Academia International College

Tribhuvan University

Institute of Science and Technology

Project Report

On

"Adi Shamir's visual secret sharing and its comparison with Ceaser cipher Algorithm for image encryption"

Submitted To:

Department of Computer Science and Information Technology

Academia International College

In partial fulfillment of the requirement for Bachelor Degree in Computer Science and Information Technology

Submitted By:

Nikhil Maharjan (10829/073)

Nirmal Maharjan (10830/073)

Sachin Maharjan (10836/073)

Sujan Raj Tuladhar (10845/073)

APRIL 2021

ACKNOWLEDGEMENT

We would like to express our sincere gratitude towards to our project mentor Mr. Krishna Bikram Shah for guiding and helping us to complete project in the topic “Adi Shamir’s visual secret sharing and its comparison with Ceaser cipher Algorithm for image encryption”. From him we had gained lot of benefits from his experience in the computer science and information technology field. We would also like to express our gratitude towards our college’s program coordinator Ms. Shristi Awlae for support and encouragement during the project duration.

Nikhil Maharjan (10829/073)
Nirmal Maharjan (10830/073)
Sachin Maharjan (10836/073)
Sujan Raj Tuladhar (10845/073)

ABSTRACT

In this paper, we have studied the Adi Shamir's visual secret sharing scheme which is different from conventional cryptography, visual cryptography is an image cryptographic technique proposed by Naor and Shamir. Visual cryptography schemes allow the encoding of a secret image into shares, which is distributed to the participants. The requirement for minimum threshold shares ultimately leads to the confidentially of secret. (k, n) visual secret sharing uses threshold scheme by using the concept of Lagrange's polynomial interpolation. Also, we have compared it with the Caesar Cipher encryption with image. Then we have calculated the efficiency of the encryption in terms of the NPCR and UACI. In Caesar cipher we have key which is used to encrypt and decrypt the image color pixel values. We had the existing pixel value with the key for encryption and subtract the encrypted pixel values with key for decryption again. From our study average value of NPCR for Shamir's Secret Share is 99.9993. Also, the average NPCR value of the Caesar cipher algorithm is 100. Similarly the average value of UACI for Shamir's Secret Share is 65.9841. Also, average UACI value of the Caesar cipher algorithm is 97.66. Hence based on NPCR and UACI the Caesar cipher seems to be better algorithm but the key used in the encryption can be guessed easily. Since, shares are distributed and threshold shares are necessary to reveal original image it is visual secret sharing is quite stronger algorithm in image encryption.

Keywords: *cryptography, (k, n) visual secret sharing, Caesar Cipher algorithm*

TABLE OF CONTENTS

LIST OF FIGURE

LIST OF TABLE

ABBREVIATION

NPCR	Number of Pixel change Rate
UACI	Unified Average Change Intensity
VSS	Visual Secret Sharing

1. INTRODUCTION

1.1 Visual Secret Sharing

Cryptography is a method of securing information by using codes so that only those who are supposed to read and process the information can do so. There are four primary goals of cryptography. Confidentiality, integrity, non-repudiation, and authentication are among them. Confidentiality means that the information cannot be understood by anyone other than the intended recipient. If information cannot be altered during storage or transit between sender and intended receiver, we can say that information integrity has been maintained. Non-repudiation means that the creator/sender of the information cannot later deny his or her intentions in creating or transmitting the information. Authentication is maintained when the sender and receiver can confirm each other's identities and the origin/destination of the information.

Till now lots of the cryptographic techniques and algorithms has developed such as AES (Advanced Encryption Standard), RSA (Rivest, Shamir, Adelman), Diffie-Hellman, DES (Data Encryption Standard), Blow Fish, MD5, SHA and many other. But these all algorithms depends upon the secret key. Imagine we encrypt our important files with one secret key and if such a key is lost then all the important files will be inaccessible. Keyed cryptographic techniques provide various way to protect secret information but most of them requires highly complex encryption and decryption. Thus, secure and efficient key management mechanisms are required. Instead of providing secret key for one individual it is better idea to distribute secret key among the multiple intended person so we can prevent damage that can happen if the key is lost or prevent any misuses. Secret sharing scheme is one of them that split the secret into several parts and distribute them among selected parties. The secret can be recovered once these parties collaborate in some way [1]. Thus confidentiality can be achieve without complex computation.

Visual cryptography is a cryptographic technique which allows visual information (pictures, images etc.) to be encrypted in such a way that the decryption can be performed by the human visual system. In this algorithm the given data is portioned into multiple shares. Then we define minimum threshold number of shares which when gathered form shared participants can reveal the secret. Suppose the data 'D' is divided into N shares. Then 'D' can be constructed from 'K' shares out of 'N'. Complete knowledge of k-1 shares reveals no information about 'D'. 'K' of 'N' shares is necessary to reveal secret data. Visual cryptography was pioneered by Moni Naor and Adi Shamir in 1994 at Eurocrypt. Visual cryptographic schemes are used to secure visual data based on authentication techniques that contain sensitive data such as military surveillance, satellite images, medical records etc. Visual cryptographic scheme also provide secure way to transfer images on the internet [2].

Caesar cipher we have key which is used to encrypt and decrypt the image color pixel values. We add the existing pixel value with the key for encryption and subtract the encrypted pixel values with key for decryption again.

1.2. Problem Statement

The conventional visual cryptography model is focused on splitting a secret image into several random shares, each of which reveals no details about the secret image other than its size. By superimposing the threshold number of shares, the hidden picture can be reconstructed.

But in Adi Shamir's visual cryptographic scheme there is use of geometrical and mathematical functions that provides the secure way to transfer the secret data on the internet. To maintain confidentiality of the encrypted image it provides shares available only to selected people. Shamir's (k, n) visual secret sharing uses threshold scheme by using the concept of Lagrange's polynomial interpolation. The original image can be generated without loss and without complex mathematical calculation. Also there are lot of drawbacks in the traditional visual secret sharing schemes. Some of them are listed below:

- **Much time consumption for encryption and decryption:** Traditional visual secret sharing schemes like Region Incremental Visual Cryptography (RIVC) divide images into regions, resulting in high protection. But as algorithm is applies to all the separate regions of the image. So it takes huge memory and takes lot of encryption and decryption time [3]

- **The decrypted images has low quality:** In a basic 2-out-of-2 scheme 2 share images are produced from an original image and share must be stacked to reproduce the original image. In this scheme original pixel is changed into 2*2 block of sub Pixels. As shown in Table 1 if the original pixel is white it can be combination of the six combinations of share pixels that is randomly created. Similar, the possible share combination for black pixels is also shown. The original hidden image will be exposed after piling the shares with white transparent and black opaque. The resulting sharing images have four times the amount of pixels as the original image (since each pixel of the original image was mapped to four sub pixel). So the recovered image has degraded visual quality [4].

