Achieving Privacy Preservation in WiFi Fingerprint-Based Localization

Hong Li[1,2,3], Limin Sun[1,2], Haojin Zhu[4], Xiang Lu[1,2] and Xiuzhen Cheng[5]

[1]State Key Laboratory of Information Security, Institute of Information Engineering, CAS, China
[2]Beijing Key Laboratory of IOT information security technology, Institute of Information Engineering, CAS, China
[3]University of Chinese Academy of Sciences, China
[4]Department of Computer Science & Engineering, Shanghai Jiao Tong University, China
[5]Department of Computer Science, The George Washington University, USA
Email: {lihong, sunlimin, luxiang}@iie.ac.cn, zhu-hj@sjtu.edu.cn, cheng@gwu.edu,

***Abstract*—WiFi fingerprint-based localization is regarded as one of the most promising techniques for indoor localization. The location of a to-be-localized client is estimated by mapping the measured fingerprint (WiFi signal strengths) against a database owned by the localization service provider. A common concern of this approach that has never been addressed in literature is that it may leak the client's location information or disclose the service provider's data privacy. In this paper, we first analyze the privacy issues of WiFi fingerprint-based localization and then propose a Privacy-Preserving WiFi Fingerprint Localization scheme (PriWFL) that can protect both the client's location privacy and the service provider's data privacy. To reduce the computational overhead at the client side, we also present a performance enhancement algorithm by exploiting the indoor mobility prediction. Theoretical performance analysis and experimental study are carried out to validate the effectiveness of PriWFL. Our implementation of PriWFL in a typical Android smartphone and experimental results demonstrate the practicality and efficiency of PriWFL in real-world environments.**

***Index Terms*—WiFi fingerprint-based localization; location privacy; data privacy; homomorphic encryption.**

I. INTRODUCTION

The high demand for location information to enable pervasive computing applications in indoor environments, coupled with the lack of GPS signals in buildings, has motivated a large body of research on indoor localization. In particular, there has been a focus on leveraging existing infrastructures (e.g., WiFi access points) to enable indoor localization, taking advantage of the fact that the cost of deploying a specialized infrastructure for localization can be avoided. One of the most popular localization techniques used for positioning with wireless access points is based on the received signal strength (RSS) and the method of fingerprinting [1], [2], [3], [4]. Commercial providers of this type of positioning services include Google, Navizon, Skyhook Wireless, as well as WiFiSLAM.

WiFi fingerprint-based localization consists of two phases, *offline training* and *online operating*. In the offline training phase, the service provider measures the WiFi signal strengths (WiFi RSS fingerprints) from various access points (APs) of known locations and records the WiFi fingerprints and the corresponding locations in a fingerprint database. The database can be stored on a localization server operated by the service provider or distributed to clients' end devices. During the online operating phase, a client that is going to locate itself could measure the signal strengths at a specific location from nearby APs to obtain the WiFi RSS fingerprints at this position. Then an algorithm is employed to compute the Euclidean distance between the sampled data and each WiFi fingerprint in the database. After that, k "closest fingerprints" are chosen to derive the client's location.

Although regarded as a promising approach for indoor localization, WiFi fingerprint-based localization can lead to potential privacy violations. On one hand, a localization query carrying the client's WiFi fingerprint can inevitably leak the client's location information. Existing research indicates that location traces can leak information about the individuals' habits, interests, activities, and relationships [5], [6]. Consequently, the loss of location privacy can expose users to unwanted advertisements and location-based spams/scams, may cause social reputation or economic damage to the users, and can make the users victims of blackmails or even physical violences. On the other hand, due to the high cost of building a fingerprint database, the service provider has a strong incentive to protect its precious WiFi fingerprint data from leaking to unauthorized mobile users. Therefore, keeping the data confidentiality while still allowing the proper functionality of the WiFi fingerprint database is another fundamental requirement to achieve a practical WiFi fingerprint-based positioning system.

Several approaches have been proposed to protect clients' location privacy in location-based services (e.g., k-anonymity [7], [8] and mix zones [9]). However, they face the challenge of lacking trusted third parties in WiFi fingerprint-based localization. Furthermore, these works intend to protect the location privacy of the users requesting location-based services by submitting their location information with the requests, assuming that each client has obtained its location without any privacy concern. Another line of related research is privacy-

978-14799-3360-0/14/$31.00 ©2014 IEEE

preserving biometric identification [10], [11], which seeks only the closest match. In contrast, in our WiFi fingerprint-based localization, k "closest fingerprints" are required simultaneously for location estimation.

In this paper, we propose a Privacy Preserving WiFi Fingerprint-based Localization scheme, coined as PriWFL. We first give a naive version of PriWFL, addressing how clients' location privacy can be protected based on homomorphic encryption. To overcome the privacy vulnerabilities of the naive scheme, several enhancements are introduced to obtain the complete version of PriWFL which can protect both the client's location privacy and the service provider's data privacy. Furthermore, to mitigate the client's computational overhead, we present a novel indoor mobility prediction algorithm. Experimental results and theoretical performance analysis both demonstrate the effectiveness of our proposed scheme in privacy protection for WiFi fingerprint-based localization. The major contributions of this paper are summarized as follows:

- To the best of our knowledge, this work is the first to address the privacy issues of WiFi fingerprint-based localization.
- We propose and justify the design motivation of PriWFL, a novel scheme that can protect the client's location privacy and the server's data privacy.
- We design an indoor mobility prediction algorithm to reduce the computational overhead at the client side.
- We implement PriWFL and carry out a comprehensive experimental study in typical office buildings to evaluate its performance.

The rest of the paper is organized as follows. Section II introduces the background and threat model. Section III details the description of our proposed scheme. Section IV presents the indoor mobility prediction algorithm. Section V reports the experimental evaluation results. We discuss the related work in Section VI and conclude this paper in Section VII.

