TELKOMNIKA Indonesian Journal of Electrical Engineering
Vol. 12, No. 10, October 2014, pp. 7523 ~ 7532
DOI: 10.11591/telkomnika.v12i8.5949

Study on Commitment Schemes of Secure Multi-party Computation

Xiaoqiang Guo, Yan Yan, Cuiling Luo, Shuai Zhang
College of Science, Hebei United University,
No.46 Xinhua West Street, Tangshan 063009, Hebei Province, China
Corresponding author, e-mail: guoxiaoqiang@heuu.edu.cn, guoxq2004@163.com

Abstract

The problem of secure multi-party computation (SMPC) is one of the most fundamental problems in information security. First, we introduce the basic concept of SMPC and four SMPC basic agreement: key distribution, oblivious transfer, bit commitment and zero knowledge proof. Secondly, we separately illustrate commitment schemes commitment transfer protocol, commitment sharing protocol and commitment multiplication protocol. Finally, we present unconditionally secure multi-party computation with a passive adversary, an active adversary, general adversary structures.

Keywords*: secure multi-party computation, information security, commitment scheme, verifiable secret sharing*

1. Introduction

Secure multi-party computation (SMPC)) is a subfield of cryptography. The goal of methods for secure multi-party computation is to enable parties to jointly compute a function over their inputs, while at the same time keeping these inputs private. For example, two millionaires can compute which one is richer, but without revealing their net worth. In fact, the example was initially suggested by Andrew C. Yao in 1982 [1]. And it was later named the millionaire problem.

The concept is important in the field of cryptography and is closely related to the idea of zero-knowledgeness. In general it refers to computational systems in which multiple parties wish to jointly compute some value based on individually held secret bits of information, but do not wish to reveal their secrets to one another in the process. For example, two individuals who each possess some secret information x and y, respectively may wish to jointly compute some function $f(x,y)$ without revealing any information about x and y other than what can be reasonably deduced by knowing the actual value of $f(x,y)$, where "reasonably deduced" is often interpreted as equivalent to computation within polynomial time. The primary motivation for studying methods of secure computation is to design systems that allow for maximum utility of information without compromising user privacy.

Secure computation was formally introduced by A. Yao in 1982 as secure two-party computation.

The millionaire problem and its solution gave way to a generalization to multi-party protocols [2]. In an MPC, a given number of participants $p_1, p_2, \cdots, p_n$ each have a private data, respectively $d_1, d_2, \cdots, d_n$. The participants want to compute the value of a public function F on N variables at the point $(d_1, d_2, \cdots, d_n)$. An MPC protocol is dubbed secure if no participant can learn more from the description of the public function and the result of the global calculation than what he or she can learn from his or her own entry under particular conditions depending on the model used.

Like many cryptographic protocols, the security of an MPC protocol can rely on different assumptions:

It can be computational (i.e. based on some mathematical problem, like factoring) or unconditional (usually with some probability of error which can be made arbitrarily small).

Received March 10, 2014; Revised July 30, 2014; Accepted August 20, 2014

The model in which the scheme is described might assume that participants use a synchronized network (a message sent at a "tick" always arrives at the next "tick"), that a secure and reliable broadcast channel exists, that a secure communication channel exists between every pair of participants (an adversary cannot read, modify or generate messages in the channel), etc.

The centrally controlled adversary considered can be passive (only allowed to read the data of a certain number of participants) or active (can corrupt the execution protocol or a certain number of participants).

An adversary can be static (chooses its victims before the start of the multi-party computation) or dynamic (can choose its victims during the course of execution of the multiparty computation). Attaining security against a dynamic adversary is often much harder than security against a static adversary.

An adversary can be defined as a threshold structure (meaning that it can corrupt or simply read the memory of a number of participants up to some threshold), or be defined as a more complex structure (it can affect certain predefined subsets of participants, modeling different possible collusions). These structures are commonly referred to as adversary structures. The opposite set consisting of the sets of honest parties that can still execute a computational task is related to the concept of access structures.

Unconditionally or information-theoretically SMPC is closely related to the problem of secret sharing, and more specifically verifiable secret sharing (VSS); many SMPC protocols that protect against active adversaries use VSS.

