SYMMETRIC SYNCHRONOUS STREAM ENCRYPTION USING IMAGES

Major Project Report

SUBMITTED IN PARTIAL FULLFILMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF

Bachelor of Technology

In

Computer Engineering

Under The Supervision of

Dr. Mohd Amjad

Submitted by:

MOHD AMIR AHMAD(11CSS35)

MD ABDULLAH HAMMAD(11CSS32)

Department of Computer Engineering
Faculty of Engineering and Technology
Jamia Millia Islamia , New Delhi – 110025

CERTIFICATE

This is to Certify that dissertation/Project Report entitled **"Symmetric synchronous stream encryption using images"** has been successfully completed by **Mohd Amir Ahmad (11CSS-35)** and **Mohammad Abdullah Hammad (11CSS-32)** is an Authentic work carried out by them at Faculty of Engineering and Technology, Jamia Millia Islamia under my guidance. The matter embodied in this project work has not been submitted earlier for the award of any degree or diploma to the best of my Knowledge and Belief.

Date: 23/05/2015

Dr. Mohd Amjad
Assistant Professor
Dept. Of Computer Engineering
Faculty of Engg & Technology
Jamia Millia Islamia
New Delhi-110025

Dr. Tanvir Ahmad
Head of Department
Dept. Of Computer Engg
Faculty of Engg & Tech.
Jamia Millia Islamia
New Delhi-110025

ACKNOWLEDGEMENT

An understanding of the presented work is never the outcome of the efforts of a single person. We take this opportunity to express our profound sense of gratitude and respect to all those who helped us throughout the duration of this project. We are greatly indebted to our supervisor and guide **Dr. Mohd Amjad**, Department of Computer Engineering for his invaluable technical guidance, great innovative ideas and overwhelming moral support during the course of the project. Most of the novel methods and solutions found in his work are the results of our numerous stimulating discussions. We are also thankful to all the faculty members of the Department and our friends who directly or indirectly helped us in completion of our project.

MOHD AMIR AHMAD (11CSS-35)
MOHD ABDULLAH HAMMAD (11CSS-32)

Department of Computer Engineering,
Faculty of Engineering and Technology,
Jamia Millia Islamia, New Delhi

ABSTRACT

A symmetric synchronous stream encryption technique is suggested. The proposed technique utilizes a selected block from the parity bit plane of a public image as a nonlinear function to confuse the relation between the ciphered bits and the key stream. Using the proposed technique, any sort of message like text or image can be encrypted as well as decrypted bit by bit. Simulation results showed that ciphered grey scale images have flat histogram with entropy almost equal to eight and a correlation between pixels in all directions which is almost equal to zero. The technique is highly sensitive to any change in the starting pixel of the selected block or the initial condition of the pseudo-random generator.

Keywords:
Symmetric cryptography
Private key
Synchronous key generator
Image encryption
Data security

TABLE OF CONTENTS

1. **Introduction**………………………………………………………
   1.1 Overview……………………………………………………7
   1.2 Cryptosystem…………………………………………….7
   1.3 Block Diagram……………………………………………8
   1.4 Other Technique………………………………………8
   1.5 What's new in this Project…………………………8
2. **Procedure**……………………………………………………………
   2.1 Design………………………………………………………11
   2.2 Theory………………………………………………………11
3. **Implementation**
   3.1 Algorithm…………………………………………………15
4. **Simulation Results and Discussion**……………………
   4.1- Statistical Analysis………………………………20
      4.1.1- Histogram……………………………………20
      4.1.2- Entropy…………………………………………22
   4.2- Correlation Analysis………………………………25
      4.2.1- Key Sensitivity Analysis……………………28
      4.2.2- Key Space Analysis……………………………28
   4.3- Computational Complexity………………………28
5. **Conclusion**……………………………………………………………
   5.1 Conclusion………………………………………………31
   5.2 Limitations………………………………………………31
   5.3 Future Scope……………………………………………32
6. **Programming Env. And Tools Used**…………………32
7. **References**……………………………………………………33

CHAPTER 1

INTRODUCTION

1.INTRODUCTION:

1.1 Overview

With the rapid development in digital multimedia technology, there has been a great demand for the security of digital images. Image encryption is one of the tools of protecting the digital images. It is a process of realigning the original image into an incomprehensible one that is not recognizable in appearance. Traditional data encryption algorithms such as DES, triple DES, RSA, IDEA or AES are not suitable for image encryption due to some intrinsic properties of image such as high redundancy and strong correlation among pixels. Shannon suggested that confusion and diffusion are the two basic techniques to overcome high redundancies and strong correlations.

1.2 Cryptosystem

Cryptosystems are classified into symmetric and asymmetric cryptosystems based on the type of the key, where the former one uses secret key and the later one uses public key. The symmetric cryptosystems can be further subdivided into block and stream ciphers. Block ciphers works on large blocks of plaintext message and has a fixed transformation over it, whereas stream ciphers works on individual plaintext bits and the transformation varies over time. Stream ciphers are mainly prevalent in military, telecommunication and business applications. Their basic design philosophy is inspired by the Vernam (One-Time-Pad) cipher, which encrypts by XOR'ing the plaintext with a random key.

The drawback of the Vernam cipher is the requirement that key must be a true random sequence, shared by the sender and the receiver, and can only be used once. This poses a practical problem in terms of key generation and distribution. Instead, stream ciphers expand a given short random key into a pseudo-random key stream, which is then XOR'ed with the plaintext to generate the cipher text. The security of stream cipher depends on the pseudo-random key stream that must be of sufficient size and randomness. Hence, key stream generator is incredibly a vital building block for stream cipher algorithms. At the receiver, the plain text is recovered by generating the identical pseudo-random sequence of bits such that it is exactly synchronized with the received cipher text stream.

1.3 Block Diagram

The block diagram of stream encryption is as shown below.