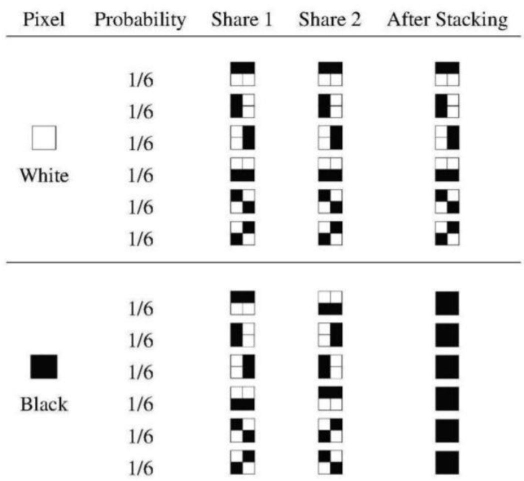

*Figure 1: Pixel combinations in 2*2 schemes*

- **The overlapping had to be done correctly:** In traditional visual secret sharing schemes the share images must be overlapped correctly to reveal the secret which is quite a difficult task. But in Shamir's visual cryptographic schemes we use Lagrange's interpolation formula for reconstruction of the secret image.

1.3 Objectives

The main objectives of research project are:

- To understand how the share construction and secret image reconstruction happens in the Naor Shamir’s visual cryptographic scheme.
- To implement the visual secret sharing by using Naor Shamir’s scheme and study the performances.
- To compute Adi Shamir’s Visual Secret Sharing algorithm and Ceaser Cipher Algorithm and provide conclusion based on result obtained.

1.4 Scope and Limitation

1.4.1 Scope of project

Visual secret Sharing can be used extensively in areas where very confidential data are need to be stored and transferred over insecure network such as internet in a authenticated way. The visual cryptographic techniques can be used in crucial data such as military surveillance, satellite images, medical records, financial transactions, maps, cloud data, biometrics, watermarking etc.

1.4.2 Limitations of the system

Although visual cryptography provides security and confidentiality, it has also got some flaws. There are many things that are not considered in secret sharing. The following are some limitations of study for this system.

- The decryption of the image totally depends upon human visual system. Therefore people with eye problem (blind people) cannot use it.
- The visual secret sharing involves sharing of encrypted images to all participants. So, it is difficult to reveal original message if the participants live in different geographical area.

2. METHODOLOGY

Visual cryptography is one of the strongest cryptographic schemes since it embodies both absolute confidentiality and a very easy method for decrypting the secret without requiring complex mathematical computations.

Implementing the Naor Shamir secret sharing algorithm and the Caesar Cipher algorithm for image encryption is part of the technique. Standard photos make up the hidden and cover images. We first extract the R, G, and B values of the image pixels in order to share the secret with 'n' participants. The Shamir (k, n) visual secret sharing algorithm is then used.

2.1 Literature Review

The 2 out of 2 sharing scheme is a well-known encryption algorithm. This technique creates two binary image shares from a single original image. This algorithm is absolutely stable, and it can be decoded by the human visual system directly. The black and white pixels are expanded with various transparency options. By stacking two hidden shares with eac, the original image is shown.There is noise in the produced shares. As a consequence, the decoded image may not be identical to the original [2].

Each pixel of a colored image is extended into a 2*2 block to create secret shares in the visual cryptography technique for colored images. Red, blue, green, and alpha color values are expressed in each 2*2 block on the shares. As a result, when we combine these color values, the original image will remain fully private. Since the human eye is unable to perceive slight variations in pixel color values, decrypted images tend to be similar to secret image [3].

In 1998, a groundbreaking new approach to image cryptography was implemented. In this case, instead of using one hidden image, we use two. Two share photos, s1 and s2, are generated using this algorithm.When these s1 and s2 are stacked by rotating s1 by 90 degrees, we get the first original image, and if we use anticlockwise 90 degrees, we get the second original image [4].

Since there is no pixel expansion in the RGB shuffling process (proposed in 2013), the pixel bits do not shift during encoding and decoding [5]. To construct the cipher image, we use translation reshaping and concatenation with the RGB values of the original image. The algorithmic method is used to reshape and concatenate the translations.

Visual cryptography was first proposed in 2006 by Zhi zhou, Gonzalo, R Arc, and Govanni D [6]. Each binary pixel is decrypted into an array of n-shared subpixels using this algorithm. Halftone cells are the name for these subpixels. Significant visual detail, such as landscape construction, is contained in the halftone shares. When working with high-quality images, such as electronic cash and watermarking, this algorithm can be used.

As more shares are stacked increasingly in Jin, W.Q. Yan, and M.S. Kankanhalli's progressive visual secret sharing scheme, the recovery of the secret picture becomes clearer and clearer [7]. When only a few shares are staked, only a sketch will be revealed; when more shares are staked, more information will be revealed.

Young-chang Hou invented an additive and subtractive model that is widely used to explain color structure [3]. In the additive mode, three primary colors are used: red, red, green, and blue (R, G, B). Having these RGB components yields all of the desired colors. The strength of red/green/blue in the compound light can be modified for calculation. The additive model is well-exemplified by television. Color is expressed in the subtractive model by adding combinations of colored-lights reflected from an object's surface. This model produces a wide variety of colors by combining Cyan(c), Magenta(m), and Yellow(y) color components. A good example of a subtraction model is a color printer. Hou's style is based on the halftone technique and color decomposition. Halftoning is the method of producing illusions of different shades by using pixel patterns of varying size and color. The color hidden shades are used in this process. The color hidden image is decomposed into three different pictures, each of which is cyan(c), magenta(m), and yellow (y). The halftone technique is used to convert the three-color images into halftone images. Finally, a color halftone image containing the three halftone images is produced.

Proactive secret sharing scheme that regularly renews the shares of all users in a network, making any share compromised by an adversary useless from one time span to the next. In [8], initially all the users obtain their secret shares by any traditional threshold secret sharing. To renew a secret share a set of polynomials, having their free coefficients equal to 0 (i.e. $\delta(0) = 0$ where q is the polynomial) is used. Each such polynomial is sent to any user, by one of the other users. This scheme relies on all members of a party renewing each other's shares.

2.2 Adi Shamir's visual secret Sharing

We construct the n shares using polynomial equation of degree k-1 where k is a threshold. Similar, we reconstruct original image using LaGrange's interpolation equation.

