# A NEW GAUSSIAN CIPHER WITH OPTIMAL KEYED PROCESS (KAM-FA)

H. Elkamchouchi[1], Fatma Ahmed[2]
[1, 2] Electrical Engineering Department, Faculty of Engineering, Alexandria University
[1] Helkamchouchi@ieee.org
[2] moonyally@yahoo.com

## ABSTRACT

Nowadays, cryptography plays a major role in protecting the information of technology applications. This paper gives a new symmetric cryptosystem having a key dependent operation, enhanced by a rotor with controlled user identification ID and user key. The plaintext block is divided into basic Gaussian sub-blocks each of thirty-two bits in length. The new Proposal uses optimal MDS matrix. The new Proposal can encrypt blocks of plaintext of length 512 bits into blocks of the same length. Also the key length is 512 bits. The total number of rounds is sixteen rounds. It uses $2^{16}$ modulo addition and thirty-two bits XORING followed by modulo $2^{16}+1$ multiplication. The secret key is encrypted using the optimal MDS matrix to avoid any weakness points in the user key. We also try to get the minimum correlation between plaintext and ciphertext, highly avalanche effect and defeat the frequency analysis and most well-known attacks. The proposed algorithm is compared with the well known AES and IDEA symmetric systems and it gives excellent results from the point of view of the security characteristics and the statistics of the ciphertext. Also, we apply the randomness tests to the proposed algorithm and the results shown that the new design passes all tests which proven its security.

## KEYWORDS

Gaussian field, MDS matrix, Branch number, Rotor bank, User ID, frequency analysis.

## 1 INTRODUCTION

### 1.1 Gaussian Integers

Gaussian integers are complex numbers on the form $x+iy$ where $x$ and $y$ are integers and $\mathrm{i}=\sqrt{-1}$.

The norm N of a Gaussian integer $x+iy$ is $x^2+y^2$. A Gaussian prime is a Gaussian integer that cannot be expressed in the form of a product of other Gaussian integers. The ring of Gaussian integers is a unique factorization domain.

### 1.2 Rotor Cryptosystem

Cryptology [1] has for much of its history involved mathematical calculations that must be performed at great speeds. Human minds are not suited to performing such calculations quickly and accurately, so we have developed machines to aid the process of encryption, decryption, and cryptanalysis. Before the widespread use of digital computers, computing was done using machines that combine electrical and mechanical components. During this time, which took place primarily between the 1930s and the 1960s, rotors and rotor-based cryptosystems were developed and used extensively.

### 1.3 The MDS Matrix

Maximum distance separable matrixes (MDS) are widely used in design of block ciphers and hash functions etc. Based on the character of its differential branch number, MDS matrix is widely used and the arithmetic using MDS matrixes can effective against differential cryptanalysis and linear cryptanalysis. A linear code over Galois field $GF(2^p)$ is denoted as an $(n,k,d)$ code, where $n$ is the symbol length of the encoded message, $k$ is the symbol length of the original

message, and $d$ is the minimal symbol distance between any two encoded messages[2].

**Definition 1:** Let $K$ be a finite field and $p$ and $q$ be two integers. Let $x \mapsto M \times x$ be a mapping from $K^p$ to $K^q$ defined by the $q \times p$ matrix $M$. We say that it is a linear multipermutation (or an MDS matrix) if the set of all pairs $(x, M \times x)$ is an MDS code, i.e. a linear code of dimension $p$, length $p+q$ and minimal distance $q+1$[3].

The following theorem [4] will depict the character of MDS matrix from the angle of a subdeterminant.

**Theorem 1:** A matrix is an MDS matrix if and only if every sub-matrix is non-singular.

MDS matrices are constructed by two types of matrices: circulant and Hadamard matrices.

***Circulant matrices:*** Given $k$ elements $\alpha_0, \alpha_1.., \alpha_{k-1}$, a circulant matrix $M$ is constructed with each entry $M_{i,j} = \alpha_{(i+j) \bmod k}$.

***Hadamard matrices:*** Given $k$ elements $\alpha_0, \alpha_1.., \alpha_{k-1}$, a Hadamard matrix $M$ is constructed with each entry $M_{i,j} = \alpha_{(i \oplus j)}$.

**Definition 2:** Let $K^*$ be a set including a distinguished one denoted 1. Let $M$ be a $q \times p$ matrix whose entries lie in $K^*$.

1. We let $v_1(M)$ denote the number of $(i, j)$ pairs such that $M_{i,j}$ is equal to 1. We call it the number of occurrences of 1.

2. We let $c(M)$ be the cardinality of $\{M_{i,j}; i = 1,....,q; j = 1,.....p\}$. We call it the number of entries.

The following lemmas provide optimal constructions for small $p$ and $q$.

**Lemma 1:** We have $v_1^{q,p} \geq p + 2q - 3$ for any $p, q$ such that $q \leq p$.

**Lemma 2:** For any $m$ we have $c^{2m-1,2m-1} \leq m$.

### 1.4 Branch Numbers of Matrices and Distance of Linear Codes

The branch number of a permutation function is representing the diffusion rate and measures security against differential and linear cryptanalysis [5]. The branch number is defined as the minimum number of nonzero elements in the input and output when the input elements are not all zero. The branch number of an $n \times n$ matrix $M$ is defined by:

$$\beta(M) = \min\left\{wt(x) + wt(M.x^T) \mid x \in \left(\{0,1\}^m\right)^n, x \neq 0\right\} \quad (1)$$

Where: $wt$ : Hamming weight.

$$x = (x_1, x_2, ..., x_n)^T, x_i \in \{0,1\}^m, i = 1, ..., n$$

**Theorem 2:** For MDS$(2n, n, d)$ code over $GF(2^8)$, then the branch number of $M^T$ is $d$.

We see that the maximum branch number of $n \times n$ binary matrices is equal to the maximum distance of binary linear $[n, 2n]$ codes. It is an important topic in the coding theory to find the maximum distance of binary linear $[n, 2n]$ codes.

