ELSEVIER

Computer Networks 38 (2002) 247–255

COMPUTER NETWORKS

www.elsevier.com/locate/comnet

Mental poker game based on a bit commitment scheme through network

Jue-Sam Chou, Yi-Shiung Yeh *

Department of Computer Science and Information Engineering, National Chiao-Tung University, 1001 Ta Hsueh Road, Hsinchu 300, Taiwan, ROC

Received 1 September 2000; received in revised form 2 September 2001; accepted 28 September 2001

Responsible Editor: I.F. Akyildiz

Abstract

There are many schemes proposed on mental poker so far. Most of them are based on the composition of each player's private permutation of cards. Yet, each one is either too complex or has some drawbacks in it. In other words, no solution has come to reality. In this paper, we propose a permutation-free method, i.e. a bit commitment scheme, along with the RSA cryptosystem (Cryptography–Theory and Practice, CRC Press, Boca Raton, 1995; Public-key, Cryptography, Springer, Berlin, 1996) to implement the mental poker game. It is not only simple but also concise in concept. © 2002 Elsevier Science B.V. All rights reserved.

Keywords: Bit commitment; Blob; RSA cryptosystem; Protocol; Permutation

1. Introduction

Much attention has been devoted to investigating the feasibility of playing mental poker on the Internet [3]. In 1979, Shamir et al. [12] proposed a method for playing poker games over the telephone between two players. However, Lipton found some shortcomings in their scheme [4]. Two years later, Goldwasser and Micali fixed the drawbacks by using probabilistic encryption. Unfortunately, their methods work just for two players [5]. In 1983, Banary and Furedi presented a simple poker protocol for three or more players. However, it is unable to minimize the players' collusion [6]. If any players want to share their knowledge, they should learn not only each other's hands, but their opponent's hands as well. In 1985, Fortune and Merritt proposed a protocol, based on the Banary–Furedi protocol, trying to solve the collusion problem. It needs a "Card Salesman" as a trusted party and cannot return one or more cards to the deck and reshuffle it. In their protocol, we must buy a new deck from the card salesman for every hand of poker each time we play [7]. In 1985, Yung presented a poker protocol using oblivious transfer and number embedding methods with a minimal assumption: there is no collusion between players [8], but their concept and computation are so complex. Crépeau presented two protocols in 1986 and 1987 respectively: the former was incomplete in

* Corresponding author. Tel.: +886-3-5731813; fax: +886-3-5724176.

E-mail addresses: gmjjs@ms29.hinet.net (J.-S. Chou), ysyeh@csie.nctu.edu.tw (Y.-S. Yeh).

PII: S 1 3 8 9 - 1 2 8 6 (0 1) 0 0 2 6 4 - X

the confidentiality of strategy, and the latter claimed to be the first complete solution to the mental poker game [9,10]. But in our view, there are still some weaknesses in it. For their lack of trusted third party (TTP), the dealing of cards may be unfair. Moreover, the coalition problem is not minimized. In 1991, Kurosawa et al. presented a mental poker protocol. Their card representation focuses on the sum of all players' random numbers rather than permutations. Yet, the method they used is complex and thus impractical [11]. Kurosawa et al. also presented a poker protocol in 1997, in addition to their complex computation as [11], when comparing with our method, they also confine their protocol to honest players and need a trusted party as well for their using π (which no player knows) to permute the 52 cards [13]. The other two papers about electronic gambling are [14,15]. However, they either just dealt with pure luck games or two-party game with a TTP. The collusion-free protocol of casino in Ref. [14] is based on the assumption that there is no secret communication link among any players, e.g., players to players or players to the dealer. On the other hand, in reality, this assumption is seldom to occur because of the usage of personal communication tool, such as mobile telephone.