We take inputs n as total shares and k as minimum number of require threshold to recover secret. Then generate polynomial equation of degree (k-1) as $q(x) = a_0+a_1x+ \ldots +a_{k-1}x_{k-1}$ mod p, where a0 = pixel value and $a_0, a_1, \ldots, a_{k-1}$ are coefficients < p, p is large prime.

Then for decryption, Apply Lagrange's Interpolation with minimum required threshold shares as:

$$Y = q(x) = \sum_{i=0}^{n} yi\ \ li\ (x) \bmod p \text{ where } l_i\ (x) = \prod_{j=0\ i\neq j}^{n} \frac{(x-xi)}{(xi-xj)}$$

2.3 Caesar cipher algorithm for image encryption

In Caesar cipher algorithm for encryption we used to add the key to the color pixel value of the original image.

Mathematically,

$I_e = I_o + key$

Similarly, for decryption we subtract the color pixel of encrypted image with the key.

Mathematically,

$I_d = I_e - key$

3. REQUIREMENT ANALYSIS

Software requirement is a functional or non-functional need to be implemented in system. Functional means providing particular service to the user. For example, in context to banking application the functional requirement will be when customer selects “View Balance” they must be able to look at their latest account balance. Software requirement can also be a non-functional, it can be performance requirement. For example, a non-functional requirement is where every page of the system should be visible to the user within 5 seconds.

3.1 Functional Requirements

Functional requirement means providing particular service to user. We have implement our comparison system in python programming language using jupyter notebook. Also we use anaconda environment which automatically provides necessary dependencies to run our project. In our project the main requirement of our system is when user defines shares and threshold the system should able to generate the decrypted images which is uploaded by user. Also when user upload threshold amount of the decrypted image, the original image should be produced.

3.2 Non Functional Requirements

Non-functional requirements specifies the quality attributes of a software system. They judge the software system based on responsiveness, usability, security, portability and other non-functional standards that are critical to success of the software system. Some of the non-functional requirements of our project are:

- Encryption and decryption process should be fast as much possible.
- The system should be secured.
- Other user should not know encryption details such as encryption image, threshold and number of participant involved.
- Every unsuccessful attempt by a user to decrypt image should be recorded for security purpose

- The application should be easy to use. Even non-technical people should easily use it.

4. SYSTEM DESIGN

4.1 System Architecture

We construct the n shares using polynomial equation of degree k-1 where k is a threshold. Similar, we reconstruct original image using LaGrange's interpolation equation.

Generation

Step 1: Take inputs n as total shares and k as minimum number of require threshold to recover secret.

Step 2: Generate polynomial equation of degree (k-1) as q (x) = $a_0+a_1x+ \ldots +a_{k-1}x_{k-1}$ mod p, where a0 = pixel value and $a_0, a_1, \ldots, a_{k-1}$ are coefficients < p, p is large prime.

Reconstruction

Step 1: Apply Lagrange's Interpolation with minimum required threshold shares as:

Y = q (x) = $\sum_{i=0}^{n} yi \;\; li\ (x)$ mod p where l_i (x) = $\prod_{j=0\ i\neq j}^{n} \frac{(x-xi)}{(xi-xj)}$

4.1.1 Process Flow Chart:

In Adi Shamir’ algorithm we first input secret image in the system. Then we divide the image into ‘n’ shares, where n is number of participants.

Similarly, for decryption (image reconstruction) we input ‘k’ number of shares in the algorithm to get original image.

