Modeling Privacy in WiFi Fingerprinting Indoor Localization

Zheng Yang[1(✉)] and Kimmo Järvinen[2]

[1] ITrust, Singapore University of Technology and Design, Singapore, Singapore
zheng_yang@sutd.edu.sg
[2] Department of Computer Science, University of Helsinki, Helsinki, Finland
kimmo.u.jarvinen@helsinki.fi

Abstract. In this paper, we study privacy models for privacy-preserving Wifi fingerprint based indoor localization (PPIL) schemes. We show that many existing models are insufficient and make unrealistic assumptions regarding adversaries' power. To cover the state-of-the-art practical attacks, we propose the first formal security model which formulates the security goals of both client-side and server-side privacy beyond the curious-but-honest setting. In particular, our model considers various malicious behaviors such as exposing secrets of principles, choosing malicious Wifi fingerprints in location queries, and specifying the location area of a target client. Furthermore, we formulate the client-side privacy in an indistinguishability manner where an adversary is required to distinguish a client's real location from a random one. The server-side privacy requires that adversaries cannot generate a fabricate database which provides a similar function to the real database of the server. In particular, we formally define the *similarity* between databases with a ball approach that has not been formalized before. We show the validity and applicability of our model by applying it to analyze the security of an existing PPIL protocol.

Keywords: Indoor localization · Wifi fingerprint
Security model · Privacy

1 Introduction

People spend significant amounts of their time in public indoor environments including shopping malls, libraries, airports, university campuses, etc. This has boosted the interest towards various indoor location-based applications [6,15] such as indoor-navigation or elderly assistance and emergency responding. However, in an indoor environment, the traditional Global Positioning System (GPS) may be not available due to weak signal strengths caused by blocking constructions. To obtain a location in a building, a client has to rely on certain *indoor location services* (ILS) provided by some server of the building. The most widely used approach for ILS is the one based on the Wifi fingerprinting technique [5,7–9,11,14,18,20,21]. This method is very effective and popular because it uses an existing Wifi infrastructure of a building. For a Wifi fingerprint based ILS, the server holds a geo-location database (e.g. [22, Table 1]) containing signal strengths of Wifi access points (AP) in various reference locations, as explained in Sect. 3. Roughly speaking, a client measures the signal strengths of Wifi APs in the client's current (unknown) location and send them to the server. The server calculates the client's location based on the geo-location database, e.g., by calculating the k-nearest Euclidean distances between the client's input and reference fingerprints in the database. Finally, the server sends the location to the client. However, this naive solution cannot prevent a malicious server from tracking its clients' locations, which of course violates the clients' privacy.

Recently, several solutions, e.g. [12,13,22,24], have been proposed to protect the clients' location privacy in ILSs. However, only a few pieces of research (e.g. [24]) have included a formal security model for privacy-preserving indoor localization (PPIL) schemes. This deficiency has resulted in the development of flawed protocols (e.g. [13,24]) which may take years to discover. Therefore, applying PPIL schemes without rigorous security proofs is inherently risky. For example, in INFOCOM 2014, Li et al. [13] presented a Wifi fingerprint localization system called PriWFL which was claimed to provide both clients' location privacy and server's database privacy (which will be referred to as client-privacy and server-privacy for short, respectively). PriWFL is based on the 'honest-but-curious' setting where the adversary does not change the protocol execution between an honest client and the server. Client-privacy roughly states that no passive adversary (including the server) can infer the honest client's location after intercepting all protocol messages. Server-privacy requires that a malicious client cannot use location queries for compromising the server's database. However, Yang and Järvinen [22] recently unveiled a practical attack (which will be called as chosen fingerprint attack) for breaking the server-privacy of PriWFL. In this chosen fingerprint attack, the malicious client chooses special fingerprints, such as all-zeros or single-one fingerprints, to compromise the whole server's database. Unfortunately, their attack idea can be applied to break also the protocol recently proposed by [24], as shown in [23]. One of the major problems here is that the server-privacy defined in [13,24] cannot cover the malicious client attack of [22]. Hence, PriWFL has not been provably demonstrated to provide security against such attack (due to lack of formal definitions). Namely, the curious-but-honest setting is not enough for proving the security for PPIL schemes.

To fix the problem of PriWFL, Yang and Järvinen proposed a new PPIL scheme (which will be referred to as YJ scheme) that relies on Paillier's public key encryption (PKE) [17] and garbled circuits based secure evaluation function (SFE). Intuitively, the YJ scheme satisfies both client- and server-privacy. However, we notice that its security is only informally justified in [22] without being analyzed under an appropriate security model. Hence, there are still open questions: (i) how many active attacks it can withstand and (ii) what the security assumptions of its build blocks and the corresponding security reductions should be. The primary motivation for this work is to develop a formal security model that allows formal analysis of the security of practical PPIL protocols.

We stress that the definitions on client- and server-privacy respectively are fundamental to the success of 'provably secure' PPIL schemes. It is therefore highly desirable to define a security model to cover the state-of-the-art attacks so that their securities can be formally proved to satisfy the security goals. Recently, Zhang et al. [24] made an effort to formulate the client- and server-privacy in a curious-but-honest setting. The definitions of client- and server-privacy in [24] can be seen as extensions from that in [13]. In the location privacy attack [24, Definition 1], a successful adversary should compromise either a client's Wifi fingerprint or location in a query. However, in practice, an adversary may violate client-privacy via learning (for instance) sensitive information about whether the client appeared at some place or its whereabouts, even without knowing its exact location or fingerprints. In particular, the definition of server-privacy is still vague in [24]. I.e., 'a certain level of accuracy' (in [24, Definition 2]) regarding ILS provided by an adversary is not clearly formalized. Specifically, there is no way to measure the accuracy of an adversary's ILS as there is no security experiment or any formulation about the adversary's advantage on breaking either client- or server-privacy. Furthermore, several important practical attacks are not modeled in [24]) such as: (i) chosen fingerprint attack introduced by Yang and Järvinen [22], (ii) known location attack (e.g. whether knowledge of an exposed (historical) location of a client affects the client's unexposed locations), and (iii) known sub-area attacks (e.g. a follower is curious about the direction of movement or location of a client within a specific area). It is still an open question on modeling these malicious attacks. Hence, we conclude that Zhang et al.'s model is rather weak and informal and it is not possible to give a thorough security analysis for a PPIL protocol using such model.

Our Results. In this paper, we present the first *unilateral-malicious* security model for Wifi fingerprint-based PPIL schemes to solve the open problems in existing models. Generally speaking, the unilateral-malicious setting is stronger than the traditional semi-honest setting but weaker than the fully malicious setting. In the unilateral-malicious setting, we particularly formulate the malicious behaviors relative to clients' sessions, e.g., manipulating Wifi fingerprints and exposing locations. We require the server to behave in semi-honest manner (for simplicity). Namely, the server may be curious about a client's location, but it should honestly run the protocol instance in order to provide a good service. We can weaken the security requirement of the server since a server's malicious behaviors (e.g., dishonest executions) would be easily caught in practice (and substantially punished) due to providing poor ILS. If the service is poor, then clients would likely just stop using the service and, consequently, make such an attack impossible. However, the server cannot easily identify a client's malicious behaviors. This is true especially when the client's messages are (non-deterministically) encrypted by its own public key. Hence, we define the first practical formal PPIL security model that focuses on modeling the most harmful malicious behaviors on the client side. We specifically apply our new security model to analyze the YJ scheme (as an example) to not only show the validity of our security model but also to exhibit another attractiveness of the YJ scheme in its provable security.

