A Mini Project on

Visual Cryptography

Submitted as part of Image Processing course taken for partial fulfillment of B.Tech (Computer Engineering)

By:

Ishaan Almeida - B007
Hrishikesh Balaji - B010
Aditi Gupta - B035
Parth Jalan - B037

Under the Guidance of
Dr. Dhirendra S Mishra
Professor, Computer Engineering

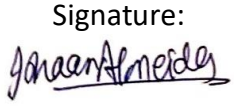
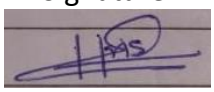
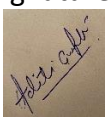
SVKM's NMIMS (Deemed- to be University)
Mukesh Patel School of Technology Management & Engineering
Vile Parle west, Mumbai-56

SVKM's NMIMS Deemed –to be University
Mukesh Patel School of Technology Management and Engineering

Declaration of Originality

We, undersigning students of the MINI Project group hereby declare that we have worked on the mini project titled **"Visual Cryptography"** under **image processing course** taken in semester V of our undergraduate B.Tech. Computer Engineering Program during **Academic year 2020-21**.

All contributions in this project are our own original contributions. To the best of our knowledge and belief this project report contains no material previously published or written by any another person, except where due acknowledgement has been made in the text.

Group Member 1	Group Member 2	Group Member 3	Group Member 4
Signature: 	Signature: 	Signature: 	Signature: 
Name: Ishaan Almeida	Name: Hrishikesh Balaji	Name: Aditi Gupta	Name: Parth Jalan
SAP ID: 70021018144	SAP ID: 70021018007	SAP ID: 70021018033	SAP ID: 70021018038

SVKM's NMIMS Deemed –to be University
Mukesh Patel School of Technology Management and Engineering

Certificate of Completion

This is to declare that **Ishaan Almeida, Hrishikesh Balaji, Aditi Gupta and Parth Jalan** have worked together in a team to complete a **Mini Project** as the part of **Image processing course** in **semester V (Division B) of their B. Tech Computer Engineering Program** during **Academic year 2020-21.**

Their performance in the project has been found to be satisfactory.

Signature of Course in-charge faculty
Dr. Dhirendra S. Mishra

Signature of Examiner
( )

Head of the Department
Dr. Pravin Shrinath

INDEX

Abstract

In today's world data security is one of the major problems we face. Cryptographic schemes are presented for communication to fulfill this need for security. Visual cryptography is a cryptographic technique which allows visual information (pictures, text, etc.) to be encrypted in such a way that the decrypted information appears as a visual image. Through this project we have demonstrated two such techniques:

1. Advance Encryption Standard (AES) algorithm – It is used for text data as well as for image data. In this project an image is given as input to AES encryption algorithm which gives encrypted output. This encrypted output is given as input to AES decryption algorithm and original image is regained as output.
2. Visual Cryptography for Gray-scale Images Using Bit-level – It is an image cryptographic scheme in which a secret image is encrypted into two separate share images. Each share individually reveals no information about the secret, but when shares are superposed the secret is revealed. We use bit-level decomposition to extract binary bit planes from a gray-scale image. Then the bit planes are encrypted and recomposed back as two gray-scale shares. The secret image is revealed when two gray-scale shares are superposed.

Keywords: Visual cryptography, AES, Image encryption, Image decryption, Bit-level decomposition, Gray-scale image, Visual secret sharing.

Introduction

In today's computer generation, data security, hiding and all such activities have become probably the most important aspect for most organizations. These organizations spend millions of their currency to just secure their data. This urgency has risen due to increase in cyber theft/crime. The technology has grown so much that criminals have found multiple ways to perform cyber-crime to which the concerned authorities have either less or not sufficient answer to counter. Hence, the method of Cryptography provides the above answers. One of the most major parts of cryptography is Visual Cryptography. Visual cryptography is a cryptographic technique which allows visual information to be encrypted in such a way that the decrypted information appears as a visual image. It has many usage & application areas like Biometric security, Watermarking, Remote electronic voting, Bank customer identification etc.

Advance Encryption Standard (AES) algorithm: Every encryption and decryption process has two aspects: the algorithm and the key use for the encryption and decryption. However, it is the key used for encryption and decryption that makes the process of cryptography secure. There are two types of cryptographic mechanisms: symmetric key cryptography in which the same key is use for encryption and decryption. In case of asymmetric key cryptography two different keys are used for encryption and decryption. Symmetric key algorithm is much faster and easier to implement and required less processing power as compare to asymmetric key algorithm. AES is a symmetric algorithm. AES algorithm is of three types i.e. AES-128, AES-192 and AES-256. The numbers represent the size of key in bits. This key size determines the security level as the size of key increases the level of security increases. AES-256 has been implemented in this project.

Visual Cryptography for Gray-scale Images Using Bit-level: Pixels in continuous tone gray-scale images have 256 gray values. In Computer Display Systems (CDS) values start from zero as a black pixel to 255 as a white pixel. Therefore, each pixel in gray-scale image can be evaluated by an eight-bit integer. Each bit in pixels gray value is called a bit level and each binary image representing a bit level is a bit plane. Dividing a gray-scale image into these bit planes and working on each plane separately is called bit level decomposition. Each bit plane is encrypted using binary visual cryptography. All the encrypted shares of the bit planes are recomposed and two gray-scale shares are created. Superposing gray-scale shares reveals the secret.

Problem Definition

This project aims to successfully implement two Visual Cryptography algorithms: Advance Encryption Standard (AES) algorithm and Visual Cryptography for Gray-scale Images Using Bit-level.

Once implemented, this project can be used by individuals and organizations for encrypting images and therefore secure data transfer.

Literature Survey

Advance Encryption Standard (AES) algorithm:

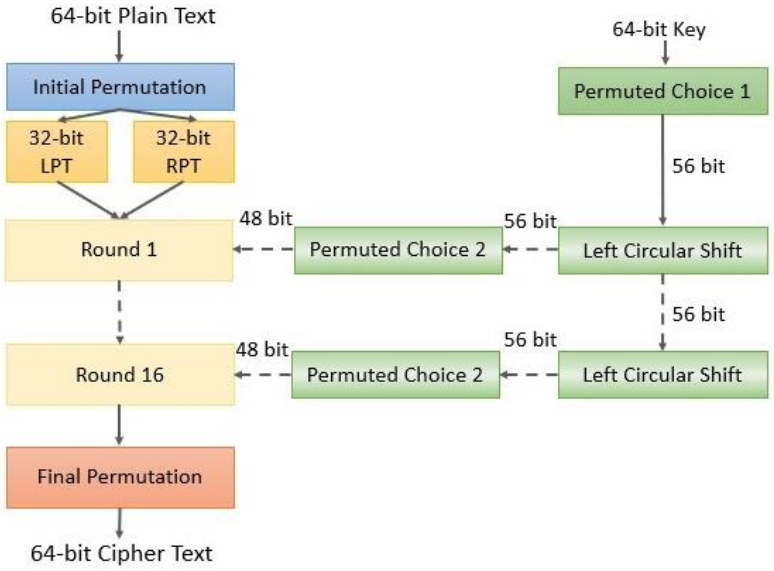
Algorithm	Operation	Status	Alternative	QCR1	Mitigation
DES	Encryption	Avoid	AES	—	—
3DES	Encryption	Legacy	AES	—	Short key lifetime
RC4	Encryption	Avoid	AES	—	—

Avoid: Algorithms that are marked as Avoid do not provide adequate security against modern threats and should not be used to protect sensitive information. It is recommended that these algorithms be replaced with stronger algorithms.

Legacy: Legacy algorithms provide a marginal but acceptable security level. They should be used only when no better alternatives are available, such as when interoperating with legacy equipment. It is recommended that these legacy algorithms be phased out and replaced with stronger algorithms.

Short key lifetime: Use of a short key lifetime improves the security of legacy ciphers that are used on high-speed connections. In IPsec, a 24-hour lifetime is typical. A 30-minute lifetime improves the security of legacy algorithms and is recommended.

Data Encryption Standard (DES): Data Encryption Standard (DES) is the symmetric block cipher which encrypts a 64-bit plain text in a 64-bit ciphertext. The DES was introduced by the National Institute of Standard and Technology (NIST) in the 1970s. Initially, DES was only used in financial applications but later it was accepted as the cryptographic algorithm by other organizations too. Being a symmetric cipher, the same key is used in encryption and decryption process of DES.



Data Encryption Standard (DES)

Disadvantages of DES:

- DES has a 56-bit key which raises the possibility of 2^{56} possible keys which make brute force impossible.
- The 8 S-boxes used in each round were not made public and even it impossible for any to discover the design of the s-boxes which makes the attack more impossible.
- The number of rounds in DES increases the complexity of the algorithm.

3DES: Although it's officially known as the Triple Data Encryption Algorithm (3DEA), it is most commonly referred to as 3DES. This is because the 3DES algorithm uses the Data Encryption Standard (DES) cipher three times to encrypt its data. DES is a symmetric-key algorithm based on a Feistel network. As a symmetric key cipher, it uses the same key for both the encryption and decryption processes. The Feistel network makes both of these processes almost exactly the same, which results in an algorithm that is more efficient to implement.

A Comparison of 3DES and AES

In this section, the differences between the two encryption standards are highlighted in terms of security and performance. AES uses three common key lengths, 128, 192, and 256 bits, whereas for 3DES the encryption key is still limited to 56 bits, according to the DES standard. However, since it is equivalent to DES applied three times, the implementer can choose to have either 2 or 3 different 56-bit keys, meaning that 3DES can have encryption key lengths of 168, 112, or 56 bits. However, due to certain vulnerabilities when reapplying the same encryption three times, a 168-bit key has a reduced security equivalent to 112 bits, and using 112 bits has a reduced security equivalent to 80 bits. The bottom line is that 3DES uses identical encryption to DES whereas AES uses a completely different one, 3DES has a shorter length and weaker encryption keys when compared to AES, and 3DES repeatedly applies encryption keys while AES does not. AES is strongly resistant to differential, truncated differential, linear, interpolation and Square attacks, in contrast to 3DES which is vulnerable to differential and linear cryptanalysis and it has weak substitution tables. In addition, the time required to check all possible keys at 50 billion keys per second in AES for a 128-bit key is 5 x 1021 years, whereas 3DES with a 56-bit key would take 400 days. In addition, 3DES uses a block length of 64 bits which is half the size of an AES block length of 128 bits. Another drawback when using 3DES is the need to switch encryption keys after every 32 GB of data transfer to reduce the possibility of leaks. Conversely, using AES provides additional insurance since it is difficult to decipher data from identical blocks. The process of 3DES encryption using 3DES is much longer than AES, because repeating the same encryption process three times in 3DES takes some time when compared to the AES encryption process which is much faster.

RC4 Encryption Algorithm: The RC4 Encryption Algorithm, developed by Ronald Rivest of RSA, is a shared key stream cipher algorithm requiring a secure exchange of a shared key. RC4 is no longer considered secure and careful consideration should be taken regarding

its use. The symmetric key algorithm is used identically for encryption and decryption such that the data stream is simply XORed with the generated key sequence. The algorithm is serial as it requires successive exchanges of state entries based on the key sequence. Hence implementations can be very computationally intensive. The RC4 encryption algorithm is used by standards such as IEEE 802.11 within WEP (Wireless Encryption Protocol) using 40 and 128-bit keys. Published procedures exist for cracking the security measures as implemented in WEP.

