This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSEN.2016.2540002, IEEE Sensors Journal

A Lottery SMC Protocol for the Selection Function in Software Defined Wireless Sensor Networks

Yi SUN*[1,2,3,4], Zhaowen LIN[1,2,3], Yan MA[1],

***Abstract*—In this paper, we focus on selecting targeted number from private sensory data in Software Defined Wireless Sensor Networks and propose a lottery secure multiparty computation protocol for the selection function based on layered homomorphic encryption and the equivalent transformation of two arrays. We also prove that it is secure via real/ideal simulation paradigm, no matter the web server is honest or semi-honest in the presented security model.**

***Index Terms*—Information Security, Distributed Computation, Secure Multiparty Computation, Software Defined Wireless Sensor Network, Homomorphic Encryption.**

I. Introduction

Secure multiparty computation (SMC) [1] enables mutually distrusted parties to jointly compute a function by their private inputs while keeping them disclosed in the whole process. It aims to solve the cooperative computation problem for mutually suspicious parties so that each entity can securely cooperate with their friends or strangers, or even the competitors without worrying about revealing privacy. Therefore, it has become a hot research topic as a powerful tool to protect the security of private information in big data [2, 3].

However, its application in practice seems limited. One of the main reasons is that the computation model in web scenario is not suited to the type of the communication pattern for the existing SMC protocols. In web setting, each party independently connects to the server, interacts with it, leaves arbitrarily and not always keeps online all the time. All of these characteristics bring forward higher requirements to SMC protocols. At the same time, Software Defined Wireless Sensor Network (SDWSN) is a new paradigm that offers significant promise to ubiquitous sensing and sensory data service. But it has not benefited the field of secure multiparty computation yet due to the limits of existing protocols. Therefore, how to construct a practical SMC protocol that is appropriate to be used to protect private sensory data in software defined wireless sensor networks is a leading and promising work.

In this paper, we focus on how to apply SMC in web scenario to securely process the sensory data collecting from SDWSNs so that the control intelligence is moved out and implemented in a logically centralized controller. Specifically, firstly, we present an adapted model which allows the clients on the web to connect to the web server in wireless sensor manner and leave at any time, and then construct a SMC protocol for the selection function based on layered homomorphic encryption and the equivalent transformation of two arrays. What's more, we also give a strict proof for our result. This work is very important and meaningful to combine SMC protocols with web applications in wireless sensor networks. The blueprint that users can conduct desired privacy-preserving computations of the sensory data via internet without communicating with each other is not only a dream as long as our research matures. In brief, our contributions can be summarized as follows.

A. Our Contributions

- We present a security model for the SMC protocol in web scenario, which allows the clients connected by the wireless sensor networks to the web server in an ad-hoc manner and can arbitrarily leave. It is different from the standard setting of SMC since the communication pattern is different and the web server participates.
- We construct a lottery SMC protocol for the selection function in SDWSNs based on layered homomorphic (onion-type) encryption and the equivalent transformation of two arrays.
  - It can pick out the agreed *h*-th greatest private number which is determined according to the size instead of the one that some client prefers from the *m* private numbers.
  - It is "*lottery secure*", that is, only the owner of the selected number knows the selected value while the server is only aware of the identity of the selected client, the rest of the clients just learn that they are not selected.
  - There is no interaction between any two clients any longer.
- As a by-product, we obtain a non-interactive SMC protocol for the sequencing problem in web scenario by taking the first phase of the selection function protocol as an independent part. In this way, clients on the web can obtain the sequence of their secret numbers by just doing some local computations without interacting with other clients of the wireless sensor networks.

B. Related Work

Until recently, the research on secure multiparty computation in web scenario is not enough. A formal study on this

[1]Network and Information Center, Institute of Network Technology, Beijing University of Posts and Communications
[2]Science and Technology on Information Transmission and Dissemination in Communication Networks Laboratory
[3]National Engineering Laboratory for Mobile Network Security ($No.$[2013]2685)
[4]Institute of Sensing Technology and Business, Beijing University of Posts and Communications
*Corresponding author: Yi SUN
E_mail: sybupt@bupt.edu.cn
Manuscript received Dec. 30, 2015; revised Jan. 27, 2016.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSEN.2016.2540002, IEEE Sensors Journal

topic is the work by S. Halevi, Y. Lindell and B. Pinkas in crypto-2011 [4]. Herein, they have proposed some solutions to a number of functions including the selection function, which is also interesting to us. They consider the setting where *m* parties, each separately holding a private number, wish to find out the *h*-th one pointed by the first party without disclosing the other *m*-1 numbers. They utilize the techniques of layered additively homomorphic encryption and layered re-randomization to solve this problem. After obtaining the ciphertext from the server, all parties cooperate sequentially, the *i*-th party decrypts, re-randomizes and transmits *i*-1 messages to the (*i*+1)-th party. The server finally receives the selected value by decrypting some ciphertexts.

In this paper, we expect a more predominant protocol for the selection function in the following two aspects.

- Firstly, we expect that the protocol can find out the *h*-th greatest number agreed by all clients instead of the one simply determined by some party alone.
- Secondly, we expect that there are few interactive rounds between mutually distrusted users (the fewer, the better).

Following this idea, we discuss the case that the selection function is to find out the *h*-th greatest number agreed by all clients instead of simply determined by some party alone. It makes the selection more meaningful and practical. We realize this computation by two phases called comparing and selecting respectively. Firstly, in the comparing phase, by using layered homomorphic encryption and equivalent transformation of the private numbers, we can securely compare them without interactions of the clients, which can be also recognized as an independent non-interactive secure sequencing protocol as a by-product. Based on this, we can select out the *h*-th greatest number and guarantee that the result is "*lottery secure*" as we defined with only three rounds of interaction between each client and the web server, while with no interactions between any two clients (reducing/removing the interaction between the client and the server is our next work).

C. Organization

The rest of this paper is organized as follows. In section 2, we briefly give an overview of SMC, layered homomorphic encryption and the equivalent transformation of two arrays. In section 3, we present our results including the new model and the lottery secure SMC protocol for the selection function in software defined wireless sensor networks. In section 4, we analyze the proposed protocol in detail and give a strict proof based on real/ideal simulation paradigm. Finally, we summarize our work of this paper in the last section.

II. Preliminaries

A. Secure Multiparty Computation

SMC is dedicated to dealing with the problem of privacy-preserving cooperative computation among distrusted participants. It was initially introduced by Yao in 1982 by putting forward the famous Millionaire's problem. Afterwards, SMC has become a research focus due to its wide applications in various areas such as image detection [5, 6], data mining [7–9], electronic commerce [10], wireless sensor networks [11, 12], electronic voting [13], computation geometry [14], recommendation system [15, 16], and a mass of research results have been published one after another [17–21].

