Multi-client Authenticated Intersection Protocol from a Negative Database Server

Hongfeng Zhu, Xueying Wang, Jingyue Zhao, Shuai Geng, Liwei Wang

Software College, Shenyang Normal University, China
zhuhongfeng1978@163.com, {751661713, 610035079, 1036103490, 1696751943}@qq.com

Abstract

The negative information (the complement of the original data) is a promising technology proven to provide diversified secure functions of Applications on Internet, such as privacy protection, intersection query, and negative authentication and so on. In this paper, we firstly propose a secure three-party authentication intersection protocol based on negative database which storing negative information. By carefully designing the proposed protocol, we expand the three-party protocol to multi-party instance while making the calculated consumption and communicating round restrict to an acceptable bound. In the two proposed protocols, we achieve the completion of authentication and data encryption transmission in one time. The server only plays the role of authentication and auxiliary transmission. At the same time, the client's private information will not be leaked to others in the process of calculating intersections, and all the clients participating in the intersection calculation can only obtain their intersection information, while other database information can be secure and confidential. This paper proves that a negative database can be used for secure multi-party computing.

Keywords: Negative database, Authenticated, Intersection protocol, Privacy preservation, Data security

1 Introduction

In recent years, with the development of network technologies and the arrival of the era of big data, information security and data security have received more and more attention. Secure Multiparty Computation (SMC) is a collaborative computing problem that researches a group of non-trusted participants on the premise of protecting private information. It is also one of the research hotspot in the international cryptography community. This problem first proposed by A. C. Yao [1] in 1982 called millionaires' protocol that can be described as: Two millionaires Alice and Bob want to know who they are wealthier, but they do not want to let the other party knows any information about their wealth. This is the millionaire problem. In Secure Multiparty Computing (SMC), p participants calculate $y = f(x_1,...,x_p)$, each of which participant p_i only provides x_i, and ultimately cannot get results other than y. In 2014, Jing et al. [20] proposed a privacy-preserving data mining models based on security multiparty computation. After many scholars have conducted in-depth research, many theoretical results have been achieved [2-3, 21]. R Cramer et al. proposed general secure multiparty computation from any linear secret-sharing scheme in 2000.

Privacy-preserving computational geometry (PPCG) is an important research direction of SMC. And the protection of private-preserving set intersection (PPSI) is a classic problem of PPCG, which is in the commercial and military field. There are important application prospects in other fields. For example, when two or more confidential databases are in the process of sharing, their owners hope to collaboratively calculate the intersection without exposing their own private data sets. This problem can be mathematically modeled as Alice and Bob own secret data sets X and Y, respectively, and both parties can calculate $X \cap Y$ without leaking their own data. This problem can also be extended to multiple parties, when three or more clients hope to collaboratively calculate the intersection without exposing their own private data sets. This article will firstly design a three-party authentication intersection protocol that includes two clients and one server; secondly design a multiparty authentication intersection protocol that includes n-1 clients and one server.

For solving the set intersection problem, the traditional protocol [4] has proposed a scheme based on collective polynomials and homomorphic encryption, the article [5] based on a threshold scheme and homomorphic encryption, and the article [6] based on scalar product protocol. However, in our protocol, we use negative database to achieve this goal, which is, calculating the intersection of two or more databases without revealing private information.

Our proposed protocol is based on the Negative database (NDB) [7-9]. The negative database is a new

*Corresponding Author: Hongfeng Zhu; E-mail: zhuhongfeng1978@163.com
DOI: 10.3966/160792642020112106008

type of privacy protection technology, which refers to the positive database can be negative, that is, according to certain algorithms the positive database can be changed into its complement. And then the algorithms compress the complement. And after the negative database compression process, the size of the negative database can be almost the same as that of the original positive database. In the intersection operation, the parties involving in the intersection operation can obtain the intersection result information without leaking other information of their own database. Negative database is a practical application of negative representation, which [10] is a new way of representing data first proposed by Esponda et al. [10-12] in 2004. This paper aims at enhancing the security of data by storing the compressed form of the complement of the positive database. In [10, 13], Esponda et al. has proved the fact that negative database is NP-hard to recover the original data from negative database. At present, some negative database generation algorithms have been proposed. On the one hand to make the data safer during the transmission; on the other hand, to carry out security authentications [14-17, 23], data can be hidden in this way. In the research, we found that the data hiding nature of negative databases can be used for secure multiparty computations to protect users' data security.

The advantages of our protocol are as follows:

- As an intermediary the server can complete the authentication and use negative database technology to encrypt the positive database, in this way, we can complete the process of authentication and intersection in one time, effectively reducing delivery times.
- Our protocol can not only achieve the goal of intersection set based on negative database, but also that of union, Cartesian product, etc.
- The original database content will not be exposed in the protocol.

The remainder of this paper is organized as follows: Section 2 is about related work. Section 3 outlines our proposed first three-party protocol. Section 4 and Section 5 conduct a security analysis and a performance analysis of our proposed protocol. Section 6 outlines our proposed second multiparty protocol. Section 7 concludes this paper.

2 Related Work

2.1 Negative Database

As [12], assuming the original data is a database containing nx records, that is, $DB = \{x_1, x_2, ..., x_{nx}\}$. Each record in the DB is a binary string of length m. The complete set is expressed as $U = \{0,1\}^m$, and the complement set of DB as $U - DB$. NDB only stores the record of $U - DB$, $U - DB$ is often much larger than DB. Usually, NDB needs to cover a large number of binary string, which is difficult to accurately represent and store one by one. "*" is defined as a "do not care" notation to compress the NDB. The symbol "*" can represent "1" and "0". Given a string defined upon the alphabet $\{0,1,*\}$, the specified positions are the positions with value "0" or "1", and the unspecified positions are the positions with "*". An example of the NDB is given as follows in Table 1.

