Abstract - Cryptography is a vital part in information handling. It renders the message unintelligible to outsider by various transformations. Data cryptography is the scrambling of the content of data like text, image, audio and video to make it unreadable or unintelligible during transmission. As the data may include some sensitive information which should not be accessed by or can only be partially exposed to the general users. The principal goal guiding the design of any encryption algorithm must be security against unauthorized attacks but performance and the cost of implementation are also important concerns. In this paper, we introduce a new randomized encryption/decryption scheme. It depends on the secret key and a randomly chosen number for every encryption process. The secret key and the random number are used to generate a significant subkey for every block of data. Generation of subkeys is done by simulating a physical operation of turning a movable disk by specific values against a fixed pointer. Forwarding the random number to the receiver is not a trivial job. The algorithm shows high strength for attacks and cryptanalysis.

Keywords – encryption; decryption; ciphertext; secret key; mutation; crossover.

LIST OF SYMBOLS:

- $S$: secret key.
- $SK_j$: subkey blocks.
- $N$: number of data blocks.
- $B_j$: a data block.
- $MB_j$: mutated data block.
- $CB_j$: crossed over data block.
- $EB_j$: encrypted data block.
- $n_0$, $n_s$: size of data block, and key block.
- $TA_j$: turning angles of the movable disk.
- $R$: random number.
- $R^*$: compound number of $R$ and $S$.
- $S_{mj}$: modified key block.
- $R_{mj}$: modified $R$ block.
- $P$: pointer.
- $P_r$: a chosen prim number.
- $MuCr$: mutation(s) and crossover processes.
- $V$: computed number of $R$ and $S$ for modulo calculations.
- $X_1$, $X_2$: fixed large numbers in encryption / decryption sides.
- $I$: Message block number. $1, 2, \ldots, N$.
- $j = I$ if $I \leq 16$, and $j = I \text{ mod } N$ if $I > 16$.

I. INTRODUCTION

Cryptography is where security engineering meets mathematics. It provides us with the tools that underlie most modern security protocols. It is probably the key enabling technology for protecting distributed systems. Cryptography refers almost exclusively to encryption, it is the process of converting ordinary information (called plaintext) into unintelligible gibberish (called ciphertext). It is also used for a variety of other information security issues including electronic signatures, which are used to prove who sent a message. Decryption is the reverse, in other words, moving from the unintelligible ciphertext back to plaintext.

A cipher in cryptosystems is a pair of algorithms that create the encryption and the reversing decryption. The detailed operation of a cipher is controlled by the algorithm and the key. Cryptography was done to attain (CIA) i.e. confidentiality, integrity, and availability [1][2].

There are two types of cryptosystems, one-key (or symmetric key), and two-key (asymmetric key) ciphers. In symmetric key ciphers, the encryption of a plaintext and the decryption of the corresponding ciphertext are performed using the same key. Until 1976 when Diffie and Hellman introduced public-key or two-key cryptography all ciphers were one-key systems [3]. Therefore one-key ciphers are also called conventional cryptosystems. They are widely used throughout the world today, and new systems are published frequently.

There are two types of conventional cryptosystems: stream
ciphers and block ciphers. In stream ciphers, a long sequence of bits is generated from a short string of key bits, and it is then added bitwise modulo 2 to the plaintext to produce the ciphertext. In block ciphers the plaintext is divided into blocks of a fixed length, then they are encrypted into blocks of ciphertext using the same key. The block cipher can be categorized into Feistel structure and SPN (Substitution Permutation Network) one. Feistel structure has an advantage of the same algorithm between encryption and decryption, and the feature of SPN structure is that it has a different algorithm between encryption and decryption. In particular, the SPN structure has a disadvantage that its area increases twice compared with the Feistel one when SPN structure is implemented via hardware. Except DES- 64 and TDES commonly, almost all existing algorithms require 128 bit and a variable length block cipher encryption algorithm, [4]. Block ciphers can be divided into three groups: Substitution ciphers, Transposition ciphers and, Product ciphers.

Since its introduction in 1977, the Data Encryption Standard (DES) has become the most widely applied private key block cipher [5]. Recently, a hardware design to effectively break DES using exhaustive search was outlined by Wiener [6]. AES by its turn is subjected to different cryptanalysis that presumes its ability to break AES. [7][8].

