Spring 2007

The Elgamal Cryptosystem is better than Th RSA Cryptosystem for Mental Poker

Ipek Tetikoglu

The Elgamal Cryptosystem is better than The RSA Cryptosystem for Mental Poker

A Thesis

Presented to the Faculty

of the Department of Mathematics and Computer Science

McAnulty College and Graduate School of Liberal Arts

Duquesne University

in partial fulfillment of

the requirements for the degree of

Masters of Science in Computational Mathematics

by

Ipek Tetikoglu

March 19, 2007

Ipek Tetikoglu

Elgamal Cryptosystem is better than RSA Cryptosystem for Mental Poker

Master of Science in Computational Mathematics

Department of Mathematics & Computer Science
Duquesne University, Pittsburgh, PA, USA

March 19, 2007

Contents

List of Tables

Chapter 1

Acknowledgements

It is a pleasure to thank the many people who made this thesis possible.

It is difficult to overstate my gratitude to my thesis supervisor, Dr. Patrick Juola. With his enthusiasm, his inspiration, and his efforts to explain things clearly and simply, he helped to make cryptography fun for me. Throughout my thesis-writing period, he provided encouragement, good advising, good company, and lots of good ideas. I would have been lost without him.

I would like thank my committee member, Dr John Kern for reading the whole thing so thoroughly. I also would like to thank all the rest of the academic and support staff of the Computational Mathematics Department and the Department of Mathematics and Computer Science at the Duquesne University.

I have to say “thank-you” to my special friend Mr. G. Brinton Motheral for all of his effort for reading my thesis, for his support and endless inspiration.

Lastly, and most importantly, I wish to thank my parents, Ruveyde Tetikoglu and Ismail Arif Tetikoglu. They supported me, taught me, and loved me. To them I dedicate this thesis.

Chapter 2

Introduction

2.1 Abstract

Cryptosystems are one of the most important parts of secure online poker card games. However, there is no research comparing the RSA Cryptosystem (RC) and Elgamal Cryptosystem (EC) for mental poker card games. This paper compares the RSA Cryptosystem and Elgamal Cryptosystem implementations of mental poker card games using distributed key generation schemes. Each implementation is based on a joint encryption/decryption of individual cards. Both implementations use shared private key encryption/decryption schemes and neither uses a trusted third party (TTP). The comparison criteria will be concentrated on the security and computational complexity of the game, collusions among the players and the debate between the discrete logarithm problem (DLP) and the factoring problem (FP) for the encryption/decryption schemes. Under these criteria, the comparison results demonstrate that the Elgamal Cryptosystem has better efficiency and effectiveness than RSA for

mental poker card games.

2.2 Problem Statement

Computer networks and especially the Internet have become very popular in daily life over the past several years. This kind of remote luxury has allowed many on-line activities such as e-shopping and e-gambling. The problem of mental poker is directly related to e-gambling. One can define this problem as how to play a fair game of poker without the need for a trusted dealer [16]. Fairness can be described as how one player can be sure that none of the players are peeking at the cards of other players after the deck has been shuffled and the cards dealt. Accomplishing fairness in the mental poker game in a non-physical environment becomes difficult in shuffling and dealing the cards. Mental poker has many advantages for players including being time-independent as well as place-independent. When the desire to play is there, a player can immediately go online to play. All one needs is a computer, an internet connection and a valid credit card.

The main disadvantage of mental poker is the inability to establish the trust of a player that the online game is being fairly and honestly conducted. In a physical casino, players are able to see the actions taken by the dealer and other players during the course of the game, so the fairness of the game can be observed. In the digital world, this is not the case. In on-line casinos, without some kind of encryption key, the card value can be obtained by third parties. Since online players are not able to see the dealer shuffling and dealing, the appearance of random fairness becomes most important.

In most on-line casinos, during shuffling and dealing, TTPs are used to provide non-manipulated random values for the cards. In this case, on-line casinos turn into a privileged entity and the players can not verify the fairness of the cards. Random card values can be jointly generated by the players using cryptographic protocols in the absence of a TTP.

In this study, we will investigate two different cryptographic protocols. The first one is called the Elgamal Cryptosystem (EC) [5], which was defined by Tahir Elgamal in 1984. The second cryptosystem is called the RSA Cryptosystem (RC) [16], which was defined by Ron Rivest, Adi Shamir and Len Adleman in 1977. This thesis will attempt to prove is that the Elgamal Cryptosystem is more efficient and secure than RSA, thus a better fit for the security requirements of on-line casinos.

2.3 The Significance of The Proposed Problem

Existing cryptosystems have been subject to comparisons in the past. However, in terms of efficiency and security, there is a need to show the specific comparison between EC and RC for mental poker. There is no research in this specific field to show the different aspects of these two cryptosystems when they are applied to mental poker. Furthermore, this research will provide online casino owners/designers with significant security information to enhance and improve the security of their internet gambling site. The aim of this research is to show the comparisons between EC and RC when they are applied to the mental poker.

2.4 Methods

We will concentrate on four different evaluation criteria. These are: the computational burden of adding one more player to the on-line poker table, “security-per-bit” issues for key material, multi-player collusions and the securities of computing logarithms. In our study, the first comparison criteria is chosen to show time efficiency of adding one more player to the mental poker game using two different cryptosystems. The second comparison criteria shows the security levels of encryption and decryption key pairs for both cryptosystems. The key strength is important for possible eavesdropping attacks such as a player who wants to retrieve the card values of his/her opponents. The third criteria is chosen to show collusion effects in a mental poker game using Elgamal and RSA cryptosystems. In a mental poker game collusion can be defined as secret player cooperation. Finally, the forth criteria focuses on the security aspects of both cryptosystems in general.

2.5 Overview

2.5.1 Cryptosystems

In this section, we will give the descriptions of Elgamal and RSA cryptosystems.

The Elgamal Cryptosystem: In Elgamal key generation, each communicating party should perform the following steps [14].

1. **Elgamal Key Generation**

   (a) Generate a random prime p and a generator α of the multiplicative group $(\mathbb{Z}/p\mathbb{Z})$.

   (b) Select a random natural number $a < p - 1$ and compute $\alpha^a (\bmod\, p)$.