Table 1. Negative database instance

DB	$U - DB$	NDB_1	NDB_2
0000	0001	**01	1*0*
0010	0100	11**	*10*
0011	0101	010*	*1*1
0110	0111	100*	111*
1010	1000		
1011	1001		
	1100		
	1101		
	1110		
	1111		

From the table we can see that the size of NDB can be much smaller than that of *U-DB*, even smaller than that of DB, and that one *U-DB* can generate many different NDBs. If an NDB can be reversed to obtain the DB in polynomial time, the NDB is said to be easy-to-reverse, otherwise, it is hard-to-reverse. The relationship between DB, NDB and *U-DB-NDB* is shown in Figure 1.

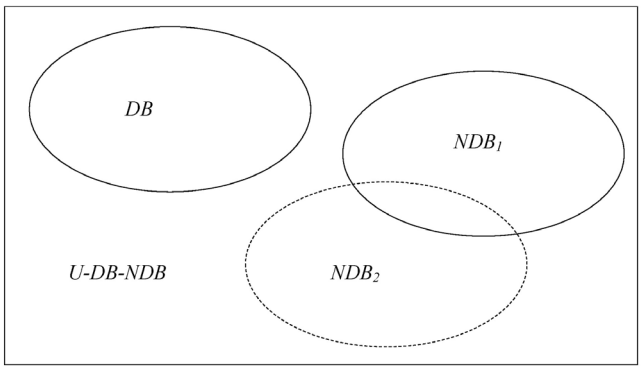


Figure 1. The relationship between DB, NDB and U-DB-NDB

2.2 Morph Operation of Negative Database

Morph operation of negative database means the process of converting a negative database to another negative database, the two negative databases correspond to the same positive database. Morph operations can be performed repeatedly, these negative databases obtained through variant operations are all equivalent. When the positive database is unknown, it is extremely difficult to determine whether the negative databases are equivalent or not. Morph operation of negative database was proposed by Esponda and

colleagues in the article [11, 18]. In addition, the morph operation used in this article need to satisfy the following properties:

In our protocol, we will use a negative database NDB1 and a random number r to complete XOR operation $NDB_2 = NDB_1 \oplus r$, then to complete $NDB_3 = morph(NDB_2)$. NDB2 is equivalent to NDB3, but according to NDB2 and NDB3 cannot calculate the random number r. We can improve security by using morph operation many times.

2.3 A Relational Algebra for Negative Databases

Relational algebra for negative databases was proposed by Esponda et al. in [19]. This article defined a series of operations on sets that correspond to the well-known relational algebra operators like Intersection, Union, Cartesian Product, and etc. Each relational algebra operation takes two or more negative databases as input, and according to the algorithm, finally a result represented by negative database was output. The notations and definitions are defined as follow:

- x, y, z : strings;
- $x[i,...,j]$: positions $i,...,j$ in string x;
- U_n : the universe set of string of length n;
- Ω_n : an ordered list of n positions of a string of length n;
- Υ_1 and Υ_2 : ordered lists of string positions for strings of length n and m, respectively;
- DB_1 and DB_2 : positive database with length n and m, respectively;
- NDB_1 and NDB_2 : negative database of DB_1 and DB_2.

Definition 2.1 Match $x \,\mathrm{M}\, y$: Two strings, x and y match iff $\forall i((x[i] = y[i]) \vee (x[i] = *) \vee (y[i] = *))$.

Definition 2.2 Coalesce $x \odot y$: Two strings x and y of length n coalesce into string z iff x matches y and for all $1 \le i \le n$:

$$z[i] = \begin{cases} x[i], if\,(x[i] = y[i]) \vee (y[i] = *) \\ y[i], if\,(x[i] = *) \end{cases}$$

Negative intersection, $\overline{\cap}$, can be defined as:

$$NDB_1 \overline{\cap} NDB_2 = \{x : x \in NDB_1\} \cup \{y : y \in NDB_2\}$$

Negative Union, $\overline{\cup}$, can be defined as:

$$NDB_1 \overline{\cup} NDB_2 = \{z : z = x \odot y, xMy, (x \in NDB_1 \wedge y \in NDB_2)\}$$

Specific algorithms and other relational algebraic operations please refer to [19].

3 Secure Three-party Authentication Intersection Protocol Based on Negative Database

In this section, we outline our proposed secure three-party authentication intersection protocol based on negative database. In our protocol, there are three parties with two clients and one server. We use the server as a medium, with the help of which the two clients complete the authentication process, at the same time transfer their own negative database to each other.

We first give the problem description, then introduce the symbols used in the protocol, finally describe the overall protocol process. This protocol comprises two phases: registration phase, authentication phase.

3.1 Problem Description

Definition of our three-party intersection is as follows:

Definition 3.1. Three-party Intersection Problem: Assuming three participants are two clients, and one server. The two clients' private sets are $X = \{x_1,...,x_n\}$ and $Y = \{y_1,...,y_n\}$. Both participants want to know the intersection of X and Y, i.e. $Z = X \cap Y$: $\{x_i \mid \exists k, x_i = y_k, 1 \le i \le nx, 1 \le k \le ny\}$, but they are not sure of each other true identity, and do not want to reveal any additional information about their private sets. Therefore, there must be a server to provide authentication and information transfer.

3.2 Notations

The notations used throughout this paper are defined in Table 2.

Table 2. Notations

Symbol	Description
id_A, id_B	Client's ID
ID_S	Server's ID
pw_A, pw_B	Client's password
m, n	Private key generted by key generation center
M, N	Public key of the clients $M = g^m$, $N = g^n$.
r	Private key of the server
R	Public key of the server $R = g^r$
NDB	Negative databse generated by clients
$morph(\cdot)$	Morph operation to NDB
a_1, a_2, b_1, b_2	Random number the clients select
$H_{i(i=0,1,2,3)}(\cdot)$	$H_{i(i=0,1,2)}(\cdot)$ are hash functions modeled as random oracles
$H : \{0,1\}^* \to \{0,1\}^{\lambda_i}$	One-way hash function
$Enc_k(\cdot)$	Symmetric encryption with key k
$Dec_k(\cdot)$	Symmetric decryption with key k

3.3 The all-over Process of Our Protocol

The whole process will be described below, and can be seen through Figure 2 and Figure 3.