## 2 THE NEW PROPOSED SYSTEM

The new system is a block cipher; it can encrypt blocks of plaintext of length 512 bits into blocks of the same length. The key length is 1024 bits. We test the new algorithm for many numbers of round, we found that the efficient number of round which gives better avalanche effect is 16. In the new system we: introduced representation for data using the Gaussian field, proposed new optimal MDS matrix, introduced new rotor bank depend on subkey of the round and the user identification to resist the frequency analysis attack and we generate the round subkey using the new MDS matrix and the rotor bank.

### 2.1 The encryption process

Our proposal system is purely block cipher. The input plaintext length is 64 bytes. These bytes are divided into 16 sub-blocks each of 32 bits. Each sub-block is represented in Gaussian field.

$$Sub\_Block = a + ib \quad (2)$$

Where: $a$ and $b$ : are short word (16 bits).

We offer the cryptosystem to the domain of Gaussian field to make the cryptosystem more secure and very difficult to be broken. The output of this system is also 16 sub-blocks arranged sequentially each of 32 bits divided into real and imaginary parts. These sub-blocks are combined again to form 64 bytes blocks. The key length is 1024 bits.

Many cryptanalytical techniques treat the top and bottom rounds of the cipher differently than the middle rounds. Typically, these techniques begin by guessing several key bits, hence "stripping out" some of the top/bottom rounds of the cipher, and then mounting the cryptanalytical attack against the remaining rounds. This suggests that the top and bottom rounds of the cipher play a different role than the middle rounds in protecting against cryptanalytical attacks [4]. Therefore, in our design the middle rounds are designed differently than the top and bottom rounds, which are viewed as "envelope rounds". Specifically, envelope rounds consist of first XORing key words with the input plaintext, and then performing several rounds of keyed transformation. Finally we XOR the output from middle rounds with key words.

### 2.2 Description of a Single Middle Round

Input block is sixty-four characters divided into sixteen sub–blocks each consisting of 32 bits. Input key is 128 characters used to generate 128 sub–blocks each consisting of 32 bits words bits. The top and bottom round consists of 16 sub-key words and the middle round consists of $6\times16$ sub-key words. Figure 1 shows one round of our system.

Considering [+] is modulo $2^{16}$ addition, [.] is modulo $2^{16}+1$ multiplication, $\oplus$ is XOR 32 bits. $<<<$ is left shift. $(a_1+ib_1)<<<(a_2+ib_2)$ is left shift $a_1$ by amount calculated from taking last four bits of $a_2$ and left shift $b_1$ by amount calculated from taking last four bits of $b_2$.

**Step 1:** $x_6 = x_6\ [+]\ x_3, \qquad x_3 = x_3 <<< x_1,$

$$x_1 = x_1 \oplus sk_{(17+6(r-1))} \tag{3}$$

**Step 2:** $x_2 = x_2 \oplus x_9, \qquad x_9 = x_9 <<< x_7,$

$$x_7 = x_7\ [\bullet]\ sk_{(19+6(r-1))} \tag{4}$$

**Step 3:**

$$x_{12} = x_{12}\ [+]\ x_4, \qquad x_4 = x_4 \oplus sk_{(18+6(r-1))} \tag{5}$$

**Step 4:** $x_8 = x_8\ [+]\ x_{11}, \qquad x_{11} = x_{11} <<< x_{10},$

$$x_{10} = x_{10} \oplus sk_{(20+6(r-1))} \tag{6}$$

**Step 5:** $x_{14} = x_{14} \oplus x_{15}, \qquad x_{15} = x_{15} <<< x_{13},$

$$x_{13} = x_{13}\ [\bullet]\ sk_{(21+6(r-1))} \tag{7}$$

**Step 6:**

$$x_5 = x_5 \oplus x_{16}, \qquad x_{16} = x_{16}\ [\bullet]\ sk_{(22+6(r-1))} \tag{8}$$

### 2.3 The New Efficient MDS Matrix

In the new algorithm, we design new MDS matrixes which provide the maximum branch number and the optimal construction conditions. The new matrixes are self inverse so that same matrix can be used for decryption algorithm, which decreases the complexity of system. The new MDS matrix is $8\times8$ Hadamard matrix. MDS is $(16,8,9)$. MDS property of the matrix is calculated i.e. a $(16,8,9)$ code is MDS if d = n–k +1.This can be done by checking the branch number of the transformation. The input with one or two active byte column is multiplied with the matrix and the output column is checked, if the total number of active bytes including input and output bytes is equal to 9 then it satisfies the property of MDS. The new MDS matrix is checked for the involution property. We design it by providing the involution conditions which can calculate from the next matrix:

$$\begin{pmatrix} b_0 & b_1 & b_2 & b_3 & b_4 & b_5 & b_6 & b_7 \\ b_1 & b_0 & b_3 & b_2 & b_5 & b_4 & b_7 & b_6 \\ b_2 & b_3 & b_0 & b_1 & b_6 & b_7 & b_4 & b_5 \\ b_3 & b_2 & b_1 & b_0 & b_7 & b_6 & b_5 & b_4 \\ b_4 & b_5 & b_6 & b_7 & b_0 & b_1 & b_2 & b_3 \\ b_5 & b_4 & b_7 & b_6 & b_1 & b_0 & b_3 & b_2 \\ b_6 & b_7 & b_4 & b_5 & b_2 & b_3 & b_0 & b_1 \\ b_7 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 \end{pmatrix} \tag{9}$$

