TABLE OF CONTENTS

APPENDIX

LIST OF TABLES

LIST OF FIGURES

LIST OF ALGORITHMS

CHAPTER 1

INTRODUCTION

1.1 ABOUT PROJECT

The main objective of our project is to provide a GUI based standalone application which will provide us a medium to both encrypt and decrypt images using one of the cryptographic algorithms provided by the application.

It will have 2 separate sections for encryption and decryption processes from where we can select an image file to be encrypted/decrypted and input key/block size for the process.

1.2 VISUAL CRYPTOGRAPHY

Visual cryptography is a cryptographic technique which allows visual information to be encrypted in specific a way that decryption becomes a mechanical operation that does not require a computer.

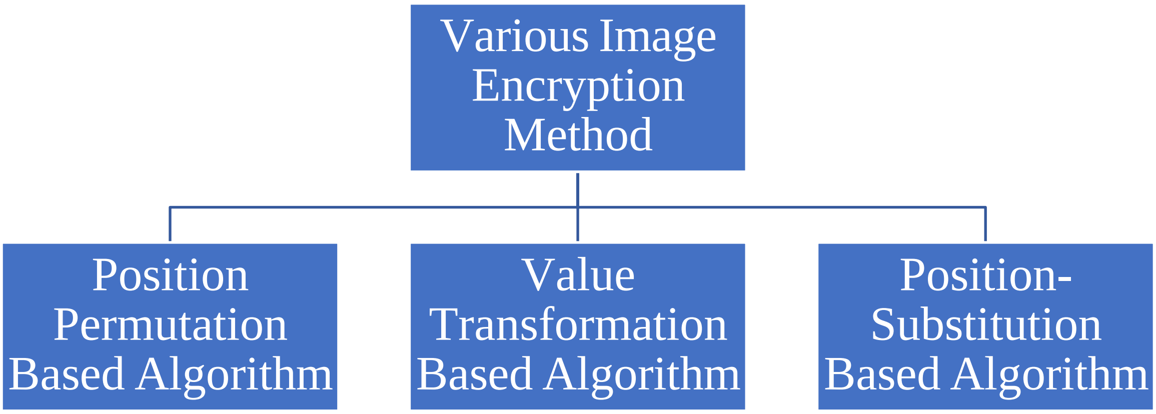
The idea was about producing image shares of a given secret image in a way that the image shares appear meaningless. Recovery of the image can be done by superimposing specified number of share images and, hence, the decoding process requires no special hardware or software and can be simply done by the human eye.

Visual cryptography is a little more advantageous for implementation, while compared to conventional cryptography schemes, since the decryption process does not need any computation. Further, the image based information becomes more secure, since only the intended recipient can reveal the true meaning of the decrypted image.

Images are broadly used in numerous processes. As a result, the safety of image data from unauthorized access is crucial at the hands of user. Image encryption plays a significant role in the field of information hiding. Image hiding or encryption methods and algorithms ranges from simple spatial domain methods to more complicated and reliable frequency domain.. Image encryption have applications in many fields including the internet communication, transmission, medical imaging etc.

1.3 VARIOUS IMAGE ENCRYPTION METHODS

There are various types of image encryption methods. The image encryption algorithms can be categories into three major groups.


Various Image Encryption Method
Position Permutation Based Algorithm
Value Transformation Based Algorithm
Position-Substitution Based Algorithm

1.3.1. Position Permutation (Transposition) Based Algorithm

Transposition means rearranging elements in the plain image. the rearrangement of element can be done by bit, pixel, and block wise. The permutation of bits decreases the perceptual information, whereas the permutation of pixels and blocks produce high level security. In the bit permutation technique, the bits in each pixel are permuted using the permutation keys with the key length equal to 8. In the pixel permutation, 8 pixels are taken as a group and permuted with the same size key. In this investigation the combination of block, bit, and pixel permutation are used respectively.

1.3.2 Value Transformation Based Algorithm

Values Transformation Based algorithm is based on the technique in which the value of each pixel is change to some other value. The new value of pixel is evaluated by applying some algorithm on pixel. Basically, algorithm is mathematical computation where we take input as a pixel value compute it, with some formulas and produce a new value for that pixel.

1.3.3 Position-Substitution Based Algorithm

This technique is combination of both position permutation and value transformation. Position permutation and value transformation can be combined. In this technique first pixels are reordered and then a key generator is used to substitute the pixel values.

1.4 LITERATURE OVERVIEW

Cryptography is the Science of information security which is derived from the Greek kryptos, meaning hidden. It is the process of protecting data, converting data into unreadable cipher format. The process of changing data into cipher format is known as encryption while the process of converted back data that is in cipher format to the original data is known as decryption. The purposes of cryptography are as follows:

1) **Confidentiality:** Assures that private data remains private.

2) **Integrity:** Assures that an object is not distorted illegitimately.

3) **Non-repudiation:** Assures against a party denying an information or interaction that they initiated.

4) **Authentication:** Assure that the characteristic of all parties attempting access.

Cryptography algorithms play an important role in information security. They can be divided into Symmetric and Asymmetric key cryptography.

Symmetric algorithms are of two types : block ciphers and stream ciphers.
The block ciphers are operating on data in groups or blocks. For instances, Data Encryption Standard (DES), Advanced Encryption Standard (AES) and Blowfish. Stream ciphers are operating on a single bit at a time. RC4 is stream cipher algorithm.

Asymmetric key (or public key) encryption is used to solve the problem of key distribution. In Asymmetric key encryption, two keys are used; private keys and public keys. Public key is used for encryption and private key is used for decryption (e.g. Digital Signatures). Public key is known to the public and private key is known only to the user.

Table 1.1 Comparison of various cryptography techniques algorithm

VARIOUS CRYPTOGRAP HY TECHNIQUES Algorithm	Key Size	Block Size	Rounds
DES	56 bits	64 bits	16
3DES	112 bits or 168 bits	64 bits	48
AES	128 bits, 192 bits, 256 bits	128 bits	10,12, 14
BlowFish	32-448 bits(128 bits by default)	64 bits	16

CHAPTER 2
PROJECT

2.1 REQUIREMENT ANALYSIS

2.1.1 Hardware Requirements:

- CPU- x86 64-bit CPU(Intel/ AMD architecture)
- RAM- 8GB
- Storage-10GB
- OS- Windows 10,Linux,Ubuntu

2.1.2 Software Requirements:

- IDE- **Visual studio code or Google Colab**

  Visual Studio Code is a streamlined code editor with support for development operations like debugging, task running, and version control. It aims to provide just the tools a developer needs for a quick code-build-debug cycle and leaves more complex workflows to fuller featured IDEs, such as **Visual Studio IDE**.

  Colaboratory, or "**Colab**" for short, is a product from Google Research. **Colab** allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education.

- Language-**Python 3.7.8 64bit**

  Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse.

2.1.3 Modules/Packages used:

- **Jupyter Kernel**

  Kernels are programming language specific processes that run independently and interact with the Jupyter Applications and their user interfaces. IPython is the reference Jupyter kernel, providing a powerful environment for interactive computing in Python.

- **Pandas**

  Pandas is a high-level data manipulation tool developed by Wes McKinney. It is built on the NumPy package and its key data structure is called the Data Frame. Data Frames allow you to store and manipulate tabular data in rows of observations and columns of variables.

- **tkinter**

  tkinter is Python's de-facto standard GUI package. It is a thin object-oriented layer on top of Tel/Tk. Tkinter is not the only Gui Programming toolkit for Python.

  This framework provides Python users with a simple way to create GUI elements using the widgets found in the Tk toolkit. Tk widgets can be used to construct buttons, menus, data fields, etc. in a Python application.

- **NumPy**

  NumPy is a python library used for working with arrays. It also has functions for working in domain of linear algebra. Fourier transform, and matrices. NumPy was created 2005 by Travis Oliphant. It is an open-source project and you can use it freely. NumPy stands for Numerical Python.

- **OS**

  The OS module in Python provides functions for interacting with the operating system. OS comes under Python's standard utility modules. This module provides a portable way of using operating system-dependent functionality. The "os" and "os.path modules I include many functions to interact with the file system.

- **Crypto**

  PyCryptodome is a self-contained Python package of low-level cryptographic primitives. We will be using it for AES cryptography algorithm

- **PIL**

  The Python Imaging Library adds image processing capabilities to your Python interpreter.

  This library provides extensive file format support, an efficient internal representation, and fairly powerful image processing capabilities.
  The core image library is designed for fast access to data stored in a few basic pixel formats. It should provide a solid foundation for a general image processing too

2.2 METHODOLOGY/ALGORITHMS USED

This covers the detail explanations of the methods/Algorithm used in Implementation of Image Encryption and Decryption.