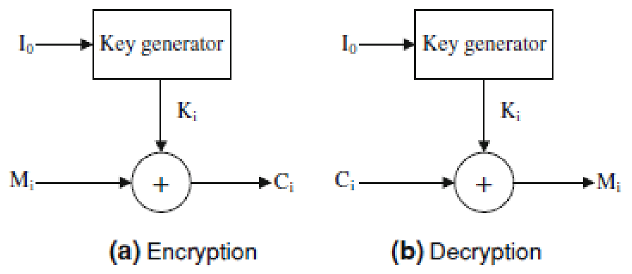


Fig.1 : Block diagram of stream encryption

The simplest way of generating the pseudo-random key streams using hardware is by making use of linear feedback shift registers (LFSR). However, pseudo-random sequence produced by LFSRs, is very vulnerable to known plain text attacks, due to the linear relation of key stream and the ciphered bits. One of the solutions to overcome the above drawback is to make the relation between key stream and cipher text nonlinear. This can be done using chaotic or nonlinear functions.

1.4 Other Technique

There have been many suggested image encryption techniques that use chaotic functions. In a number of chaotic cryptosystems that have been proposed, the chaotic pseudo-random key streams play a central role, including generating cryptographic keys and initializing variables in cryptographic protocols randomly. With researches of chaotic cryptology going more thorough, some fatal defects have been discovered, which discourage practical applications of these cryptosystems. For example, the equivalence between the initial condition and the chaotic symbolic trajectory makes such cryptosystems very weak.

1.5 What's new in this Project?

In this project, we introduce a new image encryption technique that does not use chaotic functions. The proposed algorithm, which belongs to the family of symmetric synchronous stream encryption techniques, uses a selected block from the parity bit plane of a public image as a nonlinear function to encrypt messages. The selected block is randomly accessed using the content of a LFSR to get a single bit. This bit is Xored with the message bit to generate the ciphered bit. The process is repeated until all message bits are streamed. The proposed stream cipher is sensitive to the starting pixel of the selected block and the content of the LFSR. It has a simple structure and can generate a key stream faster than other generators.

CHAPTER 2

PROCEDURE

2. PROCEDURE:

2.1 Design

The block diagrams of the encoder and decoder of the proposed encryption technique are shown in Figs. 2 and 3, respectively.

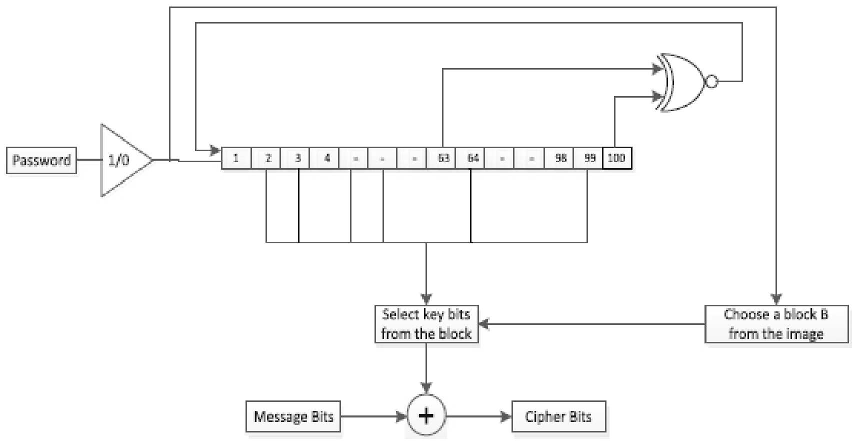

Password
1/0
1
2
3
4
63
64
98
99
100
Select key bits from the block
Choose a block B from the image
Message Bits
+
Cipher Bits

Fig.2 : Block diagram of the encoder for the proposed encryption technique

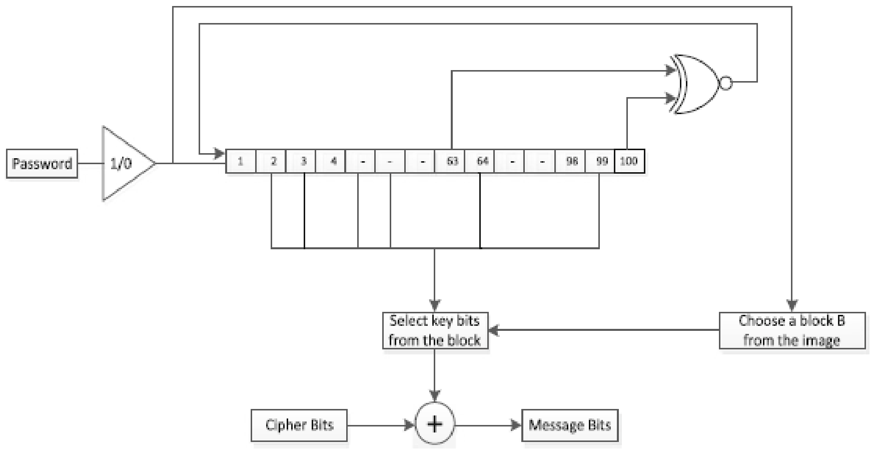
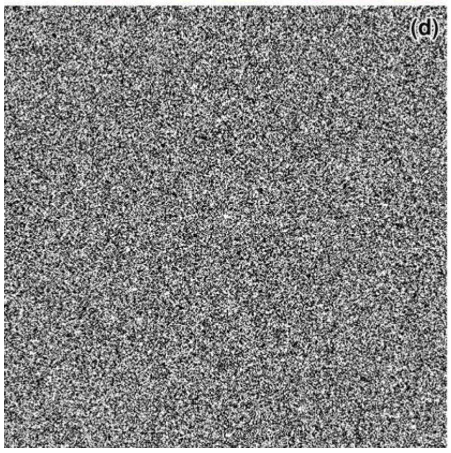

Password
1/0
1
2
3
4
63
64
98
99
100
Select key bits from the block
Choose a block B from the image
Cipher Bits
+
Message Bits

Fig.3 : Block diagram of the decoder for the proposed encryption technique

2.2 Theory

The encryption technique is symmetric and synchronous. It assumes a database of colored images that are available for the public. That is, everybody has access to these images. It is worth mentioning here that several websites of financial companies are already asking users who want to access private account information, in addition to picking a password, to also select an image from their available database. The images should be selected carefully so that they do not include smooth regions, such as, clear blue sky or a uniform background. For example, the famous Lena image will be perfectly suitable for this technique. The parity bit plane of the different public colored images is calculated by Xoring the different bits representing each pixel. For a colored image, each pixel is usually represented by 24 bits. The parity bit plane P of the colored image will be used by the proposed technique to encrypt the message bits. After selecting P from the available database of public images, the password of the user will be transformed into a binary private key. The private key will be divided into two parts. The first part of the private key will be used as an address of a random pixel P(n, k). The length of this address depends on the size of the image.



