Contributions to Mental Poker

Submitted to Universitat Autònoma de Barcelona
in partial fulfillment of the requirements for the
degree of Doctor of Philosophy in Computer Science

by Jordi Castellà-Roca
May 2005

Abstract

Computer networks and especially the Internet have allowed some common activities such as shopping or gambling to become remote (e-shopping and e-gambling). The poker game played over a network is known as mental poker. The problem with mental poker is the difficulty of keeping it practical while guaranteeing the same standards of security, fairness and auditability offered by standard casinos for physical poker. The important aspects to take into account when designing mental poker protocols are: functionality, security, and computational and communication cost. Proposals in the literature usually focus on the first two items only. This makes comparisons difficult. This thesis starts with a formal cost analysis of the main proposals in the literature. The analysis is not limited to costs, though; security is also analyzed and, in fact, our study detected a fundamental weakness in one of the compared mental poker protocols. The attack is presented in a separate chapter after the global comparative analysis. The three following chapters of this thesis present three new protocols that enhance the proposals in the literature in different ways. The first proposal belongs to the family of TTP-free protocols and does not preserve the confidentiality of player strategies; it reduces the computational cost by avoiding the use of zero-knowledge proofs. The second proposal is TTP-free, preserves the confidentiality of player strategies and reduces the computational cost by requiring players to perform less mathematical operations. The third proposal addresses a novel functionality usually not offered in the literature, namely player dropout tolerance, *i.e.* the ability to continue the game even if some players leave it.

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.

May 2005

Dr. Josep Domingo-Ferrer
(Adviser)

Dr. Francesc Sebé Feixas
(Adviser)

Dr. Joan Borrell Viader
(Tutor)

Acknowledgements

I wish to thank all the people who helped and encouraged me during the development of this thesis.
My sincere gratitude goes to Josep Domingo-Ferrer who accepted me in the CRISES research group, and jointly with Francesc Sebé advised this thesis with invaluable patience and never ending support to my research and related activities. Thanks Josep and Francesc.
The long journey of a thesis begins with a first step. I am also grateful to Joan Borrell because he encouraged to me to take this first step.
I would also like to thank all members of the CRISES research group, Josep Ma Mateo, Antoni Martínez, Anna Oganian, Carles Vallvé, Agustí Solanas and Susana Bujalance for their company and friendship.
I also wish to mention Jordi Herrera for his lessons and their friendship.
Andreu Riera must also be thanked for giving me the opportunity to be initiated on the hard way of the research.
Last but not least, I am indebted to my wife Cristina, my mother Tresina, my father Joan and my brother Joan for their unconditional support and encouragement.

Contents

List of Tables

Chapter 1

Introduction

1.1 Situation

The growth of the computer networks has allowed many activities that were ususally made physically to become remote, such as shopping, information search o gambling.

We concentrate on gambling over a computer network, also called e-gambling. The drawback of e-gambling is the difficulty of guaranteeing the same standards of security, fairness and auditability offered by physical gambling. Since each game has his different rules, every game needs specific security measures. Casino games fall into three groups according to their security requirements:

- Random draw games, with a single draw (*e.g.* dice, roulette) or with multiple draws (*e.g.* bingo, keno).

- Games where a value or a set of values are obtained in a non-secret way. Games where cards are visible (*e.g.* blackjack) fall into this category.

- Games where a value or a set of values are obtained in a secret way. Games where cards are reversed (*e.g.* poker) fall into this category.

Our contributions are focused to the third and more complex category, *i.e games where a value or a set of values are obtained in a secret way.* In cryptography this problem is known as *mental poker.*

The main contributions in the literature can be divided into two main groups: TTP-based and TTP-free.

In general, TTP-based proposals are computationally efficient and are usable in practice. However, some authors argue that a TTP is neither desirable nor realistic. The TTP is often in a privileged position, because it manages the game and participates in it.

TTP-free proposals are more desirable as far as security is concerned, but they have non-negligible computational and communication costs.

No formal comparative study exists in the literature on the computational efficiency of mental poker protocols. Such a study should take care of the following items:

Computational cost : Cryptographic protocols use modular exponentiation and multiplication as basic operations. The number of these operations determines the computational cost of a cryptographic protocol;

Communications cost : The communications cost can be split into two components:

- **Number of messages** : Sometimes the time used to open a communication and send a message is not negligible; the number of messages accounts for this cost;
- **Total length of messages** : The amount of information sent during the protocol also is an indication of efficiency: a great volume of transmitted data results in little efficiency.

1.2 Objectives

A first objective of this thesis is to undertake a formal study of the efficiency of the main contributions to mental poker, based on the above items.

Security must not be forgotten in an efficiency comparison, though. The reason is that we cannot compare two proposals with different security properties. The study must also evaluate the security properties of each protocol. Thus, the second objective of this thesis is to study the security properties of the available contributions.

The third objective of the thesis is the design of secure and efficient mental poker proposals to advance the state of the art.

Security and efficiency of mental poker must be increased without reducing functionality. Two relevant functionalities are the following:

Confidentiality of player strategies : In the poker game, it is very important that the losing players may keep their cards secret at the end of a hand. The whole concept of bluffing is based on this fact.

Player dropout : If one player leaves the game, the remaining players should be able to continue playing.

Some proposals provide these two functionalities, but they open the possibility that a coalition of several players discovers the cards of other players.

Thus, the fourth objective of this thesis is to design a secure mental poker protocol providing confidentiality of player strategies, dropout tolerance and player security.

1.3 Structure of this thesis

This thesis is organized as follows.

Chapter 2 presents the notation and basic concepts used in the following chapters.

Chapter 3 presents a comparative analysis of mental poker protocols in the literature. Insofar as this long chapter exhaustively compares the performance and the security of published mental poker methods, it constitutes an original contribution in its own right. To the best of our knowledge, no such comparative survey was available up to this date.

Chapter 4 presents an attack that exploits a security flaw of one of the mental poker protocols analyzed in Chapter 3. In fact, we found this flaw when performing the comparative analysis. The authors of the broken protocol have presented a

modification of their protocol. Nevertheless, the new proposal still has an important security flaw, which is also described in the chapter.

Chapter 5 presents a new mental poker protocol that falls in the category of TTP-free protocols that do not preserve the confidentiality of player strategies. It reduces the computational cost by avoiding the use of zero-knowledge proofs. Especially remarkable is the representation used for cards and card permutations, which allows permutation of an encrypted card using an additive and multiplicative homomorphic cryptosystem. This protocol has been patented by Scytl Online World Security S.A. Moreover, it has been implemented in a case study of mutual distrust. The authors of the implementation argue that our protocol is "practical in terms of computational requirements" as compared to the rest of proposals in the literature.

Chapter 6 presents a new mental poker protocol that does not require a TTP and preserves the confidentiality of the strategy of players. The amount of computation required stays reasonably low. We present a cost analysis and we compare the resulting cost with one of the most efficient previous proposals. The security of the proposal is analyzed and it is shown that it fulfills all security properties usually required for mental poker protocols. We conclude that the protocol is perfectly usable in practice, unlike most previous TTP-free solutions.

Chapter 7 presents our solution for player dropout in mental poker without a TTP. The solution is based on zero-knowledge proofs and allows the game to continue after dropout. Unlike prior contributions, a player coalition cannot know the cards in the hand of the rest of players. Moreover, the number of players that can leave the game is not limited. We give a theoretical assessment of the security of the proposal.

The concluding remarks and a summary of the results presented in this thesis can be found in Chapter 8. Some guidelines for future research are also hinted.

Chapter 2

Notation and basic concepts

In this chapter we introduce the notation and the basic cryptographic concepts used in the rest of this thesis.

2.1 Notation

The following notation is used in order to describe the protocols presented or analyzed.

- P_{entity}, S_{entity}: Asymmetric key pair of *entity*, where P_{entity} is the public key and S_{entity} is the private key.
- $S_{entity}(m)$: Digital signature of message m by *entity*, where digital signature means computing the hash value of message m using a collision-free one-way hash function and encrypting this hash value under the private key of *entity*.
- $E_{entity}(m)$: Encryption of message m under the public key of *entity*.
- $D_{entity}(c)$: Decryption of message c under the private key of *entity*.
- $H(m)$: Hash value of message m using a collision-free one-way hash function.

- $m_1|m_2$: Concatenation of messages m_1 and m_2.
- K_{entity}: Secret symmetric key of *entity*.
- $E(K_{entity}, m)$: Encryption of message m under the symmetric key of *entity*, K_{entity}.
- $D(K_{entity}, c)$: Decryption of message c under the symmetric key of *entity*, K_{entity}.

2.2 Basic concepts

In this section we introduce some definitions and basic concepts that we use in subsequent protocol descriptions.

2.2.1 Definitions

Definition 1 *Let $a \in \mathbb{Z}_n^*$. a is said to be a* quadratic residue modulo *if there exists an $x \in \mathbb{Z}_n^*$ such that*

$$x^2 \bmod n \equiv a \bmod n \tag{2.1}$$

Otherwise, a is a quadratic nonresidue modulo n. Any x satisfying Equation (2.1) is a square root of a modulo n.

The set of all quadratic residues modulo n is denoted by Q_n, and the set of all quadratic non-residues is denoted by $\overline{Q}_n$.

Definition 2 *Let p be an odd prime and a an integer. The Legendre symbol $\left(\frac{a}{p}\right)$ is defined to be*

$$\left(\frac{a}{p}\right) = \begin{cases} 0, & if \quad p|a \\ 1, & if \quad a \in Q_p \\ -1, & if \quad a \in \overline{Q}_p \end{cases}$$

Theorem 1 *Suppose p is an odd prime. For any integer $a \geq 0$ the Legendre symbol can be computed as follows:*

$$\left(\frac{a}{p}\right) \equiv a^{(p-1)/2} (\mathrm{mod} p)$$

Definition 3 *Let $n \geq 3$ be odd with prime factorization $n = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}$, and a an integer. Then the Jacobi symbol $\left(\frac{a}{n}\right)$ is defined from the Legendre symbol as*

$$\left(\frac{a}{n}\right) = \left(\frac{a}{p_1}\right)^{e_1} \left(\frac{a}{p_2}\right)^{e_2} \cdots \left(\frac{a}{p_k}\right)^{e_k}$$

Lemma 1 *Given x, $y \in \mathbb{Z}_n^*$ such that $x^2 \equiv y^2 \bmod n$, and $x \neq \pm y \bmod n$, there is a polynomial-time algorithm to factor n. (The* gcd *of n and $x \pm y$ is a factor of n).*

Lemma 2 *Let $n = pq$ such that $p \equiv q \equiv 3 \bmod 4$. For all $x, y \in \mathbb{Z}_n^*$, if $x^2 \equiv y^2 \bmod n$ and $x \neq y \bmod n$ then $\left(\frac{x}{n}\right) = -\left(\frac{y}{n}\right)$.*

In [MvOV96] we can find the following procedure for computing square roots modulo a prime p where $p \equiv 3(\mathrm{mod}4)$. Let $a \in Q_p$, where $p \equiv 3(\mathrm{mod}4)$ and p is an odd prime.

Procedure 1 (a, p)

1. *Compute $r = a^{(p+1)/4} \bmod p$;*
2. *Return $(r, -r)$.*

In [MvOV96] we can find the following procedure for computing square roots modulo a prime p where $p \equiv 5(\mathrm{mod}8)$. Let $a \in Q_p$, where $p \equiv 5(\mathrm{mod}8)$ and p is an odd prime.

Procedure 2 (a, p)

1. *Compute $d = a^{(p-1)/4} \bmod p$;*
2. *If $d \equiv 1$ then compute $r = a^{(p+3)/8} \bmod p$;*
3. *If $d \equiv p$ then compute $r = 2a(4a)^{(p-5)/8} \bmod p$;*

4. *Return* $(r, -r)$.

In [MvOV96] we can find the following procedure for computing square roots modulo a prime p. Let $a \in Q_p$, and p is an odd prime.

Procedure 3 (a, p)

1. *Choose random* $b \in Z_p$ *until* $b^2 - 4a$ *is a quadratic non-residue modulo* p*, i.e.* $\left(\frac{b^2-4a}{p}\right)$*;*

2. *Let* f *be the polynomial* $x^2 - bx + a$ *in* $Z_p[x]$*;*

3. *Compute* $r = x^{(p+1)/2} \bmod f$*;*

4. *Return* $(r, -r)$.

In [MvOV96] we can find the following procedure for computing square roots modulo $n = pq$ given its prime factors p and q. Let $a \in Q_p$, where $n = pq$ and p and q are prime numbers.

Procedure 4 (a, n, p, q)

1. *Use Procedure 3 (or Procedures 1 or 2 if applicable) to find the two square roots* r *and* $-r$ *of* a *modulo* p*;*

2. *Use Procedure 3 (or Procedures 1 or 2 if applicable) to find the two square roots* s *and* $-s$ *of* a *modulo* q*;*

3. *Use the extended Euclidean algorithm to find integers* c *and* d *such that* $cp+dq = 1$*;*

4. *Let* $x = (rdq + scp) \bmod n$ *and* $y = (rdq - scp) \bmod n$*;*

5. *Return* $(\pm x, \pm y)$*;*

2.2.2 Zero-knowledge proofs

A zero-knowledge protocol allows a prover to demonstrate knowledge of a secret while revealing no information that can be used by the verifier to convey this demonstration of knowledge to third parties.

We can define very informally a zero-knowledge proof as a technique that allows a prover to convince the verifier about the truth of some specific statement, but at the end of the protocol, the verifier has no idea how to prove the statement to himself or to third parties.

For more rigorous definitions of zero-knowledge proofs, see [GMR89],[BC90] or [MvOV96].

We next recall some zero-knowledge proofs that are used in the rest of the thesis.

Proof of knowledge of a discrete logarithm

Let p a prime number, where $p = 2q + 1$ and q is a prime number. The following protocol [Sch91] allows a prover to convince a verifier that, given $y = g^{\alpha} \bmod p$, the prover knows α:

1. The prover sends $a = g^{\omega} \bmod p$ to the verifier for some random value $\omega \in \mathbb{Z}_q$;
2. The verifier responds by sending a random challenge $c \in \mathbb{Z}_q$;
3. The prover responds with $r = \omega + \alpha c \pmod{q}$;
4. The verifier checks whether $g^r \bmod p \stackrel{?}{=} ay^c \bmod p$.

We shall denote this protocol by $CP(y, g; \alpha)$ or $CP(y, g)$ when the value α is not relevant.

Proof of equality of discrete logarithms

Let p a prime number, where $p = 2q + 1$ and q is a prime number.

Given $u = g^{\alpha} \bmod p$ and $v = y^{\beta} \bmod p$, the following protocol [CP92] allows a prover to convince a verifier that the prover knows α, β and that $\alpha = \beta$ holds, where g and y have order q.

1. The prover sends $(a, b) = (g^\omega, y^\omega)$ to the verifier for some random value $\omega \in \mathbb{Z}_q$;
2. The verifier responds by sending a random challenge $c \in \mathbb{Z}_q$;
3. The prover responds with $r = \omega + \alpha c \pmod{q}$;
4. The verifier checks whether $g^r \bmod p \stackrel{?}{=} au^c \bmod p$ and $y^r \bmod p \stackrel{?}{=} bv^c \bmod p$.

We shall denote this protocol by $CP(g, y, u, v; \alpha)$ or $CP(g, y, u, v)$ when the value α is not relevant. Note that this proof is easily generalizable to prove equality of an arbitrary number of discrete logarithms.

d-out-of-n proof of knowledge

In [CDS94] a solution is presented which allows a prover to show that she can corretly perform at least d executions out of a set of n zero-knoledge problem instances without revealing which.

2.2.3 n-out-of-n threshold ElGamal encryption

This is a multi-party protocol [DF90] between n parties in which they generate a single public key y. The corresponding unknown private key α is distributed in n shares α_i.

Key generation

Let p a prime number, where $p = 2q + 1$ and q is a prime number.

Each player generates a random private key $\alpha_i \in \mathbb{Z}_q$ and publishes $y_i = g^{\alpha_i}$. The public key is formed as $y = \prod_{i=1}^{n} y_i = g^\alpha$, where $\alpha = \alpha_1 + \cdots + \alpha_n$.

Message encryption

Message encryption is done using the ElGamal cryptosystem[ElG85]. Given a message m and a public key y, a random value r is generated and the ciphertext is computed as

$$E_y(m, r) = (c1, c2) = (g^r, m \cdot y^r)$$

We shall denote this encryption by $E_y(m, r)$ or $E_y(m)$ when the value r is not relevant.

Message decryption

Given a message encrypted with public key y, $E_y(m, r) = (c_1, c_2) = (g^r, m \cdot y^r)$, a decrypter j can confidentially obtain m as follows. Each party $i \neq j$ publishes $c_1^{\alpha_i}$. The message m is computed by participant j as

$$m = \frac{c_2}{c_1^{\alpha_j}(\prod_{i \neq j} c_1^{\alpha_i})}$$

This decryption can be rendered verifiable by each participant i by performing $CP(g, c_1, y_i, c_1^{\alpha_i}; \alpha_i)$.

2.2.4 ElGamal re-masking

Given a ciphertext $E_y(m)$, it can be re-masked by computing $E_y(m) \cdot E_y(1, r)$ for $r \in \mathbb{Z}_q$ randomly chosen, where $\cdot$ means componentwise scalar product —ElGamal ciphertexts can be viewed as vectors with two components. The resulting ciphertext corresponds to the same cleartext m.

Chapter 3

A comparative survey of mental poker protocols

Mental poker is played like ordinary poker but without physical elements (like cards) nor verbal communication; all exchanges between players must be accomplished using messages [Den83]. Any player may try to cheat.

A mental poker protocol must guarantee the fairness of the game and, if a player tries to cheat, the protocol must detect or avoid the cheating. In [Cré85], Crépeau enumerated the requirements and properties that must be met by a mental poker protocol.

Uniqueness of cards: Traditional decks of cards can be verified before the game starts, and players can be assured that there are not duplicate cards. In a mental poker protocol players should be able to verify that each card appears once and only once.

Uniform random distribution of cards: In a traditional hand of poker, one player shuffles the deck and the rest of players can see it. Cards are uniform randomly distributed, so that the card set of one player does not depend on the opponents' actions because the latter have no control on the shuffled deck. The hand of each player depends on decisions made by every player.

Cheating detection with a very high probability: A mental poker protocol must detect any attempt to cheat, *e.g* seeing a face-down card, changing a face-up card, etc.

Complete confidentiality of cards: If the deck is face-down then no partial or total information about any card from the deck ought to be disclosed. Also when a player draws a card, the rest of players should not be able to get information on that card.

Minimal effect of coalitions: A secret communication channel between the players of a coalition is possible in mental poker, *e.g.* one player can ring another player to tell her her cards. A mental poker protocol should reduce the effect of coalitions, so that if a player is not cheating then nobody can learn more about her hand, or about the cards in the deck, than what they can infer form the cards in their coalition.

Complete confidentiality of strategy : It is strategically very important in the game of poker that the losing players may keep their cards secret at the end of a hand. The whole concept of bluffing is based in this fact.

The last security requirement is that a mental poker protocol ought to be TTP-free.

Absence of trusted third party : It is not realistic to rely on a trusted third party, since any human can be bribed, and no machinery is entirely safe because no fully tamper-proof device has yet been produced.

Nonetheless, there are authors who argue the need of a TTP in a mental poker protocol. The main reasons are fairness and protocol efficiency.

Fairness : In [CY02] the following fact is justified; *Without a TTP, the fairness of card dealing in the mental poker game is uncertain.*

Efficiency : An implementation of the TTP-free protocol in [Cré86] on three Sparc workstations took eight hours to shuffle a deck [Edw94]. This time is not practical in a real hand.

In next sections, the main contributions to mental poker protocols are divided into those using a TTP and those that are TTP-free.

3.1 Protocol analysis

A mental poker protocol is not a single protocol but a suite of subprotocols, because there is a subprotocol for each action. Most mental poker protocols specify subprotocols for the following actions:

- Shuffling the deck
- Drawing a card
- Discarding a card
- Shuffling a discarded card
- Opening a card

Nevertheless, other contributions only specify subprotocols for the two most basic actions: shuffling the deck and drawing a card. We have decided to describe contributions in terms of the following subprotocols:

Preparation : Steps done before game starts;

Deck shuffling : Steps done by players when the deck is shuffled;

Card drawing : Steps done when a player extracts a card from the deck.

In our study we analyze the following items for each protocol (this analysis is only feasible when there is enough detail in the description):

- The number of messages;
- The total length of messages;

- The computational cost.

Based on our knowledge this is the first complete study about the main contributions in mental poker, that presents theoretical results about these three items. With this information we can state if one protocol is more or less efficient than other protocol.

Furthermore, we have analyzed the security properties of each protocol. We have take the requirements and properties enumerated in [Cré85] as reference.

Wherever we have made some assumptions, these are justified. For instance, if a player publishes or writes a message in a board we have assumed that $n-1$ messages have been sent.

The following notation has been used in the analysis:

- ξ: time cost of one modular exponentiation;
- ρ: time cost of one modular product;
- ϵ: negligible time cost;
- $[p]$: number of bits of one value x in Z_p or Z_n;
- $[r]$: number of bits that are used to represent a value r in $\{1, \ldots, 52\}$, where $6 \leq [r]$. We assume that the bitlength of one permutation π of 52 values is denoted as $52[r]$;
- $[S(m)]$: number of bits of the digital signature on m;
- $[P(m)]$: number of bits of the encryption of m;
- $[H(m)]$: number of bits of the hash on m;
- $[m]$: number of bits of a message m;
- k: is the number of shares in which a secret is divided;
- s: security parameter;
- n: number of players.

3.2 Mental poker with a TTP

In this section we describe the main mental poker protocols using a TTP. These contributions follow the order in which they were published.

3.2.1 Poker protocols (Fortune-Merritt)

Fortune *et al.* in [FM85] presented a mental poker Protocol using a TTP called Card Salesman. The Card Salesman only participates at the beginning of the hand by choosing a secret permutation π and receiving in a secure way from every player as many permutations as there are players. The Card Salesman composes π and the permutations from players, so that the final permutation is the shuffled deck of cards. For every player, the Card Salesman computes the information needed by that player to take part in the game. To authenticate the information, the Card Salesman uses a one-way function.

Let us assume that the number of players is n, and $\mathcal{P}_i$ is the i-th player in the ordered set of n players.

Protocol 1 (Card Shuffling)

1. *The Card Salesman randomly chooses a permutation* π*;*
2. *For each* $\mathcal{P}_i$ *in* $\{\mathcal{P}_1, \ldots, \mathcal{P}_n\}$ *do:*
   (a) $\mathcal{P}_i$ *chooses* n *permutations* $\{\pi_{i,1}, \ldots, \pi_{i,n}\}$ *of* 52 *elements;*
   (b) $\mathcal{P}_i$ *secretly transmits* $\{\pi_{i,1}, \ldots, \pi_{i,n}\}$ *to the Card Salesman;*
   (c) $\mathcal{P}_i$ *encrypts the permutations using a one-way function, and broadcasts the resulting cryptograms;*
3. *The Card Salesman does:*
   (a) *For* $i = 1$ *to* n *compute* $\pi_i = \pi_{i+1,i}^{-1} \circ \pi_{i+2,i}^{-1} \cdots \pi_{n,1}^{-1} \circ \pi_{1,i}^{-1} \cdots \pi_{i,i}^{-1} \circ \pi^{-1}$*, where* $i \in \{1, \ldots, n\}$*;*
   (b) *Broadcast* $\{\pi_1, \ldots, \pi_n\}$.

Let us assume that $\mathcal{P}_i$ draws a card.

Protocol 2 (Card Drawing)

1. *$\mathcal{P}_i$ chooses $y = \pi(x)$ not in any player's hand and broadcasts y and $\pi_i(y)$;*
2. *For each $\mathcal{P}_j$ in $\{\mathcal{P}_{i+1}, \mathcal{P}_{i+2}, \ldots, \mathcal{P}_n, \mathcal{P}_1, \ldots, \mathcal{P}_{i-1}\}$, $i, j \in 1, \ldots, n$, and following the specified order do:*
   (a) *receive x_{j-1} from the previous player;*
   (b) *compute $x_j = \pi_{j,i}(x_{j-1})$;*
   (c) *send x_j to the following player;*
3. *$\mathcal{P}_i$ receives x_{i-1} from $\mathcal{P}_{i-1}$;*
4. *$\mathcal{P}_i$ computes $x = \pi_{i,i}(x_{i-1})$;*
5. *All players record that $\mathcal{P}_i$ has got $y = \pi(x)$ in his hand.*

The proposal [FM85] does neither present an opening protocol nor a discarding protocol. At end of the game, each player publishes her permutations, and checks that every other player played fairly.

Protocol analysis

We have made some assumptions that are detailed next. When the protocol specifies that a message is broadcast to n users, *i.e.* $n-1$ players and the TTP, we assume that n messages are sent.

In Step 2b of Protocol 1, $\mathcal{P}_i$ secretly sends to the TTP a message. We assume that $\mathcal{P}_i$ uses a secure channel (for instance [FKCK96]) instead of encrypting the message.

In Table 3.3 we summarize the security properties satisfied by the Fortune-Merritt protocol. It can be concluded that the final publication of the players' permutations reveals their strategy.

Table 3.1: Costs of the Fortune-Merritt shuffling protocol

	Number of messages		Total length of messages		Computational cost	
	$\mathcal{P}_i$	TTP	$\mathcal{P}_i$	TTP	$\mathcal{P}_i$	TTP
Card shuffling	$n+1$	n	$n(52[r]+[H(m)])$	$n52[r]$	ϵ	ϵ
Step 1						ϵ
Step 2	$n+1$		$n(52[r]+[H(m)])$		ϵ	
Step 2a					ϵ	
Step 2b	1		$n52[r]$		ϵ	
Step 2c	n		$n[H(m)]$		ϵ	
Step 3		n		$n52[r]$		ϵ
Step 3a						ϵ
Step 3b		n		$n52[r]$		ϵ

Table 3.2: Costs of the Fortune-Merritt drawing protocol

	Number of messages		Total length of messages		Computational cost	
	$\mathcal{P}_i$	TTP	$\mathcal{P}_i$	TTP	$\mathcal{P}_i$	TTP
Card drawing	$2n-1$		$(2n-1)[r]$		ϵ	
Step 1	n		$n[r]$		ϵ	
Step 2	$n-1$		$(n-1)[r]$		ϵ	
Step 2a					ϵ	
Step 2b					ϵ	
Step 2c	1		$[r]$		ϵ	
Step 3					ϵ	
Step 4					ϵ	
Step 5					ϵ	

Table 3.3: Security properties of the Fortune-Merritt protocol suite

Uniqueness of cards	X
Uniform random distribution of cards	X
Cheating detection with a very high probability	X
Complete confidentiality of cards	X
Minimal effect of coalitions	X
Complete confidentiality of strategy	

3.2.2 Remote electronic gambling (Hall-Schneier)

Hall in [HS97] introduces an audit trail. If a player suspects that another player is cheating, she can use the audit trail to verify it. These audit trails are based on hash chains. The concept of hash chain was introduced in [Lam81]. A hash chain is digitally signed so that it can be used to convince a judge. The outcome of the game is determined by players and the TTP. The TTP chooses a random permutation of the deck, commits to the permutation and sends the output of the commitment to players. Every player receives the TTP commitment, generates a random permutation of the deck, signs the permutation and the TTP commitment, encrypts her permutation and TTP commitment and sends encrypted values to the TTP. The TTP decrypts the player permutations and composes her permutation with them. The resulting permutation is the shuffled deck of cards.

Let the number of players be n and let $\mathcal{P}_i$ be the i-th player in the ordered set of n players.

Protocol 3 (Initialization)

1. *Each player $\mathcal{P}_i$ has a certified key pair $(P_{\mathcal{P}_i}, S_{\mathcal{P}_i})$, where $P_{\mathcal{P}_i}$ is the public key and $S_{\mathcal{P}_i}$ is the secret key;*

2. *The TTP has a certified key pair (P_{TTP}, S_{TTP}).*

Protocol 4 (Card shuffling)

1. *The TTP generates a permutation π_T of 52 elements;*

2. *The TTP chooses a random salt R_0;*

3. *The TTP computes $h_{TTP} = H(\pi_T, R_0), S_{TTP}(H(\pi_T, R_0))$;*

4. *The TTP sends h_{TTP} to the rest of players;*

5. *For each $\mathcal{P}_i$ $(i = 1, \ldots, n)$:*

   (a) *$\mathcal{P}_i$ generates a permutation π_i of elements;*

(b) $\mathcal{P}_i$ *computes* $e_i = E_{TTP}(\pi_i, S_{\mathcal{P}_i}(H(\pi_T, R_0), \pi_i))$;

(c) $\mathcal{P}_i$ *sends* e_i *to the TTP;*

6. *The TTP composes all permutations,* $\pi_D = \pi_T \circ \pi_n \circ \pi_{n-1} \circ \cdots \circ \pi_1$. *The shuffled deck of cards is* π_D.

Let us assume that $\mathcal{P}_j$ draws a card.

Protocol 5 (Card drawing)

1. $\mathcal{P}_i$ *picks a random number* R_0;
2. $\mathcal{P}_i$ *generates a request* M *for a card;*
3. $\mathcal{P}_i$ *computes* $M_0 = R_0, M, S_{\mathcal{P}_i}(R_0, M)$;
4. $\mathcal{P}_i$ *sends* M_0 *to the TTP;*
5. *The TTP verifies the signature;*
6. *The TTP picks the* y*-th card; if* $y - 1 < 52$ *have previously been extracted, the* y*-th card is* $c_y = \pi_D(y)$;
7. *The TTP generates a random salt* R_1;
8. *The TTP computes* $M_1 = E_{\mathcal{P}_i}(R_1, M, c_y, S_{TTP}(M_0, R_1, M, C_n))$;
9. *The TTP sends* M_1 *to* $\mathcal{P}_i$;
10. $\mathcal{P}_i$ *decrypts the message* M_1;
11. $\mathcal{P}_i$ *verifies the signature* $S_{TTP}(M_0, R_1, M, C_n)$;
12. $\mathcal{P}_i$ *adds the card* c_y *to her hand.*

Protocol analysis

The proposal [HS97] does not specify the public key cryptosystem to be used. Let us assume that the digital signatures and encryptions are based on Rivest *et al.* [RSA77] public key criptosystem. In Step 5c of Protocol 4 we assume that $\mathcal{P}_i$ builds a digital envelope, see [Sch96] for further details. In Step 4 of Protocol 4 it is implicit that all players verify the TTP's digital signature. In Step 6 it is implicit that the TTP must decrypt the n encrypted messages sent by players, and must verify the digital signatures.

Table 3.4: Costs of the Hall-Schneier shuffling protocol

	Number of messages		Total length of messages		Computational cost	
	$\mathcal{P}_i$	TTP	$\mathcal{P}_i$	TTP	$\mathcal{P}_i$	TTP
Card shuffling	n	n	$52[r] + [S(m)] + [P(m)]$	$[H(m)] + [S(m)]$	3ξ	$\xi(2n+1)$
Step 1						ϵ
Step 2						ϵ
Step 3						ξ
Step 4		n		$[H(m)] + [S(m)]$	ξ	ϵ
Step 5	n		$52[r] + [S(m)] + [P(m)]$		2ξ	ϵ
Step 5a					ϵ	
Step 5b					2ξ	
Step 5c	1		$52[r] + [S(m)] + [P(m)]$		ϵ	
Step 6						$n(2\xi)$

In Table 3.6 we can see that the protocol satisfies the same security properties as [FM85] but does not preserve the confidentiality of strategies. The players verify the fairness of the game when the TTP reveals the secret values used in the game.

Table 3.5: Costs of the Hall-Schneier drawing protocol

	Number of messages		Total length of messages		Computational cost	
	$\mathcal{P}_i$	TTP	$\mathcal{P}_i$	TTP	$\mathcal{P}_i$	TTP
Card drawing	1	1	$[S(m)] + 2[m]$	$3[m] + [S(m)] + [P(m)]$	3ξ	3ξ
Step 1					ϵ	
Step 2					ϵ	
Step 3					ξ	
Step 4	1		$[S(m)] + 2[m]$		ϵ	
Step 5						ξ
Step 6						ϵ
Step 7						ϵ
Step 8						2ξ
Step 9		1		$3[m] + [S(m)] + [P(m)]$		ϵ
Step 10					ξ	
Step 11					ξ	
Step 12					ϵ	

Table 3.6: Security properties of the Hall-Schneier protocol suite

Uniqueness of cards	X
Uniform random distribution of cards	X
Cheating detection with a very high probability	X
Complete confidentiality of cards	X
Minimal effect of coalitions	X
Complete confidentiality of strategy	

3.2.3 Online casinos (Oppliger-Nottaris)

In [ON97] a model is presented that can be used to set up and run an online casino. The proposal was implemented in a prototype at the University of Berne. The cryptographic protocol is focused on Mental Black Jack instead of Mental Poker. Nevertheless, it can be easily adapted to Mental Poker with TTP. The security offered is similar to the one of the Hall-Schneier [HS97] proposal. In our view, this is a relevant contribution.

Let the number of players be n and $\mathcal{P}_i$ be the i-th player in the ordered set of n players.

Protocol 6 (Initialization)

1. *Each player $\mathcal{P}_i$ has a key pair $(P_{\mathcal{P}_i}, S_{\mathcal{P}_i})$;*

2. *The TTP has a key pair (P_{TTP}, S_{TTP}).*

The deck of cards is shuffled using Protocol 7. The TTP chooses a permutation of 52 elements and commits herself to the permutation. Each player chooses a list of 52 values and also commits herself to the list. The card at position j is computed by permuting the value x using the TTP permutation, where x is the sum all values at position j in the players' list.

Protocol 7 (Card Shuffling)

1. *The TTP selects a permutation π_{TTP} of 52 elements at random, $\pi_{TTP} = \{c_1, \ldots, c_{52}\}$ and $1 \leq c_i \leq 52$;*

2. *The TTP computes $m_{TTP,s} = S_{TTP}(TTP, g, H(\pi_{TTP}))$, where g is the game identifier;*

3. *The TTP commits herself to π_{TTP} by multicasting the message m_{TTP}, $m_{TTP} = (TTP, g, H(\pi_{TTP}), m_{TTP,s})$, to all players;*

4. *For each $\mathcal{P}_i$ $(i = 1, \ldots, n)$ do:*

(a) *choose at random a list of* 52 *numbers* $L_i = \{l_{i,1}, \ldots, l_{i,52}\}$*;* L_i *is kept secret by* $\mathcal{P}_i$*;*

(b) *compute* $m_{i,s} = S_{\mathcal{P}_i}(\mathcal{P}_i, g, H(L_i))$*;*

(c) *publish the message* m_i*,* $m_i = (\mathcal{P}_i, g, H(L_i), m_{i,s})$*.*

The TTP and the rest of players run Protocol 8 every time a player extracts a new card from the deck. Let us assume that $\mathcal{P}_j$ wants a card, and during the game $d-1$ cards have already been drawn, where $1 \leq d \leq 52$.

Protocol 8 (Card drawing)

1. *The TTP computes* $m_{TTP,r} = S_{TTP}(TTP, g, d)$*;*
2. *The TTP sends* $(g, d, m_{TTP,r})$ *to the rest of players as a card request;*
3. *For each* $\mathcal{P}_i$ $(i = 1, \ldots, n)$ *do:*
   (a) *compute* $m_{\mathcal{P}_i,r} = S_{\mathcal{P}_i}(\mathcal{P}_i, g, d, l_{i,d})$*;*
   (b) *make public* $(g, d, l_{i,d}, m_{\mathcal{P}_i,r})$*;*
4. *The TTP does the following steps.*
   (a) *compute* $x = (\sum_{i=1}^{n} l_{i,d}) \bmod n - d$*;*
   (b) *compute the card* c *for* $\mathcal{P}_j$*,* $c = \pi_{TTP}(x)$*;*
   (c) *remove the element* c *in* π_{TTP}*, so that the list becomes shorter;*
   (d) *compute* $m_{TTP,c} = S_{TTP}(TTP, g, d, \mathcal{P}_j, c)$*;*
   (e) *send* $(g, d, c, m_{TTP,c})$ *in a secure way to* $\mathcal{P}_j$*.*

Protocol analysis

In Step 2 of Protocol 7 the TTP makes a digital signature. Proposal [ON97] does not specify the public key cryptosystem to be used (like in [HS97]). Let us assume that the digital signatures and encryptions are based on the RSA public key cryptosystem [RSA77].