After considering pros and cons of different Methods/Algorithms available, the AES and Pixel Shuffle/RGB Shuffle method/algorithms has been chosen .

The detail about those Algorithms will be explain concisely.:-

2.2.1 Value Permutation Based Algorithm

Advanced Encryption Standard (AES256)

The Advanced Encryption Standard is a symmetric block cipher chosen by the U.S. government to protect classified information. AES is implemented in software and hardware throughout the world to encrypt sensitive data.

AES will operate on 128 bits of plaintext to produce 128 bits of ciphertext.

AES comes in 128-bit, 192-bit, and 256-bit implementations, with AES 256 being the most secure

There are three different key lengths available in AES which are 128,192 and 256 bits. Some rounds of transformation convert the plaintext into the ultimate cipher-text

The features of AES are as follows −

- Symmetric key symmetric block cipher
- 128-bit data, 128/192/256-bit keys
- Stronger and faster than Triple-DES
- Provide full specification and design details
- Software implementable in C and Java

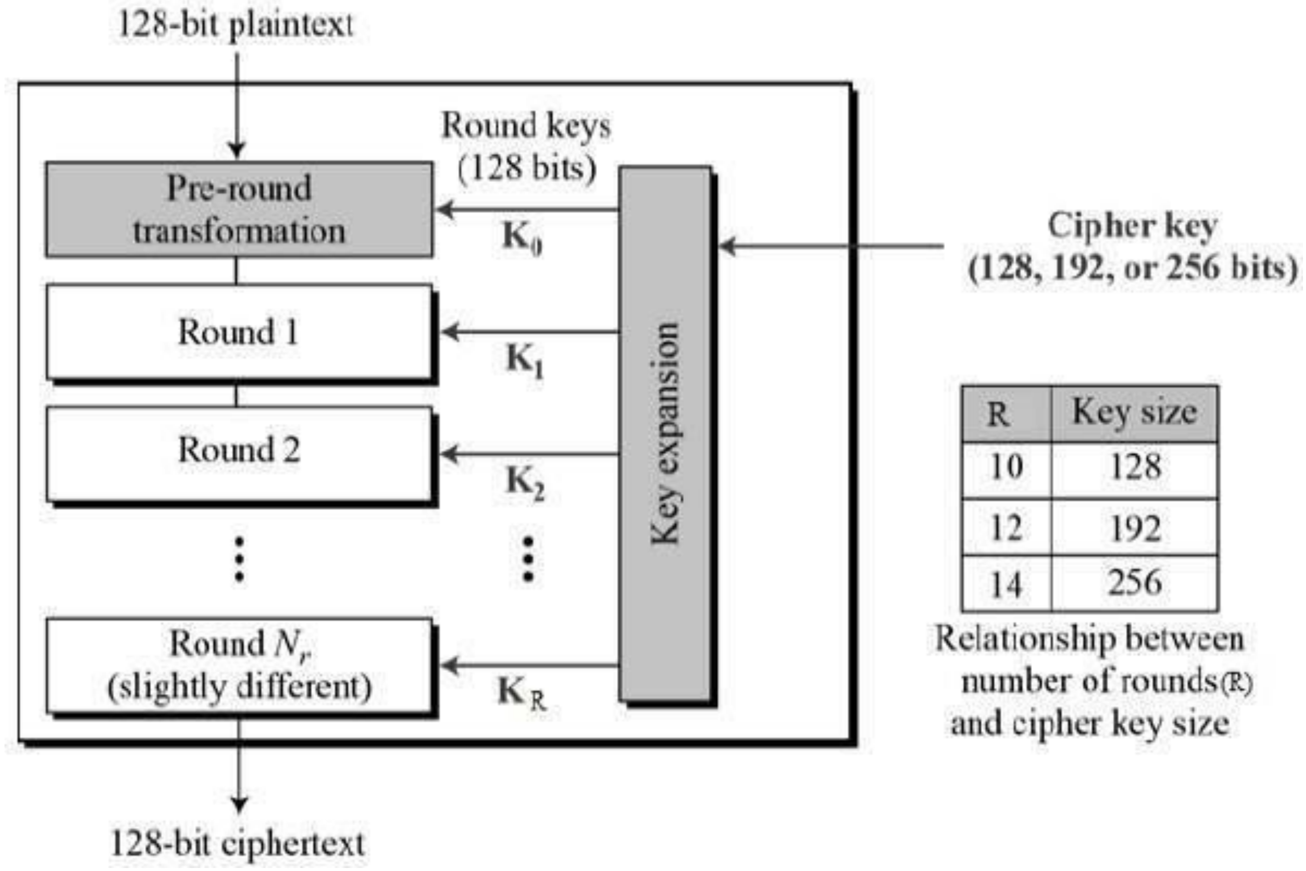
2.2.1.1 Working of AES on 128 bits Text

AES is an iterative rather than Feistel cipher. It is based on ‘substitution–permutation network’.

AES treats the 128 bits of a plaintext block as 16 bytes. These 16 bytes are arranged in four columns and four rows for processing as a matrix

AES uses 10 rounds for 128-bit keys, 12 rounds for 192-bit keys and 14 rounds for 256-bit keys.

Each of these rounds uses a different 128-bit round key, which is calculated from the original AES key

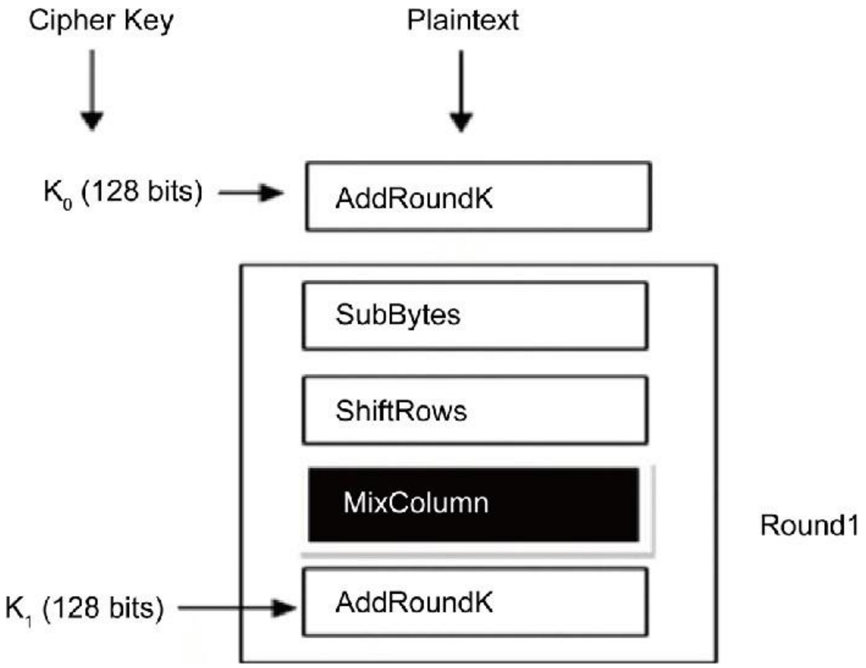


R	Key size
10	128
12	192
14	256

Relationship between number of rounds(R) and cipher key size

2.2.1.2 AES Encryption Process

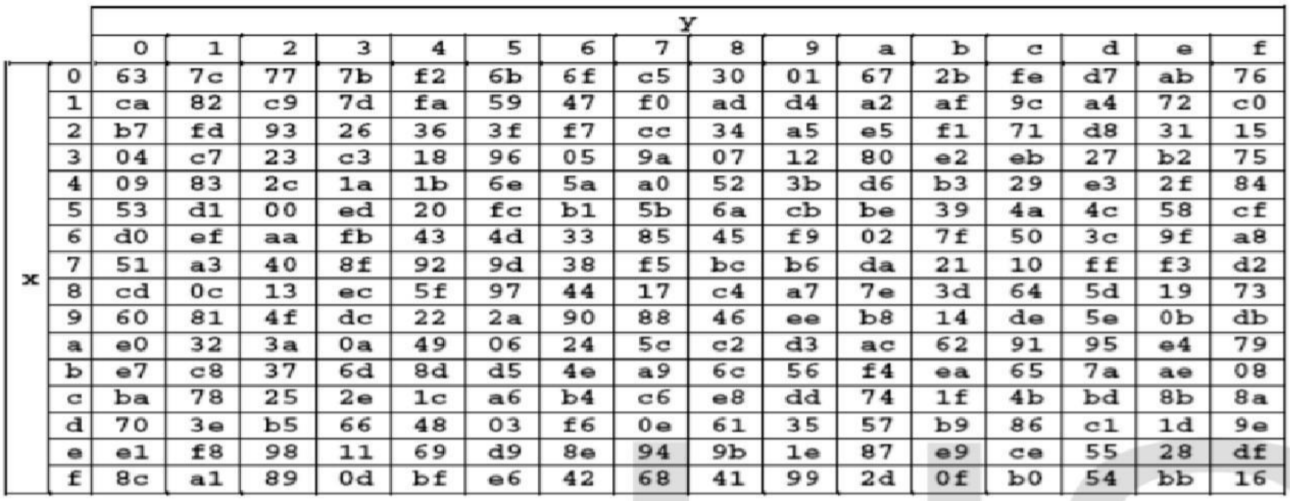
Each round comprises of four sub-processes. The first-round process is depicted below



