New Cipher Algorithm Based on Multiple Quasigroups

Haythem Zorkta, and Tarek Kabani, *Member, IACSIT*

Abstract— A new symmetric cipher algorithm is introduced in this article. This algorithm is based on multiple quasigroups (QGs), constructed by special kind of mapping called a complete mapping. All these transformations are controlled by equations for encryption and decryption QGs. So, there is a total storage reduction and an ability to use huge number of QGs with large order. Moreover, this proposal is enhanced with a new cyclic random permutation (CRP) to construct randomly big number of QGs. Analytical and comparative study concerned the proposal, was achieved. It proved its strength (key length 192 bits), its speed (encryption times comparable to standard already exists) and its immunity against most known attacks.

Index **Private Key Cryptosystems, Quasigroup, String Transformations, Random Permutations.**

I. Introduction

There is a need for simple cryptographic primitives to implement security in an environment with end users connected with terminals having limited storage and processing power.

Constructing ciphers using the algebraic structures of quasigroups based ciphers lead to particular simple yet efficient ciphers. Quasigroups are structures very similar to groups with the primary difference that they are in general not associative.

On other hand, constructing large quasigroups from smaller ones is an important problem for many applications [3-8]. In this paper, multiple quasigroups symmetric-key block cipher is proposed. This proposal uses special kind QGs constructed by complete mapping [8]. This helps to regenerate the QGs used by the sender at the receiving end, with minimal information exchange.

II. Preliminaries

In this section a brief overview of quasigroups, quasigroup operations and quasigroup string transformations is explained.

Definition 1. A quasigroup is a groupoid (Q,*) satisfying these laws:

$$(\forall\, u, v \in Q)\ (!\exists\, x, y \in Q)\colon u^* x = v,\ y * u = v \qquad (1)$$

$$x * y = x * z \Rightarrow y = z,\ y * x = z * x \Rightarrow y = z \qquad (2)$$

Manuscript received September 10, 2011, revised September 22, 2011. This paper was accepted by 4th IEEE International Conference on Computer Science and Information Technology (IEEE ICCSIT 2011)

H. Zorkta and T. Kabani are with Aleppo University- Syria. (email: drzorkta@hotmail.com; tarek_kabani@hotmail.com)

Let (Q, *) be a Quasigroup, then two operations \ and / on Q can be defined as:

$$x * y = z \Leftrightarrow y = x \setminus z \Leftrightarrow x = z / y \qquad (3)$$

Then the algebra (Q, *, \, /) satisfies the identities

$$\begin{gathered} x \setminus (x * y) = y,\ x * (x \setminus y) = y, \\ (x * y) / y = x,\ (x / y) * y = x \end{gathered} \qquad (4)$$

and (Q, \), (Q, /) are quasigroups too.

Definition 2. Let (Q, *, \, /) be a Quasigroup and $M = a_1, a_2, a_3, \ldots, a_n \in Q$. The encryption function E is defines as:

$$\begin{gathered} e_{l,*}(M) = b_1 b_2 \ldots b_n \Leftrightarrow \\ b_1 = l * a_1,\ b_2 = b_1 * a_2,\ \ldots,\ b_n = b_{n-1} * a_n \end{gathered} \qquad (5)$$

for each leader $l \in Q$ and for every string $M \in Q+$.

The decryption function D is then defines as:

$$\begin{gathered} d_{l,*}(M) = c_1 c_2 \ldots c_n \Leftrightarrow \\ c_1 = l * b_1,\ c_2 = b_1 * b_2,\ \ldots,\ c_n = b_{n-1} * b_n \end{gathered} \qquad (6)$$

Theorem 1. If (Q, *) is a finite quasigroup, then $e_{l,*}$ and $d_{l,\setminus *}$ are mutually inverse permutations of Q^+, i.e.,

$$d_{l,\setminus *}(e_{l,*}(M)) = M = e_{l,*}(d_{l,\setminus *}(M)) \qquad (7)$$

for each leader $l \in Q$ and for every string $M \in Q^+$.