   (c) The public key is (p, α, α^a) and the private key a.

2. **Elgamal Public-Key Cipher**

   If entity A wishes to send a message to entity B, then A must perform the following steps in the enciphering stage.

   (a) Obtain B's public-key (p, α, α^a).

   (b) Convert the plaintext message into an integer m in the range $\{0, 1, \ldots, p-1\}$.

   (c) Choose a random natural number $b < p - 1$.

   (d) Compute $\beta \equiv \alpha^b (\bmod\, p)$ and $\gamma \equiv m(\alpha^a)^b (\bmod\, p)$.

   (e) Send the ciphertext $c = (\beta, \gamma)$ to B. Once B recieves c, then the following is performed. This stage is called the deciphering stage.

      i. Use the private key to compute $\beta^{p-1-\alpha} (\bmod\, p)$.

      ii. Decipher m by computing $\beta^{-a}\gamma (\bmod\, p)$.

   The deciphering stage is computed using below equation.

$$\beta^{-a}\gamma \;\equiv\; \alpha^{-ab} m \alpha^{ab} \;\equiv\; m (\bmod\, p)$$

According to the scheme above all the parties should perform the same steps to encrypt and decrypt the cards on the table .

RSA Cryptosystem: We review this cryptosystem in two parts. In the part one of two each entity should perform the following procedures [16].

1. Generate the two keys, choose two random large prime numbers, p and q. For maximum security, p and q should be chosen of equal length.

2. Compute the product $n = pq$ and $\phi(n) = (p-1)(q-1)$. n is called the RSA modulus.

3. Randomly choose the encryption key e where $gcd(e, \phi(n)) = 1$.

4. Use the Extended Euclidean Theorem to compute the decryption key such that

$$ed = 1 \bmod (\phi(n)). \tag{2.1}$$

   A further computation is as follows:

$$d = e^{-1} \bmod \phi(n). \tag{2.2}$$

   The numbers e as in Equation 2.1 and n are the public key; the number d is the private key.

5. In the second part of the cryptosystem review to encrypt a message, m, one should first divide the message into numerical blocks smaller than n

and then use the encryption formula

$$c = m^e \bmod n. \tag{2.3}$$

6. To decrypt a message each encrypted message block should be computed with the formula below

$$m = c^d \bmod n. \tag{2.4}$$

2.5.2 Protocol

Games of mental poker require shuffling and dealing the cards. The properties desired of mental poker come from the properties of real poker games. Players cannot influence the shuffling, nor learn anything about other players' cards which are dealt face down. In our model protocol we assume that every randomly dealt card is drawn from a set of the cards and adversaries do not know anything about others' cards. With these properties of model protocol, players' trust is established and the fairness of the game is assured.

We now give a detailed protocol overview for our implementations. We shuffle a deck of cards using commutative cryptosystems (CC). Using CC eliminates a need for a TTP in the mental poker game. A CC, using a commutative scheme, means that if some data is encrypted more than once, the decryption order does not matter [16]. EC and RC are commutative cryptosystem schemes. There are three earlier approaches for card shuffling protocols. The first approach is using a TTP. This is a relied upon dealer that is used to fairly shuffle and deal the cards [10]. The main disadvantages of this approach are the possible collusion of some of the players and poorly shuffled cards [0]. Another approach is Creapau's algorithm for distributing trust among the players. The main drawback is the tremendous computational burden [6]. The last approach is a mix server that takes ciphertexts and corresponding outputs [10]. This approach is also highly costly in terms of modular exponentiations. In our threshold protocol schemes implementing RC and EC, players jointly generate and share public and private keys using RSA and Elgamal cryptosystems' encryption schemes.

Protocol Overview and Basics

Poker is played with a 4-suit 52 card deck. In our implementations, we assume that the deck includes exactly 52 cards. We represent the cards in the deck with the set of integers modulo 52, i.e. the set $\{0, 1, \ldots, 51\}$. Efficiency is given a measure of modular exponentiations and leads us to the computational cost. We will eliminate the existence of an TTP in shuffling and dealing the cards. In this research, we use distributed key generation without relying on a trusted third party for both implementations and comparisons of the cryptosystems.

In our implementations card shuffling and dealing is executed in the same step. The cards are generated when they are needed. We skip the initial step of encrypting all cards at once. This approach helps with the time efficiency of our structure. Players jointly encrypt $E(c)$ of a card $c \in \{1, 2, \ldots 52\}$ and then they compare the ciphertext $E(c)$ with all the encrypted cards that have been previously dealt from the current deck. If a match is found, players try to generate another one until a match is not found. Players also keep a list of encrypted cards that have been dealt from the current deck [10]. In our protocol, encryption/decryption algorithms for E are distributed among k players who are in the game and each encryption scheme E is additively homomorphic. Given only the public-key and the encryption of $m1$ and $m2$, one can compute the encryption of $m1 + m2$. There is also a protocol that allows the joint holders of the distributed key to check if a newly generated card has already been dealt.

Additionally, players jointly generate the key material of the desired encryption scheme. Each player receives the public key and their share of the private key. Key generation only needs to be repeated if any of the players leave the game or if there is a new player joining to game. Each player keeps the list of the encrypted cards that have been dealt previously from the current deck [2].

Dealing a card using the basics stated above is executed as follows:

1. Each player k_i chooses a card, $c_i \leftarrow\in \{1, 2, \ldots 52\}$ and generates the ciphertext of that card, $E(c_i)$.

2. Each player k_i reveals $E(c_i)$. If any of the commitments are incorrect, protocol fails. In this case, honest players establish another session and repeat the needs of the protocol again.

3. Each player computes $E(c)$. They use additive homomorphism property of our protocol, i.e. $E(m_1)\ E(m_2) = E(m_1 + m_2)$. This property helps to combine all the honest players' ciphertext shares to generate the final output of C, where $C = \sum_{i=1}^{k} k_i$.

4. The players keep a list of cards, L that have been already dealt from the current deck. At the very beginning of the game, L is set to 0. For the fairness of the game players must perform a test to determine if a card has already been dealth from L. If there exists a card such that $E(c) = E(c^{'}) \in L$, then the players discard that card and start generating another until the one is not found.

