A Cryptosystem Based on the Symmetric Group S_n

Javad N. Doliskani [1], **Ehsan Malekian** [2], **Ali Zakerolhosseini** [3]

[1] Teacher Training University, Tehran, Iran

[2,3] Faculty of Electrical & Computer Engineering, Shahid Beheshti University, Tehran, Iran

Summary

This paper proposes a public key cryptosystem based on the symmetric group S_n, and validates its theoretical foundation. The proposed system benefits from the algebraic properties of S_n such as non commutative, high computational speed and high flexibility in selecting keys which make the Discrete Logarithm Problem (DLP) resistant to attacks by algorithms such as Pohlig-Hellman. Against these properties, the only disadvantage of the scheme is its relative large memory and bandwidth requirements. Due to the similarities in the algebraic structures, many other cryptosystems can be translated to their symmetric group analogs, and the proposed cryptosystem is in fact the Generalized El-Gamal cryptosystem which is based on S_n instead of GF(p).

Key words:
Cryptosystem, Public Key, Discrete Logarithm Problem,

1 Introduction

1.1 Background of Public-Key Cryptosystems and Related Works

Until the late 1970's, the only cryptosystems for message transmission were symmetric key systems. In symmetric key cryptography, any two users who require communicating a message must have a same key to cipher or decipher the message. In 1976, Diffie and Hellman [14] invented a key-exchange system that was entirely a new type of cryptography. The system called ***the public-key*** was based on exponentiation in the finite fields. In Diffie-Hellman key exchange, a finite field $GF(p)$ and a generator $g \in GF(p)$ are chosen and made public. Suppose that two users "A" and "B" wish to agree upon a key. User "A" selects a random integer $2 \leq x \leq p\text{-}2$, and transmits g^x to "B" over a public channel. User "B" also selects a random integer $2 \leq y \leq p\text{-}2$, and transmits g^y to "A". The users "A" and "B" having common key g^{xy}, compute $(g^y)^x = g^{xy}$ and $(g^x)^y = g^{xy}$ respectively. Clearly, finding an efficient discrete logarithm algorithm would make this system insecure, since g, g^x and g^y are all known, and x, y or g^{xy} can be computed. Another way for breaking this scheme is to compute g^{xy} from g^x and g^y, without computing either x or y. This problem is called the Diffie-Hellman problem. The difficulty of this problem is equivalent to computing the discrete logarithms, even though it remains unproven. The partial results about the equivalence of these problems are presented in [15] and [16].

A public-key cryptosystem that is essentially a variant of Diffie-Hellman scheme was introduced by T. El-Gamal [17]. The algorithm performs as follows: Suppose $GF(q)$ is known by public. User "A" selects a generator $g \in GF(q)$, and an integer a. It then publishes (g, g^a) as the public-key and keeps a secret. User "B", who requires to send a message $m \in GF(q)$ to "A", selects an integer $2 \leq k \leq q\text{-}2$ randomly, computes $m.(g^a)^k = m.g^{ak}$, and sends the pair $(g^k, m.g^{ak})$ to "A". User "A" who knows a, recovers m by computing $m.g^{ak}.(g^k)^{-a} = m.g^{ak}.g^{-ak}$.

A public-key system that is based on knapsack problem or subset sum problem has been invented by Merkle and Hellman [20]. The problem is as follows: Given a set of positive integers $\{m_1, m_2, \ldots, m_n\}$ and an integer w, find a n-bit integer $N=(b_n b_{n-1} \ldots b_1)_2$, such that $\sum_{i=1}^{n} b_i m_i = w$. Generally in this public-key system, an instance of the knapsack problem that is easy to solve is selected. It is then transformed to an instance of general knapsack problem which is difficult to solve. The later knapsack set can serve as a public key. The general knapsack problem is **NP**-hard. However, its generality has later been contradicted. Shamir [22] presented an algorithm for the knapsack problem that is polynomial in n. In 1988, another type of knapsack cryptosystem was developed by Chor and Rivest [24], which was broken by Vaudenay [26] at 1998.

Two public key cryptosystems with their security based on intractability of integer factorization, are RSA [27] and Rabin public key encryption [28]. It has been proven that breaking the Rabin public key encryption is as difficult as integer factorization, but no such equivalence for the RSA has been proven. The overview of major attacks on the RSA encryption and signatures are presented in [30].

Another important public key cryptosystem is *Elliptic Curve Cryptosystem* (*ECC*). The first elliptic curve scheme was proposed by Koblitz [33] and Miller [34] independently. The elliptic curve systems are based on a group of points on an elliptic curve which are defined over a finite field. Systems such as Diffie-Hellman key exchange or El-Gamal can be easily modified to work in these groups. As stated in next subsection, there is no subexponential-time algorithm that could solve the Discrete Logarithm Problem (DLP) in these groups. Therefore, the use of an elliptic curve group that is smaller in size and maintains same level of security offers potential reductions in bandwidth, storage, processing power, electrical power and message sizes.

1.2 The Discrete Logarithm Problem (DLP)

The security of many modern cryptosystems depends on the intractability of the Discrete Logarithm Problem (DLP) [9]). Let G denotes a cyclic group of order n and $g \in G$ be a generator of G with $\gamma \in G$. The *discrete logarithm of* γ *to the base* g that denotes by $\log_g \gamma$, is an integer $0 \leq x < n$, such that $\gamma = g^x$. The discrete logarithm problem can be stated as follows: Given $\gamma \in G$, find an integer x that satisfies $\gamma = g^x$. The discrete logarithm has some properties that are similar to any ordinary logarithm. For example $\log_g a.b = \log_g a + \log_g b$ and $\log_g a^k = k.\log_g a$. In the Generalized Discrete Logarithm Problem (GDLP), G is an arbitrary group and $g \in G$ is not necessarily a generator. In this case, the integer x may not exists. In last two decades, there have been substantial improvements in discrete logarithm algorithms. However, the problem still appears to be intractable. This section briefly discusses the complexity of some algorithms for solution of discrete logarithm. Suppose G is a cyclic group of order n and $\alpha, \beta \in G$. The aim is to solve $\alpha^x = \beta$. Consider the field $GF(p)$ where p is a prime. Assuming $\alpha \in GF(p)$ is a generator for this field, then for any $1 \leq \beta \leq p-1$, an explicit form for the discrete logarithm function exists as follows [19]:

$$\log_\alpha \beta \equiv \sum_{i=1}^{p-2} (1-\alpha^i)^{-1} \beta^i (\bmod p)$$

However, this formula is computationally intensive and practically useless. The simplest algorithm, the ***brute-force search*** is to successively compute 1, α^1, α^2, … until β is obtained. This algorithm will clearly finds x. However, since it requires $O(n)$ group of operations, it would be inefficient for large n's. A faster algorithm is the ***baby-step giant-step*** algorithm [4] having a running time and memory requirement of $O(\sqrt{n})$. Therefore it is a time-memory trade-off of the brute-force search method [3]. See [5] for appropriate data structure fort the implementation of this algorithm.