Performing a computation using MPC protocols is still order of magnitudes slower than performing the computation using a trusted third party. However, more and more efficient protocols for MPC have been proposed, and MPC can be now used as a practical solution to various real-life problems such as distributed voting, private bidding and auctions, sharing of signature or decryption functions, private information retrieval, etc. The first large-scale and practical application of multiparty computation took place in Denmark in January 2008, as described by Bogetoft et al. [3].

2. SMPC Underlying Protocol

In this section we mainly disscuss the following four SMPC basic agreement: key distribution, oblivious transfer, bit commitment and zero knowledge proof.

2.1. Key Distribution

For the people engaged in the field of seure multi-party computation and cryptography, key distribution is the most basic agreemen. Its main goal is to make discrete communijcations securely share string (usually set to binary bit string) to prepare for the future use of secret communication tasks. We know that, in the classic environment, the biggest enemy is not channel noise, but a potential eavesdropper. If the eavesdropper mastered both sides of the legitimate communication key, in subsequent communication, he can illegally eavesdropp secrets, forged identity and engage in other acts. In the classic environment, eavesdropping can not be avoided in a certain extent. However, in quantum environment, according to the uncertainty principle and no-cloning theorem of quantum, eavesdropping detection becomes fairly easy [4].

Public key cryptography is the basic of key distribution. S. Goldwasser and S. Micali put forward the first probabilistic pubic key encryption scheme[5], called Goldwasser-Micali(GM) scheme. GM scheme is additively homomorphic. Its security is based on Quadratic Residue assumption. ρ is the message extension rate of the encryption algorithm,and $\rho = \log_2 N$, where security parameter $N = pq$. The bit complexity of the cryptographic operation on the message m is $O(|m|(\log_2 N)^2)$, so GM encryption system is inefficient.

T. ElGamal proposed homomorphic probabilistic cryptosystem [6]. It is multiplicaivaly homomorphic to realize secure multiparty multiplication of the encrypted data. Its security is based on Decision Diffie-Hellman assumption. Its shortcoming is that message expansion rate of the encryption scheme on the finite field is tremendous, to achieve practical security, large

prime number p needs at least 300 decimal digits, its bits length $|p|$ at least 1024, and at least there should be a large prime factor of $p-1$.

J. Benaloh proposed homomorphic dense probabilistic encryption scheme [7]. It is extension of the GM program, similar to the GM program, also additively homomorphic. Its security is based on the High-degree Residuosity problem, message expansion rate ρ close to 1.

P. Paillier proposed homomorphic probabilistic public key encryption algorithm [8], which is additively homomorphic. Its security is based on the Decisional Composite Residuosity assumption, message expansion rate ρ is a constant.

2.2. Oblivious Transfer

The oblivious transfer means the recipient can get their want messages from the sender's secret message set but you can not get the other message, and the sender dose not know the recipient choose which messages. Oblivious transfer is an important concept in modern cryptography. Now it widely is used to build zero-knowledge proof, verified secret sharing protocol etc. The oblivious transfer and bit commitment together constitute the basis of secure multi-party computations. It is a hot spot of research in the field of information security. An interesting application called the secret all-or-none leak, refers to such a secret learning task: the owner of several secrets Alice would like to sell any of her secrets to Bob, Bob pay money to get a secret what he wants (that is he knew nothing about the other secret), and ask Alice can not know Bob purchase which secret.

The conception of oblivious transfer was proposed by Rabin in [9]. Even, Goldreich and Lempel had given $\binom{2}{1}$-OT in [10]. And Crepeau had proven the equivalance Even's $\binom{2}{1}$-OT and Rabin's OT in [11]. Then Brassard, Crepeau and Robert had given AN-DOS and GOT in [12,13]. Cachin had constructed UOT in [14].

2.3. Bit Commitment

Consider such a scene: Alice claimed that she has some predictive capability, but Bob is skeptical. In order to make Bob to convince her predictive ability, Alice decided to show her the predictive ability for the upcoming a soccer game. How to make Bob believe her predictive power? In classic environment, this problem can be easily solved. Before the game, Alice will predict the score to written on a small piece of paper, then hand to Bob. After the game, Bob contrast to the note on the score of the game to determine the predictive capability Alice really has claimed. The core of this example is Alice in something of a prior claiming assertion, afterwards, she could not deny the assertion. Without loss of generality Alice's assertion as one bit, such SMPC model is the bit commitment. The conception of bit commitment was proposed by M.Blum. Bit commitment scheme can be used to build up zero knowledge proof, verified secret sharing, coins throwing etc. A bit commitment scheme must meet the following properties:

Correct: if Alice and Bob all honestly executive agreement, then Bob will properly gain a bit Alice commitment in reveal stage.

