

Three New Approaches to Privacy-preserving Add to Multiply Protocol and Its Application

Youwen Zhu, Liusheng Huang, Wei Yang, Dong Li,
Yonglong Luo and Fan Dong

(National High Performance Computing Center at Hefei, Department of Computer Science and Technology,
University of Science and Technology of China, Hefei, 230027, P. R. China)
(e-mail: zhuyw@mail.ustc.edu.cn, {lshuang, qubit}@ustc.edu.cn, lixido@mail.ustc.edu.cn, ylluo@ustc.edu.cn,
dongfan@mail.ustc.edu.cn)

Abstract—Privacy-preserving Data Mining aims at securely extracting knowledge from two or more parties’ private data. Secure Multi-party Computation is the paramount approach to it. In this paper, we study Privacy-preserving Add and Multiply Exchanging Technology and present three new different approaches to Privacy-preserving Add to Multiply Protocol. After that, we analyze and compare the three different approaches about the communication overheads, the computation efforts and the security. In addition, we extend Privacy-preserving Add to Multiply Protocol to Privacy-preserving Adding to Scalar Product Protocol, which is more secure and more useful in the high security situations of Privacy-preserving Data Mining. Meantime, we present a solution for the new protocol.

Keywords—Privacy-preserving; Secure Multiparty Computation; Add and Multiply exchanging technolog; Add to Multiply Protocol

I. INTRODUCTION

The purpose of data mining is to extract knowledge from the collection of data. Most existing data mining algorithms are carried out under the assumption that all the data could be available at a single central site. While two or more parties, who don’t have enough confidence in each other or are even rival individuals or organizations, have a common desire to extract knowledge from all of their private data, the privacy problems come up. Privacy-preserving Data Mining (PPDM) aims at making privacy and data mining coexist under the circumstances.

Secure Multi-party Computation (SMC) [1,2] is to allow a group of participants to perform a cooperative computation, based on their private inputs, in a special way that everyone knows the cooperative computation’s result, but no one learns anything about any other parties’ private inputs. Since 1982, when A. C. Yao [1] firstly introduced SMC, it has been a hot research topic and attracts numerous researchers. In recent years, SMC has become the paramount approach to PPDM, and there are a great many applications of basic SMC protocols [3-12].

The *Scalar Product Protocol* [14, 15] was a significant basic protocol of SMC, and lots of problems can essentially be reduced to computing scalar product [7, 8, 14, 15, 17]. We redefine the protocol in Protocol 1.

In [16], Weijiang Xu proposed *Privacy-preserving Add and Multiply Exchanging Technology*, which contains two basic protocols, *Privacy-preserving Multiply to Add Protocol (PPMtAP)* and *Privacy-preserving Add to Multiply Protocol (PPAtMP)*. PPMtAP is *one-dimension Scalar Product Protocol*. In a cooperative computation, if Alice and Bob respectively have private number x and y , and they share the secret $s=x \cdot y$. Using PPMtAP (shown in Fig. 1.), they can change the sharing form from multiplying ($s=x \cdot y$) to adding ($s=u+v$). Opposite to PPMtAP, PPAtMP (shown in Fig. 2.) changes the sharing modality from adding to multiplying. The role-exchanging of data sharing form could arouse some solutions for many privacy-preserving cooperative computations [16, 18].

Protocol 1. *Scalar Product Protocol*

Input: Alice has a private input vector \vec{x} , and Bob has his private input vector \vec{y} .

Output: Alice gets secret output u , and Bob obtains the private number v , such that $\vec{x}^T \cdot \vec{y} + u = v$.

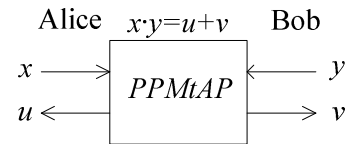


Fig. 1. Privacy-preserving Multiply to Add Protocol

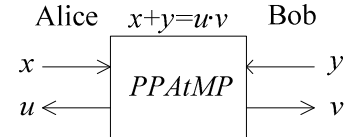


Fig. 2. Privacy-preserving Add to Multiply Protocol

This work was supported by the National Natural Science Foundation of China (No. 60773032 & 60703071), the Ph.D. Program Foundation of Ministry of Education of China (No. 20060358014), the Natural Science Foundation of Jiangsu Province of China (No. BK2007060), the Anhui Provincial Natural Science Foundation (No. 070411043), and the Science Foundation of Anhui Province for Excellent Young Scholar (08040106806).

