A Thesis
entitled
Development of Data Encryption Algorithms for Secure Communication
Using Public Images
by
Vishwanath Ullagaddi
Submitted to the Graduate Faculty as partial fulfillment of the requirements for the
Master of Science Degree in Electrical Engineering

Dr. Vijay Devabhaktuni, Committee Chair

Dr. Firas Hassan, Co-Committee Chair

Dr. Ezzatollah Salari, Committee Member

Dr. Richard Molyet, Committee Member

Dr. Patricia R. Komuniecki, Dean
College of Graduate Studies

The University of Toledo
August 2012
An Abstract of

Development of Data Encryption Algorithms for Secure Communication Using Public Images

by

Vishwanath Ullagaddi

Submitted to the Graduate Faculty as partial fulfillment of the requirements for the Master of Science Degree in Electrical Engineering

The University of Toledo
August 2012

In the information age, sharing and transfer of data has increased tremendously. Usually, the information exchange is done using open channels which can make it vulnerable to interception. The threat of intruder accessing secret information has been a continuing concern for data communication experts. One of the ways of protecting the data is by use of images.

This research tries to explore the use of public images for secure communication. Novel algorithms were developed for data hiding and data encryption using public images. Initially a method for hiding classified data in images based on a password is introduced, which is further improved into a data encryption technique. Further, two more techniques were proposed for data encryption using the parity block of the public image. Simulation results showed that ciphered gray 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 proposed stream encryption techniques are also highly sensitive to the starting pixel of the block. Both the encryption techniques have a simple structure and can generate a key stream faster than other generators.
To my family for their love, endless support, and encouragement
Acknowledgements

I wish to express my deepest gratitude to my advisor Dr. Vijay Devabhaktuni and co-advisor Dr. Firas Hassan for their continued support, guidance and encouragement, without which it would not have been possible to succeed in my research. The financial support from the EECS Department in the form of a graduate/research assistantship is gratefully acknowledged. I would like to thank my lab mates Deepak Bhatt, Bertrand Fernandes, Lakshman Srinivas Mareddy, Karpagam Manivannan and Sreenadh Depuru for helping me in my research study.

Special thanks to all my roommates and friends for making my stay in Toledo, a memorable one. Above all, I wish to express my deep appreciation to my family and friends who with their love and encouragement have made all this possible.
# Contents

Abstract iii  
Acknowledgements v  
Contents vi  
List of Tables x  
List of Figures ix  
List of Abbreviations xiii  

1. Introduction 1  
1.1 Problem Statement ................................................................. 1  
1.2 Proposed Research Approach.................................................... 2  
1.3 Thesis Organization........................................................................ 3  

2. Literature 5  
2.1 Introduction .......................................................................................... 5  
2.2 Digital Steganography............................................................................ 5  
2.3 Digital watermarking............................................................................. 9  
2.4 Data Encryption..................................................................................... 11  
2.4.1 Cryptosystems .................................................................................. 11  
2.4.2 Stream Encryption .......................................................................... 12  
2.4.3 Short-comings of Stream Encryption ............................................ 14
2.5 Image Encryption Techniques ........................................................... 15
2.6 Summary ............................................................................................ 16

3. A New Passcode Based Approach for Hiding Classified Information in Images

3.1 Introduction ....................................................................................... 18
3.2 Matching Process and Generation of 2BC .......................................... 19
3.3 The Three Different Suggested Techniques for Embedding the 2BC ....... 20
3.4 Data Retrieval ................................................................................... 23
3.5 Simulation Results ............................................................................ 25
3.6 Summary ........................................................................................... 27

4. A Robust Data Encryption Technique Using Matching Principle

4.1 Introduction ....................................................................................... 29
4.2 Proposed Hiding Algorithm ............................................................... 29
  4.2.1 Embedding Data ........................................................................ 30
  4.2.2 Pattern of 2-bit code present in higher nibble .......................... 30
4.3 Data Retrieval ................................................................................... 34
4.4 Illustrative Examples ....................................................................... 35
4.5 Results ............................................................................................... 37
4.6 Summary ........................................................................................... 38

5. Symmetric Self-Synchronous Stream Encryption Using Images

5.1 Introduction ....................................................................................... 39
5.2 Proposed Encryption Technique ........................................................ 40
5.3 Simulation Results and Discussion .............................................................. 43
  5.3.1 Key Space Analysis ........................................................................... 44
  5.3.2 Statistical Analysis ......................................................................... 44
  5.3.3 Key Sensitivity Analysis .................................................................. 52
5.4 Conclusion .................................................................................................... 54

  6.1 Introduction ............................................................................................ 55
  6.2 Proposed Encryption Technique ............................................................ 56
  6.3 Simulation Results and Discussion .......................................................... 57
    6.3.1 Statistical Analysis ....................................................................... 59
    6.3.2 Key Sensitivity Analysis ............................................................... 66
    6.3.3 Key Space Analysis ...................................................................... 68
  6.4 Conclusion .................................................................................................... 68

7. Conclusions and Future Work 69
  7.1 Conclusions ............................................................................................ 69
  7.2 Future Work ............................................................................................. 70

References 72

Appendix-A 81
  A.1 Source Code of Symmetric Synchronous Stream Encryption Technique... 81
List of Tables

Table 3.1: PSNR values in dB using technique 1 ............................................................. 27
Table 3.2: PSNR values in dB using technique 2 ............................................................. 27
Table 3.3: PSNR values in dB using technique 3 ............................................................. 27
Table 3.4: PSNR values in dB for the Lena image using the three proposed techniques 27
Table 4.1: Code words for different matching positions in pixel..................................... 32
Table 5.1: Entropy values of plain image and cipher image............................................. 47
Table 5.2: Correlation Coefficients in all the three directions ........................................ 48
Table 5.3: NPCR and UACI values for the proposed technique ...................................... 54
Table 6.1: Entropy values for the source and ciphered image ........................................ 60
Table 6.2: Correlation Coefficients of the source image and the ciphered images in three directions........................................................... 62
Table 6.3: NPCR and UACI values for the two techniques ............................................. 67
List of Figures

Figure 2-1: General Steganographic Model. ................................................................. 6
Figure 2-2: Properties of a Good Steganographic Technique. ................................. 7
Figure 2-3: Block Diagram of Stream Encryption. ..................................................... 13
Figure 2-4: Block Diagram of Self-Synchronous Stream Cipher. .............................. 13
Figure 3-1: Representation of the image pixel ‘M’ used for matching. .................... 19
Figure 3-2: Representation of the image pixel ‘E’ used for embedding the 2BC based on password. ........................................................................................................ 21
Figure 3-3: Image Bit Error Effects. ......................................................................... 22
Figure 3-4: Flowchart for embedding the data. .......................................................... 23
Figure 3-6 Lena image using the second proposed technique (a) before embedding, and (b) after embedding................................................................. 26
Figure 3-7 Baboon image using the third proposed technique (a) before embedding, and (b) after embedding................................................................. 26
Figure 4-1: Representation of image pixel................................................................. 30
Figure 4-2: Representation of 2-bit code................................................................. 30
Figure 4-3: Flowchart for encoding the data ............................................................. 33
Figure 4-4: Flowchart for decoding the data ............................................................ 34
Figure 5-1: (a) Lena image as an example of public images (b) Corresponding parity bit plane of the Lena image......................................................... 41
Figure 5-2: Block diagram of the encoder. ................................................................. 42
Figure 5-3: Block diagram of the decoder. ............................................................... 42
Figure 5-5: Histogram of Plain image..................................................................... 46
Figure 5-6: Histogram of Cipher Image................................................................. 46
Figure 5-7: Distribution of the randomly selected pairs of horizontally adjacent pixels in the source image. ................................................................. 49
Figure 5-8: Distribution of the randomly selected pairs of horizontally adjacent pixels in the cipher image................................................................. 49
Figure 5-9: Distribution of the randomly selected pairs of vertically adjacent pixels in the source image. ................................................................. 50
Figure 5-10: Distribution of the randomly selected pairs of vertically adjacent pixels in the cipher image................................................................. 50
Figure 5-11: Distribution of the randomly selected pairs of diagonally adjacent pixels in the source image. ................................................................. 51
Figure 5-12: Distribution of the randomly selected pairs of diagonally adjacent pixels in the cipher image................................................................. 51
Figure 5-13 (a) Decrypted image with correct key using the proposed technique; (b) decrypted image using the proposed technique with a slightly different starting pixel. ................................................................. 52
Figure 6-1: Block diagram of the encoder for the proposed encryption technique....... 56
Figure 6-2: Block diagram of the decoder for the proposed encryption technique....... 57
Figure 6-3: (a) Source image; (b) Ciphered image using the proposed technique....... 58
Figure 6-4: Histogram of source image................................................................. 61
Figure 6-5: Histogram of ciphered image. ................................................................. 61

Figure 6-6 Distribution of the randomly selected pairs of horizontally adjacent pixels in
the source image ........................................................................................................ 63

Figure 6-7 Distribution of the randomly selected pairs of horizontally adjacent pixels in
the cipher image ........................................................................................................ 63

Figure 6-8: Distribution of the randomly selected pairs of vertically adjacent pixels in the
source image. ............................................................................................................. 64

Figure 6-9: Distribution of the randomly selected pairs of vertically adjacent pixels in the
cipher image ............................................................................................................. 64

Figure 6-10: Distribution of the randomly selected pairs of diagonally adjacent pixels in
the source image. ..................................................................................................... 65

Figure 6-11: Distribution of the randomly selected pairs of diagonally adjacent pixels in
the cipher image ..................................................................................................... 65

Figure 6-12 (a) Decrypted image with correct key using the proposed technique; (b)
decrypted image using the proposed technique with a slightly different initial condition; (c) decrypted image using the proposed technique with the same initial condition but a different starting pixel ........................................ 67
List of Abbreviations

2BC................ Two Bit Code
DW ................ Digital Watermarking
HVS.............. Human Visual system
LFSR ............ Linear Feedback Shift Register
LSB............... Least Significant Bit
MSB............. Most Significant Bit
MSE.............. Mean Square Error
OTP............... One Time Pad
PRBG ............ Pseudo Random Bit Generator
PSNR............. Peak Signal to Noise ratio
SG............... Steganography
Chapter 1

Introduction

1.1 Problem Statement

In the information age, sharing and transfer of data has increased tremendously and usually the information exchange is done using open channels which can make it vulnerable to interception. The threat of an intruder accessing secret information has been an ever existing concern for the data communication experts [1]. Steganography and cryptography are the two important tools to protect information.

