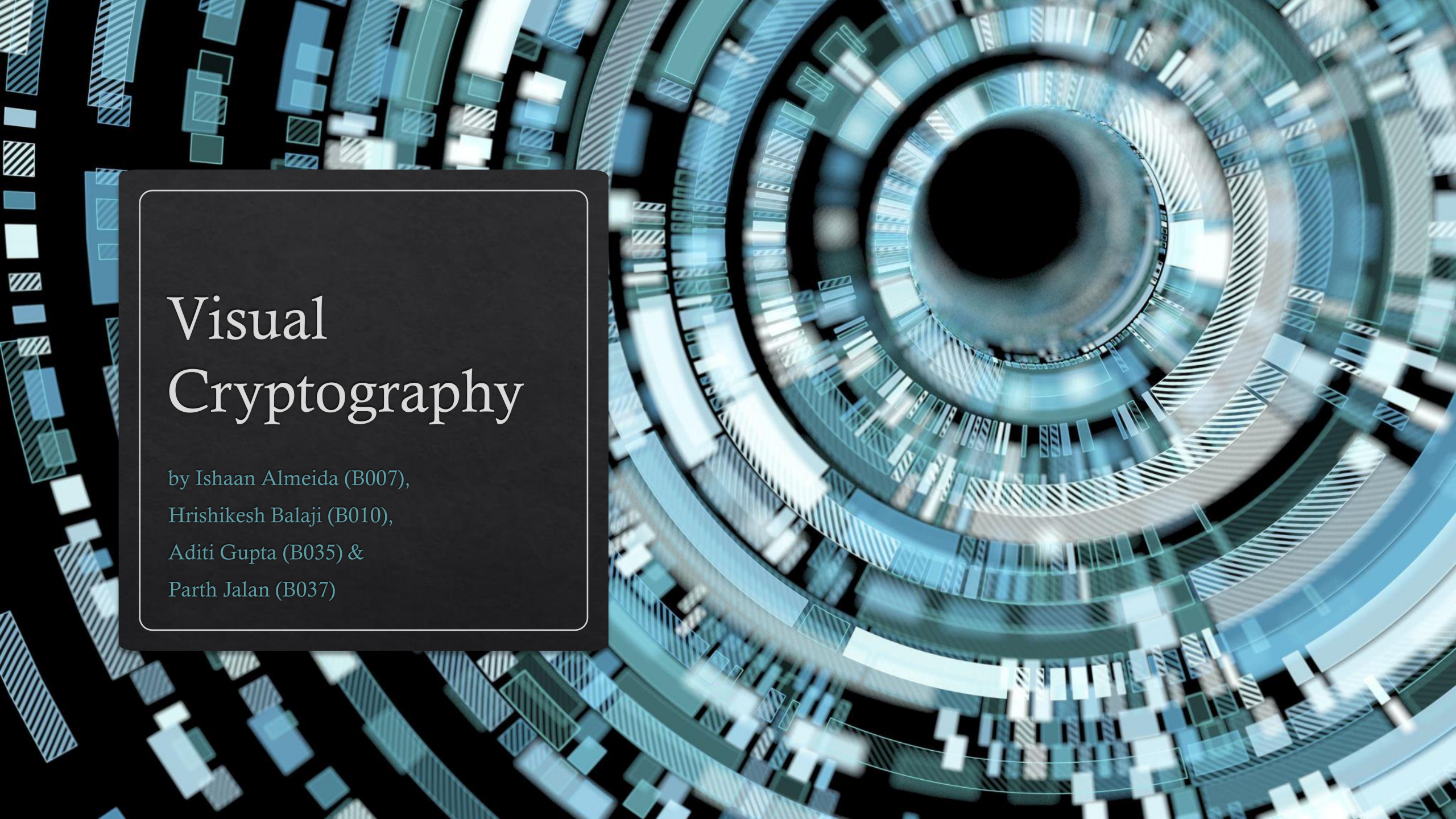


Visual Cryptography

by Ishaan Almeida (B007),
Hrishikesh Balaji (B010),
Aditi Gupta (B035) &
Parth Jalan (B037)

Why we need it?

