Server-side Fingerprint-Based Indoor Localization Using Encrypted Sorting

Andrew Quijano[1] and Kemal Akkaya[2]

[1]Department of Computer Science, Columbia University, New York, NY 10027, Email: Andrew.Quijano@columbia.edu
[2]Department of Electrical & Computer Engineering, Florida International University, Miami, FL 33174, Email: kakkaya@fiu.edu

***Abstract*—GPS signals, the main origin of navigation, are not functional in indoor environments. Therefore, Wi-Fi access points have started to be increasingly used for localization and tracking inside the buildings by relying on fingerprint-based approach. However, with these types of approaches, several concerns regarding the privacy of the users have arisen. Malicious individuals can determine a clients daily habits and activities by simply analyzing their wireless signals. While there are already efforts to incorporate privacy to the existing fingerprint-based approaches, they are limited to the characteristics of the homomorphic cryptographic schemes they employed. In this paper, we propose to enhance the performance of these approaches by exploiting another homomorphic algorithm, namely DGK, with its unique encrypted sorting capability and thus pushing most of the computations to the server side. We developed an Android app and tested our system within a Columbia University dormitory. Compared to existing systems, the results indicated that more power savings can be achieved at the client side and DGK can be a viable option with more powerful server computation capabilities.**

***Index Terms*—Efficiency, fingerprinting, localization, privacy, Wi-Fi, homomorphic encryption, socialist millionaire problem**

I. Introduction

While GPS has been revolutionary in its ability to easily locate a person outdoors accurately and provide directions to travel to their desired destination, it is not as useful to localize users indoors as it was not designed for that niche. There are various technologies that can overcome the shortcomings of GPS for indoor localization such as taking advantage of Wi-Fi Access Points (APs), sensors and RFID devices [1]. One of such promising technologies is based on fingerprinting concept where Wi-Fi signal strengths (i.e., Received Signal Strength (RSS)) on a floor of a building are pre-collected and shortest distances to those RSS values are computed to determine a user's location [2]. The pre-collection phase is called training and required before the localization can be done.

While there has been an extensive amount of research conducted to improve the performance and granularity of indoor localization, user privacy is usually not the first priority. Nevertheless, there is a growing concern among users that either malicious individuals or large enterprises such as Google use indoor localization systems to compromise their privacy [3]. Since smart phones are ubiquitous, it is conceivable that a malicious individual can collect their victim's MAC Address and data such as RSS to track the daily habits of their target when fingerprint-based approaches are used.

To address this growing concern, a number of solutions have been proposed in the literature [4]. The main objective is to either hide the identity of the user or prevent the AP/server from easily viewing the user's data. In case of fingerprint-based solutions, the approach was to hide the RSS data of user devices from the server by employing cryptographic techniques such as homomorphic encryption which enables computation on the encrypted data. In these approaches, the RSS values are encrypted before they are sent to a cloud server and the computations at the server is done on the encrypted data.

Recent approaches, utilized partially homomorphic systems to ensure privacy [5]–[7]. Typically, Paillier-based systems [8] are used to compute the distances and send them back to the user device in the encrypted form where they were decrypted to find out the minimum. This is needed because even though the distances can be computed and obtained in the encrypted form, they cannot be compared to each other or sorted in a list as Paillier cannot provide such a feature.

In this paper, we propose to extend the Paillier-based approaches by designing an alternative system where the computations for sorting the encrypted values are outsourced to the server rather than the client. This stems from the motivation that the server would have lots of computational power and through such outsourcing the client device can save more energy. To this end, we adopted the homomorphic [9] algorithm which has the capability to compare two encrypted values named after the authors Damgård, Geisler and Krøigaard (DGK). Basically, we introduce an encrypted sorting mechanism at the server and relay only the top result(s) (i.e., k minimum values) to the client.

We implemented an Android app that uses both systems which can localize users in the third floor of the Broadway dormitory at Columbia University. For comparison, we implemented a Java package containing the DGK and Paillier homomorphic encryption systems both at the client and server sides and studied the performance trade-offs. The results indicated that while DGK is faster than Paillier in both a client or server localization approach. In addition, our hypothesis of outsourcing the encrypted comparison in our server side localization approach increased computation time but we used less battery after repeated testing.