Generally speaking, SMC is a method to implement cooperative computation with all participants' private data, ensuring the correctness of the computation as well as not disclosing additional information except the necessary result. Assuming that there are *m* participants, $P_1, P_2, \ldots, P_m$, each of them has a private number, respectively $x_1, x_2, \ldots, x_m$. Then the aim of the m participants is to cooperate to compute the function $f(x_1, x_2, \ldots, x_m)$ without revealing $x_1, x_2, \ldots, x_m$.

Traditionally, we consider this problem by assuming that all parties interact simultaneously [17–21], which is not appropriate to the case in our wireless sensor networks scenario where the clients connect to the web server and then interact with it independently and leave freely. The computation model in web scenario is not suited to the type of the communication pattern for the existing SMC protocols. This is also the main reason why SMC protocols are not widely used in practice so far. In order to break this deadlock, we further study on the application of SMC protocols in wireless sensor networks.

A simple method in this case is to take the web server as a trusted party. However, one of the most important task of SMC and cryptography community is to transform systems that rely on trusted parties into the ones that do not need them. Therefore, it is meaningless to get the result by taking the web server as a trusted server. In this paper, we regard it as a special party P_{m+1} who can be either honest or dishonest and also can act differently from the other *m* clients. All related computations over the private numbers are conducted in the encrypted forms under *m*+1 keys of the *m* clients and the server.

B. Layered Homomorphic Encryption

In this subsection, we introduce a basic tool in SMC, homomorphic encryption and its variant, layered homomorphic encryption.

Allowing for security, clients usually would not like to directly transmit their private information over insecure channel. They expect other parties can perform computations in the encrypted forms of the data. In this way, they can encrypt their own private information and then transmit it to others without exposing the real data and finally decrypt the information sent back by others to get the targeted result when completing cooperative computation. To meet this demand, Rivest et al. [22] proposed homomorphic encryption in 1978. It makes feasible to operate on the ciphertexts of the private data to compute the function. This work sparked the research in this field. A lot of articles have been proposed and widely used in many applications since then [23–27].

However, traditional homomorphic encryption schemes are single-key in the sense that participants only can perform computations on inputs encrypted under the same key. The owner of the key can decrypt all ciphertexts directly. Therefore, this type of one-layered encryption is not secure enough. Currently, Adriana Lopez-Alt [26] presents the notion multi-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSEN.2016.2540002, IEEE Sensors Journal

key homomorphic encryption, that is, the layered homomorphic encryption, where *m* different public keys encrypt the input sequentially. She has constructed a SMC protocol which conducts related computations on the cloud. It proves that the layered homomorphic encryption is indeed a feasible method to protect private information. However, it is too complicated to decrypt the ciphertext if we peel it layer by layer like peeling an onion.

Following this idea, we make use of layered homomorphic encryption in the application of secure multiparty computation on the web to solve the problem of securely selecting the *h*-th greatest number from *m* private numbers. By encrypting the private numbers using layered homomorphic encryption and then comparing the pseudo numbers produced by equivalently transforming the private numbers, it is unnecessary to decrypt the ciphertext any longer. Herein, we denote $E_{pk}(x, r^{(i)})$ as the encryption of *x* using random number $r^{(i)}$ chosen by P_i under the public key pk. Then the *m*-layered homomorphic encryption by P_i, which is starting with the encryption of *x* under pk_1 using random number $r_1^{(i)}$ and re-encrypting under each $pk_j, j = 2, \ldots, m$, in turn, using random number $r_j^{(i)}$, is

$$E_{pk_{1,m}}(x, r_1^{(i)}, r_2^{(i)}, \ldots, r_m^{(i)}) = E_{pk_m}((\ldots E_{pk_1}(x, r_1^{(i)}), \ldots), r_m^{(i)}).$$

It satisfies the following property, $\forall x_1, x_2 \epsilon M$,

$$E_{pk_{1,m}}(x_1, \bar{r}^{(i)}) * E_{pk_{1,m}}(x_2, \bar{r}^{(j)}) = E_{pk_{1,m}}(x_1 \cdot x_2, \bar{r}^{(i)}, \bar{r}^{(j)}).$$

where $\bar{r}^{(i)} = (r_1^{(i)}, r_2^{(i)}, \ldots, r_m^{(i)})$, $\bar{r}^{(j)} = (r_1^{(j)}, r_2^{(j)}, \ldots, r_m^{(j)})$, $*$ and $\cdot$ denote the operators in the set of plaintexts and ciphertexts respectively.

In this paper, we utilize this technique among all clients $P_i, i = 1, 2, \ldots, m$, and the web server P_{m+1}. Each P_i encrypts its secret by the *m*+1 public keys and so no one can decrypt the ciphertext alone. Only by decrypting it one by one like peeling onions layer by layer, the adversary can finally get the secret. In this sense, our protocol can resist at most m-1 adversaries collusion.

C. Equivalent Transformation of Two Arrays

In this subsection, we simply give an equivalent transformation of two arrays as [28].

Theorem 1. Arrays $(X_1, X_2, \ldots, X_m)$ and $(x_1, x_2, \ldots, x_m)$ have the same sequence and X_i is called as the pseudo number of x_i, if $X_i = s_1 \cdot x_i + s_2 \cdot x_i^2 + \ldots + s_n \cdot x_i^n$, $s_i \geq 0$, $x_i \geq 0, i = 1, 2, \ldots, m$.

Proof.

Given $\forall S_i', S_j' \in (S_1', S_2', \ldots, S_n')$,

$S_i' = r_1' * S_i + r_2' * S_i^2 + \ldots + r_n' * S_i^n$,

$S_j' = r_1' * S_j + r_2' * S_j^2 + \ldots + r_n' * S_j^n$,

Then, $S_i' - S_j'$

$= (r_1' * S_i + r_2' * S_i^2 + \ldots + r_n' * S_i^n) - (r_1' * S_j + r_2' * S_j^2 + \ldots + r_n' * S_j^n)$

$= r_1' * (S_i - S_j) + r_2' * (S_i^2 - S_j^2) + \ldots + r_n' * (S_i^n - S_j^n)$

$= (S_i - S_j) \cdot [r_1' + r_2' * (S_i + S_j) + r_3' * (S_i^2 + S_i \cdot S_j + S_j^2) + \ldots + r_n' * (S_i^{n-1} + S_i^{n-2} \cdot S_j + S_i \cdot S_j^{n-2} + S_j^{n-1})]$

Let $Q = r_1' + r_2' * (S_i + S_j) + r_3' * (S_i^2 + S_i \cdot S_j + S_j^2) + \ldots + r_n' * (S_i^{n-1} + S_i^{n-2} \cdot S_j + S_i \cdot S_j^{n-2} + S_j^{n-1})$,

Thus, $S_i' - S_j' = (S_i - S_j) \cdot Q$,

Since $r_i' \geq 0, S_i \geq 0, i = 1, 2, \ldots, n$, we have $Q \geq 0$.