- 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.

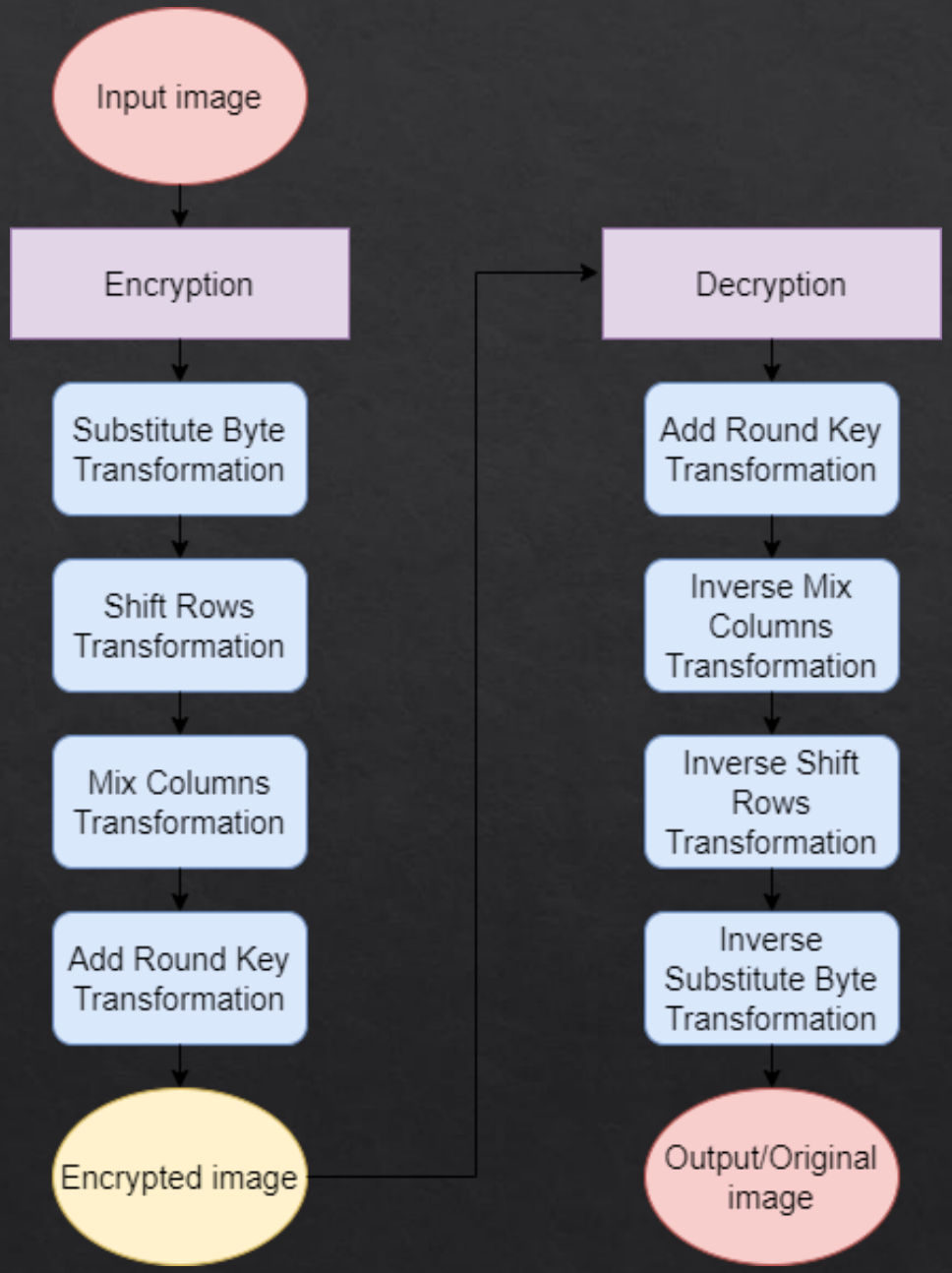
What is Visual Cryptography?

