Fair On-line Gambling

Weiliang Zhao, Vijay Varadharajan and Yi Mu
School of Computing & Information Technology
University of Western Sydney, Nepean
PO Box 10, Kingswood, NSW 2747, Australia
{wzhao, vijay, yimu}@cit.nepean.uws.edu.au

Abstract

This paper proposes a fair electronic gambling scheme for the Internet. The proposed scheme provides a unique link between payment and gambling outcome so that the winner can be ensured to get the payment. Since an optimal fair exchange method is used in gambling message exchange, the proposed system guarantees that no one can successfully cheat during a gambling process. Our system requires an off-line Trusted Third Party (TTP). If a cheating occurs, the TTP can resolve the problem and make the gambling process fair.

1. Introduction

With the growth of popularity of the Internet, the Internet has become an important marketplace for on-line gambling. There are numerous on-line gambling and casino web sites on the Internet. For on-line gambling to be "successful" several issues of security need to be properly addressed. These include fairness of gaming transactions, security of payment and other details, privacy of players, trustworthiness of the playing authorities and the ability to resolve disputes. Many of existing on-line casino games provide some level of security and privacy. However, fairness is entirely based on the trust of the casino/dealer.

We are interested in the situations where the on-line casino is not necessarily trusted. That is, we have "untrusted" gaming sites. This is particularly important in practice as in many countries on-line gambling is not regulated by government authorities. In such cases, for instance, there may not be any guarantee that the casino authorities are not having an unfair advantage over the players. In such circumstances, at least as far as the playing of the game is concerned, it is necessary to have fair exchange schemes. A fair exchange scheme [1, 2, 3, 4] requires a trusted third party (TTP) who helps to resolve disputes amongst the playing entities. In general, the TTP can be on-line or off-line. For efficiency reasons, it is preferable that TTP is off-line. In this case, the TTP only comes into play when a problem occurs in the gambling system; otherwise, TTP is not contacted. In this paper, we consider a gambling scheme where there are players and a dealer. Normally in gambling systems, a player has to bet or pay money in advance before he or she can play the game. We will assume this in our games. The fair exchange scheme we propose resolves the following disputes: (1) the dealer refuses to make a payment to the player who has won, (2) the dealer denies a payment that was made by a player in advance, and (3) the player, who payed to the dealer in advance, refuses to accept the gambling outcome after he or she has lost.

An on-line gambling scheme must be associated with an on-line payment scheme. Credit-based payment methods [5] have been popular in on-line gambling casinos. The payment scheme used in our system is based on a secure electronic credit system that is similar to 3KP [6] or SET [7].

Our contributions in this paper are twofold. First involves the proposal of a fair exchange scheme for on-line gambling transactions. Second a secure linking of the on-line gambling with payment. Hence the overall scheme we propose provides a unique link between a gambling process and its associated payment and which makes the whole transaction process fair.

The rest of the paper is organized as follows. In section 2, we review the methods of equality proof of knowledge ([8], [9]) and proof of equivalence of discrete logarithm to discrete log-logarithm [10]. These methods will be used in the rest of the paper for our fair and secure on-line gambling scheme. In section 3, we propose general fair exchange with credit based payment and describe the details of the fair exchange protocol. In section 4, we first discuss the general characteristics of on-line gambling and then develop a secure protocol for on-line "luck based" games. Finally, section 5 concludes the paper with some final remarks.

