshold distance.

In **Figure 3**, we see that the missed detection rate is below 0.05 for $\zeta \leq 0.8$. However, the false alarm rate is higher for lower values of ζ. This trade-off is succinctly depicted in **Figure 4**, which plots the probability of detection $P_D = 1 - P_{MD}$ versus P_{FA}, also known as the receiver operating characteristic (ROC) curve. Thus, the empirical results clearly illustrate the viability of our scheme for efficient private proximity detection.

The results with real data demonstrate the robustness of our scheme. Occasional loss of satellites cause missed detection. With a proper choice of decision threshold ζ, we can still achieve satisfactory detection performance.

Experiment 2: Real Data from Android Phones

With the evaluation using the IGS tracking network, the "user" locations were fixed. Further, most pairs of stations were very far apart and the locations of the stations in the tracking network do not model the distribution of mobile phone users very well. Thus, we performed some local experiments to further validate the algorithm.

We developed an Android application to log GPS range data. Upon post-processing, we can evaluate the utility of our scheme. However, working with the Android API presents a fresh set of challenges. An additional evaluation using GPS receivers is presented to further strengthen the proof of concept.

In an ideal scenario, an implementation of a proximity-based service using an Android app would just have the

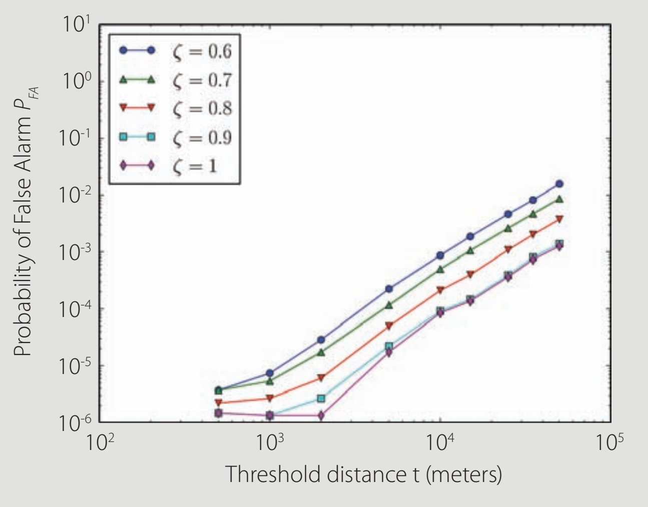

FIGURE 2 Probability of false alarm P_{FA} versus threshold distance t. A low false alarm rate ($P_{FA} \leq 10^{-4}$) is achieved when $\zeta \leq 0.7$ for threshold distance $t \leq 5{,}000$ meters.

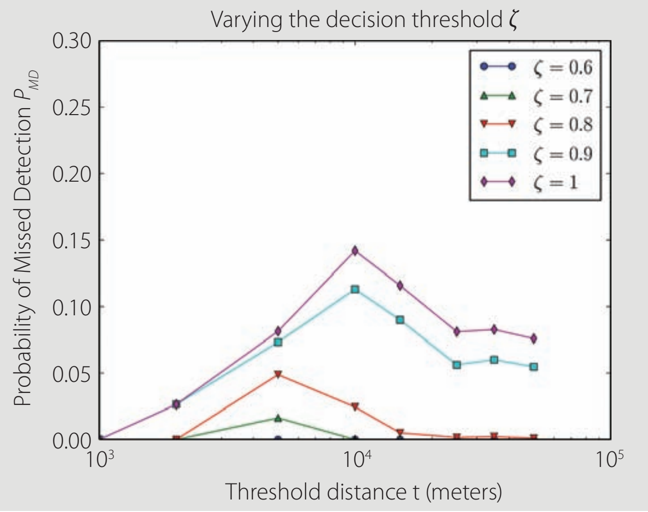

FIGURE 3 Probability of missed detection P_{MD} versus threshold distance t. A low missed detection rate ($P_{MD} \leq 0.05$) is achieved when $\zeta \leq 0.8$ for all threshold distance.