Privacy-preserving Add and Multiply Exchanging Technology [16] is the development of *Scalar Product Protocol*, and it’s more practical. Nevertheless, Xu hasn’t developed any effectual approach to PPAtMP. Then, we

propose three new different approaches to *PPAtMP*. In addition, we extend *PPAtMP* to a more secure protocol.

The remaining sections are organized as follows: Section 2 discusses the preliminaries. Section 3 presents three new different approaches to *PPAtMP*. In section 4, we analyze and compare our three new different approaches to *PPAtMP*. Finally, we summarize the paper and propose our plans for future work in section 5.

II. PRELIMINARIES

In the section, we definite the security model and describe some cryptographic tools and SMC protocols.

A. Definition of security model

Many SMC protocols for PPDM [4, 6, 8, 17] are under the semi-honest model [2, 13]. Generally, a semi-honest participant is the one who accurately follows the secure protocols, but it keeps a record of all its intermediate calculating data to analyze for more information. Here, we assume that all the participants are semi-honest behaviors.

The formal definition for the semi-honest model has been presented by Goldreich [13] in 2004.

Let $f: \{0, 1\}^* \times \{0, 1\}^* \rightarrow \{0, 1\}^* \times \{0, 1\}^*$ be a binary function, and $f_1(x, y)$ (resp., $f_2(x, y)$) denotes first (resp., second) element of $f(x, y)$. Let Π be a SMC protocol for computing f . During an execution of protocol Π , the view of first (resp., second) party, which is denoted by $view_1^\Pi(x, y)$ (resp., $view_2^\Pi(x, y)$), is $(x, r_1, m_1, \dots, m_t)$ (resp., $(y, r_2, m_1, \dots, m_t)$), where r_1 (resp., r_2) represents the outcome of first (resp., second) party's internal coin tosses, and m_i represents the i -th message it has received. The output of protocol Π is denoted by $output^\Pi(x, y)$ after an execution of the protocol.

Definition 1 (privacy with respect to semi-honest behavior): Regarding f , if there is a probabilistic polynomial-time algorithm, denoted by S , such that

$$\begin{aligned} & \left\{ (S(x, f_1(x, y)), f(x, y)) \right\}_{x, y \in \{0, 1\}^*} \\ & \equiv^c \left\{ (view_1^\Pi(x, y), output^\Pi(x, y)) \right\}_{x, y \in \{0, 1\}^*} \\ & \left\{ (S(y, f_2(x, y)), f(x, y)) \right\}_{x, y \in \{0, 1\}^*} \\ & \equiv^c \left\{ (view_2^\Pi(x, y), output^\Pi(x, y)) \right\}_{x, y \in \{0, 1\}^*} \end{aligned}$$

Where \equiv^c denotes computational indistinguishability by any polynomial-time algorithm, we say that the SMC protocol Π securely computes f .

According to definition 1, a SMC protocol is capable to securely compute f if and only if all the information of a certain participant's view during an execution could be deduced from his private input and the output to that party.

B. Homomorphic Encryption

An encryption function $H: R \rightarrow S$ is called *additively homomorphic* if the following condition holds:

$$H(x) \times H(y) = H(x+y) \quad (1).$$

We can employ the *Homomorphic Encryption* function to compute $H(x+y)$ from $H(x)$ and $H(y)$, such that $H(x)$ and $H(y)$ have no need of being decrypted.

Derived from the formula (1), *Homomorphic Encryption* function has the property,

$$H(n \times x) = (H(x))^n \quad (2).$$

Here, n ($1 < n$) is a natural number.

Many *Homomorphic Encryption* systems have been proposed in [19-22]. We will make use of *Homomorphic Encryption* system to develop a new *PPAtMP*.

