A NON-ABELIAN FACTORIZATION PROBLEM AND AN ASSOCIATED CRYPTOSYSTEM

SRINATH BABA, SRINIVAS KOTYADA, AND RAGHU TEJA

Abstract. In this note, we define a cryptosystem based on non-commutative properties of groups. The cryptosystem is based on the hardness of the problem of factoring over these groups. This problem, interestingly, boils down to discrete logarithm problem on some Abelian groups. Further, we illustrate this method in three different non-Abelian groups $\mathrm{GL}_n(\mathbb{F}_q)$, $\mathrm{UT}_n(\mathbb{F}_q)$ and the Braid Groups.

1. Introduction

The discrete logarithm problem (DLP) in a group G is the following:

$$\text{Given } g \in G, g^x \in G, \text{ find } x.$$

Natural choices for G are cyclic subgroups of algebraic groups, for example, the multiplicative group of a finite field, the set of rational points on an Abelian variety, or linear groups over finite fields. Investigations about the complexity of this problem have been carried out in many contexts. The hardness of solving this problem is the basis for many public key cryptosystems (see [6]) .

The noticeable feature in all of these investigations is the absence of the use of non-commutation properties. In this note, we demonstrate how to define a similar "one-way" function (FACTOR) in a non-Abelian group. As examples we suggest that one considers groups like $\mathrm{GL}_n(\mathbb{F}_q)$, $\mathrm{UT}_n(\mathbb{F}_q)$, Braid Groups.

Analysis of the FACTOR function in these groups show that it is an approximate one-way function. This means that the inverse to the FACTOR is easy to compute, while the function itself is hard to compute. This sets us up for a cryptographic application.

Using FACTOR function as a primitive we therefore define a Public Key Cryptosystem. We define an encryption/decryption mechanism which is comparable to the El-Gamal system in cryptography using the discrete logarithm problem. The El-Gamal system can be described as follows: *Let G be a finite cyclic group with generator g, and Let Alice have a public key which consists of G,g,g^x where x is Alice's private key. To send a message $m \in G$, Bob picks an integer y and sends the cipher text $g^y, g^{xy}m$ to Alice. To decrypt, Alice calculates $(g^y)^x$ and inverts it to retrieve m.*

The El-Gamal cryptosystem is clearly easy to crack in any group where the DLP can be solved easily, because the numbers x and y can be solved for. In current practice, the El-Gamal system is used in the group of points of an Elliptic curve over a finite field and jacobians of hyper elliptic curves. We remark that the DLP

Key words and phrases. Cryptography, Discrete logarithm problem, Diffie-Hellman key exchange.

and related cryptosystems have been studied in cyclic subgroups of GL_n of finite fields, and have been shown to be computationally insecure (see [7]).

2. FACTOR and relations to the Discrete Logarithm Problem

Let G be any finite group with identity e. Let $g \in G$, $h \in G$ and let the cyclic subgroup generated by any element $x \in G$ be denoted by $< x >$. In order to define the FACTOR problem we assume that $< g > \cap < h >= \{e\}$. Let f be a function defined as follows:

$$f :< g > \times < h > \longrightarrow G; f(g^x, h^y) = g^x \cdot h^y \in G$$

We show that f is injective, because if not, then

$$\begin{array}{rcl} f(g^x, h^y) & = & f(g^a, h^b) \implies g^x h^y = g^a h^b \\ \implies & & g^x g^{-a} = h^b h^{-y} \implies g^x = g^a, h^y = h^b. \end{array}$$

Since f is injective, given $g^x h^y \in Imf$, the image of f, it makes sense to ask for $f^{-1}(g^x h^y)$. We define FACTOR as

$$\text{FACTOR}(g^x h^y) = f^{-1}(g^x h^y)$$

We now show that if G is Abelian, the FACTOR in G reduces to the DLP in G in some cases. Suppose the orders of g and h are known to be m and n respectively, which are coprime. Raising $g^x h^y$ to the power m yields:

$$(g^x h^y)^m = g^{mx} h^{my} = e \cdot h^{my} = h^{my}.$$

The discrete logarithm of h^{my} would yield my, and one can retrieve y. This argument, of course, fails if n and m have common factors. In particular, if G is cyclic, the problem reduces to the Discrete Logarithm Problem in the group G. To see this, let α be a generator of G. Then $g = \alpha^k$ and $h = \alpha^l$. Since $< g > \cap < h >= \{e\}$, we see that the orders of g and h would have to be relatively prime.