Therefore, $(S_i' - S_j') \cdot (S_i - S_j) = (S_i - S_j)^2 \cdot Q \geq 0$,

Thus, $\forall S_i', S_j' \in (S_1', S_2', \ldots, S_n')$, $S_i' - S_j'$ and $S_i - S_j$ have the same sign,

That is, $\forall S_i', S_j' \in (S_1', S_2', \ldots, S_n')$, S_i', S_j' and S_i, S_j have the same sequence,

Obviously, $(S_1, S_2, \ldots, S_n)$ and $(S_1', S_2', \ldots, S_n')$ have the same sequence.

From theorem 1, we know that the original m-array $(x_1, x_2, \ldots, x_m)$ has the same size relations with the new m-array $(X_1, X_2, \ldots, X_m)$, which is called as the pseudo array of $(x_1, x_2, \ldots, x_m)$. Hence, we can obtain the sequence of the original array by directly comparing the pseudo array in public instead.

In this paper, we make use of this useful theorem in the comparing phase to determine which number is the *h*-th greatest one. This phase can also be considered as a sequencing protocol independently to sequence the *m* private numbers by opening $(X_1, X_2, \ldots, X_m)$. In this way, *m* clients can conduct related computations locally and compare the size of the private numbers in the encrypted forms. As a result, it avoids the decryption process and further removes the complex interaction between clients.

III. Our Results

A. Model

O_i	output of P_i
1^n	input of P_{m+1}
n	secure parameter
α	common number
(pk_i, sk_i)	public-private keys of P_i

Assuming that there are *m* clients $P_1, P_2, \ldots, P_m$ and a web server P_{m+1}. The *m* clients privately share a common number α as well as respectively owning a private number $x_i, i = 1, 2, \ldots, m$, the private sensory data collected in the wireless sensor networks. They want to securely compute the selection function $f(x_1, x_2, \ldots, x_m, 1^n) = (O_1, O_2, \ldots, O_{m+1})$, which is to select the *h*-th greatest number from the *m* private inputs of clients, O_i is the output of $P_i, i = 1, 2, \ldots, m+1$, 1^n is the input of P_{m+1}, *n* is the secure parameter. The client can only communicate with P_{m+1} since there is no connection between any two clients. All our operations are staged in the PKI model, where each $P_i, i = 1, 2, \ldots, m+1$ knows the public keys of all other clients and the web server and the private key corresponding to its own public key. Herein, we denote the public-private keys of P_i as $(pk_i, sk_i), i = 1, 2, \ldots, m+1$.

With respect to the selection function $f(x_1, x_2, \ldots, x_m, 1^n) = (O_1, O_2, \ldots, O_{m+1})$, supposing the h-th greatest number is x_s, we give a stricter requirement compared with previous protocols by defining $f(x_1, x_2, \ldots, x_m, 1^n) = (x_1', x_2', \ldots, x_{s-1}', x_s, x_{s+1}', \ldots, x_m', P_s)$, that is, for the non-selected client $P_i, i = 1, 2, \ldots, m, i \neq s$, it obtains

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSEN.2016.2540002, IEEE Sensors Journal

$x'_i \neq x_i$ as its output while the selected client P_s gets its own private number x_s as its output, and the identity of the selected client P_s is the output of the server P_{m+1}. To be more specifically, we enable the clients judge whether he is selected or not by observing whether he can recover his own private number after executing the protocol. If P_i succeeds to recover his own private number, he is the selected client. On the contrary, if he cannot regain it, he is not selected. What's more, even the server will only know the identity of the selected client but not the private number. That is, P_s can still keep his private number disclosed from others including the server. This is appropriate to design a novel lottery game where the winner is to be decided by the lottery fan themselves before and the winner will be known only by the winner and the retailer. As we know, the lucky one who wins a grand lottery prize would not like others to know for personal safety.

Herein, we say that a selection function protocol is "*lottery secure*" if clients obtain the results by observing whether they can recover their private inputs and only the selected client and the server know the identity of the winner while other clients including the server obtain nothing about the private information. The SMC protocol which holds the property of "*lottery secure*" is called as "*Lottery Secure SMC Protocol*" vividly.

B. Security

Generally speaking, a SMC protocol is dubbed secure if it satisfies two requirements, correctness and security. To the correctness, it requests the protocol indeed computes the function f so that all clients get the targeted results defined by the function. The security is refered to that no participant can learn more from the description of the public function and the result of the global calculation than what he can learn from his own information. In the following phase, we will illustrate how to describe privacy by simulation.

Normally, we define security via real/ideal simulation paradigm following [29]. Loosely speaking, a multiparty protocol privately computes f if whatever a set (or a coalition) can obtain after participating in the protocol could be essentially obtained from the input and output of these very parties. This can be stated in detail by the following definition.

Definition 1: Let $f : (\{0,1\}^*)^m \rightarrow (\{0,1\}^*)^m$ be an m-input functionality, where $f_i(x_1, x_2, \ldots, x_m)$ denotes the i-th element of $f(x_1, x_2, \ldots, x_m)$. For a coalition with t adversaries $I = \{i_1, i_2, \ldots, i_t\} \subseteq [m] = \{1, 2, \ldots, m\}$, denote $f_I(x_1, x_2, \ldots, x_m) = \{f_{i_1}(x_1, x_2, \ldots, x_m), \ldots, f_{i_t}(x_1, x_2, \cdots, x_m)\}$. Let Π be an m-party protocol for computing f. The view of the i-th party during an execution of Π on $\bar{x} = (x_1, x_2, \ldots, x_m)$ is denoted as $view_i^{\Pi}(\bar{x})$. For $I = \{i_1, i_2, \ldots, i_t\}$, we have $view_I^{\Pi}(\bar{x}) = (I, view_{i_1}^{\Pi}(\bar{x}), view_{i_2}^{\Pi}(\bar{x}), \ldots, view_{i_t}^{\Pi}(\bar{x}))$. We say that Π privately computes f if there exists a probabilistic polynomial-time algorithm, denoted $\mathcal{S}$, such that for every $I \subseteq [m]$, it is impossible to distinguish $\{\mathcal{S}(I, (x_{i_1}, x_{i_2}, \ldots, x_{i_t}), f_I(\bar{x}))\}_{\bar{x} \in (\{0,1\}^*)^m}$ and $\{view_I^{\Pi}(\bar{x})\}_{\bar{x} \in (\{0,1\}^*)^m}$. That is, it holds that,