Another algorithm is ***Pollard's ρ-algorithm*** [6]. The expected running time of this algorithm is equal to the baby-step giant-step method, but its memory requirement is negligible. The Pollard's algorithm uses a heuristic function. Oorschot and Wiener in [7] have indicated if t processors are employed, then the Pollard's ρ-method can be parallelized so the expected number of steps required by each processor for the calculation of the discrete logarithm becomes $O(\sqrt{n}/t)$. Some algorithms use the basis of **smooth numbers**. If $\lambda \geq 0$ is a real number, and a a positive integer, then a is a λ-smooth number if for every prime $p|a$, $p \leq \lambda$.

The ***Pohlig-Hellman algorithm*** introduced by Pohlig and Hellman [8], is an algorithm that takes advantage of the factorization of order n of the group G. Let;

$$n = p_1^{\lambda_1} p_2^{\lambda_2} \mathrm{K}\, p_k^{\lambda_k} \quad \lambda_i > 0, \quad i = 1, \mathrm{K}, k$$

be the prime factorization of n. The execution time of this algorithm is $O(\sum_{i=1}^{k} \lambda_i (\lg n + \sqrt{p_i}))$, and it can be adopted to require a negligible amount of memory. The Pohlig-Hellman algorithm in case the order n of group G is a smooth integer, is computationally efficient. However it can simply avoid such orders of n. For example, let G be $GF(2^m)$ and m can be selected such that 2^m-1 be a prime. Thus the order of generator α of G is $n=2^m-1$, which is a big prime, and the algorithm is clearly inefficient for G.

The most efficient algorithm known to the date for solving the discrete logarithm over finite fields is the ***index-calculus algorithm***. According to [9], the index-calculus algorithm was initially introduced by Kraitchik. This algorithm also uses the idea of smooth numbers. It selects a small $B \subseteq G$, which is called the factor base, in such a way that a relatively large subset of elements of G can be expressed as the products of elements of B. Then, the logarithms of elements of B are computed as follows:

1) Select a random integer $0 \leq i \leq n$-1, and compute α^i.
2) Express α^i as;

$$\alpha^i = \prod_{k=1}^{|B|} p_k^{\sigma_k} \qquad \sigma_k \geq 0 \qquad p_k \in B \tag{1}$$

If it can be calculated, then take logarithms of both sides of Eq. (1) in order to obtain:

$$i = \sum_{k=1}^{|B|} \sigma_k \log_\alpha p_k \qquad \sigma_k \geq 0 \qquad p_k \in B \tag{2}$$

3) Repeat steps 1 and 2 till a set of equations of the form Eq. (2) are obtained. Then solve a system of equations to find the logarithms of elements of B. At final stage, $\log_\alpha \beta$ is computed as follows:

- Select a random integer $0 \leq i \leq n-1$, and compute $\beta.\alpha^i$.
- Try to express $\beta.\alpha^i$ as follows:

$$\beta.\alpha^i = \prod_{k=1}^{|B|} p_k^{\psi_k} \qquad \psi_k \geq 0 \qquad p_k \in B \tag{3}$$

If it can not be calculated, go to step (1), or otherwise take logarithms of both sides of Eq. (1) and obtain:

$$\log_\alpha \beta = \sum_{k=1}^{|B|} \psi_k \log_\alpha p_k - i \quad \psi_k \geq 0 \quad p_k \in B \tag{4}$$

The index-calculus algorithm was also suggested independently by Pollard [6] and Adelman [10]. This algorithm is adopted specially for multiplicative group of finite field $GF(p^n)$, which p is a prime. Due to the heuristic nature of this algorithm, the execution time is often computed asymptotically. The complexity estimation is often expressed in term of L-function:

$$L_q[s;c] = \exp\{(c + o(1))(\ln q)^s (\ln \ln q)^{1-s}\}$$

The first description of an index-calculus algorithm for extension fields $F^{*}_{p^n}$ with p fixed, is given by [11]. Blake *et al.* [12] made some improvements to index-calculus algorithm in $F^{*}_{2^n}$, although they do not improve the execution time asymptotically. The algorithm was substantially improved later by Coppersmith [13]. He estimated the expected execution time of the improved algorithm is $L_{2^n}[\frac{1}{3};c]$ for some c<1.587. In general, the Coppersmith algorithm can also be used for $GF(p^n)$ with asymptotic running time $L_{p^n}[\frac{1}{3};c]$ (with fixed p and $n\rightarrow\infty$). As shown above, the index-calculus method has ***sub-exponential*** running time for a variety of discrete logarithm problems. However, an important case where the index-calculus for treating the discrete logarithm problem has been unsuccessful is, *elliptic curves* over a finite field [31]. An extension of index-calculus method for ECDL (Elliptic Curve Discrete Logarithm), was introduced by Silverman[32]. This algorithm was analyzed in [21] and shown to be inefficient. In general, no attack more efficient than Pollard's ρ-method is known for ECDL [6].

2 A Novel Step: DLP in Symmetric Group S_n

By definition, S_n is a group of all bijections $H_n \rightarrow H_n$, where $H_n = \{1,2,\ldots,n\}$ and the group operation is ordinary composition of mappings. The elements of S_n are called permutations. Let $i_1,i_2,\ldots,i_k$, $k\leq n$ be distinct elements of H_n. Then, $(i_1,i_2,\ldots,i_k)$ denotes the permutation that maps $i_1 \rightarrow i_2$, $i_2 \rightarrow i_3,\ldots$, $i_{k-1} \rightarrow i_k$ and $i_k \rightarrow i_1$ and every other elements of H_n to itself. By definition, H_n is called a cycle of length k or a k-cycle, and a 2-cycle is denoted as *transposition*. It can be proven that if θ is a k-cycle, then $\theta^{\kappa}=1$. The permutations $\theta_1,\theta_2,\ldots,\theta_s$ of S_n are said to be disjoint if for every $1 \leq i \leq s$, and every $k\in H_n$, $\theta_i(k) \neq k$ implies $\theta_j(k)=k$ for all $j\neq i$. It is apparent that if cycles $\sigma_1, \sigma_2 \in S_n$ are disjoint, then $\sigma_1.\sigma_2=\sigma_2.\sigma_1$. The proof of the following theorems and corollaries are stated in [1] and [2].