We consider the security model in a simulation environment (which covers the real world applications) with a single server and multiple clients, where each client may have multiple sessions for querying different locations. Unlike previous work [13,22,24], we formulate the attacks of an adversary via a series of oracle queries. Each query stands for a generic class of attacks. Under the unilateral-malicious setting, we assume that the adversary can only run protocol instances between the client and server by following the protocol specification. In spite of that, several important active attacks are defined via a series of oracle queries allowing an adversary to manipulate and learn sensitive information of sessions. Namely, an adversary can specify sessions' initial states such as Wifi fingerprint and target location area, record her own RSS measurements, or reveal a principal's long-term or ephemeral secret key and a client's location. The details of these queries can be found in Sect. 3.

The security goal of client-privacy is defined in an indistinguishable manner following the approach in [3]. Namely, a PPIL scheme is said to be client secure (informally) if no polynomial time adversaries can distinguish the location of an unexposed session from a random location. Whereas the security goal of server-privacy is achieved (informally) if all polynomial time adversaries are unable to generate a database D' which can provide a similar function of the server's real database D. A key problem required to be resolved is to formulate the notion of 'similar function'. Here we adopt a ball approach. Informally speaking, we say that the fabricated database D' generated by an adversary has a similar function to the real database D, if D' results in a fabricated location L' within a small ball that is centered at the corresponding real location L (which is calculated based on D for a certain location query) with a pre-defined *radius* ρ for most of the distinct location queries. Furthermore, each security goal is associated with a corresponding security experiment which defines the interactions between adversary and experiment simulator (challenger), rules of the adversary (on launching various attacks), and winning condition of the adversary. Eventually, we carefully define the client- and server- privacy in conjunction with the adversarial model, security experiment, and the corresponding adversaries' winning advantages. Here define a security model mainly for the Wifi fingerprint database. However, our security definitions and the adversarial model might be still generic enough to address the security for different kinds of PPIL schemes. It is not hard to see that the key elements (or formulation ideas) of our security model, such as adversary model, security experiment, and security definitions, can be simply applied to formulate other types of databases with small changes.

In the security analysis of the YJ scheme, we first show that the client-privacy can be linearly reduced to that of Paillier PKE and SFE. We also show that the YJ scheme does not leak any useful information about a server's database to the adversaries due to the large enough randomness space, and the security of SFE. Since adversaries cannot gain overwhelming advantages from the messages of YJ protocol, the security of the database is therefore determined by the secret entropy of the database itself.

Organization. The remainder of this paper is organized as follows. The security assumptions on the building blocks of the YJ scheme are reviewed in Sect. 2. In Sect. 3, we introduce a new security model for PPIL protocols. In Sect. 4, we review the YJ scheme and introduce the security analysis under our proposed model. Finally, we give conclusion remarks in Sect. 5.

2 Preliminaries

General Notations. We let $\kappa \in \mathbb{N}$ be the security parameter and 1^κ be a string of κ ones. Let $[n] = \{1, \ldots, n\} \subset \mathbb{N}$ denote the set of integers. Let $a \overset{\$}{\leftarrow} S$ denote the operation sampling a uniform random element from a set S. We use $\|$ to denote the concatenation operation of two strings. Let $|\cdot|$ denote an operation calculating the bit-length of a string, and # denote an operation calculating the number of elements in a set.

Paillier Public Encryption Scheme. Paillier public-key encryption (PKE) scheme [17] is a probabilistic encryption scheme. Let $\mathsf{PrimG}(\kappa)$ be a function which generates a set of primes of length κ. The Paillier PKE scheme mainly consists of the following three algorithms:

– **Key Generation** (KeyGen). Given the security parameter 1^κ, the algorithm chooses two large primes $p, q \overset{\$}{\leftarrow} \mathsf{PrimG}(\kappa/2)$, and computes $n = p \cdot q$. It also selects a group generator g for the multiplicative group $\mathbb{Z}^*_{n^2}$, such that the order of g is a non-zero multiple of n. The public key pk is a tuple (n, g) and the secret key sk is $\lambda = \mathrm{lcm}(p-1, q-1)$. This algorithm returns (pk, sk).
– **Encryption** (Enc). This algorithm takes a message $m < n$ and a public key (n, g) as inputs. It selects a random value $r \overset{\$}{\leftarrow} [n]$, and computes the ciphertext: $C = g^m \cdot r^n \bmod n^2$. The output of this algorithm is C. For simplicity, we may omit modulus n^2 in the rest of the paper.
– **Decryption** (Dec). This algorithm takes $C < n^2$ and the secret key λ as inputs, and outputs $m = \frac{L(C^\lambda) \bmod n^2}{L(g^\lambda) \bmod n^2} \bmod n$ where $L(u) = \frac{u-1}{n}$.

Paillier PKE scheme is additively homomorphic over the group $\mathbb{Z}_n$. Namely, for two ciphertexts $C_1 = \mathsf{Enc}(pk, m_1)$ and $C_1 = \mathsf{Enc}(pk, m_2)$, we have that $\mathsf{Dec}(sk, C_1 \cdot C_2 \bmod n^2) = m_1 + m_2 \pmod n$ and $\mathsf{Dec}(sk, C_1 \cdot C_2^{-1} \bmod n^2) = m_1 - m_2 \pmod n$, where the inverse can be computed via the exponentiation $C_2^{-1} = C_2^{n^2-1} \bmod n^2$. Using the above homomorphic additions, it is also possible to compute multiplications and divisions by a scalar $t \in [n]$: $\mathsf{Dec}(sk, C_1^t \bmod n^2) = t \cdot m_1 \pmod n$ and $\mathsf{Dec}(sk, C_1^{t^{-1} \bmod n} \bmod n^2) = m_1/t \pmod n$, where $t^{-1} \bmod n$ can be computed with the Extended Euclidean Algorithm.

We review the security of Paillier PKE scheme via the following definition.

Definition 1. *The security experiment for a Paillier PKE scheme* $\mathsf{Pai}=(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *is defined in the following:*

$$\begin{aligned}
&\mathsf{EXP}^{\mathsf{ind-cpa}}_{\mathsf{Pai},\mathcal{B}}(\kappa):\\
&\quad b \overset{\$}{\leftarrow} \{0,1\},\ p,q \overset{\$}{\leftarrow} \mathsf{PrimG}(\kappa/2),\ n = p \cdot q; g \leftarrow \mathbb{Z}^*_{n_2},\\
&\quad (m_0, m_1) \leftarrow \mathcal{B}(n,g),\ s.t.\ |m_0| = |m_1| and\, 0 \leq (m_0, m_1) < n;\\
&\quad r_0, r_1 \overset{\$}{\leftarrow} [n-1],\ C_0 := g^{m_0} \cdot r_0^n \mod n_2,\ C_1 := g^{m_1} \cdot r_1^n \mod n_2;\\
&\quad b' \leftarrow \mathcal{B}(pk, C_b);\ if b = b' \,\mathrm{return}\, 1,\ otherwise\ return\ 0.
\end{aligned}$$

We define the advantage of $\mathcal{B}$ *in the above experiment as:* $\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathsf{Pai},\mathcal{B}}(\kappa) := \left|\Pr[\mathsf{EXP}^{\mathsf{ind\text{-}cpa}}_{\mathsf{Pai},\mathcal{B}}(\kappa) = 1] - \frac{1}{2}\right|$. *We say that the Paillier PKE scheme* Pai *is secure, if for all probabilistic polynomial time (PPT) adversary* $\mathcal{B}$ *the advantage* $\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathsf{Pai},\mathcal{B}}(\kappa)$ *is a negligible function in* κ.