2. Preliminaries

2.1. Equality proof of knowledge

The scheme of equality proof of knowledge was initially proposed by Chaum and Pedersen [8] and Verheul and Tilborg [9]. The scheme is about proving knowledge of something without revealing anything about its content. The public information includes a prime number p and a generator $g_i \in Z_p^*,\ i = 1, 2, \cdots, l$, where l is the confidence level. In order to prove x, the prover chooses $r \in Z_p^*$ and computes

$$a_i = g_i^r \mod p,$$
$$h_i = g_i^x \mod p.$$

Challenges c and z are calculated as follows

$$c = H(g_1||g_2||\cdots||g_l||a_1||a_2||\cdots||a_l||h_1||h_2||\cdots||h_l),$$
$$z = cx + r \mod p.$$

The verifier will check the following equation to prove the knowledge

$$g_i^z \stackrel{?}{=} h_i^c a_i \mod p.$$

For all i, $g_i^z = h_i^c a_i \mod p$ indicatess that the prover has the knowledge; otherwise, he does not.

2.2 Proof of equivalence of discrete logarithm to discrete log-logarithm (PEDLDLL)

PEDLDLL was initially proposed by Stadler [10]. For two given primes p and q (where $p = 2q + 1$), let $x, y, z \in Z_q^*$ and $X, Y \in Z_p^*$. There exists an $\alpha \in \{1, 2, \cdots q - 2\}$ such that $y = x^\alpha \mod q$ and $Y = X^{z^\alpha} \mod p$. Without revealing α and z^α, a prover, who knows α, can generate a certificate to prove that $y = x^\alpha \mod q$ and $Y = X^{z^\alpha} \mod p$.

If the confidence level is l, for $i = 1, 2, \cdots, l$, the prover chooses $w_i \in \{1, 2, \cdots, q - 2\}$ and computes $t(x_i) = x^{w_i} \mod q$, $t(X_i) = X^{z^{w_i}} \mod p$. Then he could get

$$c = H_l(x||y||z||X||Y||t(x_1)||t(X_1)||\cdots||t(x_l)||t(X_l)).$$

For every bit $c = c_1 c_2 \cdots c_l$, the prover computes $R = (r_1, r_2, \cdots, r_l)$, where $r_i = w_i - c_i\alpha \mod q - 1$. The certificate is given by (R, c).

During certificate verification, the verifier will check whether

$$c = H_l(x||y||z||X||Y||u_1||U_1||\cdots||u_l||U_l)$$

where $u_i = x^{r_i} y^{c_i} \mod q$ and

$$U_i = \begin{cases} X^{z^{r_i}} \mod p & if\ \ c_i = 0 \\ Y^{z^{r_i}} \mod p & if\ \ c_i = 1 \end{cases}$$

3. General Fair Exchange with Credit Based Payment

The fair exchange scheme proposed in this paper is based on credit payment scheme discussed in [5]. We assume that the bank and a TTP (trusted third party) are off-line. We will also assume that the credit information of the client is anonymous with non-interactive equality proof.

The following notations are used in the description of the fair exchange scheme:

- PKX: Public key of user X.
- skx: Private key of user X.
- C: Client with public key PKC and private key skc
- M: Merchant with public key PKM and private key skm
- TTP: TTP with public key PKT and private key skt
- B: Bank with public key PKB and private key skb.
- t_p : Timestamp generated by party P.
- $< \cdots >_{skx}$: Signature with secret key skx.
- $P_{enc}(PKX, M)$: Encryption of message M with public key PKX.
- $P_{dec}(skx, Cipher)$: Decryption of ciphertext C with private key skx.
- $H(M)$: Hash function on message M.

3.1. System setup

The credit token is of the form

$$\mathcal{C} = \ < C, l, h_1, h_2, \cdots, h_l, E, A >_{skb}$$

The credit token contains the client's identity C, the confidence level l, the expiry date E, maximum credit amount A and $h_i = g_i^x \mod p$, where $g_i \in Z_p^*$ are common generators for $i = 1, 2, \cdots, l$, where x is the concatenation of PIN number, credit card number and salt. The credit token is signed by the bank using its private key skb. The payment slip token has the form

$$S = \mathcal{C}, M, O, \$, tc,\ H(\mathcal{C}, M, O, \$, tc),$$

where M is ID of merchant, O is the order, $\$$ is the amount of money and currency type and t_c is the timestamp generated by the client C. The payment slip is signed by the client with private key skc.