Theorem 2.1. *Every nonidentity permutation θ of S_n can be uniquely expressed as the product of disjoint cycles of a length of at least 2.*

Corollary 2.2. *Let $\sigma \in S_n$ be the product of disjoint cycles $\theta_1,\theta_2,\ldots,\theta_k$ which θ_i is an m_i-cycle, i=1,…,k. Then the order of σ is $|\sigma|=lcm(m_1,m_2,\ldots,m_k)$.*

Therefore, in order to obtain the order of $\sigma\in S_n$, it is first decomposed to disjoint cycles and then the least common multiple of orders of its disjoint cycles are computed. Let $G_\theta = \{\theta^{i} \mid i=1,\ldots,|\theta|\}$ be the cyclic subgroup of S_n generated by $\theta \in S_n$. The solution is as follows:

$$\alpha^x = \beta \qquad \alpha,\beta \in S_n \qquad x \in Z^{\geq 0} \qquad (5)$$

where $\int^{\geq 0}$ denotes a set of nonnegative integers. If $\beta \notin G_\alpha$, then Eq. (5) has no solution. Therefore, discussion is restricted to $\beta \in G_\alpha$ and $0 \leq x \leq |\alpha|$. As with many other groups, a distinct mathematical solution for Eq. (5) in S_n is not available and it is examined by algorithms introduced in subsection 1.2. Note that $|G_\alpha|$ can be very large for large values of n, since the disjoint cycles of α can be selected in such a way that their least common multiple be very large. Therefore general purpose algorithms such as the brute-force search, Pollard's ρ-method and the baby-step giant-step algorithm with execution times of $O(|G_\alpha|)$, $O(\sqrt{|G_\alpha|})$ and $O(\sqrt{|G_\alpha|})$ respectively, are inefficient to solve Eq. (5). The Pohlig-Hellman algorithm uses the smoothness of the order of α. As already illustrated, the limitation that $|\alpha|$ should not be a smooth number has led to suggestions that 2^m-1 is a prime. Primes of the form 2^m-1 are called the Mersenne primes. The main drawback is the existence of wide gap between the consecutive Mersenne primes, and relatively few of them are known. Therefore, the idea is to use values of m for which 2^m-1 is not a prime and has a large prime factor. The factorization of 2^m-1 should be known for large enough of m's. Having 2^m-1, one cannot have an arbitrary degree of smoothness and in symmetric group S_n , cyclic subgroups with orders of arbitrary smoothness do exist.

Theorem 2.3. *Let p be a prime; then, for a large number of n's, a cyclic subgroup H of S_n which its order is p-smooth and is not (p-1)-smooth can be constructed.*

Proof. Let $n\geq p$ and a p-cycle $\sigma \in S_n$ exist. Also, let $\delta,\theta_1,\theta_2,\ldots,\theta_k$ be the disjoint cycles in which $\theta_i\in S_n$ is an m_i-cycle ($1 \leq m_i \leq p$, i=0,…,k), and $\sum_{i=1}^{k} m_i \leq n-p$. Then, for any permutation $\sigma=\delta\theta_1\theta_2\ldots\theta_k$, if q is a prime and $q|lcm(p,m_1,m_2,\ldots,m_k)=|G_\sigma|$, then $q \leq p$. Therefore the order of $H=G_\sigma$ is p-smooth, but it is not (p-1)-smooth, since $p \mid |G_\sigma|$ and p □ p-1.

According to Theorem 2.3, we can generate the base α so Eq. (5) can be resistant to attacks generated by Pohlig-Hellman algorithm. The basic step in the index-calculus method is to obtain the factor base B. Suppose that $\alpha=\theta_1\theta_2\ldots\theta_k$ which $\theta_1\theta_2\ldots\theta_k$ are disjoint cycles, then at first glance, one may select $B=\{\theta_1,\theta_2,\ldots,\theta_k\}$ since it is small and all the elements of G_α can be expressed as the products of elements of B. However, for any $1 \leq d \leq |\alpha|$, $\alpha^d = \prod_{i=1}^{k}\theta_i^d$, taking logarithms of both sides, we have $d = \sum_{i=1}^{k} d\log_\alpha\theta$, hence, $1 = \sum_{i=1}^{k}\log_\alpha\theta$. Clearly, this relation does not cover the logarithms of the B elements. Furthermore, in general, the logarithms of some elements of B may not exist at all. Suppose for example, $\alpha \in S_6$ and α = (1234)(56). Then, $|\alpha|$=4 and it is clear that there is no $x\in\{1, 2, 3, 4\}$, such that α^x=(1234) or α^x=(56).

Let C_i={*all the distinct i-cycle of S_n*} and $A_{S_n} = \bigcup_{i=1}^{n} C_i$. Theorem 2.1 states that every element of S_n can be expressed as the product of disjoint cycles. Therefore each elements of S_n can be expressed as the product of elements of A_{S_n}. This indicates that A_{S_n} can be chosen as another candidate for the factor base.