C. 1-out-of- n Obvious Transfer Protocol

Obvious Transfer Protocol, which has two participants, was put forth in 1981. Since then, many attentions [2, 13, 23] have been paid to the protocol.

1-out-of- n Obvious Transfer Protocol [13, 23] deals with the following scenario. Bob has n private inputs x_1, x_2, \dots, x_n , and Alice has a secret natural number i ($1 \leq i \leq n$). The goal of them is that Alice obtains x_i and learns nothing of x_j ($1 \leq j \leq n, j \neq i$), and Bob knows nothing about Alice's secret number i . *1-out-of- n Obvious Transfer Protocol* is one of significant tools to build the privacy-preserving protocols that we will propose and analyze.

III. THREE DIFFERENT APPROACHES TO PPATMP

We define the protocol *PPAtMP* in Protocol 2. Then, three new approaches to *PPAtMP* are presented. The first two are respectively based on *Homomorphic Encryption* and *Obvious Transfer Protocol*, and the third way uses a semi-honest third party to reduce the overheads.

Protocol 2. Privacy-preserving Add to Multiply Protocol (*PPAtMP*)

Input: Alice has the private number x , and Bob has his private input y .

Output: Alice gets the secret output u , and Bob obtains the private number v , such that $x+y=u \cdot v$.

A. *PPAtMP* based on Homomorphic Encryption System

Homomorphic Encryption system allows us to compute the sum of encrypted data without decrypted them. We make use of the property to propose a new solution for *PPAtMP*. The full protocol is given in Protocol 3.

Correctness Since (pk, sk) is a *Homomorphic Encryption* key pair, we have the equations as below.

$$C' = C^p \cdot E_{pk}(y \cdot p) = E_{pk}(p \cdot x) \cdot E_{pk}(p \cdot y) = E_{pk}(p \cdot x + p \cdot y).$$

Therefore, $u = D_{sk}(C') = x \cdot p + y \cdot p$.

Let v be $1/p$. Then Alice and Bob respectively get private number u and v , which meet the equation $x+y=u \cdot v$.

Security During the execution of *PPAtMP_HES*,

- (1) the key pairs (pk, sk) and
- (2) $u, C' = C^p \cdot E_{pk}(y \cdot p)$.

Protocol 3. *PPAtMP* based on *Homomorphic Encryption System (PPAtMP_HES)*

Step 1: Alice generates *Homomorphic Encryption* key pairs (pk, sk) . Here pk is the public key, and sk is private key. Then Alice encrypts her secret input x , $C=E_{pk}(x)$, and sends C and pk to Bob.

Step 2: Bob randomly chooses plaintext p , and then computes $C' = C^p \cdot E_{pk}(y \cdot p)$. Send C' to Alice.

Step 3: Alice obtains $u=D_{sk}(C')=x \cdot p + y \cdot p$, and Bob computes $v=1/p$.

// $E_{pk}(\cdot)$ and $D_{sk}(\cdot)$ respectively denote encryption and decryption.

Bob learns (1) the public key pk and
(2) $C=E_{pk}(x)$, $v=1/p$.

Bob only knows C , but he learns nothing about the private key sk . Therefore, he can't find Alice's secret x . Alice obtains u by decrypting C' . However, she knows nothing regarding Bob's private number p , then, none of v , p and y , which are Bob's private data, will be revealed.

B. PPAtMP based on 1-out-of-n Obvious Transfer Protocol

The protocol *PPAtMP_HES* employs *Homomorphic Encryption* system, as a result the private inputs x and y must consist in the definitional domain of *Homomorphic Encryption* function. What's more, Bob's secret number $v=1/p$ is bound to the interval $(0, 1)$.

Naor [] and Cachin [] put forward a high-efficiency *1-out-of-n Obvious Transfer Protocol*. Based on *1-out-of-n Obvious Transfer Protocol*, we present another new approach to *PPAtMP*, which overcomes some shortcomings of *PPAtMP_HES*. Protocol 4 describes the new protocol.