2. Fragile permutation-based poker

There are many protocols based on permutation. This is the origin of their fragility. They are easy to suffer from coalition just by one player's revealing his permutation to another. For example, in the three-player (Alice, Bob and Charles) game of Ref. [6], each player has his own random permutation of the deck, A, B, and C respectively. If the players, Bob and Charles collude, then the player Charles can obtain all the cards in the hand of the player Bob. Furthermore, the player Charles can use the permutation CA^{-1} (where A^{-1} represents the inverse of A) of the player Bob to solve $A(H_{\mathrm{A}})$ which is all the cards in the player Alice's hand after her permutation. The Crépeau protocol uses each player's permutation to operate on a card. By using his method, we check the two-player version: When player P_j wants to get a card, he picks a card v and marks "used". Afterwards, he uses this v to ask player P_i to compute his permutation on this card. Let π_i be a permutation for player P_i, for $i = 1$ to n. The result is denoted as $\pi_i(v)$. Then player P_j operates his permutation on $\pi_i(v)$ and obtains card $\pi_j\pi_i(v)$. In this way, though player P_i may not know the right position of card v after the composition of the permutation in player P_j's hand, player P_i can know player P_j's card set. This is a serious drawback. Let three players, P_i, P_j and P_k, be included in the game of the multi-player version of Crépeau protocol. If player P_i colludes with player P_j, that is, if player P_i knows the private permutation of player P_j, then he knows π_i and π_j. Since the card's public form in each player's hand is $\pi_k\pi_j\pi_i(v)$, player P_i can deduce the π_k of player P_k on v by tracing the composition of the permutation of card v. That is why this method is subjected to the necessity for each hand of the game, each player must choose a new permutation. Besides, in the multi-player version, Crépeau protocol has the same serious drawback as in the two-player version, player P_l can know any players P_i's card set.

3. Mental poker game

We describe the specifications, card sets and operations of a mental poker game as follows.

3.1. The specifications of the mental poker game [8]

For any number of players to play a fair "mental poker game", they need the following protocols:

(1) A protocol for reading cards: one player reads one card at a time from a deck that has not been read yet.
(2) Protocols for other game steps: opening a card, discarding cards from the player's hand, exchanging cards with other players, selecting a card from other player's hand, etc. in a secure and checkable manner.
(3) A protocol for game management: composing and managing all the game steps to form a complete mental poker game.

3.2. Card sets and operations [8]

When the game is going, there are some cards on the table or in players' hands and some operations occur. Here we define private card set, public card sets and operations as follows.

3.2.1. Definition of private card set

Only player P_i that owns it can know the cards.

H_{P_i}: the card set in the hand of player P_i.

3.2.2. Definition of public card sets

(1) Card-set: the card set $\{c_1, c_2, c_3, \ldots, c_w\}$ for w cards.
(2) Tabopen: the cards that are face-up on the table.
(3) Discard: the cards that are inactive till the end of the game.
(4) Used_{P_i}: the cards that belong to the player P_i which are now face-up on the table.
(5) Deck: the card set which is face-down on the table and equals to

$$\text{Card-set} - \bigcup_{\forall i} H_{P_i} - \bigcup_{\forall i} \mathrm{Used}_{P_i} - \text{Tabopen} - \text{Discard}.$$

3.2.3. Operations

When the game is going, there are some activities (operations) held. For instance, the dealer (or TTP) shuffles cards or distributes cards, player P_i changes a card with another player $P_j, \ldots,$ and so on. We will mention the details in Section 5.2.

4. Using a bit commitment scheme

In this paper, we use a permutation-free scheme, a bit commitment scheme, to implement the mental poker game. Firstly, we define some notations.

4.1. Notations and definitions

n $n = pq$, n is public to the players, where p and q are large primes,
N number of players,
b $b \in Z_2$,
x $x \in Z_n^*$,
$f(b,x)$ function f publicly known to the players is called a bit commitment scheme. It is an encryption method of b and its output in Z_n^* is called a blob. When one wants to use f to produce a blob, he uses a b and a random x as the parameters of the function f,
QNR_n quadratic nonresidue modulo n,
m m is public to the players, $m \in \mathrm{QNR}_n$ and $m \in Z_n^*$,
X a vector composed of the order of all the x values that have been used in the function f,
B a vector composed of the order of all the blobs that have been produced by the function f,
X_i a vector represents the X portion of card c_i,
B_i a vector represents the B portion of card c_i,
TTP trusted third party,
$S_{si}(M)$ signature of message M with player P_i's private key si,
$S_{st}(M)$ signature of message M with TTP's private key st,
$E_{kj}(M)$ encryption of message M with player P_j's public key kj,
iSB equals to $S_{si}(B)$,
iPX equals to $E_{ki}(X)$.

4.2. The basic concept of our method and its advantages

The basic concept and advantages of our method are described as follows.