The encrypted payment slip token is

$$C_S = P_{enc}(PKT, < S >_{skc}).$$

$C_S Cert$ is the token to prove C_S is a ciphertext of S without disclosing the signature. Let us consider the construction of the token $C_S Cert$. Let p and q be prime numbers of the form $p = 2q + 1$. We will assume that $q - 1$ has no small prime factors except 2. We will use ElGamal system for encryption and decryption by TTP. ElGamal system has g a generator selected from Z_q^*, where q is a prime number. $skt \in \{1, 2, \cdots, q-2\}$ is the private key and $PKT = g^{skt} \mod q$ is the public key.

For encryption of message m, we have the following:

$$P_{enc}(PKT, m) = (W, V) \mod q,$$

where $W = g^w$ and $V = m(PKT)^w$, $w \in \{1, 2, \cdots, q-2\}$ is a randomly chosen number.

The signature scheme works as follows: Choose a random $k \in Z_q^*$, the signature has the form

$$< S >_{skc} \equiv (r, s)$$

where $r = G^k \mod p$ and $s = k^{-1}(H(S) + r \times skc) \mod q$ and $PKC = G^{skc} \mod p$.

Encrypting the above signature s with PKT, we have, $P_{enc}(PKT, s) = (W, V)$. The encrypted payment slip with signature is then given as follows:

$$C_S = \{r, W, V\},$$

where $W = g^w \mod q$, $V = s(PKT)^w \mod q$.

With transformation $x = g$, $y = W^{-1} \mod q$, $z = PKT$, $X = r^V \mod p$, $Y = G^{H(S)}(PKC)^r \mod p$ and $\alpha = -w$, choose $w_i \in \{1, 2, \cdots, q-2\}$, then

$$t(x_i) = x^{w_i} \mod q, t(X_i) = X^{z^{w_i}} \mod p$$

and

$$c = H_l(x||y||z||X||Y||t(x_1)||t(X_1)|| \cdots ||t(x_l)||t(X_l))$$
$$c = c_1 c_2 \cdots c_l$$
$$r_i = w_i - c_i \alpha \mod q - 1$$

(R, c) is the certificate $C_S Cert$ for C_S.

The process of verification is to check,

$$c = H_l((x||y||z||X||Y||u_1||U_1|| \cdots ||u_l||U_l)$$

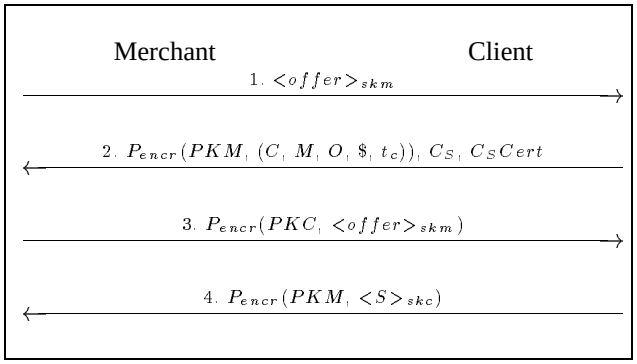
where $u_i = x^{r_i} y^{c_i} \mod q$, and

$$U_i = \begin{cases} X^{z^{r_i}} \mod p & if \ \ c_i = 0 \\ Y^{z^{r_i}} \mod p & if \ \ c_i = 1 \end{cases}$$