Definition 3. Let (G, +) be a group.
Let $i\colon G \to G$ denote the identity map on G.
$\Theta\colon G \to G$ is a complete mapping if Θ is a bijection and $i - \Theta$ is a bijection where $(i - \Theta)(x) = x - \Theta(x)$.

Example 1. Let $(G, +) = (Z_9, +)$ where $Z_9 = (0, 1, 2, 3, 4, 5, 6, 7, 8)$ and + is performed modulo 9. Then $\Theta(x) = 5x + 4$ is a complete mapping because both Θ and $i - \Theta$ are bijections (see Table I)

TABLE I: A COMPLETE MAPPING ON Z_9

x	0	1	2	3	4	5	6	7	8
Θ(x)	3	8	4	0	5	1	6	2	7
i - Θ(x)	6	2	7	3	8	4	0	5	1

Sade [10] suggested creating a quasigroup $(Q, *)$ from an admissible group $(Q, +)$ and a complete mapping Θ by, defining

$$x * y = \Theta(x - y) + y, \text{ for } x, y \in Q \qquad (8)$$

An example shown in Table II

TABLE II: QUASIGROUP USING A THETA MAPPING

*	0	1	2	3	4	5	6	7	8
0	3	8	4	0	5	1	6	2	7
1	8	4	0	5	1	6	2	7	3
2	4	0	5	1	6	2	7	3	8
3	0	5	1	6	2	7	3	8	4
4	5	1	6	2	7	3	8	4	0
5	1	6	2	7	3	8	4	0	5
6	6	2	7	3	8	4	0	5	1
7	2	7	3	8	4	0	5	1	6
8	7	3	8	4	0	5	1	6	2

III. CIPHER ALGORITHM MAIN STRUCTURE

Multiple QGs cipher Algorithm is presented in this article. It consists of the following main components (see Fig.s1-2):

1- Chaotic Generator
2- CRP Generator
3- QG Generator

Before explaining these main parts, we need to introduce new representations for QGs and the Inverse QGs using theta mapping.

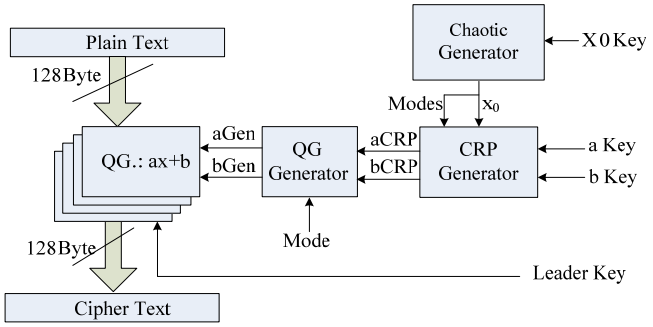


Fig. 1. A Proposal Encryption Algorithm.

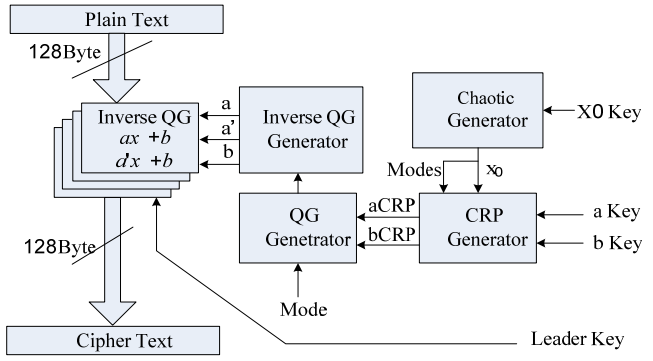


Fig. 2. A Proposal Decryption Algorithm.

A. QG & Inverse QG New Representations in Theta Mapping

A generation of QG using theta mapping as discussed previously, where:

$$\Theta(x) = ax + b. \tag{9}$$

This equation is used in encryption operations. So, it's preferred to develop an equation for the Inverse QG which used in decryption operations.

