The ISC Int'l Journal of Information Security
ISeCure

July 2022, Volume 14, Number 2 (pp. 215–227)
http://www.isecure-journal.org

Location Privacy Preservation for Secondary Users in a Database-Driven Cognitive Radio Network

Zeinab Salami [1,*], Mahmoud Ahmadian-Attari [1], Mohammad Reza Aref [2], and Hoda Jannati [3]

[1] *Department of Electrical Engineering, K. N. Toosi University of Technology, Tehran, Iran.*
[2] *Department of Electrical Engineering, Sharif University of Technology, Tehran, Iran.*
[3] *School of Computer Science, Institute for Research in Fundamental Sciences (IPM), Tehran, Iran.*

ARTICLE INFO.

Article history:
Received: August 12, 2020
Revised: August 20, 2020
Accepted: February 13, 2022
Published Online: February 20, 2022

Keywords:
Database-Driven CRN, Location Privacy, Homomorphic Encryption

Type: Research Article

doi: 10.22042/ISECURE.2022.243102.568

dor: 20.1001.1.20082045.2022.14.2.8.0

ABSTRACT

Since their introduction, Cognitive Radio Networks (CRN), as a new solution to the problem of spectrum scarcity, have received great attention from the research society. An important field in database-driven CRN studies is pivoted on their security issues. A critical issue in this context is user's location privacy, which is potentially under serious threat. The query process by secondary users (SU) from the database is one of the points where the problem rises. In this paper, we propose a Privacy-Preserving Query Process (PPQP), accordingly. This method lets SUs deal in the process of spectrum query without sacrificing their location information. Analytical assessment of PPQP's privacy preservation capability shows that it preserves location privacy for SUs against different adversaries, with very high probability. Relatively low communicational cost is a significant property of our protocol.

1 Introduction

The idea of Dynamic Spectrum Access (DSA) has received great attention over the past decade, due to ubiquitous wireless network availability and rapid growth of wireless technologies. As opposed to conventional static spectrum management strategies, in DSA, spectrum bands are not exclusively used by one group of users. Instead, a second network consisting of Secondary Users (SU) are allowed to opportunistically access the unoccupied portions of the spectrum as long as they do not cause harmful interference to the license-holders or the Primary Users (PU). The technology that enables implementation of this idea is Cognitive Radio (CR) and the secondary network is therefore called a Cognitive Radio Network (CRN).

In a CRN, the SUs usually apply two methods to detect locally unused frequency bands or white spaces. The first approach is sensing the spectrum. In this method, by listening to a channel, an SU determines whether any PU in its vicinity is utilizing the channel or not. This technique has been shown not to be effective as a standalone method [1]. The second method which has been adopted by the Federal Communications Commission (FCC) in its latest rule [2], is querying a database to achieve Spectrum Availability Information (SAI). This alternative has been considered the most efficient technique currently available to share the unused spectrum [1].

In spite of providing several advantages, querying a white space database in its present manner, im-

[*] Corresponding author.

Email addresses: z_salami@ee.kntu.ac.irm, m_ahmadian@kntu.ac.ir, aref@sharif.edu, hodajannati@ipm.ir

ISSN: 2008-2045 © 2020 ISC. All rights reserved.

poses serious privacy concerns to the users. This is because according to the latest standard, i.e. Internet Engineering Task Force (IETF) Protocol to Access White-Space (PAWS) Databases [3], SUs must issue their precise GPS coordinates as part of their query. An attacker, e.g. the untrusted database, can easily misuse this information to breach user's privacy. This is while usually users are strictly reluctant to share their location information. This is because breach of users' location privacy can reveal their personal habits, interests and secrets and may also expose them to unwanted advertisements and location-based spam or even make them victims of blackmail or physical violence [4]. Other sources of location information leakage that may arise from database-driven CRN architecture are the Maximum Transmit Power (MTP) and the list of available channels in the query's response [5]. To address these problems, location privacy in database-driven CRNs has gained researchers' great attention in recent years.

To this end, in this paper we design a protocol that prevents SU's location being revealed during the querying process. In our protocol we combine the ideas of spatial cloaking and homomorphic cryptography. Since their introduction, homomorphic cryptosystems have been used widely in providing solutions to problems in the context of privacy [6]. For instance, some existing works on Location Based Services (like [7]) utilize such systems for protecting users' privacy. Another idea to protect location privacy is cloaking methods. Spatial cloaking is a technique to blur a user's exact location into a spatial region in order to preserve her location privacy (like in [8]). The most popular privacy requirement for the spatial cloaking technique is k-anonymity [9]. It means that the user's location information reported to the service provider should be indistinguishable from at least $k-1$ other users. We take advantage of both ideas to design a privacy protecting protocol for relatively stationary users with outstanding property of low communicational cost.

The contributions of this paper can be summarized as follows. We propose the Privacy-Preserving Query Process (PPQP) as a method that lets SUs access the SAI of their cell while keeping their location coordinates unrevealed. The required level of privacy in PPQP can be achieved through adjustment of some user-controlled parameters. In fact higher level of privacy could be achieved in the expense of more computational complexity. In our design we try to avoid imposing too much communicational overhead to the cognitive radios. We analyze the security of our protocol and show that neither the untrusted database, nor any PU or even any external attacker can gain any probabilistic advantage in attempt to find out SU's location. Furthermore, we show through a complexity analysis, that the communicational overhead of PPQP is relatively low and is smaller compared to previous works. Finally, we evaluate the performance of our protocol through simulations to observe the performance in terms of runtime.

The organization of this paper is as follows. Section 2 reviews the related work. In Section 3 we talk about homomorphic encryption, describe the Paillier cryptosystem [10] and study a technique based on its properties. Section 4 describes the system model, including system architecture and the adversary model. We introduce PPQP in Section 5 and the privacy of the protocol is analyzed in Section 6. In Section 7 we present a complexity analysis of the protocol. We then evaluate the performance of our protocol in Section 8. Finally, Section 9 concludes this paper.

2 Related Work

During few recent years users' location privacy in database-driven CRNs has become a center of focus among researchers who work on contexts associated with security in CRNs. Works that issue this problem are mostly based on either k-anonymity [9] or Private Information Retrieval (PIR) [11]. However, some adopt miscellaneous other concepts.

Techniques based on k-anonymity attempt to guarantee that a user's location is indistinguishable among k users. Li *et al.* [12] apply k-anonymity to introduce a method, which cuts off the relationship between SU's location and its register data in the DB. This method protects SUs' location privacy during the commitment phase. In their framework they consider a number of Base Stations (BS), which SUs are associated with. Zhang *et al.* present a method in which SUs query the DB by sending a cloak region that includes their own location [13]. They use another k-anonymity approach to protect PUs' location privacy, too.