3.2. Fair-exchange protocol

The following is the fair-exchange protocol,

FAIR-EXCHANGE PROTOCOL



The above protocol is totally fair. If both the client and the merchant perform properly, the TTP will not be involved in the protocol. In Step 1, the merchant sends his signed offer to the client. In Step 2, the client sends $P_{encr}(PKM, (C, M, O, \$, t_c))$, C_S, $C_S Cert$ to the merchant, the merchant (1) checks credit information with equality proof of knowledge; (2) uses $C_S Cert$ to check C_S is the ciphertext of the payment slip S signed by the client. If the client performs improperly, the merchant can detect it and stop the protocol. In step 3, the merchant sends $P_{encr}(PKC, < offer >_{skm})$ to the client. If the merchant performs improperly in step 3, the client can check the offer and stop the protocol. In step 4, the client sends $P_{encr}(PKM, < S >_{skc})$ to the merchant. If the merchant can not get the payment, he would bring his $P_{encr}(PKC, (offer)_{skm})$ and C_S, $C_S Cert$ to TTP. TTP opens C_S and sends the payment to the merchant, meanwhile sends the $P_{encr}(PKC, (offer)_{skm})$ to the client.

4. Fair On-line Gambling

There are different kinds of on-line gambling in the real world, but there are some general characteristics for all gambling systems. In this paper, we only consider the credit-based payment. The following are important characteristics of on-line gambling:

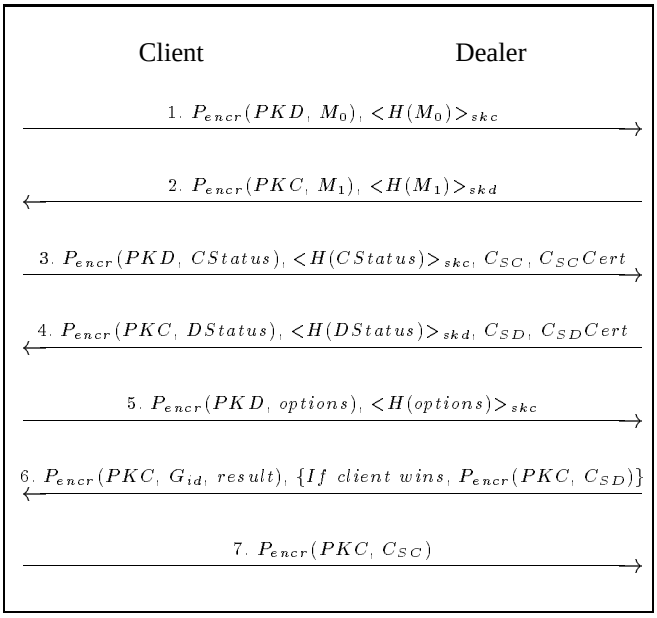
a) Two-way payments involved.
   — Anonymous.
   — Credit Card Payment.
b) Bank is off-line.
c) Trusted Third Party is off-line.
d) Cheating is prevented during whole process.
   — Information must be checked.
   — If there is a dispute, TTP will reslove it.

In the following, we will discuss games from authorized organizations and pure luck games respectively.

4.1. Games from authorized organizations

In this section, we assume that games are from authorized organizations. The most important assumption with on-line casino is that it should not be trusted. Both Client and Dealer must keep some secrets of their choice before they have given their bets. We have developed the protocol as follows:

PROTOCOL OF GAMBLING FROM AUTHORITY



1. Client chooses a game and sends the message to Dealer

   $$M_0 = (GName,\ C,\ tr_c)$$

   where $GName$ is the name of the game to play, C is the client ID, tr_c is $timestamp$ for the message.

2. Dealer prepares the game for the client

   $$M_1 = (Game, P_{encr}(PKT, \\ < Gnmae,\ G_{ID} >_{skd}), tg_c)$$

   $Game$ is the executable program. G_{ID} is the default parameter for the game to run, tg_c is the timestamp for the preparation of the game. For the client, G_{ID} is the secret until the end of the game. $< Gnmae,\ G_{ID} >_{skd}$ is encrypted with TTP's public key PKT. If something is wrong, TTP can decrypt it and get the G_{ID}.