II. Background and Threat Model

A. Overview of WiFi Fingerprint-Based Localization

WiFi fingerprint-based localization is one of the most commonly used indoor localization techniques. The process of WiFi fingerprint-based localization can be divided into two phases: offline training phase and online operating phase. During the training phase, experts (hired by the service provider) measure the WiFi fingerprint V_i at each location (x_i, y_i) in an interested area, and store $\langle i, (x_i, y_i), V_i \rangle$ in the WiFi fingerprint database D. Here $V_i = (v_1, v_2, ..., v_j, ..., v_N)$ is a N-dimensional feature vector, with v_j being the average WiFi signal strength at location (x_i, y_i) from the jth access point AP_j, and N being the total number of access points. The database D can be stored on a localization server or distributed to the clients. In the online operating phase, a to-be-localized client first samples the signal strengths from various APs, which is denoted as $V' = (v'_1, v'_2, ..., v'_j, ..., v'_N)$. Then the squared Euclidean distance d_i between V' and a WiFi fingerprint V_i is computed as follows

$$\begin{aligned} d_i &= \|V' - V_i\|^2 = \sum_{j=1}^{N} (v_{i,j} - v'_j)^2 \\ &= \underbrace{\sum_{j=1}^{N} v_{i,j}^2}_{S_{i,1}} + \underbrace{\sum_{j=1}^{N} (-2 v_{i,j} \cdot v'_j)}_{S_{i,2}} + \underbrace{\sum_{j=1}^{N} v'^2_j}_{S_3} \end{aligned} \tag{1}$$

Finally, the k nearest neighbors are selected to estimate the client's location, i.e., the centroid of the locations corresponding to the k nearest neighbors is the location of the to-be-localized client.

B. Threat Models and Assumptions

Our study considers both the client's location privacy and the service provider's data privacy. From a client's perspective, the WiFi position system should allow the client to make the geo-localization without compromising its location privacy. The attacker could be a curious service provider who collects the locations of the customers to make marketing and sales strategies, or an external attacker who harvests the locations of wireless users and sells them for profit. Two location privacy threats are considered in this study.

- **Location Privacy Attack I:** In this attack, the attacker directly obtains the client's location from the query.
- **Location Privacy Attack II:** In this attack, the attacker gets the client's sampled WiFi RSS and then infers client's location.

The service provider should also keep its WiFi fingerprint database from unauthorized leaking. For example, a malicious client may download the database and sell it for profit. The privacy attacks suffered by a service provider can be classified as follows:

- **Data Privacy Attack I:** The attacker gets a WiFi fingerprint database, which is exactly the same as the localization service provider's database D.
- **Data Privacy Attack II:** The attacker gets a WiFi fingerprint database D', which is similar to the service provider's database D and the localization accuracy of using D' is comparable with that of using D.

In this work, we consider a general curious-but-honest attack model; namely, both a client and the service provider honestly follow the designed protocol but each intends to disclose the other's private information.

C. The Paillier Cryptosystem

In this work, we employ the Paillier cryptosystem as our cryptographic primitive. Invented by Pascal Paillier [12], the Paillier cryptosystem is a probabilistic asymmetric algorithm based on the decisional composite residuosity problem. Define $n = pq$, with p and q being two large prime numbers. Let $g = n + 1$ [13], and m be the plaintext to be encrypted. Denote the ciphertext of m by $[\![m]\!]$, which is given by

$$[\![m]\!] = g^m r^n \mod n^2 \tag{2}$$

where $r \in \mathbb{Z}_n^*$ is a random number. Due to the choice of g, encryption requires one modular exponentiation and two modular multiplications while decryption involves one modular exponentiation.

The Paillier cryptosystem is additively homomorphic. Given only the public key, one can compute $[\![m_1 + m_2]\!]$ from $[\![m_1]\!]$ and $[\![m_2]\!]$ as follows:

$$[\![m_1 + m_2 \mod n]\!] = [\![m_1]\!] \cdot [\![m_2]\!] \mod n^2 \quad (3)$$

Furthermore, given $[\![m]\!]$ and a constant c, $[\![c \cdot m \mod n]\!]$ can be computed by:

$$[\![c \cdot m \mod n]\!] = [\![m]\!]^c \mod n^2 \quad (4)$$

III. Privacy Preserving WiFi Fingerprint-Based Localization Scheme

To address the privacy issues described above, we propose a Privacy Preserving WiFi Fingerprint-Based Localization scheme, termed PriWFL, in this section. To better present our design motivations, we first give a naive version of PriWFL and analyze its weakness; then we present the complete version of PriWFL and prove its strength in protecting the location privacy of the clients and the data privacy of the service provider. For simplicity, we first consider a one-building scenario; then discuss how to extend the scheme to a multi-building scenario.

A. Scheme Overview

As shown in Figure 1, PriWFL involves two components: client and localization server. The server operated by the service provider holds a fingerprint database $\langle i, (x_i, y_i), V_i = \{v_{ij}\}_{j=1}^N \rangle_{i=1}^M$, where i is an index, V_i denotes the WiFi fingerprint of location (x_i, y_i), and N is the total number of APs. PriWFL is decomposed into four phases. In the Pre-process phase, the server distributes some metadata to a to-be-localized client. This phase can be executed before the client uses the localization service, and only needs to be performed once. In the Preparation phase, the client generates a pair of keys $< PK_c, PR_c >$ using the Paillier cryptosystem, encrypts the sampled WiFi fingerprint V' using its public key PK_c, and then submits the corresponding ciphertext and PK_c to the server. In the Distance Computation phase, the server computes the encrypted distance $[\![d_i]\!]$ between V' and each WiFi fingerprint V_i in D based on homomorphic encryption. At the end of this phase, the server returns $\{[\![d_i]\!]\}_{i=1}^M$ back to the client. Finally in the Location Retrieval phase, the client decrypts $\{[\![d_i]\!]\}_{i=1}^M$ using it's private key PR_c, finds k "closest fingerprints", and estimates its location based on the metadata.

B. The Naive PriWFL and Its Privacy Analysis

In this section, we first introduce the naive PriWFL, which is based on homomorphic encryption. Then we discuss the privacy issues of the naive scheme.

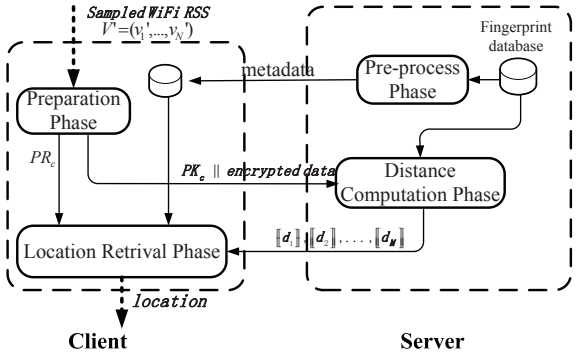


Fig. 1. The Overview of PriWFL.

1) The Naive PriWFL: The four phases of the naive PriWFL are detailed as follows.