Some other techniques on the other hand, are based on PIR technique. Gao *et al.* [14] exploit some blinding factors to hide SU's location during the query process. The SU keeps the secure blinding parameters to retrieve the SAI of its location later. This work applies PIR method of [15]. Troja *et al.* [16] offer a method based on another PIR [17]. The method is mainly efficient for mobile SUs. In this method the coverage area is divided into multi-cell blocks and neighbor SUs exchange SAI, so that fewer queries from the DB would be necessary. Same authors propose another method [18], which takes advantage of Hilbert space filling curve [19]. This work also mainly considers mobility of users.

A number of other works apply different concepts to build their techniques. Salami *et al.* introduce a cryptography-based protocol for spectrum sharing [20]. Taking advantage of some well-known cryptosystems, their protocol protects location privacy for SUs and PUs, simultaneously. They consider Base Stations for both primary and secondary networks. Other work that protects bilateral location privacy of both PUs and SUs is [21]. Both groups of users solve an optimization problem that maximizes their bilateral utility considering differential privacy [22]. They then obfuscate their location accordingly. Chen *et al.* also protects location privacy for users of primary and secondary networks [23]. To let parallel queries they use data-oblivious sorting networks in their design. They combine garbled circuits [24] and XOR secret sharing [25] on the DB side. Grissa *et al.* propose an approach that offers an unconditional privacy to SUs within the DB's coverage area [26]. In this scheme SU only sends its characteristics, but not its location to the DB using cuckoo filter [27].

Although the location privacy of PUs is of paramount importance, especially in the case of military incumbent systems that have stringent requirements in terms of security and privacy, little work has exclusively focused on it [5]. Bahrak *et al.* describe an attack by SUs to geolocate a PU and then propose techniques to thwart against the attack [28]. Authors in [29] and [30] have a general view towards the issue. The former explores whether PUs can retain a critical level of privacy in a spectrum sharing network, and the latter develops an analytical model to analyze the vulnerability of PU's frequency to inference attacks.

Therefore, the previous work that also applies spatial cloaking region idea is the one by Zhang *et al.* [13]. In their method, the SU sends a square cloak region containing its real location to the DB. In response, the DB sends the SAI of the whole square ($n \times n$ cells) to the requesting SU. In our work, however, the information of $2n$ cells is transmitted. Compared with download of all $n \times n$ cells, a reduction of $\frac{2}{n}$ times in the communicational complexity is observed for the same level of privacy.

3 Preliminaries

3.1 Homomorphic Encryption

In mathematics, the term homomorphic describes the transformation of one dataset into another while preserving relationships between elements in both sets. Homomorphic encryption schemes are cryptosystems that allow computations to be performed on data without decrypting it. They allow computations to be carried out on ciphertext to generate an encrypted result which, when decrypted, matches the result of some other known operations on the plaintext [6].

Homomorphic cryptosystems can be served as a useful tool for hiding the desired information throughout execution of a protocol, by conducting some of the original operations in the encryption domain. In other words, one can implicitly make the plaintext undergo certain operations by performing other specific operations on the corresponding ciphertext, in the encryption domain. This is a desirable feature in modern communication system architectures.

RSA [31] is the first public-key encryption scheme with homomorphic properties. In 2009, IBM researcher Craig Gentry came up with the first fully homomorphic encryption scheme [32]. Unfortunately, Gentry's method also adds immense computational requirements to computational tasks that would be simple with unencrypted data and there is a long way to go before it will be widely usable. One of the most well-known homomorphic cryptosystems is the Paillier cryptosystem [10], proposed by Pascal Paillier in 1999. It will be described briefly in the next subsection.

3.2 Paillier Cryptosystem

The general structure of Paillier public key encryption scheme is as follows.

Key Generation: To establish a public key two random primes, p and q, are selected in such a way that the factorization problem is intractable and $N = p \times q$ is constructed. However, the module with which encryption and decryption will be performed, is N^2. Then a random integer $g \in Z^*_{N^2}$ is selected, such that $\mu = \left(L\left(g^\lambda\right) \bmod N^2\right)^{-1} \bmod N$ exists, where $L(z) \bmod N^2 = \frac{z-1}{N}$. ($g$ is sometimes said to be semi-random, since there are a few values that do not satisfy the existence of μ.) The integer μ, along with $\lambda = Lcm(p-1, q-1)$ form the private key of the system, $K_{pri} = (\lambda, \mu)$. And $K_{pub} = (N, g)$ is distributed as the public key.

Encryption: To encrypt a message $m \in Z^*_N$, a random number $s \in Z^*_N$ is chosen. Then the ciphertext $c \in Z^*_{N^2}$ is produced in this way: $c = E_s(m) = g^m s^N \bmod N^2$.

Decryption: Having the ciphertext c, the plaintext message m could be obtained by:

$$m = D(m) = \left(\frac{L\left(c^\lambda \bmod N^2\right)}{L\left(g^\lambda \bmod N^2\right)} \bmod N \right),$$

where $L(z) = \frac{z-1}{N}$.

3.3 Encryption Domain Matrix Multiplication

The Paillier cryptosystem has two important homomorphic properties. The first one is the additive homomorphic property. It means that one can compute the addition of two plaintexts, m_1 and m_2, i.e. $(m_1 + m_2)$, in the encryption domain, given only their ciphertexts, $E_{s_1}(m_1)$, and $E_{s_2}(m_2)$. This is done by multiplying the two latter:

$$D(E_{s_1}(m_1).E_{s_2}(m_2) \bmod N^2) = m_1 + m_2.$$

The second homomorphic property of Paillier is obtaining the multiplication of two plaintexts, m_1 and m_2, i.e. $(m_1 \times m_2)$, in the encryption domain, given $E_{s_1}(m_1)$ and m_2 as follows:

$$D((E_{s_1}(m_1))^{m_2} \bmod N^2) = m_1 \times m_2.$$

The above two properties, imply a trick to apply in matrix calculation. Imagine we have a plaintext matrix, $A = (a_{j,k}), 0 \leq j < m, 0 \leq k < n$ and a ciphertext (encrypted) matrix, $C = E_s(B)$, i.e. $c_{i,j} = E_{s_{ij}}(b_{i,j}), 0 \leq i < t, 0 \leq j < m$. We would like to obtain the encryption domain multiplication of matrix A with matrix C, without knowing matrix B. The encrypted multiplication matrix, $D = (d_{i,k}), 0 \leq i < t, 0 \leq k < n$, can be obtained as:

$$d_{i,k} = \prod_{j=1}^{n} (c_{i,j})^{a_{j,k}} = \prod_{j=1}^{n} \left(E_{s_{ij}}(b_{i,j})\right)^{a_{j,k}}.$$

This operation is called Encryption Domain Matrix Multiplication (EDMM). Now on, we will indicate EDMM operator by $*$. That is: $D = A * E_s(B)$.