Thus a new representation of Inverse QG is developed as following:

$$f(x) = ax + b,\ f'(x) = a'x + b \tag{10}$$

To calculate variables a, b, & a' the following definitions is given:

Definition 4. If Theta mapping operation is:
$\Theta(x) = ax + b$. So the inverse Theta Mapping operation is:

$$\Theta^T(C_x) = y \Leftrightarrow \\ y(1 - a) + (ax + b) \equiv C \pmod N \tag{11}$$

Proof.
$\Theta(x - y) + y \equiv C \pmod N \Rightarrow$
$a(x - y) + b + y \equiv C \pmod N \Rightarrow$
$ax + b + y(1 - a) \equiv C \pmod N \Rightarrow$
$y(1 - a) + (ax + b) \equiv C \pmod N$.

Now, using (11) the Inverse QG variables, become:

$$b = \Theta^T(0_0) \tag{12}$$
$$a + b = \Theta^T(1_0) \tag{13}$$
$$a' + b = \Theta^T(0_1) \tag{14}$$

Example 2. Let $\Theta(x) = 4x + 2$ represent a QG $(Q,*)$, and N=5, the inverse QG then is $(Q,/)$, to represent this QG using theta mapping, form (10) we have:
$f(x) = ax + b, f'(x) = a'x + b$.
So, variables a, a', b can be calculated from equations (12-14).
$b = \Theta^T(0_0) \Leftrightarrow y(1 - 4) + (4\times0 + 2) \equiv 0 \pmod 5$
$\Rightarrow y = b = 4$.
$a + b = \Theta^T(1_0) \Leftrightarrow y(1 - 4) + (4\times0 + 2) \equiv 1 \pmod 5$
$\Rightarrow y = a + b = 2. \Rightarrow a = 3$.
$a' + b = \Theta^T(0_1) \Leftrightarrow y(1 - 4) + (4\times1 + 2) \equiv 0 \pmod 5$
$\Rightarrow y = a' + b = 2. \Rightarrow a' = 3$.
the Inverse QG is: $f(x) = 3x + 4,\ f'(x) = 3'x + 4$.

Definition 5. QG & Inverse QG Operations:
QG Operation:

$$x * y = \Theta(x - y) + y \bmod N \\ = a(x - y) + b + y \bmod N \tag{15}$$

Inverse QG. Operation:

$$x / y = ay + a'x + b \tag{16}$$

Example 3. Let (Q,*), (Q,/) be a QG, and its inverse, as in example 2. Let M=1102334, leader=3, to encrypt this message M using (5, 15).
$e_{3,*}(M) = c_1c_2 \ldots c_n \Leftrightarrow$
c1 = 3 * 1 = Θ(3 – 1) + 1 mod 5 = 1.
c2 = 1 * 1 = Θ(1 – 1) + 1 mod 5 = 3.
c3 = 4. … ⇒ C = $e_{3,*}$(M) = 1342123.

Now to decrypt this new message C using (6, 16).
$d_{3,/}(C) = a_1a_2 \ldots a_n \Leftrightarrow$
a1 = 3 / 1 = 3×1 + 3×3 + 4 mod 5 = 1.
a2 = 1 / 3 = 3×3 + 3×1 + 4 mod 5 = 1.
a3 = 0. … ⇒ C = $d_{3,/}$(C) = 1102334.= M.

Let $(Q,*)$ be a quasigroup represented by theta mapping as $\Theta(x) = ax + b$. and let $\Theta'(x) = a'x + b'$. as a CRP (it is not necessary to be complete mapping).