Confidentiality: Bob cannot learn the bit in commitment stage.

Binding: in the end of commitment stage Bob can get the only bit in reveal stage.

The first model was proposed by A. C. Yao [15], even though Yao has not emphasized the generalitty of the model, but the model was evaluated as " really applies to any actual both secure computation ". Now we called Yao model. Hoi-Kwong Lo and H. E. Chau proposed LC model in [16]. They made two changes to Yao model. First, the initial state set to pure state from mixed state in Yao model. Second, in model Yao, in each round communication do two things: measurement and unitary transformation, but measuremen is deleted, only a unitary transformation in LC model. This simplification is helpful for the analysis of uncondional security existence or not. It greatly simplifies the certification process.

2.4. Zero Knowledge Proof

Many more complex secure multi-party computation tasks need zero knowledge proof, such as identity authentication, signature, etc. Consider a scene: Alice said to Bob " I know the

mathematical proof of the theorem". Bob expressed doubt again. Alice wanted Bob to believe that she did know the methods of proof of the theorem, but not let Bob know proof process. Say simply, zero knowledge proof purpose is that make verifier believe prover who really mastered this knowledge under the premise not leaking.

The earliest Goldwasser proposed the concept of zero knowledge proof [17]. After verifier participated in the process of zero knowledge proof, any information that can be calculated in polynomial time, also can be calculated independently by verifier in polynomial time, as long as he believes the authenticity of the proposition. The definition of zero knowledge proof systems mainly consider two different probability distribution:

a) Finished with proof of interaction, the probability distribution generated by the polynomial time verifier.

b) A probabilistic polynomial time automata generated the probability distribution based on the premise to be proven proposition correctness.

The resulting three different levels of zero knowledge proof systems:

a) Perfect zero knowledge: in this system in the above two distribution completely identical.

b) Calculation zero knowledge: in this system the two distributions in polynomial time indistinguishability, that is the two distributions can not be separated from the test of any probabilistic polynomial time.

c) Statistical zero knowledge: in this system the two distribution close to the statistical characteristics, namely the statistical difference between the two can be neglected.

3. Commitment Scheme

A commitment scheme, for an adversary structure, is a scheme that allows a player p_i to commit to a value a while keeping the value hidden in the presence of an δ-adversary and also binding p_i to the value in such a way that when he in a later stage decides to reveal the value, only the value a will be accepted among the other players.

For computationally SMPC, we can use a VSS scheme for unconditionally or perfectly information. We will use a commitment scheme devised by Cramer et al. in [18].

Commitment scheme:

a) COMMIT(s): Commitment allows a player to commit to a value s.

b) CTP(s, j): Commitment transfer protocol allows a player p_i to transfer a commitment of s to player p_j.

c) CSP(s): Commitment sharing protocol allows a player p_i to convert a commitment to s into a set of commitments on the shares of $s(s_1, s_2, \cdots, s_n)$. In other words, each player p_j will be committed to his share s_j.

d) CMP: Commitment multiplication protocol allows a player p_i who is committed to a, b and $c = ab$ to prove to the other players that c is indeed equal to $a \cdot b$.

e) OPEN(s): Open reveals a commitment, i.e., the value is revealed to all participants. Only the correct value, the one D commited to, is accepted by the honest players.

The commitment scheme is also homomorphic, that is given two commitments C_a and C_b each player can compute non-interactively the commitment C_{a+b} and C_{ab}.