3. Client runs the game, gives his option and prepares his payment slip for betting.

   (a) Client gives his option

   $$GStatus = C,\ M_0,\ M_1, \\ P_{encr}(PKT,\ < option >_{skc})$$

   For Dealer, Client's option is secret until it is necessary to make it public. Client's $< option >_{skc}$ is encrypted with TTP's public key PKT and has the signature of Client. If necessary, TTP can open it.

   (b) Client prepares payment slip

   $$S_C =< CStatus >_{skc}$$
   $$CStatus = (GStatus,\ \mathcal{CC},\ D,\ \$AC,\ tcb_c)$$

   where $\mathcal{CC}$ is the credit information of Client, D is dealer identification, $\$AC$ is quantity of money, and tcb_c is timestamp.

   Encrypted Payment slip:

   $$C_{SC} = P_{encr}(PKT, S_C).$$

   The payment slip C_{SC} contains the information of game status, information of credit and information of betting. With techniques of $PEDLDLL$ discussed in the second section of this paper, $C_{SC}Cert$ could be constructed for checking that C_{SC} is the encrypted payment slip.

4. Dealer prepares his payment slip based on client's betting,

   $$S_D =< DStatus >_{skd}$$
   $$DStatus = (CStatus,\ \mathcal{CD},\ \$AD,\ tdb_c)$$

   where $\mathcal{CD}$: credit information of Dealer; $\$AD$: quantity of money; tdb_c: timestamp.

   Encrypted payment slip:

   $$C_{SD} = P_{encr}(PKT, S_D)$$

   The payment slip C_{SD} contains the current information of game status, credit and betting. With techniques of $PEDLDLL$, $C_{SD}Cert$ is constructed for ckecking that C_{SD} is the encrypted payment slip.

5. Client sends his option to Dealer by message

   $$P_{encr}(PKD,\ option),\ \ < H(option) >_{skc}$$

   The message is encrypted with Dealer's public key PKD. Dealer can read the message and know Client's option. Based on Client's options and G_{ID}, Dealer can get the result of the game.

6. Dealer sends Client token $P_{encr}(PKC,\ G_{ID},\ result)$, the Client can get G_{ID}. At this time, Client knows both the G_{ID} and his option, he can run the game and get the result. If Client wins, Dealer also sends Client token $P_{encr}(PKC,\ C_{SD})$, Client gets the payment slip C_{SD}.

7. If Dealer wins, Client sends Dealer $P_{encr}(PKC,\ C_{SC})$ and Dealer gets payment slip C_{SC}).

Whole process of the gambling and payment is fair in the above protocol. Before step 5, both the Dealer and Client can not get the result of the game. Dealer has G_{ID} as secret, Client has $options$ as secret. Both of them encrypt their secrets with TTP's public key at first. They make their secrets public in step 5 and step 6. If the loser refuses to pay, the winner can bring encrypted payment slip C_{SC} and C_{SD} to TTP. TTP can then open them and check the result of game. C_{SC} contains the information of Client's betting. C_{SD} contains the information of Dealer's betting. Both of them are necessary for the TTP to check the betting process and result. Based on checking result, TTP can forward S_C to Dealer if Dealer wins. Forward S_D to Client if Client wins. The whole process is fair for both Client and Dealer.

Before Client has sent his C_{SC}, Client has the right to quit the protocol. Before Dealer sends his C_{SD}, Dealer has the right to quit the protocol. In above cases, both of them have no peer's encrypted payment slip, so they can not get any payment or useful information of the game. the protocol is aborted but the process is fair. If one party has peer's encrypted payment slip, he can bring both C_{SC} and C_{SD} to TTP. The protocol can finish with the help of TTP. TTP can get all information of the game and betting from C_{SC} and C_{SD}. TTP can get the result of the game and forward the payment to the winner.