Limitations of RC4:

- RC4 is no longer considered secure.
- One in every 256 keys can be a weak key. These keys are identified by cryptanalysis that is able to find circumstances under which one of more generated bytes are strongly correlated with a few bytes of the key.
- A particular RC4 Algorithm key can be used only once.

Visual Cryptography for Gray-scale Images Using Bit-level:

Previous efforts in this topic are almost restricted in processing binary images, which are insufficient for many applications. A few algorithms that do exist for gray-scale images do not provide satisfactory results. One such algorithm is mentioned below.

Visual cryptography for gray-level images by dithering techniques: Instead of using gray subpixels directly to construct shares, a dithering technique is used first to convert a gray-level image into an approximate binary image. Then existing visual cryptography schemes for binary images are applied to accomplish the work of creating shares. The overall effect of the proposed method is the achievement of visual encryption and decryption functions for gray-level images. Ordered dithering is a technique that can be employed to conduct fast and parallel transformation of a gray-level image into an equal-sized binary one. For this, we adopt in this study the space-filling curve ordered dithering (SFCOD) algorithm (Zhang, 1998) that has the merit of keeping image quality by determining dither thresholds along a space-filling curve.

A sample result is shown below. The results are poor with the output image having a checkered appearance which is precisely the reason why this algorithm isn’t used.

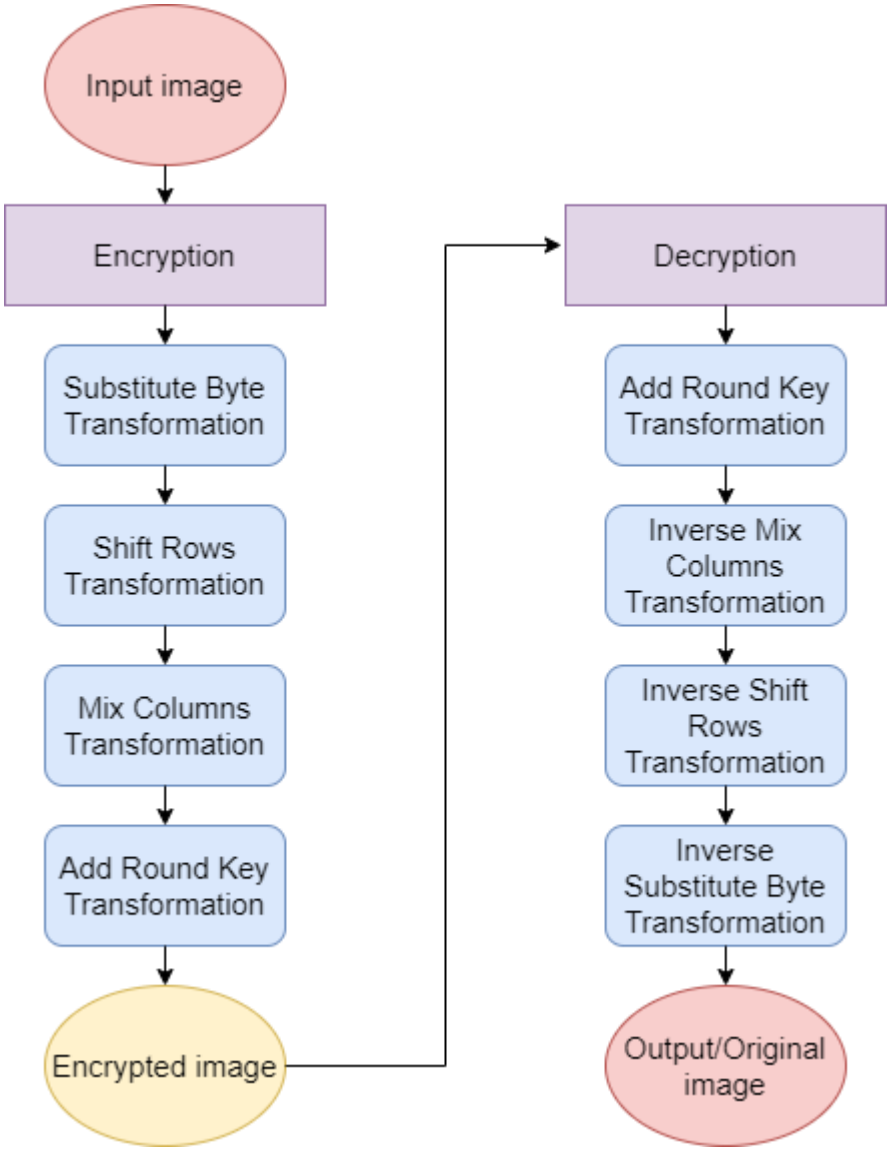
Fig. 11. The original image.

Fig. 14. The decoded image.

Proposed Solution

Advance Encryption Standard (AES) algorithm:

Idea with overall block diagram of the system



Detailed explanation of algorithmic steps involved for every block

Pre-processing

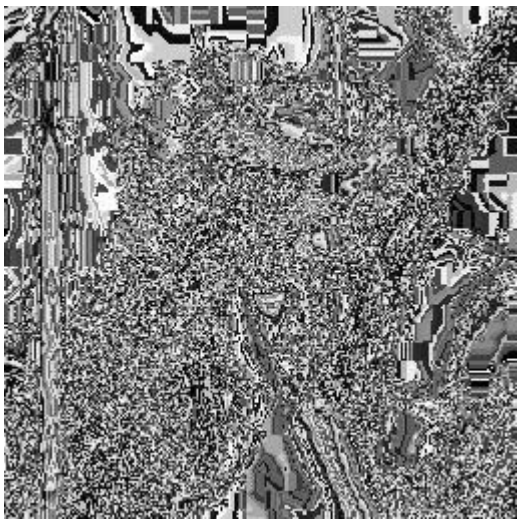
Converting the RGB image to gray-scale image of size 256x256.

Image Encryption

Substitute byte transformation

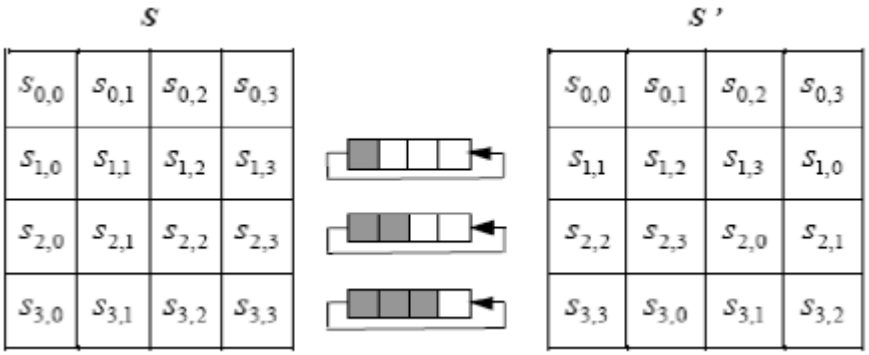
The Substitute bytes transformation is a non-linear byte substitution that operates independently on each byte of the State using a substitution table S-box.

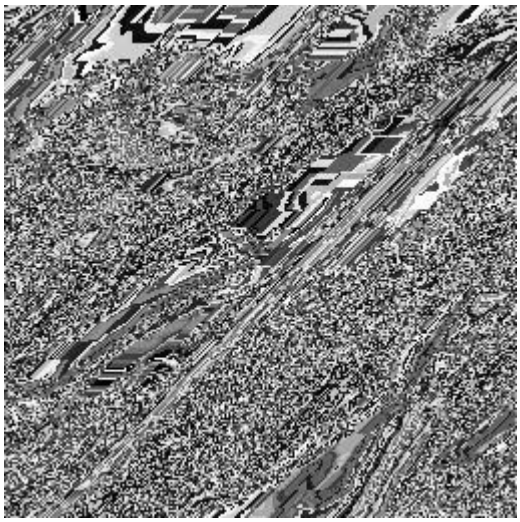
	0x	x1	x2	x3	x4	x5	x6	x7	x8	x9	xA	xB	xC	xD	xE	xF
0x	0x63	0x7c	0x77	0x7b	0xf2	0x6b	0x6f	0xc5	0x30	0x01	0x67	0x2b	0xfe	0xd7	0xab	0x76
1x	0xca	0x82	0xc9	0x7d	0xfa	0x59	0x47	0xf0	0xad	0xd4	0xa2	0xaf	0x9c	0xa4	0x72	0xc0
2x	0xb7	0xfd	0x93	0x26	0x36	0x3f	0xf7	0xcc	0x34	0xa5	0xe5	0xf1	0x71	0xd8	0x31	0x15
3x	0x04	0xc7	0x23	0xc3	0x18	0x96	0x05	0x9a	0x07	0x12	0x80	0xe2	0xeb	0x27	0xb2	0x75
4x	0x09	0x83	0x2c	0x1a	0x1b	0x6e	0x5a	0xa0	0x52	0x3b	0xd6	0xb3	0x29	0xe3	0x2f	0x84
5x	0x53	0xd1	0x00	0xed	0x20	0xfc	0xb1	0x5b	0x6a	0xcb	0xbe	0x39	0x4a	0x4c	0x58	0xcf
6x	0xd0	0xef	0xaa	0xfb	0x43	0x4d	0x33	0x85	0x45	0xf9	0x02	0x7f	0x50	0x3c	0x9f	0xa8
7x	0x51	0xa3	0x40	0x8f	0x92	0x9d	0x38	0xf5	0xbc	0xb6	0xda	0x21	0x10	0xff	0xf3	0xd2
8x	0xcd	0x0c	0x13	0xec	0x5f	0x97	0x44	0x17	0xc4	0xa7	0x7e	0x3d	0x64	0x5d	0x19	0x73
9x	0x60	0x81	0x4f	0xdc	0x22	0x2a	0x90	0x88	0x46	0xee	0xb8	0x14	0xde	0x5e	0x0b	0xdb
Ax	0xe0	0x32	0x3a	0x0a	0x49	0x06	0x24	0x5c	0xc2	0xd3	0xac	0x62	0x91	0x95	0xe4	0x79
Bx	0xe7	0xc8	0x37	0x6d	0x8d	0xd5	0x4e	0xa9	0x6c	0x56	0xf4	0xea	0x65	0x7a	0xae	0x08
Cx	0xba	0x78	0x25	0x2e	0x1c	0xa6	0xb4	0xc6	0xe8	0xdd	0x74	0x1f	0x4b	0xbd	0x8b	0x8a
Dx	0x70	0x3e	0xb5	0x66	0x48	0x03	0xf6	0x0e	0x61	0x35	0x57	0xb9	0x86	0xc1	0x1d	0x9e
Ex	0xe1	0xf8	0x98	0x11	0x69	0xd9	0x8e	0x94	0x9b	0x1e	0x87	0xe9	0xce	0x55	0x28	0xdf
Fx	0x8c	0xa1	0x89	0x0d	0xbf	0xe6	0x42	0x68	0x41	0x99	0x2d	0x0f	0xb0	0x54	0xbb	0x16



Shift rows transformation

In the Shift Rows transformation, the bytes of rows of the State are cyclically shifted over different numbers of bytes. The first row, r=0, is not shifted. This has the effect of moving bytes to “lower” positions in the row while the “lowest” bytes wrap around into the “top” of the row.