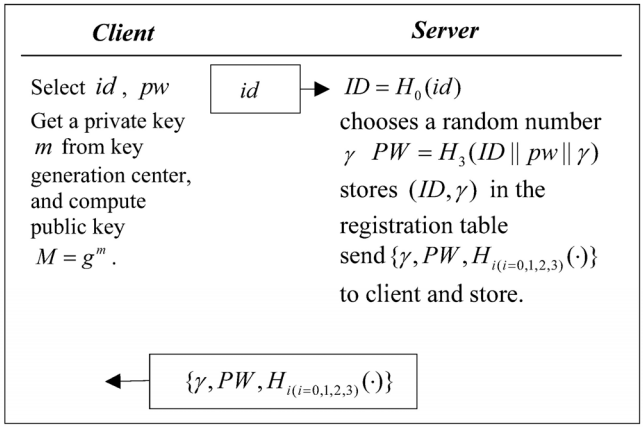


Figure 2. Registration phase

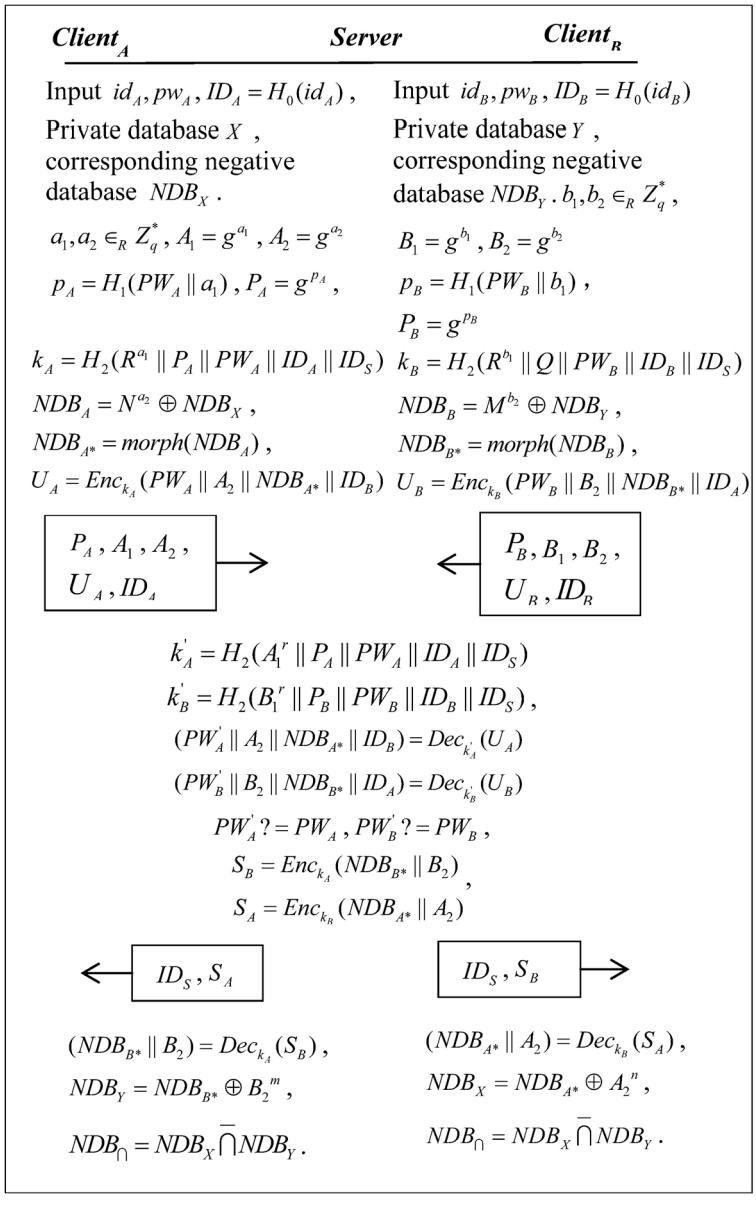


Figure 3. Three-party protocol's login, authentication and intersecion phase

3.3.1 Registration Phase

Our protocol is defined over a finite cyclic group G of prime order q with g as a generator. Hash functions $\{0,1\}^* \to \{0,1\}^{\lambda_i}$ are denoted by H_i, where $i \in \{0,1,2,3\}$ and λ_i is a security parameter of H_i. When a client wants to register his/her ID to server, he/she will do following operations:

(1) The client chooses his/her id, pw and sends these to the server.

(2) After receiving the registration request message from the client, the server first chooses a random number γ, computes $ID=H_0(id)$ and $PW=H_3(ID\| pw\| \gamma)$. Then stores (ID,γ) in the registration table. At the same time, sends $\{\gamma, PW, H_{i(i=0,1,2,3)}(\cdot)\}$ in the to the client. The client stores these messages.

(3) After receiving the information from the server, the key generation center will send a private key m to the client and according to the private key compute public key $M = g^m$.

3.3.2 Login, Authentication and Intersection Phase

(1) The client$_A$ first inputs his/her id_A, pw_A, id_B, pw_B, and opens a session with the server. The client$_A$ computes $ID_A = H_0(id_A)$, Client$_B$ computes $ID_B = H_0(id_B)$.

(2) Client$_A$ computes negative database NDB_X according to his/her private database X with the help of negative database generation algorithm.

(3) Client$_A$ chooses two random number a_1, a_2 from ${}_R Z_q^*$, computes $A_1 = g^{a_1}$, $A_2 = g^{a_2}$, $p_A = H_1(PW_A \| a_1)$, $P_A = g^{p_A}$, $k_A = H_2(R^{a_1} \| P_A \| PW_A \| ID_A \| ID_S)$.