In Step 3 of Protocol 7 each player $\mathcal{P}_i$ receives the digital signature of Step 2. We consider that verification of this signature is implicit (and must be included in the computational cost).

We have made the same consideration in Steps 4b and 4c of Protocol 7, and Steps 1, Steps 2, Steps 3a, Steps 3b, Steps 4d and Steps 4e of Protocol 8.

Table 3.7: Costs of the Oppliger-Nottaris shuffling protocol

	Number of messages		Total length of messages		Computational cost	
	$\mathcal{P}_i$	TTP	$\mathcal{P}_i$	TTP	$\mathcal{P}_i$	TTP
Card shuffling	1	1	$2[m] + [H(m)] + + [S(m)]$	$2[m] + [H(m)] + + [S(m)]$	2ξ	$\xi(n+1)$
Step 1						ϵ
Step 2						ξ
Step 3		1		$2[m] + [H(m)] + + [S(m)]$	ξ	ϵ
Step 4	1					$n\xi$
Step 4a					ϵ	ϵ
Step 4b					ξ	ϵ
Step 4c	1		$2[m] + [H(m)] + + [S(m)]$		ϵ	ξ

The strategy is revealed in order to verify the game fairness. The TTP publishes the permutations and players can verify the game.

Table 3.8: Costs of the Oppliger-Nottaris drawing protocol

	Number of messages		Total length of messages		Computational cost	
	$\mathcal{P}_i$	TTP	$\mathcal{P}_i$	TTP	$\mathcal{P}_i$	TTP
Card drawing	1	2	$3[m] + [S(m)]$	$5[m] + 2[S(m)]$	3ξ	$\xi(n+2)$
Step 1						ξ
Step 2		1		$2[m] + [S(m)]$	ξ	ϵ
Step 3	1		$3[m] + [S(m)]$		ξ	
Step 3a					ξ	
Step 3b	1		$3[m] + [S(m)]$			ξ
Step 4		1		$3[m] + [S(m)]$	ξ	ξ
Step 4a						ϵ
Step 4b						ϵ
Step 4c						ϵ
Step 4d						ξ
Step 4e		1		$3[m] + [S(m)]$	ξ	ϵ

Table 3.9: Security properties of the Oppliger-Nottaris protocol

Uniqueness of cards	X
Uniform random distribution of cards	X
Cheating detection with a very high probability	X
Complete confidentiality of cards	X
Minimal effect of coalitions	X
Complete confidentiality of strategy	

3.2.4 Fair on-line gambling (Zhao-Varadharajan-Mu)

A payment protocol is proposed in [ZVM00]. This protocol can be used in remote electronic gaming, and more specifically in electronic bets. The protocol uses a TTP, and if any player is not honest the TTP enforces the payment of the bet. The basic protocol with one player and a casino runs as follows.

Protocol 9 (Payment)

1. *The bank has a certified public key. He digitally signs one token, where the maximum credit of the player is specified. The token also contains the player's credit card number and her personal identification number (PIN).*

2. *The player has a certified public key, and she digitally signs the following information: the token sent by the Bank, the bet amount and information about the player against whom she bets.*

3. *The player encrypts the previous digital signature with the TTP's public key.*

4. *Using [Sta96], the player obtains a proof that she encrypted a digital signature, so that she does not need to show the actual signature.*

5. *The casino does the same operations. Both player and casino can verify that the encrypted data are a digital signature, but they cannot use it.*

6. *The game runs, and the result is obtained. The loser sends the digital signature and the winner gets the money of the bet. The digital signature prevents the loser from repudiating the payment.*

7. *If the loser does not send the digital signature, the TTP decrypts the digital signature and sends the result to the winner.*

In addition to the above payment protocol, the paper [ZVM00] also contains a mental poker protocol, but the latter is basically equivalent to the one previously presented in [HS97].

Protocol analysis

This protocol offers the same properties and has the same cost as [HS97].

3.2.5 Mental poker game based on a bit commitment scheme through a network (Chou-Yeh)

In [CY02], the TTP shuffles and draws the cards. A bit commitment protocol is used when the deck is shuffled. This bit commitment is described next.

A bit commitment protocol consists of two distinct stages: commitment and opening. Assume that $\mathcal{P}_i$ uses Procedure 5 to commit to a bit $b_i \in \mathbb{Z}_2$ without revealing it.

Procedure 5 (Commitment($b_i \in \{0,1\}$))

1. *Compute* $\beta_i = m^{b_i} x_i^2 \bmod n$, *where* $n = pq$ *and* p *and* q *are large primes,* $x_i \in \mathbb{Z}_n^*$, $m \in \overline{Q}_n$;

2. *Return* β_i *and* x_i.

$\mathcal{P}_i$ later opens the commitment with Protocol 10. Furthermore, she cannot open the commitment to show a value different from b_i.

Protocol 10 (Commitment opening)

1. $\mathcal{P}_i$ *publishes* x_i;

2. *Anybody can check that:*

$$\begin{cases} \textit{if} \quad ((x_i)^2)^{-1} \cdot \beta_i = m & \textit{then} \quad b_i = 1 \\ \textit{if} \quad ((x_i)^2)^{-1} \cdot \beta_i = 1 & \textit{then} \quad b_i = 0 \end{cases}$$

This bit commitment protocol is basically equivalent to the probabilistic encryption presented in [GM82].

We now describe the mental poker protocol in [CY02]. The TTP generates the deck V with Procedure 6.

Procedure 6 (Card shuffling)

1. *The TTP does the following steps:*

   (a) *Choose a random set* $D = \{d_1, \ldots, d_{52}\}$ *to represent the deck of cards, where the element at* j*-th position represents the* j*-card; let us assume that* d_j *is* r *bits long,* $d_j = \{d_{j,r}, \ldots, d_{j,r}\}$*;*

   (b) *Publish* D*;*

   (c) *For every* $d_j \in D$ *do:*

      i. *For* $k = 1$ *to* r *run Procedure 5 with* $d_{j,k}$ *and obtain* $\beta_{j,k}$ *and* $x_{j,k}$*;*

      ii. *Compute the card* $v_j = (\beta_j, x_j)$*, where* $\beta_j = \{\beta_{j,1}, \ldots, \beta_{j,r}\}$ *and* $x_j = \{x_{j,1}, \ldots, x_{j,r}\}$*;*

   (d) *Compute the deck of cards* $V = \{v_1, \ldots, v_{52}\}$.

The TTP and one player $\mathcal{P}_i$ use Protocol 11 when $\mathcal{P}_i$ wants a card.

Protocol 11 (Card drawing)

1. *The TTP chooses a card* $v_j = (\beta_j, x_j)$ *in deck* V *such that* v_j *has not been drawn previously;*

2. *The TTP sends* β_j *to* $\mathcal{P}_i$*;*

3. *The TTP encrypts* x_j *with the* $\mathcal{P}_i$*'s public key,* $c_j = E_{\mathcal{P}_i}(x_j)$*;*

4. *The TTP sends* c_j *to* $\mathcal{P}_i$*;*

5. $\mathcal{P}_i$ *decrypts* c_j *and obtains* $x_j = D_{\mathcal{P}_i}(c_j)$*;*

6. $\mathcal{P}_i$ *verifies the bit commitment with* x_i *and gets* d_i*; this verification is done with* r *executions of Protocol 10.*

Table 3.10: Costs of the Chou-Yeh shuffling protocol

	Number of messages		Total length of messages		Computational cost	
	$\mathcal{P}_i$	TTP	$\mathcal{P}_i$	TTP	$\mathcal{P}_i$	TTP
Card shuffling		1		$[r]52$		$78\rho[r]$
Step 1		1		$[r]52$		$78\rho[r]$
Step 1a						ϵ
Step 1b		1		$[r]52$		ϵ
Step 1c						$52[r](\frac{3}{2}\rho)$
Step 1(c)i						$[r](\frac{3}{2}\rho)$
Step 1(c)ii						ϵ
Step 1d						ϵ

Table 3.11: Costs of the Chou-Yeh drawing protocol

	Number of messages		Total length of messages		Computational cost	
	$\mathcal{P}_i$	TTP	$\mathcal{P}_i$	TTP	$\mathcal{P}_i$	TTP
Card drawing		2		$[p][r] + [P(m)]$	$\xi + 2[r]\rho$	ξ
Step 1						ϵ
Step 2		1		$[r][p]$		ϵ
Step 3						ξ
Step 4		1		$[P(m)]$		ϵ
Step 5					ξ	
Step 6					$[r](2\rho)$	ϵ

Protocol analysis

Players do not verify any TTP action. If the TTP is completely trusted, this proposal meets all of Crépeau's requirements.

Authors criticize the proposal [HS97] because it "is based on the assumption that there is no secret communication link among any players". Nevertheless, in [CY02] the TTP does all the work: it shuffles the deck and draws the cards. Players must trust the TTP blindly. It could be interesting to explore the result of a confabulation between the TTP and a player.

Table 3.12: Costs of the Chou-Yeh Procedure 5

	Computational cost
Procedure 5	$\frac{3}{2}\rho$
Step 1	$\rho + \frac{1}{2}\rho$
Step 2	ϵ

Table 3.13: Costs of the Chou-Yeh Protocol 10

	Number of messages		Total length of messages		Computational cost	
	$\mathcal{P}_i$	TTP	$\mathcal{P}_i$	TTP	$\mathcal{P}_i$	TTP
Protocol 10		1		$[p]$	2ρ	ϵ
Step 1		1		$[p]$		ϵ
Step 2					2ρ	

A second point is the use of a bit commitment protocol, whose properties are not fully exploited. The TTP sends the commitment and opens it in the next message. Why? The bit commitment protocol adds to the computational load without adding security.

Finally, the bit commitment protocol used is very similar to the encryption presented in [GM82]. The paper should mention that their bit commitment is inspired on the encryption presented in [GM82].

Table 3.14: Security properties of the Chou-Yeh protocol

Uniqueness of cards	X
Uniform random distribution of cards	X
Cheating detection with a very high probability	X
Complete confidentiality of cards	X
Minimal effect of coalitions	X
Complete confidentiality of strategy	X

3.2.6 Conclusions on the comparison of TTP-based protocols

Once we have described the main mental poker protocols using a TTP, we can draw some conclusions about the computational cost, the number of messages and the length of the messages.

In Table 3.15 we can see the costs of shuffling a deck of cards for every proposal. [CY02] is the most efficient protocol as far as the number and the length of messages are concerned. This efficiency is due to fact that the TTP does all the work. The players do not compute nor send anything. [ON97] is very efficient in the number of messages; however we must remember that it uses multicast. [ON97] without multicast would send about n messages, like [HS97].

The computational cost of the Protocol in [FM85] is ϵ because a secure channel is assumed between the TTP and the players. If the transmitted values are encrypted, the computational cost becomes similar to the cost of the other proposals.

Table 3.15: Costs of the card shuffling protocols using a TTP

	Number of messages		Total length of messages		Computational cost	
	$\mathcal{P}_i$	TTP	$\mathcal{P}_i$	TTP	$\mathcal{P}_i$	TTP
[FM85]	$n+1$	n	$n(52[r]+[H(m)])$	$n52[r]$	ϵ	ϵ
[HS97]	n	n	$52[r]+[S(m)]+[P(m)]$	$[H(m)]+[S(m)]$	3ξ	$\xi(2n+1)$
[ON97]	1	1	$2[m]+[H(m)]+[S(m)]$	$2[m]+[H(m)]+$ $+[S(m)]$	2ξ	$\xi(n+1)$
[ZVM00]	n	n	$52[r]+[S(m)]+[P(m)]$	$[H(m)]+[S(m)]$	3ξ	$\xi(2n+1)$
[CY02]		1		$[r]52$		$78\rho[r]$

In Table 3.16 we show the results of our analysis of drawing a deck of cards for every proposal. The number of messages is similar in all protocols but [FM85]. In [FM85] the TTP does not take part in the drawing, which is done co-operatively by all players. Player co-operation increases the number of messages.

The total length of messages in [FM85] is the lowest, $(2n-1)[r]$. Note that $r \approx 6bits$, because a value x in $\{1,\ldots,52\}$ can be represented with 6 bits ($\log_2 52$). The maximum number of players in poker is 5, so in the drawing protocol the total

length of messages is 54 bits. The rest of proposals send two or more values in Z_n, that is 2(1024) bits.

The computational cost is ϵ in [FM85] because players do not compute any cryptograhic operation. [HS97], [ON97] and [ZVM00] have a similar computational cost. Finally, the cost of [CY02] is greater than the cost of [FM85] and less than the cost of the rest of protocols.

Table 3.16: Costs of the card drawing protocols using a TTP

	Number of messages		Total length of messages		Computational cost	
	$\mathcal{P}_i$	TTP	$\mathcal{P}_i$	TTP	$\mathcal{P}_i$	TTP
[FM85]	$2n-1$		$(2n-1)[r]$		ϵ	
[HS97]	1	1	$[S(m)]+2[m]$	$3[m]+[S(m)]+[P(m)]$	3ξ	3ξ
[ON97]	1	2	$3[m]+[S(m)]$	$5[m]+2[S(m)]$	3ξ	$\xi(n+2)$
[ZVM00]	1	1	$[S(m)]+2[m]$	$3[m]+[S(m)]+[P(m)]$	3ξ	3ξ
[CY02]		2		$[r][p]+[P(m)]$	$2r\rho+\xi$	ξ

The protocols presented so far have a reasonable load, so they can be used in practice.

In all cases we must trust the TTP to some extent. In [FM85] players only use the TTP when the deck is shuffled. This is the proposal that requires less trust in the TTP. On the other side, the TTP in [CY02] does all operations. The players must trust it completely.

After reviewing the main TTP-based contributions to mental poker, we notice that in [FM85], [HS97], [ON97] and [ZVM00] the strategy of players is revealed (see Table 3.17). Nevertheless, in these protocols players take part in shuffling the deck. Also, when the game is over players can verify whether the TTP was honest.

In [CY02] the strategy is not revealed, but players must trust the TTP completely.

Table 3.17: Security properties of TTP-based mental poker protocols

	[FM85]	[HS97]	[ON97]	[ZVM00]	[CY02]
Uniqueness of cards	X	X	X	X	X
Uniform random distribution of cards	X	X	X	X	X
Cheating detection with a very high probability	X	X	X	X	X
Complete confidentiality of cards	X	X	X	X	X
Minimal effect of coalitions	X	X	X	X	X
Complete confidentiality of strategy					X

3.3 TTP-free mental poker protocols

TTP-free mental poker protocols are more complex than mental poker protocols with a TTP. Next we present the main TTP-free protocols in the literature. We present contributions in the order in which they were published.

3.3.1 Mental poker (Shamir-Rivest-Adleman)

The TTP-free mental poker proposal is [SRA81]. This protocol is for two players only and is based on a commutative cryptosystem. The encryption algorithm is the following:

Preparation :

1. The players choose a prime number p.
2. Each player chooses her private key k secretly.

Encryption A message m is encrypted as follows:

$$E_k(m) = x^k \bmod p = c$$

Decryption A cryptogram c is decrypted as follows:

$$D_k(m) = c^{k^{-1}} \bmod p = m$$

It is easy to see that the above cryptosystem is commutative, *i.e.* $E_{\mathcal{P}_1}(E_{\mathcal{P}_2}(x)) = E_{\mathcal{P}_2}(E_{\mathcal{P}_1}(x))$. Based on this cryptosystem, the mental poker protocol is constructed as:

Let us assume that Protocol 12 is run between $\mathcal{P}_1$ and $\mathcal{P}_2$. All operations are $(\mathrm{mod} p)$.

Protocol 12 (Initialization)

1. *$\mathcal{P}_1$ and $\mathcal{P}_2$ agree on a public p;*

2. *$\mathcal{P}_1$ chooses a secret key k_1;*

3. *$\mathcal{P}_2$ chooses a secret key k_2.*

Let us assume that $\mathcal{P}_2$ is the dealer in Protocol 13.

Protocol 13 (Card shuffling)

1. *$\mathcal{P}_2$ chooses 52 values to represent the cards: $D = \{x_1, \ldots, x_{52}\}$, where $1 < x_i < p$;*

2. *$\mathcal{P}_2$ computes $C^* = \{c_1^*, \ldots, c_{52}^*\}$, where $c_i^* = x_i^{k_2}$ and $x_i \in D$;*

3. *$\mathcal{P}_2$ randomly chooses a permutation π;*

4. *$\mathcal{P}_2$ computes $C = \{c_1, \ldots, c_{52}\}$, where $c_{\pi(i)} = c_i^*$ and $c_i^* \in C^*$;*

5. *$\mathcal{P}_2$ sends C to $\mathcal{P}_1$.*

Let us assume that $\mathcal{P}_1$ and $\mathcal{P}_2$ draw one card using Protocol 14.

Protocol 14 (Card drawing (C))

1. *$\mathcal{P}_1$ randomly selects an encrypted message c_1 and sends it to $\mathcal{P}_2$;*

2. *$\mathcal{P}_2$ decrypts the cryptogram c_1 to determine her card;*

3. *$\mathcal{P}_1$ randomly selects an encrypted message c_1';*

4. *$\mathcal{P}_1$ encrypts c_1' under k_1 to get $d_1 = (c_1')^{k_1}$;*

5. *$\mathcal{P}_1$ sends d_1 to $\mathcal{P}_2$;*

6. *$\mathcal{P}_2$ decrypts d_1 under k_2 to get $e_1 = d_1^{k_2^{-1}}$;*

7. *$\mathcal{P}_2$ sends e_1 to $\mathcal{P}_1$;*

8. *$\mathcal{P}_1$ decrypts e_1 under k_1 to determine her card.*

During the game, additional cards may be dealt with by repeating the procedure. At the end of the game, both players reveal their keys to prove they did not cheat.

Protocol analysis

In this section we present the analysis of the protocols for cards shuffling and card drawing.

Table 3.18 shows the number of messages, the total length of messages and the computational cost of the shuffling protocol (Protocol 13).

Table 3.18: Costs of the Shamir-Rivest-Adleman shuffling protocol

	Number of messages		Total length of messages		Computational cost	
	$\mathcal{P}_1$	$\mathcal{P}_2$	$\mathcal{P}_1$	$\mathcal{P}_2$	$\mathcal{P}_1$	$\mathcal{P}_2$
Card shuffling		1		$52[p]$		52ξ
Step 1						ϵ
Step 2						52ξ
Step 3						ϵ
Step 4						ϵ
Step 5		1		$52[p]$		ϵ

Table 3.19 shows the number of messages, the total length of messages and the computational cost of the drawing protocol (Protocol 14).

Table 3.19: Costs of the Shamir-Rivest-Adleman drawing protocol

	Number of messages		Total length of messages		Computational cost	
	$\mathcal{P}_1$	$\mathcal{P}_2$	$\mathcal{P}_1$	$\mathcal{P}_2$	$\mathcal{P}_1$	$\mathcal{P}_2$
Card drawing	2	1	$2[p]$	$[p]$	2ξ	2ξ
Step 1	1		$[p]$		ϵ	
Step 2						ξ
Step 3					ϵ	
Step 4					ξ	
Step 5	1		$[p]$		ϵ	
Step 6						ξ
Step 7		1		$[p]$		ϵ
Step 8					ξ	

Table 3.20 shows the security properties of [SRA81].

Table 3.20: Security properties of the Shamir-Rivest-Adleman protocol suite

Uniqueness of cards	X
Uniform random distribution of cards	X
Cheating detection with a very high probability	
Complete confidentiality of cards	
Minimal effect of coalitions	X
Complete confidentiality of strategy	

The confidentiality of strategy is not guaranteed and only two players can participate in the game. [SRA81] has another additional drawback: Lipton in [Lip81] shows that cards can be marked as explained below.

For any message $M = a$, encrypting (or decrypting) with the [SRA81] cryptosystem M preserves membership in Q_n, as shown by the following theorem:

Theorem 1 *Given a, $0 < a$, it holds that $a \in Q_n$ if and only if $E_K(a) = a^e \bmod n \in Q_n$, where $K = (e, n)$.*

Proof: 1 *First, assume $a \in Q_n$. Then $x^2 \bmod n = a$ for some x. Since*

$$\begin{aligned} E_K(a) &= a^e \bmod n \\ &= (x^2)^e \bmod n \\ &= (x^e)^2 \bmod n \end{aligned}$$

$E_K(a)$ is in Q_n. Now, suppose $E_K(a)$ in Q_n. Since decrypting is the same operation as encrypting but with exponent d, $(E_K(a))^d \bmod n = a$ must also be in Q_n.

$\mathcal{P}_1$ can exploit the above result by recording which cards have messages in Q_n. For example, if she observes that the plaintext messages for all four aces are quadratic residues, she could select quadratic residues for her hand and quadratic non-residues for $\mathcal{P}_2$. This is called card marking.

3.3.2 Probabilistic encryption and how to play mental poker keeping secret all partial information (Goldwasser-Micali)

The Goldwasser-Micali mental poker protocol [GM82] is also for two players only, but nonetheless it is an important contribution to cryptography. Players choose at random 52 values to represent the deck of cards. Every player computes a key pair of a public-key cryptosystem for every card. Every card is encrypted with a different key pair. Players publish encrypted values and the public keys used. When a player wants a card, she asks a question about every encrypted card, but for only one card she gains enough information to decrypt it. At the end of game players publish private components of key pairs used and game fairness is verified.

The probabilistic encryption concept is also introduced in that paper. A cryptosystem is probabilistic if it has the following property: if a message d is encrypted several times under the same public key, the cryptograms obtained are different each time.

We next review the Goldwasser-Micali probabilistic cryptosystem and the mental poker protocol that is built on the cryptosystem.

The players use Procedure 7 in order to create the cryptosystem key pair.

Procedure 7 (Initialization)

1. *Two large prime numbers p and q are chosen and $n = pq$ is computed. p and q have the following properties $p \equiv 3 \bmod 4$ and $q \equiv 3 \bmod 4$;*
2. *A value y is chosen at random so that $\left(\frac{y}{p}\right) = -1$ and $\left(\frac{y}{q}\right) = -1$.*

A message $d = \{d_1, \ldots, d_r\}$, where $d \in \mathbb{Z}_n^*$ and $d_1, \cdots, d_r$ are the bits representing d, is encrypted using Procedure 8.

Procedure 8 (Encryption(d))

1. *For every bit $d_i \in d$ do:*
   (a) *randomly choose $x_i \in \mathbb{Z}_n^*$;*

(b) *if* $d_i \equiv 0$ *then compute* $e_i = x_i^2 \bmod n$;

(c) *if* $d_i \equiv 1$ *then compute* $e_i = yx_i^2 \bmod n$;

2. *Let the cryptogram corresponding to* d *be* $e = \{e_1, \cdots, e_r\}$;

3. *Return* e.

Given a cryptogram $e = \{e_1, \ldots, e_r\}$ and the factors p, q of n (secret key), $d = \{d_1, \ldots, d_r\}$ can be retrieved using Procedure 9.

Procedure 9 (Decryption(e))

1. *For every value* $e_i \in e = \{e_1, \ldots, e_r\}$ *do:*

(a) *if* $\left(\frac{e_i}{p}\right) \equiv \left(\frac{e_i}{q}\right) \equiv 1$ *then* $d_i = 0$;

(b) *if* $\left(\frac{e_i}{p}\right) \equiv \left(\frac{e_i}{q}\right) \equiv -1$ *then* $d_i = 1$;

2. *Let the message corresponding to* e *be* $d = \{d_1, \cdots, d_r\}$;

3. *Return* d.

When 0 is encrypted, the resulting cryptogram is in Q_n and, if a 1 is encrypted, the resulting cryptogram is in $\overline{Q}_n$. An observer cannot decide whether a cryptogram is in Q_n or $\overline{Q}_n$ because she does not know the factors of n. If the observer computes the Jacobian symbol of e_i and n, she always gets 1 as a result, *i.e.* she gains no information about d_i.

The Goldwasser-Micali mental poker protocol can now be described. Let us assume that $\mathcal{P}_1$ and $\mathcal{P}_2$ participate in the protocols.

Protocol 15 (Initialization)

1. $\mathcal{P}_1$ *and* $\mathcal{P}_2$ *choose* 52 *values* $D = \{d_1, \ldots, d_{52}\}$ *to represent the cards. Let us assume that* d_i *is* r *bits long, where* $6 \leq r$;

2. $\mathcal{P}_2$ *chooses at random* 52 *pairs of large prime numbers,* $\{(p_{2,1}, q_{2,1}), \ldots, (p_{2,52}, q_{2,52})\}$, *such that* $p_{2,i} \equiv q_{2,i} \equiv 3 \bmod 4$, *where* $1 \leq i \leq 52$;

3. *$\mathcal{P}_2$ computes 52 large composite numbers $N_2 = \{n_{2,1}, \ldots, n_{2,52}\}$, where $n_{2,i} = (p_{2,i}) \cdot (q_{2,i})$, and $1 \leq i \leq 52$;*

4. *$\mathcal{P}_1$ goes through the same steps as $\mathcal{P}_2$, so as to get $\{(p_{1,1}, q_{1,1}), \ldots, (p_{1,52}, q_{1,52})\}$, and $N_1 = \{n_{1,1}, \ldots, n_{1,52}\}$, where $n_{1,i} = p_{1,i}q_{1,i}$ and $1 \leq i \leq 52$.*

$\mathcal{P}_1$ and $\mathcal{P}_2$ shuffle the deck of cards using Protocol 16.

Protocol 16 (Card shuffling (D))

1. *$\mathcal{P}_2$ uses Procedure 10 with D and N_2 and obtains $C_2 = \{c_{2,1}, \ldots, c_{2,52}\}$;*

2. *$\mathcal{P}_2$ publishes the pairs $(c_{2,i}, n_{2,i})$, where $c_{2,i} \in C_2$ and $n_{2,i} \in N_2$;*

3. *$\mathcal{P}_1$ uses Procedure 10 with D and N_1 and obtains $C_1 = \{c_{1,1}, \ldots, c_{1,52}\}$;*

4. *$\mathcal{P}_1$ publishes the pairs $(c_{1,i}, n_{1,i})$, where $c_{1,i} \in C_1$ and $n_{1,i} \in N_1$.*

Procedure 10 permutes and encrypts the deck D. For each $d_i \in D$, it uses one $n_i \in N$.

Procedure 10 (Deck encryption(D, N))

1. *Generate at random a permutation π of 52 elements;*

2. *Compute the set $D' = \{d'_1, \ldots, d'_{52}\}$, where $d'_{\pi(i)} = d_i$, $d_i \in D$;*

3. *For each $d'_i \in D'$ encrypt d'_i using Procedure 8 and obtain c_i;*

4. *Compute the set $C = \{c_1, \ldots, c_{52}\}$;*

5. *Return C.*

Let us assume that $\mathcal{P}_2$ deals a card to $\mathcal{P}_1$, and $\mathcal{P}_1$ decides to pick the k-th card in C_2, that is, $(c_{2,k}, n_{2,k})$.

Protocol 17 (Card drawing)

1. *For $1 \leq i \leq 52$, $\mathcal{P}_1$ does:*

(a) *choose* t_i *at random;*

(b) *compute* $h_i = t_i^2 \bmod n_{2,i}$*;*

(c) *if* $i = k$ *then send* h_i *and* $-\left(\frac{t_i}{n_{2,i}}\right)$ *to* $\mathcal{P}_2$*;*

(d) *if* $i \neq k$ *then send* h_i *and* $\left(\frac{t_i}{n_{2,i}}\right)$ *to* $\mathcal{P}_2$*;*

2. *For* $1 \leq i \leq 52$*,* $\mathcal{P}_2$ *does:*

(a) *compute the square roots of* h_i*,* $(x, -x, y, -y)$*;*

(b) *if* $-\left(\frac{t_i}{n_{2,i}}\right)$ *was received, then send* y *to* $\mathcal{P}_1$*;*

(c) *if* $\left(\frac{t_i}{n_{2,i}}\right)$ *was received, then send* x *to* $\mathcal{P}_1$*;*

3. $\mathcal{P}_1$ *factors* $n_{2,k} = (p_{2,k}) \cdot (q_{2,k})$*. She can factor* $n_{2,k}$ *because she knows* $x \neq y \bmod n_k$ *for the* k*-th card* $(c_{2,k}, n_{2,k})$ *by Lemma 2. With* x *and* y*, by Lemma 1,* $\mathcal{P}_1$ *can factor* $n_{2,k}$*. She cannot factor* $n_{2,i} \neq n_{2,k}$*, because she only knows* x*;*

4. $\mathcal{P}_1$ *decrypts* $c_{2,k}$ *with* $n_{2,k} = (p_{2,k}) \cdot (q_{2,k})$ *using Procedure 9.*

At the end of the game, $\mathcal{P}_1$ reveals the t_i used, so that $\mathcal{P}_2$ can verify that $\mathcal{P}_1$ has picked only one card.

Protocol analysis

Once the proposal [GM82] has been described, we present the analysis for Protocol 16 (Card shuffling) and Procotol 17 (Card Drawing).

Table 3.21 shows the number of messages, the total length of messages and the computational cost of the shuffling protocol (Protocol 16).

Table 3.22 shows the number of messages, the total length of messages and the computational cost of the drawing protocol (Protocol 17). In Step 2a of Protocol 17 we have assumed that Procedure 4 is used to compute the square roots. Authors choose p and q with the following properties, $p \equiv 3 \bmod 4$ and $q \equiv 3 \bmod 4$, so that Procedure 1 can be used. With these assumptions we have a computational cost of $2\xi + 4\rho$ in Step 2a.

Table 3.21: Costs of the Goldwasser-Micali shuffling protocol

	Number of messages		Total length of messages		Computational cost	
	$\mathcal{P}_1$	$\mathcal{P}_2$	$\mathcal{P}_1$	$\mathcal{P}_2$	$\mathcal{P}_1$	$\mathcal{P}_2$
Card shuffling	1	1	$52[p]([r]+1)$	$52[p]([r]+1)$	$78[r]\rho$	$78[r]\rho$
Step 1					$78[r]\rho$	
Step 2		1		$52[p]([r]+1)$	ϵ	
Step 3						$78[r]\rho$
Step 4	1		$52[p]([r]+1)$			ϵ

We next detail the cost of Procedure 8, Procedure 9 and Procedure 10 in Tables 3.23, 3.24 and 3.25, respectively.

Decryption of $c_{2,k}$ using Procedure 9 has a cost of $2[r]\xi$. We next break down this decryption cost. For every bit $c_{2,k,i}$ in $c_{2,k} = \{c_{2,1}, \ldots, c_{2,r}\}$ the Jacobi symbol is computed. The cost of computing the Jacobi symbol is twice the cost of the Legendre symbol (see Definition 3). Finally the cost of the Legendre symbol is ξ (see Theorem 1).

In Steps 1c and 1d of Protocol 17 $\mathcal{P}_1$ could send $-\left(\frac{t_i}{n_{2,i}}\right)$ for more than a card. Thus $\mathcal{P}_1$ would obtain more than one card, or she could discover the cards in $\mathcal{P}_2$'s hand. This cheating is detected when game is verified at the end. In game verification the players reveal their secret information.

In conclusion, this protocol is only designed for two players, whose strategy is made public at the end of the game.

Table 3.22: Costs of the Goldwasser-Micali drawing protocol

	Number of messages		Total length of messages		Computational cost	
	$\mathcal{P}_1$	$\mathcal{P}_2$	$\mathcal{P}_1$	$\mathcal{P}_2$	$\mathcal{P}_1$	$\mathcal{P}_2$
Card drawing	52	52	$52[p]$	$52[p]$	$52\rho + 2[r]\xi$	$52(2\xi + 4\rho)$
Step 1	52		$52[p]$		52ρ	
Step 1a					ϵ	
Step 1b					ρ	
Step 1c	1/52		$(1/52)[p]$		ϵ	
Step 1d	51/52		$(51/52)[p]$		ϵ	
Step 2		52		$52[p]$		$52(2\xi + 4\rho)$
Step 2a						$(2\xi + 4\rho)$
Step 2b		1/52		$(1/52)[p]$		ϵ
Step 2c		51/52		$(51/52)[p]$		ϵ
Step 3					ϵ	
Step 4					$2[r]\xi$	

Table 3.23: Costs of the Goldwasser-Micali Procedure 8

	Computational cost
Procedure 8	$[r]\rho 3/2$
Step 1	$[r]\rho 3/2$
Step 1a	ϵ
Step 1b	ρ
Step 1c	2ρ
Step 2	ϵ
Step 3	ϵ

Table 3.24: Costs of the Goldwasser-Micali Procedure 9

	Computational cost
Procedure 9	$2[r]\xi$
Step 1	$2[r]\xi$
Step 1a	2ξ
Step 1b	2ξ
Step 2	ϵ
Step 3	ϵ

Table 3.25: Costs of the Goldwasser-Micali Procedure 10

	Computational cost
Procedure 10	$78[r]\rho$
Step 1	ϵ
Step 2	ϵ
Step 3	$78[r]\rho$
Step 4	ϵ
Step 5	ϵ

Table 3.26: Security properties of the Goldwasser-Micali protocol suite

Uniqueness of cards	X
Uniform random distribution of cards	X
Cheating detection with a very high probability	X
Complete confidentiality of cards	X
Minimal effect of coalitions	X
Complete confidentiality of strategy	

3.3.3 Mental poker with three or more players (Banary-Füredi)

In [BF83] a simple mental poker protocol is presented which does not use cryptography. Let us illustrate the protocol with an ordered set of n players $\{\mathcal{P}_1, \ldots, \mathcal{P}_n\}$.

Protocol 18 (Card shuffling)

1. *Each player $\mathcal{P}_i$, $i \in \{1, \ldots, n\}$ chooses a random permutation π_i of the 52 values;*

2. *Each player $\mathcal{P}_i$, $i \in \{2, \ldots, n\}$, sends π_i to $\mathcal{P}_1$. $\mathcal{P}_i$ does this secretly, so that she remains unaware of other players' permutations;*

3. *$\mathcal{P}_1$ receives $\{\pi_2, \ldots, \pi_n\}$ from the rest of players;*

4. *For $i = 2$ to n $\mathcal{P}_1$ does:*

   (a) *compute $\pi_{i,1} = \pi_n \circ \pi_{n-1} \circ \cdots \circ \pi_{i+1} \circ \pi_{i-1} \circ \ldots \pi_1^{-1}$;*

   (b) *secretly send $\pi_{i,1}$ to $\mathcal{P}_i$.*

Once the Protocol 18 is run, the cards are ready to be dealt using Protocol 19. The player $\mathcal{P}_1$ plays a special role in the shuffling protocol (Protocol 18). $\mathcal{P}_1$'s permutation is the shuffled deck of cards. When a player $\mathcal{P}_k$ draws a card, the rest of players without player $\mathcal{P}_1$ choose a value $x \in \{1, \ldots, 52\}$ that has not been previously chosen. Let $\pi_1(x)$ be the drawn card. The set $H_{k,1}$ contains the values x that players have chosen for $\mathcal{P}_k$.

Assume that some cards have been dealt, and $\mathcal{P}_k$ wants a new card. She gets it from $\mathcal{P}_{k+1}$ using Protocol 19 below.