4.2.1. Basic concept

There are two good properties that a bit commitment scheme possesses [1]:

(1) Concealing: none of the players can determine the value b from the blob. In other words, a blob reveals no information about the value x provided that the quadratic residue problem is infeasible.

(2) Binding: the sender can open the blob later by revealing the value of x to convince the receiver that b was the committed value.

Suppose our bit commitment function $f(b,x)$ equals to $m^b x^2 \bmod n$ and there exists two different bits b_0 and b_1 that satisfy the equation

$m^{b1}X^2 \equiv m^{b0}X^2 \bmod n$. Afterwards, the receiver can argue that the card received is not the one from the sender. However, this situation is impossible. Because if mx^2 equals to $x^2 \bmod n$ for $b_1 \neq b_0$, then it implies that $m \equiv 1 \bmod n$ which is a contradiction to the predefined value $m \in \mathrm{QNR}_n$.

In conclusion, one can transfer a card to another secretly, undeniably and unalterably. The players can encode the cards at will; for example, the players can encode the 52 cards as $c_1 = 010010$, $c_2 = 100011, \ldots, c_{52} = 011100$ (here, we take the card length $r = 6$ and $w = 52$ as an example), which are the 52 random strings known to players.

4.2.2. Method and advantages

Our approach uses a bit commitment scheme to randomly choose a x to encrypt one bit b of a card's coded form. Its output is called a blob. If the card's coded form has r bits in length, we will produce r blobs by repeating same scheme and collect r blobs as a vector B (here, we can consider the vector B as the marked back of a real world poker card). The r randomly chosen various x's are collected as a vector X in accordance with the order of the blobs. Whenever player P_i wants to send a card to player P_j, he uses the function f with the desired b's and x's to produce the vector (B,X) for that card. Firstly, he sends to player P_j the *iSB*. If a hacker obtains the vector B, he will not have any ways to get anyone of the r b's due to the concealed property of the blobs. When player P_j receives the *iSB*, he decrypts it and gets the vector B. Thus, he has the evidence that he has received the card in blob form. Afterwards, player P_i sends to player P_j the *jPX* that can be used to get the corresponding bit comprised in each blob. Thus, player P_j can get the card. If the sender (player P_i) repudiates the particular card that he has sent, the receiver can indicate the *iSB* and the *jPX* he has received. The successful probability that any of the players can successfully forge the vector X is equivalent to the probability that a bit commitment scheme can be broken. If the receiver claims that the card received is not the card sent by the sender, the sender can also indicate the *iSB* and the *jPX*. Therefore, it is impossible for the receiver to change a particular card he has received. Besides, assume that a hacker has the corresponding vectors B and X, he cannot get the card because of the secrecy of the function f and the cards' coded forms. In other words, the hacker has little opportunity to obtain any information about the card. This is the overall working structure of our scheme. We delineate our method using $r = 6$ in Fig. 1.

5. Proposed scheme

Now we present the concernment of the poker game as follows.

5.1. Card expression

Before playing the game, we need to prepare a deck of w cards. Each card is composed of r bits. Each player confirms that each card is unique and corresponds to one of the w cards in reality of the poker game. Here, we take $w = 52$ as an example.

5.2. Elementary operations

5.2.1. Trusted third party shuffling cards (preparing a new deck) (OP1)

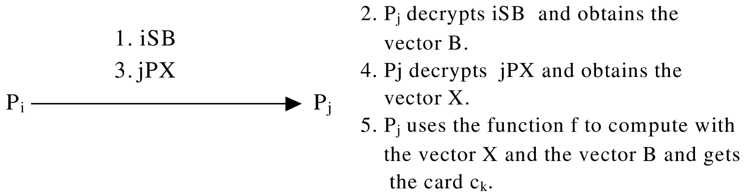
Each time when we want to hold another game or the Deck on the table is used up, we need to shuffle the cards (the 52 cards or the cards in the Tabopen ∪ Discard). To do this, TTP produces the corresponding vector $v = (B,X)$ using function f for each card expression in the new formed Deck. In other words, the cards in the Deck are represented as vectors $v_k = (B_k,X_k)$, $k = 1$ to p.