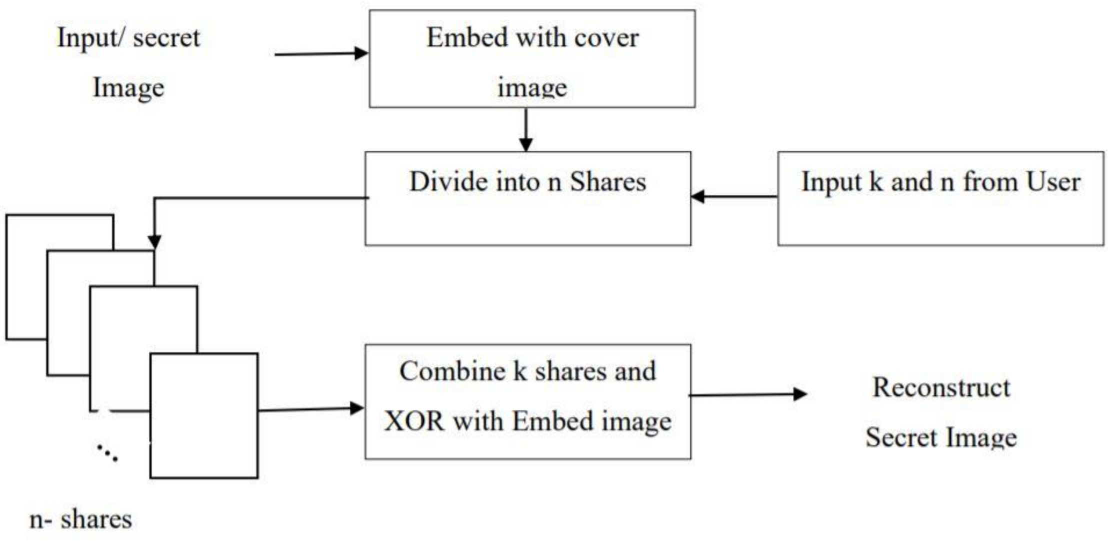

Figure 2: Shamir's Visual Secret Sharing Scheme

4.1.2 Flow chart of share generation in Adi Shamir's VSS

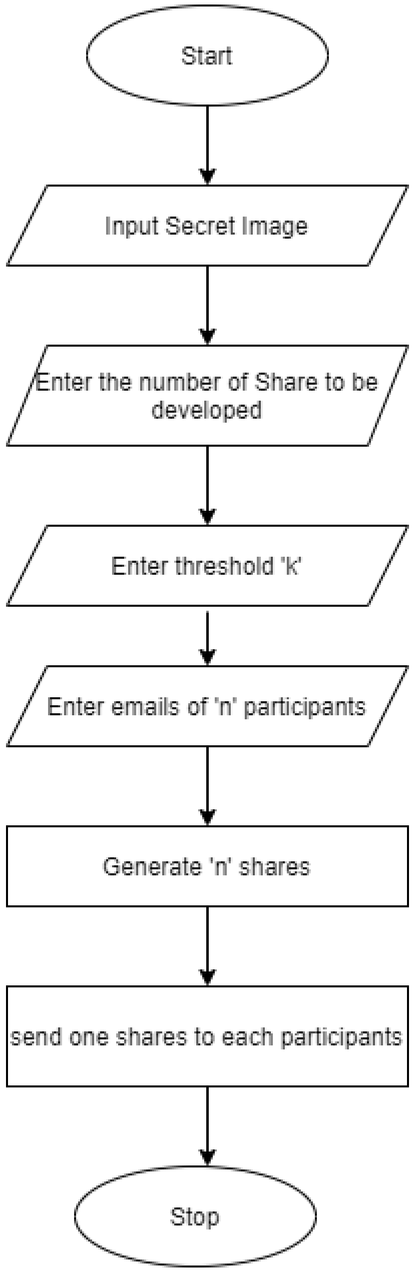

Figure 3: Flow chart of encryption of secret image in proposed system

4.1.3 Flow chart of decryption of image in Adi Shamir's VSS

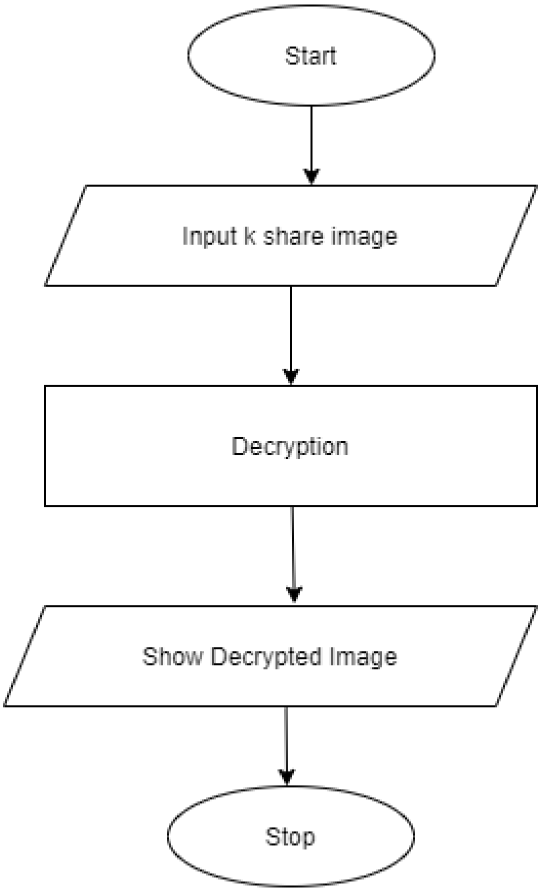

Figure 4: Flow chart of decryption of secret image in proposed system

4.2 Approaches of Algorithm Comparison

NPCR

NPCR stands for Number of Pixel Change Rate. Human eye cannot separate tiny change on the light intensity of the image pixel. Even though two different image looks same does not mean their pixel intensity are also same. Even small number of bit changes on key does not reveal the original secret. So, to distinguish how many pixels are changed from original image with shared images number of pixel change rate(NPCR) is used. It gives the total number of different pixels that are different from original images' pixels.

Mathematically, NPCR is defined as:

NPCR = $\sum_{i=1}^{M}\sum_{j=1}^{N}\frac{D(i,j)}{T} * 100\%$

Where D(i, j) = $\begin{cases} 0, if\ Io(i,j) = Ishare(i,j) \\ 1,\ ifIo(i,j) \neq Ishare(i,j) \end{cases}$

And T=M*N is total number of pixels.

The NPCR measures the percentage of different pixel numbers between original images with the shares image. A high NPCR value is interpreted as high resistance to differential attacks.

UACI

Unified Average Change Intensity (NPCR) determines the average intensity of differences between the two images. It is better to have a high UACI value to be considered as good results.

Mathematically UACI is defined as

UACI = $[\sum_{i=1}^{M}\sum_{j=1}^{N}\frac{|Io(i,j) - Ishare(i,j)|}{F}] * \frac{100\%}{T*C}$

Where $T=M*N$ is the total number of pixels. F is the largest supported pixel value and C is the total number of color component of image.

5. IMPLEMENTATION AND TESTING

5.1 Implementation Tools

All the implementation is done in python programming language using Visual Studio Code text editor. For converting the image to matrix, we used the Pillow and Numpy. Furthermore, for sending share image to the participants we used the django SMTP backend. We also have made simple system to demonstrate how to use visual secret sharing can be used in the commercial apps for authentication.

Also we used to show quick algorithm implementation using python in jupyter notebook. Jupyter notebook is a interactive python IDE which runs on the browser. Jupyter notebook is open source and part of ipython (interactive python) which is mostly used by data scientist and AI researchers.

After implementing Adi Shamir's' visual secret sharing algorithm and the Caesar cipher algorithm for image encryption, we perform the NPCR and UACI test.

5.1.1 Tools and Technology

Backend programming language: python 3

Python is a general-purpose, interpreted, high-level programming language. Python allows programmers to use different programming styles to create simple or complex programs get quicker results and write code almost as if speaking in a human language.

Modules and Framework

Django

Django is an open-source python web framework used for rapid development, pragmatic, maintainable, clean design, and secures websites. A web application framework is a toolkit of all components need for application development. The main goal of the django framework is to allow developers to focus on components of the application that are new instead of spending time on already developed components.

Numpy

Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.It is the fundamental package for scientific computing with python.It consists of many different features.

Some of them are listed below:

- Powerful N-dimensional array object.
- Sophisticated (broadcasting) functions.
- Tools for integrating C/C++ and FORTRAN code.
- Useful for calculating linear algebra, Fourier transforms, and random number capabilities.

Pillow

Pillow is a python Imaging Library (PIL), which adds support for opening, manipulating and saving images. The current version identifies and reads a large number of formats.

5.2 Testing

Testing is the process of evaluating a system or its component(s) with the intent to find whether it satisfies the specified requirements or not. Testing is executing a system in order to identify any gaps, errors, or missing requirements in contrary to the actual requirements.

An early start to testing reduces the cost and time to rework and produce error free software that is delivered to the client. However, in Software Development Life Cycle (SDLC), testing can be started from the Requirements Gathering phase and continued till the deployment of the software. It also depends on the development model that is being used.

5.2.1 Objectives of Testing

The objectives of testing are:

- Testing is a process of executing a program with the intent of finding errors.
- A Successful test case is one that uncovers an as-yet-undiscovered error**.**

5.2.2 Purpose of testing

System testing is a stage of implementation, which is aimed at ensuring that the system works accurately and efficiently as per the user need before the live operation commences. As stated, before testing is vital to the success of a system. System testing makes a logical assumption that if all parts of the as system are correct, the goal will be successfully achieved. A series of tests are performed before the system is ready for the user acceptance test.