Theorem 2.4. *For* $n \geq 2$,:

$$|A_{S_n}| = e\int_1^{\infty} e^{-t}\frac{t^n - t}{t-1}dt - n + 2$$

Proof. It is easy to see that $|C_1|=1$ and $|C_i| = \frac{n!}{i(n-i)!} = \binom{n}{i}(i-1)!$. Thus:

$$|A_{S_n}| = 1 + \sum_{i=2}^{n}|C_i| = 1 + \sum_{i=2}^{n}\binom{n}{i}(i-1)!. \tag{6}$$

Let $a_n = \sum_{i=2}^{n}\binom{n}{i}(i-1)!$, then:

$$\begin{aligned}
a_{n-1} &= \sum_{i=2}^{n-1}\binom{n-1}{i}(i-1)! \\
&= \sum_{i=2}^{n-1}\frac{n-i}{n}\binom{n}{i}(i-1)! \\
&= \sum_{i=2}^{n}\frac{n-i}{n}\binom{n}{i}(i-1)! \\
&= a_n - \frac{1}{n}\sum_{i=2}^{n}\binom{n}{i}i! \\
&= a_n - (n-1)!\sum_{i=2}^{n}\frac{1}{(n-i)!} \\
&= a_n - (n-1)!\{e - \sum_{k=n-1}^{\infty}\frac{1}{k!}\} \\
&= a_n - (n-1)!e - (1 + \frac{1}{n} + \frac{1}{n(n+1)} + \mathrm{L}\ )
\end{aligned}$$

Let $b_n = 1 + \frac{1}{n} + \frac{1}{n(n+1)} + \mathrm{L}$, then $a_{n-1} = a_n - (n-1)!e + b_n$ or:

$$a_n = a_{n-1} + (n-1)!e - b_n. \tag{7}$$

We can rewrite b_n as $b_n = 1 + \frac{1}{n}(1 + \frac{1}{n+1}(1 + \mathrm{L}\ )\mathrm{L}\ )$. It is clear that $b_n = 1 + \frac{1}{n}b_{n+1}$ or;

$$b_{n+1} = \begin{cases} e & n = 1 \\ nb_n - n & n > 1 \end{cases} \tag{8}$$

By expanding Eq. (8), we obtain:

$$b_n = (n-1)!e - \sum_{i=0}^{n-2}\prod_{j=0}^{i}(n-j) = (n-1)!e - e\Gamma(n,1) + 1 \tag{9}$$

in which $\Gamma(x,y) = \int_y^{\infty} e^{-t}t^{(x-1)}dt$. Using Eq. (9) in Eq. (7), produces $a_n = a_{n-1} + \Gamma(n,1) - 1$. Thus:

$$\begin{aligned}
a_n &= \sum_{k=2}^{n} e\Gamma(k,1) - n + 1 \\
&= e\sum_{k=2}^{n}\int_1^{\infty} e^{-t}t^{k-1}dt - n + 1 \\
&= e\int_1^{\infty} e^{-t}\{\sum_{k=2}^{n}t^{k-1}\}dt - n + 1 \\
&= e\int_1^{\infty} e^{-t}\frac{t^n - t}{t-1}dt - n + 1
\end{aligned}$$

and from Eq. (6) we obtain $|A_{S_n}| = a_n + 1$ which leads to the desired result.

Theorem 2.5. *For n≥2,* $|A_{S_n}| / |S_n| \leq 1$ *and*

$$\lim_{n\to\infty}\frac{|A_{S_n}|}{|S_n|} = 0$$

Proof. From Eq. (6) of the Theorem 2.4 we have:

$$\frac{|A_{S_n}|}{|S_n|} = \frac{1}{n!} + \frac{1}{n!}\sum_{i=2}^{n}\binom{n}{i}(i-1)! = \frac{1}{n!} + \sum_{i=2}^{n}\frac{1}{i(n-i)!} \tag{10}$$

Let $a_n = \sum_{i=2}^{n}\frac{1}{i(n-i)!}$, then;

$$a_{n-1} = \sum_{i=2}^{n-1}\frac{1}{i(n-i-1)!} = \sum_{i=2}^{n}\frac{n-i}{i(n-i)!} = na_n - \sum_{i=2}^{n}\frac{1}{(n-i)!}$$

It is apparent that $\sum_{i=2}^{n}\frac{1}{(n-i)!} \leq e$, therefore:

$$a_n \leq \frac{1}{n}a_{n-1} + \frac{e}{n} \qquad a_2 = \frac{1}{2} \tag{11}$$

Expansion of Eq. (11) results in:

$$\begin{aligned}
a_n &\leq \frac{1}{n}a_{n-1} + \frac{e}{n}(1 + \frac{1}{n-1} + \frac{1}{(n-1)(n-2)} + \\
&\mathrm{L}\ + \frac{1}{(n-1)!}) \leq \frac{1}{n!} + \frac{e^2}{n}
\end{aligned} \tag{12}$$

From Eqs. (12) and (10) respectively, we have $0 < |A_{S_n}| / |S_n| = \frac{1}{n!} + a_n \leq \frac{2}{n!} + \frac{e^2}{n}$ hence $\lim_{n\to\infty}\frac{|A_{S_n}|}{|S_n|} = 0$.

According to the Theorem 2.5, the set A_{S_n} is small relative to S_n , specially for large n's (Fig. 1). Therefore, it seems to be appropriate for a factor base. However, the main problem is the size of S_n. The $|S_n|=n!$ is very big even for a moderate n's, and although $|A_{S_n}|$ is small relative to $|S_n|$, it does not mean $|A_{S_n}|$ is small itself. For example $|A_{S_{50}}|$=1687944359857381953729251824536320980391731190310161954249338066, and $|A_{S_{100}}| > 10^{156}$. For $n>10^4$, which is a typical value in Eq. (5), the size of A_{S_n} will be intractable. However, an algorithm that can select an appropriate subset of A_{S_n} for factor base is not yet available and it is likely to be difficult to develop such an algorithm.

3 Integer Representation of S_n

There have been several attempts to introduce an effective permutations representation [29]. In this section, a bijection between integers and elements of symmetric groups is introduced which enables to represent the elements of symmetric groups by integers, and vice versa.

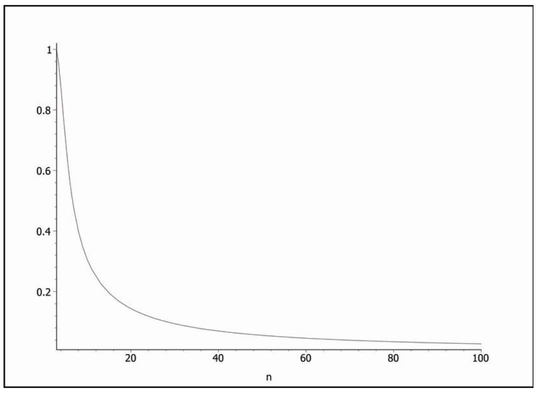

Fig. 1: Vertical and horizontal axes show $|A_{S_n}|/|S_n|$ and n respectively.