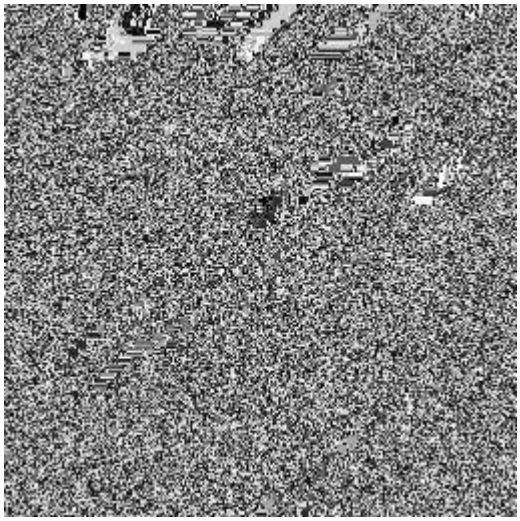




Mix columns transformation

Within this transformation, each column is taken one at a time and each byte within the column is transformed to a new value based on all four bytes in the column. For each column (a0, a1, a2 and a3) we have (where we use Galois Multiplication):

$$\begin{bmatrix} a'_0 \\ a'_1 \\ a'_2 \\ a'_3 \end{bmatrix} = \begin{bmatrix} 2\ 3\ 1\ 1 \\ 1\ 2\ 3\ 1 \\ 1\ 1\ 2\ 3 \\ 3\ 1\ 1\ 2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}$$



Add round key transformation

In the Add Round Key transformation, a Round Key is added to the State by a simple bitwise XOR operation. The Round Key is derived from the Cipher key by means of key schedule process. The State and Round Key are of the same size and to obtain the next State an XOR operation is done per element: b (i, j) = a (i, j) ⊕ k (i, j)

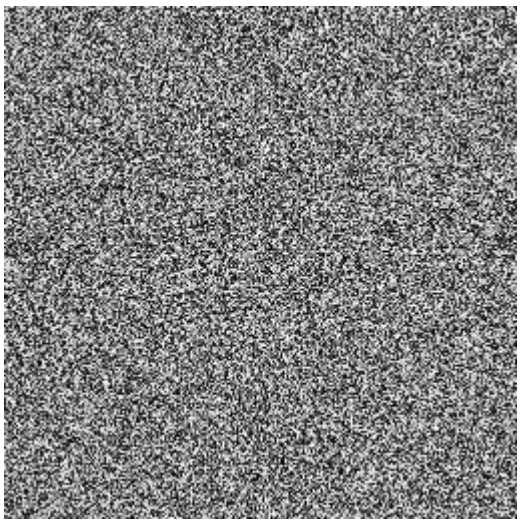
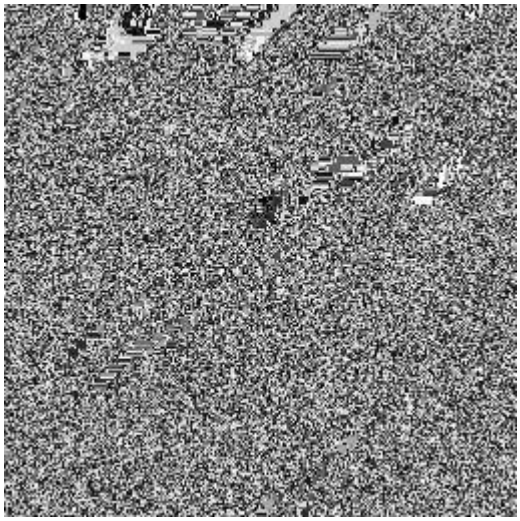


Image Decryption

Add round key transformation

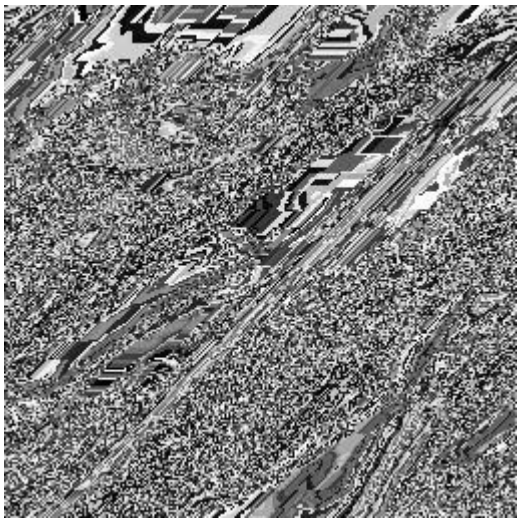
The Round Key derived previously is used again. The State and Round Key are of the same size and to obtain the next State an XOR operation is done per element: b (i, j) = a (i, j) ⊕ k (i, j).



Inverse Mix columns transformation

Inverse Mix Columns is the inverse of the Mix Columns transformation. The inverse is given by:

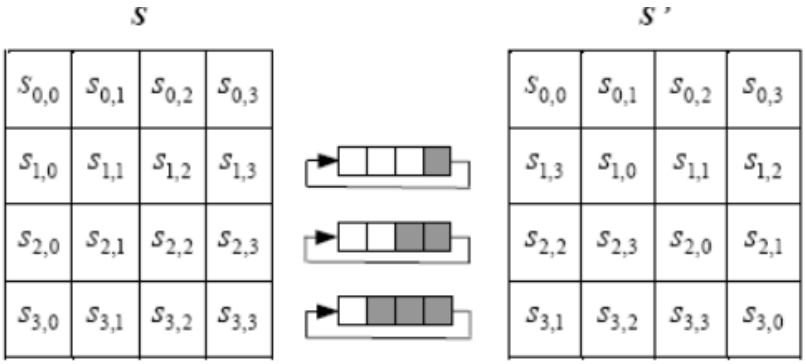
$$\begin{bmatrix} a'_0 \\ a'_1 \\ a'_2 \\ a'_3 \end{bmatrix} = \begin{bmatrix} 14\ 11\ 13\ 9 \\ 9\ 14\ 11\ 13 \\ 13\ 9\ 14\ 11 \\ 11\ 13\ 9\ 14 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

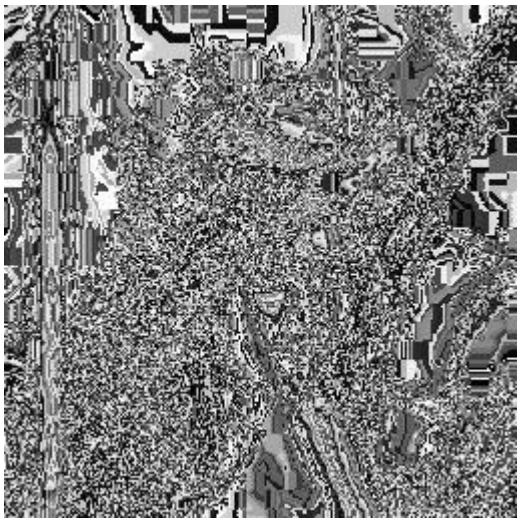


Inverse Shift rows transformation

Inverse Shift Rows is the inverse of the Shift Rows trans-formation. The bytes in the last three rows of the State are cyclically shifted over different numbers of bytes. The first row,

r=0, is not shifted. The bottom three rows are cyclically shifted by Nb-shift (r, Nb) bytes, where the shift value shift (r,Nb) depends on the row number.





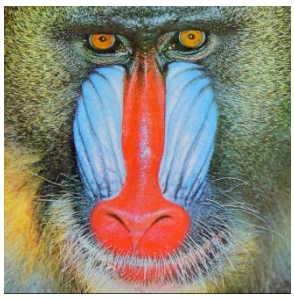
Inverse Substitute byte transformation

Inverse Substitute Bytes is the inverse of the byte substitution transformation, in which the inverse S-box is applied to each byte of the State. It is reverse process of Substitute byte transform.

	0x	1x	2x	3x	4x	5x	6x	7x	8x	9x	Ax	Bx	Cx	Dx	Ex	Fx
0x	0x52	0x09	0x6a	0xd5	0x30	0x36	0xa5	0x38	0xbf	0x40	0xa3	0x9e	0x81	0xf3	0xd7	0xfb
1x	0x7c	0xe3	0x39	0x82	0x9b	0x2f	0xff	0x87	0x34	0x8e	0x43	0x44	0xc4	0xde	0xe9	0xcb
2x	0x54	0x7b	0x94	0x32	0xa6	0xc2	0x23	0x3d	0xee	0x4c	0x95	0x0b	0x42	0xfa	0xc3	0x4e
3x	0x08	0x2e	0xa1	0x66	0x28	0xd9	0x24	0xb2	0x76	0x5b	0xa2	0x49	0x6d	0x8b	0xd1	0x25
4x	0x72	0xf8	0xf6	0x64	0x86	0x68	0x98	0x16	0xd4	0xa4	0x5c	0xcc	0x5d	0x65	0xb6	0x92
5x	0x6c	0x70	0x48	0x50	0xfd	0xed	0xb9	0xda	0x5e	0x15	0x46	0x57	0xa7	0x8d	0x9d	0x84
6x	0x90	0xd8	0xab	0x00	0x8c	0xbc	0xd3	0x0a	0xf7	0xe4	0x58	0x05	0xb8	0xb3	0x45	0x06
7x	0xd0	0x2c	0x1e	0x8f	0xca	0x3f	0x0f	0x02	0xc1	0xaf	0xbd	0x03	0x01	0x13	0x8a	0x6b
8x	0x3a	0x91	0x11	0x41	0x4f	0x67	0xdc	0xea	0x97	0xf2	0xcf	0xce	0xf0	0xb4	0xe6	0x73
9x	0x96	0xac	0x74	0x22	0xe7	0xad	0x35	0x85	0xe2	0xf9	0x37	0xe8	0x1c	0x75	0xdf	0x6e
Ax	0x47	0xf1	0x1a	0x71	0x1d	0x29	0xc5	0x89	0x6f	0xb7	0x62	0x0e	0xaa	0x18	0xbe	0x1b
Bx	0xfc	0x56	0x3e	0x4b	0xc6	0xd2	0x79	0x20	0x9a	0xdb	0xc0	0xfe	0x78	0xcd	0x5a	0xf4
Cx	0x1f	0xdd	0xa8	0x33	0x88	0x07	0xc7	0x31	0xb1	0x12	0x10	0x59	0x27	0x80	0xec	0x5f
Dx	0x60	0x51	0x7f	0xa9	0x19	0xb5	0x4a	0x0d	0x2d	0xe5	0x7a	0x9f	0x93	0xc9	0x9c	0xef
Ex	0xa0	0xe0	0x3b	0x4d	0xae	0x2a	0xf5	0xb0	0xc8	0xeb	0xbb	0x3c	0x83	0x53	0x99	0x61
Fx	0x17	0x2b	0x04	0x7e	0xba	0x77	0xd6	0x26	0xe1	0x69	0x14	0x63	0x55	0x21	0x0c	0x7d

Database designed

Ten different public-domain test images having different features have been selected to test the effectiveness of the algorithm.