In this paper, a novel randomized scheme is proposed for block cipher. It uses the secret key and a number chosen randomly in every encryption process. Both of them are implemented to generate a different subkey for every block of message. The algorithm makes use of the concept of a physical process as a development of the scheme presented in [10]. A turning disk is divided into 2\(^n\) angles. Rotating it clockwise with respect to a fixed pointer P by a specific value results a new angle value corresponding to P. Hence, this new angle is used to generate a subkey. Fig. 1a shows the original indication of P i.e., angle 0. Fig.1b shows the angle corresponding to P after turning by TA\(_1\) = 77.

Both the secret key and the block size \(n_b\) can be of any chosen size, provided key size modulo block size is zero. In each encryption process a new random number is generated, consequently, resistance to cryptanalysis whatever it is based on differential or linear is increased [9]. Also, the large range of the random number ( \(R < 2^{\text{Key-Size}}\) ) makes brute force attacks infeasible. This scheme has significant strict avalanche effect compared to the other existing algorithms.

The rest of this paper is organized as follows: Section II addresses the proposed encryption/decryption algorithm, the generation of subkeys, the mutation and crossover processes for message blocks. Also, it explains the encryption process of the random number. Section III provides two comparative examples in comparison of strict avalanche effects with those of existing algorithms. Section IV addresses performance and analysis of the proposed algorithm in comparison with other algorithms. Section V summarizes the conclusions.

II. THE PROPOSED ALGORITHM

The sizes of the secret key and data block can be chosen of any numbers. The only condition is that (key size) mod (block size) = 0. Let us consider data blocks \(B_1, B_2, \ldots, B_n\), size of each is \(n_b\) = 8 bit. Key size 128 bit, 16 block × 8 bit, \(S_1, S_2, \ldots, S_{16}\), and let the prim number \(P_r\) to be 131.

The turning disk is divided into 2\(^{128}\) angles. Originally, angle 0 corresponds to the fixed pointer P. Turning it by a certain number of angles e.g., TA\(_1\) makes the value corresponding to P to be TA\(_1\). Turning it again by TA\(_2\), the angle corresponding to P will be TA\(_1\) + TA\(_2\), and so on. These angle values TA\(_j\) will be used with modified blocks of R and S i.e., \(R_m, S_m\) to generate a significant subkey for each data block \(B_j\).

Fig. 2 shows the block diagram of the proposed algorithm, It can be summarized in the following steps:

A. Chose a Random Number \(R < 2^{128}\) i.e., \(R\) lies in the range from 0 to \(5.44 \times 10^{39}\). In every encryption process \(R\) is newly generated.

B. Choose a Fixed Pointer P as the value of one of the S blocks, e.g., \(P = S_7\).

C. Compute the Modified Key Blocks. This operation is simply computing the modus of each separate key block \(S_j\) with a suitable number \(V_j\). The reason of computing \(V_j\) is to strengthen the encryption process for short messages. This is done by exploiting all blocks of S and all blocks of \(R\) in \(V_j\). For \(j = 1, 2, \ldots, 16\).

\[
S_{mj} = S_j \text{ mod } V_j. \quad (1)
\]

\[
V_j = R \text{ mode } (S' \oplus j). \quad (2)
\]

\[
S' = S_1 \oplus S_2 \oplus S_3 \oplus \ldots \ldots \ldots S_{16}. \quad (3)
\]

D. Compute the Modified R Blocks \(R_m\); Similar to \(S_m\), the modified \(R\) blocks of random number are computed as:

\[
R_{mj} = R_j \text{ mod } S_j. \quad (4)
\]

E. Generate a Specific Values (TA\(_j\) angles ) to be turned by the disk. Each block number I of the message has a specific turning angle, and consequently a specific subkey. It considers into its computation the corresponding modified random block, the modified key block, and the previous turned angle, it is computed for all message blocks ( \(j = 1\) to \(N\)) as:

\[
TA_j = (TA_{j-1} + I + R_{mj})^{sm} \text{ mod } (P_r) \quad (5)
\]
From (10) compute SR, and then R is computed as:

$$R = (R^- \oplus SR) + X2$$  \hspace{1cm} (12)$$

After getting the random number the decryption process goes on as encryption steps.

### III. COMPARATIVE EXAMPLES

Two examples are explained here, each example encrypts two messages. The difference between the two messages is 1 bit flipped, the results show the avalanche effect of the algorithm.

**A. Example 1:**

The secret key is: EXAMPLES

In decimal: S1 = 69, S2 = 88, S3 = 65, S4 = 80, S5 = 76, S6 = 69, S7 = 83, S8 = 13, S9 = 10.

In binary: 0100,0101, 0101,1000, 0100,0001, 0101,0000, 0100,1100, 0100,0101, 0101,0011, 0000,1101, 0000,1010.