Pre-Process Phase: Assume that the server holds a fingerprint database $D = \langle i, (x_i, y_i), V_i = \{v_{ij}\}_{j=1}^N \rangle_{i=1}^M$, which is collected from a building B. In this phase, the server distributes the metadata extracted from the database to the client. The metadata consists of two tables $T_1 = \langle i, (x_i, y_i) \rangle_{i=1}^M$ and $T_2 = \langle AP_i \rangle_{i=1}^N$. Here T_1 stores the indexes and all the locations at which the WiFi fingerprints are measured. From T_2, the client gets to know which APs can be used for localization and how to line up the sensed WiFi fingerprint with the fingerprints in the database.

This phase can be executed before a client uses the localization service, and only needs to be performed once. From the metadata, the client knows the corresponding physical locations of the fingerprints in the database but not the fingerprints themselves. Note that a malicious client cannot build up its own localization system by knowing only the metadata; thus distributing the metadata to the client does not compromise the service provider's data privacy.

Preparation Phase: When a client holding a sampled WiFi fingerprint $V' = (v'_1, v'_2, ..., v'_N)$ needs to be localized, it first generates a pair of keys $< PK_c, PR_c >$ using the Paillier cryptosystem, and publishes the public key PK_c to the server. Then it encrypts $\{-2v'_1, -2v'_2, ..., -2v'_N\}$ and $S_3 = \sum_{j=1}^N v'^2_j$ using the public key PK_c. Finally, the client sends a request containing $[\![-2v'_1]\!]$,...,$[\![-2v'_N]\!]$ and $[\![S_3]\!]$ to the server.

Distance Computation Phase: After receiving the client's request, the server computes the encrypted squared Euclidean distance $[\![d_i]\!]$ between V' and V_i using homomorphic encryption: $[\![d_i]\!] = [\![S_{i,1} + S_{i,2} + S_3]\!] = [\![S_{i,1}]\!] \cdot [\![S_{i,2}]\!] \cdot [\![S_3]\!]$ where $[\![S_{i,1}]\!]$ and $[\![S_{i,2}]\!]$ can be computed by $[\![S_{i,1}]\!] = \left[\!\!\left[\sum_{j=1}^N v_{i,j}^2 \right]\!\!\right]$ and $[\![S_{i,2}]\!] = \left[\!\!\left[\sum_{j=1}^N (-2v_{i,j} \cdot v'_j) \right]\!\!\right] = \prod_{j=1}^N [\![(-2v'_j)]\!]^{v_{i,j}}$. Then, the server sends $\{[\![d_i]\!]\}_{i=1}^M$ back to the client.

Location Retrieval Phase: After receiving the encrypted distance, the client first decrypts $\{[\![d_i]\!]\}_{i=1}^M$. Then it finds out the k smallest distances $\{d_{I_1}, d_{I_2}, ..., d_{I_k}\}$. Accordingly,

$\{(x_{I_1}, y_{I_1}), (x_{I_2}, y_{I_2}), ..., (x_{I_k}, y_{I_k})\}$ are retrieved from table T_1 using indexes $\{I_1, I_2, ..., I_k\}$. Finally, the client estimates its own position by computing the centroid of the k locations.

2) Privacy Analysis: From the client's perspective, although the server gets a list of encrypted squared Euclidean distances $\{[\![d_i]\!]\}_{i=1}^{M}$, it can not find the k nearest neighbors. Furthermore, the Location Retrieval Phase is performed only at the client side; thus the naive scheme is resistant to **Location Privacy Attack I**. On the other hand, as the client's WiFi fingerprint sent to the server is encrypted, the server can not recover it without the client's private key; thus the naive scheme is resistant to **Location Privacy Attack II**.

Though the naive scheme can protect the client's location privacy, the service provider's data privacy can be threatened by a malicious client through the following two ways:

Data Privacy Attack I: A malicious client randomly generates P WiFi fingerprints, denoted as $\{V'_h\}_{h=1}^{P}$, with $V'_h = \{v'_{h,1}, v'_{h,2}, ..., v'_{h,N}\}$. Then it gets the Euclidean distance $d_{h,i}$ between V'_h and V_i from the server. For each V'_h, the following M equations can be constructed:

$$\begin{cases} \|V'_h - V_1\| = \sum_{j=1}^{N} (v'_{h,j} - v_{1,j})^2 = d_{h,1} \\ \|V'_h - V_2\| = \sum_{j=1}^{N} (v'_{h,j} - v_{2,j})^2 = d_{h,2} \\ \quad ... \\ \|V'_h - V_M\| = \sum_{j=1}^{N} (v'_{h,j} - v_{M,j})^2 = d_{h,M} \end{cases} \quad (5)$$

Thus, the client gets $M \cdot P$ equations in total for $M \cdot N$ unknowns. If $P \geq N$, the malicious client can solve the equations and get $\{V_i\}_{i=1}^{M}$.

Data Privacy Attack II: A malicious client can get a fingerprint database D' similar to the server's database D in two steps. As a WiFi RSS is usually an integer between 0 and -90 (in unit of dBm), the malicious client can fabricate as many WiFi fingerprints as possible within the valid scope in the first step. For each fabricated WiFi fingerprint V'_h, the malicious client gets the squared Euclidean distance between V'_h and each fingerprint in D, and computes the corresponding location (x'_h, y'_h). Then the malicious client adds $\langle (x'_h, y'_h), V'_h \rangle$ to its own database D'. However, if the malicious client uses D' for indoor localization, the performance may decline dramatically compared with that of using D, because many WiFi fingerprints in D' do not exist in real-world environment, even taking the RSS variation into consideration. We refer these fingerprints as *noisy fingerprints*, which can bring large localization errors.

Thus in the second step, the malicious client filters out the noisy fingerprints. There always exists a transient variation in WiFi RSS due to dynamic changes in the environment, such as that caused by a nearby moving object [3]. Assume that the variation is within ε [14] [15]. Then the malicious client can conclude that a WiFi fingerprint V'_h does not exist even taking the WiFi RSS variation into consideration if

$$\text{for } \forall i \in \{1, 2, ..., M\}, d_{h,i} > N\varepsilon^2, \quad (6)$$

i.e., a WiFi fingerprint V'_h does not exist if the squared Euclidean distance between itself and any record in D is higher than $N\varepsilon^2$. Filtering out such noisy fingerprints from D' yields a fingerprint database similar to D.

C. The Complete PriWFL Scheme

To thwart the privacy attacks described above, we enhance the naive PriWFL from the following three aspects: i) each time the server receives a request, it randomly chooses N' APs from N APs to calculate the squared Euclidean distance; ii) the server blinds the Euclidean distance by a random number; and iii) the client uses N random numbers to mask its WiFi fingerprint. In the following subsections, we first conduct a measurement study to investigate the impact of N' on the localization accuracy; then we present the details of the scheme and prove that the client can correctly retrieve its location; and finally we present our privacy analysis.