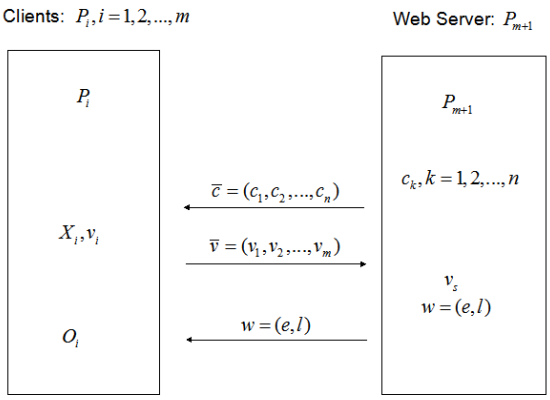

Fig. 1: Framework of Our Protocol

$$\{\mathcal{S}(I, (x_{i_1}, x_{i_2}, \ldots, x_{i_t}), f_I(\bar{x}))\}_{\bar{x} \in (\{0,1\}^*)^m} \cong \{view_I^{\Pi}(\bar{x})\}_{\bar{x} \in (\{0,1\}^*)^m}$$

C. Lottery SMC Protocol for the Selection Function in SD-WSNs

In this section, we construct a lottery secure multiparty protocol for the following selection function $f(x_1, x_2, \ldots, x_m, 1^n) = (x'_1, x'_2, \ldots, x'_{s-1}, x_s, x'_{s+1}, \ldots, x'_m, P_s)$ in wireless sensor networks for m clients $P_1, P_2, \ldots, P_m$ with inputs $x_1, x_2, \ldots, x_m$, the private sensory data collected in the wireless sensor networks, and a web server P_{m+1} with input 1^n, which is to find out the h-th private number without revealing it.

Let G be a group of prime order q with generator g. The public key of P_i is $pk_i = g^{\alpha_i}$, where α_i is the private key of P_i. The common information are the public keys of $P_i, i = 1, 2, \ldots, m+1$, and the targeted selective index h which is decided by all clients before. The one-layered homomorphic encryption can be described as $E_{pk_j}(x, r^{(i)}) = pk_j \cdot g^{r^{(i)}} \cdot x = g^{\alpha_j} \cdot g^{r^{(i)}} \cdot x = g^{\alpha_j + r^{(i)}} \cdot x$, which is the encryption of x using $r^{(i)}$ under the public key pk_j by P_i. Similarly, $E_{pk_{1,m+1}}(x, r_1^{(i)}, r_2^{(i)}, \ldots, r_{m+1}^{(i)}) = g^{\sum_{j=1}^{m+1}(\alpha_j + r_j^{(i)})} \cdot x$ denotes the (m+1)-layered homomorphic encryption starting with the encryption of x under pk_1,using $r_1^{(i)}$ and re-encrypting under each pk_j in turn, using $r_j^{(i)}$ by P_i, $j = 2, \ldots, m+1$. The framework of our protocol can be simplified in Figure 1.

The details of our protocol are as follows.

Firstly, in the comparing phase, the server P_{m+1} randomly chooses n numbers $(s'_1, s'_2, \ldots, s'_n)$ and m+1 numbers $\bar{r}^{(m+1)} = (r_1^{(m+1)}, r_2^{(m+1)}, \ldots, r_{m+1}^{(m+1)})$ and locally computes and publics $c_k = E_{pk_{1,m+1}}(s'_k, \bar{r}^{(m+1)}) = g^{\sum_{j=1}^{m+1}(\alpha_j + r_j^{(m+1)})} \cdot s'_k$, $k = 1, 2, \ldots, n$. Similarly, for $i = 1, 2, \ldots, m$, P_i randomly chooses $\bar{r}^{(i)} = (r_1^{(i)}, r_2^{(i)}, \ldots, r_{m+1}^{(i)})$ and computes $e_k^{(i)} = E_{pk_{1,m+1}}(x_i^k, \bar{r}^{(i)}) = g^{\sum_{j=1}^{m+1}(\alpha_j + r_j^{(i)})} \cdot x_i^k$, $k = 1, 2, \ldots, n$, and further obtains $X'_i = \sum_{k=1}^{n} c_k * e_k^{(i)} = g^{\sum_{j=1}^{m+1}(2\alpha_j + r_j^{(i)} + r_j^{(m+1)})} \cdot \sum_{k=1}^{n}(s'_k \cdot x_i^k)$ and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSEN.2016.2540002, IEEE Sensors Journal

$X_i = X_i'/g^{\sum_{j=1}^{m+1} r_j^{(i)}} = g^{\sum_{j=1}^{m+1}(2\alpha_j + r_j^{(m+1)})} \cdot \sum_{k=1}^{n}(s_k' \cdot x_i^k)$. Denote $s_k = g^{\sum_{j=1}^{m+1}(2\alpha_j + r_j^{(m+1)})} \cdot s_k'$, then we have $X_i = \sum_{k=1}^{n}(s_k \cdot x_i^k)$.

As described in section II, part C, the pseudo array $(X_1, X_2, \ldots, X_m)$ has the same sequence with the private array $(x_1, x_2, \ldots, x_m)$. Thus, we can get a by-product, a non-interactive sequencing protocol if the clients publicly compare the size of the pseudo numbers now. The advantage is obvious, clients can compare the secret numbers by just doing some local computations without interacting with others.