Message 1: NETWORK = 0100,1110, 0100,0101, 0101,0100, 0101,0111, 0100,1111, 0101,0010, 0100,1011, 0101,0011.

Message 2: NETWORK = 0100,1110, 0100,0101, 0101,0100, 0101,0110, 0100,1111, 0101,0010, 0100,1011, 0101,0011.

Message block size is 8 bit. Number of message blocks N = 8.

Let the random number R = 1000 = 0000,0011, 1110,0011, and the prime number Pr = 131.

Choose the pointer P to be S7, so P = 83 = 0101,0011.

- The modified subkeys are Sm1 = Sj mod 32.

Sm1 = 5, Sm2 = 24, Sm3 = 1, Sm4 = 16, Sm5 = 12, Sm6 = 5, Sm7 = 19, Sm8 = 13, Sm9 = 10.

- The modified random blocks: Rm1 = R mod Sj.

Rm1 = 34, Rm2 = 32, Rm3 = 25, Rm4 = 40, Rm5 = 12, Rm6 = 34, Rm7 = 32, Rm8 = 12, Rm9 = 0.

- The turning angles for each block: $TA_j = (TV_{j+1} + I + Rm_j)^{30} \mod (P_r)$.


- The subkeys are: $SK_j = (SK_{j-1} + P + TV_j)^{3} \mod (P_r)$:

<table>
<thead>
<tr>
<th>j</th>
<th>SK1</th>
<th>SK2</th>
<th>SK3</th>
<th>SK4</th>
<th>SK5</th>
<th>SK6</th>
<th>SK7</th>
<th>SK8</th>
<th>SK9</th>
<th>SK10</th>
<th>SK11</th>
<th>SK12</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0110,1000</td>
<td>0111,1111</td>
<td>0000,1011</td>
<td>0001,0010</td>
<td>1010,0011</td>
<td>0000,0010</td>
<td>0000,0000</td>
<td>0110,0001</td>
<td>0000,1111</td>
<td>0101,0000</td>
<td>0001,1111</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>j</th>
<th>SK1</th>
<th>SK2</th>
<th>SK3</th>
<th>SK4</th>
<th>SK5</th>
<th>SK6</th>
<th>SK7</th>
<th>SK8</th>
<th>SK9</th>
<th>SK10</th>
<th>SK11</th>
<th>SK12</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0110,1000</td>
<td>0111,1111</td>
<td>0000,1011</td>
<td>0001,0010</td>
<td>1010,0011</td>
<td>0000,0010</td>
<td>0000,0000</td>
<td>0110,0001</td>
<td>0000,1111</td>
<td>0101,0000</td>
<td>0001,1111</td>
<td></td>
</tr>
</tbody>
</table>
-Mutation Process of Message Blocks:
At arbitrary bit number perform self mutation (bit number 4 is chosen here) as follows: First block of message is \( M_1 = 0100,1110 \) m will be \( MM_1 = 1110,0100 \). With the same procedure: \( MM_2 = 0101,0100, MM_3 = 0111,0101, MM_4 = 111,0100, MM_5 = 0010,0101, MM_6 = 1011,0100, MM_7 = 0011,0101. 

-Crossover Process:
At another arbitrary bit number perform cross over processes between adjacent mutated blocks (bit number 4 is chosen here for simplicity) as:
\[ MM_1(1110, 0100) \text{ to be crossed over with } MM_2 = 0101,0100 \text{, to get } CM_1 = 1110,0100, \text{ and } CM_2 = 0101,0101, \text{ CM}_3 = 0110,0101, \text{ CM}_4 = 1011,0101, \text{ CM}_5 = 1111,0101, \text{ CM}_6 = 0010,0100, \text{ CM}_7 = 1011,0101, CM_8 = 0011,0100. \]

-Exclusive Oring:
To encrypt the crossed over blocks, XOR it with the previous one and its corresponding subkey,
\[ EM_I = EM_{I-1} \oplus CM_I \oplus SK_I. \]
The encrypted message (NETWORK) is:
\[
\begin{align*}
00111,11011 & \quad 10001,10100 & \quad 10101,01000 & \quad 10100,1001  \\
11011,01001 & \quad 01001,11011 & \quad 11011,00111
\end{align*}
\]

-Exclusive Oring:
To encrypt the crossed over blocks, XOR it with the previous one and its corresponding subkey,
\[ EM_I = EM_{I-1} \oplus CM_I \oplus SK_I. \]
The encrypted message (NETWORK) is:
\[
\begin{align*}
1101,0110 & \quad 1111,0110 & \quad 0111,0001 & \quad 0000,1110, \\
1000,1100 & \quad 1011,1011 & \quad 0110,0001 & \quad 0101,0111.
\end{align*}
\]

Applying the same procedure for the second message (NETWORK) we get:
\[
\begin{align*}
1101,0001 & \quad 1101,1111 & \quad 0011,0110 & \quad 1011,1100, \\
0010,0111 & \quad 1110,0111 & \quad 0101,0111 & \quad 1101,0011.
\end{align*}
\]
The total message length is 64 bits. By flipping one bit in the message, number of bits changed in the encrypted message is 34 bit. This means flipping one bit in the message causes 53.1 % of encrypted message bits to be flipped.