For example, if the image is $M×N$ pixels, the address will be $(M×N)$ bits.

A block B of size $L × W$ starting from point $P(n, k)$ will be extracted from P based on the following algorithm:

```
for $i = 0 : L - 1$
for $j = 0 : W - 1$
$B(i, j) = P((n + i) \bmod M, (k + j) \bmod N)$
End;
End;
```

The second part of the private key will be used to initialize the LFSR of length *U*. An address S of length log2*(L×W)* is obtained from the content of the LFSR based on a hardwired combination of bits, as shown in Fig. 2. Note that both the hardware of the LFSR and the combination of bits creating the address are not private. This address is used to randomly access a single bit from *B*. The selected bit is Xored with the message bit to get the ciphered bit. This process is repeated until all the message bits have been streamed into the encoder.

The decoder follows the same steps of the encoder to select the block B and save it in cache. The LFSR will be initialized by the second part of the private key. To decrypt the message, the ciphered bits are simply Xored with the randomly selected bits from *B*.

CHAPTER 3

IMPLEMENTATION

3. IMPLEMENTATION

3.1 Algorithm

Step 1: Firstly the images should be selected carefully so that they do not include smooth regions, such as, clear blue sky or a uniform background. For example, the famous Lena image will be perfectly suitable for this technique.

Step 2: The parity bit plane of the different public colored images is calculated by Xoring the different bits representing each pixel. For a colored image, each pixel is usually represented by 24 bits.

```
for(int i=0;i<height;i++) {
          for(int j=0;j<width;j++) {
            Color c = new Color(image.getRGB(j, i));
            int red=c.getRed();
            int green=c.getGreen();
            int blue=c.getBlue();
            String redbits=toBinary(red);
            String greenbits=toBinary(green);
            String bluebits=toBinary(blue);
            int xor=Integer.parseInt(redbits.charAt(0)+"");
            for(int m=1;m<redbits.length()-1;m++) {
                        xor^=Integer.parseInt(redbits.charAt(m)+"");
            }
            for(int m=0;m<greenbits.length();m++) {
                        xor^=Integer.parseInt(greenbits.charAt(m)+"");
            }
            for(int m=0;m<bluebits.length();m++) {
                        xor^=Integer.parseInt(bluebits.charAt(m)+"");
            }
            xorbits=xorbits.append(xor);
            if(xor==1)pixels[i][j]=255;else pixels[i][j]=0;
          }
        }
```

Step 3: The parity bit plane *P* of the colored image will be used by the proposed technique to encrypt the message bits.
Step 4: After selecting *P* from the available database of public images, the password of the user will be transformed into a binary private key.
Step 5: The private key will be divided into two parts. The first part of the private key will be used as an address of a random pixel *P(n, k)*. The length of this address depends on the size of the image.

```
void passwordBreak() {
            for(int i=0;i<100;i++) {
                  a[i]=Integer.parseInt(password.charAt(i)+"");
            }
}


void selectBlock() {
            try {
                  System.out.println("Enter the value of n, k, L and W  :
");
      sc=new Scanner(System.in);
      int n=sc.nextInt(); int k=sc.nextInt();
      L=sc.nextInt(); W=sc.nextInt();
      B=new int[L][W];
      for(int i=0;i<=L-1;i++) {
            for(int j=0;j<=W-1;j++) {
                  B[i][j]=pixels[(n+i)%height][(k+j)%width];
            }
      }
     S=(int)Math.floor(Math.log(L*W)/Math.log(2));
     key=new int[S];
}
```

Step 6: The second part of the private key will be used to initialize the LFSR of length*U*. An address S of length log2*(L×W)* is obtained from the content of the LFSR based on a hardwired combination of bits.

```
void lfsr() {
   int T=height*width,j=0;
   int N=a.length;
   int TAP=63; rSelect();

   for (int t=0;t<T;t++) {
      int next=(a[N-1]^a[TAP-1]);
      for (int i=N-1;i>0;i--)
        a[i]=a[i-1];
      a[0]=next;
      keyGen();
   }
   }
```

Step 7: This address is used to randomly access a single bit from *B*. The selected bit is Xored with the message bit to get the ciphered bit. This process is repeated until all the message bits have been streamed into the encoder.

```
void cipherImage()throws IOException {
                int m=0;
                for(int i=0;i<height;i++) {
       for(int j=0;j<width;j++) {
         cipherpix[i][j]=pixels[i][j]^xbits[m++];
       }
     }
     BufferedImage im=new
BufferedImage(width,height,BufferedImage.TYPE_BYTE_GRAY);
     for(int i=0;i<height;i++)
       for(int j=0;j<width;j++)
         im.setRGB(j,i,cipherpix[i][j]);
     File file=new File(outputimage);
     ImageIO.write(im, "png", file);
         }
```

Step 8: The decoder follows the same steps of the encoder to select the block B and save it in cache. The LFSR will be initialized by the second part of the private key. To decrypt themessage, the ciphered bits are simplyXored with the randomly selected bits from *B*.

```
void decipherImage()throws IOException {
   int m=0;
   for(int i=0;i<height;i++) {
       for(int j=0;j<width;j++) {
         pixels[i][j]=cipherpix[i][j]^xbits[m++];
       }
     }
     BufferedImage                                          im=new
BufferedImage(width,height,BufferedImage.TYPE_BYTE_GRAY);
     for(int i=0;i<height;i++)
       for(int j=0;j<width;j++)
         im.setRGB(j,i,cipherpix[i][j]);
     File file=new File(img);
     ImageIO.write(im, "png", file);
  }
}
```