The above protocol can be extended in a real application. In step 3, Client perhaps discloses part of his options for the game to progress or prepare encrypted payment slips for betting. In step 4, Dealer perhaps provides some information of the current game or prepare encrypted payment slip to response Clients betting. In step 5, Client makes his current options public. Payments are given if the Client chooses to trust the Dealer. This kind of processes can repeat again and again until the end of the game or the Client chooses not to trust the Dealer. The Dealer sends Client G_{ID}, Client can run the game on his local machine to check the whole process of gambling. If a cheating occurs, he can bring all encrypted payment slips to TTP to prove that the Dealer is cheating. The whole process is fair in this case.

4.2 Pure Luck Games

Many casino games are solely games of chance [11]. This kind of games can be abstracted as generating of a random number. We assume that there are only two parties for the pure luck game. They will cooperate with each other to generate the random number, during the process, they will bet on the result of the random number. In this part, we will discuss the fainess both of game process and the dual-payment between two parties.

The following is the two-party protocol for generating a random number and how they bet and arrange payment on the output of the random number. We out-line the protocol as:

1. Alice generates a random number and signs the hash of the random number, Alice sends Bob:

$$M_1 = H(R_A),\ < H(R_A) >_{ska},\ P_{encr}(PKT,\ < R_A >_{ska})$$

$P_{encr}(PKT,\ < R_A >_{ska})$ is the TTP's public key encryption of the random number R_A with the signature of Alice.

2. Bob generates a random number and signs the hash of the random number, Bob sends Alice:

$$M_2 = H(R_B), < M_1, H(R_B) >_{skb},\ P_{encr}(PKT,\ < R_B >_{skb})$$

$P_{encr}(PKT,\ < R_B >_{skb})$ is the TTP's public key encryption of the random number R_B with the signature of Bob.

3. Alice prepares her encrypted payment slip and the certificate of the encrypted payment slip

$$S_A =< A, \mathcal{CA}, B, M_1, M_2, Abetting, t_a >_{ska}$$

where A is Alice's identification; $\mathcal{CA}$ is credit information (defined in section 3.1) ; B is Bob identification; M_1 and M_2 are messages of step 1 and step 2. $Abetting$ contains Alice's betting options and amount of money for this betting. t_a is timestamp.

Encrypted Payment slip is:

$$C_{SA} = P_{encr}(PKT, S_A)$$

With the technique of $PEDLDLL$, Alice counstructs certificate of the encrypted payment slip $C_{SA}Cert$.

Alice sends Bob

$$M_3 = (ABetting, C_{SA}, C_{SA}Cert)$$

4. Bob prepares his encrypted payment slip and the certificate of the encrypted payment slip

$$S_B =< B, \mathcal{CB}, A, M_3, Bbetting, t_b >_{skb}$$

where B is Bob's identification; $\mathcal{CB}$ is credit information (defined in section 3.1) ; A is Alice's identification; M_3 is the message of last step. $Bbetting$ contains Bob's response of Alice's betting which contains Bob's amount of money on this betting. t_b is timestamp.

Encrypted Payment slip is:

$$C_{SB} = P_{encr}(PKT, S_B)$$

With the technique of $PEDLDLL$, Bob counstructs certificate of the encrypted payment slip $C_{SB}Cert$.

Bob sends Alice

$$M_4 = Bbetting, C_{SB}, C_{SB}Cert$$

5. Alice sends Bob the actual value of number R_A:

$$M_5 = R_A, < M_4, R_A >_{ska}$$

6. Bob sends Alice the actual value of number R_B:

$$M_6 = R_B, < M_5, R_B >_{skb}$$

7. Both Alice and Bob computes the random number

$$R = R_A \ XOR \ R_B$$

8. If Alice loses, Alice sends Bob her payment slip S_A; if Bob loses, Bob sends Alice his payment slip S_B.

In this protocol, Alice and Bob sends their hashes at first, then they give their betting and prepare their payment slips. They encrypt their payment slips with TTP's public key and construct certificates for verifying the encrypted payment slips. They send their bettings, encrypted payment slips and certificates for verifying their encrypted payment slips. In this step, Alice and Bob have given their bettings with payments and can not change, but they can not get the money at this time because payment slips are encrypted with TTP's public key. In step 5 and 6, they send their actual chosen values to peer party, then both Alice and Bob can compute the number.