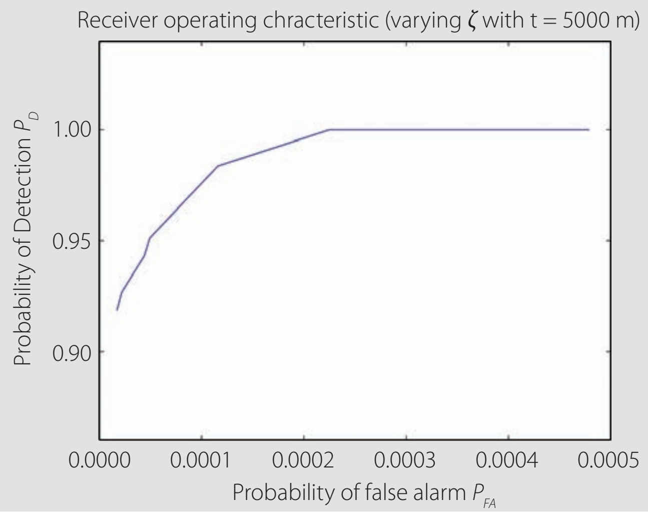

FIGURE 4 Receiver operating characteristic (ROC) curve

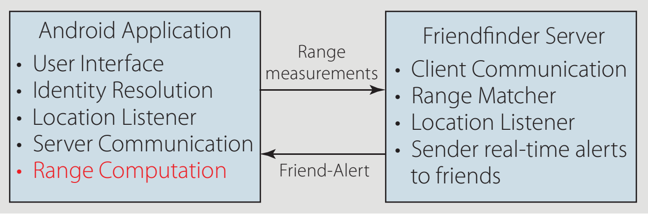

FIGURE 5 Real world implementation of an Android proximity detection service

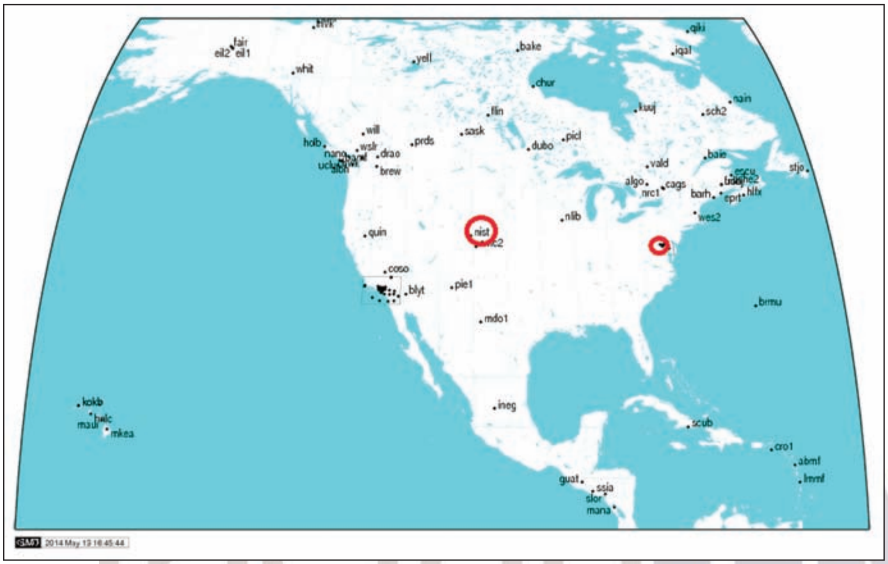

FIGURE 6 IGS stations used for downloading ephemeris (marked in red).