5.2.3 Testing Image to Matrix Transform

Testing for the image to NumPy array are done in following ways to visualize if the image is correctly transformed in to required matrix form or not**:**

```
from PIL import Image
from numpy import array, size

img = Image.open('test.jpg')
image_as_array = array(img)

print(image_as_array)
```

```
[[[238  28  37]
  [238  28  37]
  [238  28  37]
  ...
  [238  28  37]
  [238  28  37]
  [238  28  37]]
```

Figure 5: Image to Matrix

5.2.4 Testing the encryption of color pixel value

During share generation we should determine that each individual color value of each pixel are generated according to LaGrange interpolation or not:

```
import math
list_of_cofficient = [3,5,7,11,13,17,19,23]
pixel = 5 #suppose value of R value of pixel is 5
threshold = 4  #suppose there are 5 participants
epixels = list() # for storeing encrypted images
for i in range(threshold):
    temp  = pixel
    for j in range(1,threshold):
        temp = temp + list_of_cofficient[j-1] * int(math.pow(i+1,j))
    epixels.append(temp)
print(epixels)
```

```
[20, 87, 248, 545]
```

Figure 6: Encryption of color pixel

5.2.5 Testing the decryption of color value

As in share generation we should also check that original value is correctly generated from shares or not.

```
epixel = [20, 87, 248, 545] # value of R of the 4 pixles
threshold = 4
dpixel = 0
for i in range(1,threshold+1):
    li = 1
    for j in range(1,threshold+1):
        if i != j:
            li = li * (-j/(i-j))
    dpixel = dpixel + epixel[i-1] * li

print(dpixel)
```

```
5.0
```

Figure 7: Decryption of color value

6. RESULT OBTAINED AND CONCLUSION

6.1 Result Obtained

The result of NPCR obtained from the both algorithm area:

Table 1: NPCR Result

S.No	**(Threshold(k), Total Shares(n) and p=257)**	**Shares**	**NPCR (%)**	
			Visual secret Sharing	**Caesar Cipher**
1	(2,3)	1	100	100
		2	100	
		3	100	
2	(3,5)	1	100	
		2	100	
		3	100	
		4	99.9967	
		5	100	
3	(4,6)	1	100	
		2	99.9943	
		3	100	
		4	100	
		5	99.9943	
		6	100	
4	(7,7)	1	100	
		2	100	
		3	100	
		4	100	
		5	100	
		6	100	
		7	100	
Average			99.9993	100

The average value of NPCR for Visual Secret Sharing scheme is 99.9993 whereas the NPCR of the same image with Caesar Cipher Algorithm is 100. Both algorithms generate good result. But in case of NPCR the result of Caesar Cipher is quite better.

The result of UACI obtained from the both algorithm area:

Table 2: UACI Results

S.No	**(Threshold(k), Total Shares(n) and p=257)**	**Shares**	**UACI (%)**	
			Visual secret Sharing	**Caesar Cipher**
1	(2,3)	1	98.4435	97.66
		2	97.2726	
		3	96.1089	
2	(3,5)	1	96.4980	
		2	89.4941	
		3	78.5990	
		4	63.8132	
		5	45.1361	
3	(4,6)	1	93.774	
		2	67.7042	
		3	5.0583	
		4	88.7959	
		5	3.50192	
		6	31.9066	
4	(7,7)	1	77.8210	
		2	11.6731	
		3	81.7120	
		4	88.7959	
		5	88.7959	
		6	50.5836	
		7	30.3501	
Average			65.9841	97.66

The average value of UACI of secret shares using visual secret sharing is 65.9841. For same image the UACI of Caesar Cipher is 97.66. Clearly Ceaser Cipher has good result in case of UACI test.

The execution time of visual secret sharing algorithm and caser cipher algorithms according to the total number of shares and threshold are given below:

Table 3: Time consumption to share generation and image reconstruction

S.No.	**(Threshold, participants)**	**Share generation time in microseconds**	**Shares reconstruction time in microseconds**
1	(2,3)	811756	536379
2	(3,5)	390484	21839
3	(4,6)	719020	782960
4	(7,7)	862300	880284

6.2 Conclusion

The project presents the approach to demonstrate how the visual secret can be applied in the modern applications and also comparison of the visual secret sharing with the Caesar Cipher algorithm for image encryption. We found that there are different types of image encryption technology and one of the most popular one is visual secret sharing. Similarly the Caesar Cipher algorithm is one of the primitive algorithm for message encryption. In addition, there are many other algorithms for share generations. We find that many scholars in the different universities have done more research in the field of image encryption techniques. The main goal is to discover effective and efficient algorithm for image encryption that can be used to secure digital medias such as image specially. The finding highly secure which cannot be rebuild easily by intruders is the ultimate goal of this project.

We have computed the NPCR and UACI of the both algorithm. The value of NPCR and UACI lies between 0 to 100. The higher the NPCR and UACI value higher will be the secured. In terms of the NPCR and UACI the Ceaser Cipher algorithm for image encryption has the best result. We conclude that NPCR and UACI only define the difference between original image and encrypted image. So by using different techniques it is very difficult to generate old original image. Also we have to consider that the Cesar Cipher algorithm use only one key. So guessing the key will be easy and if the any intruder gets the key then he/she can easily decrypt the encrypted image and get original image.

Although NPCR and UACI value of Visual secret sharing is not so good. It is the strong technique because we need threshold number of shares to decrypt. So, we conclude this project by recommending the Visual Secret Sharing scheme is secure algorithm although it is weak in terms of NPCR an UACI.

6.3 Future Work

The project is about demonstrating how visual secret share can be used in the part of authentication in the different types of modern application. And also show comparison it with simple Caesar cipher algorithm for image encryption.

During our project we have problem to get correct result while using JPG images. We only get expected result in PNG format of image. So, we need extra time for researching the different image compression formats and relation with visual secret sharing.

References

[1] A. D. B. a. A. D. Santis, "Secret Sharing and Visual cryptography schemes," Dipartimento di Informatica ed Applicazioni, Università di Salerno,, 84081 Baronissi (SA), Italy.

[2] M. N. A. Shamir, "Visual Cryptography," Department of Applied Math and Computer Science, Weizmanu Institute, Rehovot, 1995.

[3] M. Quist-Aphetsi Kester, "A cryptographic Image Encryption technique," *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET),* Vols. Volume 2,, no. Issue 2,, ,January 2013.

[4] D. K. S. Ashish Sharma, "A Comprehensive View on Encryption Techniques of Visual," *International Journal of Recent Research and Review,* Vols. , Vol. VII,, no. Issue 2,, June 2014.