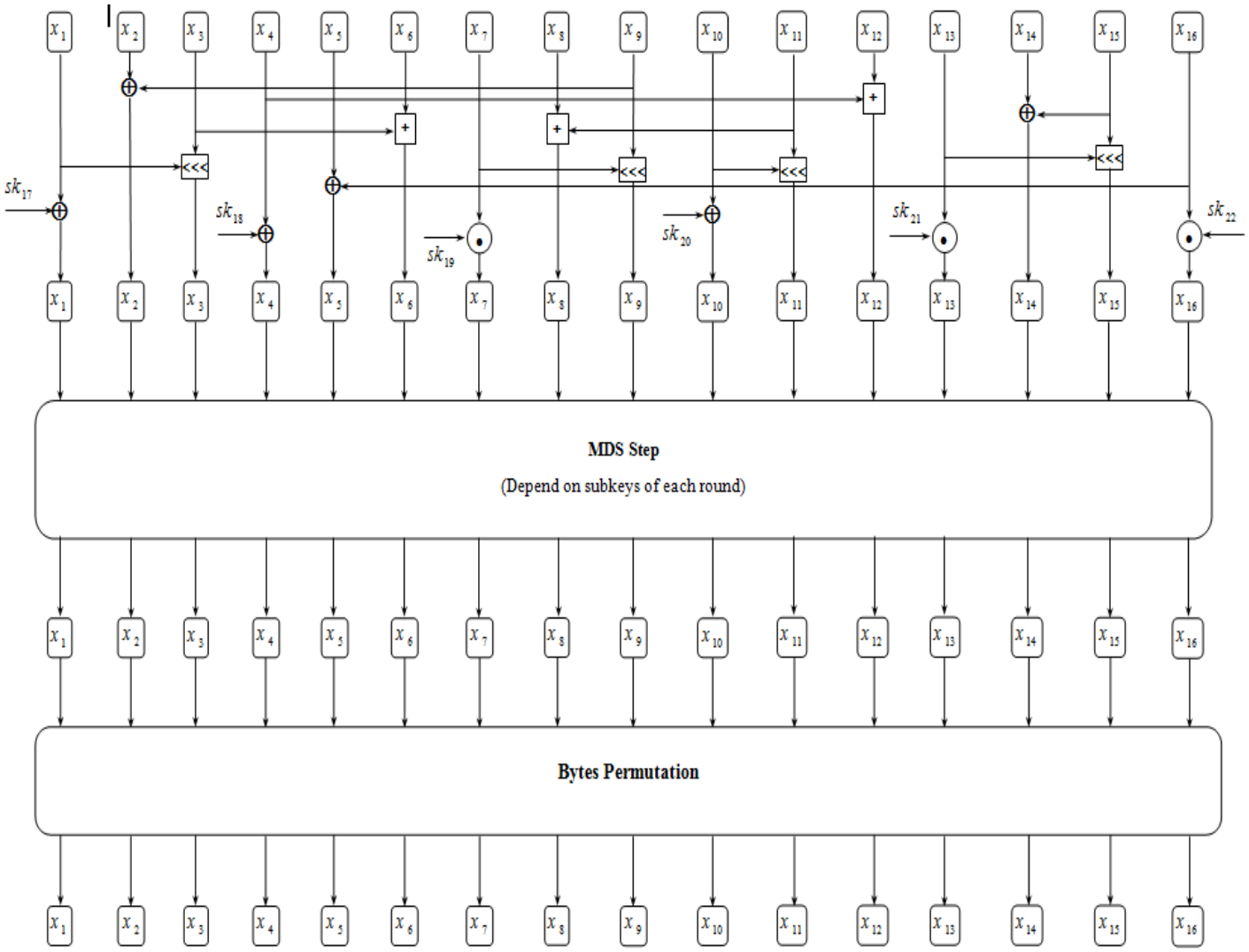


**Figure.1**. The structure of one round

These conditions relative to the above matrix are:

1- For involution:

$$b_0^2 + b_1^2 + b_2^2 + b_3^2 + b_4^2 + b_5^2 + b_6^2 + b_7^2 = 1 \quad (10)$$

2- For singular:

$$b_0 \neq b_1 \qquad b_2 \neq b_3 \qquad b_4 \neq b_5 \qquad b_6 \neq b_7$$
$$b_0 \neq b_3 \qquad b_4 \neq b_7 \qquad b_0 \neq b_7 \quad (11)$$

In order to have optimal construction for the new MDS matrix, we apply the lemma 1 and lemma 2 so we get:

$$v_1^{8,8} \geq 16 + 8 - 3 = 21 \ , \ c^{8,8} \leq 5 \quad (12)$$

To satisfy the equations 10, 11 and 12 in $GF(2^8)$ for polynomial $x^8 + x^4 + x^3 + x + 1$we we choose the elements for the new MDS matrix with the following conditions:

$$b_1 = b_3 = b_5 = 1 \ , \ b_0 = b_4 \ , \ b_2^2 + b_6^7 + b_7^2 = 0 \quad (13)$$