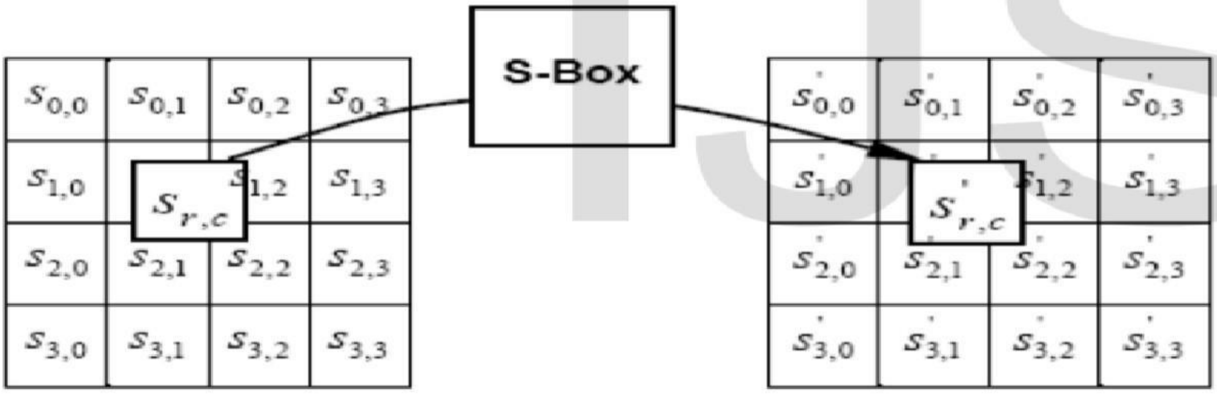
1. Byte Substitution (SubBytes)

The 16 input bytes are substituted by looking up a fixed table (S-box) given in design.

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. The operationof substitute byte is shown in figure



		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
	1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
	2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
	3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
	6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
	7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
	8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
	9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
	a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
	c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
	d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
	e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
	f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16


S-Box
$s_{0,0}$ $s_{0,1}$ $s_{0,2}$ $s_{0,3}$
$s_{1,0}$ $s_{1,2}$ $s_{1,3}$
$S_{r,c}$
$s_{2,0}$ $s_{2,1}$ $s_{2,2}$ $s_{2,3}$
$s_{3,0}$ $s_{3,1}$ $s_{3,2}$ $s_{3,3}$
$s'_{0,0}$ $s'_{0,1}$ $s'_{0,2}$ $s'_{0,3}$
$s'_{1,0}$ $s'_{1,2}$ $s'_{1,3}$
$S'_{r,c}$
$s'_{2,0}$ $s'_{2,1}$ $s'_{2,2}$ $s'_{2,3}$
$s'_{3,0}$ $s'_{3,1}$ $s'_{3,2}$ $s'_{3,3}$

2. Shiftrows

Each of the four rows of the matrix is shifted to the left. Any entries that ‘fall off’ are re-inserted on the right side of row. Shift is carried out as follows −

a. First row is not shifted.

b. Second row is shifted one (byte) position to the left.

c. Third row is shifted two positions to the left.

d. Fourth row is shifted three positions to the left.

The result is a new matrix consisting of the same 16 bytes but shifted with respect to each other.

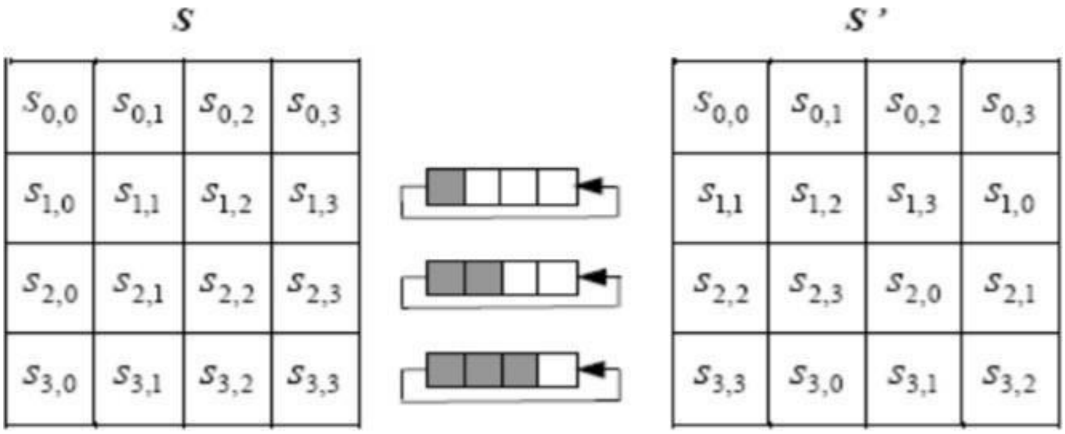
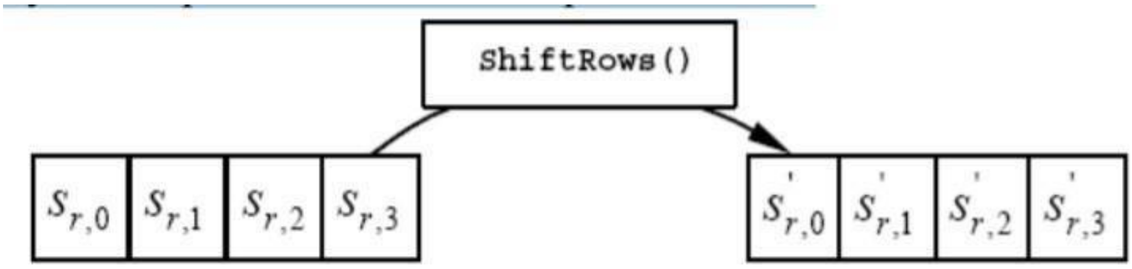


Figure.2.Cyclic shift row operation



3. MixColumns

Each column of four bytes is now transformed using a special mathematical function.

This function takes as input the four bytes of one column and outputs four completely new bytes, which replace the original column.
The result is another new matrix consisting of 16 new bytes
.
The Mix Columns transformation operates on the State column-by-column, treating each column asa four-term polynomial.
The columns are considered as polynomials over GF(2^8) and multiplied modulo x 4 + 1 with a fixed polynomial a(x), given by a(x) = {03}x ^3 + {01}x^ 2 + {01}x + {02} .

The resultant columnsare shown in the figure below. This is the operation of mix columns

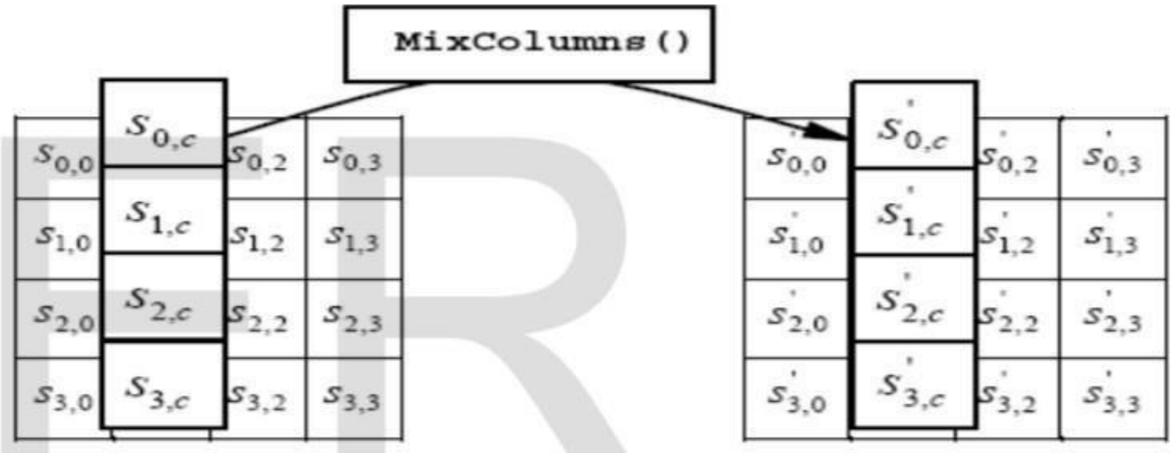

MixColumns()
$s_{0,0}$ $s_{0,c}$ $s_{0,2}$ $s_{0,3}$
$s_{1,0}$ $s_{1,c}$ $s_{1,2}$ $s_{1,3}$
$s_{2,0}$ $s_{2,c}$ $s_{2,2}$ $s_{2,3}$
$s_{3,0}$ $s_{3,c}$ $s_{3,2}$ $s_{3,3}$
$s'_{0,0}$ $s'_{0,c}$ $s'_{0,2}$ $s'_{0,3}$
$s'_{1,0}$ $s'_{1,c}$ $s'_{1,2}$ $s'_{1,3}$
$s'_{2,0}$ $s'_{2,c}$ $s'_{2,2}$ $s'_{2,3}$
$s'_{3,0}$ $s'_{3,c}$ $s'_{3,2}$ $s'_{3,3}$