[5] V. V. T. R. J. Tamilarasi, "Improving Image Quality In Extended Visual," *INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY RESEARCH,* Vols. VOLUME 3, , no. ISSUE 4,, APRIL 2014.

[6] M. I. G. R. A. F. I. a. G. D. C. Zhi Zhou, "Halftone Visual Cryptography," AUGUST 2006.

[7] Z.-Y. Q. a. C.-F. T. Young-Chang Hou1, "Progressive Visual Cryptography with Friendly and size invariant shares," The International Arab Journal of Information Technology, Tamkang, Tiwan, March 2018.

[8] Y.-C. Hou∗, "Visual cryptography for color images," pergamon, Jung Li, Taiwan, Received 6 June 2002; accepted 26 August 2002.

[9] S. J. H. k. a. M. y. Amir hergberg, "Proactive secreat sharing or how to cope with perpetual leakage," IBM TJ Watson Research Center, York Town Heights, New York.

Appendix A

1. Gantt chart

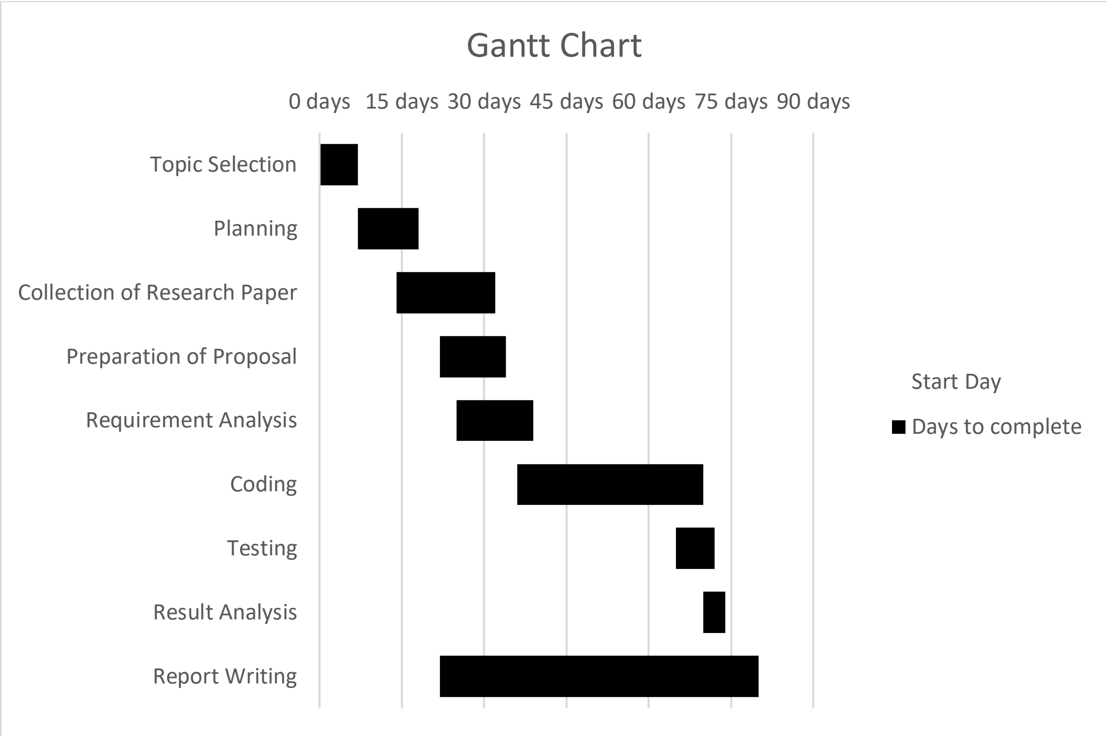

Figure 8: Gantt Chart

Appendix B

1. Screenshots

1. Home View

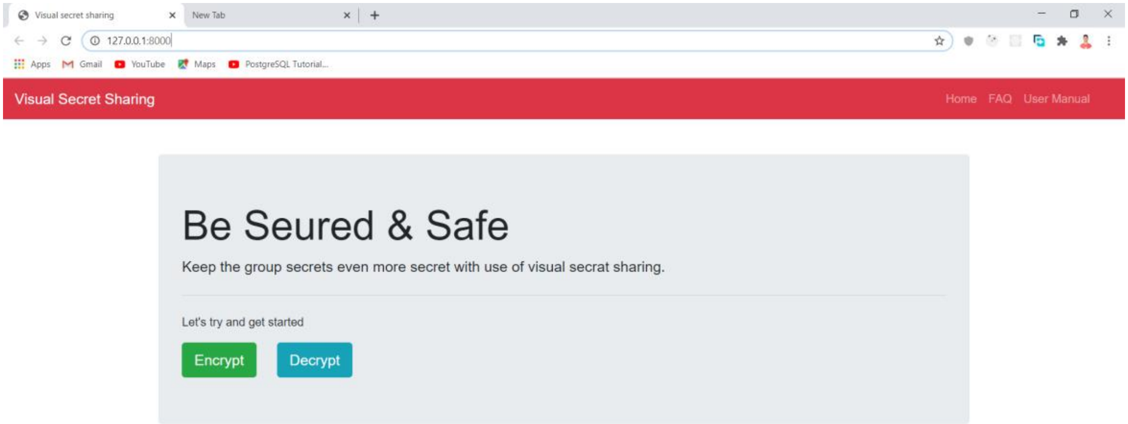

Figure 9: Home View

2. Encryption Form

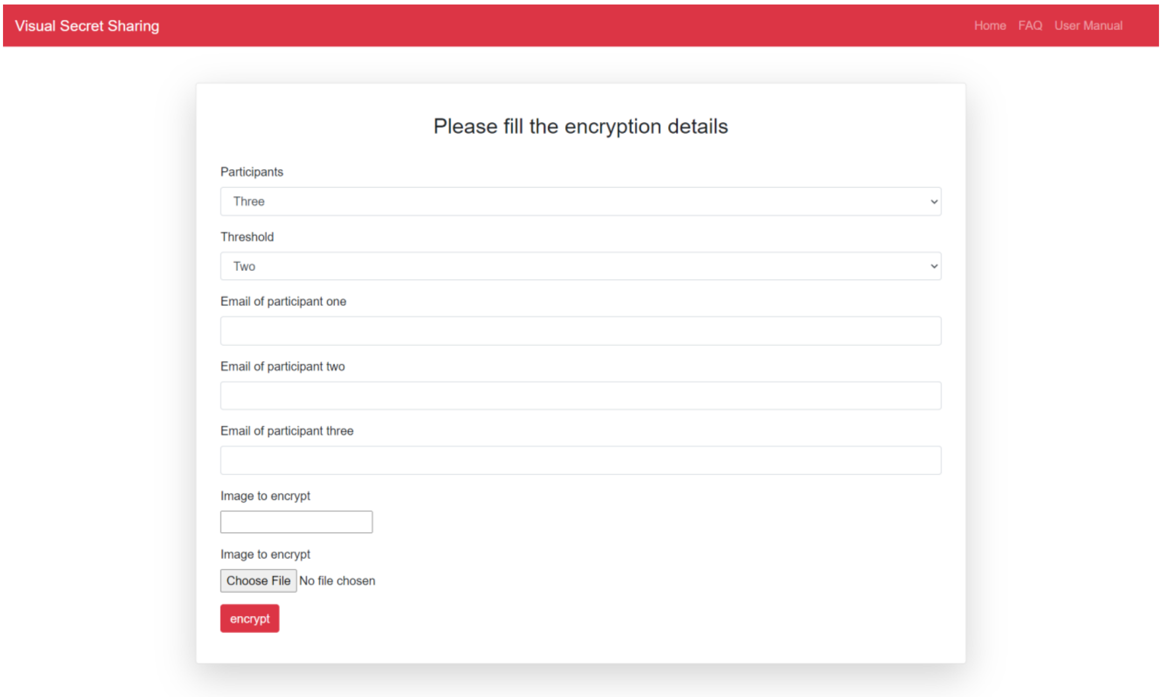

Figure 10: Encryption Form

3. Share image in the Gmail

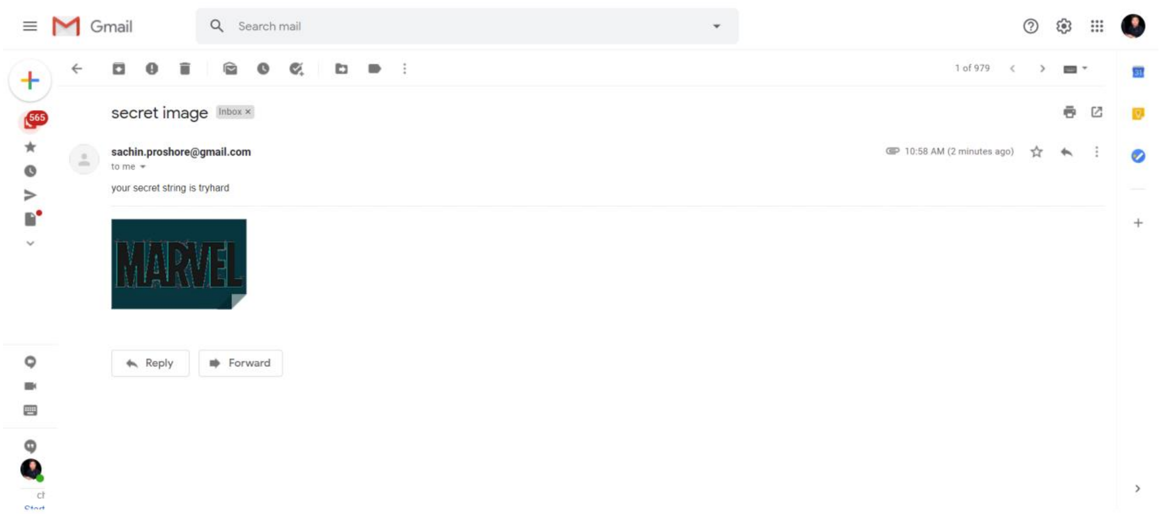

Figure 11: Share image in Gmail

4. Decryption verification

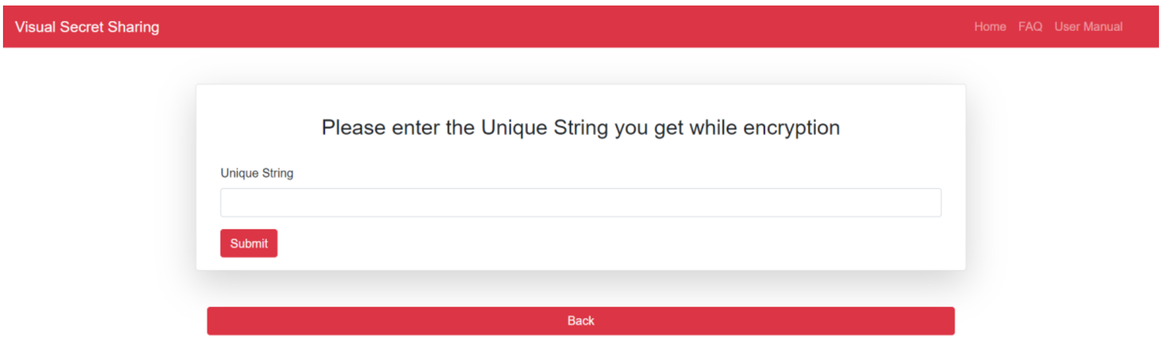

Figure 12: Decryption Verification

5. Decryption Form

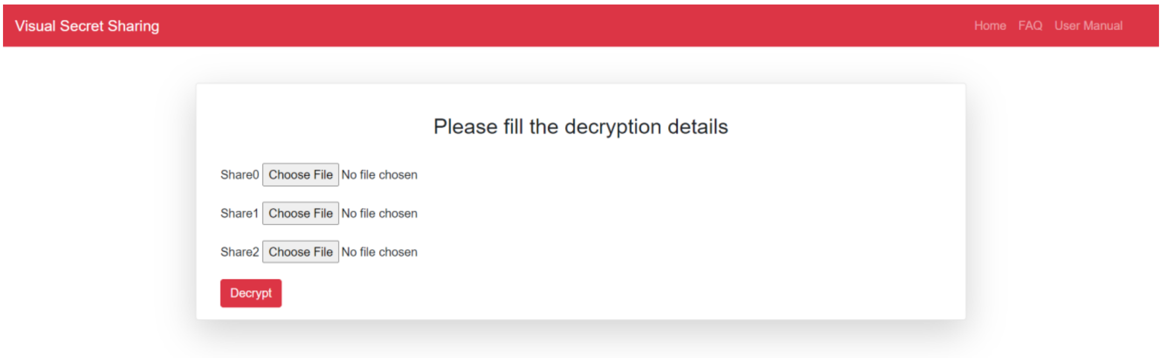

Figure 13: Decryption Form

6. Decryption Result

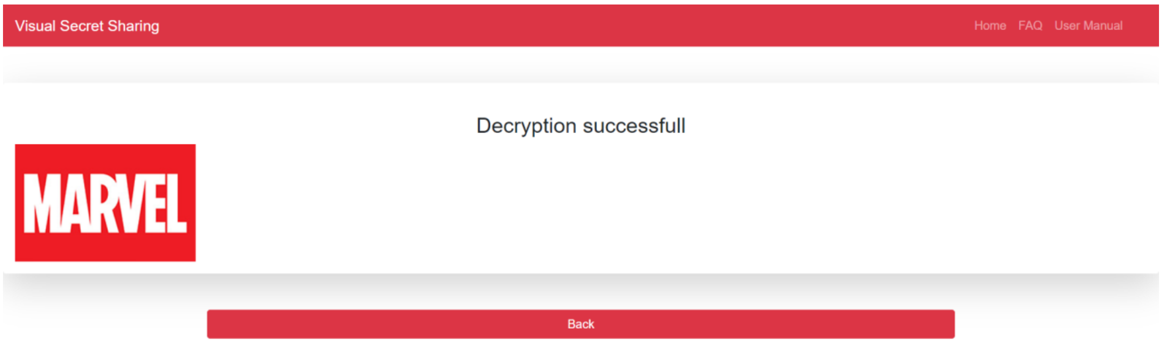

Figure 14: Decryption Result

7. Shamir's Secret Share Scheme (n=4, k=6)

Figure 15: Original Secret Image

Figure 16: Share 1

Figure 17: Share 2

Figure 18: Share 3

Figure 19: Share 4

8. Encryption using Caesar Cipher (key=5)

Figure 20: Original Image

Figure 21: Encrypted Image

Appendix C

Source code

Code for visual secret Sharing

```
# Required Modules
from PIL import Image
from NumPy import array, size
import math
# Configuration Variables
total_Shares = 4
threshold = 3
list_of_cofficient = [3,5,7,11,13,17,19,23]
# function to chnage image to numpy array (Matrix)
def changeImageToMatrix(imagename):
   img = Image.open(imagename)