In order for a dealer D to commit to a value s, he could simply share s among the players. This would work if we could guarentee that D was honest, but if D was corrupt, he could send inconsistent shares to the different players. To avoid this, we must force D to distribute consistent shares:

a) D chooses a symmetric matrix $R_{m\times m}$ at random and sets $R_{1,1}$ to s. For each row v_i belonging to p_i, D sends the vector $u_i = R^T_{v_i}$ to p_i. The first element of u_i is s_i, p_i's share in s.

b) p_i sends to each p_j the value $s_{ij} = \langle v_j, u_i \rangle$. p_j compares the received value with $\langle v_i, u_j \rangle$ and broadcasts the message fail (i, j) if they are not equal.

c) If a fail (i, j) is received, D broadcasts the value s_{ij}. If any of the players fail to agree that the value s_{ij} is correct, they broadcast an accusation that D is corrupt.

d) Say p_j accuses D of corruption. D can disprove the accusation by broadcasting the information sent to p_j in step i).

e) Each player checks the values broadcasted by D to see if they are consistent with the values they have received. If they are not he sends an accusation that D is corrupt. By the Q^2 property of the adversary structure, the protocol will only be rejected if at least one of the honest players sends an accusation. Likewise, the protocol will be accepted if all the accusing players are in δ.

Commitment Transfer Protocol: A commitment transfer protocol allows a player p_i, who has acommitment to s to transfer the commitment to a player p_j. If p_j and p_i are honest, the protocol leaks no information to the adversary. p_j learns the value s in the process.

1) p_i securely sends p_j all the information he used to create a commitment C to s. This includes s.

2) p_j creates a new commitment C' to s using the information received in step 1 and checks whether or not $C' - C = 0$.

If any of the above steps fail, p_i or p_j must be corrupt. To disprove his corruption, p_i can open s.

Commitment Sharing Protocol: A commitment sharing protocol is a protocol that allows a player p_i committed to a value s to secret share $s(s_1, s_2, \cdots, s_n)$ so that each player p_j is committed to the share s_j. To accomplish this, p_i, already committed to s, generates a random vector R of size $m-1$ and commits to each value in R. Using CTP, p_i transfers the commitment of s_j to p_j and hence p_j also learns s_j. Since s_j is a result of linear operation on the previously committed values, p_j can check that s_j is indeed a share of s.

Commitment Multiplication Protocol: A commitment multiplication protocol allows a player committed to the values a, b and $c = ab$ to convince the other players that $ab = c$. If the scheme is strongly multiplicative, the CMP will be perfectly secure. Otherwise it will have a negligible error probability ε.

CMP with negligible error:

1) p_i has already committed to the values a, b and $c = ab$. We'll denote those commitments C_a, C_b and C_c. In order to convince the other players that c is indeed equal to $a \cdot b$, p_i chooses a random β and creates the commitment C_β and another commitment for the value β', we will call it $C_{\beta'}$.

2) The other players generate a random challenge $r \in \{0,1\}$ using the appropriate protocols.

3) p_i opens the commitment $rC_a + C_\beta$, which reveals the value $r_1 = ra + \beta$, and he also opens a commitment to $r_1 C_b - C_{\beta'} - rC_c$, which should reveal 0 .

If p_i is honest, then all the opened values are either random or 0. If p_i can answer two different random challenges correctly, then $ab = c$ with an error probability of $\frac{1}{2}$. This probability can be reduced by iterating the process until the desired probability ε is reached [19].

CMP with zero error:

1) p_i has committed to three values a, b and c.

2) Using CSP, p_i creates and distributes shares of a, b and c. Each player p_j receives the shares a_j, b_j and $c_j = a_j b_j$ and is committed to them.

3) Since p_i is committed, we know that the shares of a, b and c are consistent and $f_a(0) = a$, $f_b(0) = b$ and $f_c(0) = c$ where $\deg(f_a) = t$, $\deg(f_b) = t$, and $\deg(f_c) = 2t$. Each player checks whether or not his shares compute $c_i = a_i b_i$ and broadcasts an accusation if this fails.

In order to have a CMP with zero error the secret sharing schemes must be strongly multiplicative. That is, $t < \frac{n}{3}$. That means that there are at least $n - t$ honest players in the scheme. And since $n - t > 2t$, where $2t$ is also the maximum degree of f_c, the honest players can always correctly reconstruct the polynomial if $c = ab$. If $c \neq ab$, at least one honest player, in addition to the corrupt players, would have to accuse p_i.

This protocol can also be generalised to work for any secret sharing scheme with a Q^3 adversary structure, provided that the scheme is realised with a strongly multiplicative MSP.