Figure.3.Mix columns operation

4. Add round key

The 16 bytes of the matrix are now considered as 128 bits and are XORed to the 128 bits of the round key.

If this is the last round then the output is the ciphertext. Otherwise, the resulting 128 bits are interpreted as 16 bytes and we begin another similar round.

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)

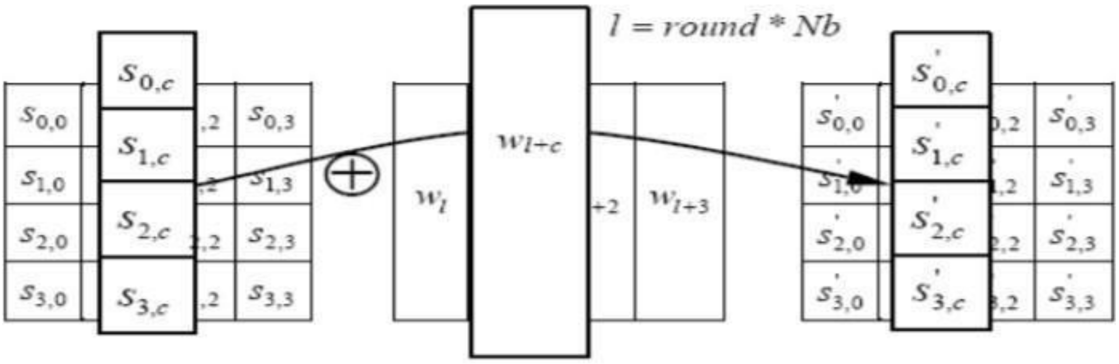
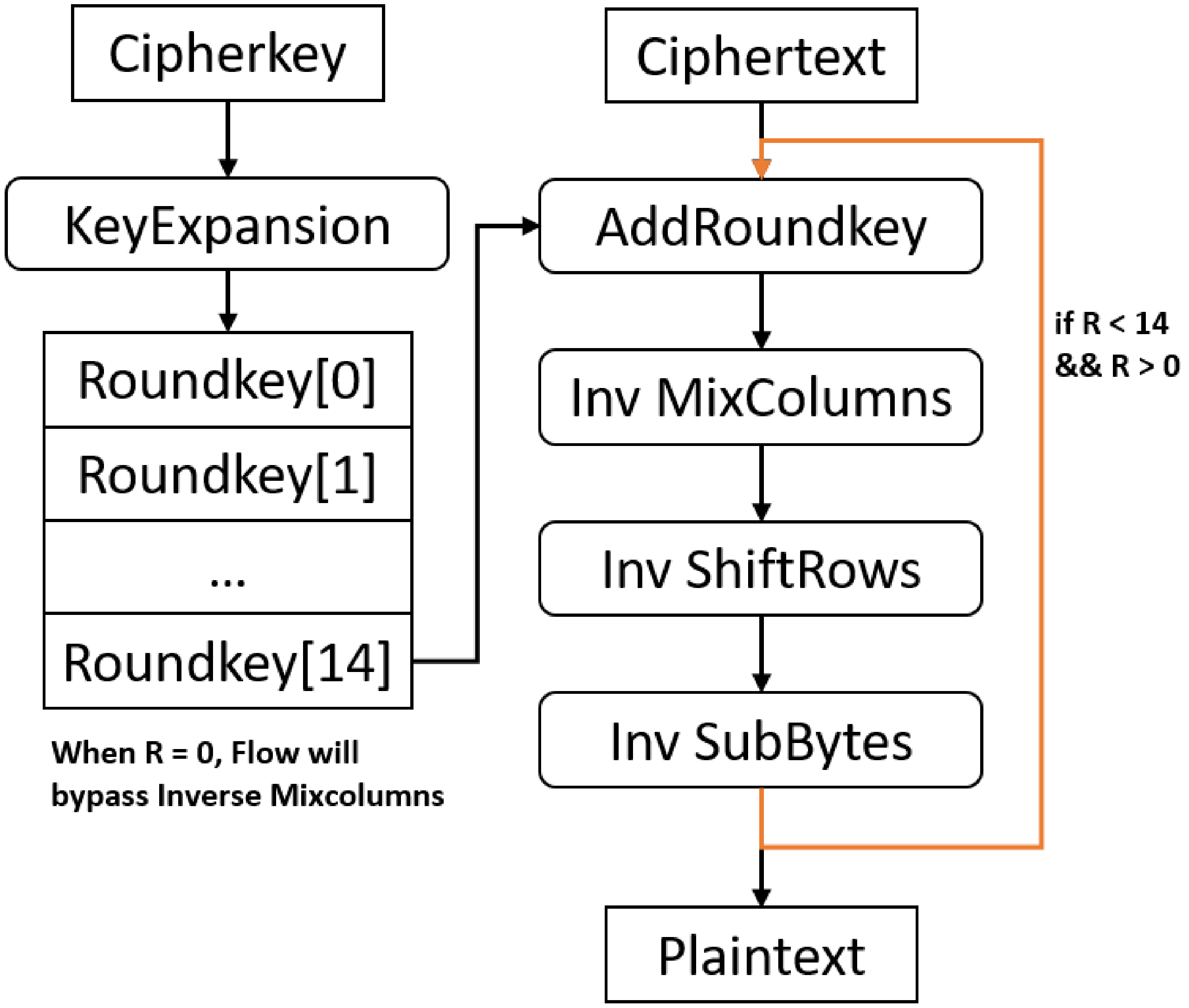


Figure 4. Add round key operation

2.2.1.3 Decryption Process

The process of decryption of an AES ciphertext is similar to the encryption process in the reverse order. Each round consists of the four processes conducted in the reverse order –

1. Add round key
2. Mix columns
3. Shift rows
4. Byte substitution



1. **Add round key**

The 16 bytes of the matrix are now considered as 128 bits and are XORed to the 128 bits of the round key.

If this is the last round then the output is the plain text. Otherwise, the resulting 128 bits are interpreted as 16 bytes and we begin another similar round.

2. **Inverse MixColumns**

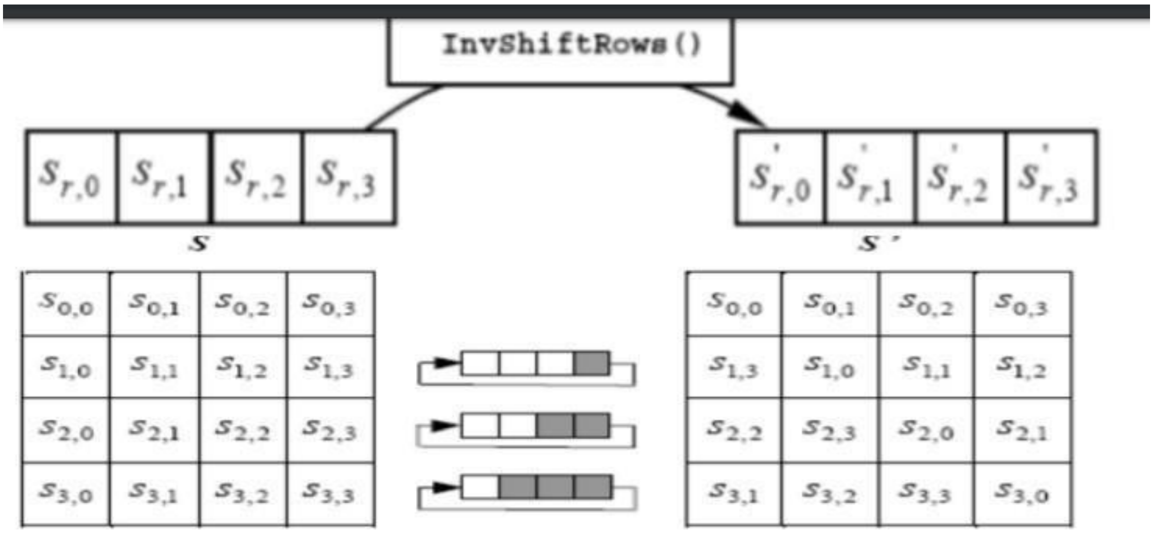
Inverse Mix Columns is the inverse of the Mix Columns transformation. Inverse Mix Columns operates on the State column-by-column, treating each column as a four-term polynomial.