Protocol 4. *PPAtMP* based on *1-out-of-n Obvious Transfer Protocol (PPAtMP_OTP)*

Step 1: Alice and Bob promise two integers n and m , such that n^m is big enough.

Step 2: Alice randomly choose x_1, x_2, \dots, x_m , such that $x = \sum_{i=1}^m x_i$; Bob secretly obtains non-zero real number v , and generates m uniform random numbers r_1, r_2, \dots, r_m , such that $y/v = \sum_{i=1}^m r_i$.

Step 3: for every $j=1, 2, \dots, m$, Alice and Bob jointly carry out the following three sub-steps,

Step 3.1: Alice generates random integer k , random real number h_1, h_2, \dots, h_n , such that $h_k = x_j$. Then Alice sends h_1, h_2, \dots, h_n to Bob.

Step 3.2: for every $i=1, 2, \dots, n$, Bob computes $s_{ji} = h_i/v + r_j$.

Step 3.3: Using *1-out-of-n Obvious Transfer Protocol*, Alice receives s_{jk} , but Bob learns nothing about k .

Step 4: Alice obtains $u = \sum_j s_{jk} = x/v + (r_1 + r_2 + \dots + r_m)$
 $= x/v + y/v$.

Correctness It is clear that Alice's private outgoing u is equal to the sum $x/v + (r_1 + r_2 + \dots + r_m)$. Besides, the sum of $r_i (1 \leq i \leq m)$ is y/v . As a consequence, u is equivalent to the sum of x/v and y/v , where v is Bob's secret. That's to say, $x + y = u \cdot v$.

Security After *PPAtMP_OTP*, Alice learns nothing about Bob's privacy y , v and $r_i (1 \leq i \leq m)$. Bob receives m groups data $h_j (1 \leq j \leq n)$, as a result he has the probability n^{-m} to guess the exact value of $x = \sum_{i=1}^m x_i$. While n^m is large enough, the probability n^{-m} is negligible. Namely, *PPAtMP_OTP* is secure.

A strong point of *PPAtMP_OTP* is that participants can adjust two parameters n and m , such that security and cost of communication and computation are in balance. When n^m is larger, the protocol *PPAtMP_OTP* is more secure, however, communication and computation complexity are higher. Therefore, they can jointly select two appropriate integers n and m , such that they satisfy security, simultaneously, the whole communication overheads and computation efforts are as low as reasonably achievable.

C. PPAtMP using Semi-honest Third Party

Semi-honest third party is a significant ministrant in some SMC protocols. The semi-honest adjuvant party can't collude with any participants who provide private data for SMC protocols. Besides, the third party will learn nothing about all parties' privacy from the execution of SMC protocols. By means of semi-honest third party, we put forward a new method for *PPAtMP*. With the help of semi-honest third party, the communication complexity and computation overheads are greatly reduced. The following is the full privacy-preserving protocol.

Protocol 5. *PPAtMP* using semi-honest third party (*PPAtMP_STP*)

Step 1: The semi-honest third party randomly chooses ca, cb, pa and pb , such that $ca + cb = pa \cdot pb$. Then it secretly sends ca and pa to Alice, and gives Bob cb and pb .

Step 2: Alice computes $x' = x + pa$, and sends x' to Bob.

Step 3: Bob chooses random non-zero real number v , and computes $t = 1/v + pb$ and $s = (x' + y)/v + cb$. Send t and s to Alice.

Step 4: Alice obtains $u = s \cdot pa \cdot t + ca$.

Correctness According to the process of *PPAtMP_STP*, we have the following equations,

$$\begin{aligned} u &= s \cdot pa \cdot t + ca = (x' + y)/v + cb - pa \cdot (1/v + pb) + ca \\ &= (x + pa + y)/v + (cb + ca) - (pa/v + pa \cdot pb) \\ &= (x + y)/v + pa/v + pa \cdot pb - (pa/v + pa \cdot pb) = (x + y)/v. \end{aligned}$$

Therefore, $x + y = u \cdot v$.

Security In *PPAtMP_STP*, Bob only receives $x' = x + pa$ from Alice. Due to the randomness of pa , Bob can't figure out Alice private input x .