3.1 Factoradic: A Mixed Radix Number System

A number system is a framework for representing numerals. Number systems can be divided into two general categories: a fixed radix and mixed radix number systems. In fixed radix number systems, a positive integer is represented as:

$$a_n r^n + a_{n-1} r^{n-1} + \mathrm{L} + a_1 r + a_0 \qquad 0 \le a_i < r \quad i = 0, \mathrm{K}, n$$

where r is known as the radix or base. This system is also called a positional base system in which the digit weights are r^n, r^{n-1},..., r^1, r^0. Now consider a positional base in which each value is a factorial. Every number in this system is represented as:

$$a_n n! + a_{n-1}(n-1)! + \mathrm{L} + a_1 1! \qquad 0 \le a_i \le i \quad i = 1, \mathrm{K}, n \tag{13}$$

Here the radices are n, n-1,…,1 and the place values are $n!$, $(n$-$1)!$,…,$1!$. This mixed radix number system is called *Factoradic* and the k-digit representation $d_k d_{k-1} \ldots d_2 d_1$ of an integer is called the k-factoradic. Eq. (13) can be rewritten as $1!(a_1 + 2(a_2 + \mathrm{L} + n(a_n)\mathrm{L})$, so transforming an integer m to this system can be performed similar to fixed radix systems i.e., by successive divisions where divisions are performed by 2, 3, 4 and so on. The following describes some properties of the factorial base system.

Lemma 3.1. *If* $a = d_n d_{n-1} \ldots d_2 d_1$ *is a number in factorial base system, then* $0 \le a \le (n+1)!$-1.

Proof. Since $d_i \ge 0$, i=1,…,n, thus $a = \sum_{i=1}^{n} d_i i! \ge \sum_{i=1}^{n} 0 \times i! = 0$. On the other hand, $d_i \le i$, i =1, 2, …, n. Therefore:

$$\begin{aligned} a = \sum_{i=1}^{n} d_i i! &\le \sum_{i=1}^{n} i \times i! \\ &= \sum_{i=1}^{n} ((i+1)-1) \times i! \\ &= \sum_{i=1}^{n} ((i+1)! - i!) \\ &= (n+1)! - 1 \end{aligned}$$

and the proof is complete.

Corollary 3.2. *If* $a = d_n d_{n-1} \ldots d_2 d_1$ *is a number in factorial base system, then* $d_n.n! \le a \le (d_n+1).n!$.

Proof. Let $b = d_{n-1} d_{n-2} \ldots d_2 d_1$. From Lemma 3.2, it is known that $0 \le b < n!$. We also have $a = d_n n! + b$, hence $d_n.n! \le a \le (d_n+1).n!$.

Like any other numbering systems, assume there is no leading zero in factorial base representation. That is, zero has unique representation $0 \times 1!$. The following theorem states the one-to-oneness between positive integers and factorial base numbers.

Theorem 3.3. *Every positive integer* α *has a unique representation in a factorial base system.*

Proof. Let $\alpha = d_m d_{m-1} \ldots d_2 d_1$, $d_m \neq 0$ and $\alpha' = d'_m d'_{m-1} \ldots d'_2 d'_1$, $d'_m \neq 0$ be two representations of a positive integer α. If $n<m$, then by Lemma 3.1 and Corollary 3.2, $a \ge d_m m! \ge m! \ge (n+1)! > a'$. Therefore $a > a'$ and similarly $m<n$ become a contradiction and hence $n=m$. The proof will be limited to a and a' having the same number of digits. Induction on number of digits is employed. If d_1 and d'_1 are two representations of α then $d_1 \times 1! = d'_1 \times 1!$. Thus, $d_1 = d'_1$, and the two representations are the same. Now suppose that if a positive integer α has a k-digit representation, then this representation becomes unique. Let $b = d_{k+1} d_k \ldots d_2 d_1$, $d_{k+1} \neq 0$ and $b' = d'_{k+1} d'_k \ldots d'_2 d'_1$, $d'_{k+1} \neq 0$ be the two representations for positive integer β. If $d_{k+1} > d'_{k+1}$, then by Corollary 3.2, $b \ge d_{k+1}(k+1)! \ge (d'_{k+1}+1)(k+1)! > b'$ yields to a contradiction. By a similar discussion, $d_{k+1} < d'_{k+1}$ does not hold. Therefore, $d_{k+1} = d'_{k+1}$ yields to b-$d_{k+1}(k+1)! = b'$-$d'_{k+1}(k+1)!$. Thus $d_k d_{k-1} \ldots d_2 d_1$ and $d'_k d'_{k-1} \ldots d'_2 d'_1$ represent the same integer and by the induction hypothesis, they are the same. That is, $d_k = d'_k$,…, $d_1 = d'_1$. Therefore, it has been shown that b and b' are the same representations of β and the proof is complete.

3.2 Mapping Factorial Base Numbers to Members of S_n

A basic idea for representing permutations is the use of *subexceedant functions.*

Definition 3.4. *Suppose that* $I_n = \{1,\ldots,n\}$. *A subexceedant function f on* I_n *is a map* $f : I_n \, \mathrm{a} \, I_n$ *such that*

$$1 \le f(i) \le i \quad \text{for all } 1 \le i \le n.$$

To simplify the discussion, it is assumed that a subexceedant function f on I_n to be the map of $f : I_n \text{ a } I_n \cup \{0\}$ such that $0 \leq f(i) \leq i$ *for all* $1 \leq i \leq n$. Let Φ_n be a set of all subexceedant functions on I_n and the subexceedant function f over I_n is represented by the word $f(n).f(n\text{-}1)\ldots f(1)$. For example, f=201 is a subexceedant function over I_3 in which $f(3)=2$, $f(2)=0$ and $f(1)=1$. It is apparent that $|\Phi_n| = (n+1)!$. In the following, a bijection between Φ_n and S_{n+1}, originated from [23], will be introduced.