The structure of this paper is as follows. In the next section, we summarize the related work. Section III describes

978-1-7281-4121-3/19/$31.00 ©2019 IEEE
DOI 10.1109/MASSW.2019.00017

the background which include fingerprint-based localization, homomorphic encryption and its applications in our system. Section V provides a comprehensive performance evaluation of both the client and server-based indoor localization system. Finally, Section VI concludes the paper.

II. Related Works

There has been a number of fingerprint-based approaches in the literature. For instance, Ahmad et al. follow an approach in the sense that all localization data is to be stored on the individual device rather than a centralized remote server [10]. Their model uses a Wi-Fi fingerprint database to localize a user and their system can dynamically update itself as the phone is consistently collecting data. It has been proven to be as accurate as traditional centralized fingerprint databases as it also passive collects beacon-stuffing information from APs as well. Beacon stuffing releases information to the phone such as Region ID, Location Number, AP ID, and AP fingerprint that is recorded at its location [10]. This system provides essentially perfect security to the user, assuming the phone is not compromised. Yet, there is a danger to the network administrator as the APs leak valuable information that could be useful of a malicious individual's reconnaissance phase in order to attempt to hack the network. Therefore, it is likely that network administrators would probably disable this information being leaked.

Zhu et al. have created an indoor localization system using a differentially private algorithm designed to both protect the fingerprint database and the user's data [11]. A common definition of differentially private is when looking at the output of the algorithm, a third party cannot distinguish any uniquely identifiable data was included in the original dataset or not. Therefore, they used the exponential mechanism on their dataset to conceal the identifiable attributes. Finally they used the J48 Decision tree classifier and showed that their system is still about 90% accurate in localizing via Wi-Fi fingerprint data. Zhu et al. mention that they tested their indoor location system only on synthetic data. Therefore, there is a possibility that it may have a reduced accuracy when it is tested indoors. Also, there is no discussion on the computational overhead for the process of creating a differentially private system when it comes mobile devices to localize with the assistance of a fingerprint database server. While mobile phones have improved dramatically in their computational power, they will consistently have less computational resources or battery lifetime than a laptop.

In this work, we improve the work in [7]. In this work, the Paillier system has been used to perform encrypted computations and send them back to the client. We will revise this approach to utilize DGK [9] which is a homomorphic system with encrypted computing and comparing capabilities and thus push the burden to the server side.

III. Background

A. Fingerprint-based Localization

The idea behind indoor localization using fingerprints is based on the signals collected from APs for each user. Specifically, a user receives Wi-Fi signals from various APs at a specific location, which is referred to as a fingerprint, and these are stored in a database. At every location, this signal collection is done for a given area. The way these locations are picked is based on the expected accuracy of the localization. The higher the resolution of this process, the better the accuracy will be. To increase the accuracy, crowd sourcing can be used with some incentives. Once all location fingerprints are stored in a database, this information can now be used to determine the location of a user moving around. This phase of collecting fingerprints is called training as shown in Fig. 1.

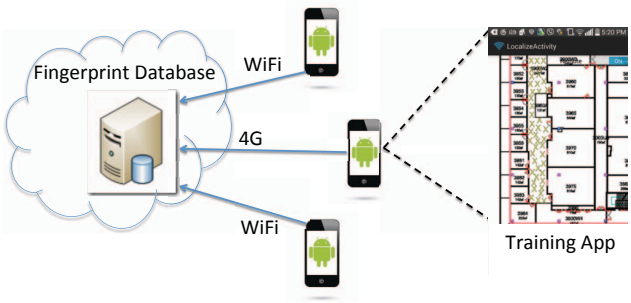

Fig. 1: Crowd sourcing-based training model [7].

After the training phase, the actual localization occurs when a user receives signals from an AP and then this signal is used to find out a similar signal in the database. Essentially, the closest signal which corresponds to a specific point in the database will be found. This process is achieved through a similarity distance computation. There may be different approaches to determine the closest distance. One simple approach is to find the single fingerprint while another approach might find k closest fingerprints and take the centroid of these fingerprints [7].