It is possible to conclude a new CRP using $\Theta(x)$ and $\Theta'(x)$,i.e. $\Theta''(x)=a''x + b''$ in two ways. First one using rows permutation on $\Theta(x)$ using $\Theta'(x)$ as following:

$$b'' = \Theta(\Theta'(0)) \quad (17)$$
$$a'' + b'' = \Theta(\Theta'^{T}(0) - 1) + 1 \bmod N \quad (18)$$

Where:

$$x = \Theta'^{T}(C) \Leftrightarrow ax' + b' \equiv C \pmod N \quad (19)$$

The second will use columns permutation as the following:

$$b'' = \Theta(0 - \Theta'^{T}(0)) + \Theta'^{T}(0) \quad (20)$$
$$a'' + b'' = \Theta(0 - \Theta'^{T}(1)) + \Theta'^{T}(1) \bmod N \quad (21)$$

Table III illustrates general definition of rows and columns permutation.

TABLE III: DEFINITION OF THE ROWS, COLUMNS PERMUTATION

rows, columns permutation [a", b"]=CRPPer(a, b, a', b')
Input: Integer a, b, where these variables achieve a complete mapping. Integer a′, b′ are variables achieve a theta mapping. ***Output*:** Integer a″, b″, which used to construct a new QG represented by a complete mapping
1. rows permutation: a) b" = Θ(Θ'(0)). b) a" + b" = Θ(Θ'T(0) – 1) + 1 mod N. 2. columns permutation: a) b" = Θ(0 – Θ' T(0)) + Θ' T(0). b) a" + b" = Θ(0 – Θ'T(1)) + Θ' T(1) mod N.

B. Chaotic Generator

Chaotic generator (CG) is used to generate random numbers. It is a keyed generator, based upon well known 1-D logistic map [10,11], and accepts its initial point as its secret key (see Fig. 3)

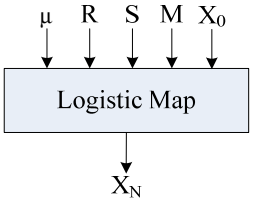


Fig. 3. Chaotic Generator

Table IV illustrated main steps of CG algorithm.

TABLE IV: CHAOTIC GENERATOR ALGORITHM.

Chaotic Generator
Input: Double X_0. Integer M, S, R, μ. Output: Integer XN.
Repeat R time 1) XN = LM(X0, R, μ). 2) Multiple output with S. 3) Adapt it according to value M.

C. Design CRP Generator using CG

Here a CRP generator is presented. This generator is achieved using RNG (Chaotic Generator), and the pair (*aKey*, *bKey*), these variables is a part of the algorithm key and must achieve a complete mapping), to generate the pair (*aGen*, *bGen* which achieve a theta mapping). That means this pair can produce random permutation in the range [*0..N–1*], so Θ is a bijection. But $\Theta - i$ is not necessarily to be a bijection. This generator will be used to produce pairs (*a, b*) randomly and use it as input to the QG generator.

Table V illustrates definition of CRP generator. While, CRP generator diagram is shown in Fig. 4.

TABLE V: DEFINITION OF THE CRP GENERATOR

CRP Generator
Input: Integer aKey, bKey, where these variables is a part of the algorithm ***key***. Double X0 where x0 is a random number generated by RNG. **Output:** Integer aGen, bGen, which used as an input to the QG generator to construct a new QG represented by a complete mapping.
1. Get x0 from CG and examine the.1st bit.
1st bit = { '0': go to step 2. '1': aGen = aKey, bGen = bKey ***(Mode 0).***
2. examine if x_0 suitable to represent a theta mapping, so if it is then go to step 4. 3. if x_0 not suitable then repeat step 2. 4. examine the.2nd bit of x_0
2nd bit = { '0': go to step.5 '1': aGen = x_0, bGen = rand(x_0). ***(Mode 1).***
5. examine the.3rd bit of x_0. *[aGen, bGen]=CRPPer(aKey, bKey, aPrv, bPrv).*
3rd bit = { '0': calculate aGen, and bGen using rows permutation. ***(Mode 2).*** '1': calculate aGen, and bGen using columns permutation ***(Mode 3).***