In the selecting phase, for $i = 1, 2, \ldots, m$, P_i transmits $v_i = (X_i \cdot g^{\alpha}, P_i, e_1^{(i)} \cdot g^{r^{(i)}})$ to the server P_{m+1}, where α is the common secret shared among the *m* clients, $r^{(i)}$ is a private number selected by P_i. And then P_{m+1} selects the targeted vector from $V = (v_1, v_2, \ldots, v_m)$ according to the size of $X_i \cdot g^{\alpha}$. Suppose that the *h*-th greatest element in $(x_1, x_2, \ldots, x_m)$ is x_s, then $v_s = (X_s \cdot g^{\alpha}, P_s, e_1^{(s)} \cdot g^{r^{(s)}})$ is the targeted vector. P_{m+1} chooses *m*+1 numbers $\bar{r}'^{(m+1)} = (r_1'^{(m+1)}, r_2'^{(m+1)}, \ldots, r_{m+1}'^{(m+1)})$ to compute $e_0 = E_{pk_{1,m+1}}(1, \bar{r}'^{(m+1)}) = g^{\sum_{j=1}^{m+1}(\alpha_j + r_j'^{(m+1)})} \cdot 1$, the encryption of the constant 1 with $pk_{1,m+1}$. Thus, $e = e_0 * e_1^{(s)} \cdot g^{r^{(s)}} = E_{pk_{1,m+1}^2}(1 \cdot x_s, \bar{r}'^{(m+1)}, \bar{r}^{(s)}) \cdot g^{r^{(s)}}$, that is, $e = g^{\sum_{j=1}^{m+1}(2\alpha_j + r_j^{(s)} + r_j'^{(m+1)})} \cdot x_s \cdot g^{r^{(s)}}$. Denote $l = g^{\sum_{j=1}^{m+1} r_j'^{(m+1)}}$, P_{m+1} publics $w = (e, l)$.

For $i = 1, 2, \ldots, m$, P_i computes its output by *w* and its own private numbers, $O_i = x_i' = e/d^{(i)} = g^{\sum_{j=1}^{m+1}((r_j^{(s)} - r_j^{(i)}) + (r^{(s)} - r^{(i)}))} \cdot x_s$, where $d^{(i)} = d_1^{(i)} \cdot d_2^{(i)}$, $d_1^{(i)} = E_{pk_{1,m+1}}(g^{r^{(i)}}, \bar{r}^{(i)}) = g^{\sum_{j=1}^{m+1}(\alpha_j + r_j^{(i)})} \cdot g^{r^{(i)}}$ and $d_2^{(i)} = E_{pk_{1,m+1}}(l, 0, \cdots, 0) = g^{\sum_{j=1}^{m+1}\alpha_j} \cdot g^{\sum_{j=1}^{m+1} r_j'^{(m+1)}}$.

If $O_i = x_i$, then P_i is selected, else he is not selected.

The pseudocode of the protocol is as follows.

Lottery SMC Protocol for the Selection Function in SDWSNs

Inputs: x_i is the input of P_i, $i = 1, 2, \cdots, m$;
1^n is the input of P_{m+1}.

Outputs: For $i = 1, 2, \ldots, m, i \neq s$, $O_i = x_i' \neq x_i$;
For $i = s$, $O_i = x_i$;
For $i = m + 1$, $O_{m+1} = P_s$.

Comparing phase

- P_{m+1} randomly chooses *n* numbers $(s_1', s_2', \ldots, s_n')$, computes and publics $c_k = E_{pk_{1,m+1}}(s_k', \bar{r}^{(m+1)}) = g^{\sum_{j=1}^{m+1}(\alpha_j + r_j^{(m+1)})} \cdot s_k'$, $k = 1, 2, \ldots, n$.
- For $i = 1, 2, \ldots, m$, $k = 1, 2, \ldots, n$, P_i locally computes $e_k^{(i)} = E_{pk_{1,m+1}}(x_i^k, \bar{r}^{(i)}) = g^{\sum_{j=1}^{m+1}(\alpha_j + r_j^{(i)})} \cdot x_i^k$ and $X_i' = \sum_{k=1}^{n} c_k * e_k^{(i)} = g^{\sum_{j=1}^{m+1}(2\alpha_j + r_j^{(i)} + r_j^{(m+1)})} \cdot \sum_{k=1}^{n}(s_k' \cdot x_i^k)$ and gets $X_i = X_i'/g^{\sum_{j=1}^{m+1} r_j^{(i)}} = g^{\sum_{j=1}^{m+1}(2\alpha_j + r_j^{(m+1)})} \cdot \sum_{k=1}^{n}(s_k' \cdot x_i^k)$. Denote $s_k = g^{\sum_{j=1}^{m+1}(2\alpha_j + r_j^{(m+1)})} \cdot s_k'$, then we have $X_i = \sum_{k=1}^{n}(s_k \cdot x_i^k)$.

Selecting phase

- For $i = 1, 2, \ldots, m$, P_i transmits $v_i = (X_i \cdot g^{\alpha}, P_i, e_1^{(i)} \cdot g^{r^{(i)}})$ to P_{m+1}. And then P_{m+1} selects the targeted vector from $V = (v_1, v_2, \ldots, v_m)$ according to the size of $X_i \cdot g^{\alpha}$. Suppose the *h*-th greatest number in $(x_1, x_2, \ldots, x_m)$ is x_s, then $v_s = (X_s \cdot g^{\alpha}, P_s, e_1^{(s)} \cdot g^{r^{(s)}})$ is the targeted vector. P_{m+1} chooses *m*+1 numbers $\bar{r}'^{(m+1)} = (r_1'^{(m+1)}, r_2'^{(m+1)}, \ldots, r_{m+1}'^{(m+1)})$ to compute $e_0 = E_{pk_{1,m+1}}(1, \bar{r}'^{(m+1)})$. Thus, $e = e_0 * e_1^{(s)} \cdot g^{r^{(s)}} = E_{pk_{1,m+1}^2}(x_s \cdot 1, \bar{r}^{(s)}, \bar{r}'^{(m+1)}) \cdot g^{r^{(s)}}$, that is, $e = g^{\sum_{j=1}^{m+1}(2\alpha_j + r_j^{(s)} + r_j'^{(m+1)})} \cdot x_s \cdot g^{r^{(s)}}$. Denote $l = g^{\sum_{j=1}^{m+1} r_j'^{(m+1)}}$, P_{m+1} publics $w = (e, l)$.
- For $i = 1, 2, \ldots, m$, P_i computes $d_1^{(i)} = E_{pk_{1,m+1}}(g^{r^{(i)}}, \bar{r}^{(i)}) = g^{\sum_{j=1}^{m+1}(\alpha_j + r_j^{(i)})} \cdot g^{r^{(i)}}$ and $d_2^{(i)} = E_{pk_{1,m+1}}(l, 0, \cdots, 0) = g^{\sum_{j=1}^{m+1}\alpha_j} \cdot g^{\sum_{j=1}^{m+1} r_j'^{(m+1)}}$ and then obtains $d^{(i)} = d_1^{(i)} \cdot d_2^{(i)} = g^{\sum_{j=1}^{m+1}(2\alpha_j + r_j^{(i)} + r_j'^{(m+1)}) + r^{(i)}}$. Finally, the output of P_i is $O_i = x_i' = e/d^{(i)} = g^{\sum_{j=1}^{m+1}((r_j^{(s)} - r_j^{(i)}) + (r^{(s)} - r^{(i)}))} \cdot x_s$.