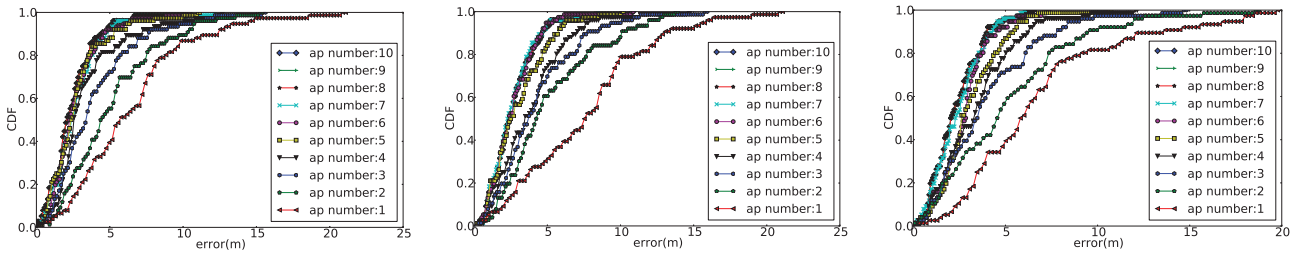
1) A Measurement Study of WiFi Fingerprint-Based Localization: The purpose of this study is to investigate the impact of N' (namely the number of APs) on the localization accuracy. We conduct an experiment in three real-world environments. During the training phase, we partition the indoor space into a mesh of $4m^2$ grids, and measure the WiFi fingerprints at each grid. During the operating phase, we utilize a conventional smartphone to sample the WiFi RSS for localization at various positions. To estimate the location, we use 3 nearest neighbors ($k = 3$).

We employ the localization error to quantify accuracy. Figure 2 presents the cumulative distribution function (CDF) of the localization error when the number of APs is varied from 1 to 10. It is observed that the localization accuracy is significantly improved when the number of APs increases from 1 to 6. However, when the number of APs goes beyond 6, the improvement of localization accuracy becomes negligible. This implies that more number of APs does not bring more information to localization when granularity of the fingerprint database remains unchanged because the k nearest neighbors in the database does not change.

These experimental results indicate that the localization accuracy improves with the increase of the number of APs. However, when the number of APs reaches a "threshold" (6 for our experimental environment), it does not obviously increase. Nowadays, WiFi APs are widely and densely deployed in many indoor environments (e.g., office buildings, shopping malls, and airport terminals). Thus within a single place, a device can hear tens of APs, which implies that the dimension N of the WiFi fingerprints is quite large. Since a large N does not improve the localization error, the server can randomly selects N' ($6 \leq N' \leq N$ in our experiment) APs to locate the client without decreasing the localization accuracy.

2) The Complete PriWFL Protocol: Like its naive version, the complete PriWFL consists of four phases, among which the first phase remains unchanged. Thus in the following we only detail the other three phases.

Preparation Phase: When a client needs to be located, it first generates a pair of keys $< PK_c, PR_c >$



(a) The location accuracy in building 1. (b) The location accuracy in building 2. (c) The location accuracy in building 3.

Fig. 2. The impact of the number of APs on localization accuracy.

using the Paillier cryptosystem, and publishes the public key PK_c. Then it selects N big random numbers $\{r_1, r_2,...,r_N\}$ and encrypts $\{-2v'_1, ... -2v'_N\}$, $\{v'^2_1 + r_1, v'^2_2 + r_2, ..., v'^2_N + r_N\}$ using PR_c. Finally, the client sends a request containing $\{[\![-2v'_1]\!], [\![-2v'_2]\!], ..., [\![-2v'_N]\!]\}$ and $\{[\![v'^2_1 + r_1]\!], [\![v'^2_2 + r_2]\!], ..., [\![v'^2_N + r_N]\!]\}$ to the server.

Distance Computation Phase: After receiving the client's request, the server first checks whether the client has submitted all the encrypted WiFi fingerprint components. If not, the server ignores the request. Otherwise, the server generates a random number R, and randomly chooses N' dimensions, denoted as $C = \{c_1, c_1, ..., c_{N'}\}$, from the original N dimensions in the WiFi fingerprint database D. Then the server computes $[\![S'_{i,1}]\!]$, $[\![S'_{i,2}]\!]$, $[\![S'_3]\!]$, and $[\![d'_i]\!]$ (for $1 \le i \le M$):

$$[\![S'_{i,1}]\!] = \left[\!\!\left[\sum_{j \in C} v^2_{i,j} \right]\!\!\right] \quad (7)$$

$$[\![S'_{i,2}]\!] = \prod_{j \in C} [\![-2v'_j]\!]^{v_{i,j}} \quad (8)$$

$$[\![S'_3]\!] = \prod_{j \in C} [\![v'^2_j + r_j]\!] \quad (9)$$

$$\begin{aligned} [\![d'_i]\!] &= [\![S'_{i,1} + S'_{i,2} + S'_3 + R]\!] \\ &= [\![S'_{i,1}]\!] \cdot [\![S'_{i,2}]\!] \cdot [\![S'_3]\!] \cdot [\![R]\!] \end{aligned} \quad (10)$$

Finally, the server returns $\{[\![d'_i]\!]\}_{i=1}^M$ to the client.

Location Retrieval Phase: After receiving $\{[\![d'_i]\!]\}_{i=1}^M$, the client decrypts the ciphertexts and estimates its location using the same method as the naive scheme. Its correctness is guaranteed by the following theorem.