5. Each player updates L. Also, dealing a card to the (k_i)th player in the game requires all the other players to partially decrypt their part of the ciphertext. The resulting ciphertext is ONLY decrypted by the (k_i)th player and only (k_i)th player can reveal the card value.

We now propose two implementations of mental poker for the above protocol. The first is based on RC and the second is based on EC.

2.5.3 Implementation of Mental Poker

RSA Implementation

We implement our protocol using a shared generation of RSA keys. Since the standard RSA scheme requires key generation in a single location, we do not use it for our implementation. At the end of the computation, an RSA modulus is publicly known and each player holds a share of the private key [1]. Once the players generate an RSA modulus, as we recall from the definition of the cryptosystem, they can compute the shares of the private key for a given encryption exponent [5]. Benolah's simple algorithm for additive homomorphic structure of the private key generation is useful to combine all the outputs of each player's share [5]. We gave an explanation for this step in in the previous section at bullet 3. Each player computes c_i and sends them to the other players. Each player follows the below path for generating and encrypting/decrypting a card:

1. RSA modulus is $N = pq = (\sum p_i)(\sum q_i)$.
2. Players compute shares of private key $d = e^{-1} \text{mod } \phi(n)$.
3. Each player locally computes $\phi_i = -p_i - q_i$, $\phi_N = \sum \phi_i$.
4. Players jointly determine the value of $l = \phi_N \text{mod } e$ using Benoloh's protocol. His protocol uses additive homomorphisim. This means each player k_i generates an additive sharing of $\phi_i = \sum_j x_{i,j} \text{mod } e$. Then, they send $x_{i,j}$'s to all other players, and each player locally computes $\sum \alpha_j = \sum_j \phi_j = l \text{mod } e$.
5. The above description of the distributed key generation allows distributed encryption and decryption of the cards.
6. Each player encrypts a card using their share of private key. The resulting ciphertext is $E(c_i)$.

7. Each player decrypts $E(c_i)$ using equality $c^d \equiv c^r \prod c^{d_i}$, where c is the ciphertext of each card [6].

Elgamal Implementation

In this section, we propose an implementation of our protocol that is based on EC encryption scheme [7].

1. Pedersen's protocol lets a number of players generate an Elgamal public/private key pair in distributed fashion [16]. Pedersen's protocol is a distributed key generation protocol that allows a set of players in the game to jointly generate a pair of public and private keys in such a way that the public key is known to all players while the private key is shared by n servers [16].
2. Each player learns a share of x_i of the private key x such that $\sum_{i=1}^{k} x_i = x \text{mod } q$.
3. To encrypt a card value C, let $E(C) = (g^r, Cy^r)$ where $r \leftarrow (\mathbb{Z}/p\mathbb{Z})^*$.
4. To decrypt a ciphertext (a,b), compute $b/a^x = C$. Also, we know that when the private is jointly generated among the k players, each player knows x_i of the private key x such that $\sum_{i=1}^{k} x_i = x \bmod q$, where q is a prime from EC's description.

Additionally, we also propose that the players play the game on an SSL secured environment to ensure the authenticity and confidentiality of connections.

Chapter 3

Model Implementation of Comparison Criteria

In this section, we will discuss and compare the Elgamal and RSA cryptosystems in four different evaluation criteria for mental poker.

3.1 Security-per-bit

Mental poker is very similar to a regular poker game with the exception that there is no verbal communications and no physical cards used in the online setting. All communication exchanges are made by the text messages among the players [9]. A mental poker protocol scheme must guarantee the fairness of the card game. The key pairs for encryption and decryption should be strong enough to resist cheating attacks. According to Schneier: " the security of a cryptosystem should rest in the key...." We assume that an adversary has the details of the algorithm, but he/she still has to retrieve the key pairs in order to get the card values. Cheating attacks target the key pairs that are used to encrypt the cards. Schneier also says that today's public key algorithms are based on the difficulty of factoring large numbers that are the product of two large primes. These two large prime numbers are used to generate the key pairs of the cryptosystems. Breaking these algorithms based on the cryptosystem schemes involves trying to factor the large number.

In our research, we will use symmetric and public-key key pair comparisons that have similar resistance to brute force attacks. A brute-force attack is trying every possible key one by one and checking whether the resulting text is meaningful. This attack is not the only known type to retrieve the card values [16]. However, it is helpful for the symmetric and public-key comparisons used in our tables.

For the game of mental poker "Security-per-bit" (SPB) can be defined as a ratio between the number of the operations necessary to identify a card and the number of key bits. In an online casino, SPB is a tradeoff between performance and security. Archiving an acceptable performance, the encryption, decryption, signing and verification of the cards in a hand of poker should be as fast as possible. However, the keys should be impervious to any kind of attacks to decrease the probability of cheating.

In an online casino, dealing and shuffling cards are rapid and there are no delays relating to counting chips for a split pot. It is not uncommon for an online poker table to average sixteen to twenty hands per hour. However, the number of players and the type of the game are parameters that influence the performance of the game in terms of time requirements per hand. In this paper, we will assume that each poker hand takes between six and eight minutes including dealing and shuffling, regardless of type of game and number of the players. This approximation partially answers how much time we need to insure the security of the cards per hand. We propose that one session of poker game needs to stay secure up to eight minutes.

SPB of RSA Cryptosystem : RSA gets its security from the difficulty of factoring large numbers [18]. A list of sample key conversions between RSA key lengths and symmetric key lengths is given in the Table (3.1). This table compares the equivalent security level for commonly known RSA key sizes.

Symmetric Key Length	**Time Only Equivalent RSA Key Size**
56 bits	512 bits
80 bits	1024 bits
112 bits	2048 bits
128 bits	3072 bits
192 bits	7680 bits
256 bits	15360 bits

Table 3.1: Symmetric and RSA Key Length Equivalents Assuming that Time is Binding Constraint in order to break RSA Keys.

According to Schneier, the efficiency of computing equipment divided by price doubles every 18 months and increases by factor of ten every five years [18]. Table (3.2) shows the security requirements for different information. Minimum key length is given in symmetric lengths numbers.