Steganography is the art of hiding data in the images. Usually, the data is hidden in the least significant bits of the pixels. But, the LSB steganography is not a robust data hiding technique as it is highly susceptible to cropping, rotation, low bit reverse attack due to LSB weakness and other geometrical attacks and compression techniques. Moreover, an adversary can also erase the message by zeroing the LSB plane, as the perceptual quality if the image doesn’t get affected with the least significant bits.

Data encryption is another way of protecting the data. Usually stream encryption is employed to protect the data. In stream encryption each plaintext bit is encrypted one at a time with the corresponding bit of the key stream, to give a bit of the cipher text stream.
The pseudo random key stream is generated sequentially from a random seed value using shift registers. However, Stream ciphers are highly susceptible to known-plaintext attacks.

1.2 Proposed Research Approach

In the first phase of the thesis, a new data hiding technique using the least significant bits (LSB) of the cover image pixels is introduced. The proposed data hiding technique makes use of a password and a two bit code associated with the matching bit of the pixel, which is discussed in chapter 1. In the proposed technique the least significant bits of the image were highly vulnerable to noise and cropping effects. Hence, to improve this, we used the most significant bits of the pixel in our matching technique which is discussed in chapter 2. The matching technique can be used for both data hiding as well as data encryption. It can be used as a potential data hiding technique without changing a single pixel in the cover image, but to decrypt the message, we needed the encoded information which is three times the length of the message. The technique could also be used as a data encrypting technique although, it still holds the drawback discussed above. But the security of the matching technique is very high, as it is highly sensitive to the key.

In the second phase of thesis, we tried to reduce these three bits into single bit by randomly accessing a single bit from the image and performing xor operation between the message bit and the image bit. In this phase, we introduced two data encryption techniques using conventional stream encryption concepts. The proposed data encryption techniques can be used for any kind of data such as text and multimedia, but to evaluate
the techniques, simulation results are shown using images. The suggested encryption techniques use traditional synchronous and self-synchronous stream encryption systems. Both the techniques use a block of parity bit plane of the public image to encrypt the data. The selection of block dictates the sensitivity of the algorithm. A minute change in the selection of the starting pixel of the block, results into a completely different input message. In the self-synchronous system, simple shift registers are used to access the parity bits randomly and are used to perform xor operation between the message bits and parity bits to obtain the ciphered bits. These ciphered bits are again fed back to the most significant bit of the shift register and all the remaining bits are encrypted in the similar fashion. In synchronous system, linear feedback shift registers are used to access the parity bits. In self-synchronous systems, the key doesn’t repeat but in synchronous system the key repeats after the period of LFSR.

1.3 Thesis Organization

The rest of the thesis is organized as follows:

Chapter 2 provides literature review on steganography and data encryption techniques.

Chapter 3 introduces a novel technique for hiding classified information in images. The proposed technique uses the least significant bits of the image to hide the data based on a password. Results are shown by calculating the PSNR values between the cover image and stego-image.

Chapter 4 presents a data encryption method using a data matching technique and a
password. The sensitivity of the technique is discussed by slightly changing the password.

Chapter 5 presents an image encryption technique using a parity block of a public image as a non-linear filtering function and a self-synchronous key generator. The results are shown in terms of statistical, key space and key sensitivity analysis.

Chapter 6 presents another image encryption technique using a synchronous stream encryption system and using the contents of linear feedback shift register (LFSR) to randomly access parity bits of a public image.

Finally, Chapter 7 gives the conclusions of the work done and future work.
Chapter 2

Literature

2.1 Introduction

Many digital services require reliable security in storage and transmission of digital data. Due to the rapid growth of the internet in the digital world today, the security of information has become more important and attracted much attention. Steganography (SG) is one of many techniques used to protect the digital data. It is a technique in which communication between two parties is done in a covert fashion using a cover object. SG is a very old practice for secret communication and can be traced back to techniques like invisible ink and microdots used by spies [2].

2.2 Digital Steganography

Image steganography is a main branch of information hiding techniques, which can be used for hiding private communication through public channels. Recently, information hiding techniques have attracted considerable research interest in the field of information
security [3]. In general, the embedding operation in SG requires a digital medium to carry the data. Images and multimedia components, such as video and audio files, are widely used and exchanged through the internet. Such mediums are the best cover media to hide messages as they do not attract any attention.

Different steganographic techniques employ different types of digital images such as binary, grayscale and color images. It is not necessary that the cover and the message have homogeneous structure. For example, it is possible to embed a recording of an audio stream message inside a digital image [4]. Data hiding requires embedding data into digital media like image, audio, or text, however, due to providing high embedding efficiency the still images are preferred as hosts as compared to others. Digital images are the most widespread cover files used for SG due to the insensitivity of the human visual system (HVS) [5].

During the embedding stage, a key is used to insert a message in a cover medium resulting in a stego-object as shown in Figure 2-1. The stego-object is then transmitted along public channels to its destination. When the stego-object is received, the embedded message is extracted from stego-object using the known stego-key.

![Figure 2-1: General Steganographic Model.](image)

Steganography has become an interesting and challenging field of research striving to achieve greater immunity of hidden data against signal processing operations on the host
cover media; e.g. a good SG technique should offer immunity of hidden data against lossy compression, scaling, interception, modification, or removal etc. and ensure that embedded data remains inviolate and recoverable [6]. However, a trade-off between the quantity of hidden data and its degree of immunity to host signal modification is needed in most cases [7].

![Diagram of Good Steganographic Technique]

**Figure 2-2: Properties of a Good Steganographic Technique.**

The basic requirements of a good steganographic technique are as shown in Figure 2-2. In order to evaluate the efficiency of image steganography systems, two fundamental characteristics must be investigated. These are the security and capacity of the approach [8]. Designing algorithms that are undetectable and also yield a large capacity is the main goal in steganography [9].

Image steganography systems can be considered secure if it is very difficult for attackers to detect the presence of a hidden message in the stego-image by using any accessible means. The hidden message must be invisible both perceptually and statistically in order to avoid any suspicions of attackers. However, a SG system fails if
an attacker is able to prove the existence of a secret message [5]. Further, these algorithms cannot deal with the opponents who not only detect a message, but also make it useless.

Data hiding in still images introduces a variety of challenges that occur due to the way the human visual system (HVS) works and the usual changes that images undergo. It is practical to expect that still images will be subject to operations ranging from simple affine transforms to nonlinear transforms such as cropping, blurring, filtering, and lossy compression. Thus, data-hiding techniques need to be resistant to as many of these transformations as possible. Despite these challenges, still images are potential candidates for data hiding. A major drawback of HVS is our relative insensitivity to very low spatial frequencies such as continuous changes in brightness across an image, i.e., vignetting [12].

Steganographic capacity is defined as the maximum number of bits that can be embedded in a given cover image with very little probability of detection by a hacker. SG systems used for secret communication, aim to maximize the steganographic capacity and minimize the perception of hidden messages in stego images [8]. However, Steganographic capacity and imperceptibility are at odds with each other, as hiding larger amounts of information introduces more artifacts into stego-images and therefore increasing the possible perceptibility of the hidden information [10, 11]. Hence, to satisfy high steganographic capacity and good imperceptibility, they must achieve a balance between these two requirements.

Uses of Steganography: Steganography can be used to hide data and to prevent unauthorized access to people from becoming aware of the existence of a message. In the
business world steganography can be used to hide a secret invention or plan of action. Steganography can also be used for corporate surveillance by sending out trade secrets without anyone noticing in the company. Steganography can also be used in non-commercial sectors to hide the presence of information from people. Spies have used steganography by making use of invisible ink, since the time of the Greeks to pass messages undetected [13].

*Detection:* Although the perceptual quality of the stego-images is very high, they usually leave behind traces of some type of statistical hint that they have been modified. It is these inconsistencies which an analysis tool may be able to detect. Statistical Analysis of an image can be performed to check if there is any hidden message inside it. The easiest way is to measure the entropy of redundant data and ensure if its statistical properties have moved away from the data collected from the cover image. The entropy test, itself is not a proof of a secret message but serves as the starting test to detect the existence of the message. After a suspected image is found then a brute-force attack must be performed to confirm the existence of a hidden message [13].

### 2.3 Digital watermarking

While the goal of steganography is to transmit a message undetected, a digital watermark is created as a sign of ownership/authorship. Since digital copies are inherently exact replicas of the original unless noise or some type of lossy operation is performed, there will be no way to tell them apart. Digital watermarks are used to show some proof of ownership by having your mark put into the file, so even if both images are
the same, if they contain your mark then you have a much stronger case for copyright or ownership disputes. Watermarks can be visible or invisible depending on the luminance in the mask. If the luminance is higher, the chances of visibility of the watermark are higher. Hackers can use different types of image processing tools to remove or degrade the watermark until it is indecipherable. There are various recovery techniques but it is usually helpful to have access to the original image when trying to rebuild the watermark.

Data hiding in images is useful in a variety of applications. The following are the three applications. [12]

**Digital watermark:** The objective of a digital watermark (DW) is to place an ineffaceable mark on an image. DW is a technology of embedding watermark with intellectual property rights into images, videos, audios and other multimedia data. The watermark may contain the author and user’s information, which could be the owner’s logo, serial and other multimedia data.