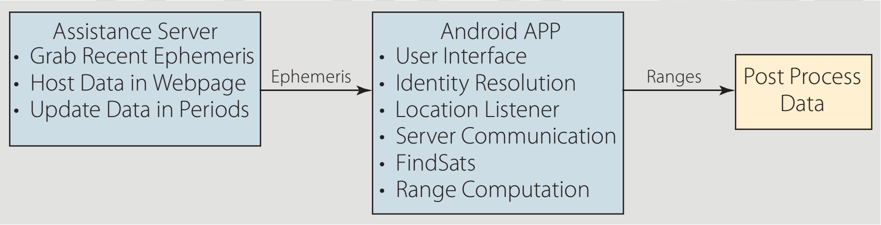

FIGURE 7 Final implementation of the Android app for logging data

Android app interacting with a friendfinder server as shown in Figure 5. However, several challenges arose while working with the Android location application programming interface (API), which is the only mode of accessing the underlying GPS engine. We provide a description of the challenges and the suitable modifications required below.

Pseudorange Measurements Not Available from the API. An app developer can only access geodetic coordinates (latitude, longitude, and height) information of an user. Thus, we had to set up an external server to continually download high rate ephemeris from nearby IGS stations. The high rate ephemeris within the last two hours from the stations NIST and GODS (shown in Figure 6) were downloaded and hosted on a server.

The Android app downloaded and updated the ephemeris from the server periodically. Further, Android provides a GPSSatellite class which reports the satellites currently in view along with the elevation and azimuth. Using the last known location, the satellites in view and the downloaded ephemeris, we find the range measurements and form the anonymous range vector for proximity detection.

Unknown Clock Bias and Time Sync Issues. Another major issue is that the Android API does not report accurate GPS time. It only reports the coordinated universal time (UTC) time at fix and the clock error can be as large as a few minutes. In order to circumvent this problem, the app was forced to compute the satellite positions and range measurements at fixed time epochs.

Initially, we manually configured each of the devices used for testing to not have time lag more than 5–10 seconds. However, the range measurements vary rapidly as the satellites are continually moving. In order to ensure time sync between users, the app was forced to report range measurements at fixed time epochs, i.e, integer multiples of 20 seconds.

Fluctuating Satellite Set. As stated earlier, the GPSSatellite class reports the satellites currently in view along with the elevation and azimuth. However, the satellites in view were very fluctuant and the 4-5 satellites reported changed very rapidly (on the order of seconds). Thus, we aggregated the reports over 10 seconds to find all the satellites in view.

Synchronous versus Asynchronous Implementation. In an asychronous implementation, the user notifies the LBS server when he/she is looking for friends nearby. The LBS server can then notify the user's friends and obtain their untagged range measurements to test for proximity. In a synchronous implementation, the users report their untagged range measurements periodically at fixed time epochs.

The former implementation is obviously better in terms of privacy because the users give away less information. Also, there is the extra communication overhead of continuously reporting data in the synchronous implementation. However, we resort to a synchronous implementation in this work for simplicity. Further, we do not implement a friend-finder server as we are simply recording data for research purposes. The Android app just logs all the pseudorange data in a text file. Figure 7 shows the final implementation used for this experiment.

We processed the files from all users after the recording to evaluate our algo-

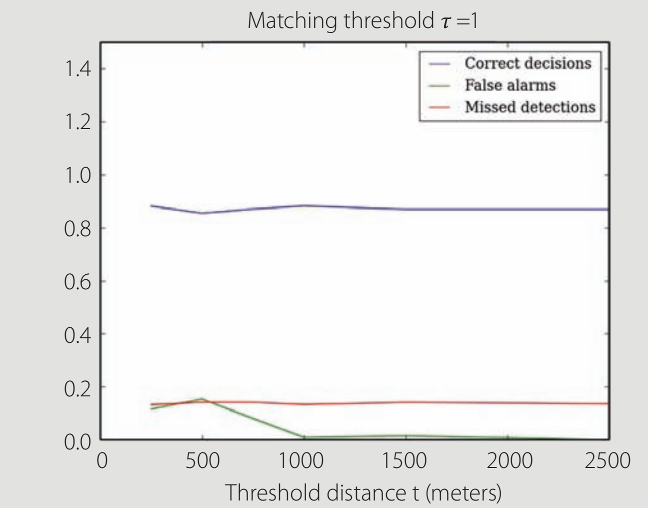

FIGURE 8 Probability of correct decisions, false alarms, and missed detections of the proximity detection algorithm with the data logged through Android phones.