IV. Proof

In this section, we give a formal proof for the proposed protocol. Suppose that the adversaries are semi-honest in our model. As defined in definition 1, it is said that Π privately computes *f* if there exists a probabilistic polynomial-time algorithm $\mathcal{S}$, such that for every adversary coalition $I = \{i_1, i_2, \ldots, i_t\}$ with *t* adversaries, it holds that,

$$\{\mathcal{S}(I, (x_{i_1}, x_{i_2}, \ldots, x_{i_t}), f_I(\bar{x}))\}_{\bar{x} \in (\{0,1\}^*)^m} \cong \{view_I^{\Pi}(\bar{x})\}_{\bar{x} \in (\{0,1\}^*)^m}$$

Herein, it is reasonable to suppose that no client can control the web server but the server itself can be a semi-honest adversary. That is, the server itself can be a "one-adversary coalition" when it is dishonest besides the adversary coalition consisted of dishonest clients. However, the client and the server are not in the same coalition. In the following, we argue the security of the proposed protocol and separately prove the case that P_{m+1} is honest and the case that it is not.

Firstly, when P_{m+1} is honest, other parties conduct the protocol following the construction since they are semi-honest. Therefore, correctness is immediate from the construction. Then for privacy, since the server is honest, it suffices to prove that the adversaries' view can be simulated without any help from the trusted party. Suppose that the adversary coalition $I = \{i_1, i_2, \ldots, i_t\}$ consists of *t* dishonest clients. From the real execution of the protocol, we can get the view of P_{i_j} (supposing P_{i_j} is P_j), denote $\bar{r}^{(j)} = (r_1^{(j)}, r_2^{(j)}, \ldots, r_{m+1}^{(j)}), \bar{c} = (c_1, c_2, \ldots, c_n)$,

$$view_{i_j}^{\Pi}(x_1, x_2, \ldots, x_m, 1^n) = \{x_j, \bar{r}^{(j)}, r^{(j)}, \bar{c}, e, l\}$$

Then, we will show how to simulate it in the ideal model. Giving P_{i_j}'s input x_j and output O_j to the simulator $\mathcal{S}_j$, it then chooses *m*+2 numbers $\bar{R}^{(j)} = (R_1^{(j)}, R_2^{(j)}, \ldots, R_{m+1}^{(j)})$ and $R^{(j)}$ and then computes $E_k^{(j)} = E_{pk_{1,m+1}}(x_j^k, \bar{R}^{(j)}) = g^{\sum_{i=1}^{m+1}(\alpha_i + R_i^{(j)})} \cdot x_j^k$, $k = 1, 2, \ldots, n$. In order to simulate $c_1, c_2, \ldots, c_n$, $\mathcal{S}_j$ randomly chooses $(S_1', S_2', \ldots, S_n')$ and $\bar{R}^{(m+1)} = (R_1^{(m+1)}, R_2^{(m+1)}, \ldots, R_{m+1}^{(m+1)})$ to compute $C_k = E_{pk_{1,m+1}}(S_k', \bar{R}^{(m+1)}) = g^{\sum_{i=1}^{m+1}(\alpha_i + R_i^{(j)})} \cdot S_k'$,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSEN.2016.2540002, IEEE Sensors Journal

$k = 1, 2, \ldots, n$. Owning these information, $\mathcal{S}_j$ can further compute $Y'_j = \sum_{k=1}^{n} C_k * E_k^{(j)}$ and $Y_j = Y'_j / g^{\sum_{i=1}^{m+1} R_i^j} = g^{\sum_{i=1}^{m+1}(2\alpha_i + R_i^{(m+1)})} \cdot \sum_{k=1}^{n}(S'_k \cdot x_j^k)$. Denote $S_k = g^{\sum_{i=1}^{m+1}(2\alpha_i + R_i^{(m+1)})} \cdot S'_k$, then we have $Y_j = \sum_{k=1}^{n}(S_k \cdot x_j^k)$. Now, $\mathcal{S}_j$ can produce $V_j = (X_j \cdot g^{\alpha}, P_j, E_1^{(j)} \cdot g^{R^{(j)}})$. Then, $\mathcal{S}_j$ transmits it to the TTP and the TTP sends P_s to the web server as its output after finding the targeted V_s. Next, $\mathcal{S}_j$ computes $L = g^{\sum_{i=1}^{m+1} R_i^{(m+1)}}$ and so $D_2^{(j)} = E_{pk_{1,m+1}}(L, 0, \ldots, 0) = g^{\sum_{i=1}^{m+1}(\alpha_i + R_i^{(m+1)})}$ and $D_1^{(j)} = E_{pk_{1,m+1}}(g^{R^{(j)}}, \bar{R}^{(j)}) = g^{\sum_{i=1}^{m+1}(\alpha_i + R_i^{(j)})} \cdot g^{R^{(j)}}$. Thus, $D^{(j)} = D_1^{(j)} \cdot D_2^{(j)} = g^{\sum_{i=1}^{m+1}(2\alpha_i + R_i^{(j)} + R_i^{(m+1)}) + R^{(j)}}$. Finally, $\mathcal{S}_j$ gets $E = O_j \cdot D^{(j)} = (g^{\sum_{i=1}^{m+1}((R_i^{(s)} - R_i^{(j)}) + (R^{(s)} - R^{(j)}))} \cdot x_s) \cdot g^{\sum_{i=1}^{m+1}(2\alpha_i + R_i^{(j)} + R_i^{(m+1)}) + R^{(j)}}$, that is, $E = g^{\sum_{i=1}^{m+1}(2\alpha_i + R_i^{(s)} + R_i^{(m+1)}) + R^{(s)}} \cdot x_s$. In short, the view of P_{i_j} in the ideal world is,

$$\{\mathcal{S}_{i_j}(x_j, O_j) = \{x_j, \bar{R}^{(j)}, R^{(j)}, \bar{C}, E, L\}$$

Where $\bar{C} = (C_1, C_2, \ldots, C_n)$. Since it is indistinguishable about two arrays of random numbers, we cannot distinguish $(r_1^{(j)}, r_2^{(j)}, \ldots, r_{m+1}^{(j)}, r^{(j)})$ from $(R_1^{(j)}, R_2^{(j)}, \ldots, R_{m+1}^{(j)}, R^{(j)})$ and $(c_1, c_2, \ldots, c_n)$ from $(C_1, C_2, \ldots, C_n)$. Moreover, the values of (x_j, O_j, E, L) have the same relations as (x_j, O_j, e, l) when constructed by the simulator $\mathcal{S}_j$. Thus, we have,

$$\{\mathcal{S}_{i_j}(x_j, O_j)\} \cong \{view_{i_j}^{\Pi}(x_1, x_2, \ldots, x_m, 1^n)\}$$