B. Homomorphic Encryption

Homomorphic encryption enables computation on encrypted numbers so as to ensure privacy of the actual numbers. While most of the homomorphic systems are partial (i.e., they only support two arithmetic operations), recent years witnessed fully homomorphic systems that support all arithmetic operations [12]. In this paper, we utilized the well-known Paillier [8] algorithm and DGK algorithm [13] which are partially homomorphic systems. Specifically, these homomorphic encryption schemes share the following two traits which permit for computation on encrypted values. Let E(x) and E(y) be encrypted x and y respectively. Under homomorphic encryption E(x)E(y) = E($x + y$). Also, they both support scalar multiplication: If we have E(x) and plain-text y, we can compute E(xy) by E(x)^y.

DGK has also the benefit of supporting a built-in comparison protocol for comparing encrypted numbers. This protocol is one solution to the "socialist millionaire's" problem, where the objective is to generate a true inequality without decrypting the cipher-texts [14]. Veugen [9] describes Protocol 2 and Protocol 4 in his work to compare encrypted numbers. When using Protocol 4 to compare DGK values, it reliably computes $[[x > y]]$. The main benefit of Protocol 4 is that the constraints enforced on the plain-text space is much smaller, allowing for successful and secure comparisons with DGK encrypted values.

C. Privacy in Fingerprint-based Localization

One of the issues raised in fingerprinting is privacy. This occurs because the fingerprints collected from a user device are exposed to the server which stores all the fingerprints. Consequently, these fingerprints could be used to determine the locations of a specific user and track him/her during the day. To address this issue, our previous work in [7] proposed employing a partial homomorphic system based on Paillier. This approach encrypts the fingerprints and when computing the closest distance at the server, it performs encrypted computation and eventually sends the encrypted results to the client to decrypt and compute the closest one.

IV. DGK-based Fingerprint Localization

In this section, we describe our proposed approach for fingerprint-based localization in details.

A. Overview

Similar to the works in the literature such as [7], our approach utilizes homomorphic encryption when collecting data from users and doing computation at the database server. However, as previous works utilized homomorphic systems such as Paillier which cannot do comparison of encrypted numbers, the computed distance results at the server were being passed back to the client which can then decrypt these results and can find the closest distance. This requires communication of all of these values as well as decryption overhead at the client. In this paper, we propose to eliminate this overhead by utilizing a comparison protocol based on DGK and hence conduct comparison of encrypted distances at the server. The DGK comparison protocol helps us to determine the closest distance to user's fingerprint and then we communicate only that encrypted value to the user. We explain the details of this approach in the balance of this section.

B. Training Phase of the Proposed Approach

We implemented a smart phone app that follows our proposed fingerprint-based approach. This smart phone app was started to be tested at Florida International University (FIU) Engineering Center but once the REU student left the program, it continued at his dormitory at Columbia University.

The fingerprint-based application first requires fingerprints (training data) stored in a remote server that is accessible by the client's device via Wi-Fi or LTE. In our case, the server uses a MySQL database to store the fingerprints and utilizes the JDBC driver to retrieve data from the database and obtain a floor map of Broadway dormitory third floor. The database contains a table with the following columns:

1) Map ID
2) Location on the map
3) MAC-address of an AP
4) RSS value associated with the AP
5) Device manufacturer, model, and software version
6) Date and time

The Android app has a training activity which allows users to collect and store the fingerprints to the remote database as shown in Fig. 1. A user does this by going to the location where s/he wants to fingerprint and by tapping the screen on the matching location in the floor map. The app will collect the coordinates, APs detected and their corresponding RSS. Upon completion, a blue "X" appears informing the user the location is recorded in the finger database.

We conducted our scans with a range of about 5 - 10 feet apart from each other to give a precise location. Whenever possible, we would conduct scans within rooms with closed doors as to give a more distinct signature. As a result, we obtained a fingerprint database as follows:

$\mathcal{D} = \langle (x_i, y_i), V_i = \{RSS_j, AP_j\}_{j=1}^{N_{\mathrm{AP}}} \rangle_{i=1}^{N_{\mathrm{F}}}$

where (x_i, y_i) is the location of the fingerprint, V_i is the fingerprint tuple comprised of AP_i and its fingerprint RSS value. N_{AP} is the total number of fingerprint reading scan result APs for a location and N_{F} is the total number of fingerprints in the database.