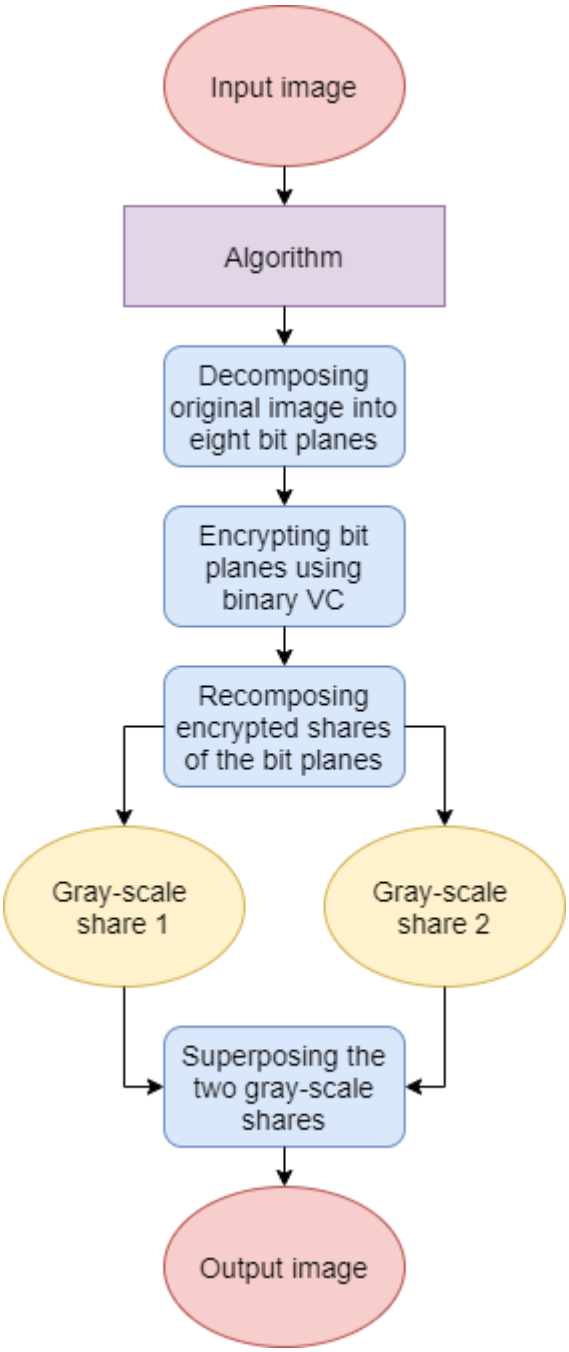


Performance metrics

Accuracy: 100% accuracy is achieved as the original image is completely retrieved. There isn’t a difference of even a single pixel value.

Visual Cryptography for Gray-scale Images Using Bit-level:

Idea with overall block diagram of the system



Detailed explanation of algorithmic steps involved for every block

Pre-processing

Converting the RGB image to gray-scale image of size 256x256.

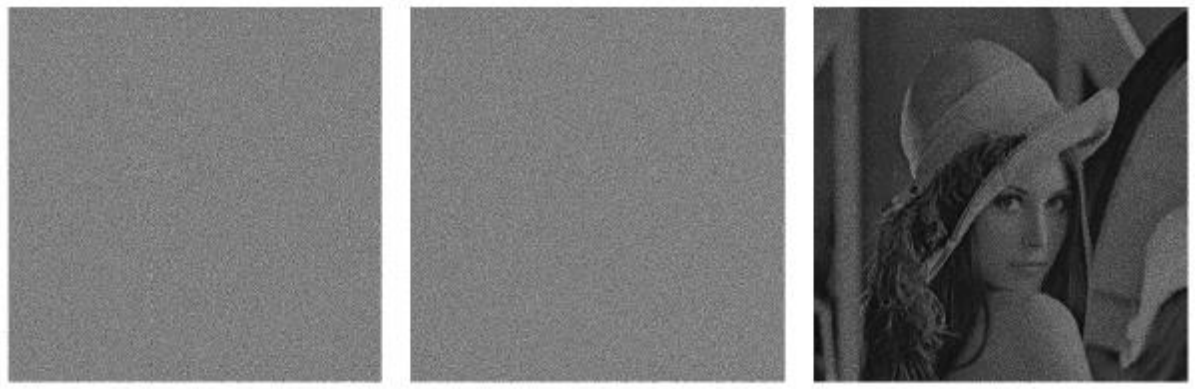
Blocks used in Shares and Stacking Results for Binary Images (Binary VC)

Contrary to traditional encryption schemes, this algorithm does not have any keys. Visual cryptography is applied to encrypt a binary image into two separate binary images called shares that are apparently random, and reveal no information about the original image. Shares are generated based on the original image pixels values; and Human Visual System (HVS) is the decryption device. To achieve this, encrypted images are generated block by block corresponding to each pixel in the original image. Table below illustrates an example of blocks that are used in share images. If the pixel in original image is white, both blocks placed in encrypted images are the same, and if the pixel is black, blocks values are inverse. Blocks for both black and white pixels are shown in table below. Superposing shares results a fully black pixel block for each black pixel in the original image; and a pixel block with black sub-pixels for each white pixel. Using transparency specification and HVS ability, original image is revealed if the encrypted images are superposed correctly. Since there are six different blocks for each pixel in a share which are randomly chosen, decryption with a single share is impossible, taking $6^{m \times n}$ states (m x n is size of the original image) for a brute force attack to decrypt the secret from a single share.

Secret pixel	White	Black
Share 1		
Share 2		
Stacking result		

Proposed Method

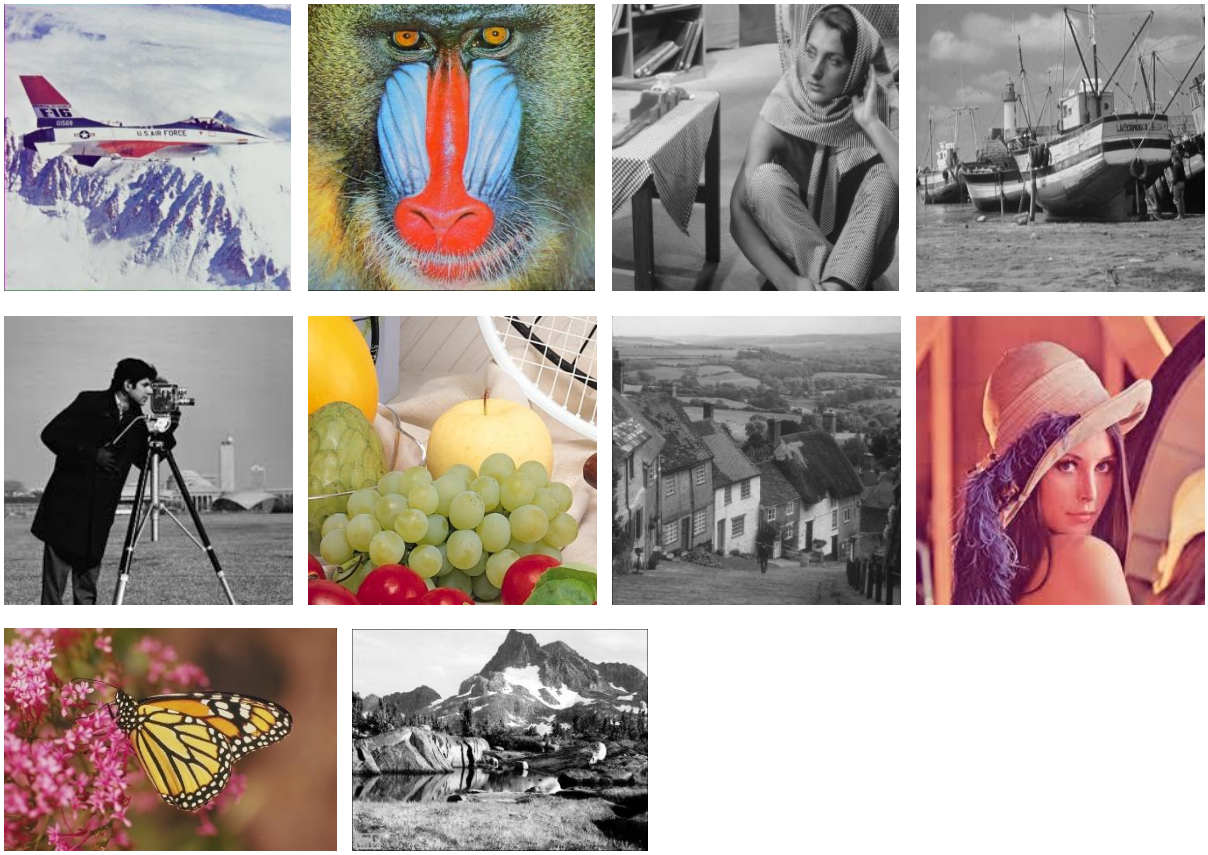
To encrypt a gray-scale image into two gray-scale shares, the original image is decomposed into eight-bit planes. Each bit plane is encrypted using binary VC (previous step). All the encrypted shares of the bit planes are recomposed and two gray-scale shares are created. Superposing gray-scale shares (using bitwise-AND operation) reveals the secret.



(a) Gray-scale share 1; (b) Gray-scale share 2; (c) Retrieved gray-scale image;

Database designed

Ten different public-domain test images having different features have been selected to test the effectiveness of the algorithm.



Performance metrics

Accuracy: 100% accuracy is not achieved. Though the original image is retrieved to a great extent, there is difference in colour tone. The resultant image is on the darker side. That being

said, traditional performance parameters aren’t suitable for this algorithm as the visual quality of the output image is subjective. An alternative option is to use a person’s feedback to establish whether the result is satisfactory enough or not.

Implementation

This project was completely implemented in Google Colab. Colab is a Python development environment that runs in the browser using Google Cloud. The following libraries were used:

- NumPy - adds support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.
- IPython.display - Public API for display tools in IPython.
- OpenCV (cv2) - library of programming functions mainly aimed at real-time computer vision.

Contributions of each group members

Ishaan Almeida (B007):

- Literature survey.
- Coded the Substitute Byte Transformation, Inverse Substitute Byte Transformation, Mix Columns Transformation and Inverse Mix Columns Transformation for AES algorithm.
- Coded the part for recomposing the encrypted shares of the bit planes for Bit-level algorithm.
- Report writing.

Hrishikesh Balaji (B010):

- Literature survey.
- Coded the Shift Rows Transformation, Inverse Shift Rows Transformation and Add Round Key Transformation for AES algorithm.
- Coded the part for RGB to gray-scale image conversion and image resizing.
- Report writing.

Aditi Gupta (B035):

- Literature survey.
- Coded the Shift Rows Transformation, Inverse Shift Rows Transformation and Add Round Key Transformation for AES algorithm.
- Coded the part for superposing the two gray-scale shares for Bit-level algorithm.
- Report writing.

Parth Jalan (B037):

- Literature survey.
- Coded the Substitute Byte Transformation, Inverse Substitute Byte Transformation, Mix Columns Transformation and Inverse Mix Columns Transformation for AES algorithm.
- Coded the part for decomposing the original image into eight-bit planes and encrypting those bit planes using binary VC for Bit-level algorithm.
- Report writing.