Two-party Secure Function Evaluation. We briefly review the formal notions regarding (circuit based) secure function evaluation (SFE) which is used by the YJ protocol. Given a public function $\hat{F}$, a classical SFE scheme allows two parties to run a protocol which results in party 1 learning only the outcome of $\hat{F}(x_1||x_2)$, while party 2 learning nothing, where x_1 and x_2 are the private inputs of party 1 and party 2 respectively. We refer the reader to [2] for more details on the security notions and concrete example of SFE.

We let $\hat{f}$ denote a circuit for a certain function $\hat{F}$ with input size $n \in \mathbb{N}$ (that may be accessed as $\hat{f}.n$). And let $\mathsf{ev}(\hat{f}, x)$ be a canonical circuit evaluation function which takes as inputs $\hat{f}$ and a string x, and computes the output of the function $\hat{F}(x)$. Here we define a function $\Phi(\hat{f})$ to describe what we allow to be revealed regarding $\hat{f}$. With respect to a garbling scheme, Φ may reveal a circuit's size, topology, identity, or many others. More concrete side information functions can be found in [1,2].

In a two-party protocol, we suppose that party i ($i \in [2]$) has a private string x_i with length n_i, and party 2 has a circuit $\hat{f}$ where $n = n_1 + n_2$. We describe a two-party protocol (for executing a SFE scheme) via a pair of PPT algorithms $\Sigma = (\Sigma_1, \Sigma_2)$. Party $i \in \{1,2\}$ will run Σ_i on its current state and the incoming message from its intended partner, to generate an outgoing message and a local output. The initial state of Σ_i includes the security parameter κ, a fresh random coin $\gamma_i \overset{\$}{\leftarrow} \mathcal{R}_i$ (chosen from a random space $\mathcal{R}_i$) and the (private) function input I_i of party i. The random coins γ_1 and γ_2 might be omitted (in the following descriptions) for simplicity, i.e., they are implicitly generated and used. In order to represent the protocol execution, we define a PPT algorithm View^i_Σ which takes as input security parameter 1^κ, and inputs (I_1, I_2) for the two parties respectively, and returns an execution view vw_i and output out_i of party i in a protocol instance. Nevertheless, we may denote an execution between two parties as $\mathsf{SF}.\Sigma(I_1, I_2)$ at a high-level view.

Then a SFE scheme is a tuple $\mathsf{SF} = (\Sigma, \mathsf{ev})$ where Σ is a two-party protocol with input (I_1, I_2) as above and ev is a circuit evaluation function. The correctness requirement states that, for all $\hat{f}$ and all $x \in \{0,1\}^{\hat{f}.n}$, we have $\Pr[\mathsf{out}_1 = \mathsf{ev}(\hat{f}, x)] = 1$, where $x = x_1||x_2$, $x_1 \in I_1$ and $(x_2, \hat{f}) \in I_2$. We here review the privacy of SFE in the honest-but-curious setting.

Definition 2. *For a SFE scheme* $\mathsf{SF} = (\Sigma, \mathsf{ev})$*, a simulator* $\mathcal{S}$ *and an adversary* $\mathcal{E}$*, the security experiment relative to* Φ *is defined as follows:*

```
$\mathsf{EXP}^{\mathsf{pri.sim},\mathcal{S}}_{\mathsf{SF},\mathcal{E},\Phi}(\kappa, i)$ :
    $b \xleftarrow{\$} \{0,1\}$;
    $b' \leftarrow \mathcal{E}^{\mathsf{Excute}_{\mathsf{SF}}(b,i,\cdot,\cdot)}(\kappa, i)$;
    $if\ b = b'\ return\ 1$,
    $otherwise\ return\ 0.$

$\mathsf{Excute}_{\mathsf{SF}}(b, i, x_i, \hat{f})$ :
    $if\ x_i \not\subseteq \{0,1\}^{\hat{f}.n_i}\ return \perp$;
    $x_{3-i} \xleftarrow{\$} \{0,1\}^{\hat{f}.n_{3-i}},\ I_1 := x_1,\quad I_2 := (x_2, \hat{f})$;
    $if\ b = 1\ \ return\ \mathsf{View}^i_{\Sigma}(1^\kappa, I_1, I_2)$;
    $if\ i = 1\ \ return\ \mathcal{S}(1^\kappa, \mathsf{ev}(\hat{f}, x_1||x_2), \Phi(\hat{f}))$;
    $if\ i = 2,\ \ return\ \mathcal{S}(1^\kappa, \hat{f}, |x_1|)$;
```

We define the advantage of $\mathcal{E}$*, which is allowed only a single* $\mathsf{Excute}_{\mathsf{SF}}$ *query, in the above experiment as:* $\mathsf{Adv}^{\mathsf{pri.sim},\mathcal{S}}_{\mathsf{SF},\mathcal{E},\Phi}(\kappa, i) := \left|\Pr[\mathsf{EXP}^{\mathsf{pri.sim},\mathcal{S}}_{\mathsf{SF},\mathcal{E},\Phi}(\kappa, i) = 1] - \frac{1}{2}\right|$*. We say that* SF *is secure relative to* Φ*, if for each* $i \in \{1,2\}$ *and all PPT adversaries* $\mathcal{E}$*, the advantage* $\mathsf{Adv}^{\mathsf{pri.sim},\mathcal{S}}_{\mathsf{SF},\mathcal{E},\Phi}(\kappa, i)$ *is a negligible function in* κ*.*

3 A New Security Model for Privacy Preserving Indoor Location Schemes

In this section, we define a new unilateral-malicious security model for privacy preserving indoor location (PPIL) protocols which are based on Wifi fingerprints. The privacy for client and server is formulated respectively following the well-known game-based modeling approach [3,10].

Simulation Preliminary. We first describe the general simulation environment which will be exploited in the following security notions (in particular for security experiment). There are two types of entities considered: client C and server S. The server S is supposed to provide the indoor location service (ILS) of a building according to a client's request. The building area (which is covered by the location service) is assumed to be delicately divided into M reference locations $\mathsf{LT} = \{i, (x_i, y_i, z_i)\}_{i=1}^{M}$, e.g. the black dot in Fig. 1, where (x_i, y_i) denotes the horizontal coordinates and z_i denotes the vertical coordinate (e.g., the position of a floor). One could consider the unit of each coordinate is meter (m) for instance. Moreover, the building is deployed with N Wifi access points (AP) to provide network service, where each i-th ($i \in [N]$) access point may have a unique identity AP_i. Let $\mathsf{APT} = \{AP_j\}_{j=1}^{N}$ be list storing all identities of Wifi access points. In particular, each location has a so-called Wifi fingerprint which comprises of Received Signal Strength (RSS) values of certain Wifi AP, where each RSS value is from a range $\mathcal{R}_v = [v_{min}, v_{max}]$ and (v_{min}, v_{max}) are minimum and maximum values respectively. Consequently, the server is assumed to hold a pre-measured Wifi fingerprint database D which consists of a set of tuples

$< i, V_i = \{v_{i,j}\}_{j=1}^{N} >_{i=1}^{M}$ (See also in [22, Table 1]) , where i is an index of a reference location $L_i \in \mathsf{LT}$, each $v_{i,j}$ denotes the RSS value obtained at L_i from AP_j. Furthermore, we let Dist be a distance function which takes as input two locations L_i and L_j (with their corresponding coordinates (x_i, y_i, z_i) and (x_j, y_j, z_j)) and outputs the distance between them. One could consider Euclidean distance, i.e. Eq. 1, as a concrete example of Dist.