C. Fingerprint Database pre-processing

Once the database has sufficient fingerprints, it needs to pre-process the training data to create look up tables. This is because not all the locations will see the same APs and some AP signal values might be empty in the database.

Basically, if the AP is not found in the database, we use v_c = -120 to automatically setup the missed constant determined by Parmengol et al. [7]. When completed, the look up tables will list all X, Y coordinates and their signal values for each AP's MAC address. Upon completion of the pre-processing phase, the remote server will be capable of securing localizing a user.

D. Homomorphic Distance Computation

The localization scheme works by computing the Euclidean distance of the user's (AP, RSS) pairs to each row on the look up table to generate N_{F} rows of distance and coordinate pairs.

More specifically, a client first conducts a localization scan and obtains a list of (AP, RSS) pairs. The client then obtains the columns in the look up table.The client organizes their scan results correspond with the columns of the server look up table. Then, the client sends the normalized (AP, RSS) pairs and the public key to the server. To ensure privacy is preserved, the RSS data is encrypted via homomorphic encryption.

In summary, the algorithm computes the distance squared using homomorphic encryption. It exploits the fact that d^2 =

$(x_2-x_1)^2 = (x_2^2)+(-2x_1)(x_2)+(x_1^2)$. Here x_2 is the server's fingerprint RSS data and x_1 is the localization scan RSS. The terms S_1, S_2, and S_3 correspond to the terms in parenthesis respectively. The pseudo-code for this computation is given in Algorithm 1.

Algorithm 1 MCA Distance Computation

```
1: L ← gatherLookupTableData()
2: scanAPs ← APs found in Localization Scan
3: for each fprint = ((x_i, y_i), V_i) in L do
4:     S_{i,1} ← 0
5:     for each AP in scanAPs do
6:         if fprint contains AP then
7:             S_{i,1} += RSS_{i,j}^2
8:             [[S_{i,2}]] += [[S_{2,j}]]^{RSS_{i,j}}
9:         else
10:            S_{i,1} += (v_c)^2
11:            [[S_{i,2}]] += [[S_{2,j}]]^{v_c}
12:        end if
13:    end for
14:    [[S_{i,1}]] ← encrypt(S_{i,1})
15:    [[d_i]] ← [[S_{i,1}]] + [[S_{i,2}]] + [[S_3]]
16:    result ← (x_i, y_i), d_i
17:    add result to resultList
18: end for
19: return resultList
```

As seen, this algorithm returns a list (i.e., $resultList$) which basically includes all distance computations to client's fingerprint x_1. All the elements of this list is encrypted. We followed both Paillier and DGK to perform this operation as they both support partial homomorphism.

E. Server-based Minimum Distance Computation

We propose to use DGK algorithm for processing the $resultList$ instead of passing it back to the client. Since DGK has the ability to compare encrypted numbers, the algorithm will compare all the elements and determine the top k of these. In our case, we followed an approach which computes the closest distance (i.e., top candidate) to a given fingerprint and reports it back to client.

To obtain the k smallest encrypted numbers, we used Bubble sort as our template as shown in Algorithm 2 and modified it to fit our needs. First, we modified line 2 to loop only k times and we changed the condition to check if arr[i] $<$ arr[i + 1]. Upon running this protocol, the smallest values are located in ascending order ranging from arr[n - 1], arr[n - 2], ..., arr[length - 1 - (k - 1)]. The main benefit of using this system over sorting the encrypted array is that its overhead is O(nk) rather than O(n^2) to sort the whole encrypted array. Considering that Protocol 4 can take about 0.3 - 0.5 seconds [9] for one execution, it is imperative we minimize the number of comparisons.

Using Algorithm 2, where k = 1, we obtain the smallest encrypted distance. The server then returns the corresponding

Algorithm 2 Integrating Protocol 4 and Bubble Sort

```
1: n ← arr.length
2: for i = 0 to k do
3:     for j = 0 to n - i - 1 do
4:         if Protocol4(arr[j], arr[j + 1]) == 0 then
5:             Cipher-text temp = arr[j]
6:             arr[j] = arr[j+1]
7:             arr[j + 1] = temp
8:         end if
9:     end for
10: end for
```

encrypted coordinates to the client. This overall process is summarized in Fig. 2.