The protocol is totally fair for both Alice and Bob. If the loser refuses to pay, the winner could bring Encrypted payment slips both C_{SA} and C_{SB} to TTP. TTP can open them and check the process of betting. TTP can then forward payment slip to the winner. Alice can quit the protocol at or before step 3, Bob can quit the protocol at or before step 4. At any other time, if one party stop the protocol, the peer party can bring C_{SA} and C_{SB} to TTP, the protocol can continue the the end (the winner receives the payment) with the help of TTP.

5. Conclusion

In this paper, we have exploited PEDLDLL techniques to develop fair protocols for on-line gambling. We have discussed the general fair-exchange protocol with credit payment. Base on general fair-exchange protocol, we have presented protocols for games from authority organization and pure luck games which are useful in implementing some real on-line games. Our protocols can guarantee the fairness of both the games and payments. The major feature of our protocols lies in the using of encrypted payment slips and certificates of the encrypted payment slips. The credit information is anonymous in our protocols. Our protocols are only suitable for some of on-line games, there are many open problems for the fairness of on-line gambling, for example, if there are more than two people invloved in a game, how to deal with the issue of collusion? How to design fair protocol for different kinds of games such as card games [12]? There are a lot of interesting issues to study on on-line gambling.

Acknowledgement

The authors would like to thank Dr. Chunkun Wu, Australian National University, for his valuable discussions and comments on this work.

References

[1] N. Asokan, M. Schunter and M. Waidner, "Optimistic protocols for fair exchange", Proceedings of 4th ACM Conference on Computer and Communications Security, Zurich, Switzerland, pp.6-17, 1997, pp.6-17.

[2] J. Zhao and D. Gollmann, "An efficient non-repudiation protocol", Proceeding of 10th IEEE Computer Security Foundations Workshop, Rockport, Massachusetts, June 1997, pp.126-132.

[3] F. Bao, R. Deng and W. Mao, "Efficient and Practical Fair Exchange Protocols with Off-line TTP", 1998 IEEE Symposium on Security and Privacy, 1998, pp.77-85.

[4] S. Even, O. Goldreich and A. Lempel, "A Randomized Protocol for Signing Contracts", CACM, Vol.28, No. 6, 1985, pp.637-647.

[5] Y. Mu and V. Varadharajan, “A new scheme of credit based payment for electronic commerce”, the Proceedings of 23rd Local Area Networks Conference, IEEE Computer Society, October, Boston, 1998, pp.278-284.

[6] G. Medvinsky and B. C. Neuman, “iKP - a family of secure electronic payment protocols”, 1995. http://www.zurich.ibm.com/Technology/Security/extern/ecommerce/.

[7] “Secure Electronic Payment Protocol”, 1995, http://www.mastercard.com/SET/.

[8] D.Chaum and T.P.Pedersen, “Wallet databases with observers”, Advances in Cryptology -CRYPTO’92 Proceedings, Springer-Verlag, 1992, pp.89-105.

[9] E. R. Verheul and H. C. A. van Tilborg, “Binding Elgamal: A fraud-detectable alternative to key-escrow proposals”, Advances in Cryptology - EUROCRYPTO’ 97 Proceedings, Springer-Verlag, 1997, pp.119-133.

[10] M. Stadler, “Publicly verifiable secret sharing”, Proceeding of Eurocrypto’ 96, LNCS 1070, Springer-Verlag, 1996, pp.190-199.

[11] C. Hall and B. Schneier, “Remote Electronic Gambling”, IEEE, 1997, pp.232-238.

[12] S. Fortune and M. Merritt, “Poker Protocols”, Advances in Cryptology - Proceeding of CRYPTO 84, Springer-Verlag, 1985, pp.454-464.