Theorem 1. *Given d'_e and d'_f returned by the server for one request. If $d'_e > d'_f$, we have $d_e > d_f$, where d_e and d_f are the squared Euclidean distances between V_e and V', and between V_f and V', over N' dimensions, respectively.*

Proof:

$$\begin{aligned} & d'_e - d'_f \\ = \; & (S'_{e,1} + S'_{e,2} + S'_3 + R) - (S'_{f,1} + S'_{f,2} + S'_3 + R) \\ = \; & (\sum_{j \in C} v^2_{e,j} + \sum_{j \in C} (-2v_{e,j} \cdot v'_j) + \sum_{j \in C} (v'^2_j + r_j)) - \\ & (\sum_{j \in C} v^2_{f,j} + \sum_{j \in C} (-2v_{f,j} \cdot v'_j) + \sum_{j \in C} (v'^2_j + r_j)) \\ = \; & (\sum_{j \in C} v^2_{e,j} + \sum_{j \in C} (-2v_{e,j} \cdot v'_j) + \sum_{j \in C} v'^2_j) - \\ & (\sum_{j \in C} v^2_{f,j} + \sum_{j \in C} (-2v_{f,j} \cdot v'_j) + \sum_{j \in C} v'^2_j) \\ = \; & d_e - d_f \end{aligned}$$

Thus if $d'_e > d'_f$, we have $d_e > d_f$. ■

3) Privacy Analysis: Though $\{[\![d'_i]\!]\}_{i=1}^M$ is computed by the server, the server is not able to pick out the k nearest neighbors, because it does not have the private key PR_c. Therefore the server cannot figure out the client's location, guaranteeing the client's location privacy. Further, the Location Retrieval Phase is performed only at the client side, thus the complete PriWFL is resistant to **Location Privacy Attack I**.

In the proposed PriWFL scheme, the client submits $\{[\![-2v'_1]\!], [\![-2v'_2]\!], ..., [\![-2v'_N]\!]\}$ and $\{[\![v'^2_1 + r_1]\!], [\![v'^2_2 + r_2]\!], ..., [\![v'^2_N + r_N]\!]\}$ to the server. The server cannot recover $\{v_1, v_2, ..., v_N\}$. Namely, PriWFL is resistant to **Location Privacy Attack II** according to the following Theorem.

Theorem 2. *Given $[\![-2v'_j]\!]$ and $[\![v'^2_j + r_j]\!]$, the server cannot recover v'_j, if r_j is kept secret.*

Proof: After getting $[\![-2v'_j]\!]$ and $[\![v'^2_j + r_j]\!]$, the server can compute the following equation:

$$\left[\!\!\left[(-2v'_j) \cdot \left(-\frac{1}{2} v'_j \right) \right]\!\!\right] = [\![v'^2_j + r_j - r_j]\!] \quad (11)$$

By applying the homomorphic property of the Paillier cryptosystem, we can transform (11) into the following equation:

$$[\![-2v'_j]\!]^{\left(-\frac{1}{2}v'_j\right)} = [\![v'^2_j + r_j]\!] \cdot [\![-r_j]\!] \quad (12)$$

In (12), r_j is a big random number generated by the client. Thus even the server knows $[\![-2v'_j]\!]$ and $[\![v'^2_j + r_j]\!]$, it still cannot recover v'_j. ■

The server's data privacy is preserved according to the following two theorems.

Theorem 3. *The proposed PriWFL is resistant to **Data Privacy Attack I**.*

Proof: To get the WiFi fingerprint database, a malicious client should generate at least N distinct WiFi fingerprints $\{V'_h\}_{h=1}^N$. However, for any V'_h submitted by the client, the server randomly chooses N' APs to compute the Euclidean

distance, where N' is a secret number that is less that N but is greater than a threshold τ. Thus (5) has $A = C_N^{\tau} + C_N^{\tau+1} + ... + C_N^N$ possible combinations for a malicious client to try. For $\{V_h'\}_{h=1}^N$, there are in total A^N combinations, resulting in A^N solutions. We claim that this is computationally infeasible, even with the most powerful computer in today's market. For example, when $\tau = 6$ and $N = 10$ as in our experiment, $A^N \approx 7.34 \times 10^{25}$. Today's most powerful computer achieves a processing speed of 33.86 petaflops per second [16]. This implies that it will take hundreds of years to solve (5) with the best computer, indicating that PriWFL is computationally secure (practically secure). ■

Theorem 4. *The proposed PriWFL is resistant to* ***Data Privacy Attack II****.*

Proof: Though a malicious client may exhaustively list all possible WiFi Fingerprint combinations and get the corresponding locations, it cannot filter out the noisy fingerprints through (6) because the server masks the squared Euclidean distance by a random number R for each request. The localization performance using D' is poor, due to the low localization accuracy (As shown in section V-D) . ■

D. Extending PriWFL to a multi-building scenario

Within a single building, the number of APs (i.e. N) and the total number of WiFi fingerprints in a database (i.e. M) are small. If a service provider wants to provide the localization service in a town-scale scenario, N and M become very large, leading to a heavy computational overhead both at the server side and the client side.

Assume that there are in total S buildings, denoted as $\{B_1, B_2, ..., B_S\}$. We further assume that within each building B_i, there are at least N APs within reach at any position, denoted as $\{AP_{ij}\}_{j=1}^N$. To achieve scalability, we reduce the dimension of the fingerprints in each building from $N \cdot M$ to N, with each new dimension denoting the RSS value from one of the N reachable APs within the building. In the Preprocess phase, the server distributes metadata to the client. The metadata consists of three tables $T_1 = \langle i, (x_i, y_i)\rangle_{i=1}^M$, $T_2 = \langle i, (a_i, b_i)\rangle_{i=1}^S$, and $T_3 = \langle i, \{AP_{i,j}\}_{j=1}^N\rangle_{i=1}^S$. Here T_1 stores the indexes and all the locations where the WiFi fingerprints are measured, and T_2 indicates that the locations $\langle l, (x_l, y_l)\rangle_{l=a_i}^{b_i}$ belong to building B_i. From T_3, the client gets to know which APs can be used for localization and how to line up the sensed WiFi fingerprint with the fingerprints in the database. Assume that the client is in building B_u, which can be determined by GPS before entering the building. In the Preparation phase, the client informs the server of w buildings it may reside and the server just needs to calculate the distance between the submitted WiFi fingerprint and the WiFi fingerprints in the w buildings. It is a tradeoff between efficiency and privacy. A larger w leads to a higher computational overhead, but implies a higher guarantee for location privacy. After receiving $\{[\![d_i']\!]\}_{i=1}^M$, the client only needs to decrypt $\{[\![d_i']\!] \,|\, a_u \le i \le b_u\}$, with a_u and b_u being retrieved from Table T_2.