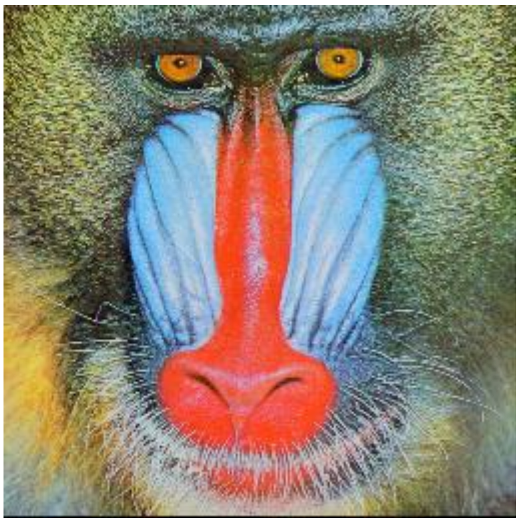
CHAPTER 4

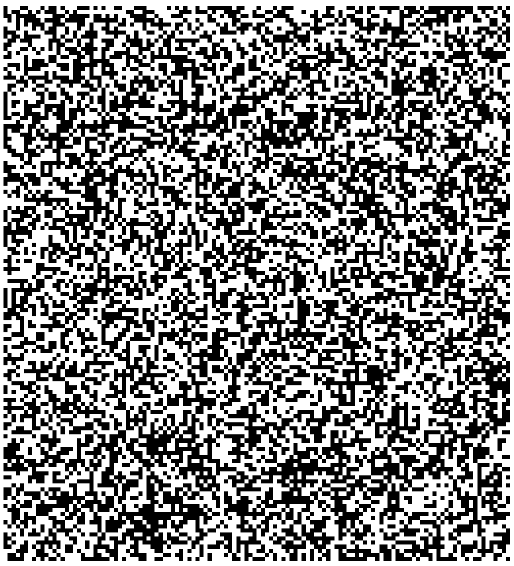
SIMULATION RESULTS & ANALYSIS

4. Simulation results and discussion

Using the proposed stream cipher encryption technique, any sort of message like text or image can be encrypted as well as decrypted bit by bit. In this section, we do experiments for validating the security and practicability of the proposed algorithm. The proposed encryption technique is implemented in MATLAB (Mathworks, Inc., Natick, MA).

An example of a source image of size 200×200 is shown in Fig. a, and its corresponding encrypted image is shown in Fig. b.


(a) a source image of size 200×200


(b) corresponding encrypted image

In cryptography, an effective encryption algorithm should have desirable features for surviving all kinds of known attacks, including brute force, statistical and sensitivity attacks [16,17]. With the present state of art computational power, the encryption algorithms are very vulnerable in a real
time if the system is not designed to look into these issues.

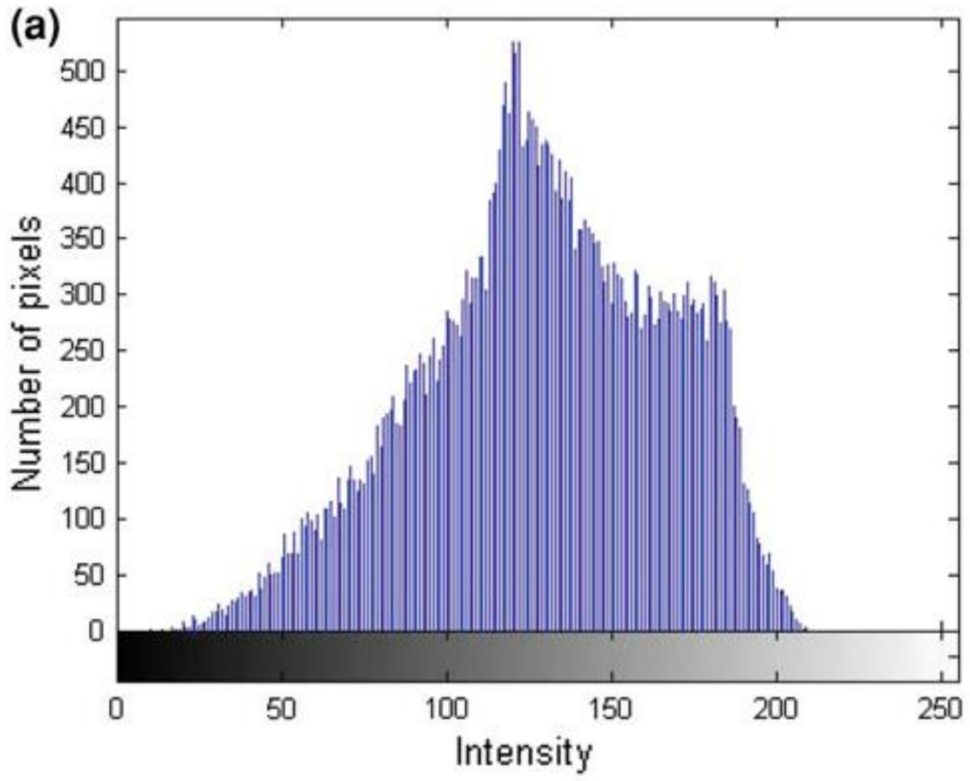
Hence, a good encryption scheme should be robust from all possible attacks. The analysis of these attacks ensures right development of the security system. We evaluate the security of the cryptosystem from the following aspects: statistical analysis, correlation analysis, key sensitivity analysis and key space analysis.

4.1 Statistical analysis

Statistical attack utilizes the intrinsic property of images to carry out cryptanalysis. In [2], Shannon pointed out that a large portion of the ciphers can be attacked by statistical means. Statistical analysis generally depends on the measure of the randomness of the ciphered image. Ciphered image must have the ability to cover up the statistical properties of the source image. It is eminent that a lot of encryption techniques have been successfully analyzed with the help of statistical analysis, and numerous statistical attacks have been formulated on them. We analyze the statistical properties of the ciphered images using the following two tests: **histogram and entropy**. The same tests were used in [6,7,18].