When C wants to know its location, it first measures the RSS values from all APs to get a real-time Wifi fingerprint $F = \{f_j\}_{j=1}^{N}$. Then it may 'privately' submit F to S as a location query, and calculate its location L from S's response. We refer the reader to [23, Sect. 2.1] for more details on the principle of Wifi fingerprint localization. Meanwhile, the private information of the client mainly includes its secret key sk, location query F and the corresponding location L. The secret of the server is the database D.

In order to emulate the behaviors of a set of entities (including λ clients and 1 server), we may realize a collection of oracles $\{\pi_\tau^s, \pi_{\lambda+1}^t : \tau \in [\lambda], s \in [d], t \in [\lambda \times d]\}$ for $(\lambda, d) \in \mathbb{N}$. Each oracle π_τ^s behaves as the s-th protocol instance (session) performed by the party τ for calculating one location. The special party $\lambda + 1$ is assumed to be server. Each party may have a pair of pubic/private key (pk_τ, sk_τ) for $\tau \in [\lambda + 1]$, where pk_τ can be accessed by all oracles. Moreover, each oracle π_τ^s for $\tau \in [\lambda]$ is supposed to keep the following internal state variables: (i) $ds_\tau^s \in \{\mathsf{accept}, \mathsf{reject}\}$ – final decision of a session; (ii) F_τ^s – fingerprint $F_\tau^s = \{v_j'\}_{j=1}^{N}$ for a location query; (iii) ins_τ^s – index selection set (INS) specifying the location indexes (in LT) which are close to the current location related to F_τ^s; (iv) er_τ^s – ephemeral randomness used to run the protocol instance; (v) T_τ^s – transcript recording all sent and received protocol messages; (vi) $L_\tau^s = (x_\tau^s, y_\tau^s, z_\tau^s)$ – location of party τ calculated in the s-th session. We assume the variable L_τ^s will be assigned if and only if $ds_\tau^s = \mathsf{accept}$ (meaning that a protocol instance is correctly executed in a session). The server's oracles only have ds_τ^s and T_τ^s.

In order to simulate a Wifi fingerprint used by a location query, we define a function $\mathsf{FPTSim}(i)$ which on input a reference location index i generates a Wifi fingerprint $F_i = \{f_j\}_{j=1}^{N}$ with the following steps: (i) $f_j \overset{\$}{\leftarrow} [v_{i,j} - \Delta, v_{i,j} + \Delta]$ where Δ is a pre-defined positive integer, where $v_{i,j} \in \mathsf{D}$; (ii) If $f_j \leq v_{min}$ or $v_{i,j} = v_{min}$ then $f_j := v_{min}$; (iii) Else if $f_j \geq v_{max}$ then $f_j := v_{max}$.

Adversarial Model. Here we define the power of an active adversaries. The active adversaries $\mathcal{A}$ in our model are considered as a probabilistic polynomial time (PPT) algorithms, which may interact with another PPT algorithm called simulator $\mathcal{C}$ via the following queries:

- **InitCorruptO**$(\tau, s, \tilde{F})$: The variables ds_τ^s, T_τ^s and L_τ^s (if any) of the client's oracle π_τ^s are initiated with an empty string $\emptyset$. This query initializes $\mathsf{ins}_\tau^s := [M]$. If $\tilde{F} \neq \emptyset$ and $\tau \neq \lambda + 1$, this query sets $F_\tau^s := \tilde{F}$. Each oracle can be initialized by this query only once.
- **InitHonestO**(τ, s, i, rds): This query first initializes ds_τ^s, er_τ^s, T_τ^s and L_τ^s with empty string $\emptyset$. Let $\widetilde{\mathsf{ins}} \subseteq [M]$ be a set of location indexes such that

$\forall j \in \widetilde{\mathsf{ins}}$ the distance between $L_i = (x_i, y_i, z_i)$ and $L_j = (x_j, y_j, z_j)$ is smaller than rds, i.e., $\mathsf{Dist}(L_i, L_j) \le rds$. Note that $\widetilde{\mathsf{ins}}$ may cover indexes within a ball centered at i with radius rds. If $\tau \neq \lambda + 1$ and $\#\widetilde{\mathsf{ins}} \ge \lceil \chi \cdot M \rceil$ (for a threshold say $0.1 \le \chi \le 1$)[1], this query initializes F_τ^s as follows: (i) $j \xleftarrow{\$} \widetilde{\mathsf{ins}}$; (ii) $F_\tau^s := \mathsf{FPTSim}(j)$; (iii) $\mathsf{ins}_\tau^s := \widetilde{\mathsf{ins}}$. Again each client's oracle can be initialized by this query only once.

- **Execute**$_{\mathsf{PPIL}}(\tau, s, t)$: This query executes the protocol instance between an unused and initialized client's oracle π_τ^s and a server's unused oracle $\pi_{\lambda+1}^t$, and returns the protocol transcript T_τ^s. We call π_τ^s and $\pi_{\lambda+1}^t$ proceeded in this query as *partner oracles*. The oracles run by this query are called *used*. All server's oracles here are assumed to be default initialized (without specific initiation query).
- **CorruptC**(τ): This query responds with the τ-th client's secret key sk_τ.
- **CorruptS**: This query responds with the server's database D and secret key $sk_{\lambda+1}$ (if any).
- **RandReveal**(τ, s): Oracle π_τ^s responds with the ephemeral secret key er_τ^s.
- **LocReveal**(τ, s): Oracle π_τ^s responds with the location L_τ^s.
- **LocTest**(τ, s): If the oracle has state $ds_\tau^s \neq \mathsf{accept}$ or $\tau = \lambda+1$, then this query returns a failure symbol $\perp$. Otherwise, it does the following steps: (i) flip a fair coin $b \xleftarrow{\$} \{0, 1\}$; (ii) choose a random index $j \in \mathsf{ins}_\tau^s$, obtain $F_j := \mathsf{FPTSim}(j)$, and calculate L_0 based on F_j and D (following the protocol specification) such that $L_0 \neq L_\tau^s$; (iii) set $L_1 := L_\tau^s$ (which is the real location). Eventually, the location L_b is returned. This query is allowed to be asked at most once during the following corresponding security experiment. We call the oracle π_τ^s selected in this query as *test oracle*.
- **DBLeak**(i): If the index i has been queried via this query, then it returns a failure symbol $\perp$. Otherwise, this query responses with a similar Wifi fingerprint $F_i' \leftarrow \mathsf{FPTSim}(i)$ according to the i-th row of database D.

InitCorruptO query is used to model the chosen fingerprint attacks against server's privacy (in the unilateral-malicious setting), i.e., the malicious client may choose special fingerprints (e.g. all zeros) to compromise the server's database. For example, the attack introduced in [22] is a kind of chosen fingerprint attack. An oracle initialized by this query is known as location exposed oracle.

InitHonestO query is used to initialize the honest (unexposed) oracle based on an area which is specified by an adversary in term of the reference location index i and a radius rds. We categorize the attacks modeled by this query as *known sub-area attacks*. Consider the attack scenario that an adversary loses his tracking target at a street corner (determined by i) and he wants to know the target's 'whereabouts' (within a range rds). In this case, the attacker may know an approximate area of the client within a range. Moreover, if rds is large enough then it may cover all location indexes in LT.

Execute$_{\mathsf{PPIL}}$ query formulates the passive adversaries which only observe the communication between the client and server.

[1] If χ is too small, then there is no privacy at all.

CorruptC and **CorruptS** queries formulate the corruption of an honest principal's long-term credentials respectively. The corrupted party is known as dishonest or malicious one.

RandReveal query models the randomness exposure attacks which may be caused by malware or careless disposal.