Open reveals a commitment. To open a commitment on s, the dealer D broadcasts s and all the shares of $s = \{s_1, s_2, \cdots, s_n\}$. If the number of players that agree to the broadcasted values of s and s_i is greater than the set of players from the adversary structure δ , then the opening of s is accepted.

4. Unconditionally Secure Multi-party Computation

First, we define the necessary arithmetic operations for computation in a passive adversary case. Computations in an active adversary setting will be addressed later in this section.

Threshold schemes used to securely compute a function f with a passive static or adaptive adversary can only compute a function securely if $t < \frac{n}{2}$. For a static or adaptive active adversary where a broadcast channel does not exist, the bound is $t < \frac{n}{3}$. Unconditionally Secure Multi-party Computation 36 multiplicative threshold function is Q^2 (Q^3). This leads us to a more general protocol for multi-party computation:

1) We can compute any function with a passive adversary structure provided that our secret sharing scheme is resilient to a Q^2 adversary structure.

2) To compute a function f in the presence of an active adversary our adversary structure must be Q^3.

4.1. SMPC with a Passive Adversary

Given two instances of Shamir's secret sharing scheme, the participants can compute the addition of secret simply by adding the shares of one instance to their corresponding shares in the other instance.

$$f + f_2 = (s_{1,1} + s_{2,1}) + (s_{1,2} + s_{2,2}) + \cdots + (s_{1,n} + s_{2,n}) = s_1 + s_2$$

Multiplication of a constant c can be computed by having each participant p_i compute $c_i = s_i \cdot c$. The resulting shares $c_1, c_2, \cdots, c_n$ determine $s \cdot c$. Multiplication is a bit more complicated as the multiplication of two polynomials would result in a new polynomial with degree of at most $\deg(f_1) + \deg(f_2) = 2t$ and the coefficients of the new polynomial would not be randomly distributed. To solve this problem, we perform a sanity operation, a reshare, after every multiplication that reduces the degree of $f_1 \cdot f_2$ and adds uniformly random values to all coefficients in $f_1 \cdot f_2$, except for first coefficients of each polynomial, ie, the secret. We will illustrate how to perform multiplication of two polynomials generated using Shamir's secret sharing scheme with an example.

Example. Given two values a and b, we can securely compute the value $c = ab$ with a passive adversary by executing the following steps:

1) Share a to $a_1, a_2, \cdots, a_n$ and b to $b_1, b_2, \cdots, b_n$ such that p_i receives shares a_i and b_i;

2) Each player p_i then computes the product of his two shares, $c_i = a_i \cdot b_i$;

3) Each player p_i then shares c_i to $c_{i,1}, c_{i,2}, \cdots, c_{i,n}$ and sends the shares to their respective players;

4) Each player p_j can now compute the value c_j' using the values received and the recombination vector.

$$\begin{pmatrix} c_{1,1} & c_{2,1} & \cdots & c_{n,1} \\ c_{1,2} & c_{2,2} & \cdots & c_{n,2} \\ \vdots & \vdots & \ddots & \vdots \\ c_{1,n} & c_{2,n} & \cdots & c_{n,n} \end{pmatrix} \begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_n \end{pmatrix} = \begin{pmatrix} c_1' \\ c_2' \\ \vdots \\ c_n' \end{pmatrix}$$

5) The shares $c_1', c_2', \cdots, c_n'$ determine $c = ab$ completing the multiplication.

Using these primitives, we can now evaluate an arithmetic circuit C over a field F computing a function f such that when the circuit completes, each player will have a share of the resulting computation.

Theorem ([20]). There exists functions that cannot be securely computed with a passive adversary if the adversary structure is not at least Q^2.

Proof. Consider for example the OR-function between two players. It is easy to see that this function can never be computed by the two participants, each providing one bit, without one of them leaking information.

Conversely, we can compute any function f securely with a passive adversary provided that the adversary structure is at least Q^2. To prove this, it is sufficient to show that given three values $a,b,c \in F$, we can always securely compute $a+b, c \cdot a, a \cdot b$ [21]. This was shown in the above example.

4.2. SMPC with an Active Adversary

In order to safely compute a function f on a set of values where we have a static or adaptive active adversary we require a method that allows the participants to check whether a player is executing the protocol correctly and providing valid shares. In other words, we need a stronger primitive that allows players to commit to a value. To achieve this, we will use the commitment scheme described in this Section.