IV. Exploiting Indoor Mobility Prediction to Reduce the Client's Computational Overhead

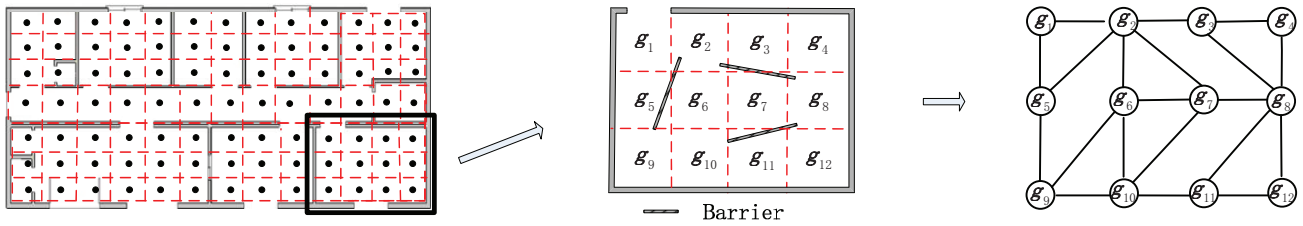
In the Location Retrieval phase described above, a client in building B_u needs to decrypt $\{[\![d_i']\!] \,|\, a_u \le i \le b_u\}$. This process is still time-consuming especially for resource-constrained devices such as smartphones. According to [17], it costs more than 10 seconds to decrypt $[\![d_i']\!]$ for $1 \le i \le 100$ on an ordinary smartphone. Furthermore, for some applications such as indoor navigation, a client needs to make frequent localization requests, leading to a high computational overhead. To overcome this problem, we propose an indoor mobility prediction algorithm, which outputs the set of possible locations (denoted by $LocSet$) at time T_{i+1} for a client based on its location at time T_i. Then the device only needs to decrypt $\{[\![d_i']\!] \,|\, (x_i, y_i) \in LocSet\}$. The design of the algorithm is motivated by the following three observations. First, a client's mobility in an indoor environment is constrained by the indoor layout. For example, it's impossible for a client to walk through walls. Second, limited by walking speed, a client cannot move a long distance in a short time interval. Third, a client can learn the indoor layout through the indoor map downloaded from the localization service provider.

For simplicity, we take Figure 3 as an example to illustrate our algorithm. Figure 3(a) is the integrated layout of an office building, in which the black spots represent locations where the recorded WiFi fingerprints are obtained. We partition the layout into a mesh of grids, ensuring that there is only one black spot at every grid. Figure 3(b) demonstrates the detailed layout of one room in the building. We model the indoor layout through a graph G, in which each vertex stands for a grid. If two grids g_i and g_j are adjacent to each other, and there is no barrier (such as a furniture or a wall) between them, then there is an edge linking g_i and g_j. A client can at most move from one grid to one of its adjacent neighbors within $\triangle t$, which depends on the client's walking speed and the grid size. The graph representation corresponding to the room in Figure 3(b) is shown in Figure 3(c), which is a subgraph of G.

Assume that a client resides in grid g_s at time t. After $\triangle t$, the client may move from g_s to one of the neighboring grids, or stay in g_s. Thus $LocSet = Nei_s \cup \{g_s\}$ (at time $t + \triangle t$), where Nei_s denotes the neighbors of g_s. For example, assume that the client is in grid g_{12} at time t in Figure 3(c). After $\triangle t$, the client may move to grid g_{11}, g_8, or stay in g_{12}. Then $LocSet = \{g_{11}, g_8, g_{12}\}$. After $2\triangle t$, the possible location set for this client is $LocSet = \{g_{12}, g_{11}, g_{10}, g_8, g_3, g_4\}$. If the client requests its location at time $t + 2\triangle t$, it only needs to decrypt $\{[\![d_i']\!] \,|\, (x_i, y_i) \in LocSet\}$ in the Location Retrieval phase to find out the k nearest neighbors and estimate its current location. We summarize this algorithm in Algorithm 1.

V. Performance Evaluation

In this section, we report the results of our theoretical analysis and experimental study to demonstrate the performance of PriWFL. As the Pre-Process phase does not involve any computation, we only evaluate the other three phases. Table I summarizes the theoretical analysis results of PriWFL,



(a) The layout of the building (b) The detailed layout of one room in (a) (c) Graph description of (b)

Fig. 3. Building layout, the black spots in (a) are the locations where the fingerprints are trained.

Algorithm 1 Mobility prediction algorithm

Input:

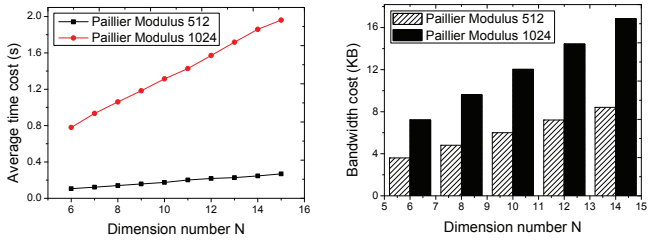
G; g_s - the grid the client resides at time T_i

Output:

$LocSet$ - the client's possible locations at time T_{i+1}

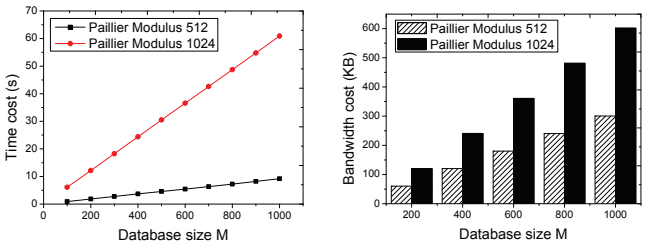
$n = \lceil (T_{i+1} - T_i) / \triangle t \rceil$
$Q = LocSet = \{g_s\}$
for $j \leftarrow 1$ to n **do**
  $P = \emptyset$
  for each g_i in Q **do**
    Get the neighbor list Nei_i of g_i
    $P = P \cup Nei_i$
    $LocSet = LocSet \cup Nei_i$
    $Q = Q - \{g_i\}$
  end for
  $Q = P$
end for
return $LocSet$;

where Exp denotes one modular exponentiation, Mul denotes one modular multiplication, and L is the bit length of a ciphertext. To evaluate the performance in practical settings, we implement the client side of PriWFL on a Android platform with a Qualcomm Snapdragon600 quad-core 1.7GHz CPU and 2G RAM, and the server side on a 32-bit computer with Intel i7 CPU of 3.4 GHz and 4G memory. The database we use contains $M = 1000$ WiFi fingerprints, among which 230 records are trained in three different office buildings, and the rest are randomly generated. The total number of APs is 15.



(a) Time cost. (b) Bandwidth cost.

Fig. 4. Cost of the Preparation phase.



(a) Time cost. (b) Bandwidth cost.

Fig. 5. Cost of the Distance Computation phase.