   image_as_array = array(img)
   return image_as_array
# function to encrypt the single color value of each pixel
def encryption(pixel):
   epixels = list() # for sotring the shares
   for i in range(total_Shares):
      temp = pixel
      for j in range(1, threshold):
         temp = temp + list_of_cofficient[j-1] * int(math.pow(i+1, j))

      epixels.append(temp % 257)
   return epixels
# function to decryt the images to original image
def decryption(epixel):
   dpixel = 0
   for i in range(1, threshold + 1):
      li = 1
      for j in range (1, threshold + 1):
         if i != j:
            li = li * (-j/(i-j))

      dpixel = dpixel + epixel[i-1] * li
   return dpixel

# code for generating the share
image = changeImageToMatrix('test.jpg')
image = list(image)
encrypted_images = list()
for b in range(total_Shares):
   encrypted_images.append(image)
encrypted_images = array(encrypted_images)

for i in range(len(image)):
   for j in range(len(image[i])):
      for k in range(len(image[i][j])):
         encrypt = encryption(image[i][j][k])
         for b in range(total_Shares):
            encrypted_images[b][i][j][k] = encrypt[b]
# displaying the encrypted image
for b in range(total_Shares):
   encrypted_images[b] = array(encrypted_images[b])
   Image.fromarray(encrypted_images[b], 'RGB').show()
# code for generating the original image from share
decrypted_image = array(image)
for i in range(len(image)):
   for j in range(len(image[i])):
      for k in range(len(image[i][j])):
         row = []
```

```
            for b in range(threshold):
                row.append(encrypted_images[b][i][j][k])
            colorpixel = decryption(row)
        decrypted_image[i][j][k] = colorpixel

Image.fromarray(decrypted_image,"RGB").show()
```

<u>Code for calculationg NPCR and UPCR for VS</u>

<u># code for calculating NPCR</u>

```
m = size(array(image), axis=0)
n = size(array(image), axis=1)

d = 0
for i in range(len(image)):
    for j in range(len(image[i])):
        if list(image[i][j]) != list(encrypted_images[0][i][j]):
            d += 1
npixel_change = (d / (m * n)) * 100