4.1.1 Histogram

The histogram of the source image is shown in Fig. 6a, and the histograms of ciphered image using the proposed technique are shown in Fig. 6b. The histogram of the source image has large spikes, whereas the histogram of the ciphered image is nearly smooth and uniform, representing almost equivalent
probability of occurrence for each intensity level. From the histograms, it can be seen that the cipher image does not tolerate any statistical similarity to the input image and also does not provide any clue to use any kind of statistical attack
on the proposed technique. Also, the flat intensity distribution of the ciphered image histogram suggests that the proposed technique possesses good confusion properties.



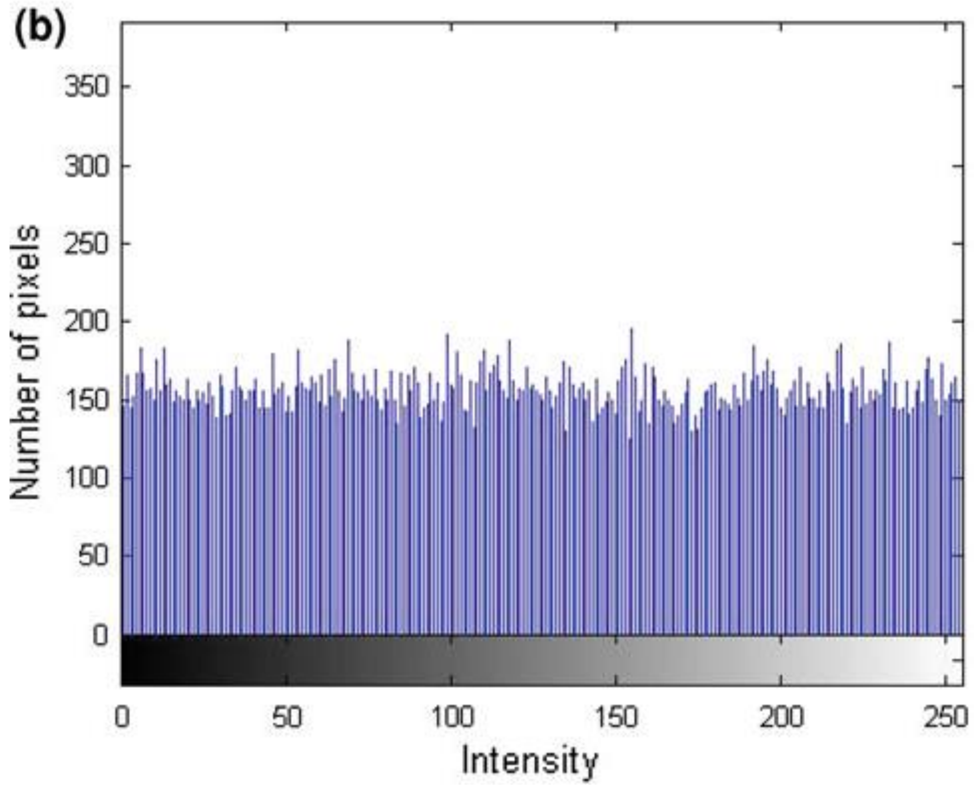


Fig. 6 a Histogram of source image and b histogram of ciphered image

4.1.2 Entropy

Histogram analysis is a visual test which shows the pixel distribution over the available intensity levels. To quantify the uniformity of image histograms, information entropy is used. It is generally known as Shannon entropy and is the most important feature of randomness [19–21]. It quantifies the amount of information contained in data, usually in bits or bits/symbol. The performance of the image encryption algorithms is measured by computing entropy of the source and ciphered images and then comparing them. The Entropy '*E*' of the image is calculated using Eq. (1):

$$E = \sum_{i=0}^{255} \left[P(i) * \log_2 \left(\frac{1}{P(i)} \right) \right], \qquad (1)$$

where *P(i)* represents the probability of symbol '*i* ', and the entropy is expressed in bits. For a purely random source emitting 2*n* symbols, the entropy should be equal to *n* [22].

For instance, consider a gray scale image, whose pixel data have 28 possible values, so the entropy of a "true random" image must be 8. If the output of an encryption technique emits symbols with entropy <8, there exists certain degree of predictability, which threatens its security. The values of entropies obtained for the source image, and the ciphered image is given in the Table 1. The entropy value for ciphered image is very close to the ideal value 8. This implies that the information leakage is negligible, and the proposed encryption technique is secure against statistical attacks.

Table 1 Entropy values for the source and ciphered image

Source Image	Ciphered Image
7.4250	7.9876

4.2 Correlation Analysis

It is a conventional task of an efficient image encryption algorithm to eliminate the correlation of pixels [8–10]. In this simulation, the correlation coefficient of 1,000 randomly selected pairs of vertically, horizontally and diagonally adjacent pixels is determined. Figure 7 displays the distribution of the randomly selected pairs of horizontally adjacent pixels in the source image and the ciphered image. The graphical result emphasizes that there is hardly any correlation between the pixels in the ciphered images. The correlation coefficient between two adjacent pixels in an image is determined using the formula described in (2), [23]:

$$R_{xy} = \frac{\sum_{i=1}^{N} (x_i - E(x))(y_i - E(y))}{\sqrt{\sum_{i=1}^{N} (x_i - E(x))^2}\sqrt{\sum_{i=1}^{N} (y_i - E(y))^2}}, \qquad (2)$$

where *E (x)* = mean *(xi)* and *x, y* are gray values of two adjacent pixels in the image. The correlation coefficients of the plain Baboon image and its corresponding cipher image in three directions are listed in Table 2. As expected, the two neighboring pixels in the source image are highly correlated in all three directions, while there is a negligible correlation between the two neighboring pixels in the ciphered image.

From the above analysis, we can conclude that the proposed encryption technique is secure against correlation attacks.