(4) Client$_A$ encrypts the negative database NDB_X with A_2, $NDB_A = N^{a_2} \oplus NDB_X$, then perform morph operation on NDB_A, $NDB_{A^*} = morph(NDB_A)$. Finally, computes $U_A = Enc_{k_A}(PW_A \| A_2 \| NDB_{A^*} \| ID_B)$. Sends $\{P_A, A_1, A_2, U_A, ID_A\}$ to the server. The client$_B$ does the same step as client$_A$ but uses different parameters. Client$_B$ finally sends $\{P_B, B_1, B_2, U_B, ID_B\}$ to the server. On receiving the messages from the clients, the server calculates $k_A^{'} = H_2(A_1^r \| P_A \| PW_A \| ID_A \| ID_S)$ and $k_B^{'} = H_2(B_1^r \| P_B \| PW_B \| ID_B \| ID_S)$ with its private key. Then the server calculates $(PW_A^{'} \| A_2 \| NDB_{A^*} \| ID_B) = Dec_{k_A^{'}}(U_A)$ and $(PW_B^{'} \| B_2 \| NDB_{B^*} \| ID_A) = Dec_{k_B^{'}}(U_B)$, checks $PW_A^{'} ? = PW_A$, $PW_B^{'} ? = PW_B$, if true, then encrypts NDBs by k_A and k_B, $S_B = Enc_{k_B}(NDB_{A^*} \| A_2)$, $S_A = Enc_{k_A}(NDB_{B^*} \| B_2)$. Finally sends $\{ID_S, S_A\}$ to client$_A$, sends $\{ID_S, S_B\}$ to client$_B$. On receiving the messages from the server, client$_A$ computes NDB_{B^*} and B_2 by $Dec_{k_A}(S_A)$, gets NDB_Y by computes $NDB_{B^*} \oplus B_2{}^m$. Finally, $NDB_\cap = NDB_X \bigcap NDB_Y$. The client$_B$ do the same step as client$_A$ but uses different parameters.

4 Secure Multiparty Authentication Intersection Protocol Based on Negative Database

This section briefly describes the issue of secure multiparty intersections, and assumes that there are M participants, $P_1, P_2, ..., P_M$, their private set is $X_1, X_2, ..., X_M$ respectively. The M participants want to know the intersection result between their private sets, but none of them want to reveal any additional information about their private collection sets.

The secure three-party intersection protocol based on negative database can be easily extended to a secure multi-party intersection protocol. The security and efficiency analysis of the secure multiparty intersection protocol based on the negative database is similar to that under the three-party conditions. The following is a detailed process of a secure multiparty intersection protocol based on a negative database.

Registration phase:

For each participant, the protocol performs the same steps as Figure 2.

Login, authentication and intersection phase

As shown in Figure 4:

(1) The client$_A$ first inputs his/her id_A, pw_A, id_B, pw_B, id_C, pw_C, and opens a session with the server. The client$_A$ computes $ID_A = H_0(id_A)$, client$_B$ computes $ID_B = H_0(id_B)$. Second, client$_A$ computes negative database NDB_X according to his/her private database X with the help of negative database generation algorithm. Third, client$_A$ chooses two random number a_1, a_2 from ${}_R Z_q^*$, computes $A_1 = g^{a_1}$, $A_2 = g^{a_2}$, $p_A = H_1(PW_A \| a_1)$, $P_A = g^{p_A}$, $k_A = H_2(R^{a_1} \| P_A \| PW_A \| ID_A \| ID_S)$. Fourth, encrypts the negative database NDB_X with A_2, $NDB_A = N^{a_2} \oplus NDB_X$. Finally, computes $U_A = Enc_{k_A}(PW_A \| A_2 \| NDB_A \| ID_B \| ID_C)$. That is, Client$_A$ will perform intersection operation with client$_B$ and client$_C$. Sends $\{P_A, A_1, A_2, U_A, ID_A\}$ to the server. (For simplicity, the morph operation is omitted here.)

The others do the same step as client$_A$ but use different parameters.

(2) On receiving the messages from the clients, the server computes $k_A^{'} = H_2(A_1^r \| P_A \| PW_A \| ID_A \| ID_S)$ with its private key. Then computes $(PW_A^{'} \| A_2 \| NDB_A \| ID_B) = Dec_{k_A^{'}}(U_A)$, checks $PW_A^{'} ? = PW_A$, if true, continue.