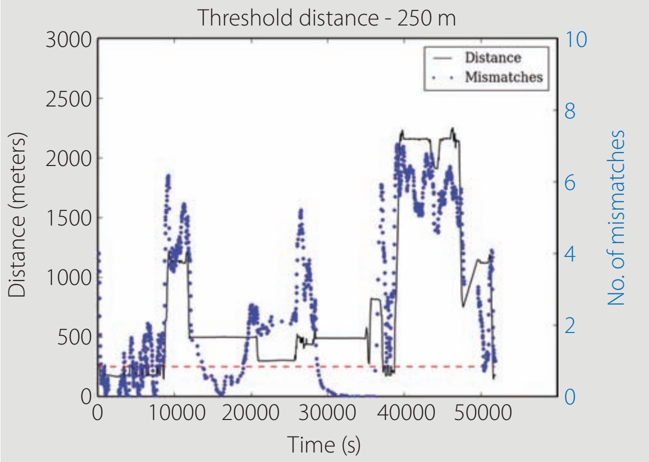

FIGURE 9 Variation of number of mismatches with distance for two users over a day with t = 250 m. The number of mismatches is higher when the distance between the users increases more than 250 meters.

rithms. The app was distributed to six graduate students who lived and worked in Urbana-Champaign, Illinois. They primarily worked indoors and were close to each other for most part of the data collection. We present the results from this data collection in Figures 20, 21, and 22.

Figure 8 shows the probability of correct decisions, false alarms, and missed detections. Unlike the IGS stations, most users in this experiment were close to each other for the most period. Hence, we see a relatively higher ratio of false alarms as the denominator in equation (9) is considerable smaller. **Figures 9** and **10** show an interesting comparison of the number of mismatches between the anonymous range vectors of two users using two different threshold distance parameters *t*. We see that the results are consistent with what we expect.

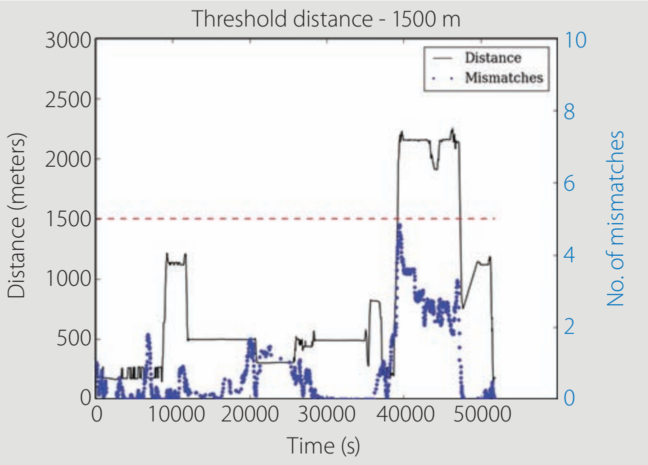

FIGURE 10 Variation of number of mismatches with distance for two users over a day with *t* = 1,500 m. The number of mismatches is higher when the distance between the users increases more than 1,500 meters.

Experiment 3: Real Data from GPS Receivers

In the previous section, we elaborated on the challenges and our approach to Android data collection. We had to construct our own range measurements from the satellite ephemeris and the last known position.

We thus decided to perform another experiment with small L1 single-frequency GPS receivers known to output pseudorange measurements. Four graduate students from University of Illinois took part in this experiment. Two of them drove in opposite directions from the Urbana-Champaign campus for a few kilometers. Two others walked around on the campus. The paths of these users are presented in **Figure 11**.