4 Protocol Model

4.1 System Model

Our system consists of a primary and a secondary network. A database (DB) houses an up-to-date repository of spectrum usage information of all PUs throughout its coverage area, which is assumed to be divided into M equal cells. When an SU requires a channel, it queries the DB to get the SAI. The SAI is calculated according to the specific ruleset of the network.

As was indicated, the current standard which is applied in database-driven CRNs is the Internet Engineering Task Force (IETF) Protocol to Access White-Space (PAWS) Databases [3]. According to this standard the SAI should contain the following information:

- A list of some available channels, like ch_i (the DB announces only a subset of all available channels and not the full list of them)
- Maximum Allowable Transmission Power (MATP) on each channel, P_i
- A time stamp, t_i, indicating how long the channel is available for secondary usage throughout that cell.

Therefore the SAI of each cell can be shown in the general form of (ch_i, P_i, t_i).

After receiving the SAI, the querying SU chooses one channel according to its own strategies and priorities. These could be for example [3]:

- The frequencies that permit the highest power
- Frequencies that are available for the longest period of time
- Just the first set of frequencies that matches its needs

Then the user starts operating on that channel considering the MATP and the valid time span. In our system model we also assume that SUs are able to perform cryptographic operations.

4.2 Adversary Model

The final goal of an adversary in the context of location privacy is to find out the location of a user. We define the privacy requirement in this paper as an adversary's incapability to find the exact location of an SU. With this regard, three kinds of adversaries could be imagined:

(1) A curious-but-honest database that follows the protocol honestly, but is willing to acquire the location coordinates of querying SUs.
(2) A PU that tries to understand where the location of a user from the secondary network (a SU) is.
(3) An external adversary that wants to find out the location of the SUs. This could be due to curiosity, financial ends, or any other reason.

Adversaries 2 and 3 can be analyzed quite similarly. We will present a detailed privacy analysis of PPQP against each adversary in Section 6.

5 Protocol Description

In this section we describe the PPQP protocol in a step by step manner.

5.1 Overview of the Protocol

- Initialization:
  Step 1: Every SU chooses a Query Region (QR), which will remain unchanged. The size

of the QR, i.e. the number of cells inside it, is decided according to the level of privacy that the SU intends to achieve. This size cannot be larger than a limit which is predefined by the DB, in order to restrict computational complexity of the protocol. The SU announces its QR to the DB. The SU also chooses a pair of public and private keys for the Paillier cryptosystem.

- Query Process:
  Step 2: Whenever an SU needs a channel, it sends a Channel Request Message (CRM) to the DB. This message contains all required characteristics according to the ruleset besides the SU's ID.
  Step 3: The SU encrypts the coordinates of row and column in which it is located inside the QR with its own private key and sends the encrypted values to the DB.

- Query Reply:
  Step 4: DB arranges SAI for cells within the QR of the requesting SU inside a matrix. This step and the previous one could be performed simultaneously and not necessarily in sequential order.
  Step 5: Then the DB multiplies the SAI matrix with user-sent vectors in encryption domain and sends the result back to the SU.
  Step 6: The SU decrypts the received message from the DB to obtain the SAI of its cell. Then it picks up its desirable channel according to its requirements and starts activation on that channel. An overview of the protocol is given in Figure 1.

5.2 Protocol Details

We will now present a detailed description of the protocol.

Step 1: At the beginning of the whole protocol, each SU decides on a QR. A QR is an $n \times m$ rectangle, where n and m are upper-limited by the DB. Every cell in the QR is indicated by a (row,col) pair, where $0 \leq row < n$ and $0 \leq col < m$ are the row and column number, respectively. The QR is arbitrarily chosen around SU's real location, $loc_{SU} = [x_{SU}, y_{SU}]$. We will show the corresponding index of SU's cell within the QR by (r,c). The coordinates of (0,0) cell (the origin cell), $loc_o = [x_o, y_o]$, along with n and m uniquely describe a QR. This triple is reported to the DB:

$$QR := \langle [x_o, y_o], n, m \rangle.$$

The QR is encrypted before reporting to the DB. For this purpose, SU and the DB may utilize a lightweight stream cipher. Therefore they should share a secret key in advance. Every SU also chooses a pair of public and private keys, K_{pub} and K_{pri}, for the Paillier cryptosystem, where $K_{pub} = (N, g)$ and $K_{pri} = (\lambda, \mu)$.

Step 2: When an SU needs a channel, it sends a CRM to the DB. The CRM contains user ID, which has been assigned by the DB through SUs initial registration to the network. Other parameters, such as antenna height and other characteristics may also be necessary according to the ruleset that governs the network.

Step 3: The SU is located in cell (r,c) of the QR. This can be translated into two 1-Hamming weight vectors, of which only the $(r)^{th}$ or $(c)^{th}$ element is equal to one. The SU applies Paillier to encrypt the two vectors element-wise and sends the resulted bi-vectors to the DB:

$$\bar{r}_{enc} = (E_{s_1}(0), \ldots, E_{s_r}(1), \ldots, E_{s_m}(0)),$$

$$\bar{c}_{enc} = (E_{t_1}(0), \ldots, E_{t_c}(1), \ldots, E_{t_n}(0)),$$

where $s_1, \ldots, s_m$ and $t_1, \ldots, t_n$ are random numbers chosen by the SU for Paillier encryption.

Regarding Paillier encryption scheme (Section 3.2), the vectors are in fact equal to:

$$\bar{r}_{enc} = (s_1, \ldots, s_r g, \ldots, s_m) \bmod N^2,$$

$$\bar{c}_{enc} = (t_1, \ldots, t_c g, , t_n) \bmod N^2.$$

Step 4: Upon reception of the CRM, the DB arranges the SAI for the cells within the QR of the requesting SU. We indicate this $n \times m$ matrix by SAI:

$$SAI = \begin{bmatrix} a_{11} & \cdots & a_{1m} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nm} \end{bmatrix}.$$

Step 5: The DB then performs an encryption domain matrix multiplication (Section 3.3) on SAI with $\bar{r}_{enc}$ and $\bar{c}_{enc}$. It is easy through some manipulation to show that:

$$\bar{r}_{enc} * SAI * \bar{c}_{enc}^{T} = E\left(a_{rc}\right) \overset{\text{def}}{=} a_{enc}^{SU}$$

which is the encrypted value of the SAI of SU's cell. Then a_{enc}^{SU} is sent to the SU.