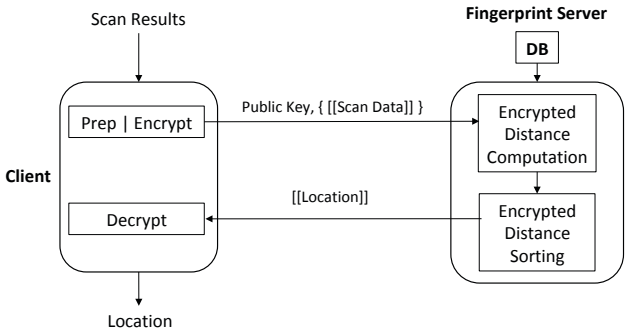

Fig. 2: Overall process for server-based fingerprinting localization. The use of brackets [[v]] on the value v denotes that the it is encrypted.

V. Performance Evaluation

In this section, we discuss the results of our experiments to test the performance of both our proposed approach.

A. Experiment Setup

We have implemented Paillier and DGK in Java using the standard *BigInteger* library, using the author's paper as reference. Also, we used a remote server which is a Desktop computer with an Intel 64-bit i7-4790K CPU running at 4 GHz with 24 GB of RAM. We have used the Samsung Galaxy 6S Edge which has a 8 core CPU running at 2.1GHz and 4 GB of RAM for both training and testing the performance of both localization systems.

We created 19 fingerprints in our training data, in which we detected 265 APs. We had filtered 90% of the APs so our column only consisted of 26 APs that were detected 8 - 13 times. We conducted 20 different trials for 2048 bits key lengths and we have kept track of how much battery was consumed upon completion.

B. Performance Metrics and Baselines

We considered the following metrics to access performance:

- *Execution Time*: This is the time for all computations and communication between the user and server.

- *Battery Life*: This is defined as the energy consumption for the user device's battery.

For comparison, we compared our server-side localization approach[1] with Parmengol et al.'s [7] client-side approach. For both approaches, we had the client send either DGK or Paillier encrypted distances. Therefore, our implementation of the comparison protocol can compare both Paillier and DGK cipher-text [2]. In both cases, we used 2048 bits of key sizes as 1024 bits is not considered secure enough anymore.

C. Performance Results

We first checked the execution time results. Based on our results in Table I, we have observed an expected increase in time with the server-side approach. For instance, the client-side with Paillier took 11.13 seconds while it was 16.65 seconds in our DGK approach. However, we observed that DGK localization is much faster overall because DGK encryption and decryption operations are much faster compared to Paillier.

TABLE I: Overhead of both approaches with 2048-bit keys

Approach	Time (Paillier)	Time (DGK)	Energy (Paillier/DGK)
Client Side	11.13	6.71	4%
Server Side	27.93	16.65	3%

We then looked at the energy consumption for both cases. Note that using Paillier or DGK did not matter for client or server sides for energy consumption. Upon completion of our tests, the client system used about 4% of the phone's battery and our system has used 3% battery. This is because when analyzing Veugen's comparison protocol, the server is executing most of the computations, which was our objective. So if this localization computations are repeated many times, for instance, as part of a tracking app, then our server-based approach would be standing out for saving significant battery power. In other words, we can safely say that if we had repeated until the battery ran out, we would have completed more localizations using our model. This would be especially useful if the user highly valued their battery life over time to locate themselves such as a disaster scenario. In summary, DGK can be a viable option whether it is used in a client-based or server-based approach for privacy preserving indoor localization due to its speed and our server-side approach has energy saving features.

VI. Conclusion

In this paper, we introduced a server-based fingerprint based localization scheme using DGK homomorphic encryption system. The goal was to push the comparison of encrypted distance values to the server so that the client can save energy and transmission time due to transmission of those values back to client in the existing approaches. We developed an Android app for indoor localization to test the proposed approach in student dormitories. Our results demonstrated that our system based on DGK can save battery life of the mobile device, while still preserving user privacy with a slight increase in execution time. However, we also found out that if implemented at the client-side, DGK can significantly outperform the existing approach with Paillier in terms of execution time.