As we know,

$$\mathcal{S}(I, (x_{i_1}, x_{i_2}, \ldots, x_{i_t}), f_I(\bar{x})) = \{I, \mathcal{S}_{i_1}(x_1, O_1), \ldots, \mathcal{S}_{i_t}(x_t, O_t), f_I(\bar{x})\}$$

and

$$view_I^{\Pi}(\bar{x}) = (I, view_{i_1}^{\Pi}(\bar{x}), view_{i_2}^{\Pi}(\bar{x}), \ldots, view_{i_t}^{\Pi}(\bar{x}))$$

for $I = \{i_1, i_2, \ldots, i_t\}$, we have,

$$\{\mathcal{S}(I, (x_{i_1}, x_{i_2}, \ldots, x_{i_t}), f_I(\bar{x}))\}_{\bar{x} \in (\{0,1\}^*)^m} \cong \{view_I^{\Pi}(\bar{x})\}_{\bar{x} \in (\{0,1\}^*)^m}$$

In fact, the view of the adversary can be simulated since everything is encrypted under the key of the honest server. According to the above construction, we know that the simulated messages have the same relationships as the messages computed in real execution and it is impossible to distinguish the two types of messages directly from the hiding property of encryption.

Next, we consider the case that the server P_{m+1} is dishonest. Herein, the coalition with *t* adversaries can be denoted as $I = I_1 \bigcup I_2 = \{i_1, i_2, \ldots, i_{t-1}\} \bigcup \{i_{m+1}\}$. The client-adversaries act as the adversaries in the case that P_{m+1} is honest. Thus, for $j = 1, 2, \ldots, t-1$, the view of P_{i_j} is

$$view_{i_j}^{\Pi}(x_1, x_2, \ldots, x_m, 1^n) = \{x_j, \bar{r}^{(j)}, r^{(j)}, \bar{c}, e, l\}$$

From the discussion above, we can deduce,

$$\{\mathcal{S}_{I_1}(I_1, (x_{i_1}, x_{i_2}, \ldots, x_{i_{t-1}}), f_{I_1}(\bar{x}))\}_{\bar{x} \in (\{0,1\}^*)^m} \cong \{view_{I_1}^{\Pi}(\bar{x})\}_{\bar{x} \in (\{0,1\}^*)^m}$$

For the server P_{m+1}, it is also impossible to distinguish the real and ideal view since in the real world he gets the identity of the *h*-th greatest private number P_s by himself and in the ideal world he gets it from TTP. Formally, in the real world, denoting $\bar{s} = (s'_1, s'_2, \ldots, s'_n)$, the view of the server is

$$view_{i_{m+1}}^{\Pi}(x_1, x_2, \ldots, x_m, 1^n) = \{1^n, \bar{s}, \bar{r}^{(m+1)}, \bar{r}'^{(m+1)}, P_s\}.$$

Thus, in the ideal model, we simply have the simulator $\mathcal{S}_{m+1}$ randomly choose $\bar{S} = (S'_1, S'_2, \ldots, S'_n), \bar{R}^{(m+1)} = (R_1^{(m+1)}, R_2^{(m+1)}, \ldots, R_{m+1}^{(m+1)}), \bar{R}'^{(m+1)} = (R_1'^{(m+1)}, R_2'^{(m+1)}, \ldots, R_{m+1}'^{(m+1)})$ corresponding to $\bar{s}, \bar{r}^{(m+1)}, \bar{r}'^{(m+1)}$ and ask the TTP for the selected identity. Therefore, the simulated messages are

$$\mathcal{S}_{m+1}(1^n, O_{m+1}) = \{1^n, \bar{S}', \bar{R}, \bar{R}', P_s\}$$

Obviously, it is indistinguishable from the real view of P_{m+1}, that is,

$$\{\mathcal{S}_{m+1}(1^n, O_{m+1}) \cong \{view_{i_{m+1}}^{\Pi}(\bar{x})\}$$

As we know $view_{I_2}^{\Pi}(\bar{x}) = \{I_2, view_{i_{m+1}}^{\Pi}(\bar{x})\}$, we can get the result,

$$\{\mathcal{S}_{I_2}(I_2, 1^n, O_{m+1}) \cong \{view_{I_2}^{\Pi}(\bar{x})\}$$

Moreover, since

$$\{\mathcal{S}_{I_1}(I_1, (x_{i_1}, x_{i_2}, \ldots, x_{i_{t-1}}), f_{I_1}(\bar{x}))\}_{\bar{x} \in (\{0,1\}^*)^m} \cong \{view_{I_1}^{\Pi}(\bar{x})\}_{\bar{x} \in (\{0,1\}^*)^m}$$

Thus, denoting $\bar{x}_{I_1} = (x_{i_1}, x_{i_2}, \ldots, x_{i_{t-1}})$

$$\{\mathcal{S}_{I_1}(I_1, \bar{x}_{I_1}, f_{I_1}(\bar{x}))\} \bigcup \{\mathcal{S}_{I_2}(I_2, 1^n, O_{m+1})\} \cong \{view_{I_1}^{\Pi}(\bar{x})\} \bigcup \{view_{I_2}^{\Pi}(\bar{x})\}$$

Therefore,

$$\{\mathcal{S}_I(I, (x_{i_1}, x_{i_2}, \ldots, x_{i_t}), f_I(\bar{x}))\}_{\bar{x} \in (\{0,1\}^*)^m} \cong \{view_I^{\Pi}(\bar{x})\}_{\bar{x} \in (\{0,1\}^*)^m}$$

In this case, we find a simulator $\mathcal{S} = \mathcal{S}_I$ such that for every adversary coalition I, the simulative view $\{\mathcal{S}(I, (x_{i_1}, x_{i_2}, \ldots, x_{i_t}), f_I(\bar{x}))\}_{\bar{x} \in (\{0,1\}^*)^m}$ is indistinguishable to the real view $\{view_I^{\Pi}(\bar{x})\}_{\bar{x} \in (\{0,1\}^*)^m}$.

In a word, we can conclude that there exists a probabilistic polynomial-time algorithm $\mathcal{S}$, such that for every adversary coalition I, it holds that $\{\mathcal{S}(I, (x_{i_1}, x_{i_2}, \ldots, x_{i_t}), f_I(\bar{x}))\}_{\bar{x} \in (\{0,1\}^*)^m} \cong \{view_I^{\Pi}(\bar{x})\}_{\bar{x} \in (\{0,1\}^*)^m}$. Thus, the proposed protocol Π privately computes the selection function *f*.