Step 6: The SU decrypts a_{enc}^{SU} with its private key to obtain the SAI of its own cell. Then it picks up a channel, tunes on it and starts utilizing the channel.

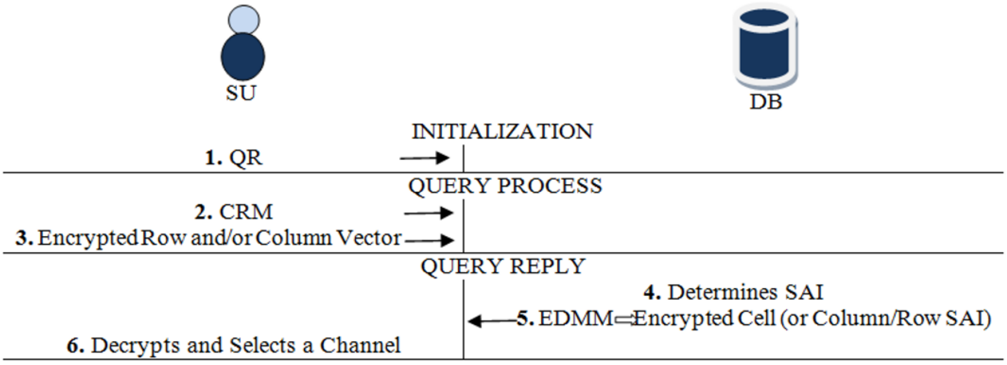

SU
DB
INITIALIZATION
1. QR
QUERY PROCESS
2. CRM
3. Encrypted Row and/or Column Vector
QUERY REPLY
4. Determines SAI
5. EDMM=Encrypted Cell (or Column/Row SAI)
6. Decrypts and Selects a Channel

Figure 1. PPQP protocol structure

5.3 Efficiency Improvement

In order to improve PPQP to make it more efficient both computationally and communicationally, small changes may be applied.

According to Section 5.2, in PPQP the user sends both its row and column vectors (in the encrypted form) to the DB, in order to access the SAI of the cell in which it is located. As an alternative, the SU may receive the SAI of the whole row or the whole column and select its own cell. While still being much more efficient than downloading the SAI matrix of the whole QR, this approach reduces the total communicational and computational cost significantly. We will discuss the cost benefit in more detail in Section 7.

In the improved version of the protocol, Steps 1, 2 and 4 remain the same as in Section 5.2. Steps 3, 5 and 6 should be modified as follows. Here we assume that SU is willing to decrypt and send its row vector. The case in which the column vector is intended would be similar.

Step 1 (unchanged): At the beginning of the whole protocol, each SU decides on a QR. A QR is an $n \times m$ rectangle, where n and m are upper-limited by the DB. Every cell in the QR is indicated by a (row,col) pair, where $0 \leq row < n$ and $0 \leq col < m$ are the row and column number, respectively. The QR is arbitrarily chosen around SU's real location, $loc_{SU} = [x_{SU}, y_{SU}]$. We will show the corresponding index of SU's cell within the QR by (r,c). The coordinates of (0,0) cell (the origin cell), $loc_o = [x_o, y_o]$, along with n and m uniquely describe a QR. This triple is reported to the DB: $QR := \langle [x_o, y_o], n, m \rangle$..

The QR is encrypted before reporting to the DB. For this purpose, SU and the DB may utilize a lightweight stream cipher. Therefore they should share a secret key in advance. Every SU also chooses a pair of public and private keys, K_{pub} and K_{pri}, for the Paillier cryptosystem, where $K_{pub} = (N, g)$ and $K_{pri} = (\lambda, \mu)$.

Step 2 (unchanged): When an SU needs a channel, it sends a CRM to the DB. The CRM contains user ID, which has been assigned by the DB through SUs initial registration to the network. Other parameters, such as antenna height and other characteristics may also be necessary according to the ruleset that governs the network.

Step 3m. The SU applies Paillier cryptosystem to encrypt its binary row vector element-wise, to achieve:

$$\begin{aligned}\bar{r}_{enc} &= (E_{s_1}(0), \dots, E_{s_r}(1), \dots, E_{s_m}(0)) \\ &= (s_1, \dots, s_r g, \dots, s_m) \bmod N^2.\end{aligned}$$

Then it sends $\bar{r}_{enc}$ to the DB.

Step 4 (unchanged): Upon reception of the CRM, DB arranges the SAI for the cells within the QR of the requesting SU. We indicate this $n \times m$ matrix by SAI:

$$SAI = \begin{bmatrix} a_{11} & \cdots & a_{1m} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nm} \end{bmatrix}.$$

Step 5m. The DB performs an encryption domain matrix multiplication (Section 3.3) on SAI with $\bar{r}_{enc}$, to achieve $\bar{r}_{enc} * SAI = (E(a_{r1}), \dots, E(a_{rc}), \dots, E(a_{rm})) \stackrel{def}{=} \bar{a}^{\bar{r}}_{enc}$, which is the ciphertext of $(r)^{th}$ row of SAI matrix of the QR, SAI. This vector is sent back to the SU.

Step 6m. SU decrypts only the $(c)^{th}$ element of $\bar{a}^{\bar{r}}_{enc}$ using its private key, to obtain the SAI of its own cell. Then it decides a channel to pick up, tunes on it and starts utilizing the channel. We will discuss the benefits of the modified version in more details in Section 8.

6 Privacy Analysis

This section models the privacy requirements through a game between adversary A and challenger C, to examine the security properties of PPQP protocol. Adversary A could be the DB, a PU or an external attacker, as indicated in Section 4.2. Then, the probabilistic advantage of adversary A in winning the game is calculated. The goal is to show that even if the adversary has some a priori information regarding the SU's location, PPQP can satisfy the privacy requirement described in Section 4.2.

The game can be generally described as follows.

Setup: The adversary A chooses two distinct random locations (cells), like loc^0 and loc^1, among the M cells that the DB covers.

Algorithm execution: The challenger C chooses a random bit $a \in \{0,1\}$. It selects loc^a as the location of the SU in the protocol. C then executes the protocol accordingly.

Challenge: C provides A with protocol transcript ($\bar{r}_{enc}$ and/or $\bar{c}_{enc}$ and a^{SU}_{enc} or $\bar{a}^{\bar{r}}_{\text{enc}}$) and asks A about a.

Guess: A guesses a bit $b \in \{0,1\}$ according to the information it has received and sends it back to C. A wins the game if $b = a$; otherwise, A loses.