Lemma 3.5. *Let $\psi : \mathrm{F}_n \text{ a } S_{n+1}$ be a map associated with the subexceedant function f. The permutation θ_f defined as the product of transpositions by: $\theta_f=(1\ f(1))(2\ f(2))\ldots(n\ f(n))$, then ψ is a bijection from Φ_n onto S_{n+1}.*

Proof. Since $|\Phi_n| = |S_{n+1}| = (n+1)!$, it is sufficient to prove that ψ is injective. Let f_1, $f_2 \in \Phi_n$, and $\psi(f_1) = \psi(f_2)$. Thus:

$$\begin{aligned}&(1\, f_1(1))(2\, f_1(2))\mathrm{L}\ (n\, f_1(n)) = \\ &(1\, f_2(1))(2\, f_2(2))\mathrm{L}\ (n\, f_2(n))\end{aligned} \qquad (14)$$

Since $\psi(f_1) = \psi(f_2)$, then $\psi(f_1)(n) = \psi(f_2)(n)$. By definition, $\psi(f_1)(n) = f_1(n)$ and $\psi(f_2)(n) = f_2(n)$, so $f_1(n) = f_2(n)$. If both sides of (14) are multiplied on the right, by permutation, $(n\, f_1(n)) = (n\, f_2(n))$,. Therefore:

$$\begin{aligned}&(1\, f_1(1))(2\, f_1(2))\mathrm{L}\ ((n-1)\, f_1(n-1)) = \\ &(1\, f_2(1))(2\, f_2(2))\mathrm{L}\ ((n-1)\, f_2(n-1))\end{aligned} \qquad (15)$$

Now, applying the same process to (15) leads to $f_1(n\text{-}1) = f_2(n\text{-}1)$. Iteration concludes that $f_1(i) = f_2(i)$ for all $1 \leq i \leq n$. Thus $f_1 = f_2$.

Let θ be the permutation of symmetric group S_{n+1} and $f=\psi^{-1}(\theta)$. Then f can be constructed as below:

- Set $f(n)=\theta(n)$.
- Multiply θ on the right by transposition $(n\ \ f(n))$ in order to obtain new permutation θ_1 in which $n \text{ a } n$. Thus, θ_1 can be considered as a permutation of S_n. Then set $f(n\text{-}1)=\theta_1(n\text{-}1)$.
- Apply the same process to θ_1 by multiplying by $((n\text{-}1)\ f(n\text{-}1))$, in order to obtain $f(n\text{-}2)$. By iteration, $f(i)$ will be determined for all the $1 \leq i \leq n$.

Now, let $a=d_n d_{n-1}\ldots d_2 d_1$ be an n-digit number in factorial base system. It is known by definition that $0 \leq d_i \leq i$, for all $1 \leq i \leq n$. Thus, $a=d_n d_{n-1}\ldots d_2 d_1$ can be interpreted as a subexceedant function representation $f(n)\mathrm{f}(n\text{-}1)\ldots\mathrm{f}(1)$, in which
$f(n)= d_n$, $f(n\text{-}1)=d_{n\text{-}1},\ldots, f(1)=d_1$. Therefore, according to lemma 3.5, the number $a=d_n d_{n-1}\ldots d_2 d_1$ can be uniquely mapped to a permutation $\theta \in S_{n+1}$. In this way, there is a bijection between n-digit factorial base numbers and S_{n+1} for all $n \in$.
$\otimes$

4 Description of the Proposed Cryptosystem

In an appropriate level of abstraction, the algebraic elements such as groups and fields have similar structures. Consequently, many of the cryptosystems based on finite groups or finite fields $GF(p^n)$ can be translated to systems using the symmetric group S_n. The Proposed Cryptosystem is described by illustrating the symmetric group analog of the Generalized El-Gamal system [17]. However it is perfectly practical to also describe other systems such as Diffie-Hellman and Massey-Omura [25] based on our proposed cryptosystem. Suppose user "A" requires sending a message m to user "B". The process of key selection, encryption and decryption in the proposed cryptosystem is as follows:

Key Selection: User "B' requires the followings:

- Select a large n for S_n.
- Generate an appropriate $\theta \in S_n$, which will be the generator of G_θ .
- Select a random integer $1 \leq a \leq |G_\theta|\text{-}1$ and compute θ^a.
- Publish $(\theta\ ,\theta^{\,a})$ as a public key, and keep a as a private key

Encryption: User "A" encrypts the message m as follows:

- Translate m to $m' \in S_n$ (according to section 3).
- Select a random integer $1 \leq k \leq n$.
- Compute the pair $(\theta_1=\theta^k,\ \theta_2= m'.(\theta^a)^k)$, and transmit to "B".

Decryption: Now, "B" is able to decrypt the message as follows:

- Compute $(\theta_1)^a=(\theta^k)^a= \theta^{k.a}$, and then be reversed in order to obtain $((\theta_1)^a)^{-1} = \theta^{\,\text{-}k.a}$.
- Compute m' by multiplying θ_2 on the right by θ_1^{-a} .
- Recover m by computing the integer representation of m'.

At the end of this section, a small example is presented which clarifies the above procedure. Here, some technical aspects of above scheme are presented.

- The multiplication in S_n which is the composition of mappings can be implemented by just n assignments.
- There are many optimized methods for exponentiation, such as *Right to left*, *k-Ary* and *Sliding Window* [18] , which can be used for any multiplicative group (commutative/ non-commutative). Therefore, they can also be used in symmetric group S_n.
- The generator θ in the above scheme can be generated in a very easy and low cost method, so the scheme could be arbitrarily is resistant to attacks by algorithms such as Pohlig-Hellman. The selection of θ in the proposed scheme has the lowest complexity among other known schemes.
- The only disadvantage of this scheme is the relatively large memory requirement and organization of permutations such as θ, requiring large bandwidth for the transmission of system parameters.

It is noteworthy that despite this disadvantage, the proposed system preserves its high security and power in its abstract and the theoretical aspects. It can foster new ideas; since according to our proposition, working on the S_n group can leads to cryptosystems with much closer to unconditional security than other existing cryptosystems. Also, it could be a framework for comparison and rating with other implementable systems. However further research are performing in order to reduce the memory requirements and make the scheme practical. At present, the aim is to prove theoretically the structure of the symmetric group S_n is appropriate for employment in cryptosystems such as the Generalized El-Gamal. This has not been considered till now. The proposed cryptosystem has been implemented in Java™ (1.6) at an Athlon-64 (3.2GHz) AMD machine, and also tested on one million bits permutations. An average execution time gained for the group operation was about 455 microseconds.