The same example was carried out by other algorithms [11], and the results are shown in Table I. It shows number of flipped bits FB and its percentage.

B. Example 2:
The input plaintext is "DISASTER".

Flipping one bit from the plaintext, we get "DISCSTER", (one flipping A (01000001) to C (01000011)).

The Key used is “SRIRAMSR”.

DISASTER encrypted message is:
\[
00111,11011 \quad 10001,10100 \quad 10101,01000 \quad 10100,10011
\]

DISCSTER encrypted message is:
\[
01010,00110 \quad 00100,01011 \quad 01100,11100 \quad 11110,10000
\]

Number of flipped bits in the encrypted message is 42 bit (out of 65 bits of the original message).

Avalanche effect = $42 \times 100 / 65 = 64.6\%$.

The same example was carried out by other algorithms [12], and the results are shown in Table II.

### TABLE I COMPARISON OF AVALANCHE EFFECT OF EXAMPLE 1

<table>
<thead>
<tr>
<th>S N</th>
<th>Encryption Techniques</th>
<th>ECB</th>
<th>%</th>
<th>CBC</th>
<th>%</th>
<th>CFB</th>
<th>%</th>
<th>OFB</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DES</td>
<td>33</td>
<td>51.5</td>
<td>34</td>
<td>53.1</td>
<td>20</td>
<td>31.2</td>
<td>1</td>
<td>1.5</td>
</tr>
<tr>
<td>2</td>
<td>AES</td>
<td>69</td>
<td>53.9</td>
<td>66</td>
<td>51.5</td>
<td>16</td>
<td>25</td>
<td>1</td>
<td>4.5</td>
</tr>
<tr>
<td>3</td>
<td>BLOFISH</td>
<td>34</td>
<td>53.1</td>
<td>31</td>
<td>48.4</td>
<td>20</td>
<td>31.2</td>
<td>1</td>
<td>1.5</td>
</tr>
</tbody>
</table>

IV ANALYSIS AND PERFORMANCE

The algorithm is tested, results are compared with known existing algorithms, the summery of its performance and analysis is explained hereinafter.

A. Performance

The algorithm runs on a 3.2 GHz PC with different lengths messages, the concluded speed is 5.19 cycle per byte (587 MiB/s) for the encryption process, which is very fast compared to different algorithms shown in the Table III benchmark, [13].

Number of overhead bits is fixed and irrelevant to the message length (double the key size) i.e., 128*2 bits only for random number mutated and crossed over with \( X_1 \).

B. Analysis

Although the algorithm does not depend on substitution permutation networks (SPNs), it keeps its cryptographic static and dynamic prosperities. Its strict avalanche effect causes 64.6 % of bits in average to be flipped in the enciphered text for one bit flipped in plaintext as shown in the Table 1.
TABLE II COMPARISON OF AVALANCHE EFFECT

<table>
<thead>
<tr>
<th>Encryption Technique</th>
<th>No. of flipped bits</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Playfair Cipher</td>
<td>4</td>
<td>6.25</td>
</tr>
<tr>
<td>Vigenere Cipher</td>
<td>2</td>
<td>3.13</td>
</tr>
<tr>
<td>Caesar Cipher</td>
<td>1</td>
<td>1.56</td>
</tr>
<tr>
<td>DES</td>
<td>35</td>
<td>54.68</td>
</tr>
<tr>
<td>Blowfish</td>
<td>19</td>
<td>28.71</td>
</tr>
<tr>
<td>The Proposed technique</td>
<td>42</td>
<td>64.6</td>
</tr>
</tbody>
</table>

Strict avalanche criterion is a measure of a cipher’s randomness. High randomness ensures the algorithm resistance to statistical, clustering, linear, and differential cryptanalysis. So that, when encrypting the same message several times, it will produce different cipher texts each time.