A. Cost of the Preparation Phase

We first measure the computation time and bandwidth cost in the preparation phase under different parameter settings. Figure 4(a) shows the relationship between the time cost and N under different Paillier Modulus. We observe that i) for Paillier Modulus 512 the computation time increases from 0.106s to 0.269s when N is increased from 6 to 15; ii) using Paillier Modulus 1024 results in a larger time cost than using Paillier Modulus 512; and iii) the time cost ranges from 0.781s to 1.960s when N is increased from 6 to 15.

At the end of this phase, the client sends the encrypted data to the server. The bandwidth cost is shown in Figure 4(b). When N is increased from 6 to 14, the consumed bandwidth ranges from 3.605KB to 9.012KB for Paillier Modulus 512, and from 7.218KB to 16.840KB for Paillier Modulus 1024.

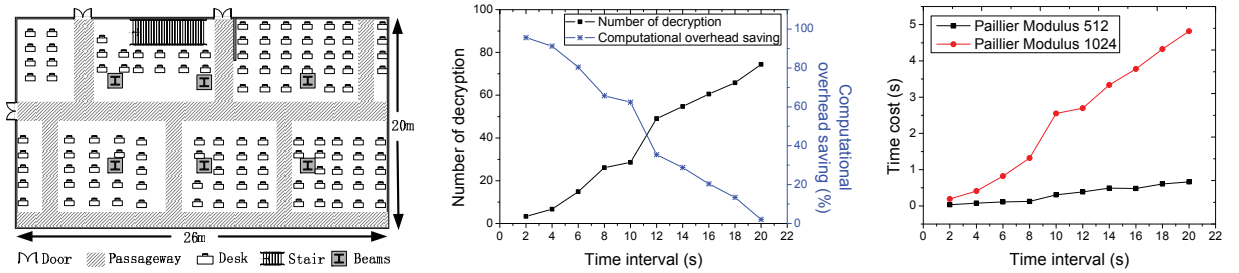
B. Cost of the Distance Computation Phase

In this phase, the server randomly selects N' ($6 \leq N' \leq N$) dimensions to compute the encrypted distance between the client's WiFi fingerprint and each fingerprint in the database. We set $N = 15$, and investigate the impact of database size M on the time cost and bandwidth cost in the Distance Computation Phase. Figure 5(a) shows the computational overhead under different database size M. It is observed that when M increases from 100 to 1000, the computation time increases from 0.911s to 9.203s for Paillier Modulus 512, and from 6.099s to 60.907s for Paillier Modulus 1024.

After computation, the server sends the encrypted distance back to the client. As depicted the Table I, the communication overhead is linear to the size of the WiFi fingerprint database. Figure 5(b) shows the relationship between the communication

TABLE I
COMPLEXITY ANALYSIS RESULTS

Phases	Computation		Communication (in bits)
	Client	Server	
Preparation Phase	$2N \cdot Exp + 4N \cdot Mul$	$none$	$2N \cdot L$
Distance Computation Phase	$none$	$M \cdot (N'+1) \cdot Exp + M \cdot (2N'+3) \cdot Mul$	$M \cdot L$
Location Retrieval Phase	$\lvert LocSet \rvert \cdot Exp$	$none$	$none$



(a) Layout of the testing floor. (b) The performance of mobility prediction. (c) Time cost in Location Retrieval Phase.

Fig. 6. The cost of the Location Retrieval Phase.

overhead and M. We observe that as M increases from 100 to 1000, the bandwidth cost ranges from 30.049KB to 300.445KB for Paillier Modulus 512, and from 60.142KB to 601.438KB for Paillier Modulus 1024.

C. Cost of the Location Retrieval Phase

To reduce the computational overhead, we propose the indoor mobility algorithm, which exploits the location at time t_i to predict the possible locations ($LocSet$) at time $t_i + 1$. In this subsection, we first evaluate the effectiveness of our prediction algorithm. For this purpose we conduct an experiment on the first floor of an office building. Figure 6(a) shows the layout of the floor, which covers approximately 520 (26×20) square meters. We partition the floor into 76 grids, ensuring there is only one WiFi fingerprint measured at a single grid. Assume that within each second, one can move at most from a grid to one of the adjacent grids. In our experiments, we randomly select the location at time t_i, and increase the time interval between t_i and t_{i+1} from 2s to 20s. The experiments are repeated 100 times and the averaged results are reported. As demonstrated in Figure 6(b), when the time interval between two localization requests is 2 seconds, the client performs 3.3 decryptions in average, saving about 96% computational overhead in the Location Retrieval phase. With the increase of the time interval, the size of the client's possible location set $LocSet$ becomes larger. For example, when the interval reaches 20 seconds, the client's possible location set $LocSet$ contains 74 positions.

Figure 6(c) demonstrates the relationship between the time cost in the Location Retrieval phase and the time interval under different Paillier Modulus. When using Paillier Modulus 512, the client can estimate its location within 1s; when using Paillier Modulus 1024 the time gets larger but it is always less than 5s.

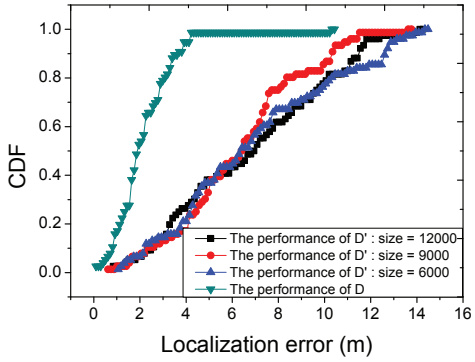


Fig. 7. The effectiveness of PriWFL in defending Data Privacy Attack II.

D. The effectiveness of PriWFL

As described above, PriWFL can protect a client's location privacy and is resistant to Data Privacy Attack I; but this can not be verified through experiments. Thus, we evaluate the effectiveness of PriWFL when defensing Data Privacy Attack II. In our experiments, the database contains 230 WiFi fingerprints collected in three office buildings. Each WiFi fingerprint is a 10-dimensional vector. The database D' is constructed as follows. In the first step, we randomly generate a WiFi fingerprint V' and query the localization system to get the distances between V' and all the items in D, i.e. $\{d_i\}_{i=1}^{M}$. We further estimate the location (x', y') corresponding to V'. In the second step, we check if all the distances are greater than $10 \times \varepsilon^2$, where $\varepsilon = 5$. If false, $\langle (x', y'), V' \rangle$ is inserted into D'; if true, V' is regarded as a noisy fingerprint. As it is computationally infeasible for a malicious client to exhaustively list all possible fingerprints, we set $|D'|$ to be 6000, 9000, and 12000. The localization accuracy of using D and D' is shown in Figure 7. We observe that the performance of using D' is significantly poorer than that of using D. Using D', nearly 50% of the localization errors are larger than $7m$,

which is unacceptable for many application scenarios [3]. We also observe that the location accuracy does not improve as the size of D' increases. Thus we claim that PriWFL can defend against Data privacy Attack II effectively.