The obtained polynomials for first MDS matrix and second MDS matrix are:

$$H_1 = had[xC1, x01, x40, x01, xC1, x01, xD1, x91]$$
$$H_2 = had[x89, x01, xC9, x01, x89, x01, x08, xC1] \quad (14)$$

By using these polynomial $8 \times 8$ matrixes are constructed. The equations 15 and 16 are

represented the first MDS matrix and the second MDS matrix respectively.

$$\begin{pmatrix} C1 & 01 & 40 & 01 & C1 & 01 & D1 & 91 \\ 01 & C1 & 01 & 40 & 01 & C1 & 91 & D1 \\ 40 & 01 & C1 & 01 & D1 & 91 & C1 & 01 \\ 01 & 40 & 01 & C1 & 91 & D1 & 01 & C1 \\ C1 & 01 & D1 & 91 & C1 & 01 & 40 & 01 \\ 01 & C1 & 91 & D1 & 01 & C1 & 01 & 40 \\ D1 & 91 & C1 & 01 & 40 & 01 & C1 & 01 \\ 91 & D1 & 01 & C1 & 01 & 40 & 01 & C1 \end{pmatrix} \quad (15)$$

$$\begin{pmatrix} 89 & 01 & C9 & 01 & 89 & 01 & 08 & C1 \\ 01 & 89 & 01 & C9 & 01 & 89 & C1 & 08 \\ C9 & 01 & 89 & 01 & 08 & C1 & 89 & 01 \\ 01 & C9 & 01 & 89 & C1 & 08 & 01 & 89 \\ 89 & 01 & 08 & C1 & 89 & 01 & C9 & 01 \\ 01 & 89 & C1 & 08 & 01 & 89 & 01 & C9 \\ 08 & C1 & 89 & 01 & C9 & 01 & 89 & 01 \\ C1 & 08 & 01 & 89 & 01 & C9 & 01 & 89 \end{pmatrix} \quad (16)$$

In our new system, the data at MDS step is converted into $8\times8$ matrix and multiplied with the MDS matrix. Each element in the product matrix is the sum of products of elements of one row and one column. We use in every round one MDS matrix from two depending on the user keys of it. At first we take the first bit in all subkeys and XOR them together. If the resulting bit is one then we use the first MDS matrix else we use the second one. In the MDS step we arrange the data according to the following equation:

$$\begin{bmatrix} subblok:1 & subblok:3 \\ subblok:2 & subblok:4 \\ subblok:6 & subblok:5 \\ subblok:7 & subblok:8 \\ subblok:11 & subblok:9 \\ subblok:12 & subblok:10 \\ subblok:15 & subblok:13 \\ subblok:16 & subblok:14 \end{bmatrix} \quad (17)$$

This arrangement makes every sub-block effect on all other because sub-blocks: 9, 3, 10, 4, and 13 effect on sub-blocks: 2, 6, 11, 12 and 15 and three of round subkeys effect on sub-blocks: 1, 7 and 16. Also, sub-blocks: 1, 16, 11, 7 and 15 effect on sub-blocks: 3, 5, 8, 9 and 14 and three of round subkeys effect on sub-blocks: 4, 10 and 13.

### 2.4 Bytes Permutation

In order to have maximum avalanche effect, we need to permute the bytes to insure that all subkeys and inputs bytes will effect in all ciphertext. The bytes permutation depends on permute the bytes of sub-blocks: 1, 4, 7, 10, 13 and 16 in all other words because these sub-blocks have the subkeys effect. The bytes permutation is shown in table 1.

**Table 1** The Bytes Permutation

| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** |
|---|---|---|---|---|---|---|---|
| **1** | 2 | 3 | 4 | 22 | 56 | 24 | 40 |
| **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** |
| **9** | 10 | 11 | 12 | 64 | 17 | 54 | 61 |
| **17** | **18** | **19** | **20** | **21** | **22** | **23** | **24** |
| **14** | 60 | 26 | 51 | 27 | 5 | 29 | 7 |
| **25** | **26** | **27** | **28** | **29** | **30** | **31** | **32** |
| **44** | 19 | 21 | 35 | 23 | 42 | 49 | 38 |
| **33** | **34** | **35** | **36** | **37** | **38** | **39** | **40** |
| **46** | 43 | 28 | 53 | 41 | 32 | 47 | 8 |
| **41** | **42** | **43** | **44** | **45** | **46** | **47** | **48** |
| **37** | 30 | 34 | 25 | 50 | 33 | 39 | 58 |
| **49** | **50** | **51** | **52** | **53** | **54** | **55** | **56** |
| **31** | 45 | 20 | 63 | 36 | 15 | 57 | 6 |
| **57** | **58** | **59** | **60** | **61** | **62** | **63** | **64** |
| **55** | 48 | 62 | 18 | 16 | 59 | 52 | 13 |

### 2.5 The Rotor Bank

The KAM-FA cryptosystem is a block cipher enhanced by using a rotor mechanism. The rotor mechanism can be considered as a stand-alone cryptosystem. The rotor in the software implementation expands the character space to include all available 256 ASCII characters. This means that each cylinder will contain 256

characters in different permutation. This advantage is appeared in the extremely long period of the KAM-FA cryptosystem $(256)^{\text{number of cylinders}}$. In this paper we introduce new rotor bank with four cylinders controlled by the user ID and key and implemented using the inverse exponent function. The rotor bank construction attempt in the following way:

1- ***Initialize the rotor bank:*** The first cylinder contains 0x00, 0x01,……, 0xFF. The second cylinder contains 0x00, 0x01,…0xFF etc, and so on until the fourth cylinder.