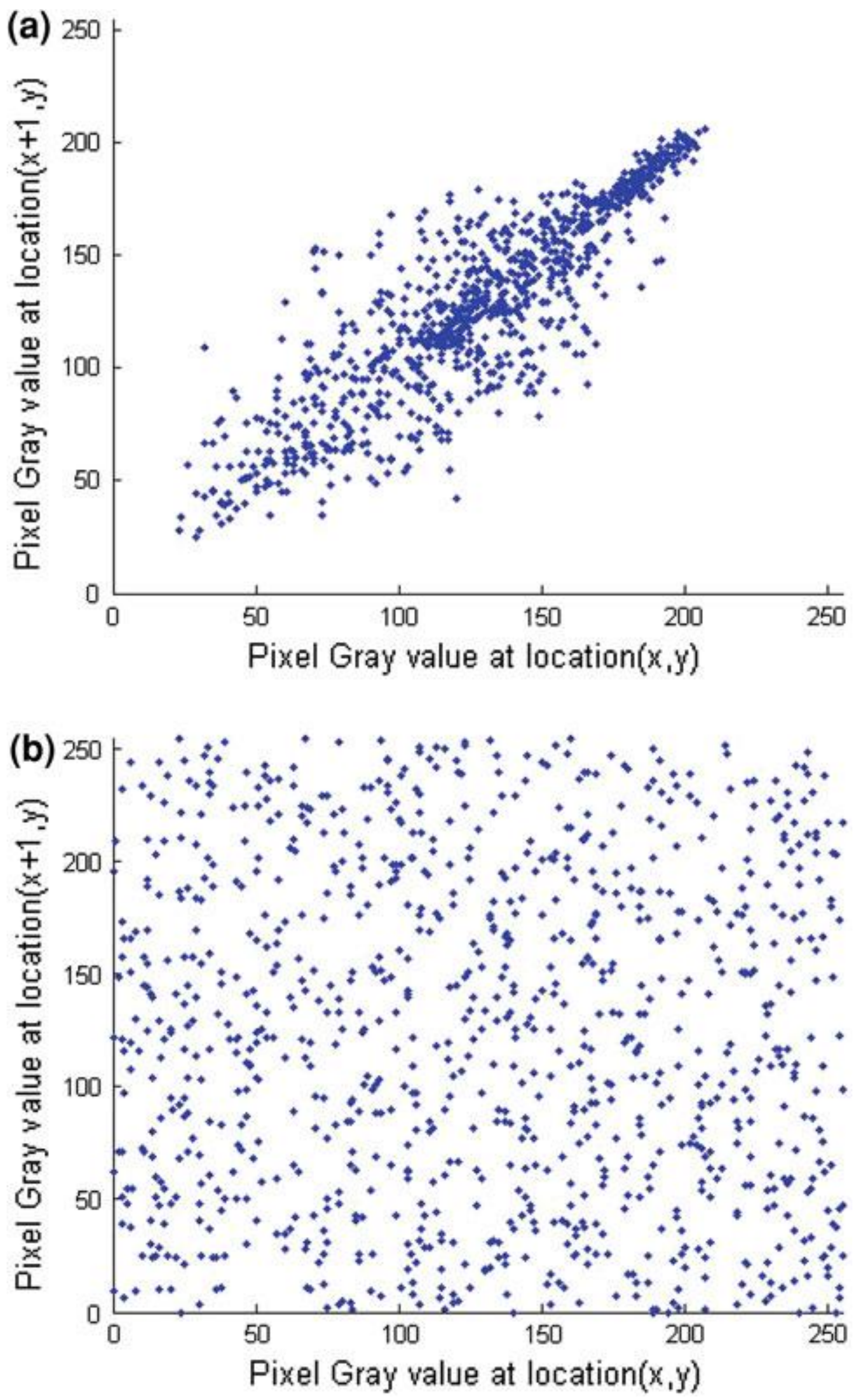


Fig. 7 Distribution of the randomly selected pairs of horizontally adjacent pixels in the a source image and b in the ciphered image

4.2.1 Key sensitivity analysis

An efficient encryption algorithm should be sensitive to secret key, that is, a small change in secret key during decryption process results into a completely different decrypted

Table 2 Correlation coefficients of the source image and the ciphered images in three directions:

Direction	Source Image	Ciphered Image
Horizontal	0.8851	0.00257
Vertical	0.8875	0.00686
Diagonal	0.8385	-0.00186

image [11,12]. The decrypted image using the proposed technique with the correct key is shown in Fig. 8a. The decrypted image using the decoder of the proposed technique with a slightly different initial condition (one bit difference) of the LFSR is shown in Fig. 8b. The decrypted image using the decoder of the proposed technique with the same initial condition for the LFSR but a slightly different starting pixel for the selected block $P(n + 1, k + 1)$ is shown in Fig. 8c. From the figures, it can be seen that the decoders failed to decrypt the images correctly in all cases except when the key was correct. Hence, the proposed technique is highly sensitive to the initial condition of the LFSR and the starting pixel $P(n, k)$.

To quantify the robustness of image cryptosystems against the key sensitivity, two most common measures NPCR (Number of pixels change rate) and UACI (unified average changing intensity) are used. The NPCR is used to measure the percentage number of pixels in difference in two cipher images obtained by applying slightly different secret keys on source images, and UACI is used
to measure the corresponding unified average changing intensity.
The UACI is defined by :

$$\text{UACI} = \frac{1}{M \times N}\left[\frac{\sum_{i,j} |C_1(i,j) - C_2(i,j)|}{255}\right] \times 100\,\% \quad (3)$$

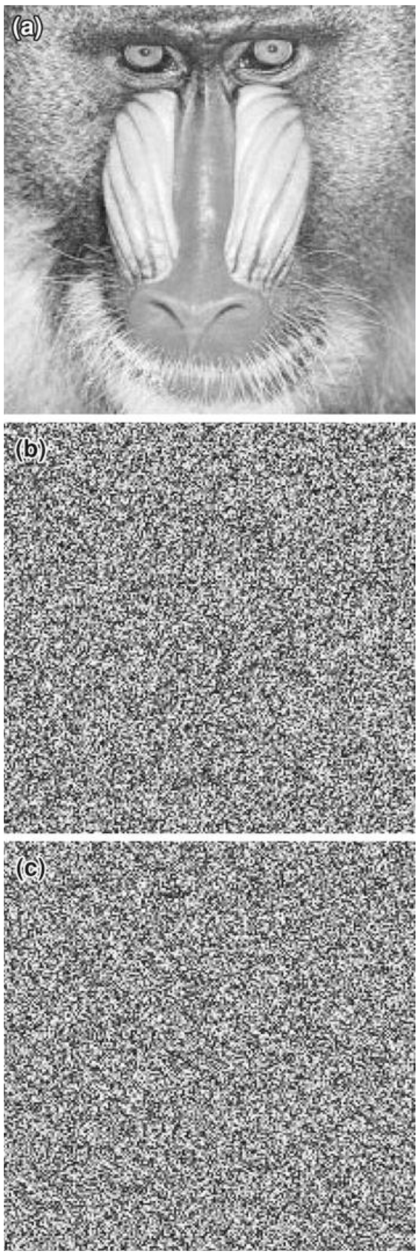


Fig.8a Decrypted image with correct key; b decrypted image using slightly different initial conditions of LFSR; c decrypted image using same initial conditions but with a different starting pixel.