Using the Tables 3.1 and 3.2, we can understand why the key size is important in terms of security and performance. By keeping the cards secure during the life of a poker hand, all the players are insured that underlying keys are sufficient

Type of Traffic	**Lifetime**	**Minimum Key Length**
Tactical military information	minutes/hours	56-64 bits
Product announcements, mergers, interest rates	days/weeks	64 bits
Long term business plans	years	64 bits
Trade secrets(e.g., recipe for Coca-Cola)	decades	112 bits
H-bomb secrets	$> 40\ years$	128 bits
Identities of spies	$> 50\ years$	128 bits
Personel affairs	$> 50\ years$	128 bits
Diplomatic embarrassments	$> 65\ years$	at least 128 bits
U.S. census data	$> 100\ years$	at least 128 bits

Table 3.2: Security Requirements for Different Information.

enough to render the best possible cheating attack infeasible. This leads us to the questions how long such keys will be secure as well as to what the key size should be. If the lifetime of the data being protected is measured in only days and/or weeks, there is no need to use a key that will take years to break to retrieve the card data [18]. Since one hand of poker takes approximately eight minutes, we have to choose our session key carefully for the performance and security of one hand poker. As a basis of comparison, we will use data from the break of RSA-512 key length. On August 22 1999, this effort required a total of 8000 MIPS-Years, 35.7 CPU years to do the sieving, represented by about 300 PCs averaging 400 MHz and with at least 64 Mbytes of RAM, running for 2 months, and 10 days and 2.3 Gbytes of memory on a Cray C90 to solve the matrix [27]. Using this data and remembering Moore's Law, processor speed doubles every 18 months, we can drive the amount of time to break RSA-512 in 2007. It would take 1.8 months to break the same size of key. In our paper, we propose that 512-bit RSA key size is adequate for an 8 minute session of one hand poker game. An example of an online casino Pokerstars.com confirms on their internet site that their configuration uses RSA for authentication and key generation. Currently, they are using 512-bit RSA key, which according to Schneier is sufficient for short and medium-term (up to several years) secrets.

SPB of Elgamal Cryptosystem Elgamal is a form of the Diffie-Hellman key exchange problem in Z which is at least as difficult as the problem of factoring [22]. Elgamal encryption gets its security from the difficulty of discrete log problem (DLP)[7]. The security is related to the difficulty of deducting the private key from the public key. This depends on the length of the public/private key pair and the computing power that might be used to break the key pair to retrieve the card value. In our research, we will show the strength of Elgamal keys in terms of symmetric key length that will allow us to make comparisons between SBP of Elgamal and RSA cryptosystems later in this chapter. We will use the conversion chart between symmetric and asymmetric key lengths that is shown in the Table(3.3) for comparisons.

Symetric Key Length	**Elgamal Prime Parameter p**	**Elgamal parameter α**
56 bits	512 bits	112 bits
80 bits	1024 bits	160 bits
112 bits	2048 bits	224 bits
128 bits	3072 bits	256 bits
192 bits	7680 bits	384 bits
256 bits	15360 bits	512 bits

Table 3.3: Symmetric and Elgamal Key Lengths with Similar Resistances to Brute-Force Attacks.

The table 3.3 shows Elgamal public-key primary security parameter lengths whose factoring difficulty almost equals the difficulty of a brute-force attack for symmetric key lengths. This table also shows that if one is choosing 56-bit symmetric key length, equal security is provided by 512-bit Elgamal public-key algorithm.

The best known attack on EC is solving the discrete logarithm problem. Elgamal cryptosystem gets its security from the difficulty of discrete logarithm problem [18]. The discrete logarithm problem in a finite group G can be stated as follows: compute x from g and $u = g^x$. The integer x is called the discrete logarithm of u in base g, $x = \log_g(u)$ [8]. There are several algorithms to solve this problem. Number field sieve (NFS) is one of the best known solving algorithm for DLP problem and based on integer factorization. Since DLP is a hard problem [18], we study these algorithms by their asymptotic and practical running time behaviors. The running time consequences are directly related to the key sizes of the cryptosystems. It is very difficult to give an estimation of the largest key size that can be solved by DLP algorithms.

Comparison of SPB for both Cryptosystems Analysis based on the best available algorithms for both factoring and discrete logarithms show that the RSA system and the Elgamal system have similar security for equivalent key lengths [20]. It is slightly harder to compute discrete logs modulo an appropriate prime than to factor a hard integer of the same size - so RSA would appear slightly weaker than DHP/Elgamal [20]. According to Schenier, "RSA users have to choose a larger key size those using than DH/Elgamal over a finite field for equivalent security." Since we know that computing discrete logarithms is closely related to factoring, if one can solve discrete logarithm problem then one can factor [18]. Similarly, if DH is broken by solving the DLP then RSA can also be broken, since, if one knows how to take discrete logs, then one can factor (that is the basis of Shor's quantum factoring algorithm [20]). Thus, DLP would seem stronger than the factoring problem (FP), since factoring does not allow one to solve discrete logs [20].

The largest known factorization for NFS factoring algorithm is 512-bit RSA number [RSA]. This consequence gives an accurate figure of what can be accomplished in terms of factorization algorithms. The NFS algorithm for DLP problem is more limited. According to Joux and Lercier, over a 120-digit prime field that corresponds to 397-bit length DLP/Elgamal key size [30].

Similarly, comparing the "largest breaks" of each key type, we see that an 512-bit RSA key has been broken but only a 283-bit DH key has been broken [20]. The 512-bit RSA break used around 8,000-MIPS years and it has been predicted that the task is equivalent to an Elgamal key in a prime field with a characteristic of 365-bits [20]. Additionally, Elgamal provides equivalent security with 365-bit public-key whose factoring difficulty is closely equal to the difficulty of the 64-bit symmetric key. Thus, Elgamal has an equivalent security to RSA with a shorter key length.

3.2 Computational Burden

Computational cost is a function of operations that require time, space and software power. These operations are shuffling, dealing and adding more players to the game. In this section we will evaluate computational cost of both cryptosystems for mental poker.