Graphical User Interface screenshots / all possible input images and its corresponding output images obtained at various stages of processing

Advance Encryption Standard (AES) algorithm

1	2	3
4	5	6
7	8	9

1 – 256x256 gray-scale input image

2 – Image after Substitute Byte Transformation

3 – Image after Shift Rows Transformation

4 – Image after Mix Columns Transformation

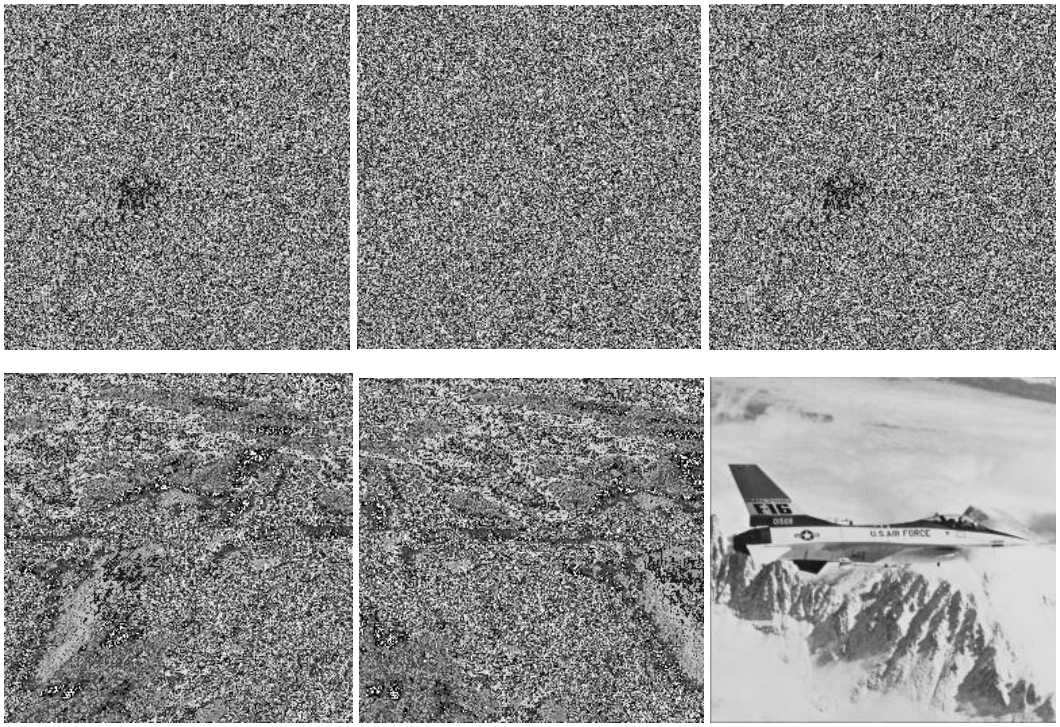
5 & 6 – Image after Add Round Key Transformation

7 – Image after Inverse Mix Columns Transformation

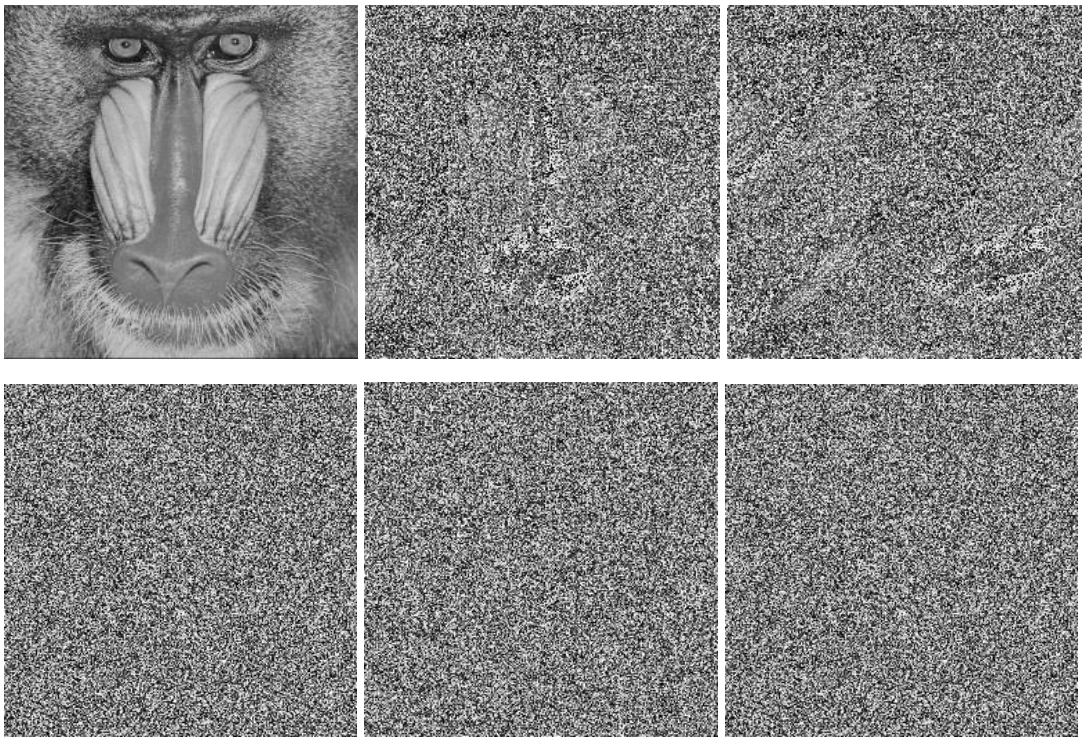
8 – Image after Inverse Shift Rows Transformation

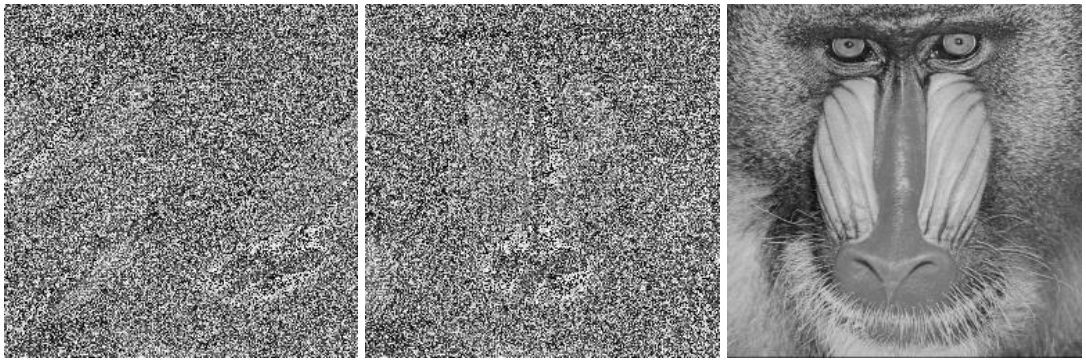
9 – Image after Inverse Substitute Byte Transformation/ Output Image

Sample 1:



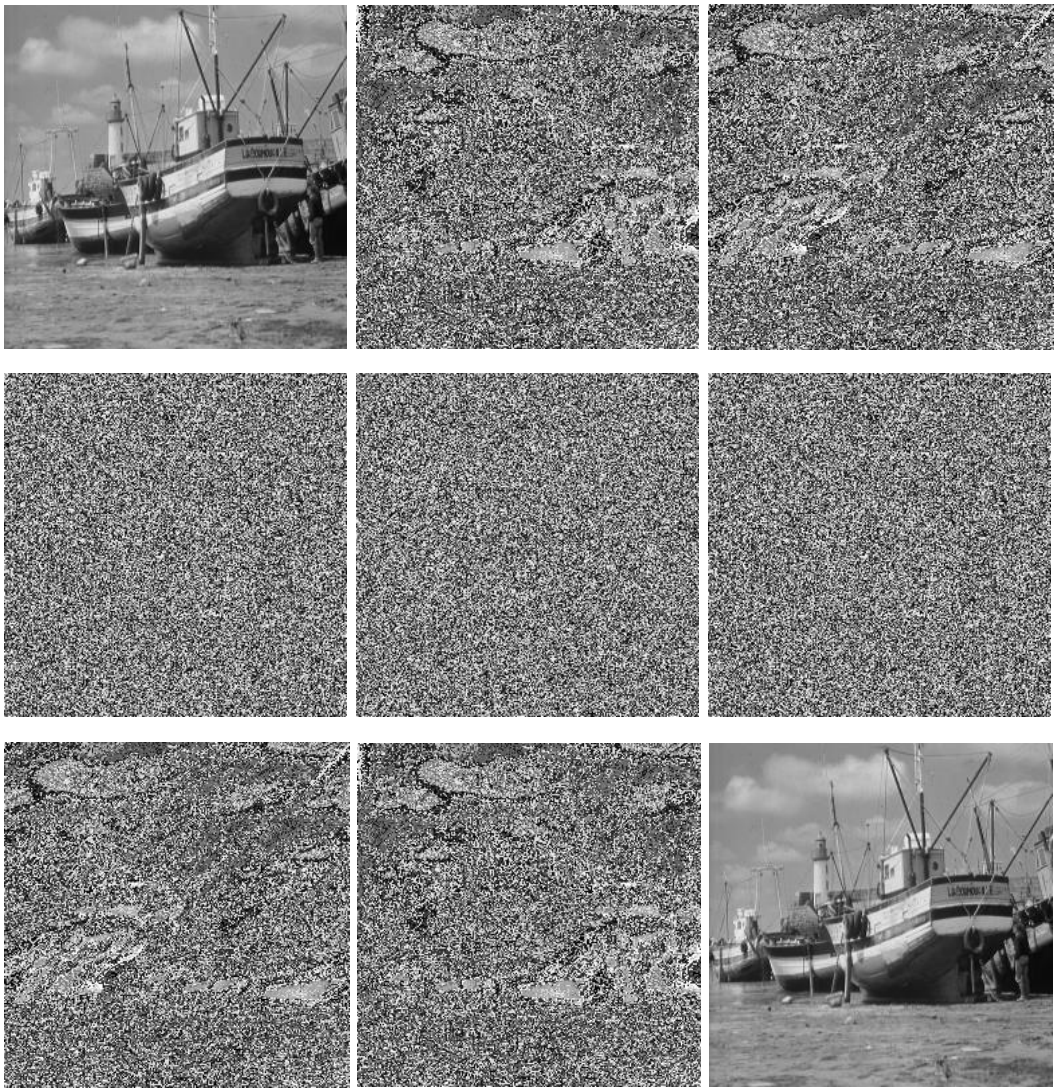
Sample 2:



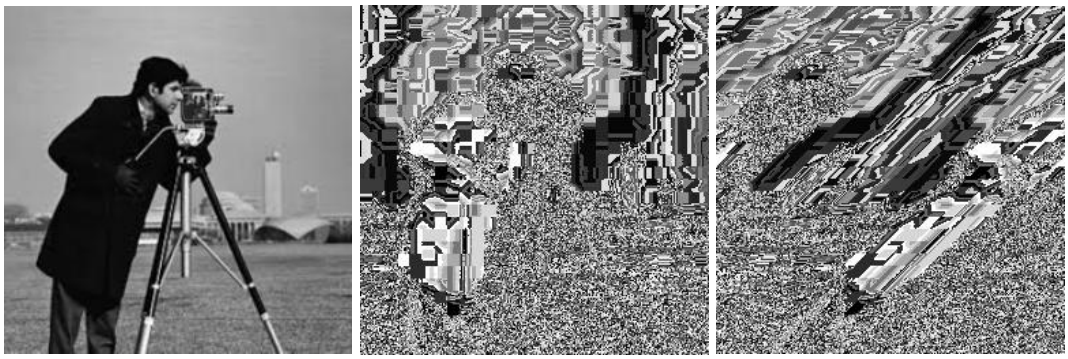


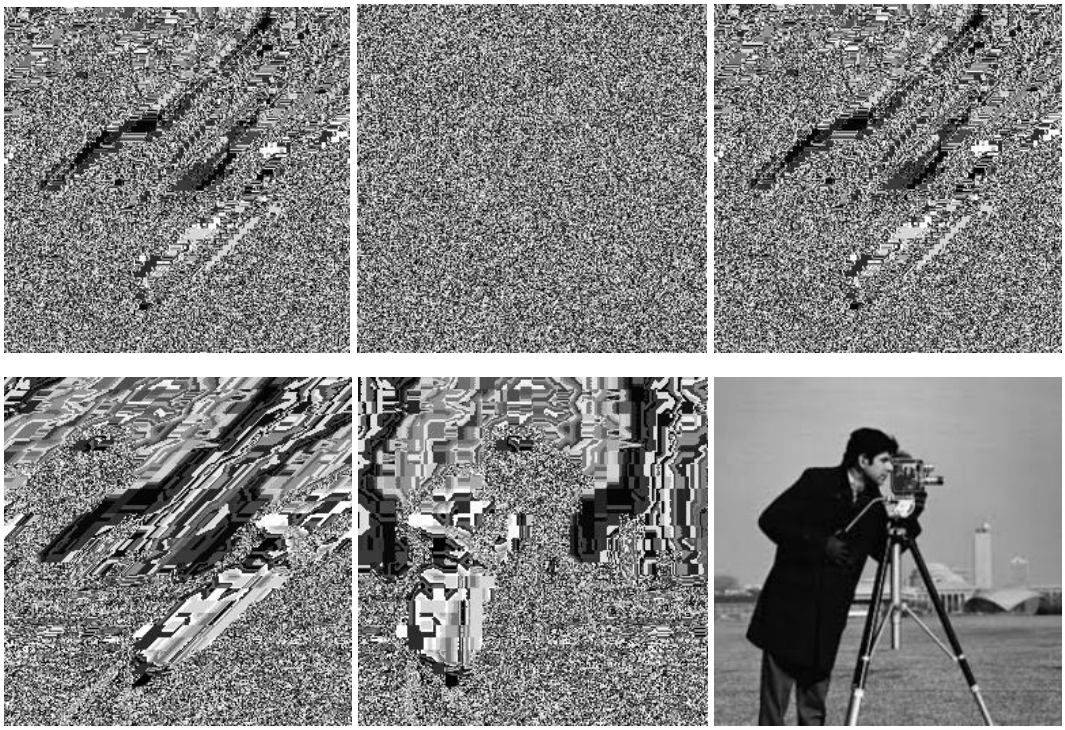
Sample 3:

Sample 4:

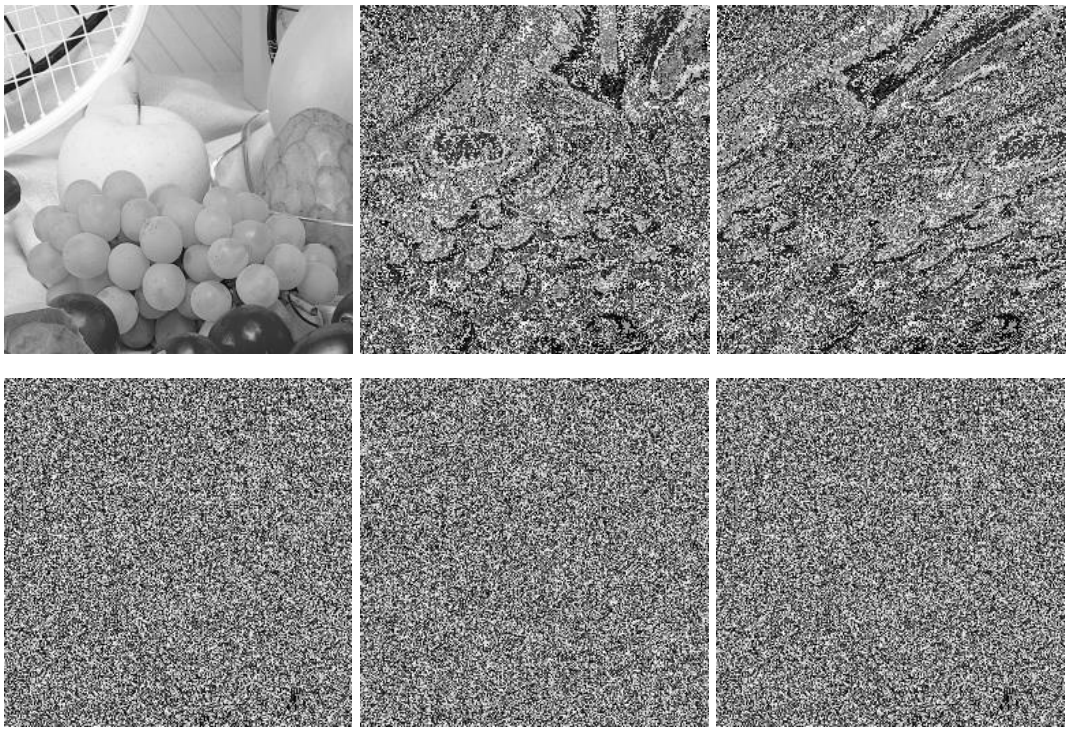


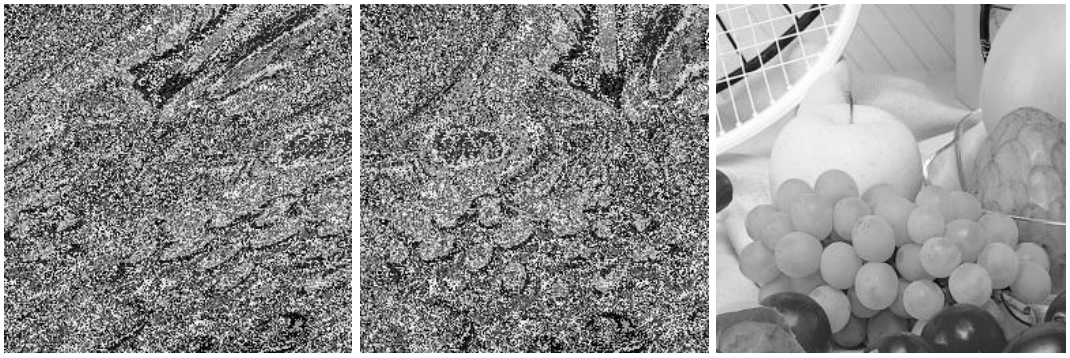
Sample 5:



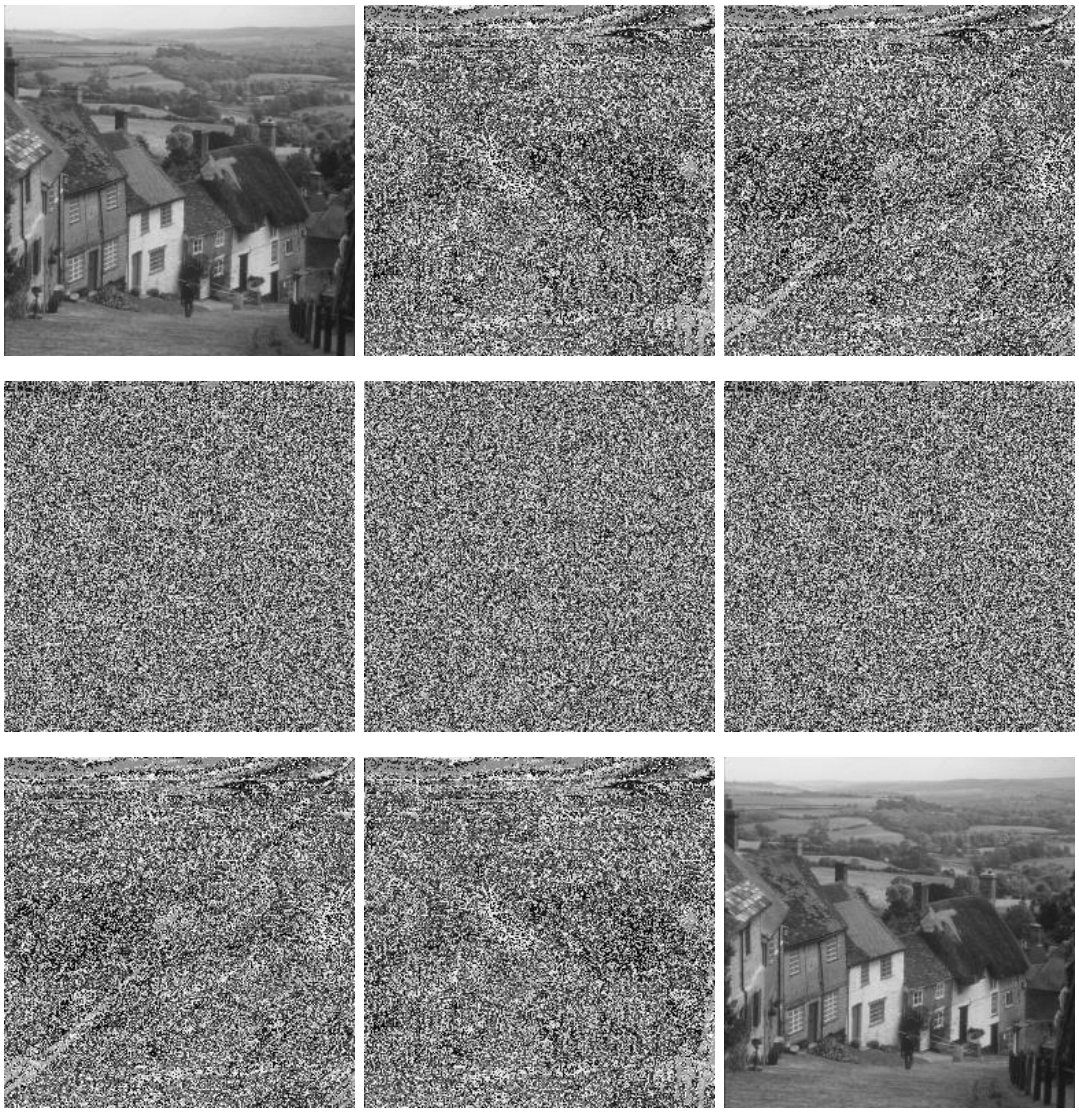


Sample 6:

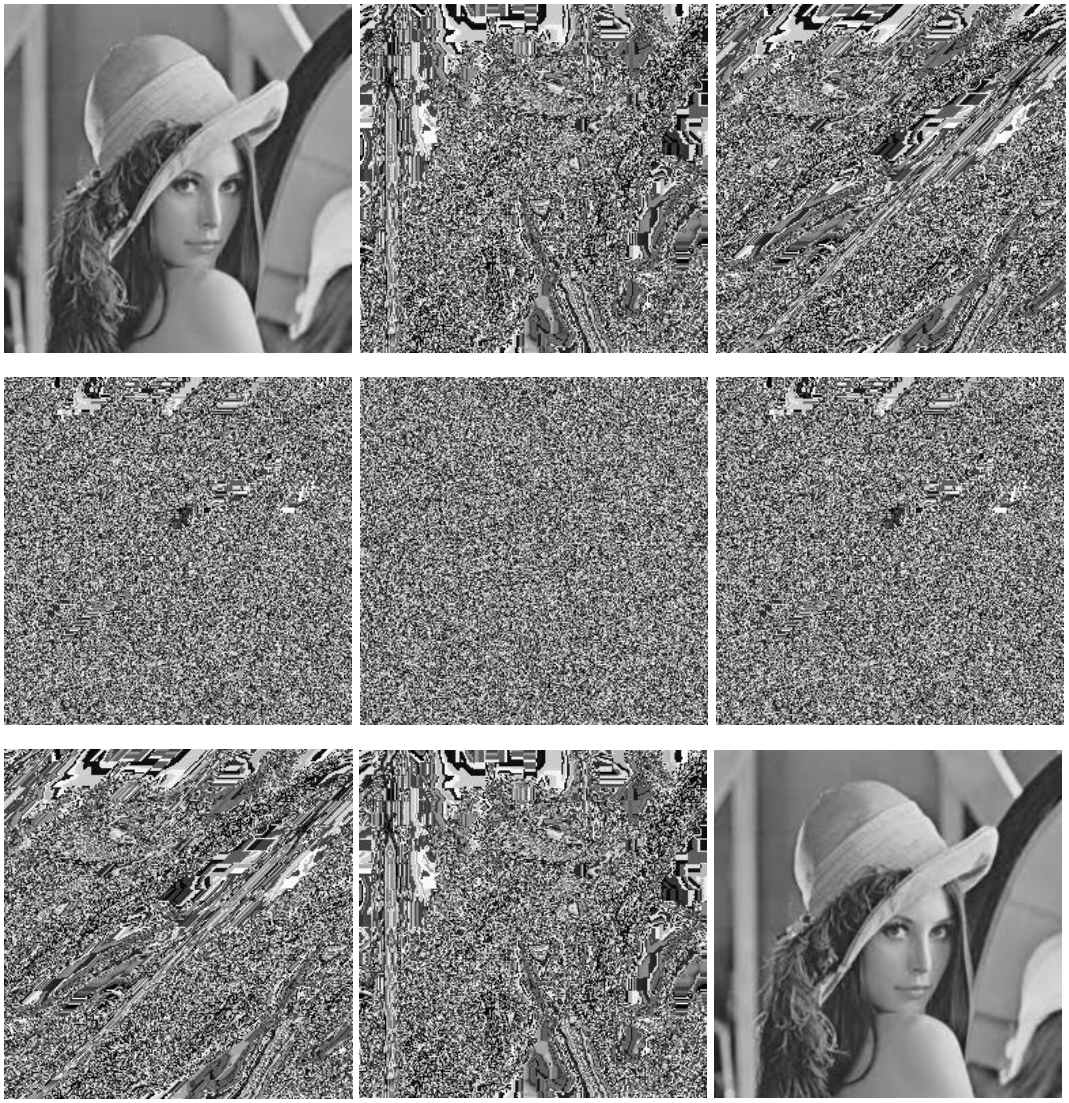




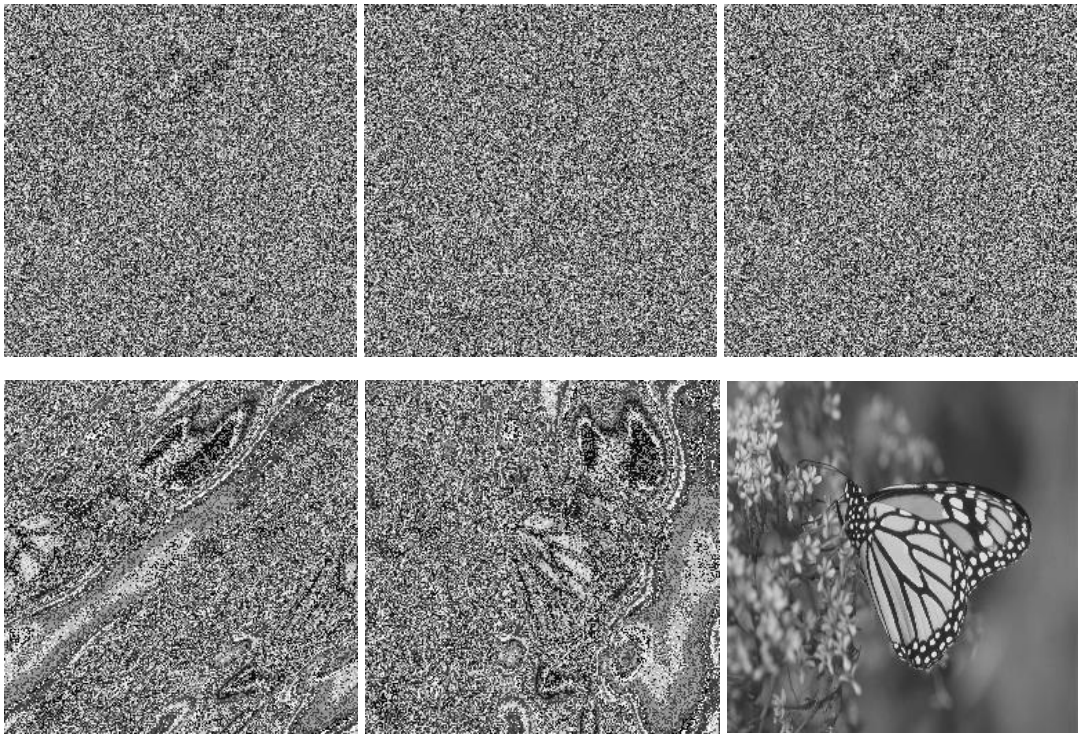
Sample 7:



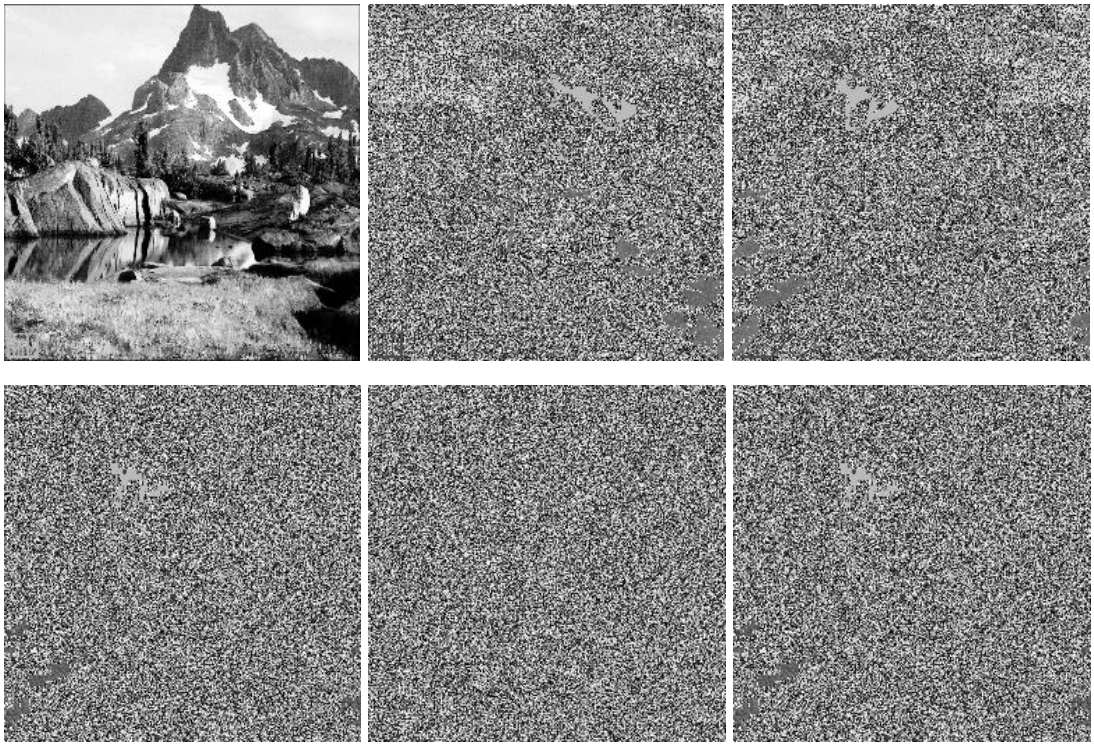
Sample 8:

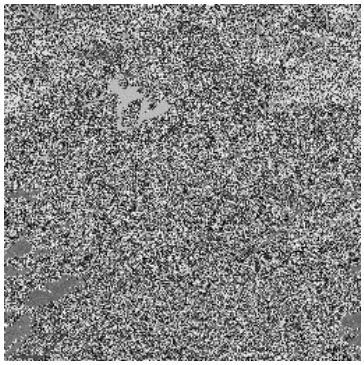
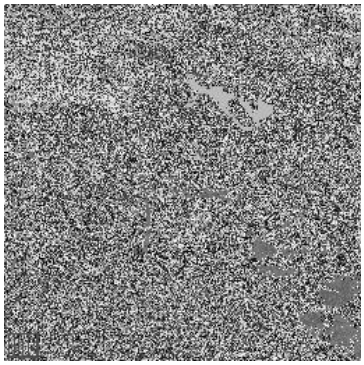
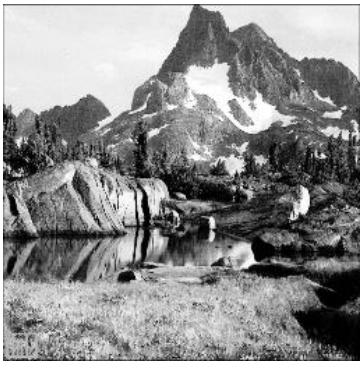


Sample 9:



Sample 10:



Visual Cryptography for Gray-scale Images Using Bit-level

1	2	3	4

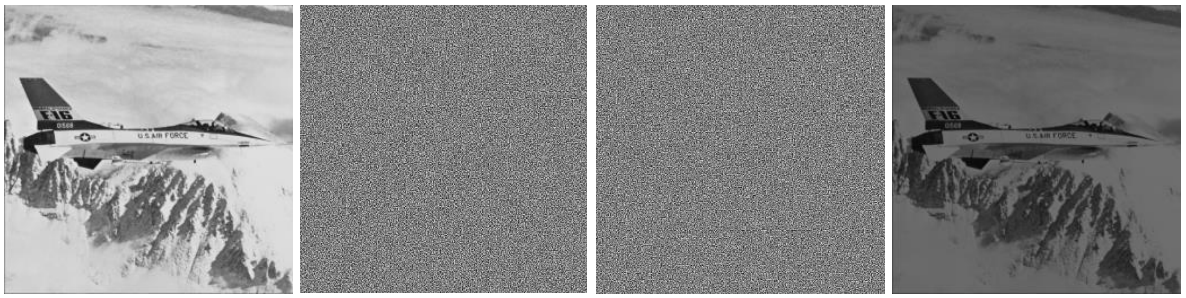
1 – 256x256 gray-scale input image

2 – Gray-scale share 1

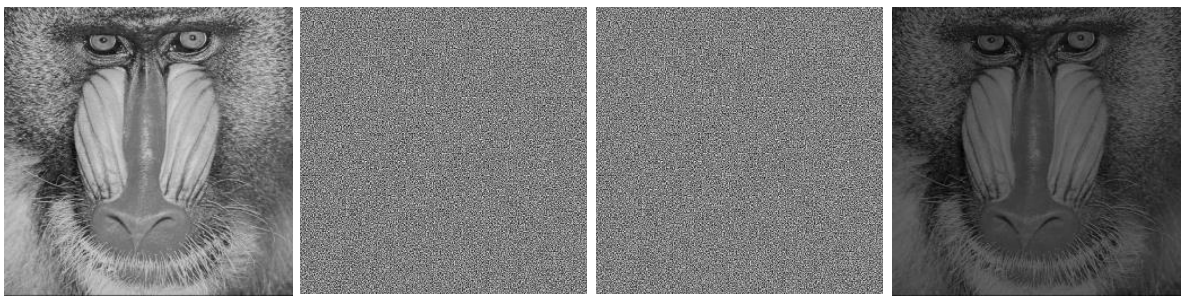
3 – Gray-scale share 2

4 – Retrieved Image

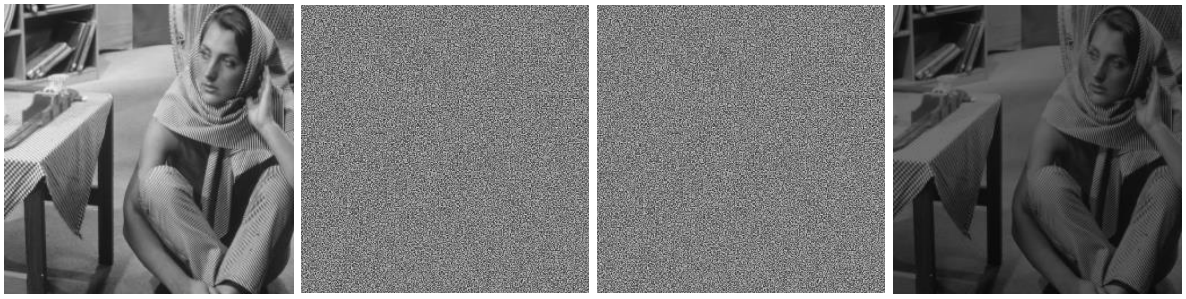
Sample 1:



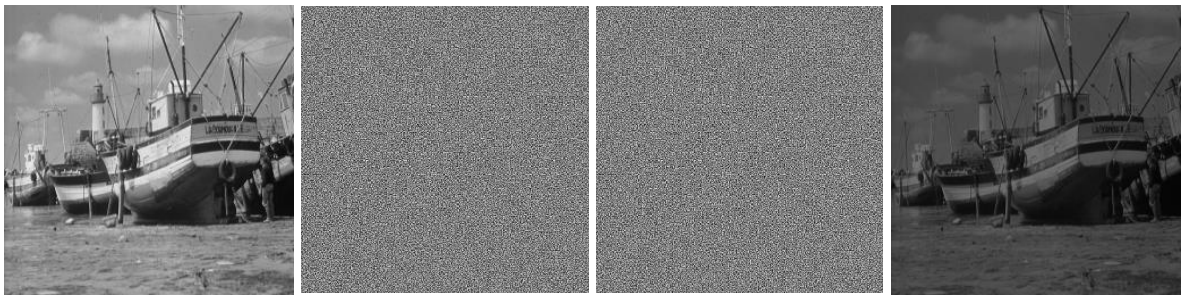
Sample 2:



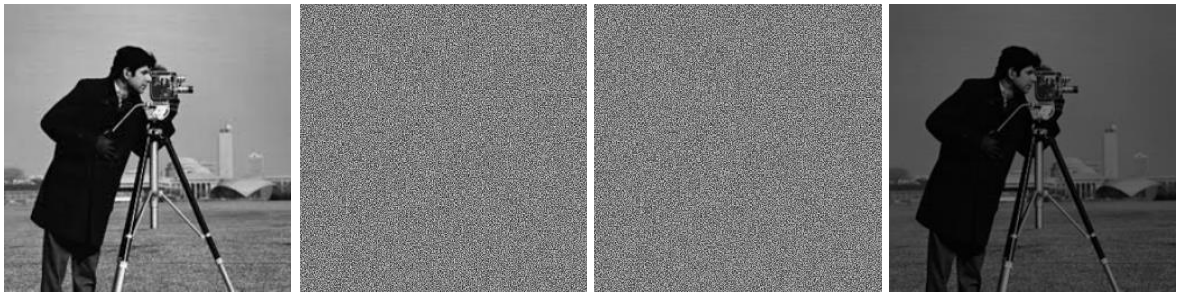
Sample 3:



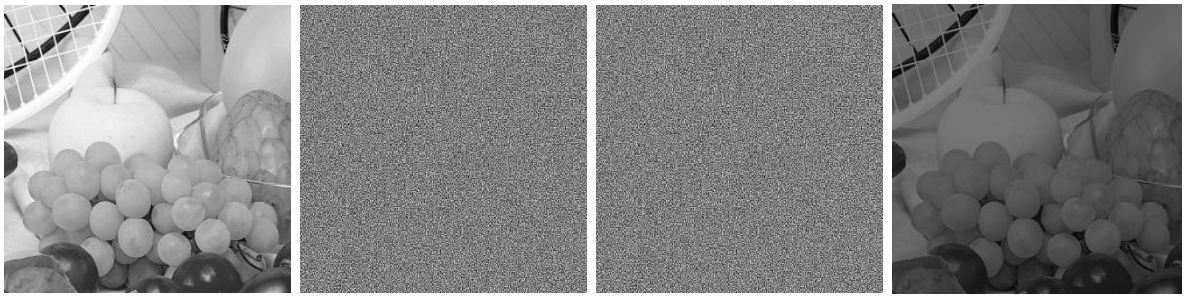
Sample 4:



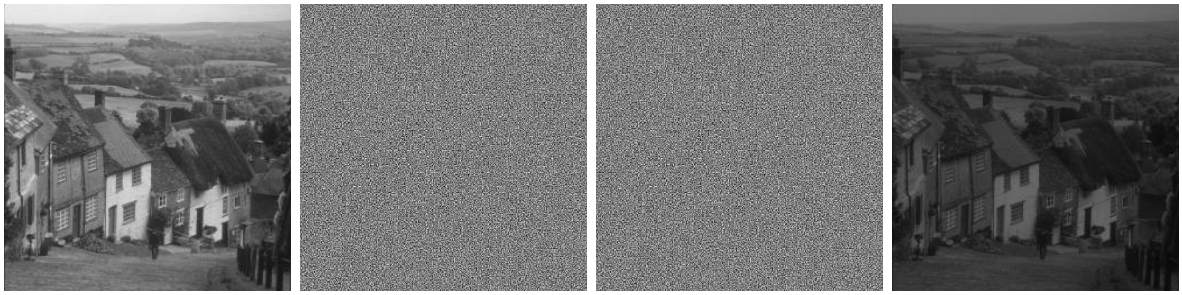
Sample 5:



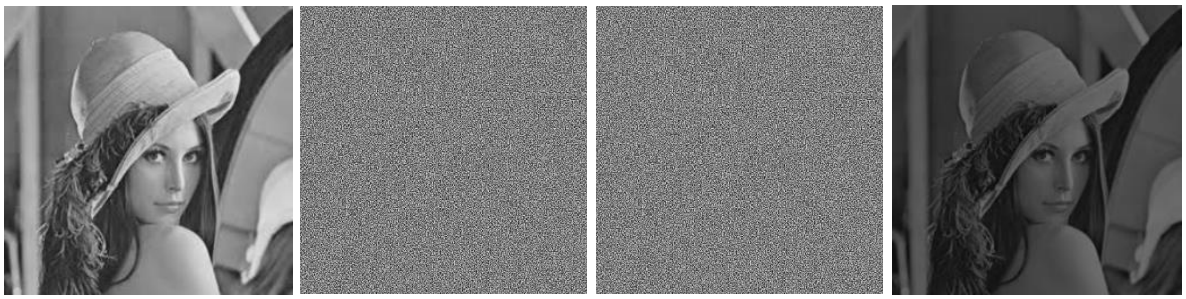
Sample 6:



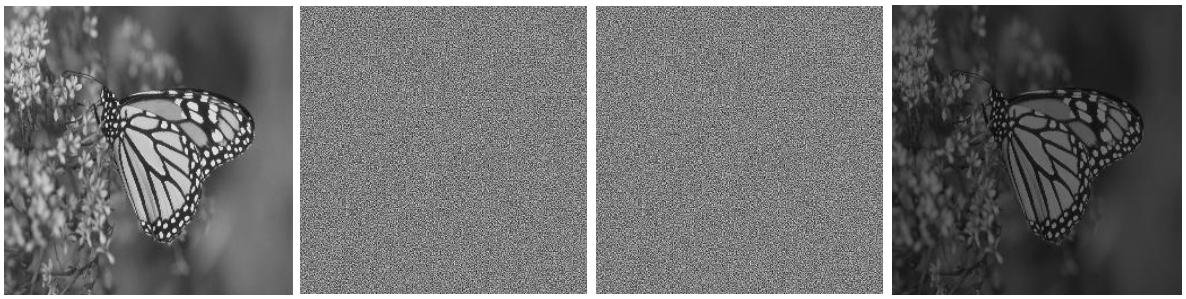
Sample 7:



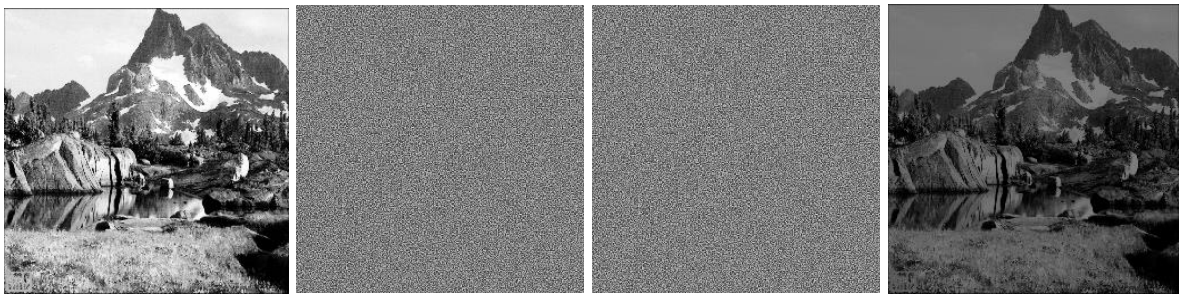
Sample 8:



Sample 9:



Sample 10:



Working code

Advance Encryption Standard (AES) algorithm

The Colab notebook has been uploaded on the following GitHub repository:

https://github.com/parthjalan37/Visual-Cryptography/Visual_Cryptography_(AES).ipynb

Visual Cryptography for Gray-scale Images Using Bit-level

The Colab notebook has been uploaded on the following GitHub repository:

https://github.com/parthjalan37/Visual-Cryptography/Visual_Cryptography_(Bit_level).ipynb

Conclusion

In this project, Image Encryption and Decryption using AES algorithm is implemented to secure the image data from an unauthorized access. A Successful implementation of symmetric key AES algorithm is one of the best encryption and decryption standards available in market. The original images can also be completely reconstructed without any distortion. It has shown that the algorithms have extremely large security key space and can withstand most common attacks such as the brute force attack, cipher attacks and plaintext attacks.

Visual cryptography using bit-level is a very useful technique in secure communication as it uses no computing devices in the decryption phase. In contrast to previous methods, this method does not need the change of the original image to binary (with halftone techniques) and it is easy to understand and implement. Also, decryption with a single share needs $8^{2m \times 2n}$ images to find the secret with a single share; so, the security of the proposed method is guaranteed because each single share leaks no information about the original image.

Limitation and Future Scope

The AES algorithm has no limitation as such, but there is always scope for improvement. So, one aspect that can perhaps be pursued is to reduce the number of sub-operations in its encryption and decryption process which is currently at four each.

As far as the Bit-level algorithm is concerned, work can be done on it to improve its colour tone and make it resemble more and more to the original image.

List of References

1. Deshmukh, P. (2016). An Image Encryption and Decryption using AES Algorithm. International Journal of Scientific and Engineering Research, 210-213.
2. Taghaddos, D. and Latif, & Alimohammad. (2014). Visual Cryptography for Gray-scale Images Using Bit-level. Journal of Information Hiding and Multimedia Signal Processing, 90-97.
3. Siahaan, A. P. U. (2017, September 21). RC4 Technique in Visual Cryptography RGB Image Encryption.
4. Lin, Chou, C., Tsai, & Hsiang, W. (2003). Visual cryptography for gray-level images by dithering techniques. Pattern Recognition Letters, 349-358.
5. https://en.wikipedia.org/wiki/Advanced_Encryption_Standard
6. http://www.brainkart.com/article/Advanced-Encryption-Standard(AES)-Transformation-Functions_8409/
7. https://crypto.stackexchange.com/questions/2402/how-to-solve-mixcolumns
8. http://www.angelfire.com/biz7/atleast/mix_columns.pdf
9. https://homepages.cae.wisc.edu/~ece533/images/
10. https://binaryterms.com/data-encryption-standard-des.html