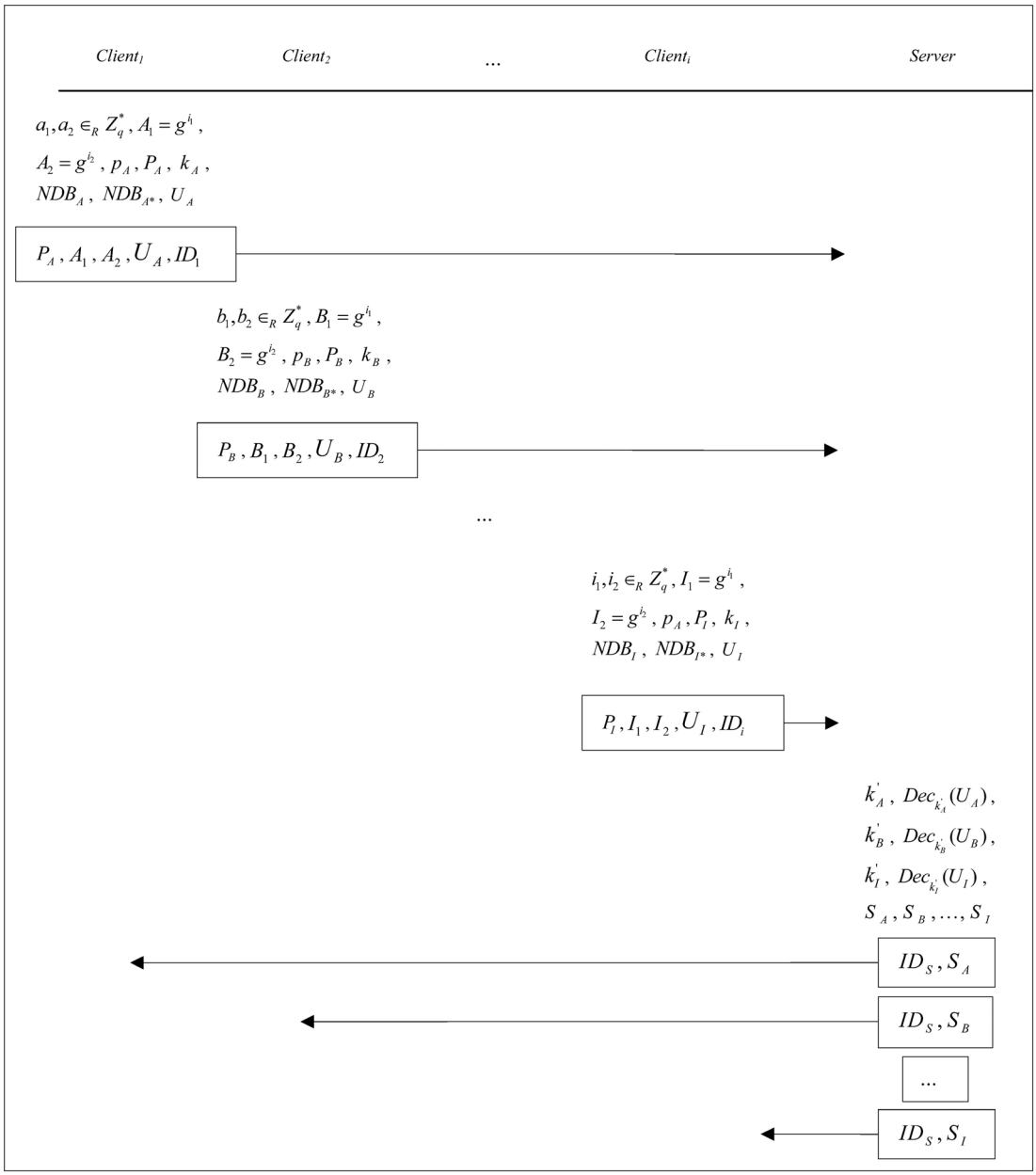


Figure 4. Multi-party protocol's login, authentication and intersecion phase

Then encrypt NDBs by k_A, $S_A = Enc_{k_A}(NDB_B \| NDB_C \| B_2 \| C_2)$. That is, includes the negative database information of other clients except the sender's own negative database. Finally sends $\{ID_S, S_A\}$ to client$_A$, sends $\{ID_S, S_B\}$ to client$_B$. $\{ID_S, S_C\}$ to client$_C$.

(3) On receiving the messages from the server, client$_A$ computes $(NDB_B \| NDB_C \| B_2 \| C_2)$ by $Dec_{k_A}(S_A)$, gets NDB_Y by computes $NDB_B \oplus B_2^m$, also can get NDB_Z by $NDB_C \oplus C_2^m$. Finally, we get $NDB_\cap = NDB_X \overline{\cap} NDB_Y \overline{\cap} NDB_Z$. The others do the same step as client$_A$ but uses different parameters.

As shown in Figure 5, when the number of clients is 3 (set $i = 3$), perform the following operation.

5 Security Analysis

In this section, we will prove that our protocol is secure under semi-honest adversary model and malicious adversary model.

5.1 Security under Semi-honest Adversary Model

In this paper we assume that the participants in the protocol are semi-honest, which means the participants can execute the agreement strictly and correctly. They will not exit the agreement or maliciously input false information; but they will keep their own calculation records, and it is possible to use these records to