5.2.2. Trusted third party distributing cards to players (OP2)

Each time TTP distributes a card, he randomly chooses a card v_k in the Deck and distributes the B portion of v_k to player P_i from the Deck in the form *tSB*. Afterwards, TTP sends the X portion of v_k in the form of *iPX* to player P_i. Thus player P_i can get the card v_k in bit form. In addition, in our scheme, the action between the distributing of one card from TTP and a player reading a card from the Deck is the same. After the distributing of the

```
$c_k=b_5b_4b_3b_2b_1b_0$ /*the k$^{th}$ card*/
$x_i \in Z_n^*$          /* 0 <= i <=5 */
$b_i \in Z_2$
$X = ( x_0, x_1, x_2, x_3, x_4, x_5 )$
$B=(blob_0,\dots, blob_5)$
                /* $blob_u=f (b_u, x_u)$, $_{0<= u <=5}$ */
```

x	b	f (b,x) = blob
x_0	b_0	$blob_0$
x_1	b_1	$blob_1$
x_2	b_2	$blob_2$
x_3	b_3	$blob_3$
x_4	b_4	$blob_4$
x_5	b_5	$blob_5$



2. P_j decrypts iSB and obtains the vector B.
4. Pj decrypts jPX and obtains the vector X.
5. P_j uses the function f to compute with the vector X and the vector B and gets the card c_k.

Fig. 1. Player P_i sends a card, c_k, to player P_j.

card, TTP removes the card v_k from the Deck. Player P_i adds the card to H_{P_i}.

5.2.3. P_i exchanging a card with P_j (OP3)

When P_i and P_j both want to exchange cards, originally, c_k in H_{P_i} and c_s in H_{P_j}, they conduct the following steps:

Step 0: P_i using function f to represent the card c_k in the form (B_k,X_k) and P_j using function f to represent the card c_s in the form (B_s,X_s).

Step 1: P_i sends the B_k of the card c_k in H_{P_i} in *iSB* form to P_j.

Step 2: P_j sends the B_s of the card c_s in H_{P_j} in *jSB* form to P_i.

Step 3: P_i sends the X_k of the card c_k in *jPX* form to P_j.

Step 4: P_j sends the X_s of the card c_s in *iPX* form to P_i.

Step 5: P_i uses X_s to compute with B_s and obtains the card c_s.

Step 6: P_j uses X_k to compute with B_k and obtains the card c_k.

Step 7: P_i removes the card c_k from H_{P_i} and P_j removes the card c_s from H_{P_j}.

Step 8: P_i adds the card c_s to H_{P_i} and P_j adds the card c_k to H_{P_j}.

5.2.4. P_i revealing a card to tabopen (OP4)

When P_i wants to reveal his card, he just throws out the bit-form card from his hand to the Tabopen. P_i removes the card from H_{P_i}, and then adds the card to Tabopen.

5.2.5. P_i discarding a card to sets, discard or Used$_{P_i}$ (OP5)

When P_i wants to discard his card, he just throws out the bit-form card from his hand to the set Discard or to the set Used$_{P_i}$. P_i removes the card from H_{P_i}, and then adds the card to the set Discard or to the set Used$_{P_i}$.

5.2.6. P_i reading a card from tabopen (OP6)

When P_i wants to read a card from Tabopen, he just takes the bit-form card from Tabopen. P_i removes the card from Tabopen. Therefore, every player knows the card is read by P_i. Afterwards, P_i adds the card to H_{P_i}.

5.2.7. P_i reading a card from the deck (OP7)

In our protocol, the action that player P_i reads a card from the Deck is the same as that TTP distributes a card to P_i (OP2).

5.2.8. P_i reading a card from P_j (OP8)

When P_i wants to read a card c_s from P_j, firstly, P_j uses function f to encode all the bit-form cards into (B, X) form, then P_j sends the B portion of all the cards in the form of *jSB* in H_{P_j} to P_i. P_i selects one and returns the others back to P_j. P_j sends the X_s of the selected card c_s in the form *iPX* to P_i. Thus, P_i can get the card c_s. P_i adds the card c_s to H_{P_i}. P_j removes the card c_s from H_{P_j}.

5.2.9. P_i passing (OP9)

P_i does nothing.

5.3. Game playing

After defining the elementary actions, we can simulate a poker game successively as follows.

5.3.1. Initializing phase

Before the poker game begins, the TTP prepares the following public and private information for the players and himself:

(1) Public information to players and TTP: function *f*, public keys of each player and TTP, a deck of 52 cards in bit form known to each player.
(2) Secret information: private keys of each player and TTP.