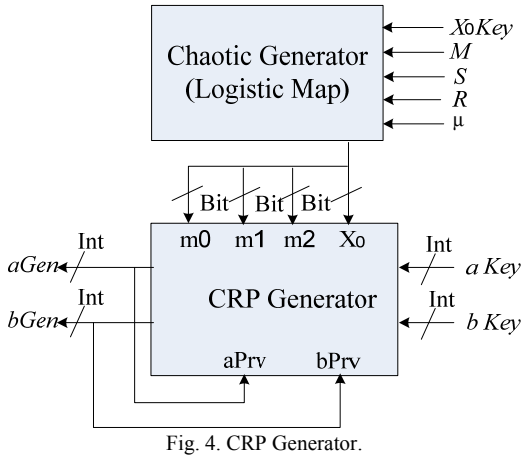


Fig. 4. CRP Generator.

D. Design QG Generator

To generate a huge number of quasigroups which represented by theta mapping, we will benefit from a CRP generator which used as a rows permutation on the previous generated quasigroup to construct a new one. The new constructed quasigroup either constructed from the Key *(aKey, bKey)* or from the last pair generated from this generator *(aGen, bGen).*Which determined by the selected mode *(m=[0, 1] respectively)*.

The generated *(aGen, bGen)* is calculated by using a rows permutation, see (20), (21), in this way the pair *(aGen, bGen)* achieve a complete mapping so it's appropriate to represent a quasigroup.

The generated pairs *(aGen, bGen)* is used to perform the encryption operations in this proposal.

Table VI illustrates definition of QG generator. While, QG generator diagram is shown in Fig. 5.

TABLE VI: DEFINITION OF THE QG GENERATOR

QG Generator
Input: Integer aKey, bKey: where these variables is neither a part of the algorithm key or the last generated pair from this generator, Integer aCRP, bCRP: from the CRP generator, Bit: m0, m1 to determine the generator modes. **Output:** Integer aGen, bGen, which represent a QG
1. according to m0, we determine the permutation mode *(row, column) (Mode 0).* 2. according to m1, we determine the pair *(a, b)* witch we will do the permutation on them *(Mode 1) (aKey, bKey)* or *(aPrv, bPrv).*
m= { '0': *[aGen, bGen] = QGGen(aKey, bKey, rows)* '1': *[aGen, bGen] = QGGen(aPrv, bPrv, rows)*

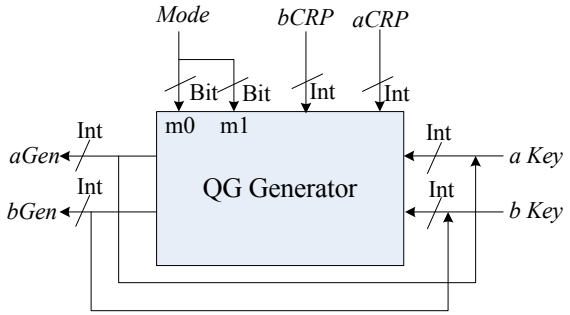


Fig. 5. QG Generator.

E. Design Inverse QG Generator

As known, inverse quasigroup is used in decryption phase, so Inverse QG generator is needed to calculate the trio *(aInv, a'Inv, bInv)* which based upon pairs *(aGen, bGen)* produced from QG generator.

Table VII. illustrates definition of Inverse QG generator. While, Inverse QG generator diagram is shown in Fig. 6.

TABLE VII: DEFINITION OF THE INVERSE QG GENERATOR

Inv QG Generator
Input: Integer aGen, bGen: where these variables is the output of the QG generator. Output: Integer aInv, a'Inv, bInv, which represent a Inverse QG
[aInv, a'Inv, bInv] =InvQGGen(aGen, bGen)

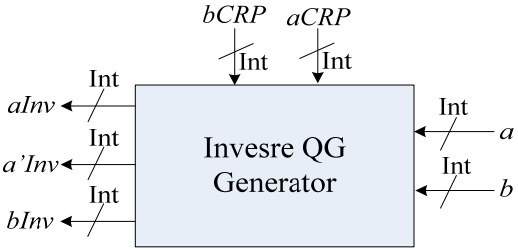


Fig. 6. Inverse QG Generator

Finally, The proposed algorithms in encryption and decryption phases are shown in Table VIII and IX respectively.