DBLeak query 'approximately' formulates the attack that $\mathcal{A}$ measures and records the Wifi fingerprints $\mathsf{V}_i{}'$ (which is similar to V_i of D) for certain location index i, say based on limited Wifi fingerprint samples.

LocReveal query models the known location attacks (ULA). The resilience of ULA requires that the exposed locations will not affect the others. For example, the PPIL scheme proposed in [12] is subject to known location attack. To get a location, the client in [12] would issue a set of camouflaged localization requests that follow a similar natural movement pattern. However, if one of the client's locations is exposed, e.g., by posting a picture, then the server can simply identify which location request is the real one.

LocTest query will be exploited to formulate the capability of an adversary on breaking the client's privacy. The job of the adversary is to distinguish the bit chosen by the **LocTest** query.

Note that we are the first one to generalize the practical attacks against PPIL schemes via the above generic queries which have not been formalized in previous work [13,22,24].

Client Privacy. We first define a security experiment as follows.

SECURITY EXPERIMENT $\mathsf{EXP}^{\mathsf{CP}}_{\Pi,\mathcal{A}}(\kappa, \mathsf{D})$: On input security parameter κ and a server's database D, the security experiment is carried out as a game between a simulator $\mathcal{C}$ and an adversary $\mathcal{A}$ based on a PPIL scheme Π, where the following steps are performed:

1. The simulator $\mathcal{C}$ first initiates the game by realizing a collection of oracles and generating all public/private key pairs for all $\lambda + 1$ honest parties and all other public information. $\mathcal{C}$ gives $\mathcal{A}$ all public information $\{pk_\tau\}_{\tau=1}^{\lambda+1}$, LT, APT and $\mathcal{PD}$.
2. $\mathcal{A}$ may adaptively issue a polynomial number of **InitCorruptO**, **InitHonestO**, **Execute**$_{\mathsf{PPIL}}$, **CorruptC**, **CorruptS**, **LocReveal**, and **RandReveal** queries. At some point, $\mathcal{A}$ may issue a single **LocTest**(τ^*, s^*) query.
3. At the end of the game, $\mathcal{A}$ may terminate and output a bit b' as its guess for b of **LocTest**(τ^*, s^*) query.
4. Meanwhile, the experiment would return a failure symbol $\perp$ if one of the following conditions is satisfied: (a) $\mathcal{A}$ has not issued a **LocTest**(τ^*, s^*) query; (b) The **LocTest**(τ^*, s) query returns a failure symbol $\perp$; (c) $\mathcal{A}$ asked an **InitCorruptO**(τ^*, s^*, F^*) query to the test oracle; (d) $\mathcal{A}$ asked a **CorruptC**(τ^*) query; (e) $\mathcal{A}$ asked either a **RandReveal**(τ^*, s^*) query or a **RandReveal**$(\lambda + 1, t^*)$ query, where $\pi^{t^*}_{\lambda+1}$ is the partner oracle of the test oracle; (f) $\mathcal{A}$ asked a **LocReveal**(τ^*, s^*) query to the test oracle $\pi^{s^*}_{\tau^*}$.
5. The experiment finally returns 1 if $b = b'$, and 0 otherwise.

We call an adversary as a 'legal' one if it runs an experiment without failure. A legal adversary should not violate the rules defined in the above step 4). Note that violating one of the rules (c) to (f) would 'trivially' break the client-privacy, i.e., asking the corresponding queries (specified in the rules) would enable the adversary to easily distinguish the bit b chosen in the **LocTest**(τ^*, s) query without breaking the underlying protocol. These situations should be therefore forbidden in the experiment. Otherwise, it would always return 1.

Definition 3 (Client-privacy). *The advantage of legal adversaries $\mathcal{A}$ in the above experiment is* $\mathsf{Adv}^{\mathsf{CP}}_{\Pi,\mathcal{A}}(\kappa, \mathsf{D}) := \left|\Pr[\mathsf{EXP}^{\mathsf{CP}}_{\Pi,\mathcal{A}}(\kappa, \mathsf{D}) = 1] - \frac{1}{2}\right|$. *We say that a PPIL scheme* Π *is client-secure, if for all PPT legal adversaries* $\mathcal{A}$*, the advantage* $\mathsf{Adv}^{\mathsf{CP}}_{\Pi,\mathcal{A}}(\kappa, \mathsf{D})$ *is a negligible function in* κ*.*

Server Privacy. Informally speaking, the server's privacy is achieved if all polynomial time adversaries are unable to generate a database D' which can provide a similar function as the server's real database D. We may call a location calculated based on D' as a *fabricated location*, and a location calculated based on D as *real location*. Given two databases D and D', we have the following similar event (as exemplified in Fig. 1):

– *Similar Event (SE)*: For a client's location query regarding Wifi fingerprint $F_i = \{f_j\}_{j=1}^{N}$, the corresponding location L_i and the fabricated location L_i' have distance at most ρ, i.e., $\mathsf{Dist}(L_i, L_i') \leq \rho$, where ρ is a pre-defined difference threshold (in meter).

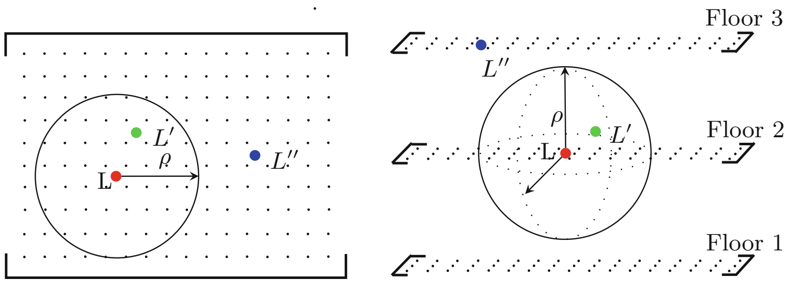


Fig. 1. Similar event occurrence examples in horizontal (left) and vertical planes (right). The small black dots represent the reference locations in LT. The red dot represents the real location L. The green dot represents the fabricated location L' in which the similar event occurs. The blue dot represents the fabricated location L'' in which the similar event does not occur. (Color figure online)

The term on 'similar function' of two databases can be roughly illustrated as follows. Given a number of distinct client's location queries, the occurrence rate of SE is larger than a pre-defined success threshold α (e.g. $\alpha = 0.7$). Let TF be a test set that consists of $|TF| > M$ distinct fingerprints of random locations.

Table 1. Parameters of server-privacy.

Params	Description
D	Real database of server
ϕ	A security parameter specifying the number of **DBLeak** queries
ρ	Distance threshold between the real location and the fabricated location
α	Probability threshold of SE
TF	Test set of random fingerprints

For example, one could generate a fingerprint $F \in TF$ by randomly choosing an index $i \overset{\$}{\leftarrow} [M]$ and running $F := \mathsf{FPTSim}(i)$. Let $\mathsf{SimilarTest}$ be a function that is used to test the functional similarity between two databases. $\mathsf{SimilarTest}$ takes as input two databases D and D' with their related reference location lists LT and LT' (respectively), and a test set TF, and outputs the test result in $\{0, 1\}$. The execution steps of $\mathsf{SimilarTest}$ comprises of the following:

- Initiate a SE count variable $cnt := 0$. Suppose that for a fingerprint $F_i \in TF$ the real location which is calculated based on F_i, D and LT is $L_i = (x_i, y_i, z_i)$, and the fabricated location which is calculated based on F_i, D' and LT' is $L'_i = (x'_i, y'_i, z'_i)$. For $i \in [|TF|]$, if $\mathsf{Dist}(L_i, L'_i) \leq \rho$ then $cnt := cnt + 1$.
- Finally, it returns 1 if $\frac{cnt}{|TF|} > \alpha$; otherwise, 0 is returned.