We measure the probabilistic advantage for adversary A in this game, when it correctly guesses bit a in the proposed protocol. This is referred to as user location advantage of adversary A and will be denoted by $Adv(A) = |\Pr(b = a) - \frac{1}{2}|$ (as in [33] and [34]). To measure this advantage we consider two cases: when $loca'$ is located inside the QR and when it is not. By a' we mean the complement of bit a. In other words, $loc^{a'}$ is the location which has not been involved in the protocol.

(1) $loc^{a'}$ is located inside QR: If the adversary is not the DB, it does not have any information about QR. Because QR is reported to the DB through an encrypted message. Therefore, it must randomly guess the value of a. If the adversary is the DB, although it knows QR, since this area has been arbitrarily chosen by the SU, the DB still cannot find out anything about a. Moreover, the Paillier encryption has indistinguishable ciphertexts under chosen-plaintext attack. This means that for any probabilistic polynomial-time adversary A, given an encryption of a message randomly chosen between two known messages, the success probability of A to identify the chosen message is negligibly better than that of randomly guessing. Hence, in this case, the success probability of A to detect a having ($\bar{r}_{enc}$ and/or $\bar{c}_{enc}$ and a^{SU}_{enc} (or $\bar{a}^{\bar{r}}_{\text{enc}}$) is negligibly better than that of randomly guessing. Therefore, we can say that for any adversary, $\Pr(b = a) = \frac{1}{2} + \varepsilon$ where ε is a small enough real number.

(2) $loc^{a'}$ is not located inside QR: Depending on whether the adversary is the DB or not (it can be a PU or any external attacker), the result is different. When the DB plays the role of the adversary, since it knows QR, it will definitely know the value of bit a, i.e. $\Pr(b = a) = 1$. However, if another entity plays the role of A, since it does not have any information about QR, it must guess the value of a randomly, i.e. $\Pr(b = a) = \frac{1}{2} + \varepsilon$, where ε is a small enough real number.

Before computing the probabilistic advantage of adversary A in this game, we should identify the occurrence probability of each of the above-cited cases. For case 1 we have $\Pr\left(\text{loc}^{a'} \in QR\right) = \frac{nm}{M}$ where n and m are the dimensions of QR and M is the total number of cells in DB's coverage area.

For case 2 we have $\Pr\left(\text{loc}^{a'} \notin QR\right) = 1 - \frac{nm}{M}$.

Now, we can derive the following probabilistic advantage for adversary A, according to whether it is the DB or not:

- Adversary A is the DB

$$
\begin{aligned}
&Adv(\text{DB}) = |\Pr(b = a) - \frac{1}{2}| \\
&= |\Pr\left(loc^{a'} \in QR\right) \Pr\left(b = a \mid loc^{a'} \in QR\right) \\
&\quad + \Pr\left(loc^{a'} \notin QR\right) \Pr\left(b = a \mid loc^{a'} \notin QR\right) - \frac{1}{2}| \\
&= \left|\frac{nm}{M} \times \left(\frac{1}{2} + \varepsilon\right) + \left(1 - \frac{nm}{M}\right) \times 1 - \frac{1}{2}\right| \\
&= \frac{1}{2}\left(1 - \frac{nm}{M}\right) + \frac{nm}{M}\varepsilon.
\end{aligned}
$$

According to the above equation, by increasing the number of cells in the QR, the information obtained by the DB about SU's location can be decreased. (Note that ε is a negligibly small number). Therefore, the larger the number of cells in the QR, the less the advantage gained by an untrusted DB. However, it would result in more complexity.

- Adversary A is not the DB

$$
\begin{aligned}
&Adv(\text{others˙than˙DB}) = |\Pr(b = a) - \frac{1}{2}| \\
&= |\Pr\left(loc^{a'} \in QR\right) \Pr\left(b = a \mid loc^{a'} \in QR\right) \\
&\quad + \Pr\left(loc^{a'} \notin QR\right) \Pr\left(b = a \mid loc^{a'} \notin QR\right) - \frac{1}{2}| \\
&= \left|\frac{nm}{M} \times \left(\frac{1}{2} + \varepsilon\right) + \left(1 - \frac{nm}{M}\right) \times \left(\frac{1}{2} + \varepsilon\right) - \frac{1}{2}\right| \\
&= \varepsilon.
\end{aligned}
$$

Therefore, the information any adversary other than the DB could obtain about SU's location is negligibly small.

7 Complexity Analysis

7.1 Computational Complexity

To evaluate the computational overhead of our protocol, we consider the operations performed by SU and the DB in one round of the protocol. Hereafter, we indicate the basic form of the protocol by Method I and call the efficiency-improved version Method II.

On SU side: $n+m-2$ (in Method I) or $n-1$ (or $m-1$) (in Method II) Paillier encryption processes on "0" plaintext and two (in Method I) or one (in Method II) encryption process(s) on "1" plaintext should be performed. This is while each encryption process requires one modular exponentiation and one modular multiplication. Although this is true in general case, considering the specific values of the plaintexts, each encryption process reduces to only two multiplications in Method I and one in Method II.

Furthermore, in the last step of the protocol, SU performs one decryption process (in both Methods), consisting of one modular exponentiation, one modular multiplication and one modular addition.

The computational complexity on SU side is summarized in Table 1 and Table 2 for Methods I and II, respectively.

Table 1. Computational cost of method I

	SU	DB
Modular Exponentiation	1	$nm+m$
Modular Multiplication	3	$nm-1$
Modular Addition	1	–

On the DB side: The computational job of the DB takes place in the 4^{th} step of the protocol. In Step 4 of Method I, the DB first performs the EDMM process on the SAI matrix with SU's encrypted column vector. Then it executes another EDMM on the resulted vector with SU's encrypted row vector. The fo4rmer requires a total of nm modular exponentiations plus $m(n-1)$ modular multiplications. The latter takes m modular exponentiations, plus $m-1$ modular multiplications. Resulting in a total of $nm+m$ modular exponentiations, plus $nm-1$ modular multiplications (see Table 1).

In Method II, only one EDMM is to be performed, reducing DB's computational task to nm modular exponentiations and $nm-n$ (or $nm-m$) modular multiplications (see Table 2).

Table 2. Computational cost of method II

	SU	DB
Modular Exponentiation	1	nm
Modular Multiplication	2	$nm-n$
Modular Addition	1	–

7.2 Communicational Complexity

In Step 3 of Method I, the SU should send the encrypted values of its row and column vectors to the DB, which contain n and m elements, respectively. Assume that a log p-bit cryptosystem is being utilized. This results in a total of $(n+m)\log\ p$ bits for the user. On the other hand in Step 5, during Query Reply Process, the DB sends back a single log p-bit message to the SU. The total communicational overhead of Method I is thus $(n+m+1)\log\ p$ bits.