Protocol 19 (Card drawing)

1. *Player $\mathcal{P}_{k+1}$ chooses a number x_{n+1}, $1 \leq x_{n+1} \leq 52$ and $x_{n+1} \notin H_{1,1} \bigcup H_{2,1} \bigcup \cdots \bigcup H_{n,1}$;*

2. *If $k \equiv 1$ then:*

   (a) *$\mathcal{P}_{k+1}$ sends x_{n+1} to the rest of players;*

(b) *All players add* x_{n+1} *to the set* $H_{1,1}$*;*

(c) $\mathcal{P}_1$ *obtains her card* $y = \pi_1^{-1}(x_{n+1})$*;*

(d) $\mathcal{P}_1$ *adds* y *to* H_1*;*

3. *If* $k \not\equiv 1$ *then:*

(a) $\mathcal{P}_{k+1}$ *sends* x_{n+1} *to the following players* $\{\mathcal{P}_n, \mathcal{P}_{n-1}, \ldots, \mathcal{P}_{k+1}, \mathcal{P}_{k-1}, \ldots, \mathcal{P}_2\}$*;*

(b) *All players in* $\{\mathcal{P}_n, \mathcal{P}_{n-1}, \ldots, \mathcal{P}_{k+1}, \mathcal{P}_{k-1}, \ldots, \mathcal{P}_2\}$ *add* x_{n+1} *to the set* $H_{k,1}$*;*

(c) *For each player* $\mathcal{P}_i$ *in* $\{\mathcal{P}_n, \mathcal{P}_{n-1}, \ldots, \mathcal{P}_{k+1}, \mathcal{P}_{k-1}, \ldots, \mathcal{P}_2\}$ *do:*

i. *Receive* x_{i+1}*;*

ii. *Compute* $x_i = \pi_i^{-1}(x_{i+1})$*;*

iii. *Send* x_i *to next player in* $\{\mathcal{P}_n, \mathcal{P}_{n-1}, \ldots, \mathcal{P}_{k+1}, \mathcal{P}_{k-1}, \ldots, \mathcal{P}_2\}$*;*

(d) $\mathcal{P}_k$ *computes his card,* $y = \pi_1^{-1}(x_2)$*, and adds* y *to* H_k*.*

When the game is over all players reveal their permutations. With this information the players can show and prove their hands. At the same time, game correctness can be verified.

Protocol analysis

In this section we analyze the number of messages, the total length of messages, the computational cost and the security of [BF83].

Table 3.27 shows the cost of the shuffling protocol (Protocol 19).

The costs of the drawing protocol (Protocol 19) are shown in Table 3.28.

Step 2 of Protocol 19 is run by $\mathcal{P}_1$ and Step 3 of Protocol 19 is run by the remaining $n-1$ players. So, the average number of messages is $1/n$ in Step 2 of Protocol 19 plus $(n-1)/n$ in Step 3 of Protocol 19. This explains the figures in Table 3.28.

The deck of cards is a permutation of 52 values. Therefore a card has value x, where $1 \leq x \leq 52$. The size in bits of x is denoted by $[r]$, where $6 \leq [r]$.

The assumption that players do not collude is crucial to the security of the protocol. If any two players share their knowledge, they learn not only each other's hands, but their opponent's hands as well.

Table 3.27: Costs of the Banary-Furedi shuffling protocol

	Number of messages	Total length of messages	Computational cost
Card shuffling	$2(n-1)$	$104[r](n-1)$	ϵ
Step 1			ϵ
Step 2	$n-1$	$52[r](n-1)$	ϵ
Step 3			ϵ
Step 4	$n-1$	$52[r](n-1)$	ϵ
Step 4a			ϵ
Step 4b	1	$[r]52$	ϵ

If players want to verify correctness, all information should be revealed, so the strategy of players is not kept confidential.

The security properties of the proposal [BF83] are summarized in Table 3.29.

Table 3.28: Costs of the Banary-Furedi drawing protocol

	Number of messages	Total length of messages	Computational cost
Card drawing	$\frac{2(n-1)^2}{n}$	$\frac{2(n-1)^2}{n}[r]$	ϵ
Step 1			ϵ
Step 2	$\frac{1}{n}(n-1)$	$\frac{1}{n}[r](n-1)$	ϵ
Step 2a	$n-1$	$[r](n-1)$	ϵ
Step 2b			ϵ
Step 2c			ϵ
Step 2d			ϵ
Step 3	$\frac{n-1}{n}(2n-3)$	$[r](2n-3)\frac{n-1}{n}$	ϵ
Step 3a	$n-1$	$[r](n-1)$	ϵ
Step 3b			ϵ
Step 3c	$n-2$	$[r](n-2)$	ϵ
Step 3(c)i			ϵ
Step 3(c)ii			ϵ
Step 3(c)iii	1	$[r]$	ϵ
Step 3d			ϵ

Table 3.29: Security properties of the Banary-Furedi protocol suite

Uniqueness of cards	X
Uniform random distribution of cards	X
Cheating detection with a very high probability	X
Complete confidentiality of cards	X
Minimal effect of coalitions	
Complete confidentiality of strategy	

3.3.4 Cryptoprotocols: subscription to a public key, secret blocking and multi-player mental poker game (Yung)

In [Yun85], a TTP-free mental poker protocol is presented. More than two players can participate in this protocol. When a player wants a card, she requests it from the rest of players. The rest of players fix a card ordering, *i.e.* a permutation of 52 values. Every player, independently, chooses at random a set of numbers that represent a deck of cards. The order in the set is the value of the card. These numbers are coded with a one-way function and the result is public. Note that this step is a bit commitment. Next, she encrypts those numbers that represent a card that does not belong to her hand. She makes these cryptograms public. The player who asked for a card runs the oblivious transfer protocol with these cryptograms, and with the rest of players. Finally, she obtains a set of numbers that represent the free cards, *i.e.* those cards that do not belong to any player. She runs another oblivious transfer protocol using this information and she gets the value of a free card.

This protocol also uses a threshold scheme, which increases security. The threshold scheme is similar to the one presented in [Sha79]. Generally speaking, this protocol is rather complex and has some similarity to Rabin's oblivious transfer protocol [Rab81].

Let us assume that $\mathcal{P}_1$ and $\mathcal{P}_2$ run the oblivious transfer protocol (Protocol 20). $\mathcal{P}_2$ can factor n, *i.e.* obtain the secret, with probability 1/2. $\mathcal{P}_1$ cannot be sure whether she transferred the secret to $\mathcal{P}_2$ successfully.

Protocol 20 (Oblivious transfer)

1. $\mathcal{P}_1$ *Chooses two large prime numbers* p *and* q*;*
2. $\mathcal{P}_1$ *Computes* $n = pq$*;*
3. $\mathcal{P}_1$ *Sends* n *to* $\mathcal{P}_2$*;*
4. $\mathcal{P}_2$ *Selects a random* x*;*
5. $\mathcal{P}_2$ *Computes* $z = x^2 \bmod n$*;*
6. $\mathcal{P}_2$ *Sends* z *to* $\mathcal{P}_1$*;*

7. *$\mathcal{P}_1$ Computes the four square roots of $z = \{x, -x, y, -y\}$;*

8. *$\mathcal{P}_1$ chooses at random one of them and sends it to $\mathcal{P}_2$;*

9. *If $\mathcal{P}_2$ receives y or $-y$, she gets the secret (see Lemma 1);*

10. *If $\mathcal{P}_2$ receives x or $-x$, she gets nothing.*

Let us assume that one player $\mathcal{P}_i$ wants to divide a secret m into k parts using Procedure 11.

Procedure 11 (Embedding(m, k))

1. *Choose at random k pairs of prime numbers, $P = \{(p_1, q_1), \ldots, (p_k, q_k)\}$;*

2. *Construct a polynomial of degree k: $P \bmod R$, where R is a large prime $R > (p_i q_i)$ for $1 \leq i \leq k$; to that end, use $k+1$ interpolation points, $(p_i, q_i) \in P$ and $(0, m)$;*

3. *Construct the set $E = \{R, n_1, \ldots, n_k, (u, v)\}$, where $n_i = p_i q_i$, $1 \leq i \leq k$ and (u, v) is a point $v = P(u)$; u is chosen at random, so that $u \neq 0$; the set E is called the embedding of m;*

4. *Return E.*

Let us assume that the protocols are run by n players, $\{\mathcal{P}_1, \ldots, \mathcal{P}_n\}$.

Before the game takes place the players run the initialization protocol (Protocol 21).

Protocol 21 (Initialization)

1. *Players choose a one-way function f;*

2. *Players choose the size of the embedding, let us assume that is k.*

The deck of cards is shuffled when a player wants a card of the deck. In Protocol 22 let us assume that $\mathcal{P}_t$ wants a new card. We need the following notation in Protocol 22:

- D_i is the set of 52 values chosen by $\mathcal{P}_i$ (these values are mapped to cards in the deck);
- H_i is the set of cards owned by $player_i$, *i.e.* $\mathcal{P}_i$'s hand;
- G_i is the set of cards not owned by $\mathcal{P}_i$, *i.e.* that are not in $\mathcal{P}_i$'s hand, $G_i = D_i - H_i$; let $\zeta = |G_i|$, $1 \leq \zeta \leq 52$.

Protocol 22 (Card shuffling)

1. *Players in* $\{\mathcal{P}_1, \ldots, \mathcal{P}_{t-1}, \mathcal{P}_{t+1}, \ldots, \mathcal{P}_n\}$ *go through the following steps:*
   (a) *Choose a permutation* π_g *of 52 elements;*
   (b) *Embed the* 52 *values of* π_g *using Procedure 11 and obtain* G*; players are now committed to permutation* π_g *and cannot change it;*
   (c) *Send* G *to* $\mathcal{P}_t$*;*
2. *Each player* $\mathcal{P}_i$ *in* $\{\mathcal{P}_1, \ldots, \mathcal{P}_{t-1}, \mathcal{P}_{t+1}, \ldots, \mathcal{P}_n\}$ *does:*
   (a) *Choose* 52 *values at random* $D_i = \{x_{i,1}, \ldots, x_{i,52}\}$*, and map each value to a card in the deck;*
   (b) *Compute the set* $F_i = \{f_{i,1}, \ldots, f_{i,52}\}$*, where* $f_{i,j} = f(x_{i,j})$ *and* $x_{i,j} \in D_i$*;*
   (c) *Permute the order of the elements in* F_i *using* π_g *to get* $F_i' = \{f_{i,\pi_g(1)}, \ldots, f_{i,\pi_g(52)}\}$*;*
   (d) *Send* F_i' *to* $\mathcal{P}_t$ *(*F_i' *represents the deck of cards face down);* $\mathcal{P}_i$ *is committed to these values and their order;*
   (e) $\mathcal{P}_i$ *embeds each card* $x_{i,j}$ *in* G_i *using Procedure 11, and obtains* $E_i = \{e_{i,1}, \ldots, e_{i,\zeta}\}$*,* $\zeta = |G_i|$*;*
   (f) *Choose a permutation* π_i *and permute the order of the elements in* E_i *to get* $E_i' = \{e_{i,\pi_i(1)}, \ldots, e_{i,\pi_i(\zeta)}\}$*;*
   (g) *Send* E_i' *to* $\mathcal{P}_t$*;*

3. *$\mathcal{P}_t$ tries to open the cards in $\{E'_1, \ldots, E'_{t-1}, E'_{t+1}, \ldots, E'_n\}$ using Protocol 20 (Oblivious Transfer) with the rest of players $\{\mathcal{P}_1, \ldots, \mathcal{P}_{t-1}, \mathcal{P}_{t+1}, \ldots, \mathcal{P}_n\}$;*

   (a) *For every embedding value $e_{i,j}$ in $E'_i = \{e_{i,\pi_i(1)}, \ldots, e_{i,\pi_i(\zeta)}\}$, $\mathcal{P}_t$ does:*

      i. *For every $n_{i,j,l}$ in $e_{i,j} = \{R_{i,j}, n_{i,j,1}, \ldots, n_{i,j,k}, (u_{i,j}, v_{i,j})\}$, $e_{i,j} \in E_i$, $\mathcal{P}_t$ $\mathcal{P}_t$ and $\mathcal{P}_i$ run Protocol 20 with $n_{i,j,l}$;*

      ii. *If $\mathcal{P}_t$ can factor k of the $n_{i,j,l}$ values then she has enough information to open the embedding and get a value $x_{i,j}$;*

   (b) *Let us assume that $\mathcal{P}_t$ has opened s values $x_{i,j}$, and T_i is the set of $x_{i,j}$ values known by $\mathcal{P}_t$; now $\mathcal{P}_t$ computes $T_i = \{y_{i,1}, \ldots, y_{i,s}\}$, where $y_{i,j} = f(x_{i,j})$ and $x_{i,j} \in T_i$; T_i is a subset of F_i (see step 2e) and also T_i is a subset of cards that are not in $\mathcal{P}_i$'s hand, H_i; $\mathcal{P}_t$ knows the $x_{i,j}$ values but she does not know the card value associated to $x_{i,j}$;*

   (c) *$\mathcal{P}_t$ computes $C_i = \{c_{i,1}, \ldots, c_{i,s}\}$ with $T_i = \{y_{i,1}, \ldots, y_{i,s}\}$ and $F'_i = \{f_{i,\pi_g(1)}, \ldots, f_{i,\pi_g(52)}\}$, where $\forall y_{i,j} \in T_i \; \exists! f_{i,\pi_g(j)} \in F'_i$ so that $y_{i,j} \equiv f_{i,\pi_g(j)}$ and $c_{i,j} = \pi_g(j)$.*

4. *$\mathcal{P}_t$ computes the free cards, $C_t = C_1 \bigcap C_2 \ldots C_{t-1} \bigcap C_{t+1} \ldots C_n$.*

At this stage, only $\mathcal{P}_t$ and one of the remaining players interact to draw one card. Let us assume that $\mathcal{P}_t$ interacts with $\mathcal{P}_i$ in Protocol 23.

Protocol 23 (Card drawing)

1. *$\mathcal{P}_i$ randomly chooses 52 new values χ_j; each value is mapped to one card of the deck, $\Delta = \{\chi_1, \ldots, \chi_{52}\}$;*

2. *$\mathcal{P}_i$ computes the set $\Delta = \{\delta_1, \ldots, \delta_{52}\}$, where $\delta_j = f(\chi_j)$, $\chi_j \in \Delta$;*

3. *$\mathcal{P}_i$ sends Δ to $\mathcal{P}_t$;*

4. *For each χ_j in G_i player $\mathcal{P}_i$ does:*

   (a) *Embed χ_j in G_i with Procedure 11 to obtain λ_j.*

5. *$\mathcal{P}_i$ computes the set $\Lambda = \{\lambda_1, \ldots, \lambda_\zeta\}$;*

6. *$\mathcal{P}_i$ uses the permutation π_i (see Step 2f of Protocol 22) and permutes the order of elements in Λ to get $\Lambda' = \{\lambda_{\pi_i(1)}, \ldots, \lambda_{\pi_i(\zeta)}\}$;*

7. *$\mathcal{P}_i$ sends Λ' to $\mathcal{P}_t$;*

8. *$\mathcal{P}_t$ chooses a free card; she knows which card in the permutation π_i is free because she has computed the set C_i and knows the relation between C_i and π_g; let us assume that she chooses λ_l;*

9. *For each $n_{l,j}$ in $\lambda_l = \{R_l, n_{l,1}, \ldots, n_{l,k}, (u_l, v_l)\}$ $\mathcal{P}_t$ and $\mathcal{P}_i$ do:*

   (a) *Run Protocol 20 with $n_{l,j}$ in λ_l;*

10. *If $\mathcal{P}_t$ gets all k factors of the embedding of the free card, she recovers χ and computes $\delta_l \equiv f(\chi)$; she looks for δ_l in Δ and the index of δ_l in Δ (call it w) is the value of the card she has drawn.*

Protocol analysis

The above protocol description allows us to analyze the number of messages, the total length of messages and the computational cost of proposal [Yun85].

The number of messages, the total length of the messages and the computational cost of Protocol 22 are presented in Table 3.30.

A step can be computed by one player, a group of players or all players. We have counted the cost of one step as the number of players times the operations in the step. Thus, the average cost per player of a protocol can be calculated by dividing the total cost by n, where n is the number of players.

Remember that ζ in Table 3.30 is the number of cards not owned by any player $\mathcal{P}_i$. It is difficult to estimate ζ; we only know that $1 \leq \zeta \leq 52$.

Table 3.31 shows the costs of the drawing protocol (Protocol 23).

The oblivious transfer protocol (Protocol 20) is used in the shuffling and drawing protocols. Table 3.32 shows the cost of Protocol 20.

Table 3.30: Costs of Yung's shuffling protocol

	Number of messages	Total length of messages	Computational cost
Card shuffling	$3(n-1)(1+r)$	$(n-1)[p](k+3)(52+\zeta)+$ $+(n-1)[p](3k\zeta)+$ $+(n-1)[H(m)]52$	$(n-1)k\rho(2k-1)(52+\zeta)+$ $+(n-1)k\zeta(6\rho+2\xi)$
Step 1	$(n-1)$	$(n-1)(k+3)[p]52$	$52k\rho(2k-1)$
Step 1a			ϵ
Step 1b			$52k\rho(2k-1)$
Step 1c	1	$52[p](k+3)$	ϵ
Step 2	$2(n-1)$	$(n-1)[H(m)]52+$ $+(n-1)\zeta[p](k+3)$	$\zeta\rho k(k-1)$
Step 2a			ϵ
Step 2b			ϵ
Step 2c			ϵ
Step 2d	1	$[H(m)]52$	ϵ
Step 2e			$\zeta\rho k(2k-1)$
Step 2f			ϵ
Step 2g	1	$\zeta[p](k+3)$	ϵ
Step 3	$3\zeta(n-1)$	$3\zeta(n-1)k[p]$	$\zeta k(n-1)(6\rho+2\xi)$
Step 3a	3ζ	$\zeta k[p]3$	$\zeta k(6\rho+2\xi)$
Step 3(a)i	3	$k3[p]$	$k(6\rho+2\xi)$
Step 3(a)ii			ϵ
Step 3b			ϵ
Step 3c			ϵ
Step 4			ϵ

In Step 2 of Procedure 11 we have assumed that Lagrange interpolation is used. The cost shown in Table 3.33 is based on this assumption.

Table 3.34 summarizes the security properties of proposal [Yun85].

Furthermore, this proposal allow player drop-out. If a player $\mathcal{P}_j$ quits during a game the rest of players can go on playing. If $\mathcal{P}_j$ leaves the game before shuffling, the cards in $\mathcal{P}_j$'s hand become free cards and are available to the rest of players. If $\mathcal{P}_j$ leaves the game after the cards have been shuffled, any of the remaining players can run the drawing protocol. The cards in $\mathcal{P}_j$'s hand are not available, so the player

Table 3.31: Costs of Yung's drawing protocol

	Number of messages	Total length of messages	Computational cost
Card drawing	$2+3k$	$52[H(m)]+[p](\zeta(k+3)+3k)$	$k(\rho(\zeta(2k-1)+6)+2\xi)$
Step 1			ϵ
Step 2			ϵ
Step 3	1	$52[H(m)]$	ϵ
Step 4			$\zeta(k\rho(2k-1))$
Step 4a			$k\rho(2k-1)$
Step 5			ϵ
Step 6			ϵ
Step 7	1	$\zeta[p](k+3)$	ϵ
Step 8			ϵ
Step 9	$k3$	$3k[p]$	$k(6\rho+2\xi)$
Step 9a	3	$3[p]$	$6\rho+2\xi$
Step 10			ϵ

who wanted a card does not obtain a card in $\mathcal{P}_j$'s hand.

This protocol is quite complex and is insecure against player coalitions. If some player $\mathcal{P}_i$ sends π_g to $\mathcal{P}_t$ then she can choose the best free cards. $\mathcal{P}_i$ and $\mathcal{P}_t$ can discover some cards of $\mathcal{P}_j$'s hand. In practice, this protocol is for two players only. Also, at the end of the game players must show all information, so that their strategy is revealed.

Table 3.32: Costs of the oblivious transfer protocol

	Number of messages	Total length of messages	Computational cost
Protocol 20	3	$3[p]$	$6\rho + 2\xi$
Step 1			ϵ
Step 2			ρ
Step 3	1	$[p]$	ϵ
Step 4			ϵ
Step 5			ρ
Step 6	1	$[p]$	ϵ
Step 7			$(2\xi + 4\rho)$
Step 8	1	$[p]$	ϵ
Step 9			ϵ
Step 10			ϵ

Table 3.33: Costs of the embedding procedure

	Computational cost
Procedure 11	$k\rho(2k-1)$
Step 1	ϵ
Step 2	$2k(k-1)\rho$
Step 3	$k\rho$
Step 4	ϵ

Table 3.34: Security properties of Yung's protocol suite

Uniqueness of cards	X
Uniform random distribution of cards	X
Cheating detection with a very high probability	X
Complete confidentiality of cards	X
Minimal effect of coalitions	
Complete confidentiality of strategy	

3.3.5 A secure poker protocol that minimizes the effect of player coalitions (Crépeau)

[Cré85] is an important contribution to mental poker protocols, because the security requirements of mental poker were first identified and listed in this paper. Also, the first mental poker protocol secure against coalitions of players was presented.

Assume that $\mathcal{P}_1, \ldots, \mathcal{P}_n$ want to play poker. Each card of the deck is mapped to a value of the set $D = \{1, 2, \ldots, 52\}$. Each $\mathcal{P}_i$ picks a permutation π_i of $\{1, 2, \ldots, 52\}$ and keeps it secret. The shuffled deck is $\pi_n \circ \cdots \circ \pi_2 \circ \pi_1$, *i.e.* the functional composition of these permutations. To get a card, player $\mathcal{P}_i$ picks a value v in D that nobody else has picked before, and she gets her card by computing $\pi_n \cdots \pi_2\, \pi_1(v)$. But, since permutations are kept secret, the player must use a special trick in order to get her value. Thus, $\mathcal{P}_i$ gets the values $k' = \pi_{i-1}(\cdots(\pi_2 \circ (\pi_1(v))))$ in the clear. $\mathcal{P}_i$ secretly computes $v'' = \pi_i(v')$ and she uses a hiding protocol to compute $\pi_n \cdots \pi_{i+1}(v'')$ in such a way that rest of players do not know the value of $\mathcal{P}_i$'s card.

Let us assume that $\mathcal{P}_1, \ldots, \mathcal{P}_n$ want to play poker, and use the first option of the protocol based in [RSA77] and [BG85].

Each player must specify some game parameters before the beginning of the game. This is done with the Protocol 24.

Protocol 24 (Initialization)

1. *Each player $\mathcal{P}_i$ does the following:*
   - (a) *$\mathcal{P}_i$ selects two larges prime numbers p and q;*
   - (b) *$\mathcal{P}_i$ computes $n = pq$ and chooses a key pair (e, d) [RSA77];*
   - (c) *$\mathcal{P}_i$ defines the following sets:*
     - i. *the value space of $\mathcal{P}_i$ is $V_i = \{1, \ldots, 52\}$;*
     - ii. *The seed space of $\mathcal{P}_i$ is $S_i \in \mathbb{Z}_n^*$;*
     - iii. *The unlocking key space of $\mathcal{P}_i$ is $K_i \in \mathbb{Z}_n^*$;*
     - iv. *The mask space of $\mathcal{P}_i$ is $M_i \in \mathbb{Z}_n^*$;*
     - v. *The ambiguous value space of $\mathcal{P}_i$ is $A_i \in \mathbb{Z}_n^*$;*

vi. The coded message space of $\mathcal{P}_i$ is $C_i \in V \times S$;

(d) $\mathcal{P}_i$ defines the functions she uses in the protocol;

i. $f(x, z)$ denotes the z least significant bits of x;

ii. Locking function L_i: $L_i(v, s) = (f(s, 6) \oplus v, (s^e \bmod n)) = c = (c_v, c_s)$ where $v \in V_i$, $s \in S_i$ and $c \in C_i$;

iii. Unlocking function U_i : $U_i(c = (c_v, c_s)) = ((c_s)^d \bmod n)$, where $c \in C_i$;

iv. Hiding function H_i: $H_i(m, c = (c_v, c_s)) = (c_s m^e) \bmod n$, where $m \in M_i$, $c \in C_i$ and $s \in S_i$;

v. Revealing function R_i: $R_i(m, c = (c_v, c_s), k) = f((m^{-1}k \bmod n), 6) \oplus (c_v)$, where $m \in M_i$, $c \in C_i$ and $k \in K_i$;

vi. Public one-way function (Blum-Goldwasser [BG85]): $O(m) = BG(m, s)$, where $m \in M_i$ and $s \in S_i$.

Every player $\mathcal{P}_i$ runs Protocol 25 in order to shuffle the deck of cards.

$\mathcal{P}_i$ chooses a permutation π_i of 52 values. For each value v_i in $\{1, \ldots, 52\}$ she computes $\pi_i(v_i)$ and locks the result using function L_i. In other words, every $\pi_i(v_i)$ is encrypted so that $\mathcal{P}_i$ is committed to π_i. The encrypted values are made public as $\{l_{i,1}, \ldots, l_{i,52}\}$. If a player needs the permutation of a value v with π_i in order to compute her card she knows that $l_{i,v}$ is the encryption of $\pi_i(v)$.

Protocol 25 (Card shuffling)

1. *Each player $\mathcal{P}_i$ in $\{\mathcal{P}_1, \ldots, \mathcal{P}_n\}$ does:*

   (a) $\mathcal{P}_i$ picks at random a permutation π_i of 52 values;

   (b) $\mathcal{P}_i$ sends L_i, H_i and R_i;

   (c) $\mathcal{P}_i$ picks at random $\{s_{i,1}, \ldots, s_{i,52}\} \in S_i$;

   (d) For each v_j in $\{1, \ldots, 52\}$ $\mathcal{P}_i$ computes $l_{i,v_j} = L_i(\pi_i(v_j), s_{i,v_j})$;

   (e) $\mathcal{P}_i$ publishes $\{l_{i,1}, \ldots, l_{i,52}\}$; note that, for every value v_j in D, $\mathcal{P}_i$ sends $\pi_i(v)$ in encrypted form.

When a player, for instance $\mathcal{P}_i$, wants a card all players run Protocol 26.

Protocol 26 (Card drawing)

1. *$\mathcal{P}_i$ chooses at random $v_0 \in D$ and marks it as used;*
2. *$\mathcal{P}_i$ sends v_0 to $\mathcal{P}_1$;*
3. *For $\mathcal{P}_j$ in $\{\mathcal{P}_1, \ldots, \mathcal{P}_{i-1}\}$ do:*
   (a) *$\mathcal{P}_j$ receives v_{j-1} from the previous player;*
   (b) *$\mathcal{P}_j$ computes $v_j = \pi_j(v_{j-1})$;*
   (c) *$\mathcal{P}_j$ sends v_j to $\mathcal{P}_{j+1}$;*
4. *$\mathcal{P}_i$ secretly computes $v_i = \pi_i(v_{i-1})$;*
5. *$\mathcal{P}_i$ picks $m_{i+1} \in M_{i+1}$ and she computes $h_{i+1} = H_{i+1}(m_{i+1}, l_{i+1,v_i})$;*
6. *$\mathcal{P}_i$ sends h_{i+1} to $\mathcal{P}_{i+1}$;*
7. *For $\mathcal{P}_j$ in $\{\mathcal{P}_{i+1}, \ldots, \mathcal{P}_n\}$ do:*
   (a) *$\mathcal{P}_j$ receives h_j from $\mathcal{P}_i$;*
   (b) *$\mathcal{P}_j$ computes $k_j = U_j(h_j)$, and sends k_j to $\mathcal{P}_i$;*
   (c) *$\mathcal{P}_i$ computes $v_j = R_j(m, l_{i+1,v_i}, k_j)$;*
   (d) *If $\mathcal{P}_j \neq \mathcal{P}_n$ then*
      i. *$\mathcal{P}_i$ picks $m_{j+1} \in M_{j+1}$ and she computes $h_{j+1} = H_{j+1}(m_{j+1}, l_{j+1,v_j})$;*
      ii. *$\mathcal{P}_i$ sends h_{j+1} to $\mathcal{P}_{j+1}$;*
8. *Finally $\mathcal{P}_i$ obtains her card.*

When the game is finished, all players disclose and prove their π_i, thus revealing all hands. The strategy of players becomes public.

Protocol analysis

In this section we present the analysis of Crépeau's protocol suite.

Table 3.35 shows the number of messages, the total length of messages and the computational cost of the shuffling protocol.

Table 3.35: Costs of Crépeau's shuffling protocol

	Number of messages	Total length of messages	Computational cost
Card shuffling	$2n(n-1)$	$52[p](n-1)n$	$52n\xi$
Step 1	$2n(n-1)$	$52[p](n-1)n$	$52n\xi$
Step 1a			ϵ
Step 1b	$(n-1)$		ϵ
Step 1c			ϵ
Step 1d			52ξ
Step 1e	$(n-1)$	$52[p](n-1)$	ϵ

Table 3.36 shows the number of messages, the total length of messages and the computational cost of the drawing protocol.

We have assumed that $\mathcal{P}_i$ is in the middle of the set of players, *i.e.* $|\mathcal{P}_1, \ldots, \mathcal{P}_{i-1}| = |\mathcal{P}_{i+1}, \ldots, \mathcal{P}_n|$. Thus, Steps 3 and 7 of Protocol 26 are run by $(n-1)/2$ players, respectively.

Finally, Table 3.37 shows the security properties satisfied by the proposal [Cré85].

In [Cré85], like in [GM82], a player $\mathcal{P}_i$ might draw a card which is in another player's hand or even draw more than one card from the deck. This cheating is detected when the game is over and players reveal their secret information. The verification implies revealing the strategy of players.

Table 3.36: Costs of Crépeau's drawing protocol

	Number of messages	Total length of messages	Computational cost
Card drawing	$(\frac{n-1}{2})^2+1$	$[6](\frac{n+1}{2})+[p](\frac{n^2-4n+7}{4})$	$(\rho+\xi)(\frac{n^2-2n+5}{4})$
Step 1			ϵ
Step 2	1	$[6]$	ϵ
Step 3	$\binom{n-1}{2}$	$\frac{n-1}{2}[6]$	ϵ
Step 3a			ϵ
Step 3b			ϵ
Step 3c	1	$[6]$	ϵ
Step 4			ϵ
Step 5			$(\xi+\rho)$
Step 6	1	$[p]$	ϵ
Step 7	$(\frac{n-1}{2})(\frac{n-3}{2})$	$(\frac{n-1}{2})(\frac{n-3}{2})[p]$	$(\frac{n-1}{2})((\xi+\rho)+(\frac{n-3}{2})(\xi+\rho))$
Step 7a			ϵ
Step 7b			ξ
Step 7c			ρ
Step 7d	$\frac{n-1}{2}-1$	$(\frac{n-1}{2}-1)[p]$	$(\frac{n-1}{2}-1)(\rho+\xi)$
Step 7(d)i			$(\rho+\xi)$
Step 7(d)ii	1	$[p]$	ϵ
Step 8			ϵ

Table 3.37: Security properties of Crépeau's protocol suite

Uniqueness of cards	X
Uniform random distribution of cards	X
Cheating detection with a very high probability	X
Complete confidentiality of cards	X
Minimal effect of coalitions	X
Complete confidentiality of strategy	

3.3.6 A zero-knowledge poker protocol that achieves confidentiality of the players' strategy or how to achieve an electronic poker face (Crépeau)

[Cré86] is the first mental poker protocol that satisfies the security requirements laid out in [Cré85]. The protocol keeps secret the strategy of players and is an extension of [Cré85].

In [Cré85] a player could take cards that are in the hand of another player, or still in the deck. For that reason, at the end of the game players had to reveal their permutations for correctness to be verified. In [Cré86] a deck of cards and a shuffled deck of cards are defined as in [Cré85]. When a player $\mathcal{P}_i$ picks a card in [Cré86] she follows the same steps as in [Cré85] from $\mathcal{P}_1$ to $\mathcal{P}_{i-1}$. Nevertheless, $\mathcal{P}_i$ runs the *all-or-nothing disclosure of secrets* (ANDOS) [BCR86] from $\mathcal{P}_{i+1}$ to $\mathcal{P}_n$ to obtain the result of the permutations. Furthermore, the player picks also some random information linked with her card. Players who wish to open their hands at the end of a game reveal this information. If nobody is cheating, the information revealed at the end of the game should match the information initially picked.

Let us recall the ANDOS protocol[BCR86]. Without loss of generality let us assume that the protocol is run by $\mathcal{P}_1$ and $\mathcal{P}_2$, where $\mathcal{P}_1$ is the seller of secrets and $\mathcal{P}_2$ is the buyer.

Let $X = \{x_1, \ldots, x_m\}$ be a set of secrets, where each secret x_i is r bits long, *i.e.* $x_i = \{x_{i,1}, \ldots, x_{i,r}\}$. $\mathcal{P}_1$ and $\mathcal{P}_2$ use Protocol 27 in order to exchange a secret.

$\mathcal{P}_1$ obtain the initial parameters that are needed to run the ANDOS protocol using Procedure 12.

Procedure 12 (ANDOS: init)

1. *Choose p and q, two large prime numbers;*
2. *Compute $n = pq$;*
3. *Choose $y \in \overline{Q}_n$ such that $\left(\frac{y}{n}\right) = +1$;*
4. *Return y, p, q and n.*

$\mathcal{P}_1$ uses Procedure 13 to encrypt her secrets. $\mathcal{P}_2$ computes one question for every secret using Procedure 14. $\mathcal{P}_2$ uses Protocol 27 to prove to $\mathcal{P}_1$ that questions are well constructed. Further details on the ANDOS protocol can be found in [BCR86].

Protocol 27 (ZKP $(X = \{x_1, \ldots, x_m\}$,$Z = \{z_1, \ldots, z_m\}$,$Q = \{q_1, \ldots, q_m\}$,$\pi)$**)**

1. *$\mathcal{P}_1$ and $\mathcal{P}_2$ agree on a security parameter s;*
2. *For $i = 1$ to s, $\mathcal{P}_2$ does:*
   (a) *Use Procedure 14 with Z and y, and obtain Q_i, Δ_i, A_i and π_i;*
   (b) *Send Q_i and keep secret Δ_i, A_i and π_i;*
3. *$\mathcal{P}_1$ selects a random subset $S \subseteq \{1, \ldots, s\}$ and sends it to $\mathcal{P}_2$ as a challenge;*
4. *For $i = 1$ to s $\mathcal{P}_1$ and $\mathcal{P}_2$ do:*
   (a) *If $i \in S$ do:*
      i. *$\mathcal{P}_2$ sends Δ_i, A_i and π_i to $\mathcal{P}_1$;*
      ii. *$\mathcal{P}_1$ uses Procedure 15 with Z, Q_i, Δ_i, A_i and π_i in order to verify whether Q_i has been computed properly;*
   (b) *If $i \notin S$ do:*
      i. *$\mathcal{P}_2$ computes $\pi_i' = \pi_i^{-1} \circ \pi$;*
      ii. *$\mathcal{P}_2$ uses Procedure 16 with Δ, Δ_i, A, A_i, y and π_i', and obtains Δ_i' and A_i';*
      iii. *$\mathcal{P}_2$ sends Δ_i', A_i' and π_i' to $\mathcal{P}_1$;*
      iv. *$\mathcal{P}_1$ uses Procedure 15 with Q, Q_i, Δ_i', A_i' and π_i' in order to verify whether Q_i has been computed properly.*

The set X contains the secrets. $\mathcal{P}_1$ uses Procedure 13 in order to encrypt her secrets. The cryptosystem used is probabilistic, like the one presented in [GM82].

Procedure 13 $(X = \{x_1, \ldots, x_m\})$

1. *For each* x_i *in* X *do:*

   (a) *For each bit* $x_{i,j}$ *of* $x_i = \{x_{i,1}, \ldots, x_{i,r}\}$ *do:*

      i. *Choose at random a value* $t_{i,j}$, *where* $t_{i,j} \in Z_m^*$;

      ii. *Compute* $z_{i,j} = t_{i,j}^2 y^{x_{i,j}}$;

   (b) *Form the set* $t_i = \{t_{i,1}, \ldots, t_{i,r}\}$;

   (c) *Form the set* $z_i = \{z_{i,1}, \ldots, z_{i,r}\}$;

2. *Form the set* $T = \{t_1, \ldots, t_m\}$;

3. *Form the set* $Z = \{z_1, \ldots, z_m\}$;

4. *Return* T *and* Z.

$\mathcal{P}_2$ cannot request a secret in the clear because her interest would be revealed. Thus, she uses Procedure 14 in order to compute a question for every secret. Questions are encrypted so they do not reveal the secret they are associated to. When $\mathcal{P}_2$ wants a secret she asks a question to $\mathcal{P}_1$, who gets no information on which secret $\mathcal{P}_2$ is after.