5.3.2. Playing phase

According to the rules of the poker game type, we can invoke the different elementary actions to achieve the game.

5.3.3. Checking phase

When the game is going, all the *iSB*'s, *tSB*'s, *jPX*'s, bit-form cards and their corresponding players' ID in the occurrences of any operations are recorded to provide the checking after the game ends for the TTP (the TTP has all the private keys of all the players) to ensure that the right card is played by the right player and also the uniqueness of all the cards.

5.4. Example

An example of our approach is for the popular poker game Poker in which the players need to read a card from the Deck, discard a card to the Discard or pass. All the actions can be achieved just by using the OP2 (OP7), OP4 or OP9 operations.

6. Analysis and comparison

6.1. Analysis

Our approach uses the bit commitment and RSA public key schemes [2]. Therefore, the security of our system is based on the infeasible computation of the quadratic residue problem (QRP) or the discrete logarithm problem (DLP) in Z_n^* for bit commitment schemes (we know that the bit commitment scheme can be based on QRP or DLP) and on the infeasible computation of the factorization problem of the RSA scheme.

In our method, the vector B can be viewed as the back of a card in reality that is known by sender P_i but unknown to receiver P_j. In more details, as in OP8, when P_j wants to select a card c_k randomly from P_i, he selects its back form B_k that is known to P_i but unknown to P_j from H_{pi}. After that, P_i has to send the corresponding X_k to P_j without cheating; otherwise, any X besides X_k will not match the B_k due to the bit commitment scheme. Therefore, this can exactly simulate the real poker reading operation, such as OP8 in our method and that is the main reason why other methods are complicated and hard to understand. In our scheme, there are only four operations using of both the bit commitment scheme and the RSA scheme, OP1, OP2,

Table 1
The differences between mental poker protocols proposed so far

Method	Players	Minimized coalition	Simple in concept	Provably or computationally secure	Need trusted party
[5]	2		√	√	
[6]	$\geqslant 3$		√		
[7]	$\geqslant 2$	√	√		√
[8]	$\geqslant 2$	√		√	
[9]	$\geqslant 2$			√	
[10]	$\geqslant 2$			√	
[11]	$\geqslant 2$	√		√	√
[13]	$\geqslant 2$	√		√	√
Our method	$\geqslant 2$	√	√	√	√

Method [*i*] represents the method described in the *i*th paper listed in the references.

OP3 and OP8. The others use merely the RSA scheme. For more simply speaking, due to the employment of TTP, we can go one step further to reduce the number of operations using the bit commitment scheme and the RSA scheme into two only, the OP3 and OP8. In reality, the situation of OP3 seldom occurs. Therefore, we can almost confirm that our scheme has just one operation OP8 using both the bit commitment scheme and the RSA cryptosystem. The other operations use only the RSA scheme to transport the bit-form cards.

6.2. Comparison

With the vector B and the vector X, our method can work consistently to a real poker game. We list the differences between our approach and other proposed methods in Table 1. The property "minimized coalition" means that if any players share their knowledge, they cannot learn any others' hands except the colluding players. In addition, the property "provably or computationally secure" means the function used in the corresponding method is computationally infeasible or provably secure.

7. Discussion

7.1. Collusion problem

The collusion problem has focus on intensive attention of many researchers, such as [16–20] in recent years. It is still an open problem in mental poker game or in other protocols, such as key agreement protocols [21]. In our mental poker game, we also must make an assumption that there is no collusion problem among players.

7.2. Why trusted third party needed

When playing a mental poker game, at the beginning, each player has to have some fair distributed cards in his/her hand. We call the initial hand of player P_i as ihand$_i$. In our method, the cards in ihand$_i$, $1 \leqslant i \leqslant N$, are obtained from the Deck. In order to ensure the fairness of card distributing among players, the Deck needs to be unknown to anyone. This is the reason why we need a TTP. Let us suppose that, there is no TTP, if there occurs the collusion phenomenon among any players; the dealing of cards is unfair.