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

- 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.



Pre-processing

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

Encryption

Substitute Byte Transformation

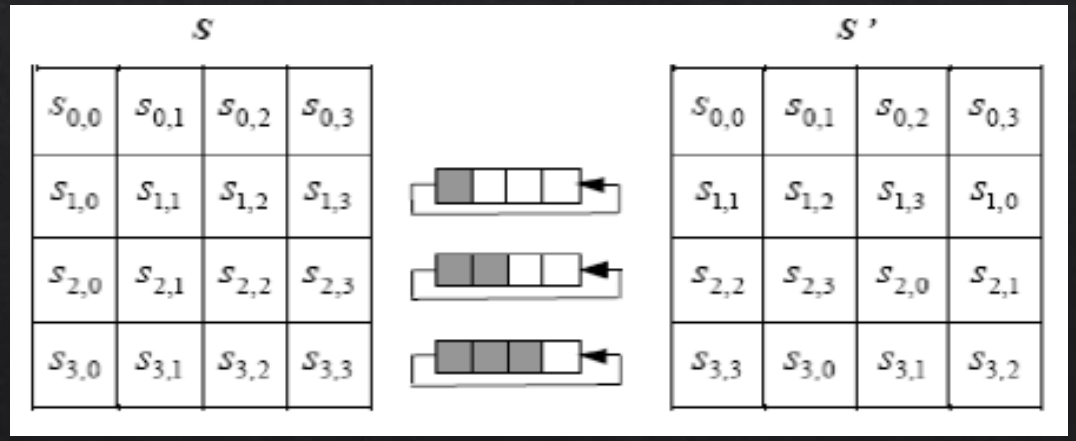
- It is a non-linear byte substitution that operates independently on each byte of the State using a substitution table S-box.

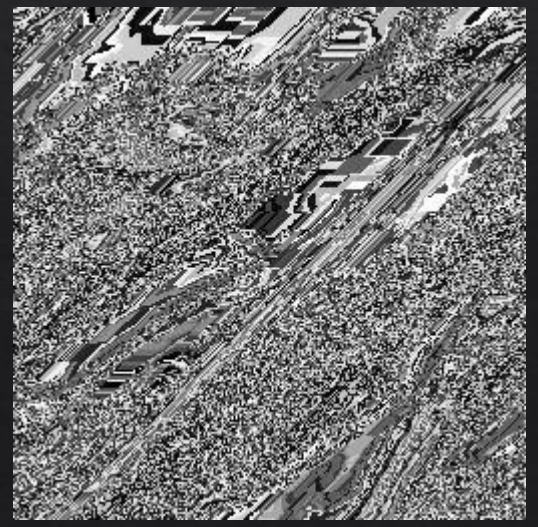
	0x	x1	x2	x3	x4	x5	x6	x7	x8	x9	xA	xB	xC	xD	xE	xF
0x	0x63	0x7c	0x77	0x7b	0xf2	0x6b	0x6f	0xc5	0x30	0x01	0x67	0x2b	0xfe	0xd7	0xab	0x76
1x	0xca	0x82	0xc9	0x7d	0xfa	0x59	0x47	0xf0	0xad	0xd4	0xa2	0xaf	0x9c	0xa4	0x72	0xc0
2x	0xb7	0xfd	0x93	0x26	0x36	0x3f	0xf7	0xcc	0x34	0xa5	0xe5	0xf1	0x71	0xd8	0x31	0x15
3x	0x04	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

Encryption

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.





Encryption

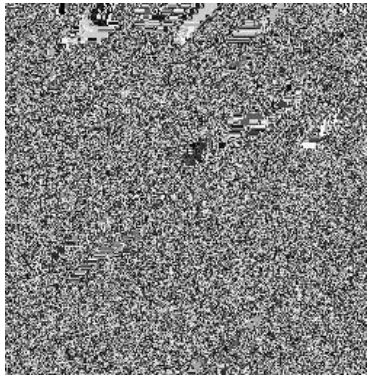
Mix Columns Transformation

- 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):

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

[redacted]