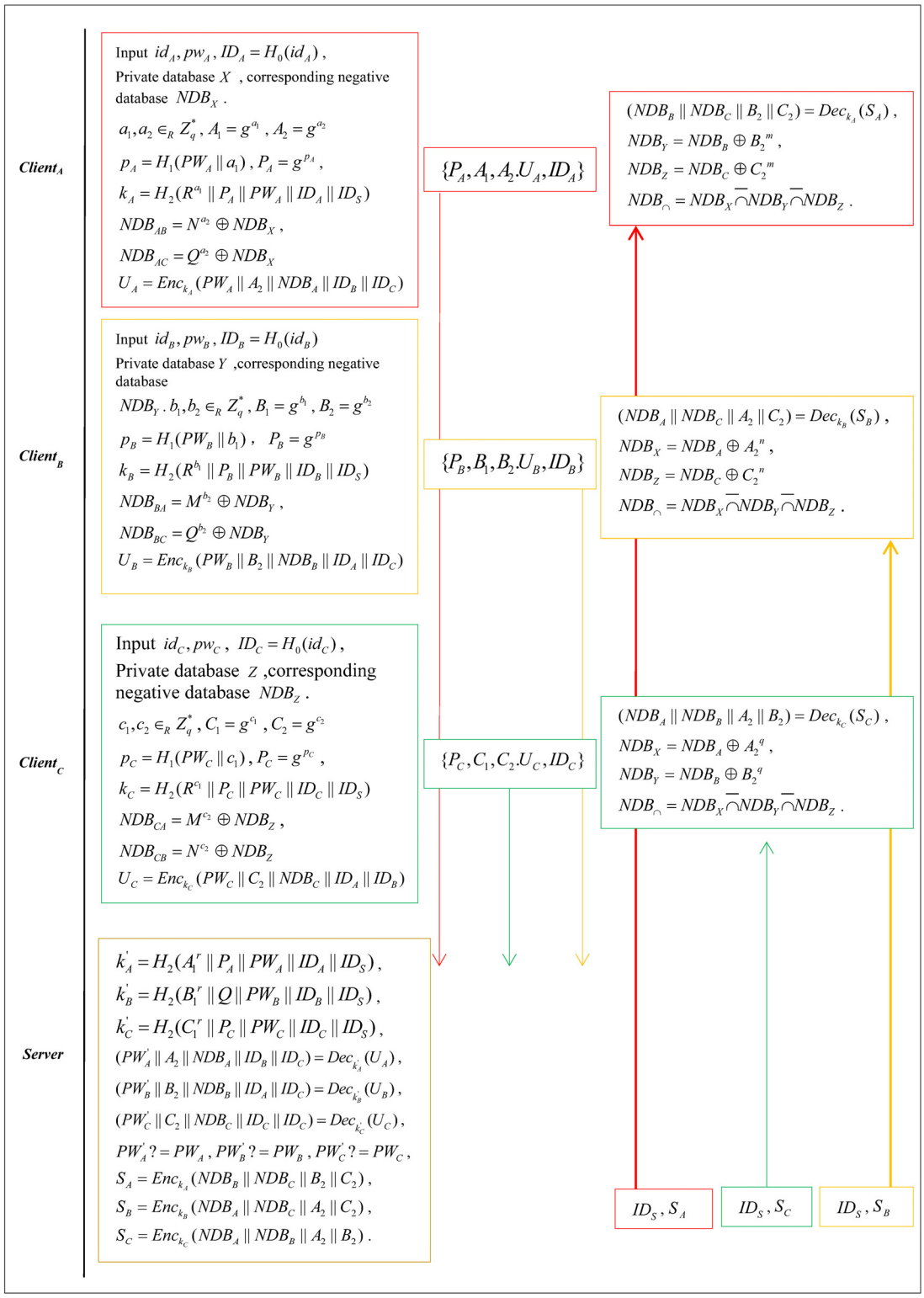


Figure 5. Multi-party protocol's login, authentication and intersecion phase (set i=3) Theorem 4.1 (Correctness)

extrapolate input from other participants. If the protocol is safe for semi-honest participants, it is safe in the semi-honest model. The security analysis of our secure three-party authentication intersection protocol based on the negative database under the semi-honest adversary model is as follows:

Lemma 4.1 A negative database NDB_1 is generated by a given positive database DB, and a random number a. First let $A = g^a$, then compute $NDB_2 = morph(NDB_1 \oplus A)$, thus it is difficult to recover a according to NDB_1 and NDB_2.

Proof. First, A computed by a, recovering a from A is difficult. Then two arbitrary strings x and y chosen from NDB_1 and NDB_2, and the select of number a is random, so it is difficult to recover the string A from the positions marked as *. The only thing one has to consider is the bits that have a certain value. So if the adversary wants to recover the information about A,

he/she has to compute every string in NDB_1 and NDB_2. However, according to the morph operation algorithm, the position with a certain value in $NDB_1 \oplus A$ and NDB_2, nearly half of the values are different, the other half of the values are same. As a result, it is difficult to recover a according to NDB_1 and NDB_2.

The secure three-party authentication intersection protocol based on negative database gets the correct intersection result.

Proof. In our protocol, three phases (login, authentication and compute intersection) are carried out together. The first login phase is done locally. And the client calculates some parameters. NDB_A and NDB_B are negative databases for $N^{a_2} \oplus NDB_X$ and $M^{b_2} \oplus NDB_Y$. The client performs morph operation on NDB_A and NDB_B and encrypts them locally. In the authentication phase, the server verifies the authenticity of the user's identity, and reorganizes and encrypts information. We explained in the previous proof recovering the random number is difficult, in addition, the server cannot know the private key of clients, so it cannot get the original negative database. In the last phase, each client gets the original negative database by his/her own private key and finally gets the intersection database between the two parties.

Theorem 4.2 (Can protect client's privacy). In the secure three-party authentication intersection protocol based on the negative database, if the negative database is difficult to reverse, the server cannot get any clients' negative database information.

Proof. In our protocol, although the server receives a negative database from the client, the server cannot reserve corresponding positive database for the negative database is difficult to reverse. In addition, because NDB_A is under the protection of the random number, the server cannot get extra information about the database X. And the server cannot compute intersection directly for the negative databases are encrypted by clients themselves locally, thus the negative database information is safe on the server side. Besides, the negative database is difficult to reverse, and the others also cannot get real information in a positive database. As a result, our protocol can protect client's privacy.

5.2 Security under Malicious Adversary Model

This section will consider the following scenario under the Malicious Adversary model: A malicious participant may enter arbitrary data in anticipation of recovering additional information about the other's private set.

Theorem 4.3 In the secure three-party authentication intersection protocol based on negative database, if the negative database is difficult to reverse, the probability that each participant eventually recovers additional information about the private set of other participants by entering specific data is low.

Proof. In our protocol, the server only plays the role of identity authentication and information transmission. If the information is tampered with, it doesn't make any sense. The above theorems have also proved that the server cannot get the user's real information, so from the server's perspective, the protocol is secure. The two clients use the server as the medium, which only sends and receives information once. In this case there is no interaction with other information in the server. Moreover, one client can only get correct number a by guessing and cracking morph operation, thus the probability of success is low.

Therefore, it is necessary to consider the situation only when one client tricks the other client's database Y 's data by entering a specific set X. In this case, client may generate a negative database corresponding to a large positive database, and this positive database may cover additional elements in Y. However, if this positive database is not enumerable, the client cannot check whether the elements in X belong to the intersection result or not in the final stage. Therefore, the probability that client will eventually get a clear intersection result is extremely low, so the probability that client gets extra information about Y is also low.

6 Performance Analysis

6.1 Three-party Protocol's Space and Time Complexity

Our three-party protocol's computational cost will be shown in Table 3.