Fact. *If there is no TTP, the dealing of cards in the mental poker game will be uncertain to be fair.*

Justification. Suppose there are N players, P_1, $P_2, \ldots, P_N$, the 52 cards in a Deck are represented as $c_1, c_2, \ldots, c_{52}$ and there is no TTP in the game. Without loss of generality, we assume that $N-1$ of the players collude, such as, P_1, $P_2, \ldots, P_{N-1}$. When the N players need to shuffle the cards, despite which means they use, there is a possibility that the colluding players, $P_1, P_2, \ldots, P_{N-1}$, know which card comes from the 52 cards. For example, without loss of generality, assume that the sequence of the operations on card c_i by the $N-1$ players is $1, 2, \ldots, N-1$, $O_{N-1} \ldots O_2 O_1(c_i)$ (IO), then the colluders know the corresponding relationship between the original card c_i and the intermediate result IO. Though the IO must undergo the operation by player N further, $O_N O_{N-1} \dots O_2 O_1$ (c_i), each player in the colluding set can record the corresponding relationship between the original card c_i and the outcome $O_N O_{N-1} \dots O_2 O_1$ (c_i) (O). Yet, the player N has no idea about the corresponding relationship between the original card c_i and the outcome O. That is why we need a TTP. □

7.3. *Why iSB not in the Form S_{si} (j,B), When P_j wants to select a card from H_{Pi}*

We know that in the asymmetric key system, the identifier j of player P_j within the scope of the signature prevents P_j from sending the signed B onto another player and impersonating player P_i. However, in our system, the identifier j in the signature is unnecessary for that we have the jPX and the two properties of the bit commitment. That is, if player P_j wants to send a card to another player P_k and impersonate player P_i, not only should he (player P_j) send $S_{si}(B)$ to P_k but also send him (player P_k) the kPX, the right X in the kPX is from the jPX he received from player P_i. If it is under this situation, it will contradict the uniqueness of the cards and be detected in the checking phase. (Another case is for P_j to select a card from TTP, in this case, the relationship is why $tSB(= S_{st}(B))$ not in the form $S_{st}(j, B)$ and the reason is the same as mentioned above.)

In addition, as to the X vector, we can also use hash function H to hash it, e.g., $P_j(X, S_{si}(H(X)))$ to assure the integrity of X for player P_j, but this again is unnecessary for the two properties of a bit commitment scheme.

7.4. *Environment of the game*

In our scheme, we make an assumption that there is no collusion problem. However, indeed, we have no way to guarantee. The only thing we can do is to make the environment become simple, secure and efficient on the Internet [21]. Moreover, in a legal gamble, one might use the key escrow system with key distribution to improve the collusion phenomenon. For example, all pairs in the system may escrow their session keys first, and then the law enforcement agent can successfully wiretap the communication messages to decrypt the suspicious communication. Our method is easy to construct for this application, compared with other complex methods proposed so far.

8. Conclusion

In this paper, we have used the nice properties of the bit commitment scheme to enable the secure transfer of a card. Our method is simpler and more concise compared to all other protocols. Since our approach has no cheating and repudiation, properties needed as the basic requirement in a gamble during the operation, e.g., OP8, taken by two players because of the binding property of b and the receiving of the blobs as the evidence. The hacker may know the function f and the coded form of the 52 cards, but there is no way for him to get the decrypted form of the encrypted vector X. As for the collusion problem, there is no efficient solution so far. Here, we may adopt the key escrow system into our mental poker game to make collusion-free become feasible. Finally, we also design a checking phase in the last phase of the game that is achieved by TTP to assure the uniqueness of all the cards. Thus, we can have a nice mental poker game.

References

[1] D. Stinson, Cryptography—Theory and Practice, CRC Press, Boca Raton, 1995.

[2] A. Salomaa, Public-Key Cryptography, second ed., Springer, Berlin, 1996.

[3] W. Stallings, Cryptography and Network Security, Principles and Practice, Prentice-Hall, Englewood Cliffs, NJ, 1999.

[4] R. Lipton, How to cheat at mental poker, Proceedings of the AMS Short Course in Cryptography, 1981.

[5] S. Goldwasser, S. Micali, Probabilistic encryption and how to play mental poker keeping secret all partial information, Proceedings of the 14th Annual ACM Symposium on Theory of Computing, ACM-SIGACT, May 1982, pp. 365–377.

[6] I. Banary, Z. Furedi, Mental poker with three or more players, Informat. Cont. 59 (1983) 84–93.

[7] S. Fortune, M. Merrit, Poker protocols, in: G.R. Blakley, D. Chaum (Eds.), Advances in Cryptology: Proc. CRYPTO84, Lecture Notes in Computer Science, vol. 196, Springer, Berlin, 1985, pp. 454–464.