Three accompanying figures present the variation of match ratio with distance for *t* = 750, 1,500, and 5,000 meters, respectively, for a pair of users. From **Figure 12**, we can see that there is a sharp decrease in match ratio when the distance between the users increases more than the threshold of *t* = 750 meters. This demonstrates the robustness of the algorithm. For the same pair of users, as can be seen in **Figure 13**, a higher threshold of *t* = 1500 meters gives appropriate results in terms of match ratio. In **Figure 14**, the threshold *t* = 5,000 meters is always above the distance between the two users. As a result, the match ratio is close to 1.0 most of the time.

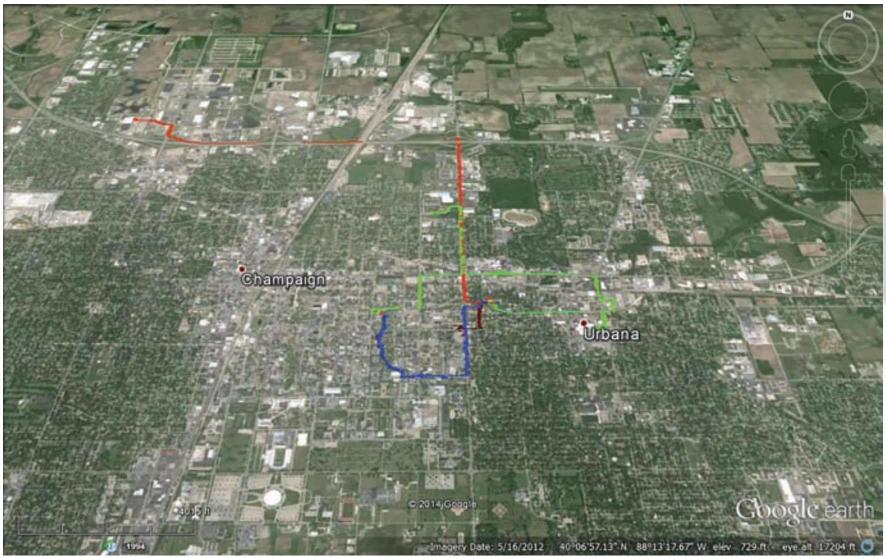
Champaign
Urbana
© 2014 Google
Google earth
1994
Imagery Date: 5/16/2012 40°06'57.13" N 88°13'17.67" W elev 729 ft eye alt 17204 ft

FIGURE 11 Paths of the four users during data collection

Conclusion

This article proposed a novel private proximity detection method, which makes use of partial GPS measurement information. We developed an efficient algorithm for proximity detection. We theoretically analyzed proximity detection performance and derived an upper bound on probability of false alarm.

We further conducted experiments using globally and locally collected data. The empirical results demonstrated the efficacy and robustness of our scheme for performing private proximity detection.

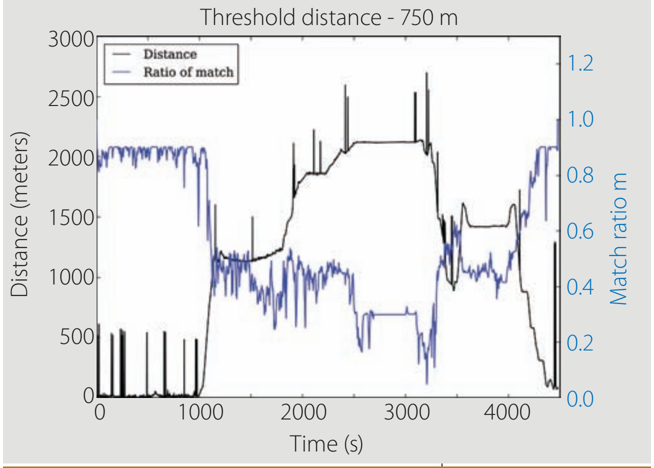

FIGURE 12 Variation of match ratio with distance for *t* = 750 meters. The match2ratio is lower when the distance between the users increases more than 750 meters.