Alice can't learn y or v from t and s , if the semi-honest third party doesn't collude with her, when Alice knows

nothing about cb and pb . For the security reasons, the semi-honest third party must re-choose the parameters ca , cb , pa and pb every time.

IV. THE ANALYSIS OF PPA_tMP

In the previous section, we present three different approaches to PPA_tMP. Next, we will compare the complexity and security of them. Further, we will analyze the applications of *Privacy-preserving Add and Multiply Exchanging Technology* and extend PPA_tMP.

A. Comparison of the three solutions

In the sub-section, we analyze PPA_tMP_HES, PPA_tMP_OTP and PPA_tMP_STP, and then discuss their cost and security. The results are as follow.

The communication and computation overheads of PPA_tMP_HES, PPA_tMP_OTP and PPA_tMP_STP are displayed in Tab. 1. Here, $O(HE)$ and $O(OT)$ respectively denotes the relevant complexity of *Homomorphic Encryption* function and *Obvious Transfer Protocol*.

In PPA_tMP_HES, Alice sends C and pk to Bob, and Bob transmits C' to Alice. Therefore, The communication complexity of it is $O(1)$. Alice performs one each of encryption and decryption, then, her computation overheads is $O(HE)$. Bob spends computation the time of $O(HE+p)$ in computing C' .

The executions of *Obvious Transfer Protocol* cost most in PPA_tMP_OTP. As a result, the communication cost and the computation overheads both are $O(m \cdot OT)$.

Regarding PPA_tMP_STP, it has the lowest computation cost and the fixed communication overheads, but it spends efforts in employing the semi-honest third party.

TABLE 1. COMPARISON OF COMPLEXITY

SMC Protocols	Communication Complexity	Computation Overheads	
		Alice	Bob
PPA _t MP_HES	$O(1)$	$O(HE)$	$O(HE+p)$
PPA _t MP_OTP	$O(m \cdot OT)$	$O(m \cdot OT)$	$O(m \cdot OT)$
PPA _t MP_STP	$O(1)$	$O(1)$	$O(1)$

From Tab. 1, PPA_tMP_HES and PPA_tMP_STP have the staid small communication expense. However, the inputs of PPA_tMP_HES are limited to the definitional domain of *Homomorphic Encryption* function, and PPA_tMP_STP has to employ a semi-honest third party, which probably increases the risk of security and brings about additional expenses. PPA_tMP_OTP has higher communication and computation complexity, but it doesn't have the above-mentioned flaws. Besides, Alice and Bob can jointly choose appropriate n and m , such that privacy is properly preserved, simultaneously, the communication and computation overheads are as low as possible.

The security of PPA_tMP_HES is the same as

Homomorphic Encryption function. In PPA_tMP_OTP, Bob has the probability n^{-m} to find out Alice's private input x . While the semi-honest third party selects mutually independent random parameters at an execution, PPA_tMP_STP reveals nothing of privacy unless colluding.

As a result, we can select the most appropriate approach to PPA_tMP for the concrete PPDM problems according to the security and complexity requirement.

B. The Applications of PPA_tMP and PPM_tAP and the Extended Protocol of PPA_tMP

Let $f(x, y)$ be an arbitrary polynomial, where x and y are respectively Alice's and Bob's private data. Besides, p and a are real number. Then, $f(x, y)$, $(f(x, y))^p$ and $a^{f(x, y)}$ etc. can be simplified to $u+v$ or $u \cdot v$ via PPA_tMP and PPM_tAP, where u and v are two secret outputs. After that, $u+v$ and $u \cdot v$ could be used to participate in PPDM protocol instead of the original complicated expression. Xu [17] presents a protocol to securely compute the Minkowski distance between two private vectors, which is applicable for various PPDM protocols.