The columns areconsidered as polynomials over GF(2^8) and multiplied modulo x^ 4 + 1 with a fixed polynomial (x), given by a^-1 (x) = {0b}x^3 + {0d}x^ 2 + {09}x + {0e}

3. **Inverse Shift Rows**

Inverse Shift Rows is the inverse of the Shift Rows transformation. The bytes in the last three rowsof 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.

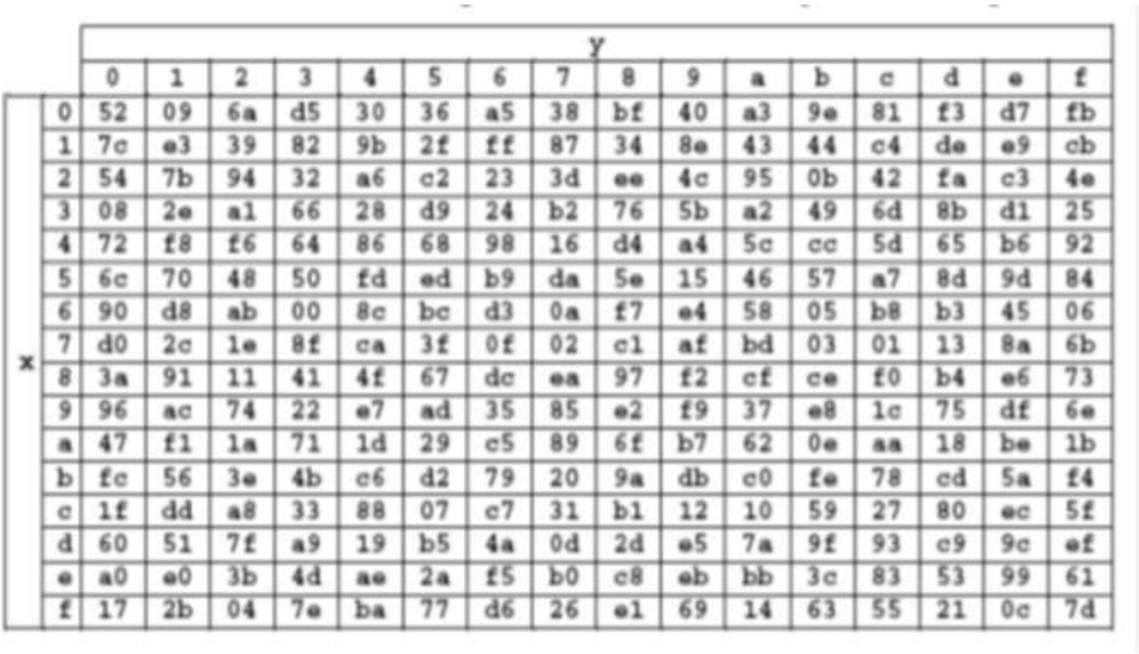


InvShiftRows()
$s_{r,0}$ $s_{r,1}$ $s_{r,2}$ $s_{r,3}$
s
$s'_{r,0}$ $s'_{r,1}$ $s'_{r,2}$ $s'_{r,3}$
s'
$s_{0,0}$ $s_{0,1}$ $s_{0,2}$ $s_{0,3}$
$s_{1,0}$ $s_{1,1}$ $s_{1,2}$ $s_{1,3}$
$s_{2,0}$ $s_{2,1}$ $s_{2,2}$ $s_{2,3}$
$s_{3,0}$ $s_{3,1}$ $s_{3,2}$ $s_{3,3}$
$s_{0,0}$ $s_{0,1}$ $s_{0,2}$ $s_{0,3}$
$s_{1,3}$ $s_{1,0}$ $s_{1,1}$ $s_{1,2}$
$s_{2,2}$ $s_{2,3}$ $s_{2,0}$ $s_{2,1}$
$s_{3,1}$ $s_{3,2}$ $s_{3,3}$ $s_{3,0}$

4. Inverse Bytes Substitution

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.

This is obtained by applying the inverse of the affine transformation followed by taking the multiplicative inverse inGF (2^8). There is an inverse s-box table for substitute the value.

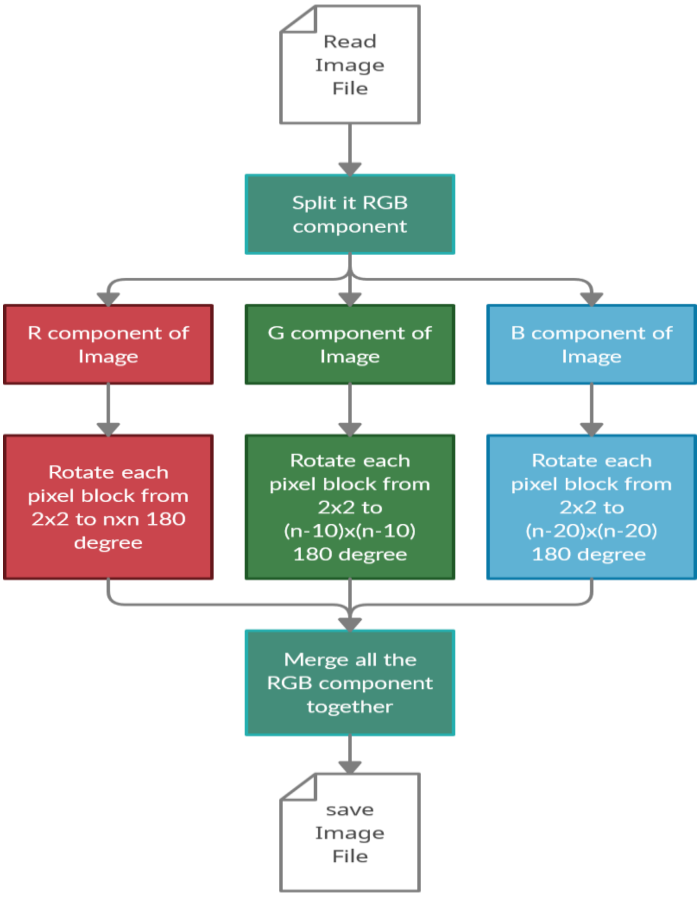


		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
	1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
	2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
	3	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
	4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
	5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
	6	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
	7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
	8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
	9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
	a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
	b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
	c	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
	d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
	e	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
	f	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

2.2.2 Position Permutation Based Algorithm

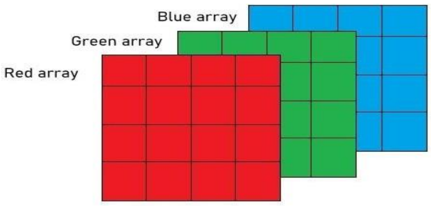
RGB Shuffling Method

In this technique we will rearrange pixels in image so it can't be recognized.
We will be breaking the image into 2x2 pixels blocks and rotate each block by 180 degrees, then we will do the same with 3x3 pixel blocks, up to n block size. Then we will do the same for n-1, then n-2, all the way back down to 2x2.


Read Image File
Split it RGB component
R component of Image
G component of Image
B component of Image
Rotate each pixel block from 2x2 to nxn 180 degree
Rotate each pixel block from 2x2 to (n-10)x(n-10) 180 degree
Rotate each pixel block from 2x2 to (n-20)x(n-20) 180 degree
Merge all the RGB component together
save Image File

This algorithm will require 3 steps

Step 1: Read image and split it into its RGB components



Arrrays stacked over each other
to form a Digital Image.

Step 2: For each component load its respective array and then perform 180 degree rotation for 2x2 to nxn pixel block and then for n-1,n-2…3.