The parameters, which are relevant to the formulation of the server-privacy, are summarized in Table 1.

Security Experiment $\mathsf{EXP}^{\mathsf{SP}}_{\Pi,\mathcal{A}}(\kappa, \mathsf{D}, \mathsf{LT}, \rho, \alpha, \phi)$: On input security parameter κ, a server's database D, and a distance accuracy threshold ρ, the security experiment is carried out as a game between a simulator $\mathcal{C}$ and an adversary $\mathcal{A}$ based on a PPIL scheme Π, where the following steps are performed:

1. The simulator $\mathcal{C}$ first implements a collection of oracles and generates all public/private key pairs for all $\lambda + 1$ honest parties and all other public information. All public information are given to $\mathcal{A}$.
2. $\mathcal{A}$ may issue a polynomial number of queries to **InitCorruptO**, **CorruptC**, **Execute**$_{\mathsf{PPIL}}$, **RandReveal**, and **LocReveal** respectively, and at most ϕ **DBLeak** queries.
3. Eventually, $\mathcal{A}$ may return a database D' and a relevant reference location list LT' that has M' reference location. Meanwhile, the experiment would return a failure symbol $\perp$ if $\mathcal{A}$ asked either a **RandReveal**$(\lambda + 1, \cdot)$ query or more than ϕ queries to **DBLeak**.
4. Finally, the experiment returns $\mathsf{SimilarTest}(\mathsf{D}, \mathsf{D}', \mathsf{LT}, \mathsf{LT}', TF)$.

Definition 4 (Server-privacy). *The advantage of a legal adversary $\mathcal{A}$ in the above experiment is* $\mathsf{Adv}^{\mathsf{SP}}_{\Pi,\mathcal{A}}(\kappa, \mathsf{D}, \mathsf{LT}, \rho, \alpha, \phi) := \Pr[\mathsf{EXP}^{\mathsf{SP}}_{\Pi,\mathcal{A}}(\kappa, \mathsf{D}, \mathsf{LT}, \rho, \alpha, \phi) = 1]$. *We say that a PPIL scheme Π is server-secure, if for all PPT legal adversaries $\mathcal{A}$, the advantage* $\mathsf{Adv}^{\mathsf{SP}}_{\Pi,\mathcal{A}}(\kappa, \mathsf{D}, \mathsf{LT}, \rho, \alpha, \phi)$ *is a negligible function in κ.*

We define the above model based on Wifi fingerprint database as an example. Of course, one could simply modify our model for other types of PPIL schemes since each query aforementioned represents a generic class of attacks against PPIL schemes. One may only need to customize the simulation environment and slightly modify the queries if necessary.

Database Hardcore. The volume of a database D is determined by the number M of reference locations (that is related to the area of a building), the number N of APs, and bit size of each RSS value $|\mathcal{R}_v|$. However, there is a general problem on how hard it is for adversaries to generate a valid fabricated database D' without any useful information from a PPIL scheme using D. I.e. is the D' itself hard to build? This question is independent of any concrete PPIL schemes. If D' is easy to generate without breaking the PPIL scheme, then we do not need a PPIL scheme at all. Since the server could just publish its database for all clients. Intuitively, the adversary should be very hard to generate a valid fabricated D' that has a similar function as D since D' also has a large number of bits to predict. In the following, we are going to give a formal definition regarding the security assumption of a database (that is non-relevant to PPIL schemes).

Definition 5. *The security experiment for testing the hardness of forging a similar database for a target database* D *is defined in the following:*

$$\begin{aligned}&\mathsf{EXP}^{\mathsf{DBH}}_{\mathcal{D}}(\kappa, \mathsf{D}, \mathsf{LT}, \rho, \alpha, \phi):\\ &\quad (D', \mathsf{LT}') \leftarrow \mathcal{D}^{\mathsf{DBLeak}(\cdot)}(\mathsf{LT}, \rho, \alpha, \phi),\ \textit{Return}\ \mathsf{SimilarTest}(\mathsf{D}, \mathsf{D}', \mathsf{LT}, \mathsf{LT}', \rho, \alpha, \phi).\end{aligned}$$

The advantage of $\mathcal{D}$ *which can ask at most* ϕ ***DBLeak*** *queries in the above experiment is* $\mathsf{Adv}^{\mathsf{DBH}}_{\mathcal{D}}(\kappa, \mathsf{D}, \mathsf{LT}, \rho, \alpha, \phi) := \Pr[\mathsf{EXP}^{\mathsf{DBH}}_{\mathcal{D}}(\kappa, \mathsf{D}, \mathsf{LT}, \rho, \alpha, \phi) = 1]$. *We say that a database* D *is hard to forge, if for all PPT adversaries* $\mathcal{D}$ *the advantage* $\mathsf{Adv}^{\mathsf{DBH}}_{\mathcal{D}}(\kappa, \mathsf{D}, \mathsf{LT}, \rho, \alpha, \phi)$ *is a negligible function in* κ.

It is straightforward to see that D is hard to forge if only a small portion of D is leaked via **DBLeak** to the adversary and D has large M, N, and $|\mathcal{R}_v|$, e.g., $M = 505$, $N = 241$ and $|\mathcal{R}_v| = 8$ in the real database [16, BUILDING1_NEW] which has $M \times N \times |\mathcal{R}_v| = 973640$ bits at all. However, an open question is how hard it is to create a valid fabricated database. Such hardness might be closely related to the structure of specific building and database generation algorithm. In the future work, one is encouraged to formally analyze the database hardcore assumption in the setting with the leakage of side-channel information, such as adversaries' own RSS measurements modeled by **DBLeak** query. In this paper, we just focus on the formalism of server-privacy for PPIL schemes.

4 On the Security of the YJ Scheme

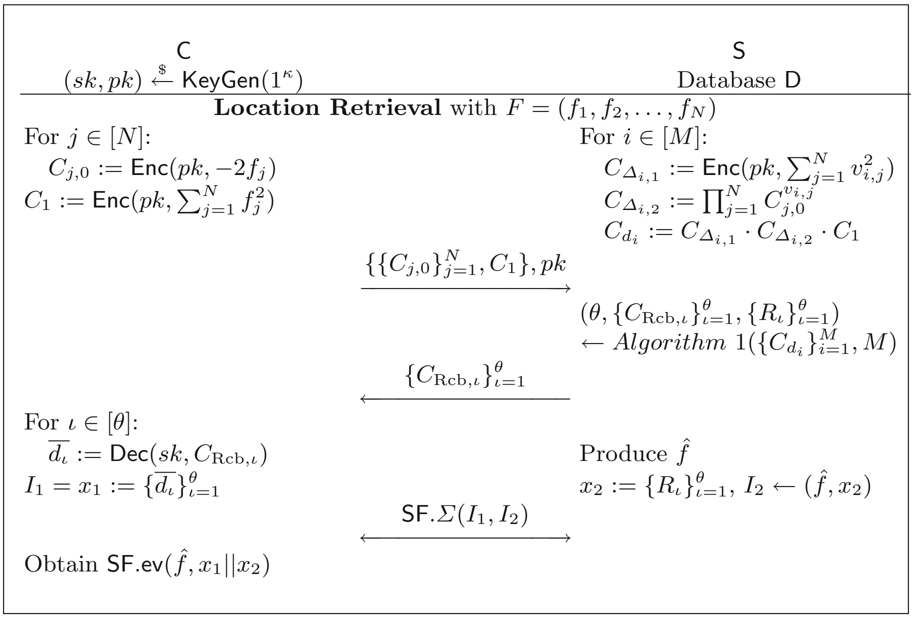
The YJ Scheme. We first review the PPIL scheme [22] recently proposed by Yang and Järvinen. The YJ scheme is built from Paillier PKE $\mathsf{Pai} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ and two-party SFE protocol $\mathsf{SF} = (\Sigma, \mathsf{ev})$. Paillier PKE scheme is used to protect a client C's fingerprint $F = (f_1, f_2, \ldots, f_N)$. In the YJ scheme, the server S should compute the distances between F and V_i (of its database D), where each distance d_i is assumed to be the following Euclidean distance:

$$d_i = ||V_i - F||^2 = \sum_{j=1}^{N}(v_{i,j} - f_j)^2 = \sum_{j=1}^{N} v_{i,j}^2 + \sum_{j=1}^{N}(-2v_{i,j}f_j) + \sum_{j=1}^{N} f_j^2. \quad (1)$$