In particular, if $< g >$ is a normal subgroup of G, then computing FACTOR in G is the same as computing the image of quotient map $G \longrightarrow G/ < g >$.

We remark here that a natural way to attack FACTOR problem can be described as follows: If $\gamma = \alpha^l \beta^m$ is given with α and β non-commuting then one has to multiply α^{-1} to γ i.e. $\alpha^{-1}\gamma = \alpha^{l-1}\beta^m$ and check if it is a member of the cyclic group $< \beta >$. Hence if solving the membership problem is difficult, then solving FACTOR problem will be difficult too and this happens in the case of infinite groups. For finite groups if the choice of order of the group is very high, then this process of finding l, m is not efficient as the order of complexity is a linear function of l and m. Though this is a naive method there seems to be no better general method for solving it in an arbitrary group.

3. A Public Key Cryptosystem based on FACTOR

Let G be a non-Abelian group and let $g, h \in G$ be two non commuting elements with orders m, n respectively. Throughout this note we shall suppose that Alice is the recipient of the messages and Bob is communicating with Alice. Let $m \in G$ be the message.

Alice picks arbitrary integers (x, y) and sets a public key

PUBLIC KEY = $(G, g, h, g^x h^y)$

Alice has a private key for decryption

PRIVATE KEY $= (g^x, h^y)$

To send the message m, Bob picks arbitrary integers (x', y') and sends

CIPHER TEXT $= (g^{x+x'}h^{y+y'}, g^{x'}h^{y'}m) = (\tau_1, \tau_2)$

To decipher the text, Alice uses her private key and performs the following operations in G:

DECRYPTION : $h^y\tau_1^{-1}g^x\tau_2 = m$

Analysis of this scheme shows that the security of the crypto system described above reduces to solving FACTOR problem in the underlying group. The existing crypto schemes based on non-Abelian groups (see for example [3]) rely on the hardness of solving word problem or conjugacy problem. As the FACTOR problem is more difficult than the DLP problem, we hope this new scheme will pave way for a new range of possibilities in non-Abelian cryptography.

4. Sample Implementations

In this section we shall consider some non-Abelian groups and study the complexity of implementing the proposed crypto scheme in these groups.

4.1. Implementation in the group $\mathbf{GL}_n(\mathbb{F}_q)$. .

Let $G = \mathrm{GL}_n(\mathbb{F}_q)$, where $\mathbb{F}_q$ is a finite field of order q. Let $g, h \in G$ with orders k, l respectively. Assume also that $k = O(q)$, $l = O(q)$, and that g and h do not commute. Then Alice sets out the public keys and Bob communicates as follows:

Alice picks arbitrary integers (x, y) and an element $c \in \mathbb{F}_q$, and sets a public key

PUBLIC KEY $= (G, g, h, cg^xh^y)$

Alice has a private key for decryption

PRIVATE KEY $= (c, g^x, h^y)$

To send the message m, Bob picks arbitrary integers (x', y') and an element $c' \in \mathbb{F}_q$ and sends

CIPHER TEXT $= (cc'g^{x+x'}h^{y+y'}, c'g^{x'}h^{y'}m) = (\theta_1, \theta_2)$

To decipher the text, Alice carries out the following operations in G:

DECRYPTION : $ch^y\theta_1^{-1}g^x\theta_2 = m$

It must be noted that the constants $c, c' \in \mathbb{F}_q$ gives additional security to the system.

In what follows, we will show how encryption and decryption can be carried out, and also analyse the complexity of the operations involved.

4.1.1. *Complexity of encryption in $GL_n(\mathbb{F}_q)$.* Encryption basically involves calculating two objects: $g^{x'}$ and $h^{y'}$. Once both of these are calculated, the rest of the operations involved just consist of multiplication in G, which is clearly fast.

Therefore, all that remains, is to understand the complexity of exponentiation in G. This has been analysed (see, for example [7]) in the context of the DLP in G, and one knows that it can be carried out in polynomial time.

4.1.2. *Complexity of decryption in $GL_n(\mathbb{F}_q)$.* Decryption involves inverting θ_1. In $\mathrm{GL}_n(\mathbb{F}_q)$, this can be carried out using row reduction extremely fast. The next step is to carry out the sequence of multiplications, which are all fast. One concludes, therefore, that decryption can also be carried out in polynomial time.