Computational Cost of Using RSA cryptosystem for mental Poker

Existing protocols for distributed RSA-key generation allow three or more parties to generate an RSA modulus $N = pq$ such that all parties are convinced that N is a product of two primes. However, none of them can factor N. These protocols also show how the parties can generate shares of a private decryption exponent to allow treshhold decryption. It is possible to test that N is a product of two primes. K parties computes the shares of N in a private distributed computation where $N = pq = (\sum p_i)(\sum q_i)$ without revealing any information about the factors is possible.

Operation	**512 bits**	**768 bits**	**1024 bits**
Encrypt	0.03 sec	0.05 sec	0.08 sec
Decrypt	0.16 sec	0.48 sec	0.93 sec
Sign	0.16 sec	0.52 sec	0.97 sec
Verify	0.02 sec	0.07 sec	0.08 sec

Table 3.4: RSA Speeds for Different Modulus Lengths with an 8-bit public key (on a SPARC II).

Implementation of mental poker using RSA requires distributed key generation. Private keys should not be shared. In RSA, while each person should have a unique modulus and private exponent (that is, a unique private key), the public exponent can be common to a group of users without security being compromised. Some public exponents in common use today are 3 and 216+1; because these numbers are small, the public key operations (encryption and signature verification) are fast relative to the private key operations (decryption and signing). If one public exponent becomes standard, software and hardware can be optimized for that value. However, the modulus should not be shared.

In a research study performed by [2] from Stanford University showed that generating a 1024-bit RSA key among the three 300Mhz Pentium machines took 90 seconds. Total network traffic was 1.16 MB. In another example of shared RSA key generation in a mobile adhoc network, a WLAN of three computers for a 512-bit key, averaged 2.5 minutes.

Table(3.6) shows some results when running the system on three servers. This table was constructed by the generation of the key in three different machines. This table shows that when computing a 512-bit modulus, each thread spent an average of 55.7ms per iteration executing the BGW protocol. Experimentally, two threads per machine is optimal for a 512-bit key and 6 threads per server is optimal for a 1024-bit key.

This estimation was generated using 333MHz Pentium II and server were connected by a 10-Megabit Ethernet [2]. Estimation does not reflect the time to generate a sharing of private key d once the modulus is found since it is not significant compared to the rest of the study. However, for a 512-bit key, this time is given as 20ms. It is also estimated that 512-bit key requires an average of 238 iterations per thread.

The consequences of the above studies can help us to understand and compare time requirements of a mental poker game with more than two players. If there are three players in the game, the RSA-key generation takes about 0.15 minutes and if we add one more player to the game the overall key generation increases 2.19 minutes to 3.70 minutes. Furthermore, if there are five players in the game the time estimation for the RSA-key generation increases 4.10 minutes to 5.61 minutes. If we think about the time estimation of one hand of mental poker game, we need approximately 6-7 minutes to shuffle and deal the cards and finish the round. Thus, we can see computational burden of adding more players to the poker game causes overall time expansion.

Additionally, Table(3.6) shows the time estimations for distributed key generation of RSA implementation for 1024-bit keys.

After the RSA key generation, each player receives the public parameters, public key and a share of the private key. A new group establishment is needed if a player leaves or a new player joins to the table.

Key Length	**Total**	**Net**	**Number**
	Time(k)	**Traffic**	**of threads**
512-bit	0.15 min	0.180 Mb	2
1024-bit	1.51 min	1.162 Mb	6
2048-bit	18.13 min	7.48 Mb	6

Table 3.5: Shared Key Generation Time Among Three Servers.

Number of Servers	**Time per iteration**	**Total Time**
	per iteration(k)	
3-server	695	1.51 min
4-server	1707	3.70 min
5-server	2589	5.61 min

Table 3.6: Shared Key Generation Time Among Multiple Servers.

Table(3.7) shows the computational cost of the protocol [12] for the stages after the RSA key generation in terms of modular exponentiations per player. As we know, modular exponentiations are one of the most important operations in public-key cryptography. However, it takes time because the modular exponentiation deals with very large operands as 512-bit integers such as RSA modulus. A modular exponentiation is composed of repetition of modular multiplications.

Operation	**Cost**
Deck setup	2
Test of Plaintext Equality	$4k$
Dealing a card	$4k\|L\|/(1-\|L\|/52)$

Table 3.7: Number Of the Modular Exponentiations Per Player for RSA.

Computational Cost of Using Elgamal Cryptosystem for mental Poker

Elgamal cryptosystem offers distributed key generation just like RSA cryptosystem. Pedersen's protocol [17] lets a number of the players generate an Elgamal public/private key in a group-oriented fashion. Players precompute and store the values of encrypted cards in the memory.

Similar to the RSA implementation, Table(3.8) shows the computational cost for generating and dealing a card after the group establishment. This is given in modular exponentiations.

Operation	**Cost**
Generating a card	2
Dealing a card face up/down	0
Recieving a card face up/down	$2(k-1)$
Reducing a card face down	$< 2(86+8k)$
Testing for one collision	$8k-1$

Table 3.8: Number of the Modular Exponentiations Per Player for Dealing a Card for Elgamal.

From Table(3.8) and Table(**??**), we can see the significant difference between card dealing stages of both cryptosystems in terms of modular exponentiations for mental poker. Table(3.7) gives us a conjecture running time as opposed to Table(3.8) where the running time of modular exponentiations is almost equal to zero.

Modular exponentiations are very significant part of public-key cryptography. There are several algorithms to make exponentiations faster. In our research, we will adopt the time estimations from a KAIST (Korea Advanced Institute of Science and Technology) study. Table(3.9) shows the execution times of two algorithms that was implemented in the study of KAIST [31]. Operations were performed on a Pentium 90 microprocessor PC.

Table(3.10) was constructed according to the study results of [12]. This table shows the expected total real cost of a game with k players, measured in terms of the number of exponentiations that each player must perform for Elgamal cryptosystem.

Algorithm	**256-bit**	**512-bit**	**768-bit**	**1024-bit**
Montgomery(WMM)	0.137	0.544	1.16	2.07
Proposed(WMM)	0.0604	0.220	0.489	0.868
Montgomery(MS)	0.121	0.445	0.917	1.65
Proposed(MS)	0.0851	0.297	0.698	1.22

Table 3.9: The Execution Time of Each Algorithm in msec.