2- ***Rotate the rotor bank with the user key:*** at the first we partition the user key into four sub blocks each with 256 bit. For every sub block we count the number of 1's bit on it. The represented four output numbers for sub blocks are used to rotate the four cylinders of the rotor bank respectively.

3- ***Map*** each byte in the rotor bank to its multiplicative inverse in the finite field $GF(2^8)$ ; the value {00} is mapped to itself.

After first eight rounds of KAM-FA rotor mechanism is used. After every character output we rotate the cylinder by using the round subkeys and the user ID. The user ID is 128 bit length which contains all information about the user. First, we divide the user ID into sub-blocks with length two bits then we divide the subkeys into sub-blocks with length two bits. After we have the first character output we XOR the first sub-block of the user ID with the first sub-block of subkeys. The resulting output is represented the number of cylinder that will be rotated and so on until the last character in plaintext. The figure 2 has shown the overall structure of KAM-FA.

### 2.6 The Subkeys Generation

The new system key expansion algorithm takes as input a 128-byte key and produces 128 words (512 bytes). This is sufficient to provide 6-word subkey for each of the 16 rounds of the cipher and 32-word to the envelope rounds. In subkey generation process, we try to have maximum avalanche effect between the user key and the ciphertext and to have minimum correlation coefficient. We use MDS matrix to make the subkeys effective against differential cryptanalysis and linear cryptanalysis. The worst case for the user key is to be repeated zeros in this case also the subkeys will be all zeros so we encrypt the first element in every column in key matrix by using the rotor bank. We divide the user key into two matrixes and use the MDS matrix to produce the first 128 bytes in the subkeys. Then we take the output matrixes and apply the second MDS matrix. We repeat this operation until we have 512 bytes of subkeys.

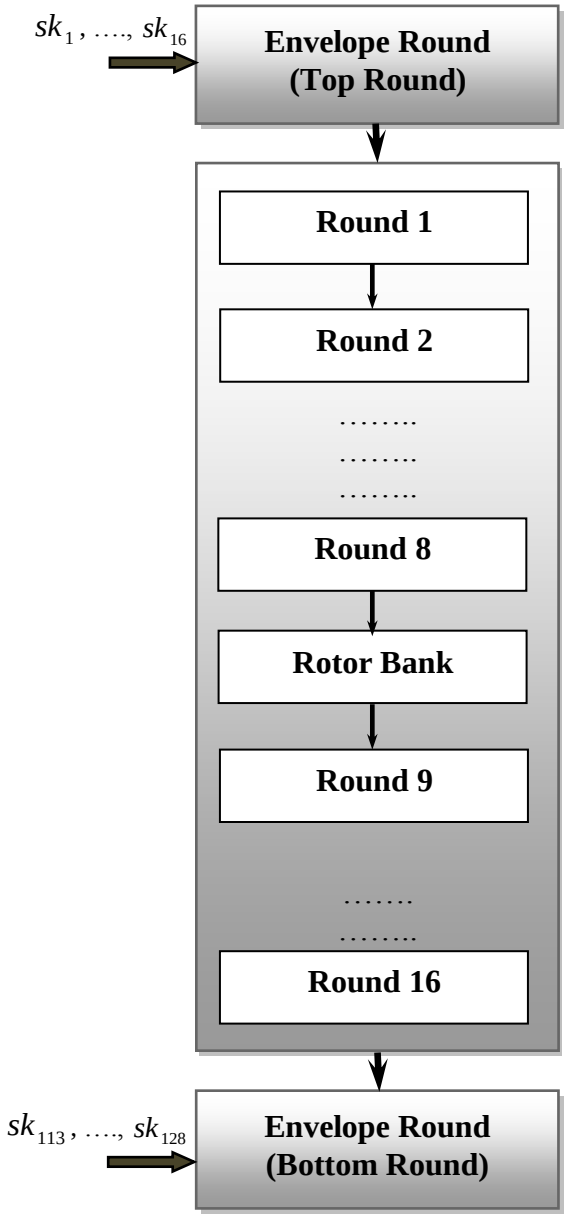


**Figure.2** KAM-FA overall structure

In order to have maximum avalanche effect we need to make sure that every bit in the user key will effect on all subkeys. This is done by XORing the first byte in first column in key matrix after applying the MDS matrix on column with the second byte in second column before use the MDS matrix on the second column. We repeat this process until the eighth column. Also before we use the output matrix from multiplying the first 64 bytes of user key with MDS matrix we XOR the first elements in every column with the first elements in the second output matrix which produces from multiplying the last 64 bytes of user key with MDS matrix and so on until we have the 128 words of the subkeys.