The following is the brief process to simplify $f(x, y)$, $(f(x, y))^p$ and $a^{f(x, y)}$. Because $f(x, y)$ is a polynomial, its elements are the factors which can be denoted by $a(bx^c \cdot y^d + ex^g \cdot y^h)^k$. By PPA_tMP and PPM_tAP, Alice and Bob can jointly perform a protocol to respectively obtain secret number u_1 and v_1 , s. t. $a(bx^c \cdot y^d + ex^g \cdot y^h)^k = u_1 + v_1$. Then, they can further privately compute u and v , such that $f(x, y) = u + v$.

Let $ua = a^u$, and $va = a^v$. Then, $a^{f(x, y)} = ua \cdot va$.

Again, they invoke PPA_tMP to securely find out ut and vt , which meet $u+v=ut \cdot vt$. Let $up = ut^p$ and $vp = vt^p$. Then, $(f(x, y))^p = up \cdot vp$.

PPA_tMP can change the secrecy's sharing style from adding to multiplying. For the reason of security, we extend PPA_tMP as follows.

Protocol 6. Privacy-preserving Adding to Scalar Product Protocol (PPA_tSPP)

Input: Alice has a private number x , and Bob has the private input y .

Output: Alice gets the secret n -dimension vector \vec{u} , and Bob obtains his private output, n -dimension vector \vec{v} , such that $x+y = \vec{u}^T \cdot \vec{v}$.

PPA_tSPP, which is the extended protocol of PPA_tMP, is the inverse *Scalar Product Protocol*. It transforms the sharing style from the summation of two private numbers to the scalar product of two private vectors. Distinctly, the scalar product of two private vectors is more secure than the product of two secret numbers. That's to say, PPA_tSPP is better than PPA_tMP in the security. PPA_tSPP is more

powerful in the scene having a high requirement on security. The Protocol 7 is a solution for *PPAtSPP*.

During an execution of *PPAtSPP*, *PPAtMP* are invoked for n times. Therefore, if $O(AM)$ is the cost when running *PPAtMP* at a time, the overhead of *PPAtSPP* is $O(n \cdot AM)$.

Protocol 7. *PPAtSPP* based on *PPAtMP*

Step 1: Alice randomly generates x_1, x_2, \dots, x_n , which meet $x = \sum_{i=1}^n x_i$. Bob selects n uniform random numbers y_1, y_2, \dots, y_n , such that $y = \sum_{i=1}^n y_i$.

Step 2: For every $i=1, 2, \dots, n$, Alice and Bob collaboratively perform *PPAtMP* to privately find out respective secret output u_i and v_i , such that $x_i + y_i = u_i \cdot v_i$. After that, Alice obtains the secret vector $\vec{u} = (u_1, u_2, \dots, u_n)$, and Bob finds out his private vector $\vec{v} = (v_1, v_2, \dots, v_n)$.

V. CONCLUSION AND FUTURE WORK

In this paper, we study some basic SMC protocols and put forward three new different approaches to *PPAtMP*, which is practical for lots of PPDM problems. Then, we analyze and compare the three approaches about the communication complexity, the computation overheads and the security. In addition, we extend *PPAtMP* to *PPAtSPP*, which has better security and is more powerful in high security situations, and propose a solution for the new privacy-preserving protocol. For further research, we will make efforts to develop practical SMC protocols for various kinds of concrete PPDM problems.