Acknowledgments

This article is based on two of the authors' papers: "Private Proximity Detection Using Partial GPS Information," submitted to *IEEE Transactions on Aerospace and Electronic Systems,* and "GPS Privacy: Enabling Proximity-Based Services While Keeping GPS Locations Private," in *Proceedings of the 27th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS+ 2014),* Tampa, Florida, USA.

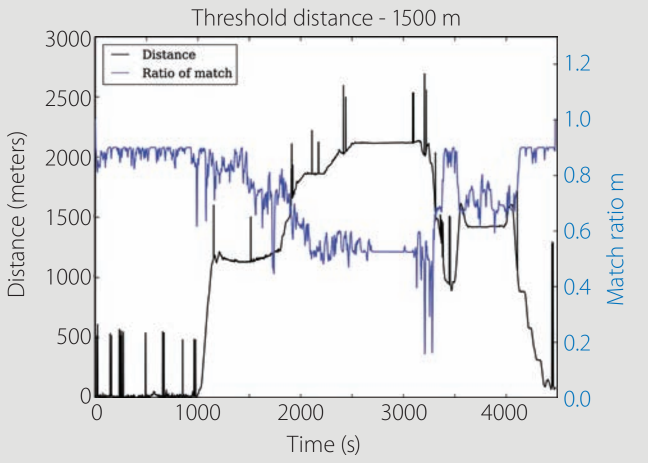

FIGURE 13 Variation of match ratio with distance for *t* = 1500 meters. The match ratio is lower when the distance between the users increases more than 1500 meters.

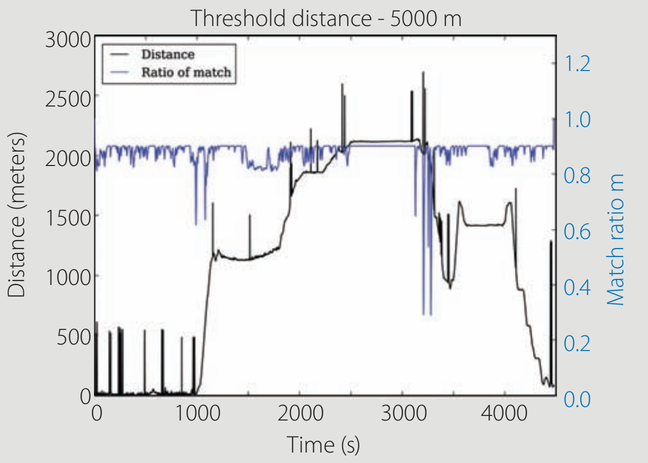

FIGURE 14 Variation of match ratio with distance for *t* = 5000 meters. The match ratio is high most of the time because the distance between the users were always within 5000 meters.

Manufacturers

The GPS receivers used to test the privacy-preserving proximity detection method described in this article were LEA-6T receivers from **u-blox**, Thalwil, Switzerland.

Additional Resources

[1] Apostolico, A., and C. Guerra, "The Longest Common Subsequence Problem Revisited," *Algorithmica*, vol. 2, no. 1-4, pp. 315–336, 1987

[2] Atallah, M. J., and W. Du, "Secure multi-party computational geometry," in *Proceedings of the 7th International Workshop on Algorithms and Data Structures,* ser. WADS '01, pp. 165–179, London, UK, Springer-Verlag, 2001

[3] Ghinita, G., "Privacy for Location-Based Services," ser. *Synthesis Lectures on Information Security, Privacy, and Trust*, Morgan & Claypool Publishers, 2013

[4] Heng, L., and T. Walter, P. Enge, and G. X. Gao, "Overcoming RFI with High Mask Angle Antennas and Multiple GNSS Constellations," in *Proceedings of the 26th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS+ 2013),* pp. 3433–3442, Nashville, Tennessee USA, September 2013