Procedure 14 ($Z = \{z_{1,1}, \ldots, z_{m,r}\}$,$y$)

1. *Choose at random a permutation* π *of* $\{1, \ldots, m\}$ *elements;*

2. *Obtain at random the set* $\Delta = \{\delta_{1,1}, \ldots, \delta_{m,r}\}$, *where* $\delta_{k,j}$ *in* Z_m^*, $k \in \{1, \ldots, m\}$ *and* $j \in \{1, \ldots, r\}$;

3. *Obtain at random the set* $A = \{a_{1,1}, \ldots, a_{m,r}\}$, *where* $a_{k,j} \in \{0, 1\}$, $k \in \{1, \ldots, m\}$ *and* $j \in \{1, \ldots, r\}$;

4. *Compute* $Q = \{q_{1,1}, \ldots, q_{m,r}\}$, *where* $q_{k,j} = z_{i,j} \delta_{k,j}^2 y^{a_{k,j}}$, $k \in \{1, \ldots, m\}$, $j \in \{1, \ldots, r\}$ *and* $i = \pi^{-1}(k)$;

5. *Return* Q, Δ, A *and* π.

$\mathcal{P}_2$ may try to cheat so that one question reveals more than one secret. $\mathcal{P}_1$ must be sure that questions are fair and one question only reveals one secret. The verification in Protocol 27 opens some questions. $\mathcal{P}_1$ verifies the opened questions using Procedure 15.

Procedure 15 (Z,Q,Δ,A,π)

1. *For each* $z_{i,j}$ *in* $Z = \{z_{1,1}, \ldots, z_{m,r}\}$, *where* $Q = \{q_{1,1}, \ldots, q_{m,r}\}$, $\Delta = \{\delta_{1,1}, \ldots, \delta_{m,r}\}$ *and* $A = \{a_{1,1}, \ldots, a_{m,r}\}$ *do:*
   (a) *Compute* $q'_{i,j} = z_{i,j}\delta_{i,j}^2 y^{a_{i,j}}$;
   (b) *Verify* $q'_{i,j} \stackrel{?}{=} q_{\pi(i),j}$;

Player $\mathcal{P}_2$ uses Procedure 16 to compute the sets A'_i and Δ'_i. These sets are used when $\mathcal{P}_1$ verifies a set of questions using the Procedure 15.

Procedure 16 ($\Delta, \Delta_i, A, A_i, y, \pi$)

1. *For each* $\delta_{j,k}$ *in* Δ, *where* $\Delta = \{\delta_{1,1}, \ldots, \delta_{m,r}\}$,$\Delta_i = \{\delta_{i,1,1}, \ldots, \delta_{i,m,r}\}$,$A = \{a_{1,1}, \ldots, a_{m,r}\}$,$A_i = \{a_{i,1,1}, \ldots, a_{i,m,r}\}$ *do:*
   (a) *Compute* $a'_{i,j,k} = a_{j,k} \otimes a_{i,\pi(j),k}$;
   (b) *If* $a'_{i,j,k} \equiv a_{i,j} \equiv 1$ *compute* $\delta'_{i,j,k} = \delta_{i,\pi(j),k}(y\delta_{i,j})^{-1}$;
   (c) *Else compute* $\delta'_{i,j,k} = \delta_{i,\pi(j),k}(\delta_{i,j})^{-1}$;
2. *Form the set* $A'_i = \{a'_{i,1,1}, \ldots, a'_{i,m,r}\}$;
3. *Form the set* $\Delta'_i = \{\delta'_{i,1,1}, \ldots, \delta'_{i,m,r}\}$;
4. *Return* A'_i *and* Δ'_i.

$\mathcal{P}_2$ and $\mathcal{P}_1$ run Protocol 28 when $\mathcal{P}_2$ wants a secret from $\mathcal{P}_1$. $\mathcal{P}_2$ asks to $\mathcal{P}_1$ the question she wants. $\mathcal{P}_1$ computes the Jacobi symbol for each element in the question, and tells to $\mathcal{P}_2$ whether the element is a quadratic residue. With this information $\mathcal{P}_2$ can compute the secret.

Protocol 28 (Get a secret (i,m,π,$Q = \{q_{1,1}, \ldots, q_{m,r}\}$,$A = \{a_{1,1}, \ldots, a_{m,r}\}$))

1. *$\mathcal{P}_2$ computes $k = \pi^{-1}(i)$;*
2. *$\mathcal{P}_2$ sends k to $\mathcal{P}_1$;*
3. *For $j = 1$ to r, $\mathcal{P}_1$ does:*
   (a) *Compute $\beta_j = \left(\frac{q_{k,j}}{n}\right)$, where $q_{k,j} \in Q$;*
   (b) *If $\beta_j \equiv +1$ set $\sigma_j = 0$;*
   (c) *If $\beta_j \equiv -1$ set $\sigma_j = 1$;*
   (d) *Compute τ_j, where $\tau_j^2 = q_{k,j} y^{\sigma_j}$;*
4. *$\mathcal{P}_1$ forms the set $S = \{\sigma_1, \ldots, \sigma_r\}$;*
5. *$\mathcal{P}_1$ forms the set $T = \{\tau_1, \ldots, \tau_r\}$;*
6. *$\mathcal{P}_1$ sends S and T to $\mathcal{P}_2$;*
7. *For $j = 1$ to r, $\mathcal{P}_2$ verifies $\tau_j^2 \equiv q_{k,j} y^{\sigma_j}$;*
8. *$\mathcal{P}_2$ computes the secret c with A and S.*

Each player $\mathcal{P}_i$ in $\{\mathcal{P}_1, \ldots, \mathcal{P}_n\}$ runs Protocol 29 in order to shuffle the deck of cards. $\mathcal{P}_i$ chooses a permutation of 52 elements. She computes and publishes the encryption of each element in the permutation. All players know that $z_{i,j}$ is the encrypted permutation of j. Each of the remaining players asks a question for each cryptogram. $\mathcal{P}_i$ does not know the relation between the cryptograms and the questions. Each player proves in zero-knowledge to $\mathcal{P}_i$ that her questions are fair. This is done using Protocol 27.

Protocol 29 (Card shuffling)

1. *Each player $\mathcal{P}_i$ in $\{\mathcal{P}_1, \ldots, \mathcal{P}_n\}$ does:*
   (a) *Run Procedure 12 and obtain y_i, p_i, q_i and n_i;*

(b) *Choose a permutation* π_i *of* 52 *elements;*

(c) *Use Procedure 13 with* $\pi_i = \{x_1, \ldots, x_{52}\}$, *where* $x_i \in \{1, \ldots, 52\}$ *and obtain* $T_i = \{t_{i,1}, \ldots, t_{i,52}\}$ *and* $Z_i = \{z_{i,1}, \ldots, z_{i,52}\}$;

(d) *For each* $\mathcal{P}_j$ *in* $\{\mathcal{P}_1, \ldots, \mathcal{P}_{i-1}, \mathcal{P}_{i+1}, \ldots, \mathcal{P}_n\}$ *do:*

i. $\mathcal{P}_i$ *sends* Z_i *to* $\mathcal{P}_j$;

ii. $\mathcal{P}_j$ *uses Procedure 14 with* Z_i *and* y_i, *and obtains* $Q_{i,j}$, $\Delta_{i,j}$, $A_{i,j}$ *and* $\pi_{i,j}$; *the set* $Q_{i,j}$ *contains one question for each secret of* $\mathcal{P}_i$, i.e. *for* $\mathcal{P}_i$*'s permutation;*

iii. $\mathcal{P}_j$ *sends* $Q_{i,j}$ *to* $\mathcal{P}_i$ *and keeps secret* $\Delta_{i,j}$, $A_{i,j}$ *and* $\pi_{i,j}$;

iv. $\mathcal{P}_i$ *and* $\mathcal{P}_j$ *run Protocol 27 with* Z_i, $Q_{i,j}$, $\Delta_{i,j}$ *and* $\pi_{i,j}$; *with Protocol 27,* $\mathcal{P}_j$ *proves to* $\mathcal{P}_i$ *that all questions are fair with probability* $1 - (1/2)^s$, *where* s *is a security parameter;*

(e) $\mathcal{P}_i$ *forms the set* $P_i = \{\pi_{i,1}, \ldots, \pi_{i,i-1}, \pi_{i,i+1}, \ldots, \pi_n\}$

Player $\mathcal{P}_i$ runs Protocol 30 with the rest of players when she wants a card. She obtains the permutations in the clear from $\mathcal{P}_1$ to $\mathcal{P}_{i-1}$. $\mathcal{P}_i$ obtains the rest of permutations, *i.e.* those from $\mathcal{P}_{i+1}$ to $\mathcal{P}_n$, using Protocol 28 (ANDOS get a secret).

Protocol 30 (Card drawing (P_i, Q_i, A_i)**)**

1. $\mathcal{P}_i$ *picks a free value* k_0 *in* $\{1, \ldots, 52\}$ *and marks it as used;*

2. $\mathcal{P}_i$ *computes* $c_0 = k_0$;

3. *For* $j = 1$ *to* $i - 1$, $\mathcal{P}_i$ *and* $\mathcal{P}_j$ *do:*

   (a) $\mathcal{P}_i$ *sends* c_{j-1} *to* $\mathcal{P}_j$;

   (b) $\mathcal{P}_j$ *computes* $c_j = \pi_j(c_{j-1})$;

   (c) $\mathcal{P}_j$ *sends* c_j *to* $\mathcal{P}_i$;

4. $\mathcal{P}_i$ *computes* $c_i = \pi_i(c_{i-1})$;

5. *For* $j = i + 1$ *to* n, $\mathcal{P}_i$ *and* $\mathcal{P}_j$ *do:*

(a) $\mathcal{P}_i$ and $\mathcal{P}_j$ run Protocol 28 with (c_{j-1},52, $\pi_{j,i}$, $Q_{j,i}$, $A_{j,i}$), where $\pi_{i,j} \in P_i$; in this way, $\mathcal{P}_i$ obtains c_j;

6. At the end of the previous iterations, $\mathcal{P}_i$ has obtained her card c_n.

Protocol analysis

We present the cost of shuffling and drawing using the proposal [Cré86].

Table 3.38 shows the number of messages, the total length of the messages and the computational cost of Protocol 29.

Table 3.38: Costs of Crépeau's shuffling protocol

	Number of messages	Total length of messages	Computational cost
Card shuffling	$(n^2-n)(2s+3)$	$(n^2-n)s52[r]([p]+1)+$ $+(n^2-n)104[r][p]+$ $+(n^2-n)s[r]\frac{105}{2}$	$n\rho[r](n-1)(299s+130)+$ $+n\rho[r]78+n\rho+n(2\xi)$
Step 1	$(n^2-n)(2s+3)$	$(n^2-n)s52[r]([p]+1)+$ $+(n^2-n)104[r][p]+$ $+(n^2-n)s[r]\frac{105}{2}$	$n\rho[r](n-1)(299s+130)+$ $+n\rho[r]78+n\rho+n(2\xi)$
Step 1a			$2\xi+\rho$
Step 1b			ϵ
Step 1c			$52[r]\rho(3/2)$
Step 1d	$(n-1)(2s+3)$	$(n-1)s52[r]([p]+1)+$ $+(n-1)104[r][p]+$ $+(n-1)s[r]\frac{105}{2}$	$(n-1)[r]\rho(299s+130)$
Step 1(d)i	1	$52[r][p]$	ϵ
Step 1(d)ii			$52[r]\rho(5/2)$
Step 1(d)iii	1	$52[r][p]$	ϵ
Step 1(d)iv	$(2s+1)$	$s52[r](2[p]+1)+$ $+s[r](\frac{105}{2})$	$s52[r]\rho(23/4)$
Step 1e			ϵ

Table 3.39 shows the cost of the Protocol 3.39.

Next, we can see the cost of the procedures and protocols used in the suffling and drawing protocols. We have assumed that Steps 4a and 4b of Protocol 27 have the

Table 3.39: Costs of Crépeau's drawing protocol

	Number of messages	Total length of messages	Computational cost
Card drawing	$2(n-1)$	$\frac{n-1}{2}([r]([p]+1)+3[r])$	$\frac{n-1}{2}([r](6\rho+4\xi))$
Step 1			ϵ
Step 2			ϵ
Step 3	$(\frac{n-1}{2})2$	$(\frac{n-1}{2})2[r]$	ϵ
Step 3a	1	$[r]$	ϵ
Step 3b			ϵ
Step 3c	1	$[r]$	ϵ
Step 4			ϵ
Step 5	$(\frac{n-1}{2})2$	$\frac{n-1}{2}([r]([p]+1)+[r])$	$(\frac{n-1}{2})[r](6\rho+4\xi)$
Step 5a	2	$[r]([p]+1)+[r]$	$[r](6\rho+4\xi)$
Step 6			ϵ

same probability. That is why in Table 3.40 their costs are multiplied by $\frac{1}{2}$.

In Steps 3d and 7 of Protocol 28 we may compute an additional product depending on the value of a bit. If the bit is 1 we compute the product, otherwise we do not compute it. We assume the bit can be 1 or 0 wit a probability 1/2. Thus, ρ is multiplied by 1/2. In the same Step 3d a square root is computed. Like in Step 2a of Protocol 17, we assume that computing a square root has a cost of $2\xi + 4\rho$.

In Table 3.43 m denotes the cardinal of the input set, in our case $m = 52$. In Step 1(a)ii of Procedure 13 an additional product is computed only if a bit is 1, which is assumed to happen with probability 1/2.

Again, in Step 4 of Procedure 14 we compute an additional product with probability 1/2 (see Table 3.44).

Table 3.45 shows the cost of Procedure 15. Again, in Step 1a of Procedure 15 we compute a product if a bit is 1, which is assumed to happen with probability 1/2.

Steps 1b and 1c of Procedure 16 are run with the same probability, namely 1/2 (see Table 3.46).

We summarize the security properties of the proposal [Cré86] in Table 3.47.

In conclusion, this protocol suite is a complete solution to the problem presented in [Cré85]. The negative side is the great computational cost incurred, which is pointed

Table 3.40: Costs of Protocol 27

	Number of messages	Total length of messages	Computational cost
Protocol 27	$2s+1$	$sm([r](2[p]+1)+[r])+(1/2)s[r]$	$sm[r]\rho(23/4)$
Step 1			ϵ
Step 2	s	$sm[r][p]$	$sm[r]\rho(5/2))$
Step 2a			$m[r]\rho(5/2)$
Step 2b	1	$m[r][p]$	ϵ
Step 3	1	$(1/2)s[r]$	ϵ
Step 4	s	$s(m[r][p]+m[r][1]+m[r])$	$sm[r]\rho(13/4)$
Step 4a	$(1/2)1$	$(1/2)(m[r][p]+m[r][1]+m[r])$	$(1/2)(m[r]\rho(5/2))$
Step 4(a)i	1	$m[r][p]+m[r][1]+m[r]$	ϵ
Step 4(a)ii			$m[r]\rho(5/2)$
Step 4b	$(1/2)1$	$(1/2)(m[r][p]+m[r][1]+m[r])$	$(1/2)(m[r]\rho 4)$
Step 4(b)i			ϵ
Step 4(b)ii			$m[r]\rho(3/2)$
Step 4(b)iii	1	$(m[r][p]+m[r][1]+m[r])$	ϵ
Step 4(b)iv			$m[r]\rho(5/2)$

out in [Edw94].

Table 3.41: Costs of Protocol 28

	Number of messages	Total length of messages	Computational cost
Protocol 28	2	$[r]([p]+1)+[r]$	$[r](6\rho+4\xi)$
Step 1			ϵ
Step 2			ϵ
Step 3			$[r](2\xi+(1/2)\rho+2\xi+4\rho)$
Step 3a			2ξ
Step 3b			ϵ
Step 3c			ϵ
Step 3d			$(1/2)\rho+2\xi+4\rho$
Step 4			ϵ
Step 5			ϵ
Step 6	1	$[r][1]+[r][p]$	ϵ
Step 7			$[r](\rho+(1/2)\rho)$
Step 8			ϵ

Table 3.42: Costs of Procedure 12

	Computational cost
Procedure 12	$2\xi+\rho$
Step 1	ϵ
Step 2	ρ
Step 3	2ξ
Step 4	ϵ

Table 3.43: Costs of Procedure 13

	Computational cost
Procedure 13	$mr\rho(3/2)$
Step 1	$mr\rho(3/2)$
Step 1a	$r(\rho + (1/2)\rho)$
Step 1(a)i	ϵ
Step 1(a)ii	$\rho + (1/2)\rho$
Step 1b	ϵ
Step 1c	ϵ
Step 2	ϵ
Step 3	ϵ
Step 4	ϵ

Table 3.44: Costs of Procedure 14

	Computational cost
Procedure 14	$mr\rho(5/2)$
Step 1	ϵ
Step 2	ϵ
Step 3	ϵ
Step 4	$mr(\rho + \rho + (1/2)\rho)$
Step 5	ϵ

Table 3.45: Costs of Procedure 15

	Computational cost
Procedure 15	$mr\rho(5/2)$
Step 1	$mr\rho(5/2)$
Step 1a	$(\rho + \rho + (1/2)\rho)$
Step 1b	ϵ

Table 3.46: Costs of Procedure 16

	Computational cost
Procedure 16	$mr\rho(3/2)$
Step 1	$mr\rho(3/2)$
Step 1a	ϵ
Step 1b	$(1/2)2\rho$
Step 1c	$(1/2)\rho$
Step 2	ϵ
Step 3	ϵ
Step 4	ϵ

Table 3.47: Security properties of Crépeau's 1986 protocol

Uniqueness of cards	X
Uniform random distribution of cards	X
Cheating detection with a very high probability	X
Complete confidentiality of cards	X
Minimal effect of coalitions	X
Complete confidentiality of strategy	X

3.3.7 General public key cryptosystems and Mental Poker Protocols (Kurosawa-Katayama-Ogata-Tsujii)

The mental poker protocol presented in [KKOT90] is a nice solution, because it fulfills all properties described in [Cré85].This proposal introduces a new public key residue cryptosystem, which is an additive homomorphism. The homomorphic property is used when the cards are shuffled. Before the game begins, every player computes a key pair for the cryptosystem, and her public key is sent to rest of players. A face-up card of value k is defined as a sum of n values which add to k modulo 52. n is the number of players, *i.e.* in every card sum there is one term per player. When a card k is face down every different summed term is encrypted with a different player's public key. In this way, every player can decrypt only a part of the card. Players shuffle the deck in turn. For every card, players compute n values such that they are zero modulo 52. Each summed term is encrypted using the public key of a different player. Each cryptogram is operated with the cryptogram of the card that has been obtained with the same public key. Thus, the order of cards is permuted. When a player wants a card, every other player decrypts her respective term in the card sum.

Next, a detailed description of the protocol is presented.

Procedure 17 (Public key residue cryptosystem: preparation (r))

1. *Choose two large prime numbers p and q, and compute $n = pq$;*

2. *Compute the following parameters based on r; the plaintext m must be less than r, $0 \leq m < r$;*

   (a) *If $r = 2$ then $\left(\frac{y}{p}\right) = \left(\frac{y}{q}\right) = -1$;*

   (b) *If r is a prime value then:*

      i. *$r|(p-1)$, and $r \not|(q-1)$;*

      ii. *y must be an $r-th$ non-residue, $y \neq z^{53} \bmod n, z \in \{1, \ldots, n-1\}$.*

3. *Return the public key (n, r, y) and the secret key (p, q).*

Procedure 18 (Public key residue cryptosystem: encryption, $P = (n, r, y)$)

1. *Choose a number x at random;*
2. *Compute $C = E_P(m, x) = y^m x^r \bmod n$;*
3. *Return C and x.*

Procedure 19 (Public key residue cryptosystem: decryption. $S = (p, q)$)

1. *For $0 \leq j < r$, compare*

$$C^{(p-1)/r} \bmod p \stackrel{?}{=} \left(y^{(p-1)/r}\right)^j \bmod p$$

The above expression is satisfied when $j = m$, so the message in the clear is j.

2. *Return j.*

Public key residue cryptosystem: homomorphic property : The cryptosystem is additive homomorphic.

1. $E_P(m + n \bmod r) = E_P(m)E_P(n) \bmod n$

$$\begin{aligned}
E_P(m) &= y^m x^r \bmod n \\
E_P(n) &= y^n z^r \bmod n \\
E_P(m+n) &= E_P(m)E_P(n) \bmod n \\
&= y^m x^r y^n x^r \bmod n \\
&= y^{m+n}(xz)^r \bmod n \\
E_P(m+n)^{(p-1)/r} &= \left(y^{m+n}(xz)^r\right)^{(p-1)/r} \bmod p \\
&= \left(y^{m+n}\right)^{(p-1)/r} \left((xz)^r\right)^{(p-1)/r} \bmod p \\
&= \left(y^{m+n}\right)^{(p-1)/r} \left((xz)\right)^{(p-1)} \bmod p \\
&= \left(y^{m+n}\right)^{(p-1)/r} \bmod p \\
\left(y^{m+n}\right)^{(p-1)/r} \bmod p &\stackrel{?}{=} \left(y^{(p-1)/r}\right)^j \bmod p
\end{aligned}$$

Let us assume that the protocols are run by n players, $\{\mathcal{P}_1, \ldots, \mathcal{P}_n\}$.
The first stage is to run Procotol 31 in order to compute a key pair for each player.

Protocol 31 (Preparation)

1. *Each $\mathcal{P}_i$ in $\{\mathcal{P}_1, \ldots, \mathcal{P}_n\}$ does:*
   (a) *Use Procedure 17 and obtain a key pair, where $S_i = (p_i, q_i)$ is the secret key and $P_i = (n_i, y_i, r_i)$ is the public key;*
   (b) *Write P_i on the public board.*

All players run Protocol 32 in order to shuffle the deck of cards.

Protocol 32 (Card shuffling)

1. *$\mathcal{P}_1$ uses Procedure 20 and obtains the deck R of face-up cards;*
2. *$\mathcal{P}_1$ writes R on the public board;*
3. *The rest of players verify the deck R of face-up cards;*
4. *$\mathcal{P}_1$ uses Procedure 21 and obtains the deck of face-down cards, C_0, and the random values X that have been used;*
5. *$\mathcal{P}_1$ writes C_0 and X on the public board;*
6. *Each of the remaining players, $\{\mathcal{P}_2, \ldots, \mathcal{P}_n\}$, uses Procedure 22 with R, X, and C_0 in order to verify whether C_0 is well computed;*
7. *For $i = 1$ to n do:*
   (a) *$\mathcal{P}_i$ chooses at random the set $S_i = \{s_{1,0}, \ldots, s_{n,51}\}$, where $(\sum_{j=1}^{n} s_{j,k}) \bmod 52 = 0, \forall k \in \{0, \ldots, 51\}$;*
   (b) *$\mathcal{P}_i$ gets a permutation π_i of 52 elements;*
   (c) *$\mathcal{P}_i$ uses Procedure 23 with C_{i-1}, π_i and S_i, and obtains C_i and X_i;*
   (d) *$\mathcal{P}_i$ writes C_i in the public board;*

(e) *$\mathcal{P}_i$ and the rest of players run Protocol 33 with C_{i-1}, C_i, π_i, X_i and S_i. She proves in zero-knowledge that C_i has been computed properly.*

A face-up card is defined as a vector of n elements. Let us assume that we have a card with value w, where $0 \leq w \leq 51$. This card is represented by the vector $\{s_{w,1}, \ldots, s_{w,n}\}$, where $(\sum_{j=1}^{n} s_{w,j}) \bmod 52 = w$. A player uses Procedure 20 in order to compute a deck of face-up cards.

Procedure 20

1. *For $k = 0$ to 51 do:*
   (a) *For $j = 1$ to n do:*
      i. *Randomly choose $r_{j,k}$;*
   (b) *Verify $k \equiv r_{1,k} + \cdots + r_{n,k} \bmod 52$;*
   (c) *Form the set $r_k = \{r_{1,k}, \ldots, r_{n,k}\}$;*
2. *Form the set $R = \{r_0, \ldots, r_{51}\}$;*
3. *Return R;*

The cards must be face down if we want to shuffle them. Any player can run Procedure 21 in order to turn the deck of cards face down.

Procedure 21 (R)

1. *For $k = 0$ to 51 do:*
   (a) *For $j = 1$ to n do:*
      i. *Compute $c_{j,k} = E_{P_j}(r_{j,k}, x_{j,k}) = y^{r_{j,k}} x_{j,k}^{r}$, where P_j is the $\mathcal{P}_j$'s public key, $r_{i,k} \in R$, and $x_{j,k}$ is a random value in [KKOT90] and $x_{j,k} \equiv 1$ in [KKO97];*
   (b) *Form the set $c_k = \{c_{1,k}, \ldots, c_{n,k}\}$;*
   (c) *Form the set $x_k = \{x_{1,k}, \ldots, x_{n,k}\}$;*

2. *Form the set* $C = \{c_0, \ldots, c_{51}\}$*;*

3. *Form the set* $X = \{x_0, \ldots, x_{51}\}$*;*

4. *Return the sets* C *and* X*;*

A face-down deck of cards C can be verified using Procedure 22. The elements in R are encrypted with X and the resulting set must be C.

Procedure 22 (R, X, C)

1. *For each* $c_{j,k}$ *in* $C = \{c_{1,0}, \ldots, c_{n,51}\}$ *do:*

   (a) *Compute* $c'_{j,k} = E_{P_j}(r_{j,k}, x_{j,k}) = y^{r_{j,k}} x^r_{j,k}$*, where* $r_{j,k} \in R$*,* $x_{j,k} \in X$ *and* P_j *is the public key of* $\mathcal{P}_j$*;*

   (b) *Verify* $c_{j,k} \equiv c'_{j,k}$.

The deck of cards is shuffled by all players. Each player does the following two operations. The first operation is re-encryption or re-masking. An observer should not know the relation between non-shuffled and shuffled cards. The second operation is permuting the order of the cards. A player can do these two operations using Procedure 23.

Procedure 23 (π, C, S)

1. *For each* $s_{j,k}$ *in* $S = \{s_{1,0}, \ldots, s_{n,51}\}$*, where* $k \in \{0, \ldots, 51\}$ *and* $j \in \{1, \ldots, n\}$ *do:*

   (a) *Choose at random a value* $x_{j,k}$*;*

   (b) *Compute* $c'_{j,k} = c_{j,\pi(k)} E_{P_j}(s_{j,k}, x_{j,k}) = c_{j,\pi(k)}(y^{s_{j,k}} x^r_{j,k})$*;*

2. *Form the set* $C' = \{c'_{1,0}, \ldots, c'_{n,51}\}$*;*

3. *Form the set* $X = \{x_{1,0}, \ldots, x_{n,51}\}$*;*

4. *Return the sets* C' *and* X.

Players can see the deck of cards before and after shuffling. They must make sure that the deck of cards has been shuffled properly. Thus, whoever shuffles the deck must prove the correctness of shuffling. This proof is done in zero-knowledge using Protocol 33.

Let us assume that $\mathcal{P}_i$ is the prover, and the rest of players are the verifier.

Protocol 33 (ZKIP for shuffle (C,C',π,X,S))

1. *All players agree on a security parameter s;*
2. *For $l = 1$ to s do:*
   (a) *$\mathcal{P}_i$ chooses at random the set $S_l = \{s_{l,1,0}, \ldots, s_{l,n,51}\}$, where $(\sum_{j=1}^{n} s_{l,j,k}) \bmod 52 = 0, \forall k \in \{0, \ldots, 51\}$;*
   (b) *$\mathcal{P}_i$ gets a permutation π_l of 52 elements;*
   (c) *$\mathcal{P}_i$ uses Procedure 23 with C, π_l and S_l, and obtains C_l and X_l;*
   (d) *$\mathcal{P}_i$ writes C_l on the public board;*
   (e) *The rest of players write a random bit e on the public board;*
   (f) *If $e \equiv 0$ do:*
      i. *$\mathcal{P}_i$ writes S_l, π_l and X_l on the public board;*
      ii. *The rest of players use Procedure 24 With C, C_l, S_l, X_l and π_l in order to verify whether C_l has been computed properly;*
   (g) *If $e \equiv 1$ do:*
      i. *$\mathcal{P}_i$ computes $\pi'_l = \pi_i \circ \pi_l^{-1}$;*
      ii. *$\mathcal{P}_i$ uses Procedure 25 with X, S π, X_l and S_l; she obtains the sets S'_l and X'_l;*
      iii. *$\mathcal{P}_i$ writes π'_l, S'_l and X'_l on the public board;*
      iv. *The rest of players use Procedure 24 with C, C_l, S'_l, X'_l and π'_l in order to verify whether C_l has been computed properly.*

Procedure 24 verifies whether a deck of cards has been shuffled properly.

Procedure 24 (C, C', R, X, π)

1. *For each* $c_{j,k}$ *in* $C = \{c_{1,0}, \ldots, c_{n,51}\}$ *do:*
   (a) *Compute* $c'_{j,k} = c_{j,\pi(k)} E_{P_j}(r_{j,k}, x_{j,k}) = c_{j,\pi(k)}(y^{r_{j,k}} x^r_{j,k})$, *where* $r_{j,k} \in R$ *and* $x_{j,k} \in X$. P_j *is the public key of* $\mathcal{P}_j$*;*
   (b) *Verify* $c''_{j,k} \equiv c'_{j,k}$, *where* $c'_{j,k} \in C'$.

Procedure 25 computes the sets X' and S'. These sets are used when players verify whether the deck has been shuffled properly.

Procedure 25 (X,S,π,Z,T)

1. *For each* $x_{j,k}$ *in* $X_{x_{1,0},\ldots,x_{n,51}}$ *do:*
   (a) *Compute* $x'_{j,k} = x_{\pi(j),k}(z_{\pi(j),k})^{-1}$, *where* $x_{\pi(j),k} \in X$ *and* $z_{\pi(j),k} \in Z$*;*
   (b) *Compute* $s'_{j,k} = s_{\pi(j),k} - t_{\pi(j),k}$, *where* $s_{\pi(j),k} \in S$ *and* $t_{\pi(j),k} \in T$*;*
2. *Form the set* $X' = \{x'_{1,0}, \ldots, x'_{n,51}\}$*;*
3. *Form the set* $S' = \{s'_{1,0}, \ldots, s'_{n,51}\}$*;*
4. *Return the sets* X' *and* S'.

A player $\mathcal{P}_i$ uses Protocol 34 with the rest of players whenever $\mathcal{P}_i$ wants a new card.

Protocol 34 (Card drawing)

1. $\mathcal{P}_i$ *chooses a card* $c_l = c_{1,l}, \ldots, c_{n,l}$ *that has not been previously chosen;*
2. $\mathcal{P}_i$ *writes on the public board that she has chosen* c_i*;*
3. *Each player* $\mathcal{P}_j$ *in* $\{\mathcal{P}_1, \ldots, \mathcal{P}_{i-1}, \mathcal{P}_{i+1}, \ldots, \mathcal{P}_n\}$ *does:*
   (a) *Decrypt* $c_{j,l}$ *using Procedure 19 to obtain* $t_{j,l}$*;*
   (b) *Send* $t_{j,l}$ *to* $\mathcal{P}_i$*;*

4. *$\mathcal{P}_i$ decrypts $c_{i,l}$ using Procedure 19 to obtain $t_{i,l}$;*

5. *$\mathcal{P}_i$ computes her card $c = \sum_{j=1}^{n} t_{j,l} \bmod 52$.*

In the same paper, a second protocol suite is presented. This is a fault-tolerant version of the first one. In the first suite, a face-up card consists of n values whose sum is the card value. In the second protocol suite, the card value is divided in shares, using an (n, k) threshold scheme. There are n players and, if k players are active, they can open a card. A detailed description of this second protocol suite can be found in [KKO97].

Protocol analysis

In this section we present the cost of shuffling a deck of cards (Protocol 32) and drawing a card (Protocol 34) using the proposal [KKOT90].

Table 3.48 shows the cost in number of messages, the total length of the messages sent, and the computational cost for shuffling a deck of cards.

Table 3.49 shows the cost in number of messages, the total length of the messages sent, and the computational cost for drawing a card.

Each player $\mathcal{P}_i$ proves that she has correctly shuffled the deck of cards using Procedure 33. The cost of Procedure 33 is given in Table 3.50. We assume that Steps 2f and 2g have the same probability 1/2; this is why their costs are multiplied by 1/2

Table 3.51 shows the cost of creating a key-pair using Procedure 3.51.

Table 3.52 shows the computational cost of encrypting a value using the cryptosystem presented in [KKOT90].

In Procedure 19 r is 52, because there are 52 cards. We have assumed that the average number of iterations of Step 1 of Procedure 19 is 52/2 (see Table 3.53).

It can be seen in Table 3.54 that the cost of Procedure 20 is negligible.

Table 3.55 shows the costs of Procedure 3.55.

The cost of Procedure 22 is shown in Table 3.56.

The computational cost of Procedure 23 is presented in Table 3.57.

Table 3.48: Costs of Kurosawa-Katayama-Ogata-Tsujii's shuffling protocol

	Number of messages	Total length of messages	Computational cost
Card shuffling	$(n-1)n(1+s3)+$ $+(n-1)2$	$(n-1)n^2 52[p](1+s(n-1))+$ $+(n-1)n104[p]+(n-1)ns+$ $+(n-1)n52[r](1+s(n+1))$	$104n^2\xi(2+sn)+$ $+156n^2\rho+$ $+26n^2\rho s(4n+1)$
Step 1			ϵ
Step 2	$(n-1)$	$(n-1)n52[r]$	ϵ
Step 3			ϵ
Step 4			$52n(2\xi+\rho)$
Step 5	$(n-1)$	$(n-1)n104[p]$	ϵ
Step 6			$(n-1)52n(2\xi+\rho)$
Step 7	$n(n-1)(3s+1)$	$(n-1)52[p]n^2(1+s(n-1))+$ $+n(n-1)(s)+$ $+n(n-1)(s52[r](n+1))$	$104n^2\xi(1+sn)+$ $+26n^2\rho 104+$ $+26n^2\rho s(4n+1)$
Step 7a			ϵ
Step 7b			ϵ
Step 7c			$104n(\xi+\rho)$
Step 7d	$(n-1)$	$(n-1)n52[p]$	ϵ
Step 7e	$3s(n-1)$	$s(n-1)104n[p]+$ $+s(n-1)+$ $s(n-1)52[r](n+1)$	$sn^2 156\xi+$ $+sn26\rho(4n+1)$

The computational cost of Procedure 24 is the same of Procedure 23 as we can see in Table 3.58.

Table 3.59 shows the computational cost of Procedure 25.

The proposal [KKOT90] satisfies all security requirements listed in [Cré85] (see Table 3.60). However, some authors [Ask05] claim that [KKOT90] and [KKO97] require a computation time unacceptable in practice.