TABLE VIII: PROPOSED ENCRYPTION ALGORITHM.

proposed encryption algorithm
Input: Integer *aKey, bKey, leaderKey*. Double *x0Key* where these variables is the algorithm key. String: *Message* (plaintext) **Output:** String: *Encrypted Message* (ciphertext).
do until end of the message: Set a vector $vi = (m_i, \ldots, m_{i+127})$ from the message *[aCRP, bCRP]= CRPPer(aKey, bKey, aPrv, bPrv, x0).* *[aGen, bGen]=QGGen(aKey, bKey, aCRP, bCRP, m).* $el,*(v_i) = u_i = c_i, c_{i+1}, \ldots, c_{i+n-1}$.

TABLE IX: PROPOSED DECRYPTION ALGORITHM.

proposed decryption algorithm
Input: Integer *aKey, bKey, leaderKey*. Double *x0Key* where these variables is the algorithm key. String: *Encrypted Message* (ciphertext). **Output:** String: *Message* (plaintext).
do until end of the encrypted message: Set a vector $u_i = (c_i, \ldots, c_{i+127})$ from the message *[aCRP, bCRP]= CRPPer(aKey, bKey, aPrv, bPrv, x0).* *[aGen, bGen]=QGGen(aKey, bKey, aCRP, bCRP, m).* *[aInv, a'Inv, bInv] =InvQGGen(aKey, bKey)* $d_{i,*}(v_i) = v_i = m_i, m_{i+1}, \ldots, m_{i+n-1}$.

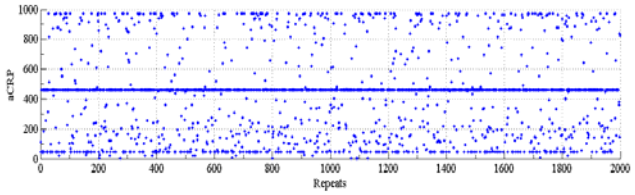
IV. ANALYSIS STUDY

Random behavior of CRP, QG generators is evaluated in this paper, with measuring the strength of new proposal. Speed comparative study between cipher algorithm in encryption and decryption, was done with the current block cipher standard (Rijndael). All these studies are done on 2.7 GHz CPU with 2GBytes RAM machine.

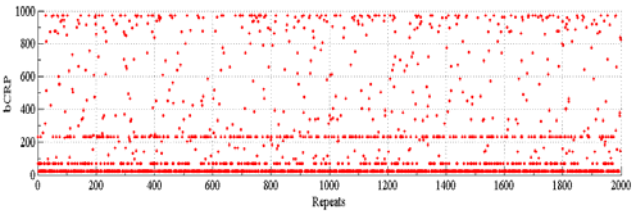
A. CRP Generator analysis study

Random behavior of CRP generator under different modes was covered in this analysis.

- *Random behavior of CRP Generator*:
  Most randomly results of CRP generator, is achieved by using Chaotic Generator (CG) as an input.
  Fig. 7 explains the random behavior of the generated pairs (aCRP, bCRP), where, 2000 repetitions were used and output values shows randomness.



7-a aCRP values.



7-b bCRP values.

Fig. 7. CRP Generator.

- *Repetitive modes of CRP Generator:*
  CRP generator works in 4 modes, which consolidate random output pairs rises in this generator. These modes are selected automatically according to the generated number from CG.
  Fig. 8 displays the repetitive selected modes for 2000 CRP generator calls.