Using this scheme, we can now construct a information theoretic SMPC protocol resilient against an active adaptive adversary Q^3 structure. Assume two committed input values a and b shared with CSP so that each player p_j holds a commitment to that share a_j and b_j. To compute the addition of a and b, each player p_i adds his two shares $c_i = a_i + b_i$ and computes a commitment for $a_i + b_i$. Multiplication of the values a and b:

1) Each player p_i multiplies his shares $c_i' = a_i \cdot b_i$ and commits to the result. Each p_i then performs CMP($C_a, C_b, C_{c_i'}$) where C_a, C_b and $C_{c_i'}$ are the commitments to a_i, b_i and c_i'.

2) Each player then shares his commitment to c_i' using the CSP protocol.

3) Every player now computes the value $c_j = \sum_{i=1}^{n} \lambda_i c_{ij}$ and a commitment C_{c_j} for it. Players can check if the value is correct because $C_{c_j} = \sum_{i=1}^{n} \lambda_i C_{ij} = \sum_{i=1}^{n} \lambda_i C_{ji}$.

If a participant fails in any of the above steps, he is disqualified, and if the adversary structure is Q^3, his input can be ignored, i.e., we remove corrupt players from the recombination vector. The reconstruction is still possible, because the number of honest players is sufficient enough to reconstruct the missing local multiplication. To illustrate this, recall that for a Q^3 threshold scheme the adversary threshold is $t < \frac{n}{3}$. Given this requirement, the number of honest players is at least $n - t > 2t$, which means that there exist enough honest players to reconstruct the missing local multiplication, which would be a polynomial of degree $2t$.

If we allow for a negligible error and assume a broadcast channel, then $\frac{n}{3} \le t < \frac{n}{2}$ is sufficient for secure multi-party computation [19]. In paper [18], it showed that we can construct a general secure multi-party computation scheme from any linear secret sharing scheme provided that the access structure allows MPC and VSS. That is, we can construct a secure multi-party computation protocol from any M with a Q^2 (Q^3) adversary structure.

4.3. SMPC with General Adversary Structures

It is relatively straightforward to use the techniques we have seen to construct protocols secure against general adversaries, i.e., where the adversary's corruption capabilities are not described only by a threshold t on the number of players that can be corrupt, but by a general adversary structure, as defined earlier.

What we have seen so far can be thought of as a way to build secure MPC protocols from Shamir's secret sharing scheme. The idea is now to replace Shamir's scheme by something more general, but otherwise use essentially the same high-level protocol.

To see how such a more general scheme could work, observe that the evaluation of shares in Shamir's scheme can be described in an alternative way. If the polynomial used is $f(X) = s + a_1 X + \cdots + a_n X^n$, we can think of the coefficients $(s, a_1, \cdots, a_n)$ as being arranged in a column vector α. Evaluating $f(X)$ in points $1, 2, \cdots, n$ is now equivalent to multiplying the vector by a Van der Monde matrix M, with rows of form $(i^0, i^1, \cdots, i^n)$. We may think of the scheme as being defined by this fixed matrix, and by the rule that each player is assigned 1 row of the matrix, and gets as his share the coordinate of M_α corresponding to his row.

It is now immediate to think of generalizations of this: to other matrices than Van der Monde, and to cases where players can have more then one row assigned to them. This leads to general linear secret sharing schemes, also known as Monotone Span Programs(MSP). The term "linear" is motivated by the fact any such scheme has the same property as Shamir's scheme, that sharing two secrets s, s' and adding corresponding shares of s and s', we obtain shares of $s + s'$. The protocol constructions we have seen have primarily used this linearity property, so this is why it makes sense to try to plug in MSP's instead of Shamir's scheme. There are, however, several technical difficulties to sort out along the way, primarily because the method we used to do secure multiplication only generalizes to MSP's with a certain special property, so called multiplicative MSP's. Not all MSP's are multiplicative, but it turns that any MSP can be used to construct a new one that is indeed multiplicative [22].

Furthermore, it turns out that for any adversary structure, there exists an MSP-based secret sharing scheme for which the unqualified sets are exactly those in the adversary structure. Therefore, these ideas lead to MPC protocols for any adversary structure where MPC is possible at all.