[8] M. Yung, Cryptoprotocols: subscription to a public key, the secret blocking and the multi-player mental poker game, in: G.R. Blakley, D. Chaum (Eds.), Advances in Cryptology: Proc. CRYPTO84, Lecture Notes in Computer Science, vol. 196, Springer, Berlin, 1985, pp. 439–453.

[9] C. Crépeau, A secure poker protocol that minimizes the effect of player coalitions, in: H.C. Williams (Ed.), Advances in Cryptology: Proc. CRYPTO85, Lecture Notes in Computer Science, vol. 218, Springer, Berlin, 1986, pp. 73–86.

[10] C. Crépeau, A zero-knowledge poker protocol that achieves confidentiality of the players' strategy, or how to achieve an electronic poker face, in: A. Odlyzko (Ed.), Theses in Advances in Cryptology: Proc. CRYPTO86, Springer, Berlin, 1987.

[11] K. Kurosawa et al., General public key residue cryptosystems and mental poker protocols, Advances in Cryptology: Proc. EUROCRPT'90, Workshop on the Theory and Application of Cryptographic Techniques, Aarhus, Denmark, 21–24 May 1990, Proceedings, Lecture Notes in Computer Science, vol. 473, Springer, Berlin, 1991.

[12] A. Shamir, R. Rivest, L. Adleman, Mental poker, MIT Technical Report, 1978.

[13] K. Kurosawa, Y. Katayama, W. Ogata, Reshuffleable and laziness tolerant mental poker game, IEICE Trans. Fundamentals E80-A (1) (1997) 72–78.

[14] C. Hall, B. Schneier, Remote electronic gambling, Computer Security Applications Conference, Proceedings, 13th Annual, 1997, pp. 232–238.

[15] W. Zhao, V. Vijay, Y. Mu, Fair on-line gambling, Computer Security Applications, ACSAC'00, 16th Annual Conference, 2000, pp. 394–400.

[16] H. Tanaka, Simply implemented identity-based non-interactive key sharing, Information Theory, 1995. IEEE International Symposium on 17–22 September 1995, p. 357.

[17] H. Tanaka, Security certified identity-based non-interactive key sharing, Information Theory, 1994. Proceedings, 1994 IEEE International Symposium on 27 June–1 July 1994, p. 495.

[18] S. Park, Y. Kim, S. Lee, K. Kim, Attacks on Tanaka's non-interactive key sharing scheme, Information Theory, 1995. Proceedings, 1995 IEEE International Symposium on 17–22 September 1995, p. 356.

[19] S.H. Low, N.F. Maxemchuk, Collusion analysis of cryptographic protocols, IEEE Global Telecommunications Conference, vol. 1, 18–22 November 1996, pp. 1–5.

[20] S.H. Low, N.F. Maxemchuk, An algorithm to compute collusion paths, IEEE INFOCOM, vol. 2, 7–12 April 1997, pp. 745–751.

[21] G. Ateniese, M. Steiner, G. Tsudik, New multiparty authentication services and key agreement protocols, IEEE J. Sel. Areas Commun. 18 (4) (2000) 628–638.

Jue-Sam Chou received his Master degree in Applied Math. from National Chung Hsing University (NCHU) in Taichung, Taiwan, ROC in 1991, and will soon received his Ph.D. degree in Computer Science and Information Engineering from National Chiao Tung University (NTCU) in Hsinchu, Taiwan, ROC. He teaches at Ta Hwa Institute of Technology. His primary research interests are electronic commerce, data security and privacy and statistics.

Yi-Shiung Yeh. Education: September 1981–December 1985 Ph.D. in Computer Science, Department of EE and CS, University of Wisconsin Milwaukee. September 1978–June 1980 MS in Computer Science, Department of EE and CS, University of Wisconsin–Milwaukee. Professional background: August 1988—Now Associate Professor, Institute of CS and IE, National Chiao-Tung University. July 1986–August 1988 Assistant Professor, Department of Computer and Information Sciences, Fordham University. July 1984–December 1984 Doctorate Intern, Johson Controls, Inc. August 1980–October 1981 System Programmer, System Support Division, Milwaukee County Government. Research interest: Cryptography and Information Security, Reliability and Performance, DNA Computation.