The NPCR is defined by Eq. (4)

$$\mathrm{NPCR} = \frac{\sum_{i,j} D(i,j)}{M \times N} \times 100\,\%, \qquad (4)$$

where

$$D(i,j) = \begin{cases} 0 & \text{if } C_1(i,j) = C_{12}(i,j) \\ 1 & \text{if } C_1(i,j) \neq C_{12}(i,j) \end{cases}$$

The NPCR value for two random images, which is an expected estimate for an ideal image cryptosystem [13], is given by Eq. (5)

$$\mathrm{NPCR\ expected} = \left(1 - 2^{-L}\right) \times 100\,\% \qquad (5)$$

where L is the number of bits used to represent the different bit planes of an image. For a gray scale image, $L = 8$ (8 bits for each pixel) hence NPCR Expected =99.6094%. The UACI value for two random images, which is an expected estimate for an ideal image cryptosystem [12], is given by Eq. (6).

Table 3 NPCR and UACI values

Measures	Slight Change in Initial Condition	Slight Change in starting Pixel
NPCR	99.6275	99.6826
UACI	33.4701	33.5694

$$\mathrm{UACI} = \frac{1}{2^{2L}} \left[\frac{\sum_{i=1}^{2^L - 1} i\,(i+1)}{2^L - 1} \right] \times 100\,\%, \qquad (6)$$

For a gray scale image, $L = 8$ (8 bits per pixel), hence the expected UACI = 33.4635%.
The NPCR and UACI values for the proposed technique using a slightly different initial condition and a different starting pixel $P(n + 1, k + 1)$ are given in Table 3. Based on Table 3, it is clear that the proposed technique gave NPCR and UACI values that are very close to ideal. Another important conclusion from this simulation is that the sensitivity of the proposed technique to the starting pixel is comparable to the sensitivity of a change in the initial condition of the LFSR.

4.2.2 Key space analysis

The key space is the total number of different keys that can be used in the encryption. For secure image encryption, the key space should be large enough to make brute force attacks infeasible [14,15]. Theoretically, key space of about 2128 is large enough for the cryptosystem to resist exhaustive attacks [6,7,13,16,24,25]. The key space for the proposed technique for a set of public images K of size $M \times N$ is $K \times M \times N \times (2U-1)$, a flexiblemoderately large key space. The key space was calculated based on the fact that the proposed technique is sensitive to any change in the pixel location and the initial conditions of the LFSR which is used to address the selected block and it is quite clear that the key space will be sensitive to the selected public image. Using the right feedback, the content of the LFSR can only repeat after $2U-1$. Because the address $S$is hardwired to the content of the LFSR, it has the same randomness properties. The password of the user will be of size $log2\ (K \times M \times N) + U$. If a hundred bit LFSR is used the key space of about 2128 can be easily exceeded. This large key space is sufficient and is immune to all kinds of brute force attacks.

4.3 Computational complexity

The number of primitive instructions in an algorithm can be used as a measure of its computational complexity. This is more objective as execution time of the same algorithm varies a lot with the programming language, the programmer and the way to optimize the program [26]. Primitive instructions such as accessing memory and simple XOR operations can typically be executed within one instruction cycle on most digital devices. In contrast, non-primitive instructions like multiplication/division operations and trigonometric functions require more time than primitive ones. For instance, it takes 64 addition and shift operations to compute the production of two double-precision numbers [26]. We compare the computational complexity of our algorithm to four typical image encryption algorithms. The algorithm proposed in Behnia et al. [27] requires too many multiplication operations and trigonometric functions to encrypt each pixel. The algorithm proposed in Akhshani et al. [16] and Wei Zhang et al. [18] requires too many primitive operations to encrypt each pixel. The algorithm proposed in Zhou and Liao [26] can be implemented with only three primitive operations to encrypt each pixel but requires 128 rounds to achieve good results.

Our encryption/decryption algorithm is divided into two stages. In the initialization stage, the content of the selected block *B* is moved to the cache. This initialization stage will take a certain number of instruction cycles *I* , but will be done only once. In the processing stage, the address S is updated in parallel with memory access and Xoring operation.
Note that because the content of *B* is guaranteed to be in the cache, there will be no page faults while accessing memory.
Hence, each memory access is considered a primitive operation. For every state of the LFSR, the address *S* gets updated. Simultaneously, using *S*, a random bit is obtained from the selected block of the parity image and is Xored with the image bit. The slowest step in an algorithm defines the speed of the algorithm. In our algorithm, the slowest step is obtaining the parity bit and performing the XOR operation.
These two primitive operations are required for each bit in a pixel. Therefore, the number of operations per pixel for a gray scale image is 16.

Table 4 Comparison of four image encryption algorithms on computational efficiency

Algotithm	Akhshani et al	Behnia et al	Zhang and Liu	Zhou and Liao	Ours
No. of primitive instruction	327	1,358	1,421	3	16
No. of Pixel	65,536	65,536	65,536	65,536	65,536
No. of Rounds	2	2	2	128	1
Total No. of Instruction	42,860,544	177,195,776	186,253,312	25,165,824	1,048,576
No. of Inst. Cycle	42,860	177,195,776	186,253,312	25,165,824	*I* +1,048,576

Table 4 compares the computational efficiencies of different algorithms measured by the number of primitive instructions required to encrypt a 256 × 256 image. The following table shows that our algorithm is faster than the other algorithms.