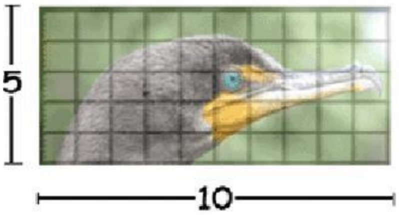
1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

6	7	8	9	5
1	2	3	4	10
16	17	18	19	15
11	12	13	14	20
21	22	23	24	25

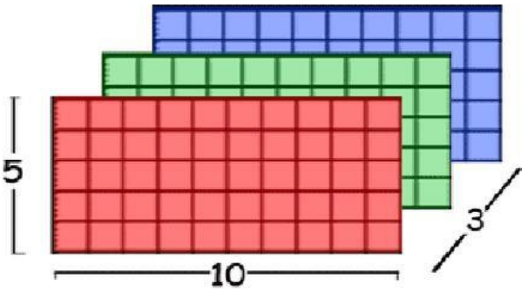
6	7	8	19	5
11	12	13	14	10
16	17	18	9	15
1	2	3	4	20
21	22	23	24	25

16	17	18	9	5
1	2	3	4	10
6	7	8	19	15
11	12	13	14	20
21	22	23	24	25

Step 3: after rotating every component merge all the component together



Original Color Image



Matlab RGB Matrix

CHAPTER 3

SNAPSHOT OF PROJECT

We will present our results for the objective -Image encryption and decryption.:

3.1 GUI framework of the Project

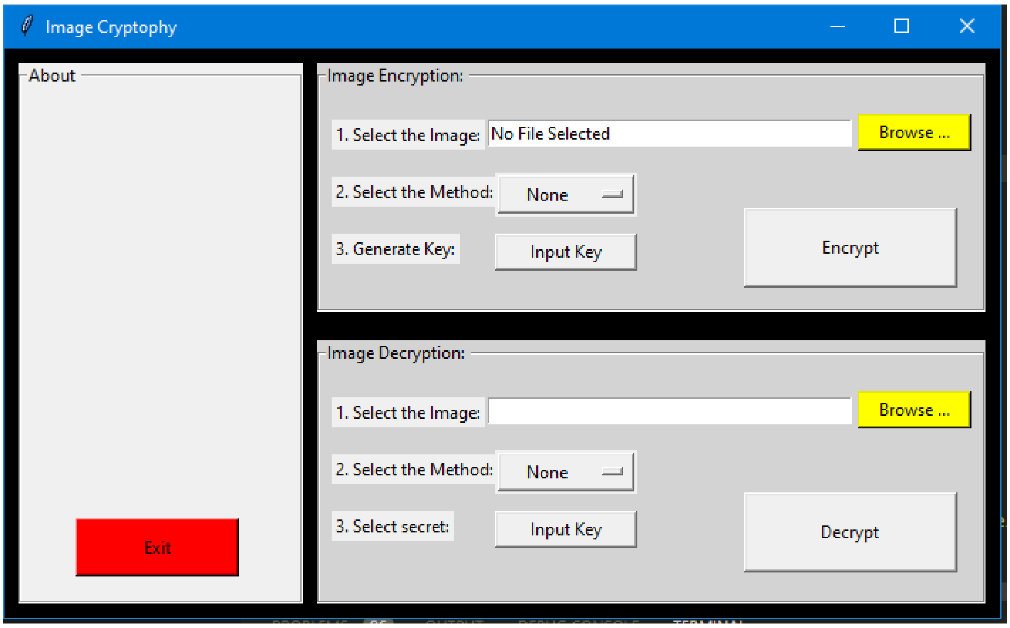


Image Encryption:

1. Select the Image: No File Selected
2. Select the Method: None
3. Generate Key:

Image Decryption:

1. Select the Image:
2. Select the Method: None
3. Select secret:

3.2 Selecting the Algorithm/Method For Encryption/Decryption:

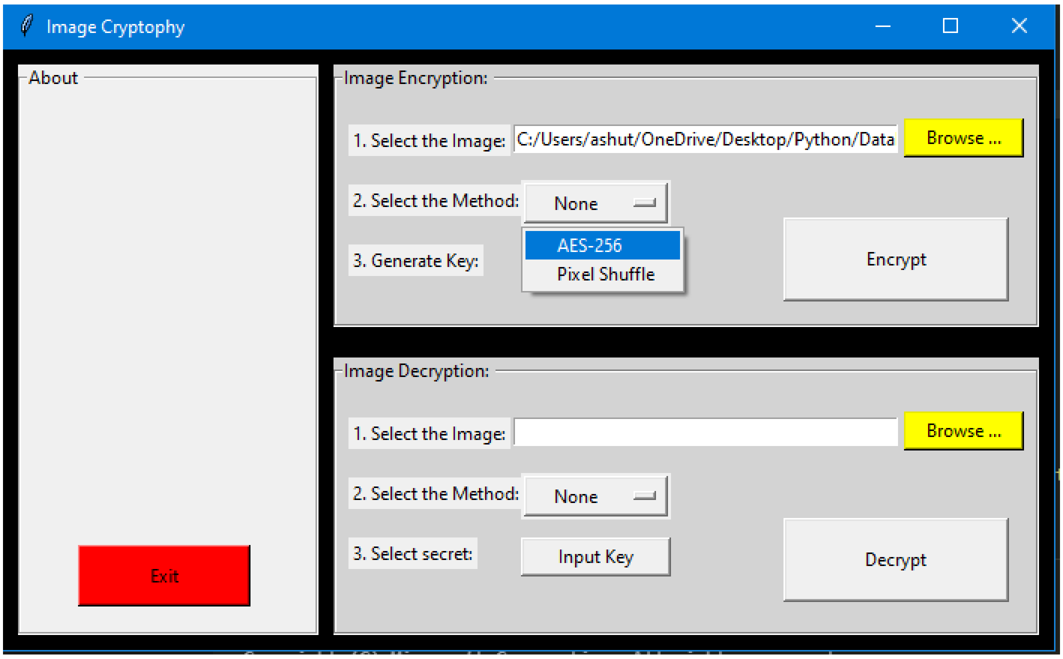


Image Encryption:

1. Select the Image: C:/Users/ashut/OneDrive/Desktop/Python/Data
2. Select the Method: None
3. Generate Key:

Image Decryption:

1. Select the Image:
2. Select the Method: None
3. Select secret:

3.3 Inserting Key

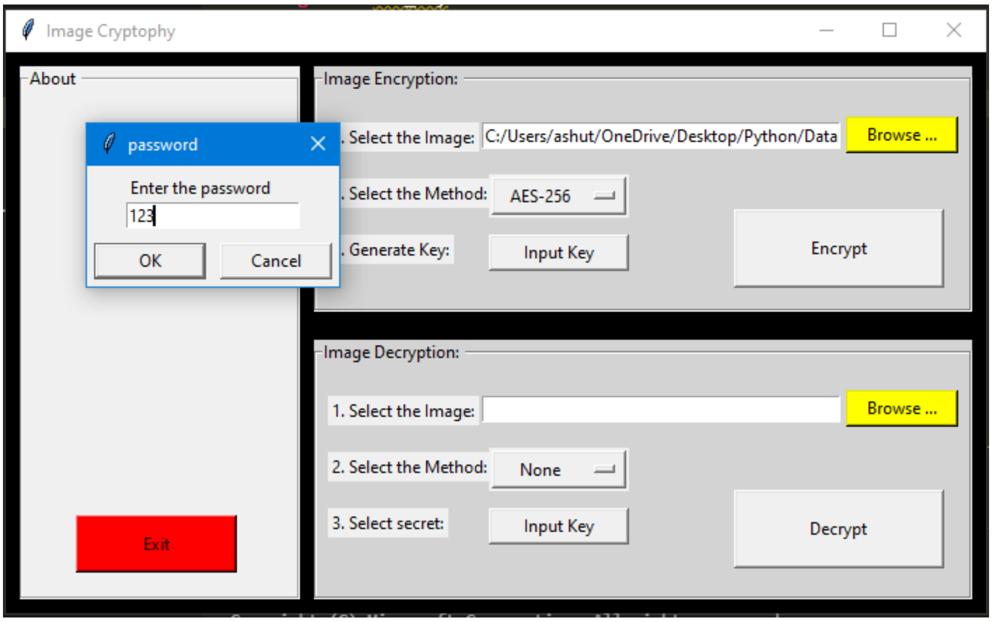


Image Cryptophy

password

Enter the password

123

Image Encryption:

. Select the Image: C:/Users/ashut/OneDrive/Desktop/Python/Data

. Select the Method: AES-256

. Generate Key:

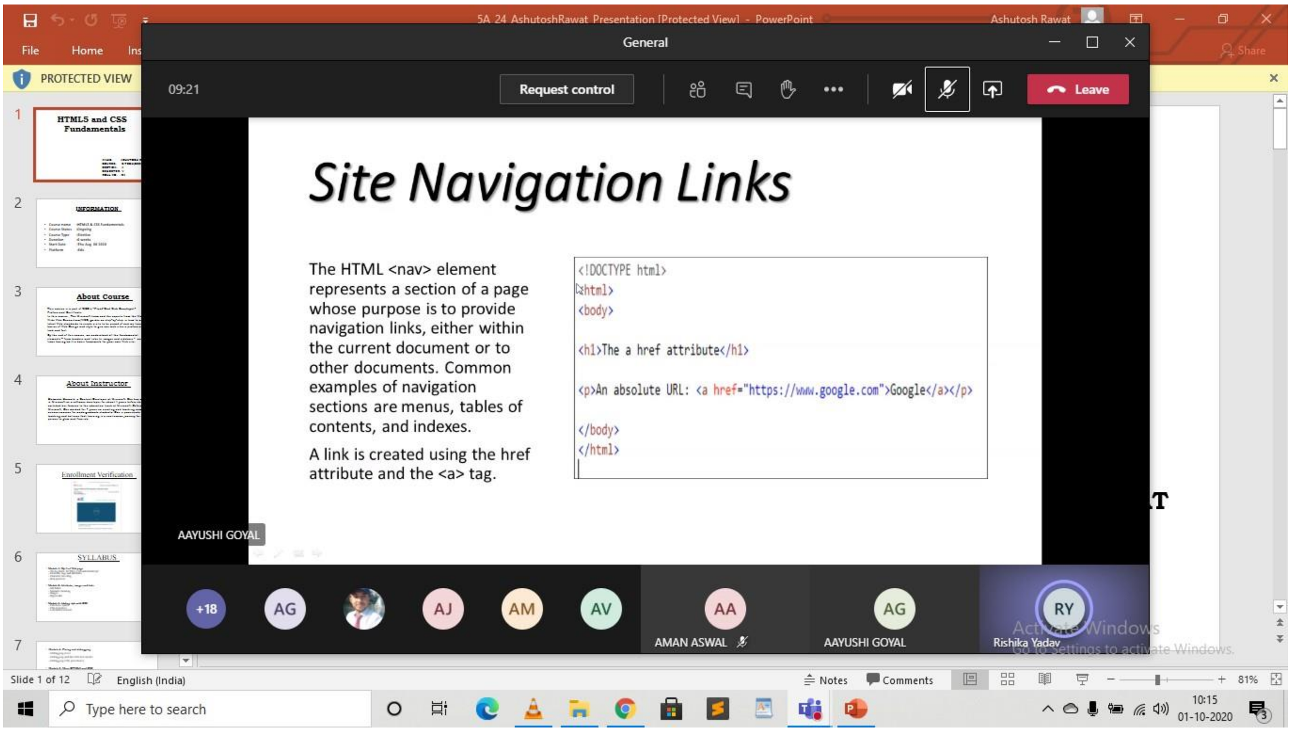
Image Decryption:

1. Select the Image:

2. Select the Method: None

3. Select secret:

3.4 Original Image:



Site Navigation Links

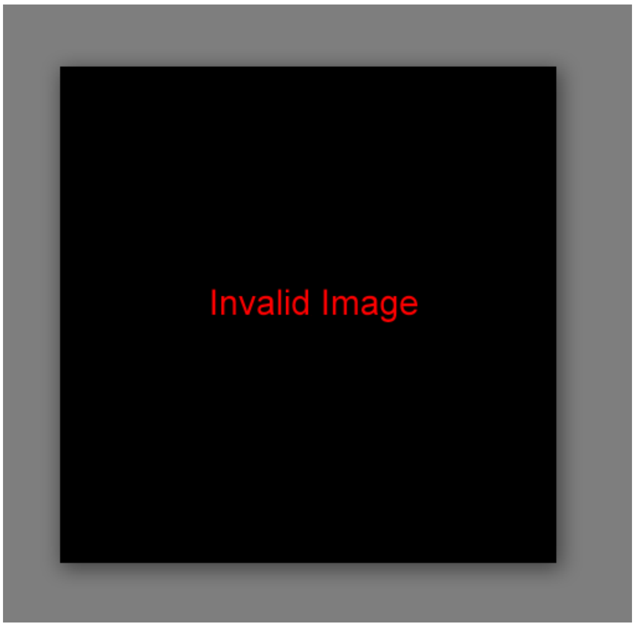
The HTML <nav> element represents a section of a page whose purpose is to provide navigation links, either within the current document or to other documents. Common examples of navigation sections are menus, tables of contents, and indexes.

A link is created using the href attribute and the <a> tag.

```
<!DOCTYPE html>
<html>
<body>

<h1>The a href attribute</h1>

<p>An absolute URL: <a href="https://www.google.com">Google</a></p>

</body>
</html>
```

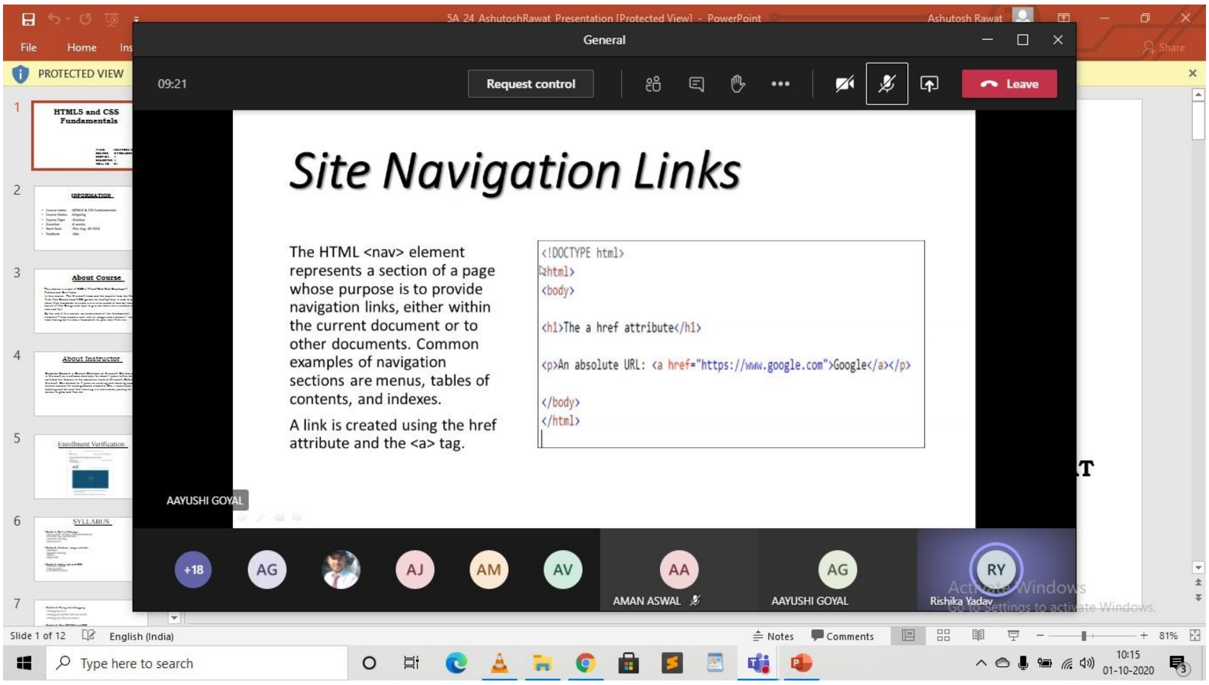
3.5 Using AES Algorithm/Method For Encryption/Decryption:

3.5.1 Encrypted Image:



Invalid Image

3.5.2 Decrypted Image:



Site Navigation Links

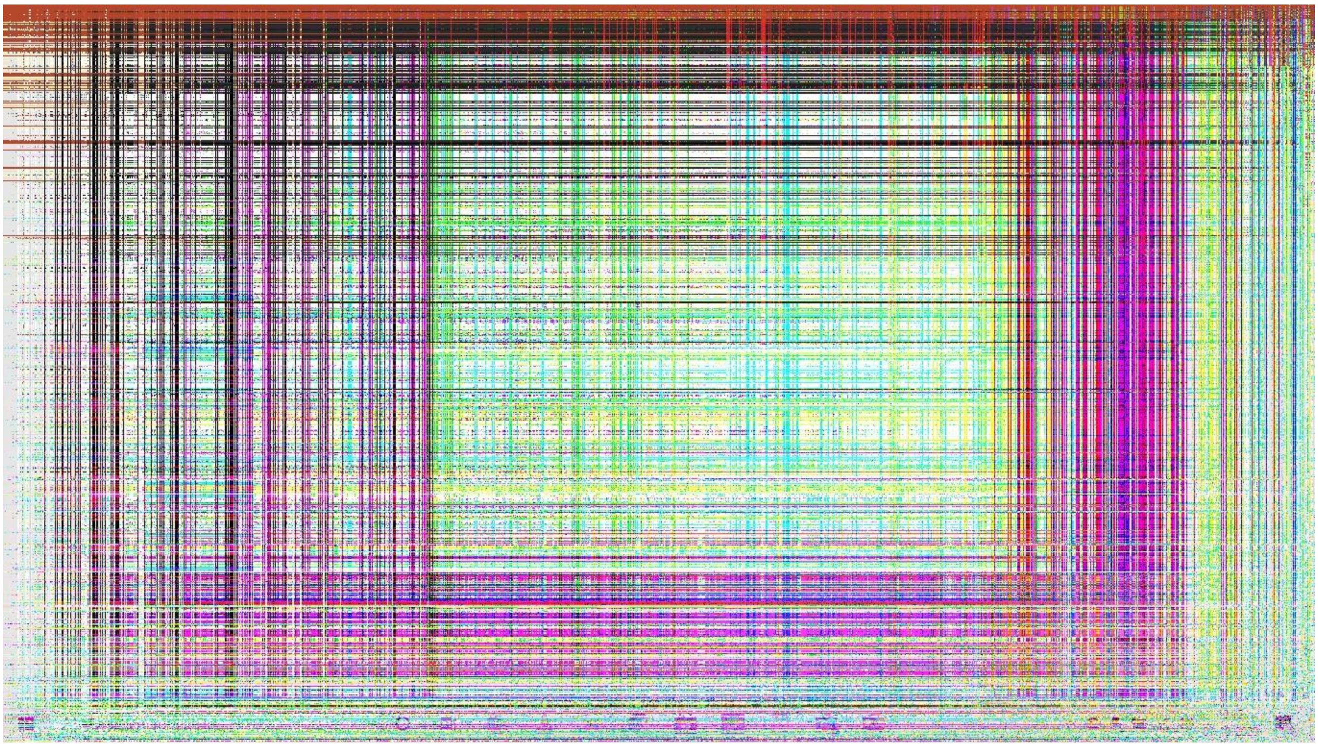
The HTML <nav> element represents a section of a page whose purpose is to provide navigation links, either within the current document or to other documents. Common examples of navigation sections are menus, tables of contents, and indexes.

A link is created using the href attribute and the <a> tag.

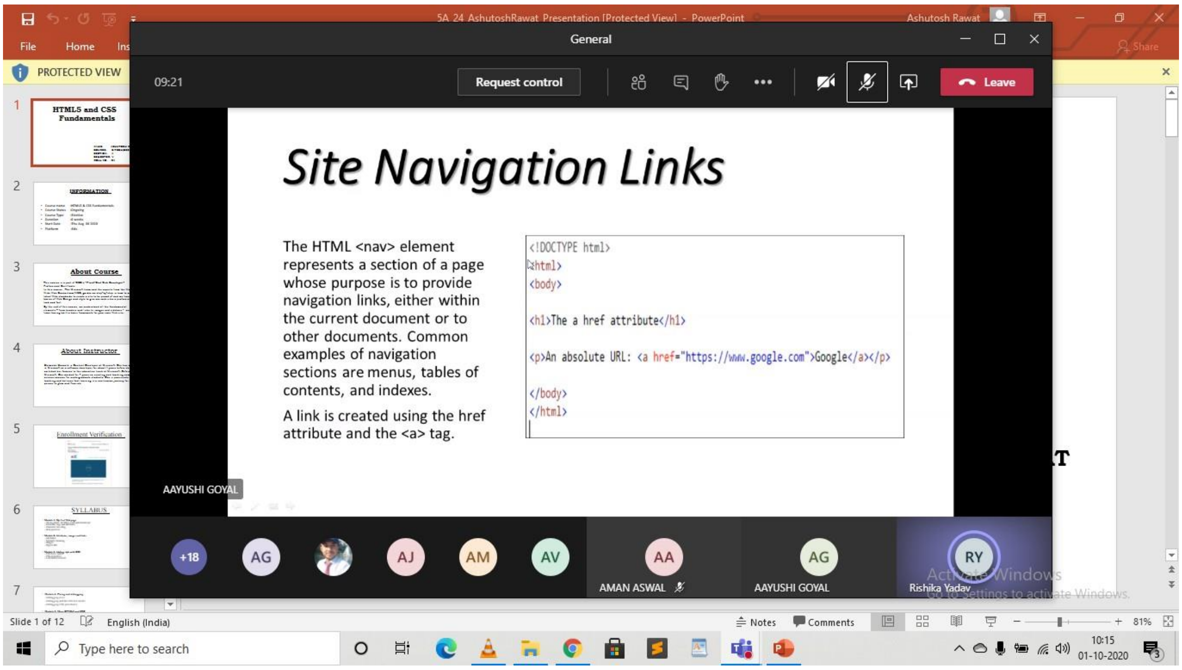
```
<!DOCTYPE html>
<html>
<body>

<h1>The a href attribute</h1>

<p>An absolute URL: <a href="https://www.google.com">Google</a></p>

</body>
</html>
```

3.6 Using Pixel Shuffle/RGB shuffle for Encryption/Decryption:

3.6.1 Encrypted image:



3.6.2 Decrypted image:



Site Navigation Links

The HTML <nav> element represents a section of a page whose purpose is to provide navigation links, either within the current document or to other documents. Common examples of navigation sections are menus, tables of contents, and indexes.

A link is created using the href attribute and the <a> tag.

```
<!DOCTYPE html>
<html>
<body>

<h1>The a href attribute</h1>

<p>An absolute URL: <a href="https://www.google.com">Google</a></p>

</body>
</html>
```

CHAPTER 4

CONCLUSION

4.1 CONCLUSION

Thus the project entitled “Image Encryption and Decryption “ was successfully completed. We created a standalone application which consist of a GUI framework which helps in encrypting and decrypting images using following algorithms:

1. **ADVANCE ENCRYPTING STANDARD(AES):-**
   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 standard available in market. With the help of python coding implementation of an AES algorithm is synthesized and simulated for Image Encryption and Decryption. The original images can also be completely reconstructed without any distortion.
2. **PIXLE SHUFFLE/RGB SHUFFLE:-**
   The transposition and reshuffling of the RGB values of the image in steps has proven to be really effective in terms of the security analysis.
   The extra swapping of RGB values in the image file after R G B component shifting has increased the security of the image against all possible attacks available currently

4.2 FUTURE SCOPE

The research paper proposes a system which could be used for effective image data encryption and key generation in diversified application areas, where sensitive and confidential data needs to be transmitted along with the image. The next step in this direction will be system implementation, calculating time and space complexity for the same using some experimental data and then comparing it with existing algorithms and schemes for its efficiency, accuracy and reliability

We are also looking forward to encrypt videos by extracting each frame and encrypting the images simultaneously

This is future of our project we are looking at and looking forward to implementing all of the above successfully

REFERENCE

[1] William Stallings, “Advance Encryption Standard,” in Cryptography and Network Security,4th Ed., India:PEARSON,pp. 134–165.

[2] AtulKahate, “Computer-based symmetric key cryptographic algorithm”, in Cryptography andNetwork Security, 3th Ed. New Delhi:McGraw-Hill, pp. 130-141.

[3] Manoj .B,Manjula N Harihar (2012, June). “Image Encryption and Decryption using AES”, International Journal of Engineering and Advance Technology (IJEAT) volume-1, issue-5, pp.290-294.

[4] KundankumarRameshwarSaraf, Vishal PrakashJagtap, Amit Kumar Mishra, (2014, May-June).”Text and Image Encryption Decryption Using Advance Encryption Standard”, InternationalJournal of Emerging Trends and Technology in computer science(IJETTCS) volume-3, issue-3, pp. 118-126.

[5] VedkiranSaini, ParvinderBangar, Harjeet Singh Chauhan, (2014, April).”Study and Literature Survey of Advanced Encryption Algorithm for Wireless Application”, InternationalJournal of Emerging Science and Engineering (IJESE) volume-2, issue-6, pp.33-37.

[6] Sourabh Singh, Anurag Jain, (2013, May). “An Enhanced Text to Image Encryption Technique using RGB Substitution and AES”, International Journal of Engineering Trends andTechnology (IJETT) volume-4,issue-5,pp.2108-2112.