V. Conclusion

In this paper, by applying layered homomorphic encryption and the equivalent transformation of two arrays, we have proposed an adapted model and further obtained a lottery SMC protocol in SDWSNs with detailed proof. We expect to provide a new perspective on this problem for researchers.

Future work should focus on the following two aspects.

(1)Introducing other practical tools besides homomorphic encryption to protect the security of private sensory data.

(2)Constructing new secure protocols in SDWSNs besides the secure selection function protocol.

ACKNOWLEDGMENT

This work is supported by the National High Technology Research and Development Program of China (863 Program) (Grant No. 2013AA014702), the Fundamental Research Funds for the Central Universities (Grant Nos. 2014ZD03-03).

REFERENCES

[1] A.C. Yao, Protocols for secure computations, in Proceedings of the 23rd Annual IEEE Symposium on Foundations of Computer Science, Chicago, 1982, pp. 160-164.

[2] Y. Zhang, M. Chen, S. Mao, et al., Cap: Community activity prediction based on big data analysis. Network, IEEE, 2014, vol. 28(4), pp. 52-57.

[3] Y. Zhang, M. Qiu, C.W. Tsai, et al., Health-CPS: Healthcare Cyber-Physical System Assisted by Cloud and Big Data. 2015.

[4] Shai Halevi, Yehuda Lindell, Benny Pinkas., Secure computation on the web: computing without simultaneous interaction, Advances in Cryptology-CRYPTO, 2011, Springer Berlin Heidellberg, pp. 132-150.

[5] Y.D. Zhang, S. Wang, P. Phillips, et al., Binary PSO with mutation operator for feature selection using decision tree applied to spam detection. Knowledge-Based Systems, 2014, vol.64,pp. 22-31.

[6] Y.D. Zhang, S. Wang, P. Phillips, et al., Detection of Alzheimer's disease and mild cognitive impairment based on structural volumetric MR images using 3D-DWT and WTA-KSVM trained by PSOTVAC[J]. Biomedical Signal Processing and Control, 2015, vol. 21, PP. 58-73.

[7] J. Vaidya, M. Kantarcloglu, C. Clifton. Privacy-preserving naive bayes classification, The VLDB Journal-The International Journal on Very Large Data Bases, 2008, vol. 17(4), pp. 879-898.

[8] C. Clifton, M. Kantarcioglu, J. Vaidya, et al., Tools for privacy preserving distributed data mining, ACM SIGKDD Explorations Newsletter, 2002, vol. 4(2), pp. 28-34.

[9] Y. Lindell, B. Pinkas, Privacy preserving data mining, Journal of cryptology, 2002, vol. 15(3), pp. 177-206.

[10] F. Herrmann, D. Khadraoui, Y. Lanuel. Secure MultiParty Computation Problem for Distributed Electronic Contract Management, Information and Communication Technologies, 2006. ICTTA'06. 2nd. IEEE, 2006, vol. 1, pp. 274-279.

[11] Y. Sun, Q. Wen, Y. Zhang, W. Li, Privacy-Preserving Self-Helped Medical Diagnosis Scheme Based on Secure Two-Party Computation in Wireless Sensor Networks. Computational Mathematical Methods in Medicine, 2014, 2014(7), pp. 214841-214841.

[12] M. Chen, Y. Zhang, L. Hu, et al., Cloud-based Wireless Network: Virtualized, Reconfigurable, Smart Wireless Network to Enable 5G Technologies. Mobile Networks and Applications, 2015, PP. 1-9.

[13] L. PANG, M. SUN, S. LUO, et al., Full privacy preserving electronic voting scheme, The Journal of China Universities of Posts and Telecommunications, 2012, vol. 19(4), pp. 86-93.

[14] Y. Sun, H. Sun, H. Zhang, Q.H. Wen, A secure protocol for point-segment position problem, Web Information Systems and Mining. Springer Berlin Heidelberg, 2010, pp. 212-219.

[15] Y. Zhang, D. Zhang, M.M. Hassan, et al., CADRE: Cloud-assisted drug recommendation service for online pharmacies. Mobile Networks and Applications, 2014, pp. 1-8.

[16] Y. Zhang, M. Chen, D. Huang, et al., iDoctor: Personalized and professionalized medical recommendations based on hybrid matrix factorization. Future Generation Computer Systems, 2016.

[17] R. Canetti, Security and composition of multiparty cryptographic protocols, Journal of Cryptology, 2000, vol. 13(1), pp. 143-202.

[18] O. Goldreich, S. Mical, A. Wigderson, How to play any mental game, in Proceedings of the nineteenth annual ACM symposium on Theory of computing STOC'87, New York: ACM, 1987, pp. 218-229.

[19] O. Goldreich, Secure multi-party computation, Manuscript, Preliminary version, 1998.

[20] M.M. Prabhakaran, A. Sahai, eds., Secure multiparty computation, IOS press, 2013.

[21] Y. Lindell, B. Pinkas., A proof of Yao's protocol for secure two-party computation, J. Cryptology, 2009, vol. 22, pp. 161-188.

[22] R. Rivest, L. Adleman,and M. Dertouzos, On data banks and privacy homomorphisms, in Foundations of Secure Computation, Academic Press, 1978, pp. 169-177.

[23] H.Y. Lin, W.G. Tzeng, An efficient solution to the millionaires problem based on homomorphic encryption, ASIACRYPT 2005, http://eprint.iacr.org/2005/043.

[24] C. Gentry., Fully homomorphic encryption using ideal lattices, In: STOC, pp. 169-178, 2009.

[25] Gilad Asharov, Abhishek Jain, Adriana LÍőpez-Alt, Eran Tromer, Vinod Vaikuntanathan, Daniel Wichs, Multiparty computation with low communication, computation and interaction via threshold FHE, In Eurocrypt 2012, the 31st Annual IACR Eurocrypt Conference.

[26] Adriana Lopez-Alt, Eran Tromer, Vinod Vaikuntanathan, On-the-Fly multiparty computation on the cloud via multikey fully homomorphic encryption, Symposium on Theory of Computing-STOC 2012.

[27] F. Caroline, G. Fabien, A survey of homomorphic encryption for nonspecialists. EURASIP Journal on Information Security, 2007, 2007(15).

[28] C.M. Tang, G.H. Shi, Z.A. Yao, Secure multiparty computation protocol for sequencing problem, Science China Information Sciences, 2011, vol. 54(8), pp. 1654-1662.

[29] O. Goldreich, Foundations of Cryptography: Volume 2, Basic Applications. Cambridge university press, 2004.