VI. Related Work

In this section, we discuss two lines of related research: location privacy and biometric identification.

A. Location Privacy in Location-Based Services

To protect the client's location privacy in Location-Based Services, several approaches have been proposed. The most popular approach is k-anonymity [7] [8], which provides a form of plausible deniability by ensuring that the client cannot be individually identified from a group of k clients. Mix Zone [9] is another popular approach. It divides the whole region into application zones and mix zones. Clients report their locations in application zones, and receive new, unused pseudonyms at mix zones. This helps to prevent an attacker from linking pseudonyms because a new pseudonym could have been assigned to anyone in a mix zone.

The above-mentioned approaches rely on a trusted third party, and they all assume that clients know their own locations. In this paper, we address the privacy issues when localizing a to-be-localized client based on RSS fingerprints; thus this work is orthogonal to the existing research in location privacy for LBSs.

B. Privacy in Biometric Identification

To protect the clients' privacy in the process of biometric identification, Erkin et al. [18] proposed a privacy-preserving face recognition system, which hides the client's biometric and the comparison result from the server that performs the matching operation. Huang et al. [10] proposed a privacy-preserving biometric identification protocol with their novel garbled circuit design and the ciphertext packing technique. Yuan et al. [11] proposed a privacy-preserving biometric identification scheme that achieves efficiency by exploiting the power of cloud computing.

All these schemes are proposed to find only the closest match. When being extended to k nearest neighbor search, multi-round computation and communication are required to retrieve the records corresponding to the k closest matches. In WiFi fingerprint-based localization, k "closest fingerprints" need to be identified to derive the client's location.

VII. Conclusions and Future Research

In this paper, we analyze the privacy issues of WiFi fingerprint-based localization. To thwart the privacy threats, we propose a novel Privacy-Preserving WiFi Fingerprint Localization scheme (PriWFL), which utilizes homomorphic encryption to protect both the client's location privacy and the localization service provider's data privacy. Furthermore, we design an indoor mobility prediction algorithm to reduce the computational overhead at the client side. To validate PriWFL, we perform extensive theoretical performance analysis, implement PriWFL in a typical Android platform, and carry out extensive experimental study. Our results demonstrate the effectiveness and practicality of PriWFL.

Acknowledgement

This work was supported by National High Technology Research and Development Program of China (Grant No. 2013AA011102), "Strategic Priority Research Program" of the Chinese Academy of Sciences (Grant No. XDA06040100), the Innovation Program of Institute of Information Engineering, Chinese Academy of Sciences (Grant No. Y3Z0071G02), the National Natural Science Foundation of China (Grant No. 60933011, 61003218, 61272444), and the National Science Foundation of the US (IIS-1343976, CNS-1318872).

References

[1] P. Bahl and V. N. Padmanabhan, "Radar: An in-building rf-based user location and tracking system," in *Proc. of IEEE INFOCOM*, 2000, pp. 775–784.

[2] Z. Yang, C. Wu, and Y. Liu, "Locating in fingerprint space: Wireless indoor localization with little human intervention," in *Proc. of ACM MobiCom*, 2012, pp. 269–280.

[3] H. Liu, Y. Gan, J. Yang, S. Sidhom, Y. Wang, Y. Chen, and F. Ye, "Push the limit of wifi based localization for smartphones," in *Proc. of ACM MobiCom*, 2012, pp. 305–316.

[4] W. Cheng, D. Wu, X. Cheng, and D. Chen, "Routing for information leakage reduction in multi-channel multi-hop ad-hoc social networks," in *WASA*, August 2012, pp. 31–42.

[5] R. Shokri, G. Theodorakopoulos, J. Le Boudec, and J. Hubaux, "Quantifying location privacy," in *IEEE Symposium on Security and Privacy*, 2011, pp. 247–262.

[6] L. Ma, A. Y. Teymorian, and X. Cheng, "A hybrid access point protection framework for commodity wi-fi networks," in *Proc. of IEEE INFOCOM*, 2008, pp. 1894–1902.

[7] D. Yang, X. Fang, and G. Xue, "Truthful incentive mechanisms for k-anonymity location privacy," in *Proc. of IEEE INFOCOM*, 2013, pp. 3094–3102.

[8] X. Liu, K. Liu, L. Guo, X. Li, and Y. Fang, "A game-theoretic approach for achieving k-anonymity in location based services," in *Proc. of IEEE INFOCOM*, 2013, pp. 3085–3093.

[9] A. R. Beresford and F. Stajano, "Mix zones: User privacy in location-aware services," in *Proc. of the IEEE PerSec*, 2004, pp. 127–131.

[10] D. Evans, Y. Huang, J. Katz, and L. Malka, "Efficient privacy-preserving biometric identification," in *Proc. of NDSS*, 2011.

[11] J. Yuan and S. Yu, "Efficient privacy-preserving biometric identification in cloud computing," in *Proc. of IEEE INFOCOM*, 2013, pp. 2752–2760.

[12] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Proc. of ACM EUROCRYPT*, 1999.

[13] I. Damgård and M. Jurik, "A generalisation, a simplication and some applications of paillier's probabilistic public-key system," in *Public Key Cryptography*. Springer, 2001, pp. 119–136.

[14] S. Yang, P. Dessai, M. Verma, and M. Gerla, "Freeloc: Calibration-free crowdsourced indoor localization," in *Proc. of IEEE INFOCOM*, 2013.

[15] A. Haeberlen, E. Flannery, A. M. Ladd, A. Rudys, D. S. Wallach, and L. E. Kavraki, "Practical robust localization over large-scale 802.11 wireless networks," in *Proc. of ACM MobiCom*, 2004.

[16] Top500, "Chinas tianhe-2 supercomputer takes no. 1 ranking on 41st top500 list," 2013, [Online; accessed 2013-06-17]. [Online]. Available: http://www.top500.org/blog/lists/2013/06/press-release/

[17] R. Zhang, J. Zhang, Y. Zhang, J. Sun, and G. Yan, "Privacy-preserving profile matching for proximity-based mobile social networking," *IEEE Journal on Selected Areas in Communications*, 2013.

[18] Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, I. Lagendijk, and T. Toft, "Privacy-preserving face recognition," in *Privacy Enhancing Technologies*. Springer, 2009, pp. 235–253.