**Feature tagging:** Another important application of data hiding is tagging the location of features within an image. Using data hiding it is possible for a machine to encode descriptive information, such as the location and identification of features of interest, directly into specific regions of an image. This enables recovery of the descriptive information wherever the image goes. It can be understood that, since feature location is providing a service, it is not likely that someone will try to remove the encoded information.

**Embedded captions:** Usually, news photograph captions contain one kb of data. Thus embedded caption is a relatively high bit-rate application for data hiding. As with feature tagging, caption data are generally not subject to malicious removal. While captions are
useful by themselves, they become even more useful when united with feature location.

2.4 Data Encryption

Confidentiality of information is an essential necessity in the information era, and encryption is one of its protecting tools. Encryption is a method for protecting information from undesirable attacks by converting it into a form, unrecognizable by its attackers. The goal of encryption is to provide an easy and inexpensive means of encryption and decryption to all authorized users in possession of the appropriate key and vice versa to all other users without use of the key. The reverse of data encryption is data decryption, which recovers the original data.

Depending on the type of plaintext, data encryption systems are classified as text encryption, audio encryption, image encryption and video encryption. In order to have a generic cryptosystem that can encrypt digital data, such as text/image/audio/video, some encryption standards have been developed. Among them, DES, RSA, AES and IDEA are widely adopted. An overview of developments in the design of traditional cryptographic algorithms is given in [14], [15].

2.4.1 Cryptosystems

Cryptosystems are classified into symmetric and asymmetric cryptosystems based on the type of the key, where the former ones use secret key and the later ones use public keys. 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 [16]. Security of stream cipher depends on the generation of unpredictable sequence called key stream that must be of sufficient size and randomness [17]. Hence key stream generator is incredibly a vital building block for stream cipher algorithms.

2.4.2 Stream Encryption

Stream ciphers are an important class of symmetric encryption algorithms. 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[14]. 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 basic form of a stream cipher involves the generation of a pseudorandom sequence of bits that is XOR’ed bit by bit with the plaintext to generate the cipher text at the transmitter. At the receiver, the plain text is recovered by generating the identical pseudorandom sequence of bits such that it is exactly synchronized with the received cipher text stream.

Stream encryption systems are categorized into synchronous and self-synchronous. In the former, the key stream is generated independently of the message, so that a lost character during transmission necessitates a resynchronization of the transmission and receiver key
generators. The block diagram of synchronous stream encryption is as shown in Figure 2-3. The starting state of the key generator is initialized with a known input, $I_0$. The cipher text is obtained by modulo addition of the $i^{th}$ message character, $m_i$. Such synchronous ciphers are generally designed to utilize confusion but not diffusion. That is, the encryption of a character is not diffused over some block length of message. For this reason, Synchronous stream ciphers do not exhibit error propagation.

Figure 2-3: Block Diagram of Stream Encryption.

In a self-synchronous stream cipher, each key character is derived from a fixed number, $n$, of the preceding cipher text characters, giving rise to the name cipher feedback. In such a system, if a cipher text character is lost during transmission, the error propagates forward for $n$ characters, but the system resynchronizes itself after $n$ correct cipher text characters are received. The block diagram of synchronous stream encryption is as shown in Figure 2-4

Figure 2-4: Block Diagram of Self-Synchronous Stream Cipher.
The pseudo-random numbers can be generated either by using hardware or software. The simplest way of generating the pseudo-random number sequence using hardware is by making use of linear feedback shift registers, which can be used in stream ciphers, and are well suited to low power or high speed requirements. Using software, RC4 is one of the ways of generating the pseudo random number sequence. As with any stream cipher, these can be used for encryption by combining it with the plain text using bit-wise exclusive-or operation. These bits are similar to the Vernam cipher except that generated pseudo random bits, rather than prepared stream, are used.

2.4.3 Short-comings of Stream Encryption

However, pseudo random bit streams produced by any of the above methods are not highly efficient for encryption as they are vulnerable to some kind of attacks. For example, pseudo-random sequence produced by using LFSR’s, is very vulnerable to known plain text attacks, due to the linear combination of PRBG bits and the ciphered bits. Even RC4, although remarkable for its simplicity and speed in software, has weaknesses that argue against its use in cryptosystems. RC4 is highly vulnerable when beginning of the output stream is not discarded, or when nonrandom or related keys are used. To overcome the drawbacks of this linear combination of the PRBG and the ciphered bits, one of the solutions is to make the combination non-linear either by permutation or by confusion or by giving cipher/plain text feedback. To make the output non-linear, we can also make use of non-linear functions or chaotic functions between the PRBG and ciphered bits. Researchers have noticed that there exists a close relationship
between chaos and cryptography [18, 19]; as many properties of chaotic systems have their corresponding counterparts in traditional cryptosystems. Recently there have been many papers on chaotic encryption scheme [21-30]. Chaotic system has characteristics such as ergodicity, high sensitivity to initial conditions, long periodicity, high randomness and mixing [31-41]. With all these advantages, scientists were supposed to introduce new and powerful tools of chaotic cryptography [20]. Because of such common properties, chaotic systems have attracted much attention for secure communication and cryptography. Most properties meet some requirements such as diffusion and mixing in the sense of cryptography. Many efforts have been made to investigate chaotic image encryption schemes in order to promote communication security [42-47].

In a number of chaotic cryptosystems that have been proposed [48, 49 and 41], the chaotic pseudorandom bit generators (PRBGs) 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 cryptosystems very weak [50-51]; those chaotic PRBGs are hard to be discretized in the finite digitized state space or to be implemented with low complexity digital hardware requirement.

2.5 Image Encryption Techniques

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 [52]. In [53], Shannon suggested that confusion and diffusion are the two basic techniques to overcome high redundancies and strong correlations.

There have been many suggested image encryption techniques that use chaotic functions. In a number of chaotic cryptosystems that have been proposed [54-59], the chaotic pseudorandom 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 cryptosystems very weak.

### 2.6 Summary

Although, there are many data hiding and data encryption techniques present in literature, there is still a lot of scope for improving them. In this thesis, some of the ways to overcome the above drawbacks is suggested. In stream encryption, we try to confuse the relation between the pseudo-random sequence and the message bits. In this thesis, we introduce two new data encryption technique using images and traditional stream cipher concepts. Data security is the main aim of any data hiding algorithm. Our data hiding
algorithm meets all the requirements such as secrecy, integrity, availability and authenticity. The proposed technique, suggests a new way of encrypting data, which is based on confusing the relation between the output of PRBG and the message bits. In this proposed, data encryption technique, we will use the output of the PRBG’s as a pointer to a single bit in an image. A logical operator such as XOR’ing is applied between the message bits and the image bits to obtain the ciphered bits. The intrinsic properties of the images, such as bulk data capacity, high redundancy and high correlation among pixels are used to improve the security of the algorithm.
Chapter 3

A New Passcode Based Approach for Hiding Classified Information in Images

This chapter proposes a method of hiding classified data in images based on three different levels of security. Instead of hiding data directly, pixels are selected randomly and their higher nibble bits are matched with the data bits. A two bit code (2BC) is generated to encode the location of the matching bits. The two bit code is embedded in the image based on a password using three different suggested techniques. Simulation results on IEEE standard images of size 512 by 512 show that the suggested techniques are capable of achieving PSNRs between 56 dB and 67 dB with hiding up to 800 ASCII characters. The three different levels of security make this method highly difficult to intercept and useful for secure open channel communications.

3.1 Introduction

The simplest steganographic techniques embed the bits of the message directly into the least significant bit (LSB) plane of the cover image in a deterministic sequence [64, 65]. Different steganographic techniques focus on a variety of requirements such as
robustness, tamper resistance, imperceptibility, security and capacity [66-71]. Our embedding technique is focused on providing security while maintaining imperceptibility. The proposed method is an extension of the work reported in [72]. The major differences between the two methods are how to deal with the case when there is no match between the higher nibble bits of the randomly selected pixel and the data bit, and further providing three different methods for embedding the two bit code (2BC) using a password. Our method can work in any transform domain, but we are illustrating the ideas in the spatial domain for convenience, although it is limited and is susceptible to cropping, rotation, low bit reverse attack due to LSB weakness and other geometrical attacks and compression techniques. Other domains, however, could further provide improved robustness and tamper resistance.

### 3.2 Matching Process and Generation of 2BC

Let $M(i, j)$ be a randomly selected pixel from the cover image $C$, where $i$ and $j$ represent the row and column of the pixel. Let $X_k$ denote the bit positions of $M(i, j)$ with $X_k = 8, 7, 6, \text{ and } 5$ representing the higher nibble as shown in Figure 3-1. Let ‘$B$’ be the data of size ‘$L$’ bits to be embedded whose $n^{th}$ bit is $B_n$. Then, the matching process and generation of 2BC is done using the following steps.

![Figure 3-1: Representation of the image pixel ‘M’ used for matching.](image-url)
Step 1: Scan the bit positions \(X_k\) (8, 7, 6 and 5) in the randomly selected \(M(i, j)\) pixel and match with \(n^{th}\) bit of data \(B_n\).

Step 2: Generate the 2BC associated with the matching position with 00, 01, 10 and 11 representing positions 5, 6, 7, and 8, respectively.

Step 3: If no match occurs, then all the bits in higher nibble are the same and they are different than the data bit. In such a special case of “no match”, assume that a matching happened in the 5\(^{th}\) position. While decoding, the receiver side will check for this special case and complement the bit obtained in position 5 to get the data bit.

Step 4: Repeat step 1 and 2 for all the bits \((n \leq L)\) of the data \(B_n\).

3.3 The Three Different Suggested Techniques for Embedding the 2BC

Let ‘S’ be the password of length ‘T’ characters. The password could be of any length and any combination of characters like lower case letters, upper case letters, and special characters. The choice of password doesn’t affect the performance of the algorithm. It is used to enhance the security of the algorithm by one more level. This password is converted into a binary code and is used for storing one of the bits of the 2BC. It is repeated until all bits are embedded.