4.1.3. *Security of the system.* We work with $\mathrm{GL}_n(\mathbb{F}_q)$ for a field $\mathbb{F}_q$ with q elements. We calculate the complexity of a simple attack on the system described above as a function of n and q.

The only attack is to crack the public key, i.e., to factorize $g^x h^y$. If both g and h are chosen with orders approximately $O(q)$, then, on the average, one would have to calculate $g^{-i} g^x h^y = g^{x-i} h^y$ for each i less that q. Moreover, one would have to test for each $g^{x-i} h^y$ if it lies in $< h >$. The simplest way to do this is to calculate the eigenvalues and eigenspaces of g and h. If $g^{x-i} h^y$ has the same eigenvalues/vectors as h, then it is probable that $x = i$.

On the other hand, if both g and h have the same eigenvalues/vectors, then the problem becomes even harder. In any case, this attack requires calculating at least $n^2 q$ quantities in $\mathbb{F}_q$, and is therefore exponential in $\log(n^2 q)$.

It should be remarked that the security of this system lies in the non-commutativity of g and h. A suitable choice of g and h could make all eigenvector calculations ineffective, in which case the attacker would have to calculate q^2 quantities. In addition, increasing n would give more choices for such elements. In fact, one could make this construction over any ring instead of a field.

4.2. **Implementation in the group $\mathbf{UT}_n(\mathbb{F}_q)$.** $\mathrm{UT}_n(\mathbb{F}_q)$ is a non-Abelian subgroup of the group $\mathrm{GL}_n(\mathbb{F}_q)$ consisting of uni-triangular matrices. The elements in $\mathrm{UT}_n(\mathbb{F}_q)$ are of the form

$$\begin{pmatrix} 1 & * & * & \dots & * \\ 0 & 1 & * & & * \\ 0 & 0 & 1 & & * \\ 0 & 0 & 0 & \ddots & * \\ 0 & 0 & 0 & \dots & 1 \end{pmatrix}$$

We shall work with an automorphism subgroup of $\mathrm{UT}_n(\mathbb{F}_q)$. The analysis in [4] suggests that solving DLP in $\mathrm{UT}_n(\mathbb{F}_q)$ reduces to solving DLP in $\mathbb{F}_q$ which is easy in some cases (see for example [8]). This is the reason to consider automorphism subgroup of $\mathrm{UT}_n(\mathbb{F}_q)$ rather than $\mathrm{UT}_n(\mathbb{F}_q)$.

The computations in $\mathrm{UT}_n(\mathbb{F}_q)$ are same as $\mathrm{GL}_n(\mathbb{F}_q)$. However, in some cases it is much efficient to work in $\mathrm{UT}_n(\mathbb{F}_q)$, since explicit formulae can be worked out for powers of the matrices.

The proposed crypto scheme in this case is as follows:

$G = \mathcal{I}_{UT_n(\mathbb{F}_q)}$ (Inner-automorphism subgroup of $UT_n(\mathbb{F}_q)$ and $\phi, \psi \in G$.

Alice chooses $c \in \mathbb{F}_q$ and $l, m \in \mathbb{Z}$ to form

PUBLIC KEY=$(G, \phi, \psi, c\phi^l \psi^m)$

Let $\xi \in UT_n(\mathbb{F}_q)$ be the message to be communicated. Bob chooses $c' \in \mathbb{F}_q$ and $l', m' \in \mathbb{Z}$ and encrypts the message as :

ENCRYPTED MESSAGE $= (\mu_1 \mu_2(\xi) = k)$

where $\mu_1 = cc'\phi^{l+l'}\psi^{m+m'}$ and $\mu_2 = c'\phi^{l'}\psi^{m'}$ (here clearly $\mu_1, \mu_2 \in G$).

The private key of Alice is:

PRIVATE KEY= (c, ϕ^l, ψ^m)

DECRYPTION $(c\psi^m \mu_1^{-1} \phi^l)(k) = m$.

Here again the incorporation of the constants c, c' in public key makes the system more secure.

Remark. As the group of inner automorphisms, $\mathcal{I}_{UT_n(\mathbb{F}_q)}$, are isomorphic to the underlying group, $\mathrm{UT}_n(\mathbb{F}_q)$, computations in both groups are same. Therefore, encryption and decryption can be carried out as in $\mathrm{GL}_n(\mathbb{F}_q)$. However, we have to perform only $(n-1)(n-2)/2$ operations as opposed to n^2 operations in $\mathrm{GL}_n(\mathbb{F}_q)$. This gives an advantage of better time complexity than $\mathrm{GL}_n(\mathbb{F}_q)$.