Example: Suppose that user "B" selects n=100 for S_n, and generates θ as follows:
θ=(0···22)(23···41)(42···58)(59···71)(72···82)(83···89)(90···94)(95 96 97)(98 99). Thus the user has $|G_\theta|$=223092870 and publishes $(\theta, \theta^{546584})$ as the public key while keeping a=546584 as the private key. Computation of θ^{546584} is very easy, and can be performed using an algorithm such as Right to Left Exponentiation in $O(\log_2 a)$ multiplications [18]. User "B" has θ^{546584} =
12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 34, 35, 36, 37, 38, 39, 40, 41, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 71, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 77, 78, 79, 80, 81, 82, 72, 73, 74, 75, 76, 86, 87, 88, 89, 83, 84, 85, 94, 90, 91, 92, 93, 97, 95, 96, 98, 99. Suppose "A" requires to send the message m="*The quick brown fox jumps over the lazy dog*" to "B". This message is a sequence of bytes, which is interpreted as an integer
m=11815744420664747200359014215611078249874077418792906203758916158211866334739307190174417697959789752167. Thus, the factoradic form of m is: m=13, 63, 28, 32, 53, 57, 33, 2, 61, 18, 27, 5, 21, 9, 57, 23, 4, 13, 50, 37, 23, 30, 25, 21, 34, 19, 12, 33, 37, 32, 28, 20, 26, 22, 23, 31, 20, 28, 24, 29, 18, 26, 16, 13, 10, 0, 13, 16, 22, 12, 21, 15, 2, 7, 13, 16, 5, 2, 4, 2, 3, 10, 5, 8, 2, 2, 4, 0, 1, 0, 1; which is in fact a subexceedant function representation, and according to subsection 3.2 it can be transformed by the permutation:
m= 6, 11, 58, 1, 67, 17, 36, 43, 8, 35, 70, 3, 14, 55, 46, 60, 44, 49, 7, 64, 15, 48, 45, 38, 42, 47, 72, 10, 54, 16, 39, 62, 29, 24, 41, 40, 31, 51, 22, 26, 20, 69, 68, 52, 65, 12, 19, 34, 59, 25, 30, 56, 37, 50, 71, 4, 23, 66, 9, 21, 5, 27, 18, 61, 2, 33, 57, 53, 32, 28, 63, 13, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 0. User "A" selects k=87493 and computes $(\theta^{546584})^k$=12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 39, 40, 41, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 69, 70, 71, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 78, 79, 80, 81, 82, 72, 73, 74, 75, 76, 77, 83, 84, 85, 86, 87, 88, 89, 92, 93, 94, 90, 91, 97, 95, 96, 98, 99; and $m.(\theta^{546584})^k$= 18, 0, 58, 13, 64, 6, 33, 43, 20, 32, 67, 15, 3, 55, 46, 70, 44, 49, 19, 61, 4, 48, 45, 35, 42, 47, 78, 22, 54, 5, 36, 59, 26, 40, 38, 37, 28, 51, 11, 23, 9, 66, 65, 52, 62, 1, 8, 31, 69, 41, 27, 56, 34, 50, 68, 16, 39, 63, 21, 10, 17, 24, 7, 71, 14, 30, 57, 53, 29, 25, 60, 2, 79, 80, 81, 82, 72, 73, 74, 75, 76, 77, 83, 84, 85, 86, 87, 88, 89, 92, 93, 94, 90, 91, 97, 95, 96, 98, 99, 12; and θ^k=1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 0, 40, 41, 23,24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 53, 54, 55, 56, 57, 58,42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 59, 60, 61, 82, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 83, 84, 85, 86, 87, 88, 89, 93, 94, 90,91, 92, 96, 97, 95, 99, 98. Then "A" sends the pair $(m.(\theta^{546584})^k, \theta^k)$ to "B". User "B" then decrypts the message as follows:
$(\theta^k)^{-546584}$=11,12,13,14,15,16,17,18,19,20,21,22, 0,1, 2,3,4,5,6,7,8,9,10,26,27,28,29,30,31,32,33,34,35,36,37,38, 39,40,41,23,24,25,42,43,44,45,46,47,48,49,50,51,52,53,54, 55,56,57,58,62,63,64,65,66,67,68,69,70,71,59,60,61,77,78, 79,80,81,82,72,73,74,75,76,83,84,85,86,87,88,89,93,94,90, 91,92,96,97,95,98,99.
$m.(\theta^{546584})^k.(\theta^k)^{-546584}$=
6,11,58,1,67,17,36,43,8,35,70,3,14,55,46,60,44,49,7,64,15, 48,45,38,42,47,72,10,54,16,39,62,29,24,41,40,31,51,22,26, 20,69,68,52,65,12,19,34,59,25,30,56,37,50,71,4,23,66,9,21 ,5,27,18,61,2,33,57,53,32,28,63,13,73,74,75,76,77,78,79,8 0,81,82,83,84,85,86,87,88,89,90,91,92,93,94,95,96,97,98,9 9,0.

5 Conclusions

There exist a variety of commutative and non commutative algebraic structures that have been proposed as a basis for the public key cryptosystems, and their theoretical correctness and security has proven to be related to known difficult problems such as DLP or factorization. In this paper, a public-key cryptosystem which is based on the symmetric group S_n has been proposed. The scheme has some important properties such as non commutative, high flexibility for selecting keys that makes DLP more resistant to known attacks, and added advantages of easy and fast implementation.

The main idea for implementation of this abstract scheme is to establish a bijection between elements of S_n and natural numbers. Therefore, any data stream with a finite length can be represented by a unique $\theta \in S_n$ for some n. The Data encryption and decryption is performed using the symmetric group analog of Generalized El-Gamal Cryptosystem. Also, the difficulty of solving DLP in the context of proposed scheme leads to higher security and the system parameters can be selected in an easy and flexible way, so that the cryptanalysis of the proposed scheme will be fairly difficult than other existing schemes. Moreover, from the implementation points of view, the multiplication and inversion in S_n can easily be performed in $O(n)$ assignments, which is very fast. The only negative point of the system has shown to be the relative large memory and bandwidth requirements for storing and transmitting permutations. The authors are seeking further research in order to reduce this limitation.