Key size can be of any chosen number. Consequently, block size can be larger, e.g., key size of 1024 bit with block 128 or 256 or 512, also, the larger the key size the larger range for the random number. \( 0 < R < 2^{\text{key-size}} \). Consequently, attacking the algorithm become harder.

The algorithm has two different arbitrary bit numbers (from 1 to block_size -1), one for mutation and the other for crossover process. Also, number of (mutation then crossover) rounds can be increased to any chosen number. In this case each mutation process can be done at a different bit number. Another list of bit numbers can be used for crossover processes.

The number of brute force trials in the worst case is \( 9.7 \times 10^{89} \) or \( 1.02 \times 10^{71} \) years (assuming \( 10^{10} \) decryption process per second). In a known message attack, if the attacker knows both plain and ciphered messages completely, he cannot get R because it is not send, but a functions of it is used. To get the function of R, i.e., \( R' \), the attacker has to make reverse mutation and crossover in \( 8^2 \) operations, (nine possible mutation bit numbers, and for each one nine possible crossover bit number). While guessing R from \( R' \) and \( X_2 \), there are \( 2^{128} \) possible changes in \( R' \). So, the attacker needs \( 4.6 \times 10^{91} \) or \( 1.45 \times 10^{89} \) years, (assuming \( 10^{10} \) decryption process per second).

The algorithm time and space complexity is \( O(n) \), i.e., the required time and space for encryption/decryption increases linearly with message length. However attacker algorithm is NP complete

TABLE III ALGORITHM SPEED COMPARISON

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>MiB/Second</th>
<th>Cycles Per Byte</th>
</tr>
</thead>
<tbody>
<tr>
<td>AES/GCM (2K tables)</td>
<td>102</td>
<td>17.2</td>
</tr>
<tr>
<td>AES/GCM (64K tables)</td>
<td>108</td>
<td>16.1</td>
</tr>
<tr>
<td>AES/CCM</td>
<td>61</td>
<td>28.6</td>
</tr>
<tr>
<td>AES/EAX</td>
<td>61</td>
<td>28.8</td>
</tr>
<tr>
<td>CRC32</td>
<td>253</td>
<td>6.9</td>
</tr>
<tr>
<td>Adler32</td>
<td>920</td>
<td>1.9</td>
</tr>
<tr>
<td>MD5</td>
<td>255</td>
<td>6.8</td>
</tr>
<tr>
<td>DES/CTR</td>
<td>32</td>
<td>54.7</td>
</tr>
<tr>
<td>DES-XEX3/CTR</td>
<td>29</td>
<td>60.6</td>
</tr>
<tr>
<td>DES-EDE3/CTR</td>
<td>13</td>
<td>134.5</td>
</tr>
<tr>
<td>PROPOSED ALGORITHM</td>
<td>590</td>
<td>5.17</td>
</tr>
</tbody>
</table>

V. CONCLUSIONS

In this paper, a randomized, novel, and immune scheme for block cipher encryption is introduced. The scheme shows high strength of confidentiality even for known message attack. In the proposed scheme, both the key length and the block size can be of any chosen size, provided key length modulo block size is zero.

The algorithm has a high degree of randomness; its strict avalanche effect is very significant and surpasses a lot of the famous algorithms. This proves its immunity to cryptanalysis. Knowing the algorithm, the plaintext and the ciphered text does not reveal useful information for the attacker to crack the key or the random number, because in every run of the algorithm a new random number and new subkeys are generated, consequently different ciphered texts for the same plaintext.

The algorithm has a strong strict avalanche criterion (SAC) (65% in average). Also, it keeps the cryptographic static properties of substitution permutation networks (SPNs) of completeness, nonlinearity. In the same time the algorithm provides perfect security, and every bit in the information message is encrypted using a different subkey.

Implementing random numbers of a large range \( (2^{\text{key-size}}) \) in encryption, makes cracking it is infeasible. For the chosen length (128 bit), brute force attack needs \( 1.02 \times 10^{71} \) Years to crack that
key, (assuming $10^{10}$ check per second). Also, it is safely distributed between sender and receiver in a new procedure.

As an additional feature of the scheme some of its parameters are selective and not fixed e.g., Key size – Block size – Bit number to perform mutation – Bit number to perform crossover – number of mutation then crossover rounds.

An attractive feature of the algorithm is that, its time and space complexity is $O(n)$. So, the required memory resources and computation time are not increased in a large scale with the increase of message length. In the same time the attacking algorithm is NP complete.

REFERENCES


Figure 1-A. Initial Position of a Moving Disk.

Figure 1-B. Same Disk After Rotating by Angle =77.
Figure 2 Block Diagram of the Proposed Algorithm.