4.3. **Implementation in Braid Groups.** Braid groups have been introduced by E.Artin in his classic paper "Theory of Braids" (see [1]). The Theory of Braid groups has found many applications in combinatorics and Knot Theory which arose interest in their practical implementations. Currently many efficient implementations exist for Braid groups. These being easily computable non-Abelian groups could be an ideal choice for doing cryptography in a non-Abelian set up. Below we briefly recall the definition (see [1], [4]) and demonstrate our cryptoscheme in this group.

4.3.1. *Braid Group:* A Braid group B_n of order n with (Artin) generators $\sigma_i, 1 \leq i \leq n$ is defined by

$$B_n =< \sigma_1, \sigma_2, \dots, \sigma_n | \sigma_i\sigma_j = \sigma_j\sigma_i, \ |i-j| > 2 \text{ and } \sigma_i\sigma_{i+1}\sigma_i = \sigma_{i+1}\sigma_i\sigma_{i+1} >$$

The proposed cryptoscheme works as follows : Alice chooses $\mu, \nu \in B_n$ and $x, y \in \mathbb{N}$ and publishes

PUBLIC KEY $= (B_n, (\mu), (\nu), (\mu^x\nu^y))$.

Alice's private key is:

PRIVATE KEY $= ((\mu^x), (\nu^y))$.

Let the message be $\omega \in B_n$. Bob chooses $x', y' \in \mathbb{N}$ and sends following ciphertext:

CIPHER TEXT $= ((\mu^{x'}\nu^{y'}), (\mu^{x+x'}\nu^{y+y'}\omega)) = (\theta_1, \theta_2)$

Alice decrypts it as

DECRYPTION : $\nu^{-y}\theta_1^{-1}(\mu^x)^{-1}\theta_2 = \omega$

Various implementations are possible for Braids and the complexity of operations depend on the data structures we use to represent them. For current discussion we shall adopt the representation used in [2]. Below we suggest one of the possible way of encoding a message using this representation:

Let us suppose that our alphabet is of size l, then we choose the order of our Braid group to be $n = \mathcal{O}(\log(l))$. Now, each alphabet is transformed as follows: to each alphabet we assign a number less than l, and we take it's binary representation $((j_1 j_2 \dots j_n)_2)$ (some of the j's may be zero). This binary number is mapped to the following element of B_n

$$\sigma_1^{i_1}\sigma_2^{i_2}\dots\sigma_n^{i_n}$$

where $i_k = \begin{cases} 1 & , j_k = 1 \\ -1 & , j_k = 0 \end{cases}$ Note that this is already in the Artin canonical form (see [2]). As the canonical form is unique for any given Braid, the encoding steps are reversible to get back the original message.

4.4. **Complexity of encryption and decryption.** Encryption requires computing $\mu^{x+x'}$ and $\nu^{y+y'}$ and multiplying together with message $w \in B_n$ which is already in Artin canonical form. This takes $\mathcal{O}((\log(x'y')n)$ time. Similarly decryption requires $\mathcal{O}(n)$ operations as it involves three multiplications (since inverting μ^x and ν^y is a one time operation).

4.5. **Security of the system.** The security of the existing Braid group based crypto systems rely on the hardness of the conjugacy problem (for example see [5]). In our proposed scheme, since the security of the system boils down to solving the factor problem, we shall try to compute FACTOR($k = p^l q^m$). Following the approach described in section 2, we repeatedly multiply p^{-1} to k, at each stage analysing if $(p^{-1})k \in < q >$. Note that if q is not an Artin generator, then it is very difficult to solve and moreover, by choosing l and m large, we can make this problem even more intractable. Hence, atleast, a naive attack does not seem to work. We also remark here that the analysis in [3] shows that DLP is hard, this suggests that FACTOR problem is much harder and therefore, Braid group can be an ideal choice for this cryptosystem.