Scheme	**Elgamal**
Texas Hold'em(3-player)	
Deck setup and 1st round	800
Additional rounds (Max)	660
Texas Hold'em(5-player)	
Deck setup and 1st round	1700
Additional rounds (Max)	2900

Table 3.10: Expected Total Real Cost of a Game with k Players in Terms of Modular Exponentiations.

Combining the results of two tables, Table(3.9) and Table(3.10), we can reach the 176 msec for approximately 800 modular exponentiations for 1024-bit key operations. For a game of 3 players this number is calculated as 694.4 msec. For a game of 5 players, we reach 7378 msec. This shows that adding two more players to the poker table increases the computational cost 6684 msec per hand.

A

dding one or more players to a poker hand RSA implementation, even at the distributed key generation stage, adds significantly to the computational time. For a three-player hand, the time is estimated as 1.51 mins. When a fourth player joins the game, the computational time increases 2.19 minutes. This number again increases by 4.1 minutes if two additional players join the game.

Elgamal implementation provides a better computational time estimation. For a game of 3 players, the time for key generation, deck setup and dealing a card is calculated as 694.4 msec. For a game of 5 players, we reach 7378 msec. This shows that adding two more players to the poker table increases the computational cost by 6684 msec per hand. Using these computational results, we can see that the Elgamal implementation is by far the most efficient for mental poker.

3.3 Multi-player Collusions

Collusion is a form of cheating in which two or more players signal their holdings or otherwise form a cheating partnership to the detriment of the other players at the same table. [35]

In card games, colluding players exchange information in order to collectively win the game. Card exchange, knowledge of a opponent's cards without player's consent, and shuffling and dealing the deck for one's own benefit, are the common forms of collusion. In the real world poker, players can usually observe and discover such collusion attempts.

However, a secret communication channel between the players of a coalition is possible in mental poker. One player can contact another player in the group to signal his/her holdings. A mental poker protocol should be designed to eliminate the possibilities of such collusions.

The algorithms for implementation of mental poker we propose use the properties of threshold cryptography. In threshold cryptography, the secret key is split into shares and each share is given to each member in the group of players [3]. The secret key is generated in a distributed manner and players jointly generate a random key such that at the end of the process all honest players have a share of the secret key.

Collusion Effects of RSA Implementation for Mental Poker

There are several algorithms proposed for a fully distributed implementation of RSA cryptosystem. Threshold RSA scheme requires n players to pick random $k - bit$ integers $\boldsymbol{p_i}$ and $\boldsymbol{q_i}$ to jointly generate the RSA modulus N on n different machines. Shoup's threshold RSA scheme allows to jointly generate, encrypt and decrypt among a set of players [17]. However, this scheme uses a TTP. The protocols implemented in

[1] and [2] are more practical but does not allow players to efficiently share RSA modulus with strong primes. Consequently, [1] is not robust against colluding cheaters. Also, [12]'s work revisited the work of [5] and [6]. It is significantly difficult to have a fully distributed version of RSA cryptosystem. In the work of [12], the proof of robustness and safe primes for the RSA modulus is given. Robustness is a measurement of damage caused by colluding players. The distributed RSA secret key generation is robust enough to keep the honest players in the game even if there is any form of collusion among the cheating players. Furthermore, colluding parties cannot prevent honest players from encrypting and signing.

The study of [2] is $\lfloor \frac{k-1}{2} \rfloor$ private. It states if $\lfloor \frac{k-1}{2} \rfloor$ players share the information they learn during the protocol, they won't be able to recover the factorization of N or the private key d. Consequently, if there are 3 players involved in a mental poker game, no single player has any information. Furthermore, we assume that the communication between player i and j is secure. We also assume that one can not eavesdrop on the communication. According to [2], a coalition of $\lfloor \frac{k-1}{2} \rfloor$ learns no other information about the private shares.

Threshold generation of RSA keys may still allow a party to cheat during the protocol [12]. Since the production of safe primes is difficult, one can factor the RSA modulus N. Cheating might cause a non-RSA modulus to be incorrectly accepted. If a group of t players out of original n players are able to cause a non-RSA modulus to be incorrectly accepted, they can eliminate the $n - t$ players from the game and they can collectively win the game. One other drawback of this protocol is that it requires at least three servers to be involved. In this case the protocol is 1-private.

I

n our study, we use the features of Threshold Elgamal Cryptosystem (TEC). We also know that Elgamal cryptosystem is a discrete-log based cryptosystem [10]. Several schemes exist to implement TEC and Pedersen's protocol is one of these schemes. However, it had some security flaws [14,28,3]. Pedersen's protocol for distributed key generation was revisited to solve the security flaws [11,15]. Furthermore, [21]'s work shows that n players can generate a distributed key as well as a signature even if there are k dishonest players. These protocols are fully distributed and $k-$secure, meaning that n members can sign the cards even if there are k dishonest members. All these protocols use the idea of generating one publicly known modulus N while the private key is created and maintained by the EDT scheme [21].

Maximum collusion protection and security is provided with TEC against player coalitions. The proof is given in [10]. Even if k players collude, they can only obtain the cards of the players in the coalition but not a card from the any $n-k$ players. No coalition among the cheating group of players can affect the cards drawn by the honest players or the cards that have not yet been drawn in the encrypted deck. The security behind these consequences are proven in [21].

A

fully distributed version of TEC exists for a more secure and minimum collusion structured $n-$player Mental Poker. However, a fully distributed version of TRC is more challenging and less collusion free [21].

In TEC case, it is easy to compute the inverses of mod q because q is publicly known. With TRC, if we do not know the factorization of RSA modulus N, we can not reveal the inverses of $\phi_N$mod e unless we use special structures [21]. Consequently, TEC can detect the colluding players without a need to use special structures that TRC would require the establishment of the same level of security for the honest players.

Starting with the key generation protocol, TRC is assumed to be non-robust. We assume that all the parties are honest in the game. If TRC is made robust, more invocations are needed to detect the cheating players. The protocol is $\lfloor \frac{t-c-1}{2} \rfloor$ private [2]. Consequently, this approach is only good when both n and k are significantly small.

On the other hand, in TEC, robustness with respect to $n-k$ coalition players is inherited from key generation and decryption protocols [7]. Furthermore, TEC does not require at least 3 players to use the protocol. It is good for any number or players.