CHAPTER 5

LIMITATIONS AND FUTURE SCOPE

5. LIMITATIONS AND FUTURE SCOPE

5.1 Conclusion

This paper introduced a newstream encryption technique that utilizes a selected block from the parity bit plane of a public image to confuse the relation between ciphered bits and key stream. The NPCR and UACI values were used to compare to ciphered images with slightly different key. Both values were close to ideal in two different cases and suggested that the algorithm is highly sensitive to a change in the initial condition of the pseudo-random generator and the starting pixel of the selected block from the parity bit plane. Future work includes implementing this algorithm in hardware to test its speed.

5.2 Limitations

The images should be selected carefully so that they do not include smooth regions, such as, clear blue sky or a uniform background. For example, the famous Lena image will be perfectly suitable for this technique.
The image should be taken of small size as it will give a fast output. Images having high size will apparently take more time as there is a XOR of every pixel consisting of 24 bits.

5.3 Future Scope

The future of encryption is brighter than ever before. The demand for more control and protection of corporation information assets and third-party information is increasing dramatically. The amount of information being communicated and stored electronically is vastly greater than even five years ago. As a result, the need for more effective information security products is growing at a higher rate than any other aspect of IT technology within the enterprise Today.

5.4 PROGRAMMING ENVIRONMENT & TOOLS USED:

Following programming languages/tools will be used for this purpose: Java

REFERENCES:

1. Li, S., Chen, G., Zheng, X., in: Furht, B., Kirovski, D. (eds.) Multimedia Security Hand-book, Chapter 4, p. 133. CRC Press (2005)
2. Shannon, C.E.: Communication theory of secrecy systems. Bell Syst. Tech. J. **28**, 656–715 (1949)
3. Robshaw, M.J.B.: RSA Laboratories Technical, Report, TR-701(2) (1995)
4. Menezes, A.J., Van Oorschot, P.C., Vanstone, S.: Handbook of Applied Cryptography. CRC Press, Boca Raton, FL, USA (1996)
5. Goldreich, O.,Goldwasser, S., Micali, S.: Howto construct random functions. J. Assoc. Comput. Mach. **33**(4), 792–807 (1986)
6. Wang, X., Zhao, J., Liu, H.: A new image encryption algorithm based on chaos. Opt. Commun. **285**(5), 562–566 (2012)
7. Zhu, C.: A novel image encryption scheme based on improved hyperchaotic sequences. Opt. Commun. **285**(1), 29–37 (2012)
8. Lian, S.: Efficient image or video encryption based on improved hyperchaotic sequences. Chaos, Solitons, Fractals **40**(5), 2509–2519 (2009)
9. Gao, H., Zhang, Y., Liang, S., Li, D.: A new chaotic algorithm for image encryption. Chaos, Solitons, Fractals **29**(2), 393–399 (2006)
10. Gao, T., Chen, Z.: Image encryption based on a new total shuffling algorithm. Chaos, Solut Fractals **38**(1), 213–220 (2008)
11. Amin, M., Faragallah, O.S., Abd El-Latif, A.A.: A chaotic block cipher algorithm for image cryptosystems. Commun. Nonlinear Sci. Numer. Simul. **15**(11), 3484–3497 (2010)
12. Abd El-Latif, A.A., Niu, X., Wang, N.: Chaotic image encryption using Bezier curve inDCT domain scrambling. Commun. Comput. Inform. Sci. **194**, 30–41 (2011)
13. Patidar, V., Pareek, N.K., Purohit, G., Sud, K.K.: A robust and secure chaotic standard map based on pseudorandom permutationsubstitution scheme for image encryption. Opt. Commun. **284**, 4331–4339 (2011)
14. Amin, M., Abd El-Latif, A.A.: Efficient modified RC5 based on chaos adapted to image encryption. J. Electron. Imaging **19**(1) (2010)
15. Chen, G., Mao, Y., Chui, C.K.: A symmetric image encryption scheme based on 3D chaotic cat maps. Chaos, Solitions Fractals **21**(3), 749–761 (2004)
16. Akhshani, A., Behnia, S., Akhavan, A., Abu Hassan, H., Hassan, Z.: A novel scheme for image encryption based on 2D piecewise

chaotic maps. Opt. Commun. **283**(17), 3259–3266 (2010)
17. Yoon, J.W., Kim, H.: An image encryption scheme with a pseudorandom permutation based on chaotic maps. Commun. Nonlinear Sci. Numer. Simul. **15**, 3998–4007 (2010)
18. Zhang,W.,Wong, K.,Yu, H., Zhu, Z.:An image encryption scheme using lightweight bit-level confusion and cascade cross circular diffusion. Opt. Commun. **285**, 2343–2354 (2012)
19. Chen, G., Mao, Y., Chui, C.K.: A symmetric image encryption scheme based on 3D chaotic cat maps. Chaos, Solitons Fractals **21**(3), 749–761 (2004)
20. De Santis,A.F., Ferrara, A.L.,Masucci, B.: Unconditionally secure key assignment schemes. Disc. Appl. Math. **154**, 234–252 (2006)
21. Li, W.: On the relationship between complexity and entropy for Markov chains and regular languages. Complex Syst. **5**, 381–399 (1991)
22. Zhang, G., Liu, Q.: A novel image encryption method based on total shuffling scheme. Opt. Commun. **284**, 2775–2780 (2012)
23. Bluman, A.G.: Elementary Statistics: A Step by Step Approach. McGraw-Hill, Boston (1997)
24. Stinson, D.: Cryptography: Theory and Practice, 2nd edn. CRC/C&H (2002)
25. Schneier, B.: Applied Cryptography: Protocols, Algorithms and Source Code in C, 2nd edn. Wiley, New Jersey (1996)
26. Zhou, Q., Liao, X.: Collision-based flexible image encryption algorithm. J. Syst. Softw. **85**, 400–407 (2012)
27. Behnia, S., Akshani, A.,Mahmodi, H., Akhavan, A.: A novel algorithm for image encryption based on mixture of chaotic maps. Chaos, Solitions Fractals **35**, 408–419 (2008)