[5] Hoh, B., and M. Gruteser, "Protecting Location Privacy through Path Confusion," in *Security and Privacy for Emerging Areas in Communications Networks,* 2005, SecureComm 2005, First International Conference on Security and Privacy in Communication Networks, 2005, pp. 194–205

[6] Krumm, J., "A survey of computational location privacy," *Personal Ubiquitous Computing,* vol. 13, no. 6, pp. 391–399, August 2009

[7] Matsuo, Y., and N. Okazaki, K. Izumi, Y. Nakamura, T. Nishimura, and K. Hasida, "Inferring long-term user property based on users location history," in *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI '07),* 2007

[8] Misra, P., and P. Enge, *Global Positioning System: Signals, Measurements, and Performance,* 2nd ed., Ganga-Jamuna Press, Lincoln, Massachusetts USA, 2006

[9] Narayanan, A., and N. Thiagarajan, M. Lakhani, M. Hamburg, and D. Boneh, "Location privacy via private proximity testing." in *Proceedings of 18th Annual Network & Distributed System Security Symposium (NDSS 2011),* 2011

[10] Qiu, D., and D. Boneh, S. Lo, and P. Enge, "Robust Location Tag Generation from Noisy Location Data for Security Applications," in *Proceedings of the Institute of Navigation International Technical Meeting (ION ITM),* September 2009

[11] Qiu, D., and S. Lo, and P. Enge, "Security for Insecure Times: Geoencryption with Loran," *GPS World,* 2007

[12] Ravichandran, R., and M. Benisch, P. G. Kelley, and N. M. Sadeh, "Capturing Social Networking Privacy Preferences," in *Proceedings of the 9th International Symposium on Privacy Enhancing Technologies,* ser. PETS '09, pp. 1–18, Springer-Verlag, Berlin, Heidelberg, Germany, 2009

[13] Riley, P. F., "The tolls of privacy: An underestimated roadblock for electronic toll collection usage," *Computer Law & Security Review,* vol. 24, no. 6, pp. 521–528, 2008

[14] Shin, K. G., and X. Ju, Z. Chen, and X. Hu, "Privacy Protection for Users of Location-Based Services," *IEEE Wireless Communications,* vol. 19, no. 1, pp. 30–39, 2012

[15] Swets, J. A., *Signal Detection Theory and ROC Analysis in Psychology and Diagnostics: Collected Papers,* Lawrence Erlbaum Associates, Mahwah, New Jersey USA, 1996

[16] Zhong, G., and I. Goldberg, and U. Hengartner, "Louis, Lester and Pierre: Three Protocols for Location Privacy," in *Privacy Enhancing Technologies,* ser. Lecture Notes in Computer Science, vol. 4776, pp. 62–76 N. Borisov and P. Golle, Eds. Springer Berlin Heidelberg, 2007

Authors

Liang Heng received the B.S. and M.S. degrees in electrical engineering from Tsinghua University, Beijing, China. He received the Ph.D. degree in electrical engineering from Stanford University. From 2013 to 2014, he was a postdoctoral research associate at University of Illinois at Urbana-Champaign, where this work was done. His research interests are cooperative navigation and satellite navigation. He is a member of the IEEE and ION.

Athindran Ramesh Kumar received his B.Tech degree in electrical engineering from the Indian Institute of Technology, Madras. He completed his M.S. degree in electrical and computer engineering at the University of Illinois, Urbana-Champaign.

Grace Xingxin Gao is an assistant professor in the Aerospace Engineering Department at University of Illinois at Urbana-Champaign. She obtained her Ph.D. degree in electrical engineering at Stanford University. Prof. Gao has won a number of awards, including RTCA William E. Jackson Award, Institute of Navigation Early Achievement Award, multiple best presentation awards at ION GNSS conferences, and College of Engineering Everitt Award for teaching excellence at University of Illinois at Urbana-Champaign. IG