3.4 Securities of Computing Logarithms

T

he security of RC depends on the factoring problem and the absence chosen-ciphertext attacks [14]. If an adversary can factor the RSA modulus n then one can compute the private key pairs from the public key for every player in the game. The factoring problem for RC is also a hard problem [26]. The factoring problem have an expected exponential running time of $O(e^{(\ln\ n)^{1/2}}\ (\ln\ \ln\ n)^{1/2}\ (1\ +\ O(1)))$ [17].

Chosen-ciphertext attacks are closely related to the factoring problem [3]. A malicious player may be able to decrypt the ciphertext of a card value if he/she can obtain the other players' encrypted card values. Rivest and Kaliski studied the chosen-ciphertext attacks [28]. Prevention from the chosen-ciphertext attacks introduces the padding schemes that reversibly transfers a plaintext before the encryption. OAEP (Optimal Asymmetric Encryption Padding) is a scheme that has been proven secure for chosen-ciphertext attacks [28].

T

he discrete logarithm problem has been the basis for several public-key cryptosystems, including the ElGamal system. The discrete logarithm is as follows: given an element g in a finite group G and another element $h \in G$ find an integer x such that $g^x = h$. The discrete logarithm problem shows the same relation to Elgamal cryptosystem as factoring does to the RSA cryptosystem. According to [29], the problems of decrypting the private key from the public key and of factoring n are the same in terms of required computations. The task that can be done by someone who has intentions to cheat is that of recovering the plaintext m from Equation 2.4. The adversary can use the ciphertext c and the public information (n, e).

The discrete logarithm problem is also NP-hard and has an expected running time. Discrete exponentiation within a group can be performed with $O(\log n)$ group of operations by using the method of fast exponentiation; however, discrete logarithms appear to be much harder to compute. Methods for computing logarithms require exponential time with the $O(\sqrt{x})$. However, the security of ElGamal cryptosystem partially equivalent to security of Decisional Diffie-Hellman (DDH) assumption. The security of the Diffie-Hellman system depends on the assumption that it is easy to raise a number to a certain power, but difficult to compute which power was used given the number and outcome. This assumption is sometimes stronger than the discrete log assumption [14].

T

he asymptotic behaviors of both cryptosystems demonstrated above are similar but in practice RSA keys are more vulnerable compared to the Elgamal keys. [25]

It is harder to compute discrete logs modulo an appropriate prime than to factor integer of the same size. Consequently, RSA would appear slightly weaker than Elgamal [25]. For an equivalent security, RSA users have to choose a larger key size than those using than Elgamal [25].

Additionally, if one can break Elgamal algorithm by solving the DLP, then one can break RSA algorithm [25]. Thusly, Elgamal based on DLP is stronger than RSA based on FP, since solving FP would not allow one to solve the DLP.

Another argument for using the Elgamal keys rather than the RSA keys would be the autonomy of encryption and signature schemes. If any player in the game manages to obtain the private key pairs of the other players, s/he can only retrieve the playing card numbers and cannot forge signatures. However in RSA, this malicious party has ability to forge signatures [5,23].

Chapter 4

Discussion

There was no comparison results generated between RC and EC in the past for the same visited comparison points and our approaches on the model has generated certain results. Furthermore, according to Sam Simpson [29], a researcher in this area, either Elgamal or RSA are, in operation, significantly less secure than the other, given correct implementation and parameter selection.

In this research, we have given the descriptions of two public-key cryptosystems, Elgamal and RSA cryptosystems, for mental poker. We applied them on the given model to compare the usefulness and effectiveness of both cryptosystems. It is evident from our comparison study results that Elgamal cryptosystem has a better SBP than the RSA cryptosystem. We also showed that the Elgamal cryptosystem has less computational burden in terms of time and modular exponentiations when there is a need to expand the number of the players in a hand of poker. Another useful result of this comparison study is exhibiting that Elgamal cryptosystem has a better resistence to multi-player collusion attacks than the RSA cryptosystem. Additionally, we showed the Elgamal cryptosystem based on the DLP has better security than the RSA cryptosystem based on the FP. We therefore conclude that the Elgamal cryptosystem is better than the RSA cryptosystem in the visited comparison points for mental poker.

4.1 Future Work

Presently, the Elgamal Cryptosystem provides the best security for mental poker. However, there are other cryptosystems being constructed to offer the same benefits. The Elliptic Curve Cryptosystem (ECC) is considered the next generation of public-key cryptography. According to Kaliski [24] from RSA Labs, ECC provides greater strength, higher speed and smaller keys than established standard cryptosystems such as RSA cryptosystem and cryptosystems based on the discrete logarithm problem, including Diffie-Hellman Key Exchange and Elgamal cryptosystems.

Table(4.1) [24] gives key size equivalents assuming that 10 million dollars is available for computer hardware. It also assumes that ECC key sizes should be twice the Symmetric Key sizes.

As we can conclude from 4.1, ECC keys provide equivalent security in smaller key sizes than RSA and discrete logarithm based cryptosystem.

Symmetric Key	**ECC key**	**RSA key**	**Time to Break**	**Machines**
56	112	430	less than 5 mins	105
80	160	760	600 months	4300
96	192	1020	3 million years	114
128	256	1620(1)	10^{16} yrs	16

Table 4.1: Cost Equivalent Key Sizes.

Additionally, further study might compare our model with ECC. For example, implementing the model that explicitly incorporates the structure of ECC and comparing the results with those from our model would provide more insight to successful mental poker modeling strategies.

References

[0] B. Arkin, F. Hill, S. Marks, M. Schmid, T. O. Walls, and G. McGraw. How we Learned to Cheat at Online Poker: A Study in Software Security. $http://www.developer.com/tech/article.php/10923_6 16221_1$.

[1] D. Boneh and M. Franklin. Efficient Generation of Shared RSA keys. *In Crypto '97*, volume 1233 of LNCS, pages 425-439, 1997. .

[2] D. Boneh, M. Malkin, and T. Wu. Experimenting with Shared Generation of RSA keys. *In Internet Society's 1999 Symposium on Network and Distributed System Security (SNDSS)*, pages 43-56, 1999.