SFE protocol is used to privately compute the location $L_{\mathsf{C}} = (x, y, z)$ of C as the centroid of the k nearest reference locations indexed by $i_1, i_2, \ldots, i_k$, where $i_1, i_2, \ldots, i_k$ indicate distances such that $d_{i_1} \leq d_{i_2} \leq \ldots \leq d_{i_k} \leq d_j$ for all $j \neq i_1, i_2, \ldots, i_k$.

Protocol Description. When C subscribes to the location service, it runs $(sk, pk) \overset{\$}{\leftarrow} \mathsf{KeyGen}(\kappa)$ to generate a key pair (sk, pk) for Paillier PKE scheme with a sufficiently large κ (e.g. $\kappa = 2048$) and sends $pk = (n, g)$ to S. The protocol execution is shown in Fig. 2.

Note that the randomness space $\mathcal{R}_R = \mathbb{Z}_n$ may result in the blinded distance being wraparound over $\mathbb{Z}_n$, i.e. a modular n operation is involved in the generation of the blinded distance.



C
$(sk, pk) \overset{\$}{\leftarrow} \mathsf{KeyGen}(1^\kappa)$
S
Database D
Location Retrieval with $F = (f_1, f_2, \ldots, f_N)$
For $j \in [N]$:
$C_{j,0} := \mathsf{Enc}(pk, -2f_j)$
$C_1 := \mathsf{Enc}(pk, \sum_{j=1}^{N} f_j^2)$
For $i \in [M]$:
$C_{\Delta_{i,1}} := \mathsf{Enc}(pk, \sum_{j=1}^{N} v_{i,j}^2)$
$C_{\Delta_{i,2}} := \prod_{j=1}^{N} C_{j,0}^{v_{i,j}}$
$C_{d_i} := C_{\Delta_{i,1}} \cdot C_{\Delta_{i,2}} \cdot C_1$
$\xrightarrow{\{\{C_{j,0}\}_{j=1}^{N}, C_1\}, pk}$
$(\theta, \{C_{\mathrm{Rcb},\iota}\}_{\iota=1}^{\theta}, \{R_\iota\}_{\iota=1}^{\theta})$
$\leftarrow$ *Algorithm* $1(\{C_{d_i}\}_{i=1}^{M}, M)$
$\xleftarrow{\{C_{\mathrm{Rcb},\iota}\}_{\iota=1}^{\theta}}$
For $\iota \in [\theta]$:
$\overline{d_\iota} := \mathsf{Dec}(sk, C_{\mathrm{Rcb},\iota})$
$I_1 = x_1 := \{\overline{d_\iota}\}_{\iota=1}^{\theta}$
Produce $\hat{f}$
$x_2 := \{R_\iota\}_{\iota=1}^{\theta}$, $I_2 \leftarrow (\hat{f}, x_2)$
$\xleftrightarrow{\mathsf{SF}.\Sigma(I_1, I_2)}$
Obtain $\mathsf{SF.ev}(\hat{f}, x_1 || x_2)$

Fig. 2. The YJ Scheme

Security Analysis. The security results of our scheme are shown by the following theorems. Here we briefly analyze the theorems. The full proofs of them will be presented in the full version of this paper.

Algorithm 1. Pack Encrypted Distance Set

```
   Input: ${C_{d_i}}_{i=1}^{M}$ and $M$
   Output: $\theta$, ${C_{\mathrm{Rcb},\iota}}_{\iota=1}^{\theta}$, and ${R_\iota}_{\iota=1}^{\theta}$
1  $\theta := 1$; $\mu := M$; $\mathcal{R}_R = \mathbb{Z}_n$
2  while $\mu > 0$ do
3      $t := \frac{\kappa-1}{m}$
4      if $t > \mu$ then
5          $t := \mu$
6      $C_{\mathrm{cb},\theta} := \prod_{i=1}^{t} C_{d_{\mu-i}}^{2^{(i-1)m}}$; $R_\theta \xleftarrow{\$} \mathcal{R}_R$; $C_{\mathrm{Rcb},\theta} := C_{\mathrm{cb},\theta} \cdot \mathsf{Enc}(pk, R_\theta)$
7      $\mu := \mu - t$
8      if $\mu \neq 0$ then
9          $\theta := \theta + 1$
10 return $(\theta, \{C_{Rcb,\iota}\}_{\iota=1}^{\theta}, \{R_\iota\}_{\iota=1}^{\theta})$
```

Theorem 1. *Suppose that the Paillier PKE scheme* Pai *is secure and the SFE scheme* SF *is secure, then the YJ scheme with a database* D *is client-secure with* $\mathsf{Adv}_{\mathsf{YJ},\mathcal{A}}^{\mathsf{CP}}(\kappa, \mathsf{D}) \leq (d\lambda) \cdot ((N+1) \cdot \mathsf{Adv}_{\mathsf{Pai},\mathcal{B}}^{\mathsf{ind\text{-}cpa}}(\kappa) + \frac{M}{2} \cdot \mathsf{Adv}_{\mathsf{SF},\mathcal{E},\Phi}^{\mathsf{pri.ind}}(\kappa, 1))$.

We summarize the games of the proof in Table 2. We use a superscript '$*$' to denote an element of the test oracle.

Table 2. Sequence of games for client-privacy.

Game	Description and Modification
0	Real experiment. $\{\{C_{j,0}^*\}_{j=1}^N, C_1^*\}$ and $\{C_{\mathrm{Rcb},\iota}^*\}_{\iota=1}^{\theta}$ of the test oracle are computed with $F^* = \{f_\iota^*\}_{\iota=1}^N \xleftarrow{\$} \mathsf{FPTSim}(i^*)$
1	Abort if the challenger fails to guess the test oracle
2	$\{C_{\iota,0}^*\}_{\iota=1}^N$ are computed with $F^{*\prime} = \{f_\iota^{*\prime}\}_{\iota=1}^N$, but $\{C_{\mathrm{Rcb},\iota}^*, C_1^*\}_{\iota=1}^{\theta}$ are computed with $F^* = \{f_\iota^*\}_{\iota=1}^N$, where $f_1^{*\prime} \neq f_1^*$ but $\{f_\iota^{*\prime}\}_{\iota=2}^N = \{f_\iota^*\}_{\iota=2}^N$
3.j $j \in [N]$	Game 2 = Game 3.1 In Game 3.j: $f_\iota^{*\prime} \neq f_\iota^*$ for $1 \leq \iota \leq j$, but $\{f_\iota^{*\prime}\}_{\iota=j+1}^N = \{f_\iota^*\}_{\iota=j+1}^N$
4	Generating C_1^* using a random squared RSS values. $\forall\{\{C_{j,0}^*\}_{j=1}^N, C_1^*\}$ and $\{C_{\mathrm{Rcb},\iota}^*\}_{\iota=1}^{\theta}$ are independent now
5	A random location is chosen to answer the **LocTest** query

Theorem 2. *Suppose that the SFE scheme* SF *is secure, the database* D *is hard to forge, then the YJ scheme is server secure with* $\mathsf{Adv}_{\mathsf{YJ},\mathcal{A}}^{\mathsf{SP}}(\kappa, \mathsf{D}, \mathsf{LT}, \rho, \alpha, \phi) \leq d \cdot \ell \cdot \mathsf{Adv}_{\mathsf{SF},\mathcal{E},\Phi}^{\mathsf{pri.ind}}(\kappa, 2) + \frac{\theta \cdot d \cdot \ell}{2^\kappa} + \mathsf{Adv}_{\mathcal{D}}^{\mathsf{DBH}}(\kappa, \mathsf{D}, \mathsf{LT}, \rho, \alpha, \phi)$.