Table 3. Computational cost of our three-party protocol

	Registration phase	Login, authentication and intersecion phase	Total
User		$12T_e + 4T_h + 2T_{NDB} + 4T_{XOR} + 2T_{NDB_\cap}$	$12T_e + 4T_h + 2T_{NDB} + 4T_{XOR} + 2T_{NDB_\cap}$
Server	$2T_h$	$2T_e + 2T_h$	$2T_e + 4T_h$
Total	$2T_h$	$14T_e + 6T_h + 2T_{NDB} + 4T_{XOR} + 2T_{NDB_\cap}$	$14T_e + 8T_h + 2T_{NDB} + 4T_{XOR} + 2T_{NDB_\cap}$

In our proposed protocol, the server only needs to store client's ID and the password PW in registration table, and it does not have client's real password pw. If we assume that l is the length of the password, and

there are m clients have registered successfully, the space complexity is $O(l*m)$.

In the registration phase, the server needs to choose a random number γ, and the PW needs to be generated by a one-way hash function. The time complexity for concatenating the ID, PW and the random number γ is $O(l)$. Supposing the time complexity of the hash function is T_h, the time complexity of the registration phase is $O(l)+T_h$. In the login phase, we used public key encryption system to complete encryption of NDB, which involves the exponentiation. We used one-way hash function to verify the authenticity of the identity. Supposing the time complexity for generating an NDB is T_{NDB}, the time complexity of XOR operation is T_{XOR}, and the time complexity of the public key encryption system is T_e. Therefore, the time complexity of the authentication phase is $14T_e+6T_h+2T_{NDB}+4T_{XOR}$. If SHA-512 is selected as the one-way hash function, then $T_h=O(l)$. If the NDB generated by the K-hidden algorithm [22], the $T_{NDB}=O(l*r*K)$, and if K and r are small, the time complexity of T_{NDB} will be $O(l)$. Consequently, if SHA-512 and the K-hidden algorithm are adopted, then the time complexity of the login and authentication phase will reduce to $T_e+O(l)+T_{XOR}$. Finally, the time complexity will add $2T_{NDB_\cap}$.

6.2 Multiparty Protocol's Space and Time Complexity

Our multiparty protocol's computational cost will be shown in Table 4.

Table 4. Computational cost of our multiparty protocol

	Registration phase	Login, authentication and intersecion phase	Total
User		$(2m^2+2m)T_e+2mT_h+mT_{NDB}+(2m^2-2m)T_{XOR}$ $+2mT_{NDB_\cap}$	$(2m^2+2m)T_e+2mT_h+mT_{NDB}+(2m^2-2m)T_{XOR}$ $+2mT_{NDB_\cap}$
Server	$2mT_h$	mT_e+mT_h	mT_e+3mT_h
Total	$2mT_h$	$(2m^2+3m)T_e+3mT_h+mT_{NDB}+(2m^2-2m)T_{XOR}$ $+2mT_{NDB_\cap}$	$(2m^2+3m)T_e+3mT_h+mT_{NDB}+(2m^2-2m)T_{XOR}$ $+2mT_{NDB_\cap}$

Note. In table 3, m represents the number of clients.

In the registration phase and login phase, the client does the same process as three-party. So if the length of the password is l, and m-clients have registered successfully, the space complexity in this phase is $O(l*m)$, the time complexity is $O(l)+T_h$. Unlike the three-party protocol, the multiparty protocol requires more calculations, and needs to calculate a separate encrypted negative database for each participating client, for example, cilent$_A$ needs to compute $NDB_{AB}=N^{a_2}\oplus NDB_X$ and $NDB_{BC}=Q^{b_2}\oplus NDB_Y$. The more users that participate in the protocol, the greater the amount of calculation required. We assume the number of clients is m, then the time complexity in this phase is $(2m^2+3m)T_e+3mT_h+mT_{NDB}+(2m^2-2m)T_{XOR}$.

Finally, the time complexity will add $2mT_{NDB_\cap}$. Our protocol still needs to be improved, so does the time complexity.

At present, there are some efficient negative database generation algorithms that improve the operating efficiency of the protocol. However, there is still much room for improvement in this algorithm. If the operation steps in the algorithm can be simplified, for example, removing the public key and the private key, the security of the algorithm can be improved or equivalent. We will continue to study and improve this.

7 Conclusion

This paper we first proposed a secure three-party authentication intersection protocol based on the negative database, and analyzed the security and efficiency of the protocol, then expanded it to multiparty. But it still has some room for improvement, and there is still some work need to be carried out, for example, simplifying the calculation steps. The use of negative databases for building secure multiparty computing protocols is a new and promising research direction. In the future work, we will study how to design more efficient variant operations and negative database generation algorithms. We will also study how to use negative databases for more security multiparty computing problems. In our protocol, intersection is just an example, and by this protocol we also can compute union, cartesian product and so on.

Acknowledgements

This work was supported by Liaoning Provincial Natural Science Foundation of China (Grant No. 2019-MS-286), and Basic Scientific Research Project of

Liaoning Provincial Department of Education (Grant No. LJC202007).

References

[1] A. C. Yao, Protocols for Secure Computations, *23rd IEEE Symposium on Foundations of Computer Science*, Chicago, Illinois, 1982, pp. 160-164.

[2] K. B. Frikken, *Secure Multi-Party Computation, Algorithms and Theory of Computation Handbook*, Chapman & Hall/CRC, 2010.

[3] L. Kamm, *Privacy-preserving Statistical Analysis Using Secure Multi-party Computation*, Ph.D. Thesis, University of Tartu, Tartu, Estonia, 2015.

[4] M. J. Freedman, K. Nissim, B. Pinkas, Efficient Private Matching and Set Intersection, *23rd Annual Eurocrypt Conference*, Interlaken, Switzerland, 2004, pp. 1-19.

[5] Q.-S. Ye, H.-X. Wang, C. Tartary, Privacy-Preserving Distributed Set Intersection, *Third International Conference on Availability, Reliability and Security*, Barcelona, Spain, 2008, pp. 1332-1339.