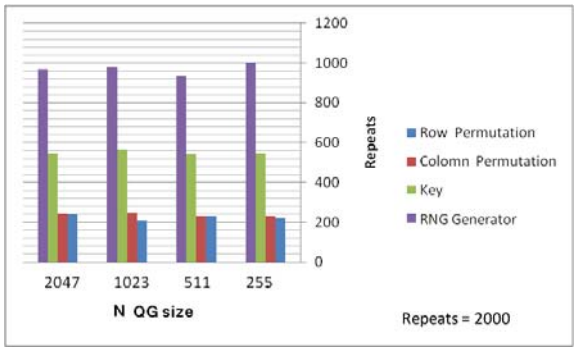


Fig. 8. CRP Generator modes repetitions.

B. QG Generator Analysis Study

QG generator goal was, to get different QGs each time. It based on CRP generator random behavior in generating these different QGs. Fig. 9 illustrates experiment on 2000 QG generator calls, and shows the chaotic results for the generated QGs.

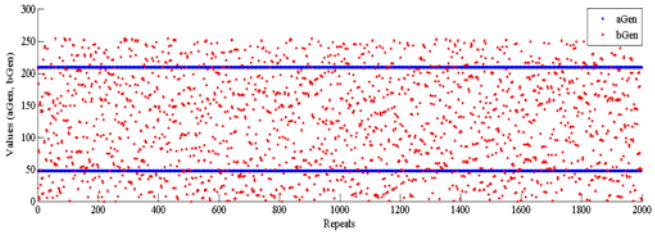


Fig. 9. QG generator results.

C. Proposed Cipher Algorithm's strength

Secrecy of the proposed cipher algorithm is depend on group of unequal initial points (x0Key, aKey, bKey, leaderKey, N). According to the data type used, there are 2^{192} possible combinations for its initial points. This is equal to key length 192 bits. If we assume that there are computers works with computation power from order of 10^{20} operation per second, it will need Tbreak$\approx 3.1 \times 10^{50}$ year to predict the secret key. Thus, Exhaustive attack seems impractical.

Number of iteration ***R*** in logistic map is related directly to x0Key and domain M, and leaving R to be chosen randomly from this domain will make our proposal more secure, e.g. M=1000 will add 10 bits to length of the key.

Using different leaderKey for each generated QG, will add K × 2N-1 bits to key length, where K is the number of generated QGs of order N.

D. Proposed Cipher Algorithm's speed

Different File sizes [3.5 KB - 1MB] was encrypted, decrypted with the proposed cipher algorithm.

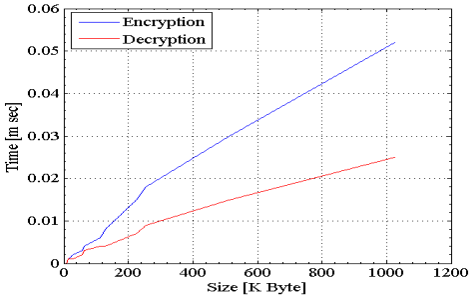


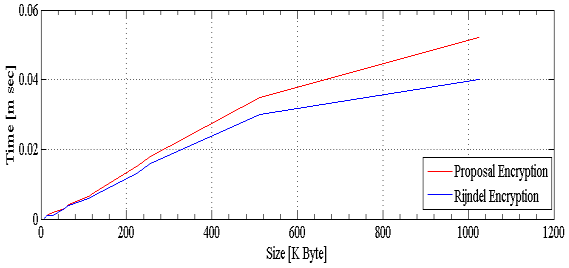
Fig. 10. Encryption, Decryption Times vs. file sizes.

Fig. 10 shows that time is increased proportionally according to the file size, and decryption time is less than encryption time specially on large file sizes, that is because of equations (15,16) which used in these operations.

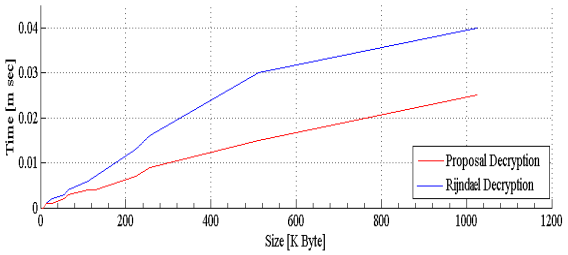
E. Comparative Study

Comparative study between new proposal and the standard Rijndael was done under the same parameters and the results was drawn on Fig(11).