[3] R. Canetti and S. Goldwasser. An Efficient Threshold Public Key Cryptosystem Secure Against Adaptive Chosen Ciphertext Attack. *In Eurocrypto '99*, volume 1592 of LNCS, pages 90-106, 1999. .

[4] J. Castella-Roca, V. Daza, J. Domingo-Ferrer, and F. Sebe. Privacy Homomorphisms for E-gambling and Mental Poker. *IEEE Transactions on Information Theory.*

[5] D. Catalano, R. Gennaro, and S. Halevi. Computing Inverses over a Shared Secret Modulus. *In Eurocrypto '00*, volume 1807 of LNCS, pages 190-207, 2000.

[6] D. Chaum. Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. *Communications of the ACM*, 24(2):84-88, Feb 1981.

[7] R. Cramer, R. Gennaro, and B. Schoenmakers. A Secure and Optimally Efficient Multi-Authority Election Scheme. *In Advances in Cryptography EUROCRYPT '97*, volume 1233 of LNCS, pages 103-118, 1997. .

[8] C. Crepeau. A Zero-Knowledge Poker Protocol that Achieves Confidentiality of the Players' Strategy or How to Achieve an Electronic Poker Face. In *Proceedings of*

Crypto '86, volume 263 of LNCS, pages 239-247.

[9] D.-R. Denning. *Cryptography and Data Security*, 2nd ed. Addison - Wesley, January 1983.

[10] T. Elgamal. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. *IEEE Transactions on Information Theory*, 31(4):469-472, Jul 1985.

[11] P. A. Fouque and J. Stern. One Round Threshold Discrete-Log Key Generation without Private Channels. *In PKC '01*, volume 1992 of LNCS 1992, 2001.

[12] P. A. Fouque and J. Stern. Fully Distributed Threshold RSA under Standart Assumption. $http://citeseer.ist.psu.edu/cache/papers/cs/26989/$ $http : zSzzSzwww.di.ens.frzSz \ sternzSzdatazSzSt98.pdf/fully - distributed - threshold - rsa.pdf$.

[13] Y. Frankel, P. MacKenzie, and M. Yung. Robust Efficient Distributed Key Generation. *In STOC '98*, pages 663-672, 1995.

[14] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. Robust Threshold DSS Signatures. *In Eurocrypt '96*, volume 1070 of LNCS, Springer-Verlag, pages 425-438, 1996.

[15] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. Secure Distributed Key Generation for Discrete-Log Based Cryptosystems. *In Eurocrypt '99*, volume 1592 of LNCS, Springer-Verlag, pages 295-310, 1996.

[16] P. Golle. Dealing Cards in Poker Games. *Proceedings of the International Conference on Information Technology : Coding and Computing.(ITCC'05) 0-7695-2315-3/05 IEEE* , 2005.

[17] S. Hong, S. Oh, H. Yoon. New Modular Multiplication Algorithms for Fast Modular Exponentiation. *http://coblitz.codeen.org*:3125/*citeseer.ist.psu.edu/cache/papers/cs* /2510/*http*: *zSzzSzcamars.kaist.ac.krzSzpublicationzSzpaperzSzeurocrypt*96.*pdf*/ *hong*96*new.pdf*

[18] M. Malkin, T. Wu, and D. Boneh. Experimenting with Shared Generation of RSA Keys. *Internet Society's 1999 Symposium on Network and Distributed System Security (SNDSS)*, pages 43-56.

[19] S. Miyazaki, K. Sakurai and M. Yung. On Threshold RSA-signing with no Dealer. *In ICICS '99*, volume 1787 of LNCS, Springer-Verlag, pages 663-672, 1999

[20] R. A. Mollin. *An Introduction to Cryptography.* Boca Raton: Chapman and Hall, 2001.

[21] C. Park, and K. Kurosawa. New ElGamal Type Threshold Digital Signature Scheme. *In IEICE Transactions on Communications/Electronics/Information and Systems, 1996*

[22] T. P. Pedersen (Ed). A Threshold Cryptosystem without a Trusted Party. *Advances in Cryptology – EUROCRYPT'91*, volume 547, pages 522-526, 1991.

[23] R. L. Rivest and B. Kaliski. RSA Problem. *http://theory.lcs.mit.edu/ rivest/RivestKaliski−RSAProblem.pdf*

[24] RSA Labs. *http://www.rsa.com/rsalabs/node.asp?id* = 2088

[25] RSA Labs. *http://www.rsa.com/rsalabs/node.asp?id* = 2004

[26] B. Schneier. *Applied Cryptography: protocols, algorithms, and source code in C.*

New York : John Wiley and Sons, Inc., 1996.

[27] A. Shamir, R. Rivest and L. Adelman. Mental Poker. *Mathematical Gardener* New York : John Wiley and Sons, Inc., 1981

[28] V. Shoup and R. Gennaro. Securing Threshold Cryptosystems against Chosen Ciphertext Attack. *www.shoup.net/papers/*

[29] S. Simpson.
http://www.scramdisk.clara.net/pgpfaq.html/REFEC98

[30] J. Stern. Evaluation Report on the Discrete Logarithm Problem over the finite fields.
$http : //www.ipa.go.jp/security/enc/CRYPTREC/fy15/doc/1027_R5_DLOG.pdf$

[31] V. Shoup. Practical Threshold Signatures. *In Eurocrypt '00*, volume 1807 of LNCS, pages 207-220. Springer-Verlag, 2000.

[32] Y. Tsiounis and M. Yung. On the Security of Elgamal Based Encryption. *In Proc. of PKC '98*, volume 1431 of LNCS, pages 117-134, 1998.

[33] Y. Tsiounis and M. Yung 1997. The Semantic Security of El Gamal Based Encryption is Equivalent to the Decision Diffie-Hellman. *Technical Report GTE Laboraties Inc.*

[34] W. Zhao, V. Varadharajan, and Y. Mu. A Secure Mental Poker Protocol Over The Internet. *Conferences in Research and Practice in Infornation Technology.*, volume 21.

[35] Rules of Poker. $http : //www.gambling - poker.com/$

[36] Data Privacy for our Poker Software.
$http://http://www.pokerstars.com/poker/room/features/security//$