## 3. SECURITY ANALYSIS

### 3.1 Avalanche Effect

In cryptography, the **avalanche effect** refers to a desirable property of cryptographic algorithms. The avalanche effect is evident when an input is changed slightly (for example, flipping a single bit) the output changes significantly (e.g. half the output bits flip). In the case of quality block ciphers, such a small change in either the key or the plaintext should cause a drastic change in the ciphertext. Constructing a cipher to exhibit a substantial avalanche effect is one of the primary design objectives. The avalanche effect is calculated as:

$$\text{Avalanche Effect}=\frac{\text{No. of flipped in the ciphered text}}{\text{No. of bits in the ciphered text}}\times 100\% \quad (18)$$

In our case, we take four plaintexts; first one is normal message with normal key. The second one is repeated zeros binary with repeated zeros key. The third one is repeated ones binary with repeated ones key. The fourth ones is repeated ones binary with repeated zeros key. Each one of them is only 512 in length, flipping one bit from everyone in different positions and calculate the avalanche effect. Then we flip the user key in different positions and calculate the avalanche effect [5]. The following results are obtained after calculating the respective Avalanche Effects for our system, AES and IDEA.

**Table.2** Avalanche effect for 1 bit change in the plaintext

| Length of text in bits | Change first bit in plaintext | | | Change last bit in plaintext | | | Change middle bit in plaintext | | |
|---|---|---|---|---|---|---|---|---|---|
| | KAM-FA | AES | IDEA | KAM-FA | AES | IDEA | KAM-FA | AES | IDEA |
| 512 | 52% | 12.5% | 5.7% | 53.5% | 11.7% | 6.4% | 52.2% | 11.1% | 5.9% |
| 512 | 55.1% | 11.9% | 0% | 52.4% | 12.7% | 0% | 54.7% | 11.5% | 0% |
| 512 | 52.2% | 13.5% | 6.8% | 54.3% | 13.9% | 6.3% | 53.5% | 10.9% | 6.4% |
| 512 | 54.1% | 12.9% | 0.1% | 53.3% | 11.7% | 0.1% | 55.4% | 11.5% | 0.1% |

**Table.3** Avalanche effect for 1 bit change in the user key

| Length of text in bits | Change first bit in key | | | Change last bit in key | | | Change middle bit in key | | |
|---|---|---|---|---|---|---|---|---|---|
| | KAM-FA | AES | IDEA | KAM-FA | AES | IDEA | KAM-FA | AES | IDEA |
| 512 | 53% | 48.4% | 51% | 53.7% | 50% | 50% | 54% | 50.8% | 49.6% |
| 512 | 53.7% | 41.4% | 5.6% | 54.3% | 50% | 3.1% | 53.7% | 48.4% | 3.1% |
| 512 | 52.5% | 51.6% | 49.2% | 53.7% | 46.1% | 47.7% | 52.4% | 48.4% | 36% |
| 512 | 53.1% | 48.4% | 20.3% | 52.2% | 45.3% | 29.7% | 53.3% | 46.1% | 23.4% |

The avalanche effect of the proposed algorithm is producing very high as comparison with AES and IDEA because in AES and IDEA the data length is 128 bits and 64 bits respectively, while in our proposed system the data block length is 512 bits.

### 3.2 Secret Data Groups

Considering the secret data used in AES, the brute force attack for the key in the case of 128 bit block and plaintext is $(2^{128}=3.4\times10^{38})$. The secret data used in IDEA, the brute force attack for the key in the case of 128 bit block is $(2^{128}=3.4\times10^{38})$. The brute force attack for the data block in the case of 64 bit block is $(2^{64}=1.8\times10^{19})$.Considering the secret data used in our proposed system, the brute force attack for the key for 1024 bits block is $2^{1024}=1.8\times10^{308}$. The brute force attack for the data block for 512 bits block is $2^{512}=1.34\times10^{154}$.

### 3.3 Language Statistics

Language redundancy [6] is the greatest problem for any cryptosystem. The cryptanalyst uses the language redundancy to attack cryptosystems ciphertext. If the message is long enough, the cryptanalyst computes the frequency of each of the characters and consider different number of combinations up to the length of the cryptosystem block. The cryptanalyst will then try to estimate the plaintext from this statistical result. A cryptosystem is considered unbreakable against statistical analysis if its ciphertext has flat distribution. To implement the strength of new system, Figs 3&4 show the plaintext statistics of the used file. The ciphertext statistics of AES, IDEA and new proposed system are plotted in Figs 5 to 10.