4. Conclusion

Study on SMPC is a hotspot in international cryptography. SMPC plays an important role in e-voting, e-auction, secret sharing,threshold signature etc. In this paper, we introduce the basic concept of SMPC and four basic agreement. And we separately illustrate commitment transfer protocol, commitment sharing protocol and commitment multiplication protocol. Last, we present unconditionally secure multi-party computation. In-depth work is needed for SMPC further researches and applications in related fields.

Acknowledgements

This work was supported by the Scientific Technology Research and Development Plan Project of Tangshan (No. 121302001a).

References

[1] Ach Yao. *Protocols for Secure Computation*. Proceeding of the 23rd IEEE Symposium on Foundations of Computer Science. 1982: 160-164
[2] O Goldreich, S Micali, A Wigderson. *How to play Any mental game*. In Proceedings of the nineteenth annual ACM conference on Theory of computing. 1987: 218-229.
[3] P Bogetoft, DL Christensen. Secure Multiparty Computation Goes Live. *Cryptology e-Print Archive Report*. 2008.
[4] Guo Xiaoqiang, Zhang Shuai, Li Ying. Key Technologies and Applications of Secure Multiparty Computation. *TELKOMNIKA*. 2013; 11(7): 3774- 3779.
[5] Goldwasser S, Micali S. Probabilistic Encryption. *Journal of Computer and System Sciences*. 1984; 28(2): 270-299.
[6] ElGamal T. A Public-Key Cryptosystem and A Signature Scheme Based on Discrete Logarithms. *IEEE Transactions on Information Theory.* 1985; 31(4): 469-472.
[7] Benaloh J. *Dense Probabilistic Encryption*. Proc. of the Workshop on Selected Areas of Cryptography. Kingston, Canada. 1994: 120-128.
[8] Paillier P. *Public-Key Crytosystems Based on Composite Degree Residuosity Classes*. Advances in Cryptology EUROCRYPT LNCS1592. Berlin: Springer-Verlag. 1999: 223-238.

[9] Michael O Rabin. How to exchange secrets with oblivious transfer. *Harvard University Technical Report*, 1981.
[10] Shimon Even, Oded Goldreich, A Lempel. *A randomized protocol for signing contracts*. Proc. CRYPTO'82. New York. 1983: 205-210.
[11] C Crepeau. *Equivalence between two flavours of oblivious transfers*. CRYPTO'87. Berlin Heidelberg. 1987.
[12] G Brassard, C Crepeau, JM Robert. *Information theoretic reductions among disclosure problems*. Proc. the 27th IEEE Symposium Foundations of Computer Science. Califorilia, 1986: 168-173.
[13] G Brassard, C Crepeau, JM Robert. *All or nothing disclosure of secrets*. CRYPTO'86. Berlin Heidelberg. 1986; 234-238.
[14] C Cachin. *On the foundation of oblivious transfer*. LNCS, CRYPTO'98. Berlin Heidelberg. 1998.
[15] A Yao. *Protocols for Secure Computation*. Proc of the 23rd IEEE Symposium on Foundations of Computer Science. 1982: 160-164.
[16] HK Lo, HE Chau. Why Quantum Bit Commitment and Ideal Quantum Cointossing are Impossible. *Physica D*., 1998; 120: 177-187.
[17] S Goldwasser, S Micali, C Rackoff. *The Knowledge Complexity of Interactive Proofs Systems*. Proc. STOC, New York: ACM Press, 1985: 291-304.
[18] R Cramer, I Damgard, U Maurer. General secure multiparty computation from any linear secret sharing scheme. *Cryptology e-Print Archive Report.* 2000.
[19] R Cramer, I Damgard. Multiparty computationan introduction. 2002.
[20] B Michael, S Goldwasser, A Wigderson. *Completeness theorems for non-cryptographic fault-tolerant distributed computation.* In STOC '88: Proceedings of the twentieth annual ACM symposium on Theory of computing. 1988: 1–10.
[21] R Cramer, I Damgard, JB Nielsen. Secure Multiparty Computation. 2012.
[22] China Science and Technology Association. Cryptography Subject Development Report, China Science and Technology Press. 2007-2010.