The communicational overhead in the second method seems to be much more fairly distributed between the two sides. While the SU sends an n(or m)-element vector in Step 3, the DB replies with an m(or n)-element vector in Step 5. The total communication complexity for Method II is therefore $(n+m)\log\ p$ bits.

The communicational cost for PPQP is summarized in Table 3.

Table 3. Communicational cost in PPQP

	SU	DB	Total
Method I	$(n+m)\log\ p$	$\log\ p$	$(n+m+1)\log\ p$
Method II	$n\log\ p$	$m\log\ p$	$(n+m)\log\ p$

7.3 Comparison with Other Works

We will compare PPQP's communicational cost with similar previous approaches. Scaled for providing privacy among whole coverage area, communicational complexities are shown in Table 4. Here, M indicates number of cells under coverage of the DB and log p is the output length of the cryptosystem. Other parameters will be introduced in each case. We put $n=\sqrt{M}$ in PPQP's complexity phrase, $2n\log\ p$, to scale the QR to the whole coverage area.

Table 4. Communicational cost comparison

Method	Communicational Complexity
K-Spectrum Query [13]	$M\log\ (\lvert SAI\rvert)$
LPDB [26]	$query + \varrho.s.M.\frac{\log\ (\frac{1}{\varepsilon})+2}{\alpha}$
PriSpectrum [14]	$(2\sqrt{M}+3)\log\ p$
PPQP	$2\sqrt{M}\log\ p$

K-Spectrum Query, the method represented in [13], like ours, takes advantage of the concept of cloaking region. SUs choose square query regions and report it to the DB. In response to a CRM, the DB sends the SAI of all cells within the query region to the requesting SU. This, of course, occupies a great portion of the available bandwidth and in turn reduces network's throughput significantly. The communicational complexity of K-Spectrum Query is given as $M\log(|SAI|)$, where $\log(|SAI|)$ is the bit-length of the SAI of each cell. It can be easily seen that if the following inequality is satisfied, PPQP's complexity is less than that of K-Spectrum Query:

$$\log(|SAI|) > \frac{\log p}{\sqrt{M}}.$$

The worst case happens when $\log p$ has its maximum and $\sqrt{M}$ its minimum values. A bit-length of 2048 bits usually provides satisfactory level of security and on the other hand, no longer bit-lengths are intended in order to avoid very large communicational cost. M may take different values. For instance, in simulations presented in [26] M is taken of order 10^5. In [14] authors have assumed values around 10^4 for M. Putting $\log p = 2048$ and $M = 10^4$ we will have $\frac{\log p}{\sqrt{M}} = 20.48$. This is while 21 bits is barely enough to represent the time stamp (t_i). So, the SAI length of every cell is of course more than that. (It should be reminded that the SAI of every cell can be represented in its simplest form as (ch_i, t_i, P_i) triples.) Therefore, it is obvious that the communicational complexity of PPQP is much less than that of K-Spectrum Query.

In PriSpectrum method [14], the SU uses some blinding factors (like some randomly chosen numbers) to hide its horizontal and vertical coordinates from the DB. The user can later use these factors (or their inverse, for example) to retrieve the information of his interest. This method is in fact based on the PIR introduced in [15]. The communicational complexity of PriSpectrum is given by $(2\sqrt{M}+3)\log p$. Expanding the QR to the whole coverage area in PPQP, the communicational cost would be $2\sqrt{M}\log p$, which is less than that of PriSpectrum. The notable point here is that the communicational complexity of PriSpectrum is exactly the mentioned value (due to the nature of the method), but in PPQP one can (and usually does) choose a smaller QR to reduce the complexity, $2n\log p$, where $n < \sqrt{M}$. This capability is very useful specially when the network is congested or when small bandwidth is available.

LPDB method [26] is based on Cuckoo filter introduced in [27]. Using this filter, a different representation of DB's information can be provided. The SU arranges a query and checks if an available channel exists in its location. As can be seen in Table 4, the communicational complexity phrase for LPDB contains several parameters. Some are related to the Cuckoo filter. (ϱ is the number of DB entries which contain available channels; s the number of all TV band channels; ε the false positive rate of Cuckoo filter; α the load factor of Cuckoo filter; and query is the bit-length of user's query.) According to graphs presented in [26], LPDB's communicational cost in no information leakage mode is larger than that of PPQP, even in its best case.

In the light of the above discussions, it can be seen that PPQP has reduced the communicational cost compared to previous similar methods and has the least cost among them.

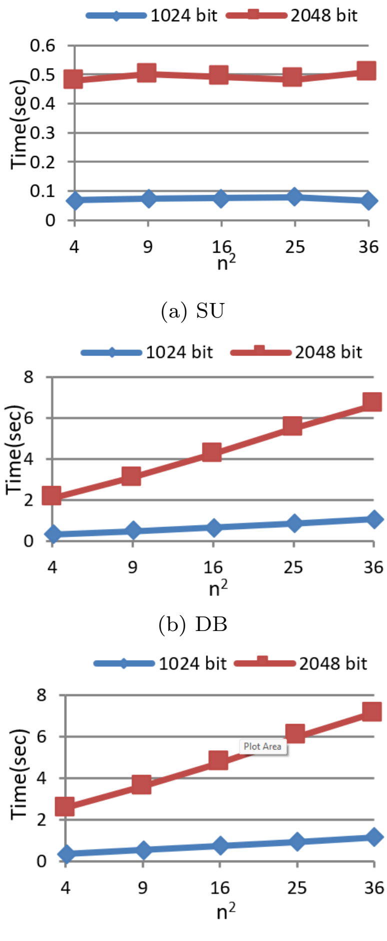
8 Simulation Results

In this section, we present and analyze the simulation results of PPQP protocol. To this end, we implement the protocol for both entities (SU and DB) running on a Microsoft PC with a dual-core 2GHz CPU, 2GB RAM and a 64-bit Windows7 Ultimate OS, using Java programming framework. We consider the performance for key lengths of 1024 and 2048 bits of the Paillier cryptosystem. We measure and compare the average computation latency in one round of PPQP protocol for SU and DB as well as the aggregated latency, for different values of QR size. For ease of display, we have considered square QRs, where both dimensions (n and m) are the same. We simulate both methods and also compare them with each other.

Computational delay for Method I, Method II and their comparison are shown in Figure 2, Figure 3 and Figure 4, respectively. As can be seen in all figures, the execution time for SU is independent from QR dimension, n. This is because in the forward direction (Step 3), SU just multiplies g with s twice (in Method I, Figure 2a) and once (in Method II, Figure 3a) for Paillier encryption. In the backward direction (Step 6), it does the same once, for Paillier decryption. Note that only two (in Method I) and one (in Method II) of the vectors' elements are non-zero.