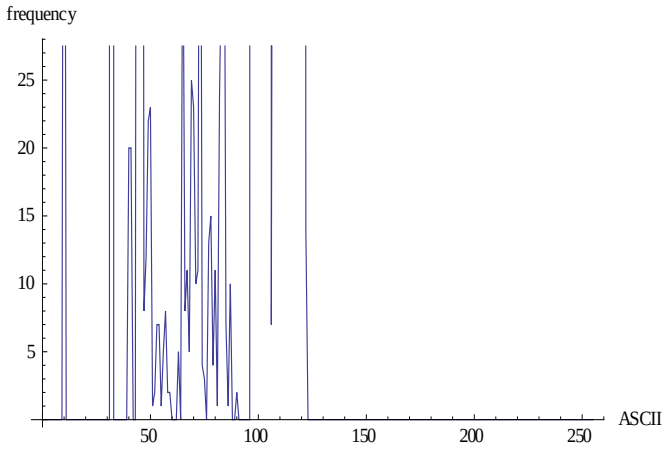


**Figure.3** Plaintext statistics of a text file

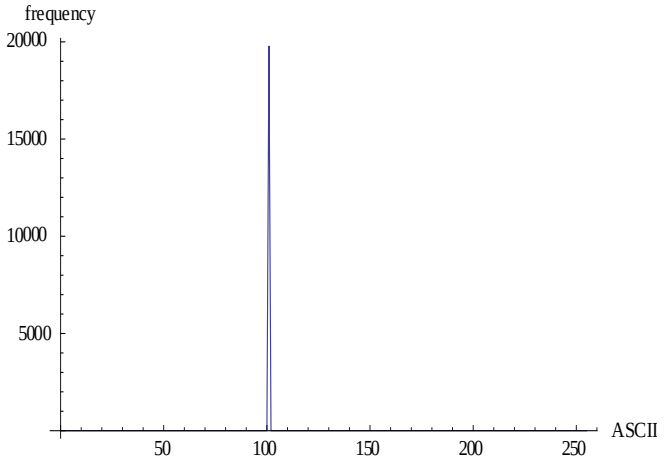


**Figure 4** Plaintext statistics of repeated text file

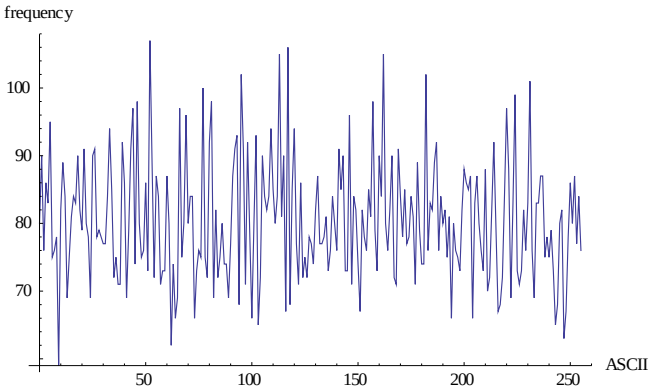


**Figure.5** Proposed system ciphertext statistics

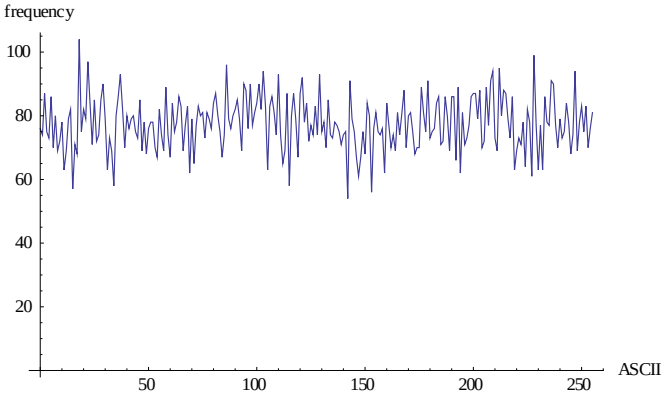


**Figure.6** AES ciphertext statistics

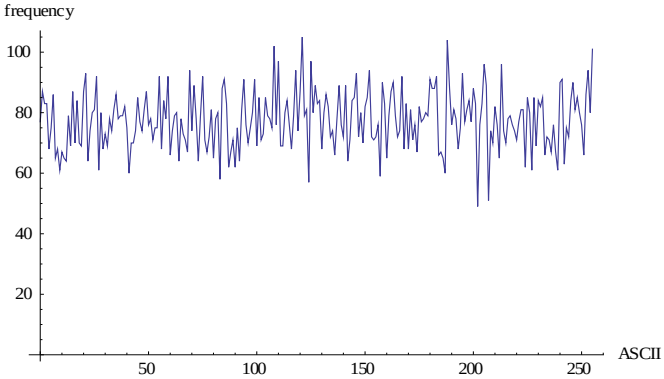


**Figure.7** IDEA ciphertext statistics

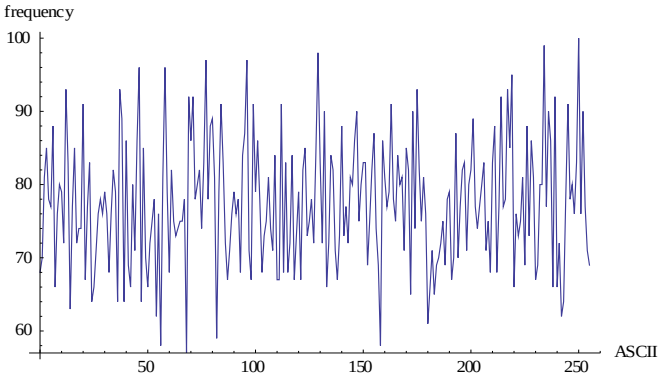


**Figure.8** Proposed system ciphertext statistics of a message consisting of 20 Kbytes of character "e"

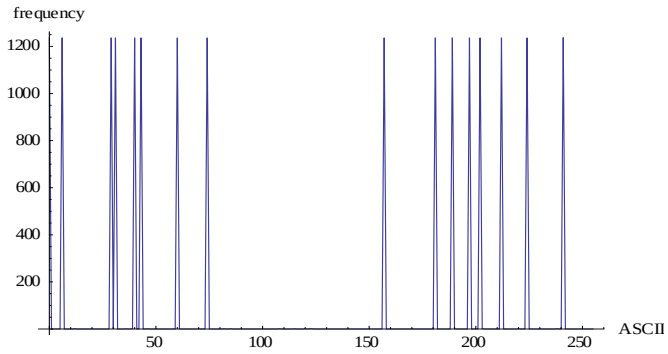


**Figure.9** AES ciphertext statistics of a message consisting of 20 Kbytes of character "e"

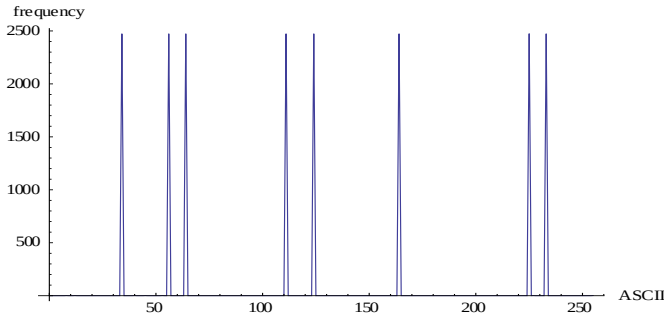


**Figure.9** IDEA ciphertext statistics of a message consisting of 20 Kbytes of character "e"

### 3.4 NIST Statistical suite

The National Institute of Standards and Technology (NIST) [7] develops a Test Suite as a statistical package consisting of 16 tests that were developed to test the randomness of (arbitrarily long) binary sequences produced by either hardware or software based Cryptographic random or pseudorandom number generators. These tests focus on a variety of different types of non randomness that could exist in a sequence. Some tests are decomposable into a variety of subtests. The average values of the statistical tests for KAM-FA, AES and IDEA algorithms were given in Table 4.

**Table.3** Proposed system vs. IDEA Statistical tests

| Algorithm / Test name | KAM-FA | | AES | | IDEA | |
|---|---|---|---|---|---|---|
| Frequency Monobit | 100% | Pass | 100% | Pass | 100% | Pass |
| Frequency within a Block | 100% | Pass | 98% | Failed | 96% | Failed |
| Runs Test | 100% | Pass | 100% | Pass | 100% | Pass |
| the Longest Run of Ones in a Block | 99% | Pass | 99% | Pass | 100% | Pass |
| Binary Matrix Rank | 100% | Pass | 100% | Pass | 99% | Pass |
| Discrete Fourier Transform | 100% | Pass | 100% | Pass | 100% | Pass |
| Non-overlapping Template Matching | 100% | Pass | 99% | Pass | 100% | Pass |
| Overlapping Template Matching | 100% | Pass | 99% | Pass | 99% | Pass |
| Maurer's Universal Statistical | 100% | Pass | 100% | Pass | 100% | Pass |
| Lempel-Ziv Compression | 100% | Pass | 99% | Pass | 98% | Failed |
| Linear Complexity | 99% | Pass | 99% | Pass | 100% | Pass |
| Serial Test | 100% | Pass | 97% | Failed | 88% | Failed |