Let \(E(i, j)\) be another randomly selected pixel from the cover image. The lower nibble of \(E\) will be used to hide the 2BC’s obtained from the matching positions of \(M\), in different positions as shown in Figure 3-2 based on a password. If the password bit is ‘0’,
then the first bit of 2BC is saved in position 1, else it is saved in position 2. The second bit of 2BC can be saved using three different techniques.

![Diagram of pixel positions](image.png)

**Figure 3-2:** Representation of the image pixel ‘E’ used for embedding the 2BC based on password.

**Technique 1:** Explicitly send the second bit to the receiving side. This technique will give the highest PSNR at the expense of slightly reducing the level of security and the quantity of encoded data.

**Technique 2:** Save the second position in a fixed location such as position 3 of the pixel.

**Technique 3:** Save the second bit in a specific order. For example, the first 10 bits are hidden in position 1 or 2, whichever, is available after embedding the first bit of the 2BC, the next five bits are hidden in position 3 and next bit is hidden in position 4. Repeat the same pattern until all bits are embedded. By using this approach, the PSNR value can be controlled to some degree.

The mean squared error between the cover image and the stego-image is used as the measure to assess the relative perceptibility due to the embedded data [5]. The Mean
Square Error (MSE) is given by eq. 3-1,

\[
\text{MSE} = \sum_{i=1}^{L} \sum_{j=1}^{W} (f_{i,j} - g_{i,j})^2
\]  

(3-1)

where \(L\) and \(W\) are number of rows and columns respectively in the image, \(f_{i,j}\) is the pixel value of the cover image and \(g_{i,j}\) is the pixel value of the stego-image. The PSNR can be calculated using MSE which is given in eq. 3-2,

\[
\text{PSNR} = 10 \log_{10} \left( \frac{P^2}{\text{MSE}} \right)
\]

(3-2)

where \(P\) is the peak signal value of the cover image (for 8-bit images, \(P = 255\)). Say, we want the PSNR of the image to be around a specific value (50 db). Since the value of ‘\(P\)’ is constant to be ‘255’. To obtain the required PSNR we can play with the MSE. This MSE is dependent on the change in the pixel values. The value to which the intensity of the pixel can change depends upon in which position the data is embedded as shown in Figure 3-3. Hence to obtain the required PSNR we can decide how many bits and in which position they must be changed. In this way the PSNR can be controlled to some degree using the third technique.

![Figure 3-3: Image Bit Error Effects.](image)

The flowchart for embedding the data at the transmitter side including the matching process and the generation and embedding of the 2BC is shown in Figure 3-4.
3.4 Data Retrieval

The same pseudo random algorithms used at the transmitter side are used at the receiver side to find the $M$ and $E$, which are the pixels used for matching the data and
embedding the 2BCs based on password. The location where the first bit of the 2BC is embedded in $E$ can be obtained from the password. Then, depending on the technique
used to embed the second bit of the 2BC, the second bit can be read directly, read from position 3, or read from different positions in a particular order according to techniques 1, 2 and 3, respectively. The extracted bits are then combined to obtain the different 2BCs. The data bits are extracted from the $M$ pixels based on the locations obtained from the 2BC’s. When the location is ‘5’, the receiver will check if the bits in locations 8, 7, 6 and 5 are the same. If they are the same, then this case corresponds to a “no match” and hence the complement of the bit in position 5 is taken as the data bit, else the same bit is taken. The flow chart for data retrieval at the receiver side is shown in Figure 3-5.

3.5 Simulation Results

The proposed data hiding techniques were implemented in MATLAB® (Mathworks, Inc., Natick, MA) using various IEEE standard images for different number of ASCII characters. Tables 3.1 to 3.3 indicate the PSNR values for various stego-images of size 512 x 512 using the three proposed techniques, calculated for 3200 characters. We conducted further experiments with the Lena image using more characters. Table 3.4 indicates the PSNR values for the Lena image calculated for up to 10,400 characters using the three suggested techniques. Thus, using the proposed method, up to 10,400 characters (or 83,200 bits) have been embedded in a 512 x 512 pixel image. This shows that, 31% of the image pixels are used to embed 83,200 bits and yet the perceptual quality of the stego-image is still high. In the first technique, only one bit of the pixel is changed, resulting in very high PSNR values. The second and third methods have comparable PSNR values, but, as mentioned before, the PSNR of the third method can be controlled
depending on number of bits being stored in various positions. Considering a minimum of 34 dB PSNR threshold for stego-image perceptual quality [2], it is clear from the obtained PSNR values that all three techniques can generate stego-images with good perceptual quality. For qualitative assessment, Figures 3-6a and 3-6b show the Lena image before and after steganography using the second technique and Figures 3-7a and 3-7b show the Baboon image using the third embedding technique, for 3200 ASCII characters.

![Lena Image](image1.png)

(a) (b)

Figure 3-6 Lena image using the second proposed technique (a) before embedding, and (b) after embedding

![Baboon Image](image2.png)

(a) (b)

Figure 3-7 Baboon image using the third proposed technique (a) before embedding, and (b) after embedding.
Table 3.1: PSNR values in dB using technique 1

<table>
<thead>
<tr>
<th>No. of characters</th>
<th>Lena (dB)</th>
<th>Baboon (dB)</th>
<th>Peppers (dB)</th>
<th>Airplane (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>800</td>
<td>67.31</td>
<td>67.36</td>
<td>67.21</td>
<td>67.32</td>
</tr>
<tr>
<td>1600</td>
<td>64.34</td>
<td>64.39</td>
<td>64.30</td>
<td>64.17</td>
</tr>
<tr>
<td>2400</td>
<td>62.66</td>
<td>62.62</td>
<td>62.47</td>
<td>62.44</td>
</tr>
<tr>
<td>3200</td>
<td>61.40</td>
<td>61.34</td>
<td>61.20</td>
<td>61.16</td>
</tr>
</tbody>
</table>

Table 3.2: PSNR values in dB using technique 2

<table>
<thead>
<tr>
<th>No. of characters</th>
<th>Lena (dB)</th>
<th>Baboon (dB)</th>
<th>Peppers (dB)</th>
<th>Airplane (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>800</td>
<td>54.17</td>
<td>54.20</td>
<td>54.18</td>
<td>54.40</td>
</tr>
<tr>
<td>1600</td>
<td>51.08</td>
<td>51.20</td>
<td>51.29</td>
<td>51.37</td>
</tr>
<tr>
<td>2400</td>
<td>49.39</td>
<td>49.48</td>
<td>49.54</td>
<td>49.55</td>
</tr>
<tr>
<td>3200</td>
<td>48.14</td>
<td>48.17</td>
<td>48.32</td>
<td>48.31</td>
</tr>
</tbody>
</table>

Table 3.3: PSNR values in dB using technique 3

<table>
<thead>
<tr>
<th>No. of characters</th>
<th>Lena (dB)</th>
<th>Baboon (dB)</th>
<th>Peppers (dB)</th>
<th>Airplane (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>800</td>
<td>55.77</td>
<td>56.07</td>
<td>56.06</td>
<td>55.97</td>
</tr>
<tr>
<td>1600</td>
<td>52.78</td>
<td>52.92</td>
<td>53.01</td>
<td>52.86</td>
</tr>
<tr>
<td>2400</td>
<td>50.99</td>
<td>51.15</td>
<td>51.18</td>
<td>51.07</td>
</tr>
<tr>
<td>3200</td>
<td>49.77</td>
<td>49.97</td>
<td>50.07</td>
<td>49.86</td>
</tr>
</tbody>
</table>

Table 3.4: PSNR values in dB for the Lena image using the three proposed techniques

<table>
<thead>
<tr>
<th>No. of characters</th>
<th>4000</th>
<th>4800</th>
<th>5600</th>
<th>6400</th>
<th>7200</th>
<th>8000</th>
<th>8800</th>
<th>9600</th>
<th>10400</th>
</tr>
</thead>
<tbody>
<tr>
<td>Technique1</td>
<td>60.26</td>
<td>59.49</td>
<td>58.84</td>
<td>58.24</td>
<td>57.73</td>
<td>57.28</td>
<td>56.84</td>
<td>56.49</td>
<td>56.14</td>
</tr>
<tr>
<td>Technique2</td>
<td>47.16</td>
<td>46.38</td>
<td>45.69</td>
<td>45.14</td>
<td>44.58</td>
<td>44.12</td>
<td>43.74</td>
<td>43.35</td>
<td>43.10</td>
</tr>
<tr>
<td>Technique3</td>
<td>48.81</td>
<td>48.03</td>
<td>47.37</td>
<td>46.74</td>
<td>46.33</td>
<td>45.80</td>
<td>45.34</td>
<td>45.02</td>
<td>44.70</td>
</tr>
</tbody>
</table>

3.6 Summary

A new steganography method for hiding classified data based on matching of bit
values has been presented. The most important feature of this method is the difficulty to which a third party would encounter in trying to intercept the hidden data. This difficulty arises from the two random algorithms used to select the matching and embedding pixels, the fact that the data bits are not hidden directly, and the use of password. In the proposed algorithm, an eavesdropper can destroy the message by low-bit reverse attack due to LSB weakness but cannot interpret the message. Simulation results on the IEEE standard Lena image with hiding 800 text characters indicate that the demonstrated embedding techniques can achieve PSNR of up to 67dB. Lastly, the reported method has significant potential to serve as an effective means for secure transmission using open channel communications.
4.1 Introduction

In the previous chapter, we have seen a data hiding technique based on a password. Although the PSNR values obtained were considerably good in the previous technique, the technique is still vulnerable to some kinds of geometric and compression attacks as the LSB bits are used for hiding the data. In the proposed technique, we make use of most significant bits of the pixel to encrypt the data. The suggested technique can be used for both data hiding as well as data encryption.