References

[1] I.N. Herstein, *Topics in Algebra*, John Wiley & Sons, second edition, New York, (1975).
[2] J.J. Rotman, *Advanced Modern Algebra*, Prentice Hall, 1st edition, New Jersey, (2002).
[3] M.E. HELLMAN,*A cryptanalytic time-memory tradeoff*, IEEE Transactions on Information Theory,26 (1980), 401-406.
[4] D. Shanks. *Class number, a theory of factorization, and genera*. In Proceedings of Symposia in Pure Mathematics, volume 20,(1969), 415-440.
[5] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, second edition, (2001).
[6] J.M. POLLARD, *Monte Carlo methods for index computation (mod p)*, Mathematics of Computation,32 (1978), 918-924.
[7] P.C. van Oorschot and M. J. Wiener. *Parallel collision search with cryptanalytic applications*, J. Cryptology,12 (1999), 1-28.
[8] S.C. POHLIG AND M.E. HELLMAN, *An improved algorithm for computing logarithms over GF(p) and its cryptographic significance*,IEEE Transactions on Information Theory, 24(1978), 106-110.
[9] K. McCurley, *The discrete logarithm problem*, Cryptology and Computational Number Theory, volume 42 of Proceedings of Symposia in Applied Mathematics, American Mathematical Society, (1990), 49-74.
[10] L. M. Adleman, *A sub-exponential algorithm for the discrete logarithm problem with applications to cryptography*. In 20th IEEE Annual Symposium on Foundations of Computer Science, (1979), 49-74.
[11] M.E. HELLMAN AND J.M. REYNERI,*Fast computation of discrete logarithms in GF(q)*, Advances in Cryptology Proceedings of Crypto 82, (1983), 3-13.
[12] I.F. BLAKE, R. FUJI-HARA, R.C. MULLIN, AND S.A. VANSTONE, *Computing logarithms in finite fields of characteristic two*, SIAM Journal on Algebraic and Discrete Methods, 5 (1984), 276-285.
[13] D. COPPERSMITH, *Fast evaluation of logarithms in fields of characteristic two*, IEEE Transactions on Information Theory, 30 (1984), 587-594.
[14] W. Diffie and M. Hellman, *New directions in cryptography*, IEEE Transactions on Information Theory, 22 (1976), 644-654.
[15] U. Maurer and S. Wolf, *The relationship between breaking the Diffe-Hellman protocol and computing discrete logarithms*, SIAM Journal on Computing, 28(5) (1999),1689-1731.
[16] D. Boneh and R. Lipton, *Algorithms for black-box fields and their applications to cryptography*, Advances in Cryptology-CRYPTO '96, Lecture Notes in Computer Science, Springer-Verlag, 1109 (1996), 283-297.
[17] T. El-Gamal, *A public key cryptosystem and a signature scheme based on discrete logarithms*, IEEE Transactions on Information Theory, 31 (1985), 469-472.
[18] A. J. Menezes, P. C. van Oorschot, S. A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, Inc., third edition, (1997).
[19] A. L. Wells, Jr., *A polynomial form for logarithms modulo a prime*, IEEE Transactions on Information Theory. 30 (1984), 845-846.
[20] M. E. Hellman and R. C. Merkle, *Hiding information and signatures in trapdoor knapsacks*, IEEE Transactions on Information Theory, 24 (1978), 525-530.
[21] M.J. Jacobson, N. Koblitz, J.H. Silverman, A. Stein, and E. Teske, *Analysis of the Xedni calculus attack*, Design, Codes and Cryptography, 20 (2000), 41-64.
[22] A. Shamir, *A polynomial time algorithm for breaking the basic Merkle-Hellman cryptosystem*, Proceedings of the 23rd Annual Symposium on Foundations of Computer Science, IEEE Computer Society Press, (1982), 145-152.
[23] R. Mantaci and F. Rakotondrajao, *A permutation representation that knows what "Eulerian" means*, Discrete Mathematics and Theoretical Computer Science, 4 (2001), 101-108.
[24] B. Chor and R. Rivest, *A knapsack-type public key cryptosystem based on arithmetic in finite fields*, IEEE Transactions on Information Theory, 34 (1988), 901-909.
[25] P. K. S. Wah and M. Z. Wang, *Realization and application of the Massey-Omura lock*, in Proc. Intern. Zurich Seminar, March 6-8, (1984), 175-182.
[26] S. Vaudenay, *Cryptanalysis of the Chor-Rivest Cryptosystem*, Advances in Cryptology-CRYPTO '98, Lecture Notes in Computer Science, 1462 Springer-Verlag, (1998), 243-256.
[27] R. L. Rivest, A. Shamir and L. Adleman, *A method for obtaining digital signatures and public key cryptosystems*, Communication of the ACM, 21 (1978), 120-126.
[28] M.O. RABIN, *Digitalized signatures and publickey functions as intractable as factorization*, MIT/LCS/TR-212, MIT Laboratory for Computer Science, (1979).
[29] D. H. Lehmer, *Teaching combinatorial tricks to a computer*, Proc. Sympos. Appl. Math. Combinatorial Analysis, Vol. 10, Amer. Math. Soc., Providence, R. I., (1960), 179-193.
[30] B.S. KALISKI JR. AND M. ROBSHAW, *The secure use of RSA*, CryptoBytes, 1 (Autumn 1995), 7-13.
[31] M.-D. Huang, K. L. Kueh, and K.-S. Tan, *Lifting*

elliptic curves and solving the elliptic curve discrete logarithm problem, In ANTS, Lecture Notes in Computer Science, Volume 1838 Springer-Verlag, (2000).

[32] J. Silverman, *The xedni calculus and the elliptic curve discrete logarithm problem*, Designs, Codes and Cryptography, 20 (2000), 5-40.

[33] N. Koblitz, *Elliptic curve cryptosystems*, Mathematics of Computation, 48 (1987), 203-209.

[34] V. Miller, *Uses of elliptic curves in cryptography*, Advances in Cryptology- CRYPTO '85, Lecture Notes in Computer Science, Springer-Verlag, 218 (1986), 417-426.

J.N.Nazari received the B.S. degree in Computer Engineering from Kharazmi (teacher training university) university, Tehran, Iran in 2008. His main research interest is Information theory (specially Data compression).

E. Malekian received the B.S. degree in Computer Engineering from Isfahan University of Technology, Isfahan, Iran in 1993 and M.S. degree from Shiraz University, Shiraz, Iran in 1995. He is now pursuing his PhD at the University of Shahid Beheshti, Tehran, Iran. His research interests include Data Security, Computer Networks, Algebraic Model and Arithmetic Architecture of public-key cryptosystems.

Ali Zakerolhosseini received the BSC degree from the University of Coventry, UK, in 1985, MSc from the Bradford University, UK, in 1987, and PhD degree in Fast transforms from the University of Kent, UK, in 1998. He is currently been an associate professor in the Department of Electrical and Computer Engineering at Shahid Beheshti University, Iran. His research focuses on Reconfigurable devices and Multi classifiers. His current research interests are Data Security, reconfigurable devices and parallel processing