Table 3.49: Costs of Kurosawa-Katayama-Ogata-Tsujii's drawing protocol

	Number of messages	Total length of messages	Computational cost
Card drawing	$2(n-1)$	$(n-1)([r]+[p])$	$n\xi 28$
Step 1			ϵ
Step 2	$(n-1)$	$(n-1)[p]$	ϵ
Step 3	$(n-1)$	$(n-1)[r]$	$(n-1)28\xi$
Step 3a			28ξ
Step 3b	1	$[r]$	ϵ
Step 4			28ξ
Step 5			ϵ

Table 3.50: Costs of the ZKIP protocol

	Number of messages	Total length of messages	Computational cost
Protocol 33	$3s(n-1)$	$s(n-1)104n[p]+$ $+s(n-1)+$ $+s(n-1)52[r](n+1)$	$sn^2104\xi+$ $+sn26\rho(4n+1)$
Step 1			ϵ
Step 2	$3s(n-1)$	$s(n-1)104n[p]+$ $+s(n-1)+$ $+s(n-1)52[r](n+1)$	$sn^2104\xi+$ $+sn26\rho(4n+1)$
Step 2a			ϵ
Step 2b			ϵ
Step 2c			$104n(\xi+\rho)$
Step 2d	$(n-1)$	$(n-1)52n[p]$	ϵ
Step 2e	$(n-1)$	$(n-1)[1]$	ϵ
Step 2f	$\frac{1}{2}(n-1)$	$\frac{1}{2}(n-1)52(n[p]+[r](n+1))$	$\frac{1}{2}(n-1)(104n(\xi+\rho))$
Step 2(f)i	$(n-1)$	$(n-1)52(n[p]+[r](n+1))$	ϵ
Step 2(f)ii			$(n-1)104n(\xi+\rho)$
Step 2g	$\frac{1}{2}(n-1)$	$\frac{1}{2}(n-1)52(n[p]+[r](n+1))$	$\frac{1}{2}(n-1)104n(\rho+\xi)+$ $+\frac{1}{2}(n-1)n52\rho$
Step 2(g)i			ϵ
Step 2(g)ii			$n52\rho$
Step 2(g)iii	$(n-1)$	$(n-1)52(n[p]+[r](1+n))$	ϵ
Step 2(g)iv			$(n-1)104n(\xi+\rho)$

Table 3.51: Costs of Procedure 17

	Computational cost
Procedure 17	$2\xi + \rho$
Step 1	ρ
Step 2	2ξ
Step 2a	2ξ
Step 2b	ϵ
Step 2(b)i	ϵ
Step 2(b)ii	ϵ
Step 3	ϵ

Table 3.52: Costs of Procedure 18

	Computational cost
Procedure 18	$2\xi + \rho$
Step 1	ϵ
Step 2	$2\xi + \rho$
Step 3	ϵ

Table 3.53: Costs of Procedure 19

	Computational cost
Procedure 19	28ξ
Step 1	$\xi + \xi + \xi\frac{52}{2}$
Step 2	ϵ

Table 3.54: Costs of Procedure 20

	Computational cost
Procedure 20	ϵ
Step 1	ϵ
Step 1a	ϵ
Step 1(a)i	ϵ
Step 1b	ϵ
Step 1c	ϵ
Step 2	ϵ
Step 3	ϵ

Table 3.55: Costs of Procedure 21

	Computational cost
Procedure 21	$52n(2\xi + \rho)$
Step 1	$52n(2\xi + \rho)$
Step 1a	$n(2\xi + \rho)$
Step 1(a)i	$(2\xi + \rho)$
Step 1b	ϵ
Step 1c	ϵ
Step 2	ϵ
Step 3	ϵ
Step 4	ϵ

Table 3.56: Costs of Procedure 22

	Computational cost
Procedure 22	$n52(2\xi + \rho)$
Step 1	$n52(2\xi + \rho)$
Step 1a	$2\xi + \rho$
Step 1b	ϵ

Table 3.57: Costs of Procedure 23

	Computational cost
Procedure 23	$104n(\xi + \rho)$
Step 1	$104n(\xi + \rho)$
Step 1a	ϵ
Step 1b	$2(\xi + \rho)$
Step 2	ϵ
Step 3	ϵ
Step 4	ϵ

Table 3.58: Costs of Procedure 24

	Computational cost
Procedure 24	$104n(\xi + \rho)$
Step 1	$104n(\xi + \rho)$
Step 1a	$(\xi + \rho)$
Step 1b	ϵ

Table 3.59: Costs of Procedure 25

	Computational cost
Procedure 25	$n52\rho$
Step 1	$n52\rho$
Step 1a	ρ
Step 1b	ϵ
Step 2	ϵ
Step 3	ϵ
Step 4	ϵ

Table 3.60: Security properties of Kurosawa-Katayama-Ogata-Tsujii's protocol suite

Uniqueness of cards	X
Uniform random distribution of cards	X
Cheating detection with a very high probability	X
Complete confidentiality of cards	X
Minimal effect of coalitions	X
Complete confidentiality of strategy	X

3.3.8 Bounded-to-unbounded poker game (Harn-Lin-Gong)

The protocol presented in [HLG00] is a modification of [SRA81] that prevents card marking. In [SRA81], if the value used to represent a card is a quadratic residue, then its encrypted value is also a quadratic residue. Thus, if the plaintext card is a quadratic residue, the encrypted card is "marked". In [HLG00], the values used to represent cards are quadratic non-residues so that the card marking attack does not work.

Assume that $\mathcal{P}_1$ constructs the deck of cards as follows.

Preparation :

1. $\mathcal{P}_1$ chooses a large prime number p such that $p = 2q + 1$, where q is a large prime too.
2. $\mathcal{P}_1$ chooses a primitive element α of $\mathbb{Z}_p^*$.
3. $\mathcal{P}_1$ chooses at random a set of 52 odd numbers, $S = \{s_1, \ldots, s_{52}\}$, where $s_i \neq q$ and $1 < s_i < p - 1$.
4. $\mathcal{P}_1$ computes the values that will represent the deck of cards as $T = \{m_1, \ldots, m_{52}\}$, where $m_i = (\alpha)^{s_i}$.

The values m_i are such that $m_i \in \overline{Q_p}$ and the attack presented in [Lip81] is not possible. The authors of [HLG00] introduce no further modifications on [SRA81], so the protocols for shuffling the deck and drawing a card are the same given in [SRA81]. At the end of the game, both parties reveal their secret keys to prove that they did not cheat.

Protocol analysis

The shuffling and drawing protocols have the same computational cost as in [SRA81]. Moreover, the protocol [HLG00] is also limited to two players whose strategy must be revealed when the hand finishes.

Table 3.61 lists the security properties of [HLG00].

Table 3.61: Security properties of Harn-Lin-Gong's protocol suite

Uniqueness of cards	X
Uniform random distribution of cards	X
Cheating detection with a very high probability	X
Complete confidentiality of cards	X
Minimal effect of coalitions	X
Complete confidentiality of strategy	

3.3.9 A secure mental poker protocol over the Internet (Zhao-Varadharajan-Mu)

In [ZVM03], a TTP-free mental poker protocol is presented whereby more than two players can participate in the game. The protocol is similar to [SRA81] but uses the ElGamal cryptosystem [ElG85] instead. All players use the same large prime number p so that ElGamal becomes a commutative cryptosystem. To show how it works, we can put an example in which x is first encrypted by player $\mathcal{P}_1$ and later by player $\mathcal{P}_2$; next, the cryptogram is decrypted by $\mathcal{P}_1$ and later by $\mathcal{P}_2$, who obtains x again.

1. $\mathcal{P}_1$ randomly chooses a factor r_1 and encrypts x with her private key K_1

$$E_{K_1}(x) = (y_{1,1}, y_{2,1}) \begin{cases} y_{1,1} = \alpha_1^{r_1} \mod p \\ y_{2,1} = x\beta_1^{r_1} \mod p \end{cases}$$

2. $\mathcal{P}_2$ in a similar way chooses at random a factor r_2 and encrypts $y_{2,1}$ with her private key K_2:

$$E_{K_2}(y_{2,1}) = (y_{1,2}, y_{2,1,2}) \begin{cases} y_{1,2} = \alpha_2^{r_2} \mod p \\ y_{2,1,2} = x\beta_2^{r_2}\,\beta_2^{r_2} \mod p \end{cases}$$

3. $\mathcal{P}_1$ decrypts $y_{2,1,2}$:

$$\begin{aligned} D_{K_1}(y_{2,1,2}) &= y_{2,1,2}(y_{1,1}^{k_1})^{-1} \bmod p \\ &= x\beta_1^{r_1}\,\beta_2^{r_2}\,(\beta_1^{r_1})^{-1} \bmod p \\ &= y_{2,2} \end{aligned}$$

4. Later $\mathcal{P}_2$ decrypts $y_{2,2}$ and obtains x:

$$D_{K_2}(y_{2,2}) = y_{2,2}(y_{1,2}^{k_2})^{-1} \bmod p = x$$

Players choose a set of 52 values to represent a deck of cards. Every card is encrypted by every player in turn, and a deck of encrypted cards is obtained. When a

player wants a card, the rest of players decrypt that card. The player who requested the card receives it encrypted with her key, and she secretly decrypts the card and obtains the card in the clear. Next, we describe in detail the various protocols of proposal [ZVM03], assuming there are n players, $\{\mathcal{P}_1, \ldots, \mathcal{P}_n\}$.

Protocol 35 (Initialization)

1. *Players choose a large prime number p and all subsequent operations are done over Z_p;*

2. *Each player $\mathcal{P}_i$ computes her secret key pair K_i, where $K_i = \{(p, \alpha_i, k_i, \beta_i) : \beta_i \equiv \alpha_i^{k_i} \, mod \, p\}$;*

3. *The deck of cards is represented by 52 values, $D = \{d_1, \ldots, d_{52}\}$, which are agreed by all players.*

In Protocol 36 every player encrypts x_i chosen values with her public key and obtains a set of cryptograms, which are sent to other players. Next, every player encrypts the sets received from other players with her public key. The players finally obtain as many sets with the same cryptograms as there are players. If one or more players are not fair, then the sets are different.

Protocol 36 (Card shuffling (D))

1. *For each $\mathcal{P}_i$ in $\{\mathcal{P}_1, \ldots, \mathcal{P}_n\}$ do;*

   (a) *Use Procedure 26 with D, α_i and β_i and obtain $E_{0,i}$ and r_i;*

   (b) *Publish $E_{0,i}$;*

2. *All players form the set $E_0 = \{E_{0,1}, \ldots, E_{0,n}\}$;*

3. *For each $\mathcal{P}_i$ in $\{\mathcal{P}_1, \ldots, \mathcal{P}_n\}$ do;*

   (a) *Receive the set $E_{i-1} = \{E_{i-1,1}, \ldots, E_{i-1,n}\}$ form $\mathcal{P}_{i-1}$;*

   (b) *Compute $\beta_i^{r_i}$ and $\alpha_i^{r_i}$;*

(c) *For each* $E_{i-1,j} = \{e_{i-1,j,1}, \ldots, e_{i-1,j,52}\}$ *in* $\{E_{i-1,1}, \ldots, E_{i-1,n}\} - \{E_{i-1,i}\}$ *do:*

i. *For each* $e_{i-1,j,k} = (y_{i-1,j,k,1}, y_{i-1,j,k,2})$ *in* $E_{i-1,j}$ *do:*

A. *Compute* $y_{i,j,k,2} = y_{i-1,j,k,2}\beta_i^{r_i}$;

B. *Compute* $y_{i,j,k,1} = y_{i-1,j,k,1} \cup \{\alpha_i^{r_i}\}$;

C. *Compute* $e'_{i,j,k} = (y_{i,j,k,1}, y_{i,j,k,2})$;

ii. *Compute the set* $E'_{i,j} = \{e'_{i,j,1}, \ldots, e'_{i,j,52}\}$;

iii. *Choose a permutation* π_i *of 52 values;*

iv. *Permute the order of cryptograms in* $E'_{i,j}$ *to obtain* $E_{i,j} = \{e_{i,j,1}, \ldots, e_{i,j,52}\}$, *where* $e_{i,j,\pi_i(k)} = e'_{i,j,k}$;

(d) *Compute the set* $E_i = \{E_{i,1}, \ldots, E_{i,n}\}$;

(e) *Publish* E_i;

4. *All players verify that* $E_{n,i} \equiv E_{n,j}$ $\forall i, j \in \{1, \ldots, n\}$.

Each player uses Procedure 26 in order to encrypt the deck of cards D. The procedure returns and encrypted and shuffled deck of cards E' and the value r that has been used to encrypt the deck.

Procedure 26 (D, α, β)

1. *Choose one value* r, *where* $1 < r < p - 1$;

2. *For each* $d_i \in D$ *do:*

(a) *Encrypt* d_i *using* r, *where* $e_i = (y_{i,1}, y_{i,2}) = (\alpha^r, d_i\beta^r)$;

3. *Compute the set of encrypted cards* $E = \{e_1, \ldots, e_{52}\}$;

4. *Choose a permutation* π *of 52 values;*

5. *Permute the order of cryptograms in* E *to obtain* $E' = \{e'_1, \ldots, e'_{52}\}$, *where* $e'_{\pi(i)} = e_i$;

6. *Return* E' *and* r;

When a player wants a card she chooses a cryptogram in $E_{n,i}$, where $i \in \{1, \ldots, n\}$, and marks that cryptogram as chosen. Players can only take a cryptogram that is not marked. The rest of players in turn decrypt the cryptogram chosen by the player. The last player sends the cryptogram to the player that requested the card. At this point, the card value is protected only by the key of the player drawing the card.

Let us suppose that $\mathcal{P}_k$ wants a card and uses Protocol 37.

Protocol 37 (Card drawing)

1. *$\mathcal{P}_k$ chooses a cryptogram $e_{n,i,l} \in E_{n,i}$ and marks it as chosen;*
2. *$\mathcal{P}_k$ sends $c_0 = e_{n,i,l}$ to the rest of players;*
3. *For each player $\mathcal{P}_j \in \{\mathcal{P}_1, \ldots, \mathcal{P}_{k-1}, \mathcal{P}_{k+1}, \ldots, \mathcal{P}_n\}$ do:*
   (a) *Receive $c_{j-1} = (y_{j-1,1}, y_{j-1,2})$;*
   (b) *Compute $c_j = (y_{j,1}, y_{j,2})$ where $y_{j,1} = y_{j-1,1} = \{y_{j-1,1,1}, \ldots, y_{j-1,1,n}\}$ and $y_{j,2} = y_{j-1,2} \cdot ((y_{j,1,i})^{k_i})^{-1} = y_{j-1,2} \cdot (((\alpha_i)^{r_i})^{k_i})^{-1}$;*
   (c) *Send c_j to the next player;*
4. *$\mathcal{P}_k$ receives $c_n = (y_{n,1}, y_{n,2})$;*
5. *$\mathcal{P}_k$ decrypts c_n and obtains her card $d = y_{n,2} \cdot ((y_{n,1,k})^{k_k})^{-1}$*

When the game is over, players reveal their secret random numbers $\{r_1, \ldots, r_n\}$. Every player can check whether the rest of players have been honest. This game verification reveals the strategy of players. The authors suggest to use a TTP, named Dealer, in order to keep the strategy of players confidential. At the end of the game the Dealer receives the secret random numbers $\{r_1, \ldots, r_n\}$ and checks the fairness of the game.

Protocol analysis

In this section we present the cost of the shuffling (Protocol 36) and drawing protocols (Protocol 37) of the proposal [ZVM03].

Table 3.62: Costs of Zhao-Varadharajan-Mu's shuffling protocol

	Number of messages	Total length of messages	Computational cost
Card shuffling	$2n(n-1)$	$(n-1)n(n+1)104[p]$	$4n\xi + 52n^2\rho$
Step 1	$n(n-1)$	$(n-1)104[p]n$	$n(52\rho + 2\xi)$
Step 1a			$52\rho + 2\xi$
Step 1b	$(n-1)$	$2[p]52(n-1)$	ϵ
Step 2			ϵ
Step 3	$n(n-1)$	$n(n-1)n(104[p])$	$n(2\xi + (n-1)52\rho)$
Step 3a			ϵ
Step 3b			2ξ
Step 3c			$(n-1)52\rho$
Step 3(c)i			52ρ
Step 3(c)iA			ρ
Step 3(c)iB			ϵ
Step 3(c)iC			ϵ
Step 3(c)ii			ϵ
Step 3(c)iii			ϵ
Step 3(c)iv			ϵ
Step 3d			ϵ
Step 3e	$(n-1)$	$(n-1)n104[p]$	ϵ
Step 4			ϵ

Table 3.62 shows the cost of Protocol 36.

The cost of the drawing protocol (Protocol 37) is presented in Table 3.63.

Table 3.64 shows the cost of Procedure26.

Unless a TTP is used, the strategy of players is made public when the game is over and players verify the game. But, more important, this protocol has a security flaw: any player can decrypt the cryptograms without knowing any private keys [CD04]. See Chapter 4 of this Thesis for more details. With this security flaw the protocol is not usable in practice (see Table 3.65).

Table 3.63: Costs of Zhao-Varadharajan-Mu's drawing protocol

	Number of messages	Total length of messages	Computational cost
Card drawing	$2(n-1)$	$4(n-1)[p]$	$n(\rho+\xi)$
Step 1			ϵ
Step 2	$(n-1)$	$(n-1)2[p]$	ϵ
Step 3	$(n-1)$	$2[p](n-1)$	$(n-1)(\rho+\xi)$
Step 3a			ϵ
Step 3b			$\rho+\xi$
Step 3c	1	$2[p]$	ϵ
Step 4			ϵ
Step 5			$\rho+\xi$

Table 3.64: Costs of Zhao-Varadharajan-Mu's Procedure 26

	Computational cost
Procedure 26	$52\rho+2\xi$
Step 1	ϵ
Step 2	$52\rho+2\xi$
Step 2a	ρ
Step 3	ϵ
Step 4	ϵ
Step 5	ϵ
Step 6	ϵ

Table 3.65: Security properties of the Zhao-Varadharajan-Mu protocol suite

Uniqueness of cards	X
Uniform random distribution of cards	X
Cheating detection with a very high probability	
Complete confidentiality of cards	
Minimal effect of coalitions	
Complete confidentiality of strategy	

3.3.10 Mental poker revisited (Barnett-Smart)

Last but not least, the Barnett-Smart [BS03] protocol suite is an interesting one. On the positive side, this proposal seems to meet all requirements in [Cré85] and relies on efficient zero-knowledge techniques to prove that the deck has been properly shuffled. On the negative side, the paper [BS03] is not self-contained and fails to precisely describe the proposed protocol in a way that can be readily implemented. Several parts of the protocol are left implicit or undescribed (references to the literature are given). Also, no computing time figures are offered and no security proofs are provided.

The protocol can be implemented using the ElGamal [ElG85] encryption, or the Paillier probabilistic encryption function [Pai99]. We expose and analyze the ElGamal version of the protocol suite.

If there are n players, $\{\mathcal{P}_1, \ldots, \mathcal{P}_n\}$, they use Protocol 38 to set up the system.

Protocol 38 (Preparation)

1. *All players agree on a prime number p, where $p = 2q+1$ and q is an odd prime number. The computations are done over Z_p;*
2. *All players agree on a generator $g \in Z_q$;*
3. *Each player $\mathcal{P}_i$ generates a random private key x_i;*
4. *Each player $\mathcal{P}_i$ publishes $h_i = g^{x_i}$;*
5. *All players compute the public key $h = \prod_{i=1}^{n}(h_i)$;*
6. *All players agree on a security parameter s.*

Players then use Protocol 39 to shuffle the deck of cards. In Step 1 of Protocol 39 all players agree on the values to be used to represent the deck of cards. These values are the deck of cards face up. The value of each card is public. In Step 2 all players compute the face-down deck of cards. Cards are encrypted with the public key h using

1 instead of a random factor in ElGamal encryption. Next, all players permute and re-mask the face-down deck of cards as described in [SSG02] using ElGamal re-masking (see Section 2.2.4 in Chapter 2 for details). Finally in Step 3d each player proves in zero-knowledge that she has re-masked and permuted the deck of cards properly. The zero-knowledge proof used is based on the zero-knowledge proof presented in [BCR86].

Protocol 39 (Card shuffling)

1. *All players agree on* 52 *values* $D = \{d_1, \ldots, d_{52}\}$ *to represent the deck of cards;*
2. *All players use Procedure 27 with* D*, and obtain* C_0*;*
3. *For* $i = 1$ *to* n $\mathcal{P}_i$ *does:*
   (a) $\mathcal{P}_i$ *obtains a permutation* π_i *of* 52 *elements;*
   (b) $\mathcal{P}_i$ *uses Procedure 28 with* C_{i-1} *and* π_i*, and obtains* C_i *and* R_i*;*
   (c) $\mathcal{P}_i$ *publishes* C_i*;*
   (d) $\mathcal{P}_i$ *uses Protocol 40 with* C_i*,* R_i *and* π_i*.*

The set C_n is the encrypted and shuffled deck of cards. Procedure 27 computes the face-down deck of cards from a face-up deck of cards.

Procedure 27 ($D = \{d_1, \ldots, d_{52}$)

1. *For* $i = 1$ *to* 52 *do:*
   (a) *compute* $c_i = (c_{i,1}, c_{i,2})$*, where* $c_{i,1} = g$ *and* $c_{i,2} = d_i h$*;*
2. *Form* $C = \{c_1, \ldots, c_{52}\}$*;*
3. *return* C*.*

Procedure 28 computes a re-masked and permuted deck of cards from a face-down deck of cards.

Procedure 28 ($C = \{c_1, \ldots, c_{52}\}, \pi$)

1. *For* $i = 1$ *to* 52 *do:*

   (a) *obtain a value* r_i *at random, where* $1 < r_i < q$*;*

   (b) *compute* $c'_i = (c'_{i,1}, c'_{i,2})$*, where* $c'_{i,1} = c_{i,1}g^{r_i}$ *and* $c'_{i,2} = c_{i,2}h^{r_i}$*;*

2. *Form* $C' = \{c'_{\pi(1)}, \ldots, c'_{\pi(52)}\}$*;*

3. *Form* $R = \{r_1, \ldots, r_{52}\}$*.*

4. *Return* C' *and* R*;*

All players use Protocol 40 to verify that other players permute and re-mask the deck of cards properly.

Protocol 40 (C, R, π)

1. *For* $j = 1$ *to* s *do:*

   (a) $\mathcal{P}_i$ *obtains a permutation* π_j *of* 52 *values at random;*

   (b) $\mathcal{P}_i$ *uses Procedure 28 with* C *and* π_j*, and obtains* C_j *and* R_j*;*

   (c) $\mathcal{P}_i$ *publishes* C_j*;*

2. *All players choose a random subset* $S \subset \{C_1, \ldots, C_s\}$ *and send* S *to* $\mathcal{P}_i$*;*

3. *For* $j = 1$ *to* s *do:*

   (a) *if* $C_j \in S$ *do:*

      i. $\mathcal{P}_i$ *publishes* π_j *and* R_j*;*

      ii. *all players use the Procedure 29 to verify* C_j*;*

   (b) *If* $C_j \notin S$ *do:*

      i. $\mathcal{P}_i$ *computes* $\pi'_j = \pi \circ \pi_j$*;*

      ii. $\mathcal{P}_i$ *computes* $R'_j = \{r'_{j,1}, \ldots, r'_{j,52}\}$*, where* $r'_{j,k} = r_{j,k} + r_{\pi(j)}$*, and* $r_{\pi(j)} \in R$*;*

      iii. $\mathcal{P}_i$ *publishes* π'_j *and* R'_j*;*

iv. all players use the Procedure 29 to verify C_j;

Procedure 29 verifies that a deck of cards C has been properly permuted and encrypted in C'.

Procedure 29 (C, C', R, π)

1. *For $i = 1$ to 52 do:*

   (a) *compute $c''_i = (c''_{i,1}, c''_{i,2})$, where $c''_{i,1} = c_{i,1}g^{r_i}$ and $c'_{i,2} = c_{i-1,2}h^{r_i}$, where $c_{i,1}, c_{i,2} \in C$ and $r_i \in R$;*

   (b) *verify $c''_{i,1} \stackrel{?}{=} c'_{i,1}$ and $c''_{i,2} \stackrel{?}{=} c'_{i,2}$, where $c'_{i,1}, c'_{i,2} \in C'$.*

All players run Protocol 41 whenever a $\mathcal{P}_i$ wants a new card. We see in Step 1 as $\mathcal{P}_i$ chooses a card that has not been selected previously. Next, all players decrypt this card and they prove in zero-knowledge using the Chaum-Pedersen proof [CP92] that they have properly decrypted the card.

Protocol 41 $(C = \{c_1, \ldots, c_{52}\})$

1. *$\mathcal{P}_i$ chooses a cryptogram $c_t = \{c_{t,1}, c_{t,2}\}$, where $1 \leq t \leq 52$ and t has not been previously selected;*

2. *$\mathcal{P}_i$ publishes c_t;*

3. *Each $\mathcal{P}_j$ in $\{\mathcal{P}_1, \ldots, \mathcal{P}_{i-1}, \mathcal{P}_{i+1}, \ldots, \mathcal{P}_n\}$ does:*

   (a) *Compute $e_j = c_{t,1}^{x_j}$;*

   (b) *Publish d_j;*

   (c) *Prove in zero-knowledge that $\log_g(h_i) = \log_{c_{t,1}}(e_j)$ using the Chaum-Pedersen [CP92] zero-knowledge proof (see Section 2.2.2 in Chapter 2):*

4. *$\mathcal{P}_i$ computes $e_i = c_{t,1}^{x_i}$;*

5. *$\mathcal{P}_i$ computes her card $d = c_{t,2}\left(\prod_{k=1}^{n}(e_k)\right)^{-1}$*

Protocol analysis

In this section we analyze the number of messages, the total length of messages, the computational cost and the security of the proposal [BS03].

Table 3.66 shows the cost of the shuffling protocol (Protocol 39).

Table 3.66: Costs of Barnett-Smart's shuffling protocol

	Number of messages	Total length of messages	Computational cost
Card shuffling	$(n-1)2n(s+1)+$ $+(n-1)$	$n(n-1)[p]104(1+2s)+$ $+[r](n-1)n(1+s52)+$ $+[r](n-1)52$	$n104(\rho+\xi)(ns+1)+$ $+n52\rho$
Step 1	$(n-1)$	$(n-1)52[r]$	ϵ
Step 2			$n(52\rho)$
Step 3	$n(2(n-1)(s+1))$	$n(n-1)[p]104(1+2s)+$ $+n(n-1)[r](1+s52)$	$n(104(\rho+\xi)(ns+1))$
Step 3a			ϵ
Step 3b			$52(2\xi+2\rho)$
Step 3c	$(n-1)$	$(n-1)104[p]$	ϵ
Step 3d	$(n-1)(2s+1)$	$(n-1)[p]208s+$ $+(n-1)[r](1+s52)$	$ns104(\rho+\xi)$

Table 3.67 shows the cost of the drawing protocol (Protocol 41). In Step 3c we assume that Chaum-Pedersen's zero-knowledge proof has the following costs:

- Number of messages: $3(n-1)$;
- Total length of messages: $(n-1)4[p]$;
- Computational cost: $(2\xi+\rho)(2n-1)$

See Section 2.2.2 in Chapter 2 for more details.

Table 3.68 shows the cost of Protocol 40.

Table 3.69 shows the cost of Procedure 27.

Table 3.70 shows the cost of Procedure 28.

Table 3.71 shows the costs of Procedure 29.

Table 3.67: Costs of Barnett-Smart's drawing protocol

	Number of messages	Total length of messages	Computational cost
Card drawing	$(n-1)(4n-3)$	$(n-1)(6n-4)[p]$	$\xi(4n^2-5n+2)+\rho(2n^2-2n+1)$
Step 1			ϵ
Step 2	$(n-1)$	$(n-1)[p]2$	ϵ
Step 3	$(n-1)4(n-1)$	$(n-1)(n-1)6[p]$	$(n-1)((4n-1)\xi+(2n-1)\rho)$
Step 3a			ξ
Step 3b	$(n-1)$	$(n-1)2[p]$	ϵ
Step 3c	$3(n-1)$	$(n-1)4[p]$	$(2\xi+\rho)(2n-1)$
Step 4			ξ
Step 5			$n\rho$

The security properties of the proposal [BS03] are summarized in Table 3.72. In conclusion, this proposal is significant because it offers a complete solution to the mental poker problem and has a moderate computational cost.

Table 3.68: Costs of Protocol 40

	Number of messages	Total length of messages	Computational cost
Protocol 40	$(n-1)(2s+1)$	$(n-1)([p]208s+[r](1+s52))$	$ns104(\rho+\xi)$
Step 1	$s(n-1)$	$s(n-1)104[p]$	$s(104)(\xi+\rho)$
Step 1a			ϵ
Step 1b			$52(2\xi+2\rho)$
Step 1c	$(n-1)$	$(n-1)104[p]$	ϵ
Step 2	$(n-1)$	$(n-1)[r]$	ϵ
Step 3			$104s(\xi+\rho)(n-1)$
Step 3a	$(1/2)(n-1)$	$(1/2)(n-1)(104[p]+52[r])$	$(1/2)(52(2\xi+2\rho))(n-1)$
Step 3(a)i	$(n-1)$	$(n-1)(104[p]+52[r])$	ϵ
Step 3(a)ii			$52(2\xi+2\rho)(n-1)$
Step 3b	$(1/2)(n-1)$	$(1/2)(n-1)(104[p]+52[r])$	$(1/2)(2\xi+2\rho)(n-1)$
Step 3(b)i			ϵ
Step 3(b)ii			ϵ
Step 3(b)iii	$(n-1)$	$(n-1)(104[p]+52[r])$	ϵ
Step 3(b)iv			$52(2\xi+2\rho)(n-1)$

Table 3.69: Costs of Procedure 27

	Computational cost
Procedure 27	52ρ
Step 1	52ρ
Step 1a	ρ
Step 2	ϵ
Step 3	ϵ

Table 3.70: Costs of Procedure 28

	Computational cost
Procedure 28	$52(2\xi + 2\rho)$
Step 1	$52(2\xi + 2\rho)$
Step 1a	ϵ
Step 1b	$(2\xi + 2\rho)$
Step 2	ϵ
Step 3	ϵ
Step 4	ϵ

Table 3.71: Costs of the Procedure 29

	Computational cost
Procedure 29	$104(xi + \rho)$
Step 1	$52(2\xi + 2\rho)$
Step 1a	$(2\xi + 2\rho)$
Step 1b	ϵ

Table 3.72: Security properties of the Barnett-Smart protocol suite

Uniqueness of cards	X
Uniform random distribution of cards	X
Cheating detection with a very high probability	X
Complete confidentiality of cards	X
Minimal effect of coalitions	X
Complete confidentiality of strategy	X

3.3.11 Conclusions on the comparison of TTP-free protocols

In this section we draw some conclusions about the computational cost, the number of messages and the length of the messages for the various TTP-free proposals examined.

Table 3.73 shows the security properties of the TTP-free mental poker protocols described in this chapter. We group the protocol suites based on their security properties. We cannot compare two protocols with different security properties. On one hand, [Cré86], [KKOT90] and [BS03] are complete solutions to Crépeau's requirements [Cré85]. On the other hand, [Yun85] and [BF83] have similar security properties, in that they reveal the strategy of players and do not withstand player coalitions. A third group is formed by proposals [GM82] and [HLG00]. In the fourth place, [Cré85] has the same security properties as [HLG00] and [GM82], but it cannot be added to the third group because [GM82] and [HLG00] are two-player only whereas [Cré85] has no limitation in the number of players. Finally, the proposals [SRA81], [Cré85] and [ZVM03] cannot be compared with any other proposal, because each one has different security properties.

The proposals [SRA81], [GM82] and [HLG00] are two players only, and are analyzed in the following sections Section 3.3.1, Section 3.3.2 and Section 3.3.8 respectively. Looking at the corresponding result tables it can be seen that the cost has been split among the two players. Nevertheless in the rest of protocols the cost of all participants in the game has been grouped. In the tables of this section we present the addition of the cost of all players for each protocol. So, in the protocols for two players only we have added the cost of the two players.

The notation used in the tables has been defined in Section 3.1. We just recall the meaning of symbol ζ in Yung's contribution [Yun85]: ζ is the number of cards that are not owned by a player. It is difficult to estimate the value of ζ (we only know that $1 \leq \zeta \leq 52$).

In Table 3.74 we show the computational cost of TTP-free mental poker protocols. We maintain the protocol grouping based on the security properties.

We first compare [Cré86], [KKOT90] and [BS03]:

Table 3.73: Security properties of TTP-free mental poker protocols

Scheme	Uniqueness of cards	Uniform random distribution of cards	Cheating detection with a very high probability	Complete confiden--tiality of cards	Minimal effect of coalitions	Complete confiden--tiality of strategy	Number of players
[SRA81]	X	X			X		2
[GM82]	X	X	X	X	X		2
[BF83]	X	X	X	X			$3 \leq n$
[Yun85]	X	X	X	X			$2 \leq n$
[Cré85]	X	X	X	X	X		$2 \leq n$
[Cré86]	X	X	X	X	X	X	$2 \leq n$
[KKOT90]	X	X	X	X	X	X	$2 \leq n$
[HLG00]	X	X	X	X	X		2
[ZVM03]	X	X					$2 \leq n$
[BS03]	X	X	X	X	X	X	$2 \leq n$

- In the shuffling protocol, the number of products is very high in [Cré86], but the number of exponentiations is very low. The cost in time of a product is very low in comparison with the cost of a modular exponentiation. When comparing the shuffling cost of [Cré86], [KKOT90] and [BS03] we see that [Cré86] is the most efficient. This seems contradictory with the fact that in [Edw94] an implementation of the protocol [Cré86] on three Sparc workstations is reported to have taken eight hours to shuffle a deck. We must assume that this poor performance was probably influenced by the cost of communication. [BS03] is more efficient than [KKOT90], because the number of modular exponentiations is lower in [BS03].

- In the drawing protocol, [Cré86] is the most efficient followed by [BS03] and [KKOT90].

We compare [Yun85] and [BF83].

- In the shuffling protocol [BF83] does not use modular exponentiations nor products. There is no doubt that [BF83] is more efficient that [Yun85]. However we

must remark that [Yun85] is an important contribution. The author does not mention that his proposal tolerates player drop-out.

- In the drawing protocol the comparison is similar.

We compare [GM82] and [HLG00].

- In the shuffling protocol [HLG00] is more efficient than [GM82], because the number of modular exponentiations is lower in [HLG00]. Nonetheless, the probabilistic encryption presented in [GM82] is a significant contribution to cryptography. So, we can state that [GM82] is an outstanding work.

- In the drawing protocol [HLG00] is again more efficient than [GM82].

[SRA81] has a reduced computational cost for shuffling and drawing. We can state that it opened the research in mental poker.

The computational cost in [Cré85] is moderated if we compare it with other contributions with less security properties.

In principle, [ZVM03] was supposed to have the same security properties as [Cré85], and [ZVM03] is more efficient than [Cré85]. However, [ZVM03] has a security flaw that reduces its actual security.

In Table 3.75 we show the number of messages of TTP-free mental poker protocols.

We first compare the contributions [Cré86], [KKOT90] and [BS03]:

- In the shuffling protocol, [BS03] and [Cré86] require a similar number of messages. The number of messages is a bit higher in [KKOT90].

- In the drawing protocol, the number of messages is similar in [KKOT90] and [Cré86] and a bit higher in [BS03].

We compare [Yun85] and [BF83].

- In the shuffling protocol the number of messages is higher in [Yun85] than in [BF83]. The cryptographic protocol presented in [Yun85] is complex and requires a non negligible number of communications.

Table 3.74: Computational cost of TTP-free mental poker protocols

Computational cost	Card shuffling	Card drawing	Number of players
[SRA81]	52ξ	4ξ	2
[GM82]	$156[r]\rho$	$260\rho + 2\xi([r] + 52)$	2
[BF83]	ϵ	ϵ	$3 \leq n$
[Yun85]	$(n-1)k(\rho(2k-1)(52+\zeta)+$ $+\zeta(6\rho+2\xi))$	$k\rho\zeta(2k-1)+$ $+k\rho 6 + 2k\xi$	$2 \leq n$
[Cré85]	$52n\xi$	$(\rho+\xi)(\frac{n^2-2n+5}{4})$	$2 \leq n$
[Cré86]	$n\rho[r](n-1)(299s+130)+$ $+n\rho + n\rho[r]78 + n(2\xi)$	$\frac{n-1}{2}(r(6\rho+4\xi))$	$2 \leq n$
[KKOT90]	$104n^2\xi(2+sn)+$ $+26n^2\rho(6+s(4n+1))$	$n28\xi$	$2 \leq n$
[HLG00]	52ξ	4ξ	2
[ZVM03]	$n(4\xi + 52\rho n)$	$n(\rho+\xi)$	$2 \leq n$
[BS03]	$n(104\xi(ns+1) + 52\rho(2ns+3))$	$\xi(4n^2-5n+2)+$ $+\rho(2n^2-2n+1)$	$2 \leq n$

- In the shuffling protocol the number of messages is higher in [BF83] than in [Yun85]. When one card is drawn in [BF83] all players cooperate, nonetheless in [Yun85] the protocol is run by two players only.

We compare [HLG00] and [GM82].

- In the shuffling protocol [HLG00] sends less messages that [GM82]. The protocol in [HLG00] is less complex than [GM82] and this means less messages.

- In the shuffling protocol, the comparison is similar as in the shuffling protocol.