4.2 Proposed Hiding Algorithm

Select the pixels in a random fashion from the cover image $C$ of size $M \times N$, where $C(i, j)$ denotes a pixel at $i^{th}$ row and $j^{th}$ column and $U_k$ be the intensity value of that pixel. Let $X_k$ denote the bit position of pixel with $X_k = 8$ and $X_k = 1$ representing the MSB and LSB respectively as shown in Figure 4-1. Let $B$ be the data of size ‘$L$’ bits to be
embedded whose \( n^{th} \) bit is \( B_n \).

![Figure 4-1: Representation of image pixel.](image1)

![Figure 4-2: Representation of 2-bit code.](image2)

### 4.2.1 Embedding Data

**Step 1:** We scan the bit position \( (X_k = 8, 7, 6, 5) \) in the randomly selected \( I(i,j) \) pixel and match with \( n^{th} \) bit of data \( B_n \).

**Step 2:** Note the pattern in which the two bit code associated with each of the position \( (8, 7, 6, 5) \) 11, 10, 01 and 00 respectively is present in the higher nibble based on password which is discussed below.

**Step 3:** If no match occurs then assume that the bit in 5\(^{th}\) position was matching with data bit.

**Step 4:** Repeat 1 and 2 steps for all the bits \( (n \leq L) \) of the data \( B_n \).

### 4.2.2 Pattern of 2-bit code present in higher nibble
The pattern in which the two bit code (2BC) is present in the higher nibble is made a note. The pattern of the two bit code is based on a password. The password could be any combination of characters like lower case letters, upper case letters and special characters and could be of any length. This password is converted into a binary code and is used for storing the 2-bit codes. This password is repeated till the pattern of all the characters of the data is noted.

The password when converted into binary code will comprise of 0’s and 1’s. If the password bit is ‘0’ then the second bit of two bit code as shown in Figure 4-2 is matched with the bit in position 5 as shown in Figure 4-1. If a match is found then ‘M’ is noted else as ‘N’. The first bit of 2 bit code is matched for complement or same bit based on the ‘M’ and ‘N’ which is noted above.

If ‘M’ is noted above then look for complement of the first bit in positions 6, 7 and 8, if complement is found then we note it as ‘PM’ where P is either of 6, 7 or 8 where the match is found. If the complement is not found then we note it as ‘NM’ where N signifies no match.

And if ‘N’ was noted above then we look for same bit as of the first bit in positions 6, 7 and 8, if same bit is found then we note it as ‘PN’ where P is either of 6, 7 or 8 where the match is found. If the same bit is not found then we note it as ‘NN’ where N signifies no match.

If the password bit is ‘1’ then the second bit of two bit code as shown in Figure 4-2 is matched with the bit in position 6 of the pixel as shown in Figure 4-1. If a match is found then ‘M’ is noted else as ‘N’. The first bit of 2 bit code is matched for complement or same bit based on the ‘M’ and ‘N’ which is noted above.
If ‘M’ is noted above then we look for complement of the first bit in positions 5, 7 and 8, if complement is found then we note it as ‘PM’ where P is either of 5, 7 or 8 where the match is found. If the complement is not found then we note it as ‘NM’ where N signifies no match.

And if ‘N’ was noted above then we look for same bit as of the first bit in positions 5, 7 and 8, if same bit is found then we note it as ‘PN’ where P is either of 5, 7 or 8 where the match is found. If the same bit is not found then we note it as ‘NN’ where N signifies no match. The flowchart for encoding data is shown in Figure 4-3. Table 4.1 shows the code words for different matching positions in pixel.

### Table 4.1: Code words for different matching positions in pixel.

<table>
<thead>
<tr>
<th>Code word</th>
<th>Password ‘1’</th>
<th>Password ‘0’</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>5N</td>
<td>6N</td>
</tr>
<tr>
<td>001</td>
<td>5M</td>
<td>6M</td>
</tr>
<tr>
<td>010</td>
<td>7N</td>
<td>7N</td>
</tr>
<tr>
<td>011</td>
<td>7M</td>
<td>7M</td>
</tr>
<tr>
<td>100</td>
<td>8N</td>
<td>8N</td>
</tr>
<tr>
<td>101</td>
<td>8M</td>
<td>8M</td>
</tr>
<tr>
<td>110</td>
<td>NN</td>
<td>MM</td>
</tr>
<tr>
<td>111</td>
<td>MM</td>
<td>NN</td>
</tr>
</tbody>
</table>
Figure 4-3: Flowchart for encoding the data.

where P in PM and PN is 2, 3 and 4 when password is ‘0’ and is 1, 3 and 4 when password is ‘1’
4.3 Data Retrieval

The hidden data can be extracted by using the same random selection algorithm to select the pixels \( I(i, j) \) of the stego-image. Use the embedding code word and password to find out the data bit using known pixel number \( I(i, j) \) and its corresponding code bit pair positions. The extracted data bits are then arranged in their original order to complete the retrieval process. The flowchart for decoding the data is shown in Figure 4-4.

---

**Figure 4-4: Flowchart for decoding the data.**
4.4 Illustrative Examples

Say the word ‘HERO’ needs to be embedded in the image. The first character to be embedded is ‘H’. The ASCII value of ‘H’ is 72. This decimal value when converted into binary will be ‘01001000’. Now these 8 bits need to be hidden in the image. Each bit is stored in one pixel; hence 8 pixels are required to hide these 8 bits. Let the password be ‘pen’. The password when converted into binary would be ‘011100000110010101101110’. This three lettered word is converted into 24 bits of binary code. Thus 24 bits of data can be embedded based on this password and if more bits are required to be embedded then the password is repeated as many times as required to embed all the data bits.

CASE 1: The first bit of the character ‘H’ which is ‘0’ needs to be embedded. A pixel is randomly selected say at location [50, 50] and let its intensity be ‘202’. Converting this decimal value into binary value we get ‘11001010’.

<table>
<thead>
<tr>
<th>1</th>
<th>1</th>
<th>0</th>
<th>0</th>
<th>1</th>
<th>0</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>7</td>
<td>6</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

This bit ‘0’ is compared with bits in position 8, 7, 6 and 5. Here in this case the match is found in position 6. Now the pattern in which the 2BC associated with position 6 which is ‘01’ needs to be noted in higher nibble of the pixel based on password.

The first bit of password is ‘0’, therefore the second bit of two bit code which is ‘1’ is matched with the bit in position 5 of the pixel. Here match is not found hence it is noted as ‘N’. Now for the first bit of the 2BC we look for same bit which is ‘0’ in positions 6, 7 and 8. Here the match is found in position 6. Hence the 2BC is noted as ‘6N’
CASE 2: The second bit of the character ‘H’ which is ‘1’ needs to be embedded. Again we select another pixel. A pixel is randomly selected say at location [150, 150] and let its intensity be ‘250’. Converting this decimal value into binary value we get ‘11111010’.

This bit ‘1’ is compared with bits in position 8, 7, 6 and 5. Here in this case the match is found in position 8. Now the pattern in which the 2-bit code associated with 8th position ‘11’ needs to be noted in higher nibble of the pixel based on password.

The second bit of password is ‘1’, therefore the second bit of two bit code which is ‘1’ is matched with the bit in position 6 of the pixel. Here match is found hence it is noted as ‘M’. Now for the first bit of the 2BC we look for complement of that bit which is ‘0’ in positions 5, 7 and 8. Here the match is not found in any of the positions. Hence the 2BC is noted as ‘MM’.

CASE 3: The third bit of the character ‘H’ which is ‘0’ needs to be embedded. Again we select another pixel. A pixel is randomly selected say at location [15, 15] and let its intensity be ‘15’. Converting this decimal value into binary value we get ‘00001111’.

This bit ‘0’ is compared with bits in position 8, 7, 6 and 5. Here in this case the match is found in position 8. Now the pattern in which the 2-bit code associated with 8th position ‘11’ needs to be noted in higher nibble of the pixel based on password.

The third bit of password is ‘1’; therefore the second bit of two bit code which is 1 is matched with the bit in position ‘6’ of the pixel. Here match is not found hence it is noted
as ‘N’. Now for the first bit of the 2BC we look for same bit which is ‘1’ in positions 5, 7 and 8. Here the match is not found in any of the positions. Hence the 2BC is noted as ‘NN’.

**CASE 4:** The fourth bit of the character ‘H’ which is ‘0’ needs to be embedded. Again we select another pixel. A pixel is randomly selected say at location [75, 75] and let its intensity be ‘159’. Converting this decimal value into binary value we get ‘10011111’.

\[
\begin{array}{cccccccc}
8 & 7 & 6 & 5 & 4 & 3 & 2 & 1 \\
1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 \\
\end{array}
\]

This bit ‘0’ is compared with bits in position 8, 7, 6 and 5. Here in this case the match is found in position 7. Now the pattern in which the 2-bit code associated with 7th position which is ‘10’ needs to be noted in higher nibble of the pixel based on password.

The fourth bit of password is ‘1’; therefore the second bit of two bit code which is ‘0’ is matched with the bit in position 6 of the pixel. Here match is found hence it is noted as ‘M’. Now for the first bit of the 2BC we look for complement of that bit which is ‘0’ in positions 5, 7 and 8. Here the match is found in position ‘7’. Hence the 2BC is noted as ‘7M’.

### 4.5 Results

Experiments were conducted by encrypting 100 ASCII characters. The sensitivity of the proposed algorithm is tested by slightly changing the password. The input message given was “Novel Data Encryption and Data hiding algorithms for secure wireless communication through images”.

The output of 100 decrypted characters using different password looked like
The proposed technique uses three times the length of the message bits to decrypt the data. Although the sensitivity of the algorithm is very high, the proposed technique is not practical. Hence, in our next chapters we try to reduce the number of bits required to decrypt the message.