We summarize the proof of this theorem in Table 3.

Table 3. Sequence of games for server-privacy

Game	Description and modification
0	Real experiment
1	Abort if two random values are equal
2	The random values used to generate the ciphertexts $\{C^*_{\mathrm{Rcb},\iota}\}^{\theta}_{\iota=1}$ and corresponding SFE protocol instance are different
3	Apply database entropy assumption as Definition 5

5 Conclusion

We presented the first formal privacy model for Wifi fingerprint based PPIL schemes, where both client- and server- privacy are formulated in a unilateral-malicious setting to cover state-of-the-art active attacks. The client-privacy is defined based on the classic notion of indistinguishability, and the server privacy is defined in a computational manner. The proposed model is verified by applying it for proving a recent PPIL protocol. An interesting open question here is whether or nor our security analysis approach can be applied to prove other kinds of privacy-preserving schemes which have a similar construction (i.e., using Paillier PKE and SFE) to the YJ scheme, e.g., the protocols for face recognition [4,19]. For theoretical interesting, the reader is encouraged to define a stronger security model in the full malicious setting based on our model, and to proposed PPIL protocols which can be proven secure under such model. For example, one could allow the active adversaries to send her own messages to oracles (masquerading as either client or server). In the future work, it is also required to formally study the complexity of Definition 5. Nevertheless, it might be also interesting to consider whether or not it is possible to model the server-privacy based on indistinguishability.

Acknowledgments. This work was funded by the INSURE project (303578) of Academy of Finland, and the research project of the Humanities and Social Sciences of the Ministry of Education of China (Grant No. 16YJC870018), and supported by the project “Research on Cryptographic Techniques for Privacy Preserving Location Schemes” funded National Natural Science Foundation of China (Grant No. 61872051).

References

1. Bellare, M., Hoang, V.T., Keelveedhi, S.: Efficient garbling from a fixed-key block-cipher. In: IEEE Security and Privacy 2013, pp. 478–492 (May 2013)
2. Bellare, M., Hoang, V.T., Rogaway, P.: Foundations of garbled circuits. In: ACM CCS 2012, pp. 784–796. ACM, October 2012
3. Bellare, M., Rogaway, P.: Entity authentication and key distribution. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 232–249. Springer, Heidelberg (1994). https://doi.org/10.1007/3-540-48329-2_21

4. Blanton, M., Gasti, P.: Secure and efficient protocols for iris and fingerprint identification. In: Atluri, V., Diaz, C. (eds.) ESORICS 2011. LNCS, vol. 6879, pp. 190–209. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-23822-2_11
5. Elnahrawy, E., Li, X., Martin, R.P.: The limits of localization using signal strength: a comparative study. In: SECON 2004, pp. 406–414. IEEE (2004)
6. Ferreira, A., Fernandes, D., Catarino, A., Monteiro, J.: Localization and positioning systems for emergency responders: a survey. IEEE Commun. Surv. Tutor. **19**(4), 2836–2870 (2017)
7. He, S., Chan, S.H.G.: WI-FI fingerprint-based indoor positioning: recent advances and comparisons. IEEE Commun. Surv. Tutor. **18**(1), 466–490 (2016)
8. Honkavirta, V., Perala, T., Ali-Loytty, S., Piché, R.: A comparative survey of WLAN location fingerprinting methods. In: WPNC 2009, pp. 243–251. IEEE (2009)
9. Hossain, A.M., Soh, W.S.: Cramer-Rao bound analysis of localization using signal strength difference as location fingerprint. In: INFOCOM 2010, pp. 1–9. IEEE (2010)
10. Jager, T., Kohlar, F., Schäge, S., Schwenk, J.: On the security of TLS-DHE in the standard model. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 273–293. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32009-5_17
11. Kaemarungsi, K., Krishnamurthy, P.: Modeling of indoor positioning systems based on location fingerprinting. In: INFOCOM 2004, pp. 1012–1022. IEEE, March 2004
12. Konstantinidis, A., Chatzimilioudis, G., Zeinalipour-Yazti, D., Mpeis, P., Pelekis, N., Theodoridis, Y.: Privacy-preserving indoor localization on smartphones. IEEE Trans. Knowl. Data Eng. **27**(11), 3042–3055 (2015)
13. Li, H., Sun, L., Zhu, H., Lu, X., Cheng, X.: Achieving privacy preservation in WIFI fingerprint-based localization. In: INFOCOM 2014, pp. 2337–2345 (2014)
14. Li, S., Li, H., Sun, L.: Privacy-preserving crowdsourced site survey in WIFI fingerprint-based localization. EURASIP J. Wireless Commun. Netw. **2016**(1), 123 (2016)
15. Liu, H., Darabi, H., Banerjee, P., Liu, J.: Survey of wireless indoor positioning techniques and systems. IEEE Trans. Syst. Man Cybern. Part C (Appl. Rev.) **37**(6), 1067–1080 (2007)
16. Lohan, E.S., et al.: Indoor WLAN measurement data. Data Set (2014). http://www.cs.tut.fi/tlt/pos/MEASUREMENTS_WLAN_FOR_WEB.zip. Accessed June 2017
17. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 223–238. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48910-X_16
18. Roos, T., Myllymäki, P., Tirri, H., Misikangas, P., Sievänen, J.: A probabilistic approach to WLAN user location estimation. Int. J. Wirel. Inf. Netw. **9**(3), 155–164 (2002)
19. Sadeghi, A.-R., Schneider, T., Wehrenberg, I.: Efficient privacy-preserving face recognition. In: Lee, D., Hong, S. (eds.) ICISC 2009. LNCS, vol. 5984, pp. 229–244. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14423-3_16
20. Swangmuang, N., Krishnamurthy, P.: Location fingerprint analyses toward efficient indoor positioning. In: PerCom 2008, pp. 100–109. IEEE (2008)
21. Talvitie, J., Renfors, M., Lohan, E.S.: Distance-based interpolation and extrapolation methods for RSS-based localization with indoor wireless signals. IEEE Trans. Veh. Technol. **64**(4), 1340–1353 (2015)

22. Yang, Z., Järvinen, K.: The death and rebirth of privacy-preserving WIFI fingerprint localization with paillier encryption. In: INFOCOM 2018, April 2018
23. Yang, Z., Järvinen, K.: The death and rebirth of privacy-preserving WIFI fingerprint localization with paillier encryption. Cryptology ePrint Archive, Report 2018/259 (2018). http://eprint.iacr.org/2018/259
24. Zhang, T., Chow, S.S.M., Zhou, Z., Li, M.: Privacy-preserving WI-FI fingerprinting indoor localization. In: Ogawa, K., Yoshioka, K. (eds.) IWSEC 2016. LNCS, vol. 9836, pp. 215–233. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-44524-3_13