[SRA81] jointly with [HLG00] are the two proposal with the minimum number of messages in both protocols.

The number of messages in [Cré85] is relatively moderated. There are proposals with less security properties that send more messages.

[ZVM03] sends a great number of messages. In the shuffling protocol each player creates a deck of cards. This deck is sent to the rest of players. So, each player

computes n decks of cards and sends them to the rest of players. There are n players and each sends one message to the rest $n-1$ players.

Table 3.75: Number of messages of TTP-free mental poker protocols

Number of messages	Card shuffling	Card drawing	Number of players
[SRA81]	1	3	2
[GM82]	2	104	2
[BF83]	$2(n-1)$	$\frac{2(n-1)^2}{n}$	$3 \leq n$
[Yun85]	$3(n-1)(1+r)$	$2+3k$	$2 \leq n$
[Cré85]	$2n(n-1)$	$(\frac{n-1}{2})^2+1$	$2 \leq n$
[Cré86]	$(n^2-n)(2s+3)$	$2(n-1)$	$2 \leq n$
[KKOT90]	$(n-1)(n(3s+1)+2)$	$2(n-1)$	$2 \leq n$
[HLG00]	1	3	2
[ZVM03]	$2n(n-1)$	$2(n-1)$	$2 \leq n$
[BS03]	$(n-1)(2n(s+1)+1)$	$(n-1)(4n-3)$	$2 \leq n$

In Table 3.76 we show the total length of messages of TTP-free mental poker protocols.

We first deal with contributions [Cré86], [KKOT90] and [BS03].

- The total length of messages in the shuffling protocol is lower in [BS03] than [Cré86] and [KKOT90]. The greater length of messages in [Cré86] could explain the high cost reported in [Edw94].

- The total length of messages in the drawing protocol is lower in [Cré86] than in [KKOT90] and [BS03].

We compare [Yun85] and [BF83].

- The total length of messages in the shuffling protocol is greater in [Yun85] than in [BF83]. In [BF83] the messages contain a value in $\{1,\ldots,52\}$. On the other side [Yun85] sends values in $\{1,\ldots,n\}$, where $n=pq$ and p and q are two large prime numbers.

- The total length of messages in the drawing protocol compares similarly .

We compare [HLG00] and [GM82].

- The total length of messages in the shuffling protocol is lower in [HLG00] than [GM82]. In [HLG00] an encrypted card is in $\{1, \ldots, p\}$, where p is a prime number. A cryptogram in [GM82] is formed by $[r]$ values in $\{1, \ldots, n\}$.

- The total length of messages in the drawing protocol is greater in [GM82] than [GM82]. The reason has been detailed previously.

The total length of messages in [SRA81] is minimal.

The total length of messages in [Cré85] is affordable and does not suppose a drawback.

In [ZVM03] the total length of messages is great. In the shuffling protocol players shuffle as many decks as players there are. Each deck is sent to all players. This method increases the number of messages and the amount of information sent. In the drawing protocol the length of messages is affordable in practice.

In general, if we compare the shuffling cost with the drawing cost, we see that the former is higher than the latter. So, we can conclude that the efficiency of a protocol suite is dominated by the shuffling protocol. In conclusion, card shuffling is what distinguishes practical from impractical mental poker protocols.

Table 3.76: Total length of messages of TTP-free mental poker protocols

Total length of messages	Card shuffling	Card drawing	Number of players
[SRA81]	$52[p]$	$3[p]$	2
[GM82]	$104[p]([r]+1)$	$104[p]$	2
[BF83]	$104[r](n-1)$	$\frac{2(n-1)^2}{n}[r]$	$3 \leq n$
[Yun85]	$(n-1)[p](k+3)(52+\zeta)+$ $+(n-1)([p]3k\zeta+[H(m)]52)$	$52[H(m)]+$ $+[p](\zeta(k+3)+3k)$	$2 \leq n$
[Cré85]	$52[p](n-1)n$	$[6](\frac{n+1}{2})+$ $+[p](\frac{n^2-4n+7}{4})$	$2 \leq n$
[Cré86]	$(n^2-n)s52[r]([p]+1)+$ $+(n^2-n)(104[r][p]+s[r]\frac{105}{2})$	$\frac{n-1}{2}([r]([p]+1)+3[r])$	$2 \leq n$
[KKOT90]	$(n-1)n^2 52[p](1+s(n-1))+$ $(n-1)n52[p]2+$ $+(n-1)n52[r](1+s(n+1))+$ $+(n-1)ns$	$(n-1)[r]+[p]$	$2 \leq n$
[HLG00]	$52[p]$	$3[p]$	2
[ZVM03]	$(n-1)(n+1)n104[p]$	$4(n-1)[p]$	$2 \leq n$
[BS03]	$n(n-1)[p]104(1+2s)+$ $+[r](n-1)(n(1+s52)+52)$	$(n-1)(6n-4)[p]$	$2 \leq n$

Chapter 4

On the security of an efficient TTP-free mental poker protocol

We present in this chapter an attack that exploits a security flaw of the Zhao *et al.* mental poker protocol [ZVM03]. We found this weakness when carrying out the comparative survey presented in Chapter 3. The attack was published in [CD04]. The authors of [ZVM03] have presented a modification of their protocol in [ZV05]. Nevertheless, the new proposal still has an important security flaw, which we describe briefly.

4.1 Introduction

As can be seen in Section 3.3.11, the Zhao *et al.* [ZVM03] proposal has a reasonable computational cost. Unfortunately, the use by Zhao *et al.* of an ElGamal-like commutative cryptosystem introduces a basic weakness. With little computation, a player can decrypt the encrypted cryptograms and find the cleartext cards x_i, for $i = 1$ to 52.

4.2 The attack

First of all, note that the attack scenario is a known-cleartext one. The cleartext card values $X = \{x_1, \ldots, x_{52}\}$ are assumed to be public or at least known to all players. However, when choosing an encrypted card or cryptogram, the player does not see the cleartext card x_i in it because this value is multiplied by a *hiding factor*, which will be denoted by f_h.

In Step 4 of Protocol 36 (shuffling protocol in Zhao *et al.*'s proposal, see Chapter 3) we can see that all sets $E_{n,i}$ have the same elements. Therefore, in order to simplify the notation we will denote $E = E_{n,i} \forall i \in \{1, \ldots, 52\}$. This set has the following elements $E = \{e_1, \ldots, e_{52}\}$, where $e_i = (y_{i,1}, y_{i,2})$. In Step 3(c)iB of Protocol 36 it can be seen that α_i^i is added to $y_{i,2}$. This operation is done $\forall i \in \{1, \ldots, 52\}$. Then we can assert that $y_{i,2} = \{\alpha_1^{r_1}, \ldots, \alpha_{52}^{r_{52}}\}$ $\forall i \in \{1, \ldots, 52\}$. Moreover, in Step 3(c)iA of Protocol 36 we can see that all elements are multiplied by the same factor $\beta_i^{r_i}$. In other words, all values x_i are hidden by the same hiding factor

$$f_h = \beta_1^{r_1} \cdots \beta_n^{r_n}$$

If an attacker multiplies any encrypted card $y_{i,1}$ by f_h^{-1}, she will be able to find its cleartext value x_i without knowing the encryption key:

$$y_{i,1} \cdot f_h^{-1} = x_i(\beta_1^{r_1} \cdots \beta_n^{r_n})(\beta_1^{r_1} \cdots \beta_n^{r_n})^{-1} = x_i$$

Now it remains to show how an attacker can find and invert f_h. This computation can be done as follows:

Procedure 30 (X)

1. *Let x_i and x_j ($i \neq j$) be two card values chosen during the initialization step;*
2. *The attacker computes the multiplicative inverses x_i^{-1} and x_j^{-1} modulo p;*
3. *The attacker multiplies each encrypted card $y_{k,1}$, for $k = \{1, \cdots, 52\}$ by x_i^{-1} and a first set of values D_1 is obtained. One of the values in this set is f_h:*

$$y_{i,1} \cdot x_i^{-1} = x_i \beta_1^{r_1} \cdots \beta_n^{r_n} x_i^{-1}$$

$$= \beta_1^{r_1} \cdots \beta_n^{r_n} = f_h$$

4. *The procedure of the previous step is repeated by the attacker with* x_j^{-1} *and a second set of values* D_2 *is obtained. Again, one of the values in* D_2 *is* f_h*:*

$$y_{j,1} \cdot x_j^{-1} = x_{i_2} \beta_1^{r_1} \cdots \beta_n^{r_n} x_{j,}^{-1}$$
$$= \beta_1^{r_1} \cdots \beta_n^{r_n} = f_h$$

5. *With overwhelming probability,* $D_1 \cap D_2$ *contains a single element and this element is* f_h*.*

6. *Once* f_h *is known, computing its multiplicative inverse* f_h^{-1} *modulo* p *is trivial.*

The cost of this attack is 52 products for Step 3 plus 52 products for Step 4; that is, 104 products are enough to decrypt the deck of cards.

4.3 Efficient TTP-free mental poker protocols

Zhao *et al.* presented in [ZV05] a modification of [ZVM03]. Next, this new protocol is briefly reviewed and its (new) security flaw is described.

Assume there are n ordered players. The cleartext card values $X = \{x_1, \ldots, x_{52}\}$ are assumed to be public or at least known to all players. The deck of cards is shuffled for all players. Player $\mathcal{P}_1$ encrypts and permutes the values x_i in X for the first time and sends the cryptograms to $\mathcal{P}_2$. Player $\mathcal{P}_2$ re-encrypts the cryptograms. She permutes the re-encrypted cryptograms and sends them to $\mathcal{P}_3$. This process is repeated up to $\mathcal{P}_n$.

Player $\mathcal{P}_1$ knows the cleartext of each cryptogram in each iteration, and of course she knows the cleartext of each cryptogram in the final shuffled deck of cards.

The encryption and decryption system used in [ZV05] is the same presented in [ZVM03] (see Section 3.3.9 for more details).

Without loss of generality, we can assume that Protocol 42 is run by $\mathcal{P}_1$ and $\mathcal{P}_2$, who have each an ElGamal-like key pair: $K_1 = \{p, \alpha_1, k_1, \beta_1 = \alpha_1^{k_1}\}$ for $\mathcal{P}_1$ and $K_2 = \{p, \alpha_2, k_2, \beta_2 = \alpha_2^{k_2}\}$ for $\mathcal{P}_2$.

Let $E_0 = \{e_{0,1}, \ldots, e_{0,52}\}$, where $e_{0,i} = \{y_{0,i,1} = 1, y_{0,i,2} = \emptyset\}$

Protocol 42 (Card shuffling (X))

1. *$\mathcal{P}_1$ uses Procedure 31 with E_0, α_1 and β_1. She obtains E_1 and π_1;*

2. *$\mathcal{P}_1$ sends E_1 to $\mathcal{P}_2$;*

3. *$\mathcal{P}_2$ uses Procedure 31 with E_1, α_2 and β_2. She obtains E_2 and π_2*

4. *$\mathcal{P}_2$ publish E_2;*

The set E_2 is the shuffled deck of cards. Procedure 31 encrypts and permutes the elements in E.

Procedure 31 ($E = \{e_1, \ldots, e_{52}\}$,$\alpha$,$\beta$)

1. *Choose a set of secret random numbers $R = \{r_1, \ldots, r_{52}\}$;*

2. *Choose a random permutation π of 52 values;*

3. *Let $e_i = (y_{i,1}, y_{i,2})$; for $i = 1$ to 52 do:*

   (a) *compute $y'_{i,1} = y_{i,1}\beta^{r_i} \bmod p$;*

   (b) *compute $y'_{i,2} = y_{i,2} \cup \alpha^{r_i} \bmod p$;*

   (c) *form $e'_i = (y'_{i,1}, y'_{i,2})$;*

4. *form $E' = \{e'_{\pi(1)}, \ldots, e'_{\pi(52)}\}$;*

5. *return E' and π;*

We now examine the sets E_1 and E_2. We have that $E_1 = \{e_{1,1}, \ldots, e_{1,52}\}$, where $e_{1,i} = \{y_{1,i,1}, y_{1,i,2}\}$ and $y_{1,i,2} = \{\alpha_1^{r_{1,i}}\}$. On the other hand, $E_2 = \{e_{2,1}, \ldots, e_{2,52}\}$, where $e_{2,j} = \{y_{2,j,1}, y_{2,j,2}\}$ and $y_{2,j,2} = \{\alpha_1^{r_i}, \alpha_2^{r_j}\}$.

$\mathcal{P}_1$ knows that $e_{1,i}$ is the encryption of a cleartext x_s. $\mathcal{P}_1$ does not need to decrypt $e_{1,i}$ to know this, because she knows the permutation π_1. She can link $y_{1,i,2}$ to the cleartext x_s. We can see as $e_{2,j}$ contains the element $\alpha_1^{r_i} = y_{1,i,2}$, so we can conclude that $e_{2,j}$ is the encryption of x_s.

In this way, player $\mathcal{P}_1$ can find the cleartext of all cryptograms in E_2. The second element of $e_{2,i}$ reveals the value of the card. This shortcoming could have been detected with an accurate security analysis.

One possible solution is to use the ElGamal re-masking (see Section 2.2.4) instead of the proposed encryption and decryption system. This solution is used by other authors, such as [SSG02] and [BS03]. However, in this case the novelty of [ZV05] would be very thin.

4.4 Conclusions

We have shown that replacing an exponential commutative cryptosystem with an ElGamal-like commutative cryptosystem as done in [ZVM03] turns a safe TTP-free mental poker protocol (Shamir-Rivest-Adleman) into a weak protocol (Zhao *et al.*). The weakness is that any player (or any outsider knowing the deck coding) can decrypt encrypted cards without knowing the encrypted key.

In [ZV05], Zhao *et al.* do not re-use the same secret value in all cards. Nevertheless, the proposed encryption and decryption system allows the first player to find the cleartexts of the final encrypted shuffled deck of cards.

Chapter 5

TTP-free protocol based on homomorphic encryption

A solution [CDRB03] for obtaining impartial random values in on-line gambling is presented in this chapter. Unlike most previous proposals, our method does not require any TTP and allows e-gambling to reach standards of fairness, security and auditability similar to those common in physical gambling.

Although our solution is detailed here for the particular case of games with reversed cards (*e.g.* poker), it can be easily adapted for games with open cards (*e.g.* blackjack) and for random draw games (*e.g.* keno).

Thanks to the use of permutations of homomorphically encrypted cards, the protocols described have moderate computational requirements. Askarov *et al.* implemented [CDRB03] in their case study of mutual distrust [Ask05] because they argued that [CDRB03] is "practical in terms of computational requirements".

Our solution is described in detail in Section 5.1. A security analysis considering various possible attacks is reported in Section 5.2. Section 5.4 is a conclusion.

This protocol has been patented by Scytl Online World Security, S.A. [CRBD02], and all rights of this protocol are property of Scytl Online World Security, S.A. This company have implemented some parts of this protocol in their gambling products.

5.1 Our protocol suite for e-gambling with reversed cards

In our protocol suite for e-games with reversed cards, deck shuffling is carried out in an efficient and fair way and without the help of any TTP.

All players co-operate in shuffling, so that no player or player coalition can force a particular outcome, *i.e.* determine the card that will be obtained after shuffling. Every player generates a random permutation of the card deck and keeps it secret; the player then commits to her permutation using a bit commitment protocol. The shuffled deck is formed by the composition of all player permutations. The use of permutations to shuffle the deck was introduced in [BF83], and also was used in [Cré86] [Cré85].

Note that reversing cards in the physical world translates to encrypting cards in e-gambling. Now, permuting (*i.e.* shuffling) encrypted cards requires encryption to be homomorphic, so that the outcome of permuting and decrypting (*i.e.* opening) a card is the same that would be obtained had the card been permuted without prior encryption (*i.e.* reversal). The use of an additive homomorphic criptosystem to shuffle the deck of cards and to maintain the privacy of the cards was used initially in [KKOT90].

5.1.1 Card representation and permutation

In most e-gambling approaches, a prescribed ordering of cards in the deck is assumed, so that a card is represented by a scalar corresponding to its rank. In our protocol, a card representation is needed which allows card operations and permutations. Thus, we will map the usual scalar representation to a vector representation in the way described below.

Definition 4 (Card vector representation) *Let t be the number of cards in the deck. Let z be a prime number chosen by a player. A card can be represented as a vector*

$$v = (a_1, \cdots, a_t) \tag{5.1}$$

where there exists a unique $i \in \{1, \cdots, t\}$ *for which* $a_i \bmod z \neq 0$*, whereas* $\forall j \neq i$ *it holds that* $a_j \bmod z = 0$*. The value of the card is* i*; assuming a prescribed ordering of cards,* i *is interpreted as a rank identifying a particular card.*

The above vector representation for cards allows card permutations to be represented as matrices in a natural way.

Definition 5 (Card permutation matrix) *A permutation* π *over a deck of* t *cards is a bijective mapping that can be represented as a square matrix* Π *with* t *rows called* card permutation matrix*, where rows and columns are vectors of the form described by Expression (5.1):*

$$\Pi = \begin{pmatrix} \pi_{11} & \pi_{12} & \cdots & \pi_{1t} \\ \vdots & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ \pi_{t1} & \pi_{t2} & \cdots & \pi_{tt} \end{pmatrix} \tag{5.2}$$

The i*-th row of matrix* Π *is card* $\pi(i)$*,* i.e. *the card resulting from applying permutation* π *to the card having rank* i*. Thus, all elements in the* i*-th row of* Π *are* $0 \bmod z$ *except* π_{ij}*, where* $j = \pi(i)$*.*

See Example 1 in Section 5.3 for an illustration of the above definition. The result below is straightforward from the above construction:

Proposition 1 (Permutation algebra) *With the above representation for cards and permutations, the result* $w = \pi(v)$ *of permuting a card* v *using a permutation* π *can be computed as* $w = v \cdot \Pi$*, where* $\cdot$ *denotes vector product. For this computation to work properly, the same value* z *must be used for* v *and* Π*.*

Example 2 in Section 5.3 illustrates the above proposition. Every player $\mathcal{P}_i$ will use her own prime modulus z_i for representing and permuting cards. In order for $\mathcal{P}_i$ to be able to operate her card permutation matrix with a card coming from a previous player $\mathcal{P}_{i-1}$, player $\mathcal{P}_i$ must represent her permutation matrix using the modulus z_{i-1} corresponding to $\mathcal{P}_{i-1}$. This means transforming her permutation matrix based on z_i into an equivalent permutation matrix based on z_{i-1}, as defined below:

Definition 6 (Equivalent card permutation matrices) *Let* Π *and* Π' *be two* $t \times t$ *card permutation matrices using moduli* z *and* z'*, respectively. Then* Π *and* Π' *are said to be equivalent if they represent the same permutation* π *of a set of* t *cards. Specifically,* $\Pi = \{\pi_{ij}\}$ *and* $\Pi' = \{\pi'_{ij}\}$ *are equivalent if and only if,* $\forall i, j \in \{1, \cdots, t\}$*, one has*

$$\pi_{ij} \bmod z \neq 0 \Leftrightarrow \pi'_{ij} \bmod z' \neq 0$$

$$\pi_{ij} \bmod z = 0 \Leftrightarrow \pi'_{ij} \bmod z' = 0$$

5.1.2 Distributed notarization chains

In order to meet the game auditability requirement, we introduce in this paper a new tool called distributed notarization chains (DNC). DNCs have a philosophy similar to the one of Lamport's hash chains [Lam81]. Each operation performed during the e-game will be notarized as a link of a DNC. DNCs are efficiently computable and consist of links which are constructed and chained as follows:

Link structure Every link m_k of the DNC is formed by two fields: a data field D_k and a chaining value X_k. The data field D_k consists of three subfields:

- Timestamp T_k, which contains the link generation time according to the clock of the participant who generated the link (synchronizing the clocks of all participants is not needed).
- Link subject or concept C_k, which describes the information contained in the link, *e.g.* a step in the game, a commitment or an outcome.
- Additional attributes V_k, which depend on the subject C_k. For a link corresponding to a commitment, V_k will contain the encrypted commitment.

Link chaining Chaining is guaranteed by chaining values X_k included in each link. First, the chaining value X_{k-1} of the previous link is concatenated with the data field D_k of the current link; then the hash value of the concatenation is computed and signed with the private key of the author of the k-th link, *i.e.*

$$X_k = S_{author}(D_k | X_{k-1}) \tag{5.3}$$

There are moments during the game at which operations do not need to be sequential, but can be carried out in parallel by the participants. Parallel execution can be accomodated in the distributed notarization system by introducing two operations:

Chain expansion Parallel execution can be notarized by *expanding the DNC*, whereby participants $\{\mathcal{P}_1, \cdots, \mathcal{P}_n\}$ independently compute their chaining values

$$X_k^{\mathcal{P}_1}, \cdots, X_k^{\mathcal{P}_n}$$

using the chaining value X_{k-1} of the previous link:

$$\begin{aligned} X_k^{\mathcal{P}_1} &= S_{\mathcal{P}_1}(D_k^{\mathcal{P}_1}|X_{k-1}) \\ &\vdots \\ X_k^{\mathcal{P}_n} &= S_{\mathcal{P}_n}(D_k^{\mathcal{P}_n}|X_{k-1}) \end{aligned} \tag{5.4}$$

Chain contraction It happens when the protocol requires sequential execution after an expansion stage where n participants have computed links in parallel. A single link is obtained which is chained to previous links. The participant initiating the sequential execution concatenates its data field D_k with all chaining values of previous links $\{X_{k-1}^{\mathcal{P}_1}|\ldots|X_{k-1}^{\mathcal{P}_n}\}$, computes the hash of the concatenation and signs it to obtain the chaining value X_k of the first sequential link:

$$X_k = S_{\mathcal{P}_i}(D_k|X_{k-1}^{\mathcal{P}_1}|\cdots|X_{k-1}^{\mathcal{P}_n}) \tag{5.5}$$

The following properties of a DNC make it a good tool for distributed notarization:

- The DNC is not possessed by a single participant, but by all of them. Whenever a participant builds a link of a DNC, she sends it to the other participants, so that all of them see the same DNC. If a participant recomputes the chain to add false links to it, the manipulated chain will not match the copies held by the rest of participants, and manipulation will be detected.

- If someone deletes or modifies one or more links, the chain will show an inconsistency at the point of deletion.

- The chaining and the structure of links allows the exact sequence and time of link construction to be securely determined.
- As links are signed by the participant who built them, link authorship can be securely determined.
- Link computation is performed in parallel whenever the protocol allows it (using the expansion operation). This results in improved performance without degrading security.

5.1.3 Protocol description

As mentioned above, DNCs are used for distributed notarization. Whenever a player builds a link of a DNC, she sends it the other players, so that all of them see the same DNC. Furthermore, each link is digitally signed by the player who built it, which guarantees authentication, integrity and non-repudiability for that link. A link can only be appended at the end of the chain and no participant can modify or delete any link without being detected.

The initialization protocol is as follows:

Protocol 43 (Shuffle the cards)

1. *Each player $\mathcal{P}_i$ is assumed to have an asymmetric key pair (P_i, S_i) whose public key has been certified by a recognized certification authority. Assume the card deck consists of t cards.*
2. *$\mathcal{P}_i$ does:*
   (a) *Generate a permutation π_i of the card deck and keep it secret.*
   (b) *Generate a symmetric secret key K_i corresponding to a homomorphic cryptosystem allowing algebraic operations (additions and multiplications) to be carried out directly on encrypted data. The preferred choice for this homomorphic cryptosystem is [DF02], also described in patent [DS00], which is secure against ciphertext-only attacks; an alternative choice is [DF96].*

(c) *Choose a prime value z_i which falls within the range of the cleartext space of the homomorphic cryptosystem used.*

(d) *Build a link of the DNC such that its subject C_k says that the link contains z_i used by $\mathcal{P}_i$ and the field V_k contains z_i. Link building can be done in parallel by all players, which expands the DNC at this point.*

(e) *Build the card permutation matrix Π_i corresponding to π_i, using z_i.*

(f) *Commit to this permutation Π_i using a bit commitment protocol [Sch96]. Denote the resulting commitment by Cp_i.*

(g) *Build the next link of the chain using the previous expanded link. The subject C_k indicates that this link contains the commitment and the field V_k contains Cp_i.*

(h) *Choose s values $\{\delta_1, \cdots, \delta_s\}$ such that $\delta_j \bmod z_i = 0$, $\forall j \in \{1, \cdots, s\}$ and $s > t$.*

(i) *Choose s values $\{\epsilon_1, \cdots, \epsilon_s\}$ such that $\epsilon_j \bmod z_i \neq 0$, $\forall j \in \{1, \cdots, s\}$ and $s > t$.*

(j) *Encrypt the previous values under the symmetric key K_i to get $d_j = E(K_i, \delta_j)$ and $e_j = E(K_i, \epsilon_j)$, $\forall j \in \{1, \cdots, s\}$.*

(k) *Build the next link of the chain with a field V_k containing the set $\mathbf{D} = \{d_1, \cdots, d_s\}$ and a subject C_k indicating that the link contains encrypted versions of a set of values which are 0 modulo z_i.*

(l) *Build another link of the chain with a field V_k containing the set $\mathbf{E} = \{e_1, \cdots, e_s\}$ and a subject C_k indicating that the link contains encrypted versions of a set of values which are different from 0 modulo z_i.*

(m) *Generate the vector representation for the t cards in the deck $\{w_1, \cdots, w_t\}$ and encrypt them under K_i using the aforementioned homomorphic cryptosystem to obtain $w'_j = E(K_i, w_j)$.*

(n) *Permute the encrypted cards.*

(o) *Build the next link of the chain. The subject C_k indicates that the link*

contains a card deck encrypted by $\mathcal{P}_i$. The field V_k contains the encrypted cards in the order resulting from the permutation in Step (2n).

Once initialization is over, the player playing the croupier role performs chain contraction by building a link with a chaining value mixing together the last link created by every player $\mathcal{P}_i$. The subject C_k of this link indicates that initialization is over and that the game can start. The field V_k is empty.

Note that the initialization protocol results in each player $\mathcal{P}_i$ publishing in the DNC her z_i and a commitment to her permutation; the actual permutation stays secret, though. When $\mathcal{P}_i$ wants a card, the following protocol is started:

Protocol 44 (Draw a card)

1. *$\mathcal{P}_i$ does:*

   (a) *Pick a value v_0 such that it falls within the range of cards in the deck,* i.e. $1 \leq v_0 \leq t$, *and which has not previously been requested. This operation is simple because it is public. All participants know the initial values chosen in previous steps.*

   (b) *Build the next link of the chain, where V_k contains the vector representation w_0 of card v_0, and the subject C_k indicates that $\mathcal{P}_i$ wants the card represented by w_0. The link is received by all players and, in particular, by $\mathcal{P}_1$.*

2. *$\mathcal{P}_1$ does:*

   (a) *Check the validity of the link sent by $\mathcal{P}_i$, compute her equivalent card permutation Π'_1 for the modulus z_i published by $\mathcal{P}_i$ and permute w_0 to obtain $w_1 = w_0 \cdot \Pi'_1$.*

   (b) *Build the next link of the chain, whose subject indicates that the link contains w_1 and that $\mathcal{P}_2$ is supposed to perform the next computation; the field V_k contains the value w_1.*

3. *For $j = 2$ to $i - 1$, player $\mathcal{P}_j$ does:*

(a) *Check the validity of the link sent by* $\mathcal{P}_{j-1}$*, compute her equivalent card permutation matrix* Π'_j *for the modulus* z_{j-1} *published by* $\mathcal{P}_{j-1}$ *and permute* w_{j-1} *to obtain* $w_j = w_{j-1} \cdot \Pi'_j$.

(b) *Build the next link of the chain, whose subject indicates that the link contains* w_j *and that* $\mathcal{P}_{j+1}$ *is supposed to perform the next computation; the field* V_k *contains the value* w_j.

4. *Player* $\mathcal{P}_i$ *does:*

(a) *Check the validity of the link sent by* $\mathcal{P}_{i-1}$*, compute her equivalent permutation matrix for* z_{i-1} *and permute* w_{i-1} *to obtain* $w_i = w_{i-1} \cdot \Pi'_i$.

(b) *Modify the* m*-th row of* Π_i *where* $m \in \{1, \cdots, t\}$ *is the value of card* w_{i-1}. *All values in the* m*-th row are changed to values that are nonzero modulo* z_i

(c) *Pick the encrypted card* w'_i *corresponding to clear card* w_i. *Note that the encrypted deck has been published in the last step of Protocol 43.*

(d) *Build the next link of the chain, with* $V_k = w'_i$ *and a subject* C_k *indicating that the link contains* w'_i *and that* $\mathcal{P}_{i+1}$ *is supposed to perform the next computation.*

5. *For* $j = i + 1$ *to* n*, player* $\mathcal{P}_j$ *does:*

(a) *Check the validity of the link sent by* $\mathcal{P}_{j-1}$ *and compute her equivalent card permutation matrix* Π'_j *for the modulus* z_{j-1} *published by* $\mathcal{P}_{j-1}$. *Use Protocol 45 below to encrypt* Π'_j *as* Π^c_j *under the key* K_i *corresponding to* $\mathcal{P}_i$.

(b) *Permute the encrypted card* w'_{j-1} *using her encrypted matrix* Π^c_j *to obtain* $w'_j = w'_{j-1} \cdot \Pi^c_j$.

(c) *Build the next link of the chain, with* $V_k = w'_j$ *and a subject indicating that the link contains* w'_j*; if* $j < n$*, the subject also indicates that* $\mathcal{P}_{j+1}$ *is supposed to perform the next computation.*

6. *When $\mathcal{P}_i$ sees the link computed by $\mathcal{P}_n$, she does:*

   (a) *Check the validity of the link computed by $\mathcal{P}_n$.*

   (b) *Decrypt the card w'_n contained in the link computed by $\mathcal{P}_n$ under her private key K_i to obtain the drawn card $w_n = D_{K_i}(w'_n)$. This finishes the card draw protocol.*

A loose end that remains is how $\mathcal{P}_j$ encrypts her permutation matrix Π'_j under $\mathcal{P}_i$'s secret key K_i. To do that, player $\mathcal{P}_j$ can only use the sets of encrypted values $\mathbf{D}$ and $\mathbf{E}$ published by $\mathcal{P}_i$ in Step (2k) of Protocol 43. The specific procedure is described below:

Protocol 45 (Permutation matrix homomorphic encryption)

Let the homomorphic encryption of the cleartext matrix $\Pi_j = \{\pi_{kl}\}$ under K_i be $\Pi^c_j = \{\pi^c_{kl}\}$. To compute π^c_{kl}, for $1 \leq k, l \leq t$, player $\mathcal{P}_j$ does the following:

1. *Generate a pseudorandom value g, such that $1 \leq g \leq s$, where s is the size of the sets $\mathbf{D}, \mathbf{E}$ of encrypted values published by $\mathcal{P}_i$ in Protocol 43.*
2. *Randomly pick g values $\{d_1, \cdots, d_g\}$ of the set $\mathbf{D}$ and add them to obtain $h = d_1 + \cdots + d_g$. Remember that values in $\mathbf{D}$ are 0 modulo z_i, because they are homomorphically encrypted versions of values which are 0 modulo z_i, so by the homomorphic properties, h is also 0 modulo z_i.*
3. *Generate a pseudorandom value c such that $c \bmod z_i \neq 0$ and compute $h' = c \cdot h$.*
4. *If $\pi_{kl} \bmod z_i = 0$, then $\pi^c_{kl} := h'$.*
5. *If $\pi_{kl} \bmod z_i \neq 0$, then*

   (a) *Generate a pseudorandom value g' such that $1 \leq g' \leq s$.*

   (b) *$\pi^c_{kl} := h' + e_{g'}$, where $e_{g'}$ is the g'-th element of the set E.*

A player can also discard a card w by building a link of the DNC whose subject says that the player is discarding a card and whose field V_k contains the encrypted version of the discarded card.

5.1.4 Extensions

In each run of Protocol 44, player $\mathcal{P}_i$ gets only one card. The protocol can be extended so that the player draws several cards in a single protocol run. To do this, let us define a way to pack several cards together:

Definition 7 (Multicard) *Given a deck with t cards and a prime value z, a multicard ξ is a vector of t elements*

$$\xi = (a_1, \cdots, a_t) \tag{5.6}$$

where there are up to $M < t$ components a_i, such that $a_i \bmod z \neq 0$. The index i of each component a_i that is nonzero modulo z represents one of the cards contained in the multicard. By convention, a multicard with all t components nonzero modulo z is not valid (it would contain the whole deck).

Protocol 44 can be adapted to multicards as explained below:

- At Step (1a) of Protocol 44, $\mathcal{P}_i$ should choose the cards she wishes. Let these be $w_0^1, \cdots, w_0^x, x < t$ in vector representation.
- A multicard $\xi_0 = \sum_{j=1}^{x} w_0^j$ is obtained by adding the chosen cards.
- At Step (1b), the chain link would be computed using the multicard ξ_0 rather than a single card w_0.
- The protocol carries on until Step (4b), where all rows corresponding to values of cards in the multicard are modified.
- The protocol then proceeds as described above. In the final Step (6b), $\mathcal{P}_i$ decrypts the card computed by $\mathcal{P}_n$ and obtains a multicard.

5.1.5 Game validation

When a hand of the game is over, players should reveal their encryption keys and their permutations. The validation process is specified next:

Protocol 46 (Game validation) *Each player does the following:*

1. *Check that the permutation revealed and used by each player $\mathcal{P}_i$ is the same permutation π_i to which she committed when publishing the commitment Cp_i in Protocol 43 (initialization). This check implies verifying the bit commitment for player $\mathcal{P}_i$.*

2. *Decrypt cards $\{w_1', \ldots, w_t'\}$ published by each player $\mathcal{P}_i$ in the last step of Protocol 43 and check that the card deck is correct.*

3. *Use the private key K_i of each player $\mathcal{P}_i$ to decrypt the result of permuting encrypted cards at Step (5b) of Protocol 44. Check that permutations were correctly performed.*

4. *Check that cards discarded by players have not been used during the game.*

5. *If necessary, use the DNC chain to prove any detected misbehaviors by any player to a third-party (casino, court, etc.).*

5.2 Security analysis

Let us examine a collection of possible attacks and check that they fail:

A coalition wants to get the cards drawn by a player Player $\mathcal{P}_i$ draws one or more cards in an instance of Protocol 44. In subsequent instances belonging to the same hand, a coalition of players attempt to determine the cards drawn by $\mathcal{P}_i$. To do that, the coalition must construct a (multi)card with the card value(s) drawn by $\mathcal{P}_i$ and encrypt that (multi)card. However, the coalition will not get $\mathcal{P}_i$'s card(s) because, if a card is requested which had been previously requested, what is obtained is a vector with all components different from 0 modulo z_i. This is due to the modification of matrix Π_i at Step (4b) of Protocol 44. The resulting multicard has all components different from 0 modulo z_i and is thus a non-valid multicard (by definition).

A player uses a wrong modulus Assume that, to permute a card coming from player $\mathcal{P}_{k-1}$, player $\mathcal{P}_k$ computes her equivalent permutation matrix Π'_k using $z_j \neq z_{k-1}$. This is detected by $\mathcal{P}_i$ in the last step of Protocol 44, because w'_n decrypts into a non-valid card (with more than M nonzero components modulo z_i). Player $\mathcal{P}_i$ has no option other than reporting the wrong decryption; otherwise $\mathcal{P}_i$ would not be able to show her cards during game validation. Verification performed during Protocol 46 discloses the identity of the cheater $\mathcal{P}_k$.

A player does not use the permutation she committed to A player may choose to use a permutation Π different from the one she committed to during initialization. If the player changes her permutation for the whole game, this is detected during game validation. If the player changes her permutation only during some parts of the game, two things may happen:

- Some of the other players get duplicated cards. A player getting a duplicated card is forced to report it (otherwise she will not be able to show her cards during game validation). Upon such report, the game is stopped and game validation started.
- The change is not detected during the game. In this case, it is detected during game validation and the dishonest player is identified.