### 4.6 Summary

A novel data encryption is introduced for encoding ASCII text characters based on matching of bit values has been presented. The proposed algorithm can also be used as data hiding technique in images which retains perceptual quality of the cover media as none of the pixels are changed. The Simulation results are shown by encrypting 100 ASCII characters. The security of the proposed algorithm is tested using key sensitivity analysis. The enhanced robustness in data hiding is due to data matching but not data substitution. Therefore the proposed method can be effectively utilized for secure transmission using open channel environment.
Chapter 5

Symmetric Self-Synchronous Stream Encryption Using Images

5.1 Introduction

This Chapter introduces a self-synchronous stream cipher with a considerable key space. In the proposed technique, an image is used as a non-linear filtering function to encrypt the message. The proposed technique can be used for any form of message such as text or image. Simulation results for IEEE standard Baboon image are presented. This chapter discusses the security aspects of the proposed cipher using histogram, entropy and correlation analysis. The histogram of the cipher image is uniform representing almost equivalent probability of occurrence of each intensity level. The entropy of the cipher image is almost equal to the theoretical value of 8, indicating that the information leakage in the proposed encryption process is negligible. Finally, the correlation coefficients prove that there exists almost zero correlation between the pixels. Therefore, the proposed stream cipher can be efficiently used in real time multimedia and wireless applications because it has simple structure and generates a key stream faster than other
generators.

Synchronous stream ciphers are generally designed to utilize confusion but not diffusion. Self-synchronous stream encryption systems are mainly used because a non-repeating key is generated and secondly the statistics of the plaintext message are diffused throughout the cipher text. Moreover, it avoids the necessity of generating the pseudo random key stream at receiver exactly in synchronous to the received cipher text stream. However, the fact that the key is exposed in the cipher text is a basic weakness. This problem can be eliminated by passing the cipher text characters through a non-linear block to obtain the key characters.

5.2 Proposed Encryption Technique

The block diagram of the encoder and decoder of our encryption technique are shown in Figure 5-2 and 5-3, respectively. As mentioned before, this technique is based on symmetric self-synchronous stream encryption. The technique assumes a database of images that are available for the public that is everybody has access to these images. The images should be selected carefully so that they don’t 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. 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 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. Figure 5-1 shows the Lena image and its corresponding parity bit-plane.

![Lena Image](image1)

Figure 5-1: (a) Lena image as an example of public images (b) Corresponding parity bit plane of the Lena image.

After selecting $P$, 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 in the selected public image. The length of this address depends on the size of the image. For example, if the image is $M \times N$ pixels, the address will be $\log_2(M \times N)$ bits. A block $B$ of size $L \times W$ pixels around that randomly selected pixel will be extracted from the image. The starting point $I(n, k)$ of this block will be computed from the randomly selected pixel $P(n, k)$ will be extracted from $P$. 

41
based on the following algorithm:

\[
\text{for } i = 0: L - 1 \\
\text{for } j = 0: W - 1 \\
B(i, j) = P((n + i) \mod M, (k + j) \mod N) \\
\text{end;}
\]

end;

Figure 5-2: Block diagram of the encoder.

![Block Diagram of the Encoder](image)

Figure 5-3: Block diagram of the decoder.

![Block Diagram of the Decoder](image)

The second part of the private key will be used to initialize the shift registers. The
contents of the shift register will be used as an address to randomly access a single bit from $B$. The size of the register is $S = \log_2(L \times W)$. The selected bit is xored with the message bit to get the ciphered bit. The ciphered bit is fed back to the most significant bit of the shift register. 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 ‘$B$’. Also, the shift register will be initialized by the second part of the private key. To decrypt the message, the ciphered bits are fed into the input of the most significant bit of the shift register and also Xored with the output bits from $B$ to get the message bits.

### 5.3 Simulation Results and Discussion

For the proposed stream cipher encryption technique, any sort of messages like text, or image can be encrypted as well as decrypted bit by bit. We have introduced a novel random sequence generation process based on shift registers and using images. The initial conditions of the shift register and the selection of block of an image dictates the complexity of encryption algorithm for breaking. The ciphered bit is fed back to the most significant bit of the shift register, after encryption of each bit of the message with the image bit, thus, generating a self-synchronous key.

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). In this chapter, a public image of size $(512 \times 512)$ is considered as an example. The source image of size $200 \times 200$ is encrypted and
transformed into a cipher image. A small block of size 200 × 200 is used from the parity image. The histogram, entropy and correlation between two adjacent pixels computations for the plain and cipher images are carried out using the procedure given in [14, 63, and 15]. The plain image and its corresponding cipher image are shown in Figure 5-4.

5.3.1 Key Space Analysis

The key space is the total number of different keys that can be used in the encryption. For a secure image encryption, the key space should be large enough to make brute force attacks infeasible [32, 33 and 41]. The key space for the proposed technique is $M \times N \times L \times W \times P$, a flexible, moderately large key space, which comprises of number of shift registers, size of the image. Hence for this image encryption, this large key space is sufficient which is immune to all kinds of brute force attacks.

5.3.2 Statistical Analysis

Statistical analysis generally depends on the measure of the randomness of the cipher image. Also it works on the relative frequency of the occurred cipher image. It is eminent that a lot of ciphers have been successfully analyzed with the help of statistical analysis and numerous statistical attacks have been formulated on them. The following aspects narrated to statistical attack are considered in this thesis.

A. Histograms

To prevent the leakage of information to an adversary, it is important to ensure that cipher image does not have any statistical resemblance to the input image. An image
The histogram shows how pixels in an image are distributed by plotting the number of pixels at each intensity level. In this chapter, the histograms are plotted for input and cipher image and are as shown in Figures 5-5 and 5-6. The histogram of the input image has large spikes. But, the histogram of the cipher image is nearly smooth and uniform, representing almost equivalent probability of occurrence of each intensity level. They are considerably different and tolerate no statistical similarity to the input image. Hence, this does not give any hint to use any statistical attack on the proposed stream cipher.

![Input Image](image1.png) ![Cipher Image](image2.png)

Figure 5-4: (a) Input image, (b) Cipher Image.
Figure 5-5: Histogram of Plain image.

Figure 5-6: Histogram of Cipher Image.
B. Entropy

Entropy is a statistical measure of randomness in information theory. The performance of the encryption algorithms are measured by computing entropy of the input and cipher images and then comparing them. The Entropy $E$ of the image is calculated using eq. 5-1

$$E = \sum_{i=0}^{255} P(i) \log_2 \left( \frac{1}{P(i)} \right)$$  (5-1)

where $P(i)$ represents the probability of symbol $i$ and the entropy is expressed in bits. Let us suppose that the source emits $2^8$ symbols with equal probability, i.e., 0 to 255, $m = \{m_0, m_1, \ldots, m_{255}\}$ after evaluating the above eq. 5-1, we obtain its entropy $E = 8$, corresponding to a truly random source. Actually, given that a practical information source seldom generates random messages, in general its entropy value is smaller than the ideal one. However, when the messages are encrypted, their entropy should ideally be 8. If the output of such a cipher emits symbols with entropy less than 8, there exists certain degree of predictability, which threatens its security. The values of entropies obtained for plain images and cipher images of the proposed scheme are given in the Table 5.1. The entropy values for cipher images of the proposed scheme are very close to the ideal value 8. This implies that the information leakage in the proposed encryption process is negligible and the encryption algorithm is secure against the entropy based attack.

<table>
<thead>
<tr>
<th>Source image</th>
<th>Ciphered image</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.4250</td>
<td>7.9881</td>
</tr>
</tbody>
</table>
C. Correlation Analysis

There exists high correlation among adjacent pixels in most of the original images. It is mainstream task of an efficient image encryption algorithm to eliminate the correlation of pixels [34, 60 and 61]. Two highly uncorrelated sequences have approximately zero correlation coefficient. The correlation coefficient between two adjacent pixels in an image is determined using the formula given in eq. 5-2[65].

\[
R_{xy} = \frac{\sum_{i=1}^{N}(x_i - E(x))(y_i - E(y))}{\sqrt{\sum_{i=1}^{N}(x_i - E(x))^2 \sum_{i=1}^{N}(y_i - E(y))^2}}
\] (5-2)

where \( E(x) = \text{mean} (x_i) \) and \( x, y \) are gray values of two adjacent pixels in the image. In the proposed algorithm, the correlation coefficient of 1000 randomly selected pairs of vertically, horizontally and diagonally adjacent pixels is determined. The correlation coefficients of plain Lena image and its corresponding cipher image in three directions are listed in Table 5.2. It is seen that the two neighboring pixels in the input image are correlated too much, while there is a negligible correlation between the two neighboring pixels in the cipher image. From the above analysis, we can conclude that the proposed stream cipher is secure against correlation attacks. Figures 5-7 to 5-12 show the distribution of randomly selected pairs in the source image and cipher image in horizontal, vertical, and diagonal directions respectively.

Table 5.2: Correlation Coefficients in all the three directions

<table>
<thead>
<tr>
<th>Direction</th>
<th>Vertical</th>
<th>Horizontal</th>
<th>Diagonal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plain-image</td>
<td>0.9621</td>
<td>0.9959</td>
<td>0.9498</td>
</tr>
<tr>
<td>Cipher-image</td>
<td>0.0176</td>
<td>-0.0951</td>
<td>0.0469</td>
</tr>
</tbody>
</table>
Figure 5-7: Distribution of the randomly selected pairs of horizontally adjacent pixels in the source image.

Figure 5-8: Distribution of the randomly selected pairs of horizontally adjacent pixels in the cipher image.
Figure 5-9: Distribution of the randomly selected pairs of vertically adjacent pixels in the source image.

Figure 5-10: Distribution of the randomly selected pairs of vertically adjacent pixels in the cipher image.
Figure 5-11: Distribution of the randomly selected pairs of diagonally adjacent pixels in the source image.