REFERENCES

- [1] A. C. Yao. "Protocols for secure computations," Proc. of 23rd Annual IEEE Symposium on Foundations of Computer Science. Los Alamitos: IEEE Computer Society Press, 1982, pp.160-164.
- [2] O. Goldreich, "Secure multi-party computation," (manuscript), 2002, <http://theory.lcs.mit.edu/doi>.
- [3] C. Cachin, "Efficient private bidding and auctions with an oblivious third party," Proc. of the 6th ACM conference on Computer and Communications Security, Singapore, 1999, pp.120-127.
- [4] W. L. Du, M. J. Atallah, "Secure multi-party computation problems and their applications: A review and open problems," Proc. of New Security Paradigms Workshop, Cloudfroft, New Mexico, USA, 2001, pp.11-20.
- [5] W. L. Du, M. J. Atallah, F. Kerschbaum, "Protocols for secure remote database access with approximate mathing," Proc. of the First Workshop on Security and Privacy in E-Commerce, Nov. 2000.
- [6] Yong-Long LUO, Liu-Sheng HUANG, "An algorithm for privacy-preserving Boolean association rule mining," *Acta Electronica Sinica (in Chinese)*, Vol.33, No.5, 2005, pp.900-903.
- [7] Yong-Long LUO, Liu-Sheng HUANG, "Privacy-preserving Distance Measurement and Its Application," *Chinese Journal of Electronics* Vol.15, No.2, 2006, pp.237-241.
- [8] Yong-Long LUO, Liu-Sheng HUANG, "Privacy-preserving Cross Product Protocol and Its Applications," *Chinese Journal of Computers*, Vol.30, No.2, Feb.2007.
- [9] Yong-Long LUO, Liu-Sheng HUANG, "Secure Two-Party Point-Circle Inclusion Problem," *Journal of Computer Science & Technology*, Jan. 2007, Vol.22, No.1, pp.88-91.
- [10] M. J. Atallah, W. L. Du, "Secure multi-party computational geometry," Lecture Notes in Computer Science, Vol. 2125, Springer. Proc. of 7th International Workshop on Algorithms and Data Structures (WADS), 2001, pp.165-179.
- [11] W. L. Du, Y. S. Han and S. G. Chen, "Privacy-preserving multi-variety statistical analysis: linear regression and classification," Proc. of the 4th SIAM International Conference on Data Mining, 2004, pp.222-233.
- [12] Qi WANG, Yong-Long LUO, Liu-Sheng HUANG, "Privacy-preserving Protocol for Finding the Convex Hulls," Proc. of the Third International Conference on Availability, Reliability and Security, Barcelona, Spain, 2008, pp.727-732.
- [13] O. Goldreich, "*Fotmdations of Cryptography: Volume II, Basic Applications*," Cambridge: Cambridge University Press, 2004.
- [14] J. Vaidya, C. Clifton, "Privacy preserving association rule mining in vertically partitioned data," Proc. of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Edmonton, Canada, 2002, pp. 639-644.
- [15] I. Ioannidis, A. Grama, M. Atallah, "A secure protocol for computing dot-products in clustered and distributed environments," Proc. of the 2002 International Conference on Parallel Processing,, Vancouver, British Columbia, 2002, pp. 379-384.
- [16] Wei-Jiang Xu, "Research on Private Data Protection and Its Applications in Network Computing [Ph. D. dissertation]," P. R. China: University of Science and Technology of China, 2008.
- [17] A. Amirbekyan, V. Estivill-Castro, "A New Efficient Privacy-Preserving Scalar Product Protocol", Proc. of the Sixth Australasian Data Mining Conference (AusDM 2007), Gold Coast, Australia, 2007, pp. 209-214.
- [18] You-wen ZHU, Liu-Sheng HUANG, Wei YANG, "Privacy-preserving Practical Convex Hulls Protocol", the 2008 Japan-China Joint Workshop on Frontier of Computer Science and Technology (FCST 2008), Nagasaki, Japan, Dec. 2008.
- [19] J. Benaloh, "Dense probabilistic encryption", Proc. of the Workshop on Selected Areas of Cryptography, Kingston, Canada, May 1994, pp.120-128.
- [20] D. Naccache, J. Stern, "A new public key cryptosystem based on higher residues", Proc. of the 5th ACM conference on Computer and Communications Security, San Francisco, USA, 1998, pp.59-66.
- [21] T. Okamoto, S. Uchiyama, "A new public-key cryptosystem as secure as factoring", *Advances in Cryptology-Eurocrypt 1998*, LNCS 1403, Springer, 1998, pp.308-318.
- [22] P. Paillier, "Public key cryptosystems based on composite degree residuosity classes", *Advances in Cryptology-Eurocrypt 1999*, LNCS 1592, Springer, 1999, pp.223-238.
- [23] G.Brassard, C.Crepeau, J.Robert. "All-or-nothing disclosure of secrets", *Advances in Cryptology-Crypto86*, Lecture Notes in Computer Science, 1987, pp.234-238.