A player supplies wrong sets D or E If, during initialization, a player does not supply correct sets $\mathbf{D} = \{d_1, \cdots, d_s\}$ or $\mathbf{E} = \{e_1, \cdots, e_s\}$, then permutations of encrypted cards cannot be correctly computed. Two things can happen:

- The card obtained by $\mathcal{P}_i$ at the end of Protocol 44 is not valid. In this case, $\mathcal{P}_i$ is forced to report the problem.
- The card obtained by $\mathcal{P}_i$ is valid. In this case, the problem is detected during game validation.

A player supplies a wrong encrypted deck The game validation protocol verifies that all players have supplied correct encrypted decks during initialization.

A player builds an incorrect chain link During execution of Protocol 44, each player checks that the previous link of the chain has been correctly built. Any wrong link is reported (all players see all links so it is risky not to report a wrong link when discovered).

A player requests a card which had already been requested Cards requested at the beginning of Protocol 44 are recorded in the DNC. If a player requests a card that had already been requested, this is detected by the rest of the players (all of them see the DNC links).

A player does not correctly encrypt her permutation An incorrectly encrypted card permutation matrix can yield non-valid or duplicated cards, which is detected during the game. Even if all cards are valid, a wrongly encrypted permutation is detected during game validation.

Player withdrawal Depending on when a player withdraws from the game, different things happen:

- If a player withdraws during initialization, the game simply proceeds without her.
- If a player withdraws immediately after initialization, the game proceeds without her. In this case, the composition of permutations must be computed without using the permutation of the withdrawn player.
- If a player withdraws during the game (*i.e.* during Protocol 44), the game stops at the moment of withdrawal. Players show their cards and game validation is started. If the player has withdrawn without justification, she is fined.

5.3 Examples

Example 1 *Let $t = 5$ and consider the following permutation of five cards*

$$\pi = \begin{pmatrix} 3 & 5 & 4 & 1 & 2 \end{pmatrix} \tag{5.7}$$

If $z = 5$ is taken, the card permutation matrix is as follows

$$\Pi = \begin{pmatrix} 5 & 10 & 3 & 5 & 15 \\ 25 & 10 & 35 & 5 & 6 \\ 50 & 10 & 10 & 8 & 5 \\ 4 & 60 & 5 & 50 & 25 \\ 15 & 7 & 35 & 60 & 10 \end{pmatrix} \tag{5.8}$$

Note that, in the first row, the only element which is nonzero modulo 5 is the third one, that is, the one in the position corresponding to $\pi(1) = 3$. Similarly, in the second row, the only nonzero element modulo 5 is in the 6-th position, because $\pi(2) = 6$. And so on for the other rows.

Example 2 *Using $t = 5$, $z = 5$ and the permutation matrix of Example 1, card $v = (10, 15, 8, 20, 20)$ (with value 3) is permuted as*

$$w = (10, 15, 8, 20, 20) \cdot \begin{pmatrix} 5 & 10 & 3 & 5 & 15 \\ 25 & 10 & 35 & 5 & 6 \\ 50 & 10 & 10 & 8 & 5 \\ 4 & 60 & 5 & 50 & 25 \\ 15 & 7 & 35 & 60 & 10 \end{pmatrix} = (1205, 1670, 1435, 2381, 980) \tag{5.9}$$

Thus, the permuted card w has value 4 (the fourth card of the ranked deck). In this way, $\pi(3) = 4$, which is consistent with the fact that the only nonzero component in the third row of the permutation matrix is the fourth one.

5.4 Conclusion

A solution for obtaining impartial random values in on-line gambling has been presented in this chapter. Unlike most previous proposals, our method does not require any TTP and allows e-gambling to reach standards of fairness, security and auditability similar to those common in physical gambling. In addition, our solution has moderate computational requirements.

The solution has been specified for the particular case of games with reversed cards (*e.g.* poker), but it can be easily adapted for games with open cards (*e.g.* blackjack) and for random draw games (*e.g.* keno).

Chapter 6

A TTP-free mental poker protocol achieving player confidentiality

Computationally efficient TTP-free mental poker has an interest beyond e-gambling, namely as a testbed for multiparty computation. Indeed, many of the cryptographic primitives and protocols devised for TTP-free mental poker can be re-used for other multiparty applications such as e-voting, multiparty function evaluation, etc.

We present in this chapter a mental poker protocol [CSD04b] where:

- No TTP is required;
- The confidentiality of the strategy of players is preserved;
- The amount of computation required stays reasonably low. As will be shown in the cost analysis presented, this protocol is perfectly usable in practice, unlike most previous TTP-free solutions.

Section 6.1 describes our proposed protocol suite. The computational cost of our proposal is assessed in Section 6.2. A security analysis is given in Section 6.3. Finally, conclusions are drawn in Section 6.4.

6.1 Our protocol suite

Our proposal consists of five stages: initialization, card shuffling, card draw, card opening and card discarding. Each stage corresponds to the basic operations done by players in one poker hand. All messages sent by players during the five stages are posted on a bulletin board.

6.1.1 Initialization

The following public parameters are agreed upon by all players for card encryption:

- Two large prime numbers p and q such that $p = 2q + 1$;
- A generator α of a subgroup G of Z_p^* of order q, such that α is a quadratic non-residue;
- A security parameter s.
- A poker deck is composed by 52 cards.

Next, every player $\mathcal{P}_i$:

1. Selects her private key K_i so that $2 < K_i < q$ and K_i is an odd number.
2. Publishes her public key $\beta_i = \alpha^{K_i}$.

All players thereafter co-operatively generate a random set $X = \{x_1, \ldots, x_m\}$, where x_j are odd numbers $2 < x_j < q$.

Finally, players compute $\beta = \alpha^{K_1 \cdots K_n}$ without revealing their own keys K_i. This can be done in a secure way using [BCPQ01].

6.1.2 Card shuffling

The deck of shuffled, face-down cards will be co-operatively computed by all players in turn. We define an initial set $C_0 = \{c_{0,1}, \ldots, c_{0,52}\}$, where $c_{0,j} = (d_{0,j}, \alpha_{0,j}) = (\alpha^{x_j}, \beta)$. Note that, by construction, any α^{x_j} is a quadratic non-residue.

During shuffling, each player $\mathcal{P}_i$ takes C_{i-1} and generates C_i. As a result of this process, the set C_n of shuffled face-down cards is obtained. The shuffling protocol is:

Protocol 47 (Cards shuffling)

1. *For $i = 1$ to n:*

   (a) *Using Procedure 32 below, $\mathcal{P}_i$ computes*

   $$\{C_i, R_i, \pi_i\} = \textit{Procedure } 32(C_{i-1});$$

   (b) *$\mathcal{P}_i$ runs Protocol 48$(C_{i-1}, C_i, \pi_i, R_i, s)$ below to prove that C_i has been properly computed;*

   (c) *$\mathcal{P}_i$ broadcasts C_i (while keeping R_i and π_i secret).*

In the above protocol, the last player $\mathcal{P}_n$ obtains the set C_n, which is the deck of cards encrypted and shuffled by all players $\mathcal{P}_i$. Note that every player $\mathcal{P}_i$ in $\{\mathcal{P}_1, \ldots, \mathcal{P}_n\}$ can verify that all $c_{n,j} \in C_n$ are quadratic non-residues. This requirement is meant to thwart the card-marking attack in [Lip81]: all cards share the same property of being quadratic non-residues.

We describe below Procedure 32 used in Protocol 47.

Procedure 32 (C)

1. *Generate a random set $R = \{r_1, \ldots, r_{52}\}$ of odd numbers r_j so that $2 < r_j < q$, using a uniform distribution;*

2. *For $j = 1$ to 52 compute $c'_j = (d_j^{r_j}, \alpha_j^{r_j})$, where $c_j = (d_j, \alpha_j)$ is the j-th element of C;*

3. *Generate a random permutation π over $\{1, 2, \ldots, 52\}$ (*e.g. *using the Trotter-Johnson unranking algorithm [KS98]);*

4. *Permute the values c'_j and obtain $C^* = \{c'_{\pi(1)}, \ldots, c'_{\pi(52)}\}$;*

5. *Return $\{C^*, R, \pi\}$.*

As indicated in Protocol 47 above, Protocol 48 is used by $\mathcal{P}_i$ to prove to the rest of players that she has properly computed C_i.

Protocol 48 $(C_{i-i}, C_i, \pi_i, R_i, s)$

1. *For* $k \in \{1, \ldots, s\}$ $\mathcal{P}_i$ *does:*
   (a) *Using Procedure 32 above, compute* $\{C_{i,k}, R_{i,k}, \pi_{i,k}\}$ = *Procedure* $32(C_i)$*;*
   (b) *Publish* $C_{i,k}$*;*
2. *The rest of players send a random index* $u = \{u_1, \ldots, u_s\}$ *of* s *bits;*
3. *For* $u_k \in \{u_1, \ldots, u_s\}$*:*
   (a) *If* $u_k \equiv 1$ *then:*
      i. $\mathcal{P}_i$ *reveals the set* $R_{i,k} = \{r_{i,k,1}, \cdots, r_{i,k,52}\}$ *and the permutation* $\pi_{i,k}$*;*
      ii. *All players verify* $(C_i, C_{i,k})$ *using Procedure 33* $(C_i, C_{i,k}, R_{i,k}, \pi_{i,k})$*;*
   (b) *If* $u_k \equiv 0$ *then:*
      i. $\mathcal{P}_i$ *computes* $\pi'_{i,k} = \pi_{i,k} \circ \pi_i$ *and* $R'_{i,k} = \{r'_{i,k,1}, \ldots, r'_{i,k,52}\}$*, where* $r'_{i,k,j} = r_{i,\pi_i(\pi_{i,k}(j))} \cdot r_{i,k,\pi_{i,k}(j)}$ *(note that* $r_{i,\pi_i(\pi_{i,k}(j))} \in R_i$ *and* $r_{i,k,\pi_{i,k}(j)} \in R_{i,k}$*);*
      ii. $\mathcal{P}_i$ *reveals* $R'_{i,k}$ *and* $\pi'_{i,k}$*;*
      iii. *All players verify* $(C_{i-1}, C_{i,k})$ *using Procedure 33* $(C_{i-1}, C_{i,k}, R'_{i,k}, \pi'_{i,k})$.

In Step 1 of Protocol 48, C_i is encrypted s times, so we obtain s encryptions of C_i. When verifying whether C_i has been well encrypted into $C_{i,k}$ using $R_{i,k}$, or C_{i-1} into $C_{i,k}$ using $R'_{i,k}$, one just needs to verify whether the exponentiations are correct. This is done in Procedure 33.

Procedure 33 (C_a, C_b, R, π)

1. *For* $j = 1$ *to* 52 *do:*
   (a) *Consider* $c_{b,j} = (d_{b,j}, \alpha_{b,j}) \in C_b$ *and* $c_{a,\pi(j)} = (d_{a,\pi(j)}, \alpha_{a,\pi(j)}) \in C_a$ *and* $r_{\pi(j)} \in R$*;*
   (b) *Check* $d_{b,j} \stackrel{?}{=} (d_{a,\pi(j)})^{r_{\pi(j)}}$*;*
   (c) *Check* $\alpha_{b,j} \stackrel{?}{=} (\alpha_{a,\pi(j)})^{r_{\pi(j)}}$.

6.1.3 Card draw

We now specify the card draw protocol. Let us assume that player $\mathcal{P}_u$ wants to draw a card from the shuffled face-down deck C_n.

Protocol 49 (Card drawing)

1. *$\mathcal{P}_u$ chooses an index j so that card $c_{n,j} = (d_{n,j}, \alpha_{n,j}) \in C_n$ has not been drawn yet;*
2. *$\mathcal{P}_u$ publishes $e_0 = \alpha_{n,j}$. Let $r = 0$;*
3. *For each i in $\{1, \cdots, u-1, u+1, \cdots, n\}$:*
   (a) *Let $r = r + 1$;*
   (b) *$\mathcal{P}_i$ computes and publishes $e_r = (e_{r-1})^{K_i^{-1}}$;*
   (c) *$\mathcal{P}_i$ proves in zero-knowledge that $\log_\alpha \beta_i = \log_{e_r} e_{r-1}$. This can be efficiently done using the Chaum-Pedersen proof [CP92];*
4. *$\mathcal{P}_u$ computes $e_n = (e_{n-1})^{K_u^{-1}}$;*
5. *The extracted card corresponds to the value $x \in X$ that satisfies $d_{n,j} = (e_n)^x$;*
6. *$\mathcal{P}_u$ obtains x as the drawn card.*

6.1.4 Card opening

In traditional poker, players can show their cards easily. In mental poker that is not so easy: a player must prove to the rest of players that she is the owner of her cards. We use the card opening protocol to prove it.

Let us assume that a player $\mathcal{P}_u$ has drawn a card $c_{n,j}$ and has received e_{n-1} from $\mathcal{P}_n$. It is publicly known that e_{n-1} was obtained during the card draw protocol and is a card in $\mathcal{P}_u$'s hand. In this scenario, $\mathcal{P}_u$ does the following steps:

Protocol 50 (Card opening)

1. *Publish* $e_n = (e_{n-1})^{K_u^{-1}}$ *and* x*;*

2. *Prove in zero-knowledge that* $\log_{e_n} e_{n-1} = \log_\alpha \beta_u$*. This can be efficiently done using the Chaum-Pedersen proof [CP92].*

Any player can verify that $x \in X$ satisfies $d_{n,j} = (e_n)^x$ in G.

6.1.5 Card discarding

A player discards a card when she commits herself to not using it. Using the notation of Protocol 50, let us assume that a player $\mathcal{P}_u$ has drawn a card $c_{n,j}$ and has received e_{i-1}. $\mathcal{P}_u$ discards $c_{n,j}$ when she sends the *discard* message with $c_{n,j}$ to the bulletin board. If $\mathcal{P}_u$ wants to open one discarded card $c_{n,j}$ the rest of players can detect the fraud because $c_{n,j}$ is on the bulletin board.

6.2 Computational cost

The computational cost of the protocol suite presented in Section 6.1 is dominated by Protocol 47 (card shuffling). In general, card shuffling is what distinguishes practical from impractical mental poker protocols. Therefore, we focus in this section on the analysis of the computational cost of Protocol 47.

As we have noted in section 3.1 ξ and ρ are the computational cost (time) incurred by exponentiation and multiplication, respectively. The computational cost of other operations is assumed negligible and is denoted by ϵ. s is the security parameter that we have defined in section 6.1.1.

Table 6.1 gives the breakdown of costs for the different steps of Protocol 47.

Table 6.2 shows the cost of Protocol 48.

Table 6.3 shows the cost of Procedure 32.

Table 6.4 shows the cost of Procedure 33.

We can compare the cost of shuffling cards using our Protocol 47 with the cost of shuffling using the Barnett-Smart scheme [BS03]. In our proposal the cost of shuffling and verifying a deck is $n(104\xi(sn+1) + 26s\rho)$ (see Table 6.1). According

Table 6.1: Costs of shuffling protocol

	Number of messages	Total length of messages	Computational cost
Card shuffling	$n2(n-1)(s+1)$	$n([p]52((n-1)(2+s)+s))+$ $+n([r](n-1+s52))$	$104n\xi(sn+1)+$ $+26ns\rho$
Step 1	$n2(n-1)(s+1)$	$n([p]52((n-1)(2+s)+s))+$ $+n([r](n-1+s52))$	$104n\xi(sn+1)+$ $+26ns\rho$
Step 1a			104ξ
Step 1b	$(n-1)(2s+1)$	$[p]52s(2n-1)+[r](n-1+s52)$	$sn104\xi+26s\rho$
Step 1c	$(n-1)$	$[p]104(n-1)$	ϵ

to our analysis of the operations required by Barnett-Smart (see section 3.3.10), the total Barnett-Smart cost is at least $(104(\rho+\xi)(ns+1)+52\rho)$ (see Table 3.66). With our proposal we save at least $\rho 26n((4n-1)+6)$ operations.

In order to give some empirical execution times for our proposal, average values for times ξ and ρ have been found using a program running on a PC with an IBM T41 Centrino 1.5 GHz processor with 512MB of RAM and a Debian operating system with a GNU/Linux 2.6.7-1 kernel. The program computes a large number of exponentiations and multiplications and records the average time for each operation. Table 6.2 shows the estimated values for ξ, ρ and the overall running time $n(104\xi(sn+1)+26s\rho)$ of Protocol 47 for several values of the security parameter s and of the length $|p|$ of p. The number of cards is $m=52$, the number of players is $n=5$ and time is expressed in seconds. We can know the computational cost of one player if we divide the total cost by 5.

6.3 Security analysis

In Subsection 6.3.1, we give several lemmata that will be used for the subsequent security analysis. In Subsection 6.3.2 we examine the fulfillment of the security requirements stated in Chapter 3.

Table 6.2: Costs of Protocol 48

	Number of messages	Total length of messages	Computational cost
Protocol 48	$(n-1)(2s+1)$	$[p]52s(2n-1)+[r](n-1+s52)$	$sn104\xi + 26s\rho$
Step 1	$s(n-1)$	$s(n-1)104[p]$	$s(104\xi)$
Step 1a			104ξ
Step 1b	$(n-1)$	$(n-1)(104[p])$	ϵ
Step 2	$(n-1)$	$(n-1)[r]$	ϵ
Step 3	$s(n-1)$	$s(52[p]+52[r])$	$s(104\xi(n-1)+26\rho)$
Step 3a	$(1/2)(n-1)$	$(1/2)(n-1)52([p]+[r])$	$(1/2)(104\xi)(n-1)$
Step 3(a)i	$(n-1)$	$(n-1)(52[p]+52[r])$	ϵ
Step 3(a)ii			$(104\xi)(n-1)$
Step 3b	$(1/2)(n-1)$	$(1/2)(52[p]+52[r])$	$(1/2)(104(n-1)+52\rho)$
Step 3(b)i			52ρ
Step 3(b)ii	$(n-1)$	$(n-1)(52[p]+52[r])$	
Step 3(b)iii			$(104\xi)(n-1)$

6.3.1 Supporting lemmata

Card shuffling lemmas

Lemma 3 *If Protocol 48 is used to verify an encrypted deck of cards, a coalition of* $n-1$ *cheating players have a probability of success of* $(1/2)^s$*, where* s *is a security parameter.*

Table 6.3: Costs of the Procedure 32

	Computational cost
Procedure 32	104ξ
Step 1	ϵ
Step 2	$52(2\xi)$
Step 3	ϵ
Step 4	ϵ
Step 5	ϵ

Table 6.4: Costs of Procedure 33

	Computational cost
Procedure 33	104ξ
Step 1	$52(2\xi)$
Step 1a	ϵ
Step 1b	ξ
Step 1c	ξ

Table 6.5: Estimated values (in seconds) for ξ, ρ and the running time of Protocol 47 for several values of s and $|p|$

	$\|p\|$			
	256	512	768	1024
ξ	$1.268e-3$	$7.882e-3$	$24.510e-3$	$55.638e-3$
ρ	$0.012e-3$	$0.027e-3$	$0.05e-3$	$0.087e-3$
s	$\|p\|$			
	256	512	768	1024
5	17.151	106.582	331.407	752.282
8	27.046	168.072	522.605	1186.292
10	33.642	209.0657	650.070	1475.633
15	50.135	311.549	968.7327	2198.983

Proof: Protocol 48 is used to verify that C_i is the encryption of C_{i-1} without revealing anything about C_i. We will show that, at the end of the protocol, there is a probability $(1/2)^s$ that a wrong computation of C_i stays undetected, where s is a security parameter.

In Step 1, C_i is encrypted s times, so s encrypted decks $C_{i,k}$ are obtained. In Step 2, a s-bit long value u is obtained between all players. This value is random if at least one player is honest, and cannot be manipulated by any player. In Step 3, it is verified for all s encrypted decks: i) that C_i has been well encrypted into $C_{i,k}$; or ii) that C_{i-1} has been well encrypted into $C_{i,k}$. These verifications are made using Protocol 48. The u_k-th bit of u is used to choose whether the first or the second verification are made:

- If C_i has been correctly computed from C_{i-1} and $C_{i,k}$ has been correctly computed from C_i, both verifications pass;
- If C_i is not correctly computed from C_{i-1} but $C_{i,k}$ is correctly computed from C_i, Step 3(a)ii of Protocol 48 passes, but Step 3(b)iii of Protocol 48 fails;
- If $C_{i,k}$ has been correctly computed from C_{i-1} but C_i is not correctly computed from C_{i-1}, then $C_{i,k}$ is not correctly computed from C_i. Therefore, Step 3(b)iii of Protocol 48 passes, but Step 3(a)ii of Protocol 48 fails.

In order to cheat by taking a wrong C_i and then computing a suitable $C_{i,k}$, the coalition must guess whether $(C_{i-1}, C_{i,k})$ or $(C_i, C_{i,k})$ will be verified, *i.e.* the value u_k. We have defined this value as random, so the coalition have a probability 1/2 of guessing it for each deck. Since there are s decks, we can conclude that a cheating coalition passes all verifications with probability $(1/2)^s$. □

Definition 8 *The Decisional Diffie-Hellman Problem [Bon98] (DDH) consists of deciding whether four elements (g_1, y_1, g_2, y_2) of a group are a valid Diffie-Hellman tuple, namely, whether they satisfy* $\log_{g_1} y_1 = log_{g_2} y_2$.

The DDH problem is believed to be hard over the multiplicative subgroup G of Z_p^* of size q generated by α (see [Can02]).

Lemma 4 *Given the encryption of $c_{0,j}$ by $\mathcal{P}_1$ as $c_{1,j} = (d_{1,j}, \alpha_{1,j})$ where $d_{1,j} = \alpha^{x_j r_{1,j}}$ and $\alpha_{1,j} = \beta^{r_{1,j}}$, checking whether x_j is the cleartext encrypted in $c_{1,j}$ without knowing $r_{1,j}$ is as hard as solving the DDH problem in G.*

Proof: Given x_j,$d_{1,j}$,$\alpha_{1,j}$, β, the problem of checking whether the cleartext is x_j is to decide whether $\log_\alpha d_{1,j} = \log_\beta(\alpha_{1,j}) \cdot x_j$. Assuming there exists a polynomial-time algorithm $\mathcal{A}(x_j, \alpha, d_{1,j}, \beta, \alpha_{1,j})$ solving the above problem, an instance (g_1, y_1, g_2, y_2) of the DDH problem in G could be solved in polynomial time taking a random x and running $\mathcal{A}(x, g_1, y_1^x, g_2, y_2)$. □

Lemma 5 *Given two encrypted and permuted cards $c_{i-1,j} \in C_{i-1}$ and $c_{i,j} \in C_i$, deciding whether $c_{i,j}$ is the encrypted version of $c_{i-1,j}$ without knowing $r_{i,\pi(j)}$ is as hard as solving the DDH problem in G.*

Proof: An encrypted and permuted card $c_{i,j}$ in C_i has the form $c_{i,j} = (d_{i,j}, \alpha_{i,j}) = ((d_{i-1,\pi(j)})^{r_{i,\pi(j)}i}, (\alpha_{i-1,\pi(j)})^{r_{i,\pi(j)}})$, where $r_{i,\pi(j)} \in R_i$ and $c_{i-1,\pi(j)} = (d_{i-1,\pi(j)}, \alpha_{i-1,\pi(j)}) \in C_{i-1}$.

Without knowledge of π nor $r_{i,\pi(j)}$, deciding whether $c_{i,j}$ corresponds to an encryption of $c_{i-1,j}$ is equivalent to deciding whether $\log_{d_{i-1,\pi(j)}} d_{i,j} = \log_{\alpha_{i-1,\pi(j)}} \alpha_{i,j}$. This is an instance of the DDH problem in G. □

Card drawing lemma

Lemma 6 *If a coalition of $n-1$ players incorrectly decrypt in Protocol 49 the card drawn by the remaining player, this is detected by that player.*

Proof: At each stage of the protocol, a player $\mathcal{P}_i$ receives e_{r-1}, and then computes and sends $e_r = (e_{r-1})^{K_i^{-1}}$. Let us assume that the coalition of $n-1$ players send $e_r^{'} \neq (e_{r-1})^{K_i^{-1}}$. In Step 3c of Protocol 49, they must prove that $\log_\alpha \beta_i = \log_{e_r^{'}} e_{r-1}$ using the Chaum-Pedersen protocol [CP92]. The proof will fail if $e_r^{'} \neq (e_{r-1})^{K_i^{-1}}$. □

6.3.2 Fulfillment of security requirements

We now use the above lemmata to assess the fulfillment of the properties enumerated in Chapter 3:

Uniqueness of cards: This property is fulfilled by the zero-knowledge-proof technique used in Protocol 48. The probability that card non-uniqueness goes undetected is the probability that the player who runs the protocol guesses a random s-bit number, that is, $(1/2)^s$ (see Lemma 3).

Uniform random distribution of cards: Every player encrypts and permutes the cards. If at least one player is honest and chooses a truly random permutation, the deck of cards will be uniformly permuted.

Absence of trusted third party: No TTP is used in the described protocol suite. All players have the same influence.

Cheating detection with a very high probability: Cheating during card shuffling is detected using a zero-knowledge proof with a probability $1-(1/2)^s$ (see Lemma 3). From Lemma 6, a player $\mathcal{P}_u$ drawing a card with Protocol 49 cannot cheat some honest player $\mathcal{P}_i$ with whom $\mathcal{P}_u$ is interacting by replacing the chosen card with another one or using a key $K' \neq K_i$ (K_i is player $\mathcal{P}_i$'s private key).

Complete confidentiality of cards: As long as there is at least one honest player $\mathcal{P}_i$ who keeps π_i secret (π_i is the permutation used by $\mathcal{P}_i$ in her unopened deck), it is not possible for an intruder to discover the correspondence between the set of cryptograms $c_{n,j} \in C_n$ and the set of cleartext cards x_j. The other approach to breaking card confidentiality is to try to decrypt the cryptograms. It follows from Lemma 4 that α raised to values in X cannot be linked to the elements in C_1 by players $\mathcal{P}_2, \cdots, \mathcal{P}_n$. On the other hand, it follows from Lemma 5 that the elements in C_{i-1} cannot be linked to those in C_i by players other than $\mathcal{P}_i$. In this way, after running the card shuffling protocol, nobody can link α raised to values in X to those in C_n.

Minimal effect of coalitions As pointed out in [Cré85], a cryptographic protocol cannot avoid player coalitions, but it must try to minimize their effect. In practice, this means that a coalition of players should not be able to discover the cards of an honest player. By Lemma 6, a coalition of $n-1$ players cannot change the card that is being drawn, or avoid using their private keys in the protocol. If they use other keys, this is detected. Finally, it follows from Lemmas 4 and 5 that it is hard to know the cleartext value w of an encrypted card.

Complete confidentiality of strategy The protocol does not require players to reveal their secret keys K_i nor their permutations π_i in order to verify the game.

6.4 Conclusions

We have presented a mental poker protocol which does not rely on trusted third parties, while preserving confidentiality of player strategy. Our proposal uses zero knowledge proofs for shuffling and drawing cards, but the proofs used are well-known and efficient. Thus, unlike for some previous TTP-free schemes, the amount of computation of the protocol suite presented here stays affordable and, as shown, can been accurately quantified.

Chapter 7

Dropout-tolerant TTP-free mental poker

The properties that, according to [Cré85], should be satistied by any mental protocol have been recalled in Chapter 3.

Dropout tolerance was not listed in [Cré85] as a requirement, but nonetheless stays a major challenge in remote gaming. In electronic gaming, no one can prevent a player from quitting a game. Two kinds of dropout can be distinguished:

- **Intentional:** A player decides to quit the game. This may be attractive for a player to whom the game is not being favorable.
- **Accidental:** A player cannot go on playing, for example due to a network problem.

Whatever the reason for player dropout, the remaining players should be able to continue the game. If a Trusted Third Party (TTP) is controlling the game, handling player dropout is greatly simplified [CSD04a]. However, a TTP is not always available or desirable: it may not be trusted by everybody, it may charge some fee, etc. When no TTP is assumed, dropout becomes a nontrivial problem.

In this chapter, we proposes a solution for player dropout in mental poker without a TTP [CSD05]. The solution is based on zero-knowledge proofs and allows the game

to continue after the dropout.

Section 7.1 reviews literature on TTP-free mental poker offering player confidentiality, and it analyzes what happens when one player leaves the hand during the game. Our proposed protocol is described in Section 7.2. Security is examined in Section 7.3. Finally, Section 7.4 is a conclusion.

7.1 Background on TTP-free mental poker offering player confidentiality

All schemes mentioned in this section fulfill all security requirements identified in [Cré85], including the confidentiality of player strategy. We next review them by focusing on their ability to handle player dropout.

Schemes [Cré86, Sch98] do not consider player dropout. In both proposals, each player has some secret information needed to draw cards from the deck. Without this information, the game cannot proceed.

In [BS03] it is proposed that players who quit the the game should disclose their secret information. However, this solution is only applicable if dropout is intentional *and* the player leaving the game is willing to collaborate. In case of accidental dropout (*e.g.* due to a network problem) or malicious intentional dropout, there is no guarantee that the remaining players can go on playing.

Schemes [KKO97, SSG02] represent each card in the deck by a different numerical value. During card shuffling, those values are encrypted and permuted by each player. The effect of encryption is analogous to reversing cards in a physical deck. A secret-sharing scheme is used, so that at least t players are needed to decrypt values. The goal is that the game can proceed if at least t players remain, which allows for some dropouts. In [KKO97] the secret sharing scheme is applied to cards. Each value representing a card is divided into as many shares as there are players. Then each share is encrypted under the public key of a different player. A card cannot be decrypted unless at least t players co-operate. In [SSG02], players create a key pair using the procedure proposed in [Ped92]. Players generate a public key so that each

player gets a share of the private key; thus, the private key cannot be used unless at least t players co-operate. Even if those schemes based on secret sharing do offer some dropout tolerance, the bad news is that secret sharing makes it possible for a sufficiently large collusion of players to obtain all deck information. Thus, dropout tolerance is traded off against collusion tolerance. This is frustrating because collusion tolerance is a basic security property already identified as relevant in [Cré85].

7.2 Our proposal

There is a first round where cards are dealt as in the poker game, and each player obtains five cards. If a player discards some cards from her hand, a new dealing round is started so that the player can obtain as many cards as she has discarded.

We use Protocol 51 to obtain a new deck of cards in each dealing round, in a similar way as proposed in [Yun85]. In the second and successive dealing rounds each player *vetoes* (*i.e.* marks as unavailable) those cards that she has previously drawn. Protocol 52 is used to veto drawn cards. If a player obtains a vetoed card, she cannot use it and she does not know either the value of the vetoed card or who vetoed it; what the player can do is to show that she obtained a vetoed card and then draw a new card.

If a player leaves the game, the rest of players generate a new deck and use it in the game. The new deck includes the cards that were drawn by the player who left the game, because the latter is no longer there to veto her cards when the new deck is generated.

We shall use the following notation in the subsequent protocols.

n : number of players (we assume some ordering among the n players);

$\mathcal{P}_i$: the i-th player in the ordered set of n players;

λ_i : set of cards in $\mathcal{P}_i$'s hand;

Λ : set of all cards in the hands of all players, *i.e.* $\Lambda = \cup_{i=1}^{n} \lambda_i$;

δ_i : set of cards discarded by $\mathcal{P}_i$.

7.2.1 System set-up

Before a game starts, players $\mathcal{P}_1, \ldots, \mathcal{P}_n$ must set some parameters. They choose a large prime p so that $p = 2q+1$ and q is also prime; they also pick one element $g \in \mathbb{Z}_p^*$ of order q.

Using the key generation protocol described in [DF90], players jointly generate a public key $y = \prod_{i=1}^{n} y_i$. Each player $\mathcal{P}_i$ keeps her corresponding share α_i of the private key and publishes $y_i = g^{\alpha_i}$.

7.2.2 Deck generation

Each card is represented by a value jointly computed by all players in Protocol 51. We first explain what Protocol 51 does and then describe the protocol in detail.

Let us assume that we are in the k-th dealing round. We can see in Step 1 of Procotol 51 below that every player uses Procedure 34 to compute 52 new values. These values are sent to the rest of players in Step 2 of Protocol 51. Once every player gets the new values from the rest of players, the new deck D_k is computed by all players at Step 3. We use the term face-up deck of cards because every player can see the value of each card; the j-th value $d_{k,j}$ in D_k represents the j-th card in the deck.

If $d_{k,j}$ is a face-up card, then we denote by $e_{k,j}$ the corresponding face-down card. $e_{k,j}$ contains the encrypted version of the exponents that have been used to compute $d_{k,j} \in D_k$ from $d_{k-1,j} \in D_{k-1}$. To prove ownership of a card $d_{k,j}$, a player must prove knowledge of those exponents, *i.e.* prove knowledge of the discrete logarithm $\log_{d_{k-1,j}}(d_{k,j})$.

In the first round, all cards are available and $E_1 = C_{1,0}$ is the face-down deck of cards without shuffling (see Step 4 of Protocol 51). In subsequent rounds, each player vetoes the cards in her hand using Protocol 52 (called at Step 5a of Protocol 51); the goal is that cards already drawn should become unavailable. After using Protocol 52, a player gets one re-masking factor for each card in the deck; a vetoed card is re-masked with a factor which does not allow decryption, whereas a non-vetoed card is re-masked with a factor allowing decryption.

In Step 5b players re-mask the encrypted exponents with these factors, and obtain the face-down deck of cards without shuffling, $C_{k,0}$.

We denote by D_l the deck of cards of the l-th round; we denote by $\mathbf{D}$ the set of all decks that have been generated in all rounds, *i.e.* $\mathbf{D} = \{D_1, \ldots, D_k\}$. In order to run our protocol, we define $D_0 = \{d_{0,1}, \ldots, d_{0,52}\}$, where $d_{0,j} = g$, $\forall j \in \{1, \ldots, 52\}$.

Protocol 51 $(k \geq 1, D_{k-1})$

1. *Each player $\mathcal{P}_i$ uses Procedure 34 on D_{k-1} and obtains $D_{k,i} = \{d_{k,i,1}, \ldots, d_{k,i,52}\}$ and $E_{k,i} = \{e_{k,i,1}, \ldots, e_{k,i,52}\}$, where $d_{k,i,j} = d_{k-1,j}^{m_{k,i,j}}$ and $e_{k,i,j} = E_y(m_{k,i,j})$;*

2. *Each $\mathcal{P}_i$ publishes $D_{k,i}$ and $E_{k,i}$;*

3. *All players compute the face-up deck of cards $D_k = \{d_{k,1}, \ldots, d_{k,52}\}$ and $E_k = \{e_{k,1}, \ldots, e_{k,52}\}$, where $d_{k,j} = \prod_{i=1}^{n} d_{k,i,j} = d_{k-1,j}^{m_{k,1,j}+\cdots+m_{k,n,j}}$ and $e_{k,j} = \{e_{k,1,j}, \ldots, e_{k,n,j}\}$;*

4. *If $k = 1$, players compute the face-down deck of cards $C_{1,0} = \{c_{1,0,1}, \ldots, c_{1,0,52}\}$, where $c_{1,0,j} = e_{1,j} \in E_1$;*

5. *If $k > 1$ then players do the following*

   (a) *Run the vetoing protocol (Protocol 52) and obtain $G_k = \{g_{k,1}, \ldots, g_{k,52}\}$, where $g_{k,j} = \{g_{k,1,j}, \ldots, g_{k,n,j}\}$;*

   (b) *Compute the face-down deck of cards $C_{k,0} = \{c_{k,0,1}, \ldots, c_{k,0,52}\}$, where $c_{k,0,j} = e_{k,j} \cdot g_{k,j} = \{e_{k,1,j} \cdot g_{k,1,j}, \ldots, e_{k,n,j} \cdot g_{k,n,j}\}$. Drawn cards in this face-down deck have been vetoed.*

In the k-th dealing round, each player $\mathcal{P}_i$ computes a new value $d_{k,i,j}$ for each card j. This new value is obtained from $d_{k-1,j}$ (the value used in round $k-1$ to represent card j) raised to a random value $m_{k,i,j}$. The exponent $m_{k,i,j}$ is encrypted into $E_y(m_{k,i,j})$ and sent along with the new value. Players use Procedure 34 to compute these new values for each card.