Figure 5-12: Distribution of the randomly selected pairs of diagonally adjacent pixels in the cipher image.
5.3.3 Key Sensitivity Analysis

An efficient encryption algorithm should be sensitive to secret key i.e. a small change in secret key during decryption process results into a completely different decrypted image [32, 33]. The decrypted image using the proposed technique with the correct key is shown in Figure 5-13a. 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 Figure 5-13b. From the figures, it can be seen that the decoders failed to decrypt the images correctly in all cases except when the starting pixel was correct. Hence, the proposed technique is highly sensitive to the starting pixel \( P(n, k) \).

![Decrypted images](image)

Figure 5-13 (a) Decrypted image with correct key using the proposed technique; (b) decrypted image using the proposed technique with a slightly different starting pixel.

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 as shown in eq. 5-3

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

(5-3)

where \( C_1 \) and \( C_2 \) are the two cipher images, whose corresponding keys slightly differ and the gray-scale values of the pixels at location \((i, j)\) are labeled as \( C_1(i, j) \) and \( C_2(i, j) \), respectively. The NPCR is defined as in eq. 5-4

\[ NPCR = \frac{\sum_{i,j} D(i,j)}{M \times N} \times 100\% \]  

(5-4)

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. 5-5

\[ NPCR \text{ expected} = (1 - 2^{-L}) \times 100\% \]  

(5-5)

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. 5-6

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

(5-6)

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 5.3. Based on
Table 5.3, 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.

Table 5.3: NPCR and UACI values for the proposed technique

<table>
<thead>
<tr>
<th>Measures</th>
<th>Slight change in starting pixel</th>
</tr>
</thead>
<tbody>
<tr>
<td>NPCR</td>
<td>99.7892</td>
</tr>
<tr>
<td>UACI</td>
<td>33.6826</td>
</tr>
</tbody>
</table>

5.4 Conclusion

In this chapter, shift register based self-synchronous key generation for stream cipher is proposed. The key streams are generated based on the combination of shift register and a public image. In this chapter, (200×200) input image is considered as an example. The input image and the respective cipher image histograms are conversed. It is seen that cipher image does not have residual information and the histogram is nearly smooth and uniform, offering good security for images. The entropy and the correlation between two neighboring pixels for the input and cipher images are computed and analyzed. All the simulation and experimental analysis show that the proposed image encryption system has large key space, high sensitivity to secret keys, better diffusion of information in the ciphered images and low correlation coefficients. Hence, the proposed image encryption algorithm has high level of security with less computation and is more robust towards cryptanalysis. It can be scrutinized that the proposed encryption scheme can be a potential candidate for real time multimedia applications.
Chapter 6

Symmetric Synchronous Stream Encryption Using Images

6.1 Introduction

In this chapter, 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. 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. 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 gray 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. It has a simple structure and can generate a key stream faster than other generators.
6.2 Proposed Encryption Technique

The block diagrams of the encoder and decoder of the proposed encryption technique are shown in Figure 6-1 and 6-2, respectively. 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 don’t 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. Figure 5-1 shows the Lena image and its corresponding parity-bit plane.

![Block diagram of the encoder for the proposed encryption technique.](image)

Figure 6-1: Block diagram of the encoder for the proposed encryption technique.
Figure 6-2: Block diagram of the decoder for the proposed encryption technique.

6.3 Simulation Results and Discussion

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 $M \times N$ pixels, the address will be $\log_2(M \times N)$ bits. A block $B$ of size $L \times W$ starting from point $P(n, k)$ will be extracted from $P$ based on the following algorithm:

\[
\begin{align*}
& \text{for } i = 0: L - 1 \\
& \quad \text{for } j = 0: W - 1 \\
& \quad \quad B(i, j) = P((n + i)mod M, (k + j)mod N) \\
& \quad \text{end}; \\
& \text{end};
\end{align*}
\]

The second part of the private key will be used to initialize the LFSR. The contents of
A LFSR will be used as an address to randomly access a single bit from $B$. The size of the register is $S = \log_2(L \times W)$. 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 $B$.

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 x 200 is shown in Figure 6-3a, and its corresponding encrypted image is shown in Figure 6-3b.

![Source Image](image1.png)  ![Ciphered Image](image2.png)

Figure 6-3: (a) Source image; (b) Ciphered image using the proposed technique.

In cryptography, an effective encryption algorithm should have desirable features for
surviving all kinds of known attacks, including brute-force, statistical and sensitivity attacks [60 and 61]. 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.

6.3.1 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 [54, 55 and 62].

A. Histograms

The histogram of the source image is shown in Figure 6.4 and the histograms of ciphered images using the proposed technique is shown in Figure 6.5. 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.

**B. 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 [41, 63]. It quantifies the amount of information contained in data, usually in bits or bits/symbol. The performance of the image encryption algorithms are measured by computing entropy of the source and ciphered images and then comparing them. The Entropy ‘E’ of the image is calculated using eq. 5-1. The values of entropies obtained for the source image and the ciphered image are given in the Table 6.1. 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>
<thead>
<tr>
<th>Source image</th>
<th>Ciphered image</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.4250</td>
<td>7.9876</td>
</tr>
</tbody>
</table>

Table 6.1: Entropy values for the source and ciphered image
Figure 6-4: Histogram of source image.

Figure 6-5: Histogram of ciphered image.
C. Correlation Analysis

It is a conventional task of an efficient image encryption algorithm to eliminate the correlation of pixels [34, 37, and 39]. In this simulation, the correlation coefficient of 1000 randomly selected pairs of vertically, horizontally and diagonally adjacent pixels is determined. Figure 6-6 to 6-11 displays the distribution of the randomly selected pairs of adjacent pixels in horizontal, vertical and diagonal directions 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 eq. 5-2.

The correlation coefficients of the plain Baboon image and its corresponding cipher image in three directions are listed in Table 6.2. 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.

Table 6.2: Correlation Coefficients of the source image and the ciphered images in three directions

<table>
<thead>
<tr>
<th>Direction</th>
<th>Source Image</th>
<th>Ciphered Image</th>
</tr>
</thead>
<tbody>
<tr>
<td>Horizontal</td>
<td>0.8851</td>
<td>0.0257</td>
</tr>
<tr>
<td>Vertical</td>
<td>0.8875</td>
<td>0.006861</td>
</tr>
<tr>
<td>Diagonal</td>
<td>0.8385</td>
<td>-0.0018</td>
</tr>
</tbody>
</table>
Figure 6-6 Distribution of the randomly selected pairs of horizontally adjacent pixels in the source image

Figure 6-7 Distribution of the randomly selected pairs of horizontally adjacent pixels in the cipher image
Figure 6-8: Distribution of the randomly selected pairs of vertically adjacent pixels in the source image.

Figure 6-9: Distribution of the randomly selected pairs of vertically adjacent pixels in the cipher image.
Figure 6-10: Distribution of the randomly selected pairs of diagonally adjacent pixels in the source image.

Figure 6-11: Distribution of the randomly selected pairs of diagonally adjacent pixels in the cipher image.
6.3.2 Key Sensitivity Analysis

An efficient encryption algorithm should be sensitive to secret key i.e. a small change in secret key during decryption process results into a completely different decrypted image [56 and 57]. The decrypted image using the proposed technique with the correct key is shown in Figure 6-12a. 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 Figure 6-12b. 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 Figure 6-12c. 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 \( P(n, k) \). The UACI and NPCR values are calculated using the equations 5-3 and 5-4. 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 6.3. Based on Table 6.3, 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.
Figure 6-12 (a) Decrypted image with correct key using the proposed technique; (b) decrypted image using the proposed technique with a slightly different initial condition; (c) decrypted image using the proposed technique with the same initial condition but a different starting pixel

Table 6.3: NPCR and UACI values for the two techniques

<table>
<thead>
<tr>
<th>Measures</th>
<th>Slight change in initial condition</th>
<th>Slight change in starting pixel</th>
</tr>
</thead>
<tbody>
<tr>
<td>NPCR</td>
<td>99.6275</td>
<td>99.6826</td>
</tr>
<tr>
<td>UACI</td>
<td>33.4701</td>
<td>33.5694</td>
</tr>
</tbody>
</table>
6.3.3 Key Space Analysis

The key space is the total number of different keys that can be used in the encryption. For a secure image encryption, the key space should be large enough to make brute force attacks infeasible [41 and 59]. The key space for the proposed technique for a set of public images $K$ of size $M \times N$ and a selected block of size $L \times W$ is $K \times M \times N \times L \times W$, a flexible moderately 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. This large key space is sufficient and is immune to all kinds of brute force attacks. The password of the user will be of size $\log_2(K \times M \times N \times L \times W)$.

6.4 Conclusion

This chapter introduced a new stream 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.
Chapter 7

Conclusions and Future Work

7.1 Conclusions

For the first time, this thesis introduces the use of public images for secure communication of information. In the first phase of the thesis, a novel data hiding technique is proposed based on a password. The most important feature of this method is the complexity to which an adversary would encounter in trying to decode the hidden data. This difficulty arises from the two random algorithms used to select the matching and embedding pixels, the fact that the data bits are not hidden directly, and the use of password. In the proposed algorithm, an eavesdropper can destroy the message by low-bit reverse attack due to LSB weakness but cannot interpret the message. Simulation results on the IEEE standard Lena image with hiding 800 text characters indicate that the demonstrated embedding techniques can achieve PSNR of up to 67dB. This partial data matching technique is further developed into a complete matching technique, where no pixel is changed. The new matching technique can be used to for data hiding as well as data encryption.

In the second phase of thesis, a shift register based self-synchronous key generation
for stream cipher is proposed. The key streams are generated based on the combination of shift register and a randomly accessed bit from the public image. Simulation tests were conducted on an image of size 200 x 200. The input image and the respective cipher image histograms are conversed. It is observed that cipher image does not have residual information and the histogram is nearly smooth and uniform, offering good security for images. The entropy and the correlation between two neighboring pixels for the input and cipher images are computed and analyzed. All the simulation and experimental analysis show that the proposed image encryption system has large key space, high sensitivity to secret keys, better diffusion of information in the ciphered images and low correlation coefficients. Hence, the proposed image encryption algorithm has high level of security with less computation and is more robust towards cryptanalysis. This thesis also introduced another stream 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 two 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. It can be scrutinized that the proposed encryption scheme can be a potential candidate for real time multimedia applications.