5. Generalized Discrete Logarithm Problems and Other Protocols

5.1. **Generalized FACTOR problem.** The definition of the FACTOR problem uses only two subgroups $< g >$ and $< h >$. One can extend this to many variables: If $g_1, \dots g_n \in G$ with $< g_i > \cap < g_{i+1} \dots g_n >= \{e\}$, then Generalized FACTOR is the inverse function to f, where f is defined as follows:

$$f : \prod_{i=1}^{n} < g_i > \longrightarrow G; f(g_1^{x_1}, \dots g_n^{x_n}) = \prod_{i=1}^{n} g_i^{x_i}$$

The condition on the intersection of the groups ensures that f is injective.

5.2. **Generalized public key cryptosystem.** In any group N, let G and H be two subgroups, and g and h be elements in the commutators of G and H respectively. Let the Public key be (N, G, H, gh), and the private key be the factors(g, h). Encryption of a message $m \in N$ would be a pair of the form

$$(xghy, xym) = (\Theta_1, \Theta_2)$$

where $x \in G$, $y \in H$. Decryption would involve calculating $h\Theta_1^{-1} g\Theta_2$.

5.3. **Generalized DLP.** In general, one can define a generalized discrete logarithm problem in G: Let g_i be as above. We have a function h with

$$h : \mathbb{Z}^n \longrightarrow G; h(x_1, \dots x_n) = \prod_{i=1}^{n} g_i^{x_i}$$

We define Generalized DLP as the inverse of h. In the situation $n = 1$, we recover the usual discrete logarithm problem in G.

5.4. **Key exchange protocol using FACTOR.** One can define a key exchange, analagous to the Diffie-Hellman key exchange protocol (see [6]) in a non-Abelian setting using FACTOR. Suppose Alice and Bob want to exchange keys. Suppose G, g, h are as in FACTOR. Let Alice pick two integers (x_A, y_A), and Bob pick two integers (x_B, y_B). Let Alice send the element $g^{x_A} h^{y_A}$ to Bob, and let Bob send $g^{x_B} h^{y_B}$. Both Alice and Bob can recover the element $g^{x_A + x_B} h^{y_A + y_B}$. This is their private key. An intruder would see $g^{x_A} h^{y_A}$ and $g^{x_B} h^{y_B}$ and would have to recover

$g^{x_A+y_A}h^{x_B+y_B}$. If g and h do not commute, this problem looks hard. Of course, a solution to FACTOR would break the security of this key exchange.

5.5. **Further remarks.** We hope one can construct an effective, implementable and highly secure cryptosystem using non-Abelian groups in the directions suggested in this paper. We conclude with the remark that one can define a non-Abelian analogue of most of the protocols using the DLP in Abelian groups, using either the Generalized DLP in this context, or FACTOR.

Acknowledgement Authors are thankful to Pradeep Kumar Mishra for going through the manuscript and suggesting improvements.

References

[1] Artin, E. Theory of Braids. *The Annals of Mathematics 48*, 1 (1947), 101–126.

[2] Jae Choon Cha, Ki Hyoung Ko, Sang Jin Lee, and Cheon, J. H. An efficient implementation of Braid groups. *ASIACRYPT LNCS 2248* (2001), 144–156.

[3] Ki Hyoung Ko, Sang Jin Lee, Jung Hee Cheon, Jae Woo Han, Ju-sung Kang and Choonsik Park New public-key cryptosystem using Braid groups. *Springer-Verlag* (2000), 166-183.

[4] Mahalanobis, A. A simple generalization of the elgammal cryptosystem to non-Abelian groups. *Communications in Algebra 36* (2008), 3878–3889.

[5] Mahlburg, K. An overview of Braid group cryptography. *"http://citeseer.ist.psu.edu/mahlburg04overview.html."* (2004).

[6] Menezes, A.J.; van Oorschot, P.C.; vanstone, S.A. Handbook of Applied Cryptography, *CRC Press Series on Discrete Mathematics and its applications. CRC Press*, 1997.

[7] Menezes, A., and Wu, Y.-H. The discrete logarithm problem in gl(n,q). *Ars Combin 47* (1997), 23–32.

[8] Neal Koblitz A course in Number theory and Cryptography, *Springer Verlag* (1994), 102–103.

Department of Mathematics and Statistics, Concordia University, Montreal, QC, Canada

E-mail address: sbaba@mathstat.concordia.ca

The Institute of Mathematical Sciences, C.I.T. Campus, Taramani, Chennai, India 600013

E-mail address: srini@imsc.res.in

Birla Institute of Technology and Science(BITS-Pilani), Vidya Vihar Campus, Pilani Rajasthan, India 333031

E-mail address: rteja.bits@gmail.com