[6] Y.-F. Sun, H. Zhong, F.-F. Yan, H.-S. Huang, Protocol for Privacy-preserving Set Intersection, *Journal of Computer Applications*, Vol. 30, No. 2, pp. 506-509, February, 2010.

[7] D.-D. Zhao, W.-J. Luo, A Study of the Private Set Intersection Protocol Based on Negative Databases, *IEEE International Conference on Dependable, Autonomic and Secure Computing*, Chengdu, China, 2013, pp. 58-64.

[8] D.-D. Zhao, W.-J. Luo, R. Liu, L.-H. Yue, Experimental Analyses of the K-hidden Algorithm, *Engineering Applications of Artificial Intelligence*, Vol. 62, pp. 331-340, June, 2017.

[9] D.-D. Zhao, W.-J. Luo, R. Liu, L.-H. Yue, A Fine-grained Algorithm for Generating Hard-to-reverse Negative Databases, *2015 International Workshop on Artificial Immune Systems (AIS-2015)*, Taormina, Italy, 2015, pp. 84-91.

[10] F. Esponda, *Negative Representations of Information*, Ph.D. Thesis, The University of New Mexico, Albuquerque, New Mexico, 2005.

[11] F. Esponda, E. S. Ackley, S. Forrest, P. Helman, Online Negative Databases, *International Conference on Artificial Immune Systems*, Catania, Sicily, Italy, 2004, pp. 175-188.

[12] F. Esponda, S. Forrest, P. Helman, *Enhancing Privacy through Negative Representations of Data*, University of New Mexico TR-CS-2004-18, January, 2004.

[13] F. Esponda, E. S. Ackley, P. Helman, H. Jia, S. Forrest, Protecting Data Privacy through Hard-to-Reverse Negative Databases, *International Journal of Information Security*, Vol. 6, No. 6, pp. 403-415, October, 2007.

[14] M. M. Groat, B. Edwards, J. Horey, W. He, S. Forrest, Application and Analysis of Multidimensional Negative Surveys in Participatory Sensing Applications, *Pervasive and Mobile Computing*, Vol. 9, No. 3, pp. 372-391, June, 2013.

[15] Y. Bao, W. Luo, X. Zhang, Estimating Positive Surveys from Negative Surveys, *Statistics & Probability Letters*, Vol. 83, No. 2, pp. 551-558, February, 2013.

[16] Y. Bao, W. Luo, Y. Lu, On the Dependable Level of the Negative Survey, *Statistics & Probability Letters*, Vol. 89, pp. 31-40, June, 2014.

[17] Y. Lu, W. Luo, D. Zhao, Fast Searching Optimal Negative Surveys, *Proceeding of the 2014 International Conference of Information and Network Security (ICINS)*, Beijing, China, 2014, pp. 82-90.

[18] F. Esponda, Everything That is Not Important: Negative Databases, *IEEE Computational Intelligence Magazine*, Vol. 3, No. 2, pp. 60-63, May, 2008.

[19] F. Esponda, E. D. Trias, E. S. Ackley, S. Forrest, *A Relational Algebra for Negative Databases*, University of New Mexico Technical Report, November, 2007.

[20] Z.-J. Jing, G.-P. Jiang, C.-S. Gu, Feasibility Analysis of Privacy-preserving Data Mining Models Based on security Multiparty Computation, *Application Research of Computers*, Vol. 31, No. 2, pp. 543-546, February, 2014.

[21] R. Cramer, I. Damgard, U. Maurer, General Secure Multi-Party Computation from Any Linear Secret-Sharing Scheme, *International Conference on Theory & Applications of Cryptographic Techniques*, Bruges, Belgium, 2000, pp. 316-334.

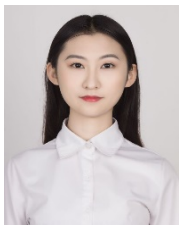
[22] D.-D. Zhao, *Research on Several Applications of the Negative Representation of Information*, Ph.D. Thesis, University of Science and Technology of China, Hefei, China, 2016.

[23] H.-J. Wang, H. Zhang, J.-X. Li, C. Xu, A(3,3) Visual Cryptography Scheme for Authentication, *Journal of Shenyang Normal University*, Vol. 31, No. 3, pp. 397-400, July, 2013.

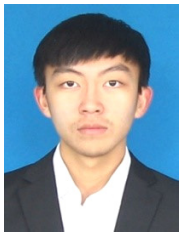
Biographies

Hongfeng Zhu obtained his Ph.D. degree in Information Science and Engineering from Northeastern University. Hongfeng Zhu is a full professor of the software college at Shenyang Normal University. He is also a master's supervisor. He has research interests in wireless networks, mobile computing, cloud computing, social networks, network security and quantum cryptography. Dr. Zhu had published more than 70 international journal and international conference papers on the above research fields.

Xueying Wang obtained her Ph.D. degree in Management Science and Engineering from Wuhan University. Xueying Wang is a Dean of the Kexin software college at Shenyang Normal University. She is also a full professor and a master's supervisor. She has research interests in cloud computing, social networks, network security and E-commerce. Dr. Wang had published more than 40 international journal papers on the above research fields.



Jingyue Zhao, a graduate student of Shenyang Normal University. She has research interests in cloud computing, social networks and quantum cryptography. After completing her studies, she enjoys reading the books related to this major. Under the guidance of the teacher, she has published two article on the above research fields.



Shuai Geng, an undergraduate from Shenyang Normal University. He has research interests in wireless networks, mobile computing, cloud computing, social networks, network security and quantum cryptography. In the four years of college, after completing her studies, he enjoys reading the book related to this major. Under the guidance of the teacher, he has published 3 international journal papers on the above research fields.

Liwei Wang, a postgraduate studying at Shenyang Normal University. She has research interests in wireless networks, mobile computing, cloud computing, social networks and quantum cryptography. After completing her studies, she enjoys reading the book related to this major. Under the guidance of the teacher, she has published one article in EI journals.