| Approximate Entropy | 100% | Pass | 99% | Pass | 96% | Failed |
| Cumulative Sums (Cusum) | 100% | Pass | 100% | Pass | 99% | Pass |
| Random Excursions | 100% | Pass | 97% | Failed | 98% | Failed |
| Random Excursions Variant ($\alpha = 0.05$) | 96% | Pass | 96% | Pass | 95% | Pass |

## 4. CONCLUSION

A new symmetric key block ciphering algorithm is proposed in Gaussian domains. We have improved the security by increasing the size of data block to 512 bits and the size of key to 1024 bits. We use modulo the prime field $2^{16}+1$ multiplication to increase the strength of the proposal cipher. We introduce a new optimal diffusion 8x8 MDS matrixes controlled by the subkeys. These new matrixes meet all requirements of optimum design. We use the MDS matrix in the key expansion procedure to make it strong against the known attacks and we use rotor bank to make sure that the subkey never be zero. In our proposed system if we change a few bits in the plaintext or the user key it cause more than half of the ciphertext to be change. A software program using Mathematica 9 language is developed to simulate the proposed cryptosystem. From the statistics of the ciphertext, it is concluded that KAM-FA perfectly hides

ciphertext statistics. We extend the cryptosystem to the domain of Gaussian integer to make the cryptosystem more secure and very difficult to be broken. Finally, our proposal is rigid to withstand the well-known methods of brute-force.

## 5 REFERENCES


1. William Stallings, "Cryptography and Network Security Principles and Practice", Prentice Hall, 2nd Edition, 1995.
2. Behrouz A.Forouzan "Cryptography and network security "TATA-Mcgraw hill publication 2007 edition.
3. Junod, P., Vaudenay, S.: Perfect Diffusion Primitives for Block Ciphers: Building Efficient MDS Matrices. In: Selected Areas in Cryptography (2004).
4. M. Naor and O. Reingold, "On the construction of pseudo-random permutations: Luby- Rackoff Revisited", Proceedings of the 29'th ACM Symposium on Theory of Computing, 1997, pages 189-199.
5. Amish Kumar, "effective implementation and avalanche effect of AES", International Journal of Security, Privacy and Trust Management ( IJSPTM), Vol. 1, No 3/4, August 2012.
6. Bruce Schneier, "Applied Cryptography, Protocols, Algorithms, and Source Code in C" Wiley Computer Publishing, Second Edition, John Wiley & Sons, Inc.
7. NIST, "A Statistical Test Suite for Random and Pseudorandom Generators for Cryptographic Applications", NIST Special Publication 800-22, 2003.