7.2 Future Work

Future work in this direction could be exploiting other ways to improve the utilization
of public images for secure communication. The matching technique proposed can be used as a robust data hiding technique, if the number of bits to decrypt the message is further reduced. Moreover, if both data encryption and steganography are used together, the data security can be highly improved. Future work also includes implementing the proposed stream encryption algorithms in hardware to test its speed.
References


Appendix-A

A.1 Source Code of Symmetric Synchronous Stream Encryption Technique

cle
clear all
close all

tic;
f = imread('D:\All three methods final code\lena512.tiff');
k2 = size(f);
% figure,imshow(f,[]);
% impixelinfo();

fr = f(:,:,1);
fg = f(:,:,2);
fb = f(:,:,3);
f = cat(3,fr,fg,fb);

% FINDING PARITY BIT IN EACH PIXEL
% CONVERTING THE INTENSITY VALUES INTO BINARY

Tr = cell2mat(num2cell(de2bi(fr,8,'left-msb')));
Tg = cell2mat(num2cell(de2bi(fg,8,'left-msb')));
Tb = cell2mat(num2cell(de2bi(fb,8,'left-msb')));

T = cat(3,Tr,Tg,Tb);

% CALCULATING THE PARITY BIT

A = zeros(k2(1)*k2(1),1);
B = zeros(k2(1)*k2(1),1);
for i1 = 1:k2(1)*k2(1)
    A(i1,:) = A(i1,:)+length(find(T(i1,:)));
    if mod(A(i1,:),2) == 0
        B(i1,:) = 0;
    else
        B(i1,:) = 1;
    end
end

u2 = zeros(k2(1),k2(1));
w = reshape(B',[],8);
NI = bi2de(w,'left-msb');
n = k2(1);
for l = 1:k2(1)
    u2(l,:) = w(((l-1)*n)+1:l*n);
end

% Randomly select one pixel
rand('state',sum(300*500)); %#ok<RAND>
H = randi(k2(1),1,2);

% SELECT A SMALL PARITY BLOCK of size 128 x 128
P = [ 128 128];
J = [ H(1)+P(1) H(2)+P(2)];
if (J(1)>512)
    H(1) = H(1)-127;
end
if (J(2)>512)
    H(2) = H(2)-127;
end
J = [ H(1)+P(1) H(2)+P(2)];
L1 = zeros(J(1),J(2));
for l1 = H(1):J(1)
    for l2 = H(2):J(2)
        L1(l1,l2) = L1(l1,l2) + f(l1,l2);
    end
end
L = L1(H(1):H(1)+127,H(2):H(2)+127);
f1 = u2(H(2):H(2)+127,H(1):H(1)+127);
% Select another SMALL PARITY BLOCK just adjacent to size 128 x 128

\[ Hw = H + [1 \ 1]; \]
\[ Pw = [128 \ 128]; \]
\[ Jw = [Hw(1)+Pw(1) \ Hw(2)+Pw(2)]; \]
\[ \text{if} \ (Jw(1)>512) \]
\[ Hw(1) = Hw(1)-127; \]
\[ \text{end} \]
\[ \text{if} \ (Jw(2)>512) \]
\[ Hw(2) = Hw(2)-127; \]
\[ \text{end} \]
\[ Jw = [Hw(1)+Pw(1) \ Hw(2)+Pw(2)]; \]
\[ L1w = zeros(Jw(1),Jw(2)); \]
\[ \text{for} \ l1w = Hw(1):Jw(1) \]
\[ \text{for} \ l2w = Hw(2):Jw(2) \]
\[ L1w(l1w,l2w) = L1w(l1w,l2w) + f(l1w,l2w); \]
\[ \text{end} \]
\[ \text{end} \]
\[ Lw = L1w(Hw(1):Hw(1)+127,Hw(2):Hw(2)+127); \]
\[ f1w = u2(Hw(2):Hw(2)+127,Hw(1):Hw(1)+127); \]

% INPUT IMAGE TO BE ENCRYPTED

\[ u3 = imread('D:\All three methods final code\baboon.jpg'); \]
\[ u4 = rgb2gray(u3); \]
\[ \text{figure,imshow(u4,[]);} \]
\[ \%impixelinfo; \]
\[ \text{figure,imhist(u4);} \]
\[ u5 = u4; \]
\[ E1 = entropy(u4) \]

% Converting the intensity values into binary

\[ Tin1 = \text{cell2mat(num2cell(de2bi(u4,8,'left-msb')));} \]

% TOTAL LENGTH MESSAGE

\[ w1 = \text{reshape(Tin1,[],1);} \]

% LINEAR SHIFT REGISTERS OUTPUT

\[ s = [1 \ 0 \ 1 \ 0 \ 1 \ 0 \ 1 \ 0 \ 1 \ 0 \ 0 \ 0]; \]
\[ t = [3,4,5,14]; \]
n=length(s);
c(1,:)=s;
m=length(t);
b = zeros(1,m-1);

k = 1;
while (k < 2^n-1)
b(1)=xor(s(t(1)), s(t(2)));
if m>2
    for i=1:m-2
        b(i+1)=xor(s(t(i+2)), b(i));
    end
end
j=1:n-1;
s(n+1-j)=s(n-j);
s(1)=b(m-1);
c(k+1,:)=s;
k = k+1;
end

c = [c ; s];
c = [c; c; c; c; c; c; c; c; c; c; c; c; c; c; c; c; c; c; c; c; c; c; c; c; c; c; c; c; c; c; c; c; c; c; c];
size(c)
w2 = reshape(c',[],1);
size(w2);

% Using the LFSR bits to address the pixels

Z1 = zeros(320000,7);
Z2 = zeros(320000,7);

for i1 = 1:320000
    Z1(i1,:) = Z1(i1,:)+(c(i1,1:7));
    Z2(i1,:) = Z2(i1,:)+(c(i1,8:14));
end

UNI = ones(320000,1);
P1 = bi2de(Z1)+UNI;
P2 = bi2de(Z2)+UNI;

% Finding the intensities of pixels

Int = zeros(320000,1);
for i2 = 1:320000
    Int(i2,:) = Int(i2,:) + f1(P1(i2),P2(i2));
end

size(Int)
w2 = reshape(Int',[],1);
size(w2)

% Performing the XOR operation

ci = zeros(320000,1);
for i1 = 1:320000
    ci(i1,:) = xor(w1(i1,1),w2(i1,:));
end

% Regenerating image

w4 = reshape(ci',[],8);
NI2 = bi2de(w4,'left-msb');
n = 200;
for l = 1:200
    u4(l,:) = NI2(((l-1)*n)+1:l*n);
end
u4 = u4';
figure,imshow(u4,[])
impixelinfo;
figure,imhist(u4);
E2 = entropy(u4)
u6 = u4;

% DECRYPTING THE IMAGE
% LINEAR SHIFT REGISTERS OUTPUT

sw = [1 0 1 0 1 0 1 0 1 0 1 0 1 1];
%sw = [1 0 1 0 1 0 1 0 1 0 1 0 1 0];
tw = [3,4,5,14];
nw=length(sw);
cw(1,:) = sw;
mw=length(tw);
bw = zeros(1,mw-1);
kw = 1;
while (kw < 2^nw-1)
    bw(1) = xor(sw(tw(1)), sw(tw(2)));
if mw>2
    for iw=1:mw-2
bw(iw+1)=xor(sw(tw(iw+2)), bw(iw));
end
end
jw=1:nw-1;
sw(nw+1-jw)=sw(nw-jw);
sw(1w)=bw(mw-1);
cw(kw+1,:)=sw;
kw = kw+1;
end

for kw=1:2^nw-2
bw(1)=xor(sw(tw(1)), sw(tw(2)));
if mw>2
    for iw=1:mw-2
        bw(iw+1)=xor(sw(tw(iw+2)), bw(iw));
    end
end
jw=1:nw-1;
sw(nw+1-jw)=sw(nw-jw);
sw(1w)=bw(mw-1);
cw(kw+1,:)=sw;
end

Seqw = cw(:,1)';
Seqw = Seqw';
cw = [cw ; sw];
cw = [cw; cw; cw; cw; cw; cw; cw; cw; cw; cw; cw; cw; cw; cw; cw; cw];

% Using the LFSR bits to address the pixels

Zw1 = zeros(320000,7);
Zw2 = zeros(320000,7);

for iw1 = 1:320000
    Zw1(iw1,:) = Zw1(iw1,:)+(cw(iw1,1:7));
    Zw2(iw1,:) = Zw2(iw1,:)+(cw(iw1,8:14));
end

Pw1 = bi2de(Zw1) + 1;
Pw2 = bi2de(Zw2) + 1;
% Finding the intensities of pixels

Intw = zeros(320000,1);

for iw2 = 1:320000
    Intw(iw2,:) = Intw(iw2,:)+f1(Pw1(iw2),Pw2(iw2));
end

% Selecting Adjacent Block

%% Intw(iw2,:) = Intw(iw2,:)+f1w(Pw1(iw2),Pw2(iw2));

ww2 = reshape(Intw',[],1);

% Performing the XOR operation

ciw = zeros(320000,1);
for iw1 = 1:320000
    ciw(iw1,:) = xor(ci(iw1,1),ww2(iw1,:));
end

% Regenerating image

ww4 = reshape(ciw,8,[]);
N12w = bi2de(ww4(:,),'left-msb');
lw = 200;
for nw = 1:200
    u4(nw,:) = N12w(((nw-1)*lw)+1:nw*lw);
end
u4 = u4';
figure,imshow(u4,[])
impixelinfo;
figure,imhist(u4);
E3 = entropy(u4)
toc;