This study shows that new proposal is comparable to Rijndael in encryption times, specially with file sizes (smaller than 500 KB), but it exceeds Rijndael in decryption times. Thus, we recommend to use the new representations formulas in digital signature schemas.



11-a Proposal vs. Rijndael in Encryption phase.



11-b Proposal vs. Rijndael in Decryption phase.

Fig. 11. Proposal vs. Rijndael.

V. Conclusions

New cipher algorithm based on multiple quasigroups is presented in this article. The main idea is to represent quasigroups by special kind of mapping called a complete mapping controlled by equations in all transformations of these QGs.

New representation of the inverse QG formed by complete mapping is introduced in this paper. These formulas make all the cipher algorithm transformations is done in high speed.

The new Proposal has the following features:

- Key length (192 bits at least) which is comparable to the strength of standards already in use.
- Encryption times comparable to Rijndael standard's encryption times, specially with file sizes (smaller than 0.5 MB).
- Decryption times are better than Rijndael standard's decryption times. Thus, we recommend to use the new representations formulas in digital signature schemas.

References

[1] S. Markovski, "Quasigroup string processing and applications in cryptography", in Proc. 1-st Inter. Conf.Mathematics and Informatics for industry – MII, Thessaloniki 2003, pp. 278–290, 2003.

[2] Markovski S., Gligoroski D., Bakeva V.: "Quasigroup String Processing – Part 1", Contributions, Sec. math. Tech. Sci., MANU, XX, 1-2(1999) 13-28.

[3] C.Z. Koscielny, "Generating Quasigroups for Cryptographic Applications", International Journal of Applied Mathematics & Computer Science, Vol. 12, No. 4, pp. 559-569, 2002.
[4] S.K. Pal, S. Kapoor, A. Arora, R. Chaudhary, J. Khurana, "Design of Strong Cryptography Schemes based on Latin Squares", Proceedings of the Pre-ICM International Convention on Mathematical Sciences, New Delhi, 2008.
[5] S.M. Hussain, N.M. Ajlouni, "Key Based Random Permutation", Journal of Computer Science, Vol. 2, No. 5, pp. 419-421, 2006
[6] A. Klimov, A. Shamir, "A New Class of Invertible Mappings", CHES, LNCS-2523, 2002.
[7] A. Klimov, A. Shamir, "Cryptographic Applications of T-functions", Selected Areas in Cryptography, SAC-2003, LNCS-3006, Springer Verlag, pp. 248-261, 2003
[8] K.A. Meyer, "A New Message Authentication Code Based on the Non-associativity of quasigroups", Doctoral Dissertation, Iowa State University Ames, Iowa, 2006, [Online] Available. http://orion.math.iastate.edu/dept/thesisarchive/PHD/KMeyerPhDSp06.pdf [Accessed: Feb. 24, 2008].
[9] A. Sade, "Quasigroupes Automorphes par le Groupe Cyclique", Canadian Journal of Mathematics, 9, pp. 321-335, 1957
[10] http://www.egwald.ca/nonlineardynamics/logisticsmapchaos.php
[11] Nagata, K. Wayne. Nonlinear Dynamics and Chaos: Mathematics 345 Lecture Notes. Vancouver: University of B.C., 2006.

Assoc. Prof. Dr. Haythem Zorkta
1970, Damascus – Syria (drzorkta@hotmail.com)
A lecturer at Aleppo University- Syria.
Master and PhD degree at computer networks security from MTC- Cairo- Egypt.
He has published a lot of international and local papers at the same field, and as a technical reviewer for many international and local conferences, supervisor for many Master and PhD thesis's.

Eng. Tarek Kabani
1982, Swaida-Syria (tarek_kabani@hotmail.com)
Master degree from Aleppo University (2011) at network security.