However, DB should perform the EDMM operation in both methods. This process depends on the size of vectors and the SAI matrix dimension. In other words, DB's execution time increases as the QR size, n, increases. This is true about both methods and can be observed in Figure 2b and Figure 3b. The total computational latency for Methods I and II, which follows the same trend as the DB's, is shown in Figure 2c and Figure 3c.

Figure 4 presents a visual comparison between execution latency of Methods I and II for 1024-bit Paillier. As was discussed, SU's computational delay is independent from n, thus the method would not affect it (Figure 4a). However, Figure 4b shows a significant decrease in computation time for the DB in Method II compared with the first method. This reduction is due to fewer EDMM operations in Method



(a) SU

(b) DB

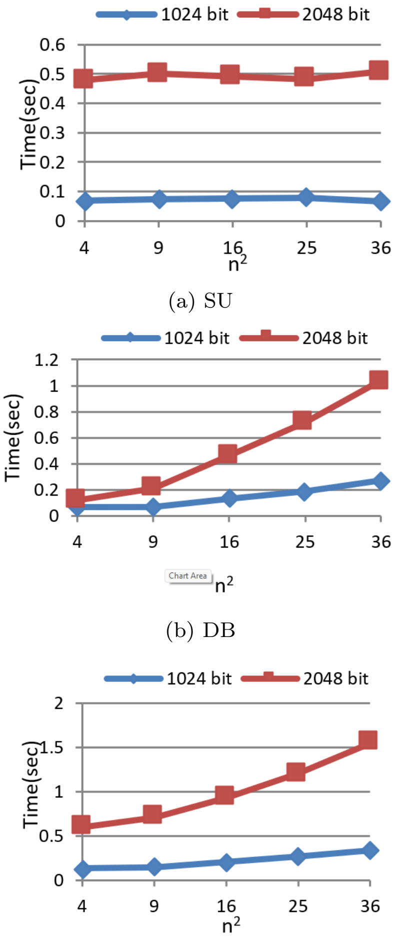
(c) Total runtime

Figure 2. Performance measurements for method I

II. Since SU's execution time is constant, the total protocol latency follows DB's trend, hence experiencing significant reduction in Method II (Figure 4c). It can be concluded that Method II is much more efficient in terms of execution time. Especially in cases where stricter privacy is aimed, Method II would be preferable. The difference between the two methods becomes more significant as the QR size, n, increases.

9 Conclusion

In this paper, we introduced a cryptography-based protocol (PPQP) for spectrum query in database-driven cognitive radio networks, which preserves location privacy of secondary users. This method takes advantage of homomorphic properties of some well-known cryptosystems. We examined PPQP's capability for preserving users' location privacy and showed that it does the duty well against different adversaries. Our protocol was also observed to have relatively low communicational cost.



(a) SU

(b) DB

(c) Total runtime

Figure 3. Performance measurements for method II

Acknowledgment

This work was partially supported by Iran-NSF Grant No. 92-32575.

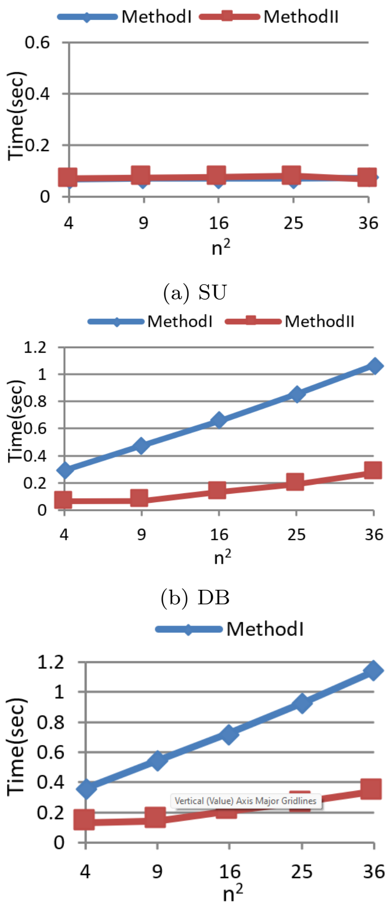
References

[1] ECC. Report 159, technical and operational requirement for the possible operation of cognitive radio system in the white space of the frequency band 470-790 mhz. 2011.

[2] FCC. Third order and memorandum opinion and order, in the matter of unlicensed operation in the TV broadcast bands, additional spectrum for unlicensed devices below 900 mhz and in the 3 ghz band. 2012.

[3] V. Chen (Ed.), S. Das, L. Zhu, J. Malyar, and P. McCann. RFC 7545, Protocol to Access White-Space (PAWS) databases. *DOI 10.17487/RFC7545, Available: http://www.rfc-editor.org/info/rfc7545¿*, 2015.

[4] R. Shokri, G. Theodorakopoulos, J. Y. Le Boudec, and J. P. Hubaux. Quantifiying location privacy. In *IEEE Symposium on Security*



(a) SU

(b) DB

(c) Total runtime

Figure 4. Runtime comparison between method I and method II

and Privacy, pages –, 2011.

[5] M. Grissa, B. Hamdaoui, and A. A. Yavuz. Location privacy in cognitive radio networks: a survey. *IEEE Communications Surveys and Tutorials*, 19:1726–1760, 2017.

[6] R. L. Rivest, L. Adleman, and M. L. Dertouzos. On data banks and privacy homomorphisms. *Foundations of Secure Computation*, 19:169–179, 1978.

[7] I. T. Lien, Y. H. Lin, J. R. Shieh, and J. L. Wu. A novel privacy preserving location-based service protocol with secret circular shift for K-NN search. *IEEE Transactions on Information Forensics and Security*, 8:863–873, 2013.

[8] J. Xu, H. Yu, C. Xu, and N. Zheng. A dynamic spatial cloaking algorithm for location privacy. In *IET International Conference on Information Science and Control Engineering*, pages –, 2012.

[9] L. Sweeney. Achieving k-anonymity privacy protection using generalization and suppression. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10:571–588, 2002.

[10] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *17th Int. Conf. on Theory Application of Cryptographic Techniques*, pages 223–238, 1999.

[11] B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan. Private information retrieval. *Journal of the ACM (JACM)*, 45:–, 1998.

[12] H. Li, Q. Pei, and W. Zhang. Location privacy-preserving channel allocation scheme in cognitive radio networks. *International Journal of Distributed Sensor Networks*, 12:–, 2016.