[1]The code for the Indoor Localization Android app and server are located at: https://github.com/AndrewQuijano/SSTREU2017

[2]We have made our implementations of all the homomorphic encryption systems and their protocols publicly available at: https://github.com/AndrewQuijano/Homomorphic_Encryption

Acknowledgments

Andrew Quijano was supported by the US NSF REU Site at FIU under the grant number REU CNS-1461119. The work is also supported in part by a grant from Cisco Silicon Valley Foundation. We would also like to thank Alice Niu for helping us with collecting experimental results and Dr. Samet Tonyali for his help in understanding the details of DGK.

References

[1] H. S. Hasan, M. Hussein, S. M. Saad, and M. A. M. Dzahir, "An overview of local positioning system: Technologies, techniques and applications," *International Journal of Engineering & Technology*, vol. 7, no. 3.25, pp. 1–5, 2018.

[2] C. BASRI and A. El Khadimi, "Survey on indoor localization system and recent advances of wifi fingerprinting technique," in *2016 5th International Conference on Multimedia Computing and Systems (ICMCS)*. IEEE, 2016, pp. 253–259.

[3] V. Sadhu, D. Pompili, S. Zonouz, and V. Sritapan, "Collabloc: Privacy-preserving multi-modal localization via collaborative information fusion," in *2017 26th International Conference on Computer Communication and Networks (ICCCN)*. IEEE, 2017, pp. 1–9.

[4] A. Konstantinidis, G. Chatzimilioudis, D. Zeinalipour-Yazti, P. Mpeis, N. Pelekis, and Y. Theodoridis, "Privacy-preserving indoor localization on smartphones," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 11, pp. 3042–3055, 2015.

[5] S. Tonyali, K. Akkaya, N. Saputro, A. S. Uluagac, and M. Nojoumian, "Privacy-preserving protocols for secure and reliable data aggregation in iot-enabled smart metering systems," *Future Generation Computer Systems*, vol. 78, pp. 547 – 557, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167739X17306945

[6] N. Saputro and K. Akkaya, "On preserving user privacy in smart grid advanced metering infrastructure applications," *Security and Communication Networks*, vol. 7, no. 1, pp. 206–220, 2014.

[7] P. Armengol, R. Tobkes, K. Akkaya, B. S. iftler, and I. Gven, "Efficient privacy-preserving fingerprint-based indoor localization using crowdsourcing," in *2015 IEEE 12th International Conference on Mobile Ad Hoc and Sensor Systems*, Oct 2015, pp. 549–554.

[8] P. Paillier *et al.*, "Public-key cryptosystems based on composite degree residuosity classes," in *Eurocrypt*, vol. 99. Springer, 1999, pp. 223–238.

[9] T. Veugen, "Correction to" improving the dgk comparison protocol"." *IACR Cryptology ePrint Archive*, vol. 2018, p. 1100, 2018.

[10] N. M. Ahmad, A. H. M. Amin, S. Kannan, A. M. M. Ali, M. F. Abdollah, and R. Yusof, "A passive and privacy-friendly area based localization for wireless indoor networks," in *2016 IEEE Region 10 Symposium (TENSYMP)*. IEEE, 2016, pp. 213–218.

[11] Y. Zhu, Y. Wang, Q. Liu, Y. Liu, and P. Zhang, "Wifi fingerprint releasing for indoor localization based on differential privacy," in *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*. IEEE, 2017, pp. 1–6.

[12] M. Van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan, "Fully homomorphic encryption over the integers," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2010, pp. 24–43.

[13] I. Damgård, M. Geisler, and M. Krøigaard, "Efficient and secure comparison for on-line auctions," in *Australasian Conference on Information Security and Privacy*. Springer, 2007, pp. 416–430.

[14] F. Boudot, B. Schoenmakers, and J. Traore, "A fair and efficient solution to the socialist millionaires problem," *Discrete Applied Mathematics*, vol. 111, no. 1-2, pp. 23–36, 2001.