Procedure 34 (y, p, D)

1. *For each* d_j *in* $D = \{d_1, \ldots, d_{52}\}$ *do:*

   (a) *generate a random value* m_j*, where* $2 < m_j < q$*;*

   (b) *compute* $d_j^{m_j}$*;*

   (c) *encrypt* m_j *into* $E_y(m_j)$ *under public key* y*;*

   (d) *prove in zero-knowledge to the rest of players that* $E_y(m_j)$ *is the encryption of* $\log_{d_j}(d_j^{m_j})$ *using [Sta96];*

2. *Return the sets* $D' = \{d_1^{m_1}, \ldots, d_{52}^{m_{52}}\}$ *and* $E = \{E_y(m_1), \ldots, E_y(m_{52})\}$.

Prior to describing Protocol 52 we define $\xi_{l,i}$ as the number of cards that $\mathcal{P}_i$ has drawn in the l-th dealing round; we also define ξ_i as the sum of all cards drawn by $\mathcal{P}_i$ in all previous dealing rounds, that is, $\xi_i = \sum_{l=1}^{k-1} \xi_{i,l}$.

In Step 1a of Protocol 52 each player in turn computes a re-masking factor for each card $d_{k,j} \in D_k$. Using the construction of [CDS94], $\mathcal{P}_i$ proves in Step 1b of Protocol 52 that $52 - \xi_i$ factors have been properly computed (as many factors as the number of cards $\mathcal{P}_i$ has not drawn); in this proof, $\mathcal{P}_i$ does not reveal which subset of factors was properly computed. In Step 1c, $\mathcal{P}_i$ again uses the construction of [CDS94] to prove that she has computed ξ_i re-masking factors which veto the cards $\mathcal{P}_i$ has drawn (see in Section 7.3 the lemma that a player vetoes the cards she has drawn). The re-masking factors that veto all drawn cards by any player are pooled together in Step 2.

Protocol 52 (D_k)

1. *For each* $\mathcal{P}_i$ $(i = 1, \ldots, n)$*:*

   (a) $\mathcal{P}_i$ *uses Procedure 35 with* $(D_k, \lambda_i, \delta_i, \mathbf{D})$ *and obtains* $G_i = \{(u_{i,1}, v_{i,1}), \ldots, (u_{i,52}, v_{i,52})\}$*;*

   (b) $\mathcal{P}_i$ *proves in zero-knowledge that at least* $52 - \xi_i$ *values* $(u_{i,j}, v_{i,j})$ *properly re-mask a card,* i.e. *they do not veto the card. This is done using the construction of [CDS94] in order to show that* $\mathcal{P}_i$ *can correctly perform at least* $52 - \xi_i$ *executions of the set of zero-knowledge proofs*

$\{CP_{i,1}, \ldots, CP_{i,52}\}$, *where* $CP_{i,j} = CP(g, y, u_{i,j}, v_{i,j})$ *(*$CP(g, y, u, v)$ *denotes the Chaum-Pedersen [CP92] zero-knowledge proof, see section 2.2.2).*

(c) *For* $l = 1$ *to* k, $\mathcal{P}_i$ *proves in zero-knowledge that she has vetoed as many cards as the number* $\xi_{l,i}$ *of cards she obtained in the* l*-th dealing round. This is done using the construction of [CDS94] in order to prove that she can perform at least* $\xi_{l,i}$ *executions among the following set of zero-knowledge proofs* $\{CP_{l,i,1}, \ldots, CP_{l,i,52}\}$, *where* $CP_{l,i,j} = CP(d_{l-1}, u_{i,j}, d_{l,j}, d_j)$;

2. *Compute* $G = \{g_1, \ldots, g_{52}\}$, *where* $g_j = (u_j, v_j)$, *and* $u_j = \prod_{i=1}^{n} u_{i,j}$ *and* $v_j = \prod_{i=1}^{n} v_{i,j}$, *with* $(u_{i,j}, v_{i,j}) \in G_i$;

3. *Let* $G_0 = \{(g_{0,1,1}, \ldots, g_{0,n,1}), \ldots, (g_{0,1,52}, \ldots, g_{0,n,52})\} := G$, *that is,* $g_{0,\zeta,j} = g_j$ $\forall \zeta \in \{1, \ldots, n\}$ *and* $\forall j \in \{1, \ldots, 52\}$, *and* $g_j = (u_j, v_j) \in G$;

4. *For each* $\mathcal{P}_i$ $(i = 1, \ldots, n)$:

   (a) *receive* $G_{i-1} = \{(g_{i-1,1,1}, \ldots, g_{i-1,n,1}), \ldots, (g_{i-1,1,52}, \ldots, g_{i-1,n,52})\}$ *from* $\mathcal{P}_{i-1}$;

   (b) *compute* $G_i = \{(g_{i,1,1}, \ldots, g_{i,n,1}), \ldots, (g_{i,1,52}, \ldots, g_{i,n,52})\}$, *where* $g_{i,\zeta,j} = g_{i-1,\zeta,j}^{r_{i,\zeta,j}} = (u_{i-1,\zeta,j}^{r_{i,\zeta,j}}, v_{i-1,\zeta,j}^{r_{i,\zeta,j}})$ *and* $1 < r_{i,\zeta,j} < q$ *is a value obtained at random;*

   (c) *for each* $g_{i,\zeta,j} = (u_{i,\zeta,j}, v_{i,\zeta,j})$ *in* G_i *run* $CP(u_{i-1,\zeta,j}, v_{i-1,\zeta,j}, u_{i,\zeta,j}, v_{i,\zeta,j})$;

   (d) *send* G_i *to the next player;*

5. *Return* G_n.

Procedure 35 is used by every player to compute the values used in re-masking. If card j is not vetoed then a pair (u_j, v_j) is computed such that $\log_g u_j = \log_y v_j$. However, if card j must be vetoed, a pair (u_j, v_j) such that $\log_g u \neq \log_y v$ is computed, in order to prevent a correct decryption.

As will be described in Section 7.2.4, when $\mathcal{P}_i$ obtains a card j at round k, $\mathcal{P}_i$ obtains the discrete logarithm $\tau_{k,j} = \log_{d_{k-1,j}} d_{k,j}$. In subsequent rounds $\mathcal{P}_i$ uses that logarithm to veto this card.

Procedure 35 $(D, \lambda, \delta, \mathbf{D})$

1. *For each* d_j *in* $D = \{d_1, \ldots, d_{52}\}$ *do:*

   (a) *if the card represented by* d_j *is in* $\lambda \cup \delta$ *do:*

      i. *generate a random value* R_j*, where* $1 < R_j < p$*;*

      ii. *let us assume that the card represented by* d_j *has been obtained in round* l*. In this case* $\tau_{l,j} = \log_{d_{l-1,j}}(d_{l,j})$ *is known (where* $d_{l-1,j}$ *and* $d_{l,j}$ *are in* $\mathbf{D}$*), so* $g_j = (u_j, v_j)$ *is computed, where* $u_j = d_j^{\tau_{l,j}^{-1}}$ *and* $v_j = R_j$*;*

   (b) *if the card represented by* d_j *is not in* $\lambda \cup \delta$ *do:*

      i. *generate a random* r_j*, where* $1 < r_j < q$*;*

      ii. *compute* $g_j = (u_j, v_j)$*, where* $u_j = g^{r_j}$ *and* $v_j = y^{r_j}$*;*

2. *Return* $G = \{g_1, \ldots, g_{52}\} = \{(u_1, v_1), \ldots, (u_{52}, v_{52})\}$.

7.2.3 Card shuffling

This is done using the procedure described in [BS03]. The different players in turn shuffle and re-mask the face-down deck C_0 obtained with Protocol 51.

Protocol 53 (C_0)

1. *For each player* $\mathcal{P}_i$ $(i = 1, \ldots, n)$ *do:*

   (a) *Generate a permutation* σ_i *of 52 elements;*

   (b) *Permute the elements* $c_{i-1,j}$ *of the face-down deck* C_{i-1} *with* σ_i *to obtain* C_i^**;*

   (c) *Re-mask the encrypted messages contained in each card of* C_i^* *without modifying their content to obtain* C_i*; this is done by re-masking all ciphertexts contained in each face-down card* $C_i^* = \{c_{i,1}^*, \ldots, c_{i,52}^*\}$*, where* $c_{i,j}^* = \{e_{i,j,1}^*, \ldots, e_{i,j,n}^*\}$*; specifically,* $\mathcal{P}_i$ *computes* $C_i = \{c_{i,1}, \ldots, c_{i,52}\}$*, where* $c_{i,j} = \{e_{i,j,1}^* \cdot E_y(1, r_{i,j,1}), \ldots, e_{i,j,n}^* \cdot E_y(1, r_{i,j,n})\}$*; values* $\{r_{i,j,1}, \ldots, r_{i,j,n}\}$ *are obtained at random;*

   (d) *Use the proof in [BS03] to prove in zero-knowledge that* C_i *is a permuted and re-masked version of* C_{i-1}.

After running the shuffling protocol, players get the set C_n, that is, the shuffled face-down deck of cards.

7.2.4 Card drawing

The card extraction procedure is as follows. Let us assume that extraction is performed by player $\mathcal{P}_i$:

Protocol 54

1. *$\mathcal{P}_i$ randomly selects an element from C_n, namely, $c_j = (e_{j,1}, \ldots, e_{j,n})$;*
2. *$\mathcal{P}_i$ asks the rest of players to verifiably send her information to decrypt the messages $e_{j,\zeta}$ contained in c_j using [DF90];*
3. *After decrypting these messages, $\mathcal{P}_i$ obtains $\{m_1, \ldots, m_n\}$, where $m_\zeta = D(e_{j,\zeta})$;*
4. *$\mathcal{P}_i$ searches for an element $d_{k,t} \in D_k$ (the k-th dealing round deck) such that $d_{k-1,t}^{m_1+\cdots+m_n} \equiv d_{k,t}$;*
5. *If $d_{k,t} \in D_k$ then $\mathcal{P}_i$ stores $\tau_t = (m_1 + \cdots + m_n)$;*
6. *If $d_{k,t} \notin D_k$ then $\mathcal{P}_i$ has obtained a vetoed card. In this case, she shows that the card was vetoed and requests a new one.*

7.2.5 Card opening

A player must prove to the rest of players that she is the owner of her cards. The card opening protocol is used to that end.

Let us assume that a player $\mathcal{P}_i$ has drawn a card $c_j \in C_n$. $\mathcal{P}_i$ has received the partial decryption from the rest of players, and she has verified that each partial decryption is correct.

$\mathcal{P}_i$ opens a card when she publishes the remaining part of the decryption of c_j. With this information, all players can know the value of c_j, that is, the decrypted card exponents $D(e_{j,1}), \cdots, D(e_{j,n})$. The decryption is verifiably performed as detailed in [DF90].

7.2.6 Card discarding

A player discards a card when she commits herself to not using it. Let us assume that player $\mathcal{P}_i$ has drawn a card $c_j \in C_n$ using Protocol 54.

$\mathcal{P}_i$ discards c_j by sending a message *discard* with c_j to the rest of players. c_j is added to the set λ_i. If $\mathcal{P}_i$ wants to open a discarded card, the rest of players can detect the cheating because the card is in λ_i.

7.2.7 Player dropout

In case one of the players leaves the game, the rest of players can go on playing. Assuming that player $\mathcal{P}_i$ with public key y_i leaves the game, the game public key is updated as $y := y/y_i$. Next, the rest of players continue as if player $\mathcal{P}_i$ had never joined the game. This implies that cards once extracted by $\mathcal{P}_i$ will be back in the deck.

7.3 Security

Security results in this section basically state that: i) vetoed cards cannot be opened; and ii) the set of vetoed cards is the set of drawn cards. Each security result is followed by the respective Proofs.

Lemma 7 *If $\mathcal{P}_i$ succeeds in performing $CP_{i,j} = CP(g, y, u_{i,j}, v_{i,j})$ at Step 1b of Protocol 52, then $(u_{i,j}, v_{i,j})$ will not veto the face-down card $c_{k,0,j}$ at Step 5b of Protocol 51.*

Proof: A face-down card $c_{k,0,j}$ is formed by a set of ciphertexts $(e_{k,1,j}, \cdots, e_{k,n,j})$. When a card is *not* vetoed in Protocol 52, a re-masking factor $(u_{i,j}, v_{i,j})$ is used which still allows recovery of the cleartexts from the card ciphertexts. To allow correct decryption, the re-masking factor must satisfy $\log_g u_{i,j} = \log_y v_{i,j}$. This is exactly the property proven by $CP(g, y, u_{i,j}, v_{i,j})$. □

The Corollary below follows from the above lemma and from Step 1b of Protocol 52.

Corollary: 1 *Let* 52 *be the total number of cards. Let* ξ_i *be the cards drawn by a player* $\mathcal{P}_i$*. Then the number of cards* x_i *not vetoed by* $\mathcal{P}_i$ *is such that* $x_i \geq 52 - \xi_i$*.*

Lemma 8 *If, at round* k*,* $\mathcal{P}_i$ *succeeds in performing* $CP_{l,i,j} = CP(d_{l-1}, u_{i,j}, d_{l,j}, d_j)$*,* $l < k$*, at Step 1c of Protocol 52, then* $(u_{i,j}, v_{i,j})$ *vetoes the face-down card* $c_{k,0,j}$ *at Step 5b of Protocol 51.*

Proof: Let us assume a card drawn in a dealing round l previous to the current round k (*i.e.* $l < k$). Let the face-up value of that card be $d_{k,j}$. $CP(d_{l-1}, u_{i,j}, d_{l,j}, d_{k,j})$ proves that:

$$\tau = \log_{d_{l-1,j}}(d_{l,j}) = \log_{u_{i,j}} d_{k,j} \tag{7.1}$$

From Equation 7.1 we have

$$u_{i,j} = (d_{k,j})^{\tau^{-1}} \tag{7.2}$$

Now if $(u_{i,j}, v_{i,j})$ does not actually veto $d_{k,j}$, the following holds:

$$\log_g(u_{i,j}) = \log_y(v_{i,j}) = \log_{g^\alpha}(v_{i,j}) = \frac{1}{\alpha}\log_g(v_{i,j})$$

The above is equivalent to

$$\alpha \cdot \log_g(u_{i,j}) = \log_g(v_{i,j}) \tag{7.3}$$

Combining Equations (7.2) and (7.3) yields

$$\log_g((d_{k,j})^{\tau^{-1}})^\alpha = \log_g v_{i,j} \tag{7.4}$$

If logarithms are removed, we get $v_{i,j} = ((d_{k,j})^{\tau^{-1}})^\alpha$. Thus, re-masking factor $(u_{i,j}, v_{i,j})$ will pass $CP(d_{l-1}, u_{i,j}, d_{l,j}, d_{k,j})$ without actually vetoing $d_{k,j}$ only if it has the form

$$(u_{i,j}, v_{i,j}) = (d_{k,j}^{\tau^{-1}}, ((d_{k,j})^{\tau^{-1}})^\alpha) \tag{7.5}$$

However, computing $v_{i,j}$ in Expression (7.5) without knowledge of α nor $(d_{k,j})^\alpha$ is as hard as the Diffie-Hellman problem. Obtaining α from the public key y is as hard as the discrete logarithm problem. Thus, passing the verification at Step 1c of Protocol 52 implies that card $d_{k,j}$ is actually vetoed. □

Lemma 9 *A player can only veto a card she has drawn.*

Proof: Assume that the current dealing round is round k;
let $\{d_{1,j}, \cdots, d_{l,j}, \cdots, d_{k,j}\}$ be the expressions for the j-th card at each dealing round l, where $\tau_{l,j} = \log_{d_{l-1,j}} d_{l,j}$ for $1 \leq l < k$. Assume now that $\mathcal{P}_i$ wants to construct a proof of veto for $d_{t,j}$, for some $t < k$. Then $\mathcal{P}_i$ needs to construct $(u_{i,j}, v_{i,j})$ so that she can perform $CP_{t,i,j} = CP(d_{t-1}, u_{i,j}, d_{t,j}, d_{k,j})$. This means $\log_{d_{t-1,j}} d_{t,j} = \log_{u_{i,j}} d_{k,j}$, which requires $u_{i,j} = d_{k,j}^{\tau_{t,j}^{-1}}$ so that $\mathcal{P}_i$ needs to know $\tau_{t,j} = \log_{d_{t-1,j}}(d_{t,j})$ But this logarithm is only known to $\mathcal{P}_i$ if she drew the card at round t (see Protocol 54). □

Lemma 10 *The number of cards vetoed by a player is at least the number of cards drawn by the player.*

Proof: Let us assume that $\mathcal{P}_i$ has extracted ξ_i cards in previous dealing rounds. At Step 1c of Protocol 52 $\mathcal{P}_i$ uses the proof by Cramer *et al.* [CDS94] for the ξ_i re-masking factors corresponding to the ξ_i drawn cards. According to Lemma 8, this guarantees that the ξ_i drawn cards are vetoed. □

Theorem 2 *The set of cards vetoed by a player is the same as the set of cards drawn by the player.*

Proof: If a re-masking factor $(u_{i,j}, v_{i,j})$ passes the proof that it is a vetoing factor for card j, then by Lemma 8 it vetoes card j. On the other hand, if a re-masking factor $(u_{i,j}, v_{i,j})$ passes the proof that it is a non-vetoing factor for card j, then by Lemma 7, it does not veto card j. Now, a re-masking factor $(u_{i,j}, v_{i,j})$ cannot at the same time veto and not veto card j. Thus, $(u_{i,j}, v_{i,j})$ cannot pass both the proof that it is a vetoing factor and the proof that it is a non-vetoing factor.

Let us assume that player $\mathcal{P}_i$ has drawn ξ_i cards. By Lemma 1, the number of cards not vetoed by $\mathcal{P}_i$ is at least $52 - \xi_i$. By Lemma 10, the number of cards vetoed by $\mathcal{P}_i$ is at least ξ_i. Therefore, $\mathcal{P}_i$ vetoes *exactly* ξ_i cards. Finally, by Lemma 9, a player can only veto cards she has drawn. Therefore the set of drawn cards is the same as the set of vetoed cards. □

Theorem 3 *A vetoed card cannot be opened.*

Proof: Without loss of generality we assume $n = 2$ players $\mathcal{P}_1$ and $\mathcal{P}_2$. Assume that a round k card $d_{k,j}$ is computed from $d_{k-1,j}$ by raising a round $k-1$ card $d_{k-1,j}$ to exponents m_1 and m_2, *i.e.* $d_{k,j} = d_{k-1,j}^{m_1+m_2}$. Note that m_i is secret and only known to $\mathcal{P}_i$, for $i = 1, 2$, whereas $d_{k,j}$ is public and obtained at Step 3 of Protocol 51.

At Step 5a of Protocol 51 the vetoing protocol is called to compute re-masking factors $g_{k,1,j} = (u_1, v_1)$ and $g_{k,2,j} = (u_2, v_2)$; at Step 5b these factors are applied to the encrypted card exponents $e_{k,j} = (e_{k,1,j}, e_{k,2,j})$ to veto card $d_{k,j}$. Now, (u_1, v_1) and (u_2, v_2) have been computed by the vetoing protocol (Protocol 52), so they satisfy

$$\log_g u_i \neq \log_y v_i \text{ for } i = 1, 2 \tag{7.6}$$

The computations for vetoing $d_{k,j}$ at Step 5b of Protocol 51 are:

$$e_{k,j} \cdot g_{k,j} = \{e_{k,1,j} \cdot g_{k,1,j}, e_{k,2,j} \cdot g_{k,2,j}\} = \{E_y(m_1) \cdot (u_1, v_1), E_y(m_2) \cdot (u_2, v_2)\}$$

$$= \{(g^{r_1} \cdot u_1, m_1 \cdot y^{r_1} \cdot v_1), (g^{r_2} \cdot u_2, m_2 \cdot y^{r_2} \cdot v_2)\} \tag{7.7}$$

Opening card $d_{k,j}$ means extracting the secret exponents m_1 and m_2 from the face-down card expression $e_{k,j} \cdot g_{k,j}$. From Expression (7.7), we have that

$$m_1 = \frac{m_1 \cdot y^{r_1} \cdot v_1}{(g^{r_1} \cdot u_1)^\alpha} \quad , \quad m_2 = \frac{m_2 \cdot y^{r_2} \cdot v_2}{(g^{r_2} \cdot u_2)^\alpha} \tag{7.8}$$

Some algebraic manipulation of Equations (7.8) leads to

$$\log_g u_1 = \log_y v_1 \quad , \quad \log_g u_2 = \log_y v_2 \tag{7.9}$$

Equations (7.9) contradict Equations (7.6). Thus, the card cannot be opened. □

7.4 Conclusions

We have presented a mental poker protocol which, to our best knowledge, is the first TTP-free proposal tolerating both intentional and accidental player dropout. Future research will explore applications of the protocol in this paper to secure multi-party computation problems other than mental poker.

Chapter 8

Conclusions

In this thesis, we have focused on the study and design of cryptographic protocols that offer suitable security when poker is played over a computer network. This kind of protocols are collectively known as *mental poker*.

8.1 Results of this thesis

Our first ambition has been to do a comparative analysis of the main contributions to mental poker. We have divided mental poker protocols in two groups: TTP based and TTP-free.

Using the presence or absence of a TTP as the main classification criterion is justified because the assumption of a TTP dramatically diminishes the complexity of the mental poker problem and the computational cost of the solutions. Thus, TTP-based and TTP-free protocols cannot be compared and must stand in different groups.

We have examined the following items for each protocol:

- Functionality
- Security
- Computational and communication cost

Each protocol offers some functionality with a degree of security at a certain cost. This cost is related to the mathematical operations performed and the messages exchanged. We can compare two solutions only if they offer the same functionality and security.

To the best of our knowledge, in Chapter 3 we have presented the first exhaustive comparison of the main contributions to mental poker. Being novel, such an analysis is a contribution in its own right.

In Chapter 4 we have presented an attack that exploits a security flaw in the mental poker protocol by Zhao *et al.*. We found this weakness when carrying out the comparative survey presented in Chapter 3. With little computation, a player can decrypt the encrypted cryptograms and find the cleartext cards. As a reaction to our attack, the same authors of the broken protocol have presented a modified version of their initial proposal. Nevertheless, the new version still has an important security flaw, which we discuss in the same chapter: the proposed encryption and decryption system allows one of the players to find the cleartexts of the final encrypted shuffled deck of cards.

In Chapter 5 we have presented a new mental poker protocol that falls in the category of TTP-free protocols that do not preserve the confidentiality of player strategies. The computational cost is reduced by avoiding the use of zero-knowledge proofs. The method used to represent a card and a permutation allows an encrypted card to be permuted using an additive and multiplicative homomorphic cryptosystem. All rights on this protocol are owned by Scytl Online World Security S.A. through a patent. Moreover, the protocol has been implemented in a case study of mutual distrust. The authors of the implementation argue that our proposal is "practical in terms of computational requirements" in comparison to the rest of proposals in the literature.

In Chapter 6 we have presented a new TTP-free mental poker protocol which preserves the confidentiality of the strategy of players. We remark that the amount of computation required stays reasonably low. We have presented a cost analysis and we have compared it with one of the most efficient previous proposals. The security of this new protocol has been analyzed: we conclude that it fulfills all security properties

normally required in a mental poker protocol.

In Chapter 7 we have presented a new TTP-free protocol which allows the game to continue after player dropout without diminishing player security. This is an unusual functionality. Our solution is based on zero-knowledge proofs, while previous dropout-tolerant proposals were based on secret sharing (cards were encrypted with a shared secret key and, if some player left the game, the rest of players could still decrypt the remaining encrypted cards). One drawback of using secret sharing is that a sufficiently large collusion of players can obtain all cleartext cards; another drawback is that the number of dropouts that can be tolerated is fixed. Our new dropout-tolerant solution does not allow any player coalition to find out the cards in the hands of the rest of players. Moreover, the number of players that can leave the game is not limited.

8.2 Future research

Although the computational and communication costs have been reduced by our three new proposals with respect to the state of the art, further reductions are needed to come up with commercial mental poker products. Our future research will be directed to cost reduction while maintaining the same levels of security and functionality.

Player dropout opens a new research problem in mental poker. The efficiency of our proposal must be increased. Our future research here will be directed to improving our dropout protocol.

Finally, mental poker is a special case of secure multiparty computation. Thus, an interesting research line is to explore applications of mental poker protocols to other secure multiparty computation problems.

Bibliography

[Ask05] A. Askarov. Secure implementation of cryptographic protocols: A case study of mutual distrust. Master's thesis, Chalmers University of Technology, April 2005.

[BC90] G. Brassard and C. Crépeau. Sorting out zero-knowledge. In *Advances in Cryptology –Eurocrypt '89*, volume 434 of *Lecture Notes in Computing Science*, pages 181–191. Springer-Verlag, 1990.

[BCPQ01] E. Bresson, O. Chevassut, D. Pointcheval, and J.-J. Quisquater. Provably authenticated group diffie-hellman key exchange. In *Proc. of ACM-CCS'01*, pages 255–264, Philadelphia PA, 2001. ACM Press.

[BCR86] G. Brassard, C. Crépeau, and J.M. Robert. All-or-nothing disclosure of secrets. In *Advances in Cryptology Crypto'86*, volume 263 of *Lecture Notes in Computer Science*, pages 234–238. Springer-Verlag, 1986.

[BF83] I. Barany and Z. Furedi. Mental poker with three or more players. Technical report, Mathematical Institute of the Hungarian Academy of Sciences, 1983.

[BG85] M. Blum and S. Goldwasser. An efficient probabilistic public-key encryption scheme which hides all partial information. In G.R. Blakley and D. Chaum, editors, *Advances in Cryptology: Proc of Crypto 84*, volume 196 of *Lecture Notes in Computer Science*, pages 289–299. Springer-Verlag, Berlin, 1985.

[Bon98] D. Boneh. The decisional diffie-hellman problem. In *Third Algorithmic Number Theory Symposium*, volume 1423 of *Lecture Notes in Computer Science*. Springer-Verlag, 1998.

[BS03] A. Barnett and N. Smart. Mental poker revisited. In *Proc. Cryptography and Coding*, volume 2898 of *Lecture Notes in Computer Science*, pages 370–383. Springer-Verlag, December 2003.

[Can02] R. Canetti. The decisional diffie-hellman assumption. entry for the *Encyclopaedia of Information Security*, Dec. 18 2002. `http://www.cs.dartmouth.edu/žyle/papers/DDHentry.pdf`.

[CD04] J. Castellà-Roca and J. Domingo-Ferrer. On the security of an efficient ttp-free mental poker protocol. In *International Conference on Information Technology ITCC 2004*, volume II, pages 781–784, Las Vegas, NV, April 2004. IEEE Computer Society.

[CDRB03] J Castellà-Roca, J. Domingo-Ferrer, A. Riera, and J. Borrell. Practical mental poker without a ttp based on homomorphic encryption. In T. Johansson and S. Maitra, editors, *Progress in Cryptology-Indocrypt'2003*, number 2904 in Lecture Notes in Computer Science, pages 280–294. Berlin: Springer-Verlag, 2003.

[CDS94] R. Cramer, I. Damgård, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *Advances in Cryptology – CRYPTO'94*, volume 839 of *Lecture Notes in Computer Science*, pages 174–187. Springer-Verlag, 1994.

[CP92] D. Chaum and T. Pedersen. Wallet databases with observers. In *Advances in Cryptology – CRYPTO'92*, volume 740 of *Lecture Notes in Computer Science*, pages 89–105. Springer-Verlag, 1992.

[CRBD02] J. Castellà-Roca, A. Riera-Jorba, J. Borrell-Viader, and J. Domingo-Ferrer. A method for obtaining an impartial result in a game over a

communications network and related protocols and programs. international patent application PCT ES02/00485, Oct. 14 2002.

[Cré85] C. Crépeau. A secure poker protocol that minimizes the effect of player coalitions. In Hugh C. Williams, editor, *Advances in Cryptology - Crypto '85*, volume 218 of *Lecture Notes in Computer Science*, pages 73–86, Berlin, 1985. Springer-Verlag.

[Cré86] C. Crépeau. A zero-knowledge poker protocol that achieves confidentiality of the players' strategy or how to achieve an electronic poker face. In A. M. Odlyzko, editor, *Advances in Cryptology - Crypto '86*, volume 263, pages 239–250, Berlin, 1986. Springer-Verlag. Lecture Notes in Computer Science.

[CSD04a] J. Castellà-Roca, F. Sebé, and J. Domingo-Ferrer. Abandono de jugadores en esquemas distribuidos de juego de cartas. In B. Ramos Álvarez and A. Ribagorda Garnacho, editors, *Avances en criptologia y seguridad de la información*, pages 243–249. Diaz de Santos, 2004. ISBN 84-7978-650-7.

[CSD04b] J. Castellà-Roca, F. Sebé, and J Domingo-Ferrer. A ttp-free mental poker protocol achieving player confidentiality. *Theoretical Computer Science*, under review since december 2004.

[CSD05] J. Castellà-Roca, F. Sebé, and J. Domingo-Ferrer. Dropout-tolerant ttp-tree mental poker. In J. Lopez and G. Pernul, editors, *Trust and Privacy in Digital Business – Trustbus'05*, volume to appear of *Lecture Notes in Computer Science*, page to appear. Springer-Verlag, 2005.

[CY02] J-S. Chou and Y-S. Yeh. Mental poker game based on a bit commitment scheme through network. *Computer Networks*, 38(2):247–255, 2002. Elsevier North-Holland, Inc.

[Den83] D.E-R. Denning. *Cryptography and Data Security*. Addison-Wesley, second edition, January 1983.

[DF90] Y. Desmedt and Y. Frankel. Threshold cryptosystems. In *Advances in Cryptology – CRYPTO'89*, volume 335 of *Lecture Notes in Computer Science*, pages 307–315. Springer-Verlag, 1990.

[DF96] J. Domingo-Ferrer. A new privacy homomorphism and applications. *Information Processing Letters*, 60(5):277–282, December 1996.

[DF02] J. Domingo-Ferrer. A provably secure additive and multiplicative privacy homomorphishm. In A. Chan and V. Gligor, editors, *Information Security*, volume 2433 of *Lecture Notes in Computer Science*, pages pp. 471–483. Springer Verlag, 2002.

[DS00] J. Domingo-Ferrer and R.X. Sánchez-del-Castillo. Metodo para la delegacion de datos estadisticos de forma segura. Solicitud publicada en el Boletin Oficial de la Propiedad Industrial, 16 May 2000.

[Edw94] J. Edwards. Implementating electronic poker: A practical exercise in zero-knowledge interactive proofs. Masters thesis, Department of Computer Science, University of Kentucky, May 1994.

[ElG85] T. ElGamal. A public-key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31:469–472, July 1985.

[FKCK96] O. A. Freier, P. Karlton, P. C., and Kocher. The ssl protocol version 3.0. Netscape Communications, March 1996. http://wp.netscape.com/eng/ssl3/ssl-toc.html.

[FM85] S. Fortune and M. Merritt. Poker protocols. In G.R. Blakley and D. Chaum, editors, *Advances in Cryptology - CRYPTO'84*, volume 196 of *Lecture Notes in Computer Science*, pages 454–464. Springer-Verlag, 1985.

[GM82] S. Goldwasser and S. Micali. Probabilistic encryption and how to play mental poker keeping secret all partial information. In *In Proc. 14th*

ACM Symposium on Theory of Computing (STOC), pages 365–377, San Francisco, 1982. ACM.

[GMR89] M. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal of Computing*, 18(1):186–208, 1989.

[HLG00] L. Harn, H.-Y. Lin, and G. Gong. Bounded-to-unbounded poker game. *Electronics Letters*, 36(3):214–215, February 2000.

[HS97] C. Hall and B. Schneier. Remote electronic gambling. In *13th Annual Computer Security Applications Conference*, pages 227–230. ACM, December 1997.

[KKO97] K. Kurosawa, Y. Katayama, and W. Ogata. Reshufflable and laziness tolerant mental card game protocol. *TIEICE: IEICE Transactions on Communications/Electronics/Information and Systems*, E00-A, 1997.

[KKOT90] K. Kurosawa, Y. Katayama, W. Ogata, and S. Tsujii. General public key cryptosystems and mental poker protocols. In Ivan B. Damgård, editor, *Advances in Cryptology - Eurocrypt '90*, volume 473 of *Lecture Notes in Computer Science*, pages 374–388. Springer-Verlag, 1990.

[KS98] D.R. Kreher and D.L. Stinson. *Combinatorial Algorithms-Generation Enumeration*. CRC Press, Boca Raton FL, 1998.

[Lam81] L. Lamport. Password authentication with insecure communications. *Communications of the ACM*, 24(11):770–772, 1981.

[Lip81] R. Lipton. How to cheat at mental poker. In *Proc. AMS Short Course on Cryptography*, 1981.

[MvOV96] A.J. Menezes, P.C. van Oorschot, and S.A. Vanstone. *Handbook of Applied Cryptography*. series on discrete mathematics and its. CRC Press, 1996. ISBN 0-8493-8523-7.

[ON97] R. Oppliger and J.L. Nottaris. Online casinos. In *Kommunikation in verteilten Systemen*, pages 2–16, 1997.

[Pai99] P. Paillier. Public key cryptosystems based on composite residue classes. In *Advances in Cryptology – EuroCrypt'99*, volume 1592 of *Lecture Notes in Computer Science*, pages 223–238. Springer-Verlag, 1999.

[Ped92] T.P. Pedersen. A threshold cryptosystem without a trusted party. In *Advances in Cryptology - Eurocrypt'91*, volume 547 of *Lecture Notes in Computer Science*, pages 522–526. Springer-Verlag, 1992.

[Rab81] M.O. Rabin. How to exchange secrets by oblivious transfer. Tecnnical Memo TR-81, Aiken Computer Laboratory, Harvard University, 1981.

[RSA77] R.L. Rivest, A. Shamir, and L. M. Adelman. A method for obtaining digital signatures and public key cryptosystems. Technical Report MIT/LCS/TM-82, MIT, 1977.

[Sch91] C.P. Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, 4(3):161–174, 1991.

[Sch96] B. Schneier. *Applied Cryptography, Protocols, Algorithms, and Source Code in C.* John Wiley & Sons Inc, second edition, 1996.

[Sch98] C. Schindelhauer. A toolbox for mental card games, 1998. Medizinische Universität Lübeck.

[Sha79] A. Shamir. How to share a secret. In *Communications of the ACM*, volume 22, pages 612–613, 1979.

[SRA81] A. Shamir, R.L. Rivest, and L. Adleman. Mental poker. *Mathematical Gardner*, pages 37–43, 1981.

[SSG02] W.H. Soo, A. Samsudin, and A. Goh. Efficient mental card shuffling via optimised arbitrary-sized benes permutation network. In *Information Security Conference*, volume 2433 of *Lecture Notes in Computer Science*, pages 446–458. Springer-Verlag, 2002.

[Sta96] M. Stadler. Publicly verifiable secret sharing. In *Advances in Cryptology - Eurocrypt'96*, number 1070 in Lecture Notes in Computer Science, pages 190–199. Springer-Verlag, 1996.

[Yun85] M. Yung. Cryptoprotocols: Subscription to a public key, the secret blocking and the multi-player mental poker game. In *Advances in Cryptology: Proceedings of Crypto'84*, number 196 in Lecture Notes in Computer Science, pages 439–453. Springer-Verlag, 1985.

[ZV05] W. Zhao and V. Varadharajan. Efficient ttp-free mental poker protocols. In H. Selvaraj and P.K. Srimani, editors, *International Conference on Information Technology, ITCC 2005*, volume I, pages 745–750, Las Vegas, Nevada, USA, April 2005. IEEE Computer Society. ISBN 0-7695-2315-3.

[ZVM00] W. Zhao, V. Varadharajan, and Y. Mu. Fair on-line gambling. In *16th Annual Computer Security Applications Conference (ACSAC'00)*, pages 394–400, New Orleans, Louisiana, December 2000. IEEE.

[ZVM03] W. Zhao, V. Vadaharajan, and Y. Mu. A secure mental poker protocol over the internet. In C. Johnson, P. Montague, and C. Steketee, editors, *Australasian Information Security Workshop*, volume 21 of *Conferences in Research and Practice in Information Technology*, pages 105–109, Adelaide, Australia, 2003. ACS.