[13] L. Zhang, C. Fang, Y. Li, H. Zhu, and M. Dong. Optimal strategies for defending location inference attack in database-driven crns. In *IEEE International Conference on Communications (ICC)*, pages 7640–7645, 2015.

[14] Z. Gao, H. Zhu, Y. Liu, M. Li, and Z. Cao. Location privacy in database-driven cognitive radio networks: Attacks and countermeasures. In *IEEE Conference on Computer Communications (INFOCOM'13)*, pages 2751–2759, 2013.

[15] J. Trostle and A. Parrish. Efficient computationally private information retrieval from anonymity or trapdoor groups. In *the 13th International Conference on Information security (ISC'10)*, pages 114–128, 2010.

[16] E. Troja and S. Bakiras. Leveraging p2p interactions for efficient location privacy in database-driven dynamic spectrum access. In *the 22nd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages –, 2014.

[17] C. Gentry and Z. Ramzan. Single-database private information retrieval with constant communication rate. In *International Colloquium on Automata, Languages and Programming (ICALP'05)*, pages 803–815, 2005.

[18] E. Troja and S. Bakiras. On packing r-trees. In *24th IEEE International Conference on Computer Communication and Networks (ICCCN)*, pages 1–8, 2015.

[19] I. Kamel and C. Faloutsos. On packing r-trees,. In *the second international conference on Information and knowledge management (ACM)*, pages 490–499, 1993.

[20] Z. Salami, M. Ahmadian-Attari, H. Jannati, and M. R. Aref. A location privacy-preserving method for spectrum sharing in database-driven cognitive radio networks. *Wireless Personal Communications*, 95:3687–3711, 2017.

[21] Z. Zhang, H. Zhang, S. He, and P. Cheng. Achieving bilateral utility maximization and location privacy preservation in database-driven cognitive radio networks. In *IEEE 12th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, pages 181–189, 2015.

[22] M. E. Andres, N. E. Bordenabe,

K. Chatzikokolakis, and C. Palamidessi. Geo-indistinguishability: Differential privacy for location-based systems. In *ACM SIGSAC conference on Computer and communications security*, pages 901–914, 2013.

[23] Z. Chen, L. Chen, and H. Zhong. Towards secure and verifiable database-driven spectrum sharing. In *47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pages –, 2017.

[24] A. C. C. Yao. How to generate and exchange secrets. In *16th Annual Symposium on Foundations of Computer Science (FOCS)*, pages –, 1975.

[25] Z. Chen, L. Huang, and L. Chen. ITSEC: An information-theoretically secure framework for truthful spectrum auctions. In *International Conference on Computer Communications (INFOCOM)*, pages 2065–2073, 2015.

[26] M. Grissa, A. A. Yavuz, and B. Hamdaoui. Cuckoo filter-based location-privacy preservation in database-driven cognitive radio networks. In *IEEE World Symposium on Computer Networks and Information Security (WSCNIS)*, pages 1–7, 2015.

[27] B. Fan, D. G. Andersen, M. Kaminsky, and M. D. Mitzenmacher. Cuckoo filter: Practically better than bloom. In *10th ACM International on Conference on emerging Networking Experiments and Technologies*, pages 75–88, 2014.

[28] B. Bahrak, S. Bhattarai, A. Ullah, J. Park, J. Reed, and D. Gurney. Protecting the primary users' operational privacy in spectrum sharing. In *IEEE International Symposium on Dynamic Spectrum Access Networks (DYSPAN'14)*, pages 236–247, 2014.

[29] M. Clark and K. Psounis. Can the privacy of primary networks in shared spectrum be protected? In *35th IEEE International Conference on Computer Communications (INFOCOM)*, pages 1–9, 2016.

[30] A. B. Mosbah, T. A. Hall, M. Souryal, and H. Afifi. An analytical model for inference attacks on the incumbent's frequency in spectrum sharing. In *IEEE International Symposium on Dynamic Spectrum Access Networks (DYSPAN)*, pages –, 2017.

[31] R. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21:120–126, 1978.

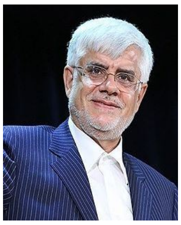
[32] C. Gentry. Fully homomorphic encryption using ideal lattices. In *the 41st ACM Symposium on Theory of Computing (STOC)*, pages 169–178, 2009.

[33] I. Bilogrevic, M. Jadliwala, V. Joneja, K. Kalkan, J. P. Hubaux, and I. Aad. Privacy-preserving optimal meeting location determination on mobile devices. *IEEE Trans. on Information Forensics and Security*, 9:1141–1156, 2014.

[34] Y. Ling, S. Ma, Q. Huang, and X. Li. A general two-server framework for ciphertext-checkable encryption against offline message recovery attack. In *Cloud Computing and Security (ICCCS 2018)*, pages 370–382, 2018.

Zeinab Salami received the B.Sc. degree in Electrical Engineering and the M.Sc. degree in Communications Engineering, both from the Department of Electrical Engineering, Sharif University of Technology, Iran, in 2007 and 2010, respectively. She is currently working toward the Ph.D. degree in Communications Engineering at the Department of Electrical Engineering, K. N. Toosi University of Technology, Iran, and is a member of Information Systems and Security Lab (ISSL) at Sharif University of Technology. Her research interests include network security, cognitive radio networks, and cryptography.

Mahmoud Ahmadian-Attari is a Professor at the Department of Electrical Engineering, K. N. Toosi University of Technology, Iran. He received the combined B.Sc. and M.Sc. degree in Electrical Engineering from University of Tehran, Iran, in 1977. He received the Ph.D. degree in Digital Communication Systems from University of Manchester in 1997. His research interests include coding theory and cryptography.



Mohammad Reza Aref received his B.Sc. in 1975 from University of Tehran, his M.Sc. and Ph.D. in 1976 and 1980, respectively, from Stanford University, all in Electrical Engineering. He was a faculty member of Isfahan University of Technology from 1982 to 1995. He has been a Professor of Electrical Engineering at Sharif University of Technology since 1995. His current research interests include communication theory, information theory, and cryptography.

Hoda Jannati is providing consultancy services in the field of information security to MCI R&D center. She was a post-doctoral researcher in the School of Computer Science at Institute for Research in Fundamental Sciences (IPM) from 2014 to

2018, Iran. She received the B.Sc. degree in Electrical Engineering in 2006, the M.Sc. degree in Cryptography Communications in 2008, and the Ph.D. degree in Communications Systems in 2014. Her main research interests include security in wireless communication systems specially in RFID and sensor network systems, localization algorithms and location privacy.