print("NPCR : ",npixel_change)
#code for calculating UACI
m = size(image, axis=0)
n = size(image, axis=1)
original_image = array(image)
total_diff = 0
for i in range(len(image)):
    for j in range(len(image[i])):
        for k in range(len(image[i][j])):
            total_diff = total_diff + (original_image[i][j][k] - encrypted_images[3][i][j][k])
# f= 257, t = M *N, c = 3 s
diff_intensity = (total_diff/257) * (100 / (m * n * 3))
print("UACI : ", diff_intensity)
```

<u>Program to demonstrate ceaser Cipher Algorithm</u>

```
# configurariton variable
key = 5

def cencryption(pixel):
    return (pixel + key)

def cdecryption(pixel):
    return pixel – key
# encryption
image = changeImageToMatrix('test.jpg')
image  = list(image)
encrypted_image = array(image)
for i in range(len(image)):
    for j in range(len(image[i])):
        for k in range(len(image[i][j])):
            encrypted_image[i][j][k] = cencryption(image[i][j][k])
print("Encrypted Image")
Image.fromarray(encrypted_image,"RGB").show()

# decryption
decrypted_image = array(image)
for i in range(len(image)):
    for j in range(len(image[i])):
        for k in range(len(image[i][j])):
            decrypted_image[i][j][k] = cdecryption(encrypted_image[i][j][k])

print("Decrypted Image")
Image.fromarray(decrypted_image,"RGB").show()
```

Calculating NPCR and UACI for Caesar Cipher

```
# NPCR for ceaser cipher
m = size(array(image), axis=0)
n = size(array(image), axis=1)

d = 0
for i in range(len(image)):
   for j in range(len(image[i])):
      if list(image[i][j]) != list(encrypted_image[i][j]):
         d += 1
npixel_change = (d / (m * n)) * 100
print("NPCR : ",npixel_change)
# UACI for Caesar cipher
m = size(decrypted_image, axis=0)
n = size(decrypted_image, axis=1)
original_image = array(image)
total_diff = 0
for i in range(len(image)):
   for j in range(len(image[i])):
      for k in range(len(image[i][j])):
         total_diff = total_diff + (original_image[i][j][k] - encrypted_image[i][j][k])
# f= 256, t = M *N, c = 3 s
diff_intensity = (total_diff/257) * (100 / (m * n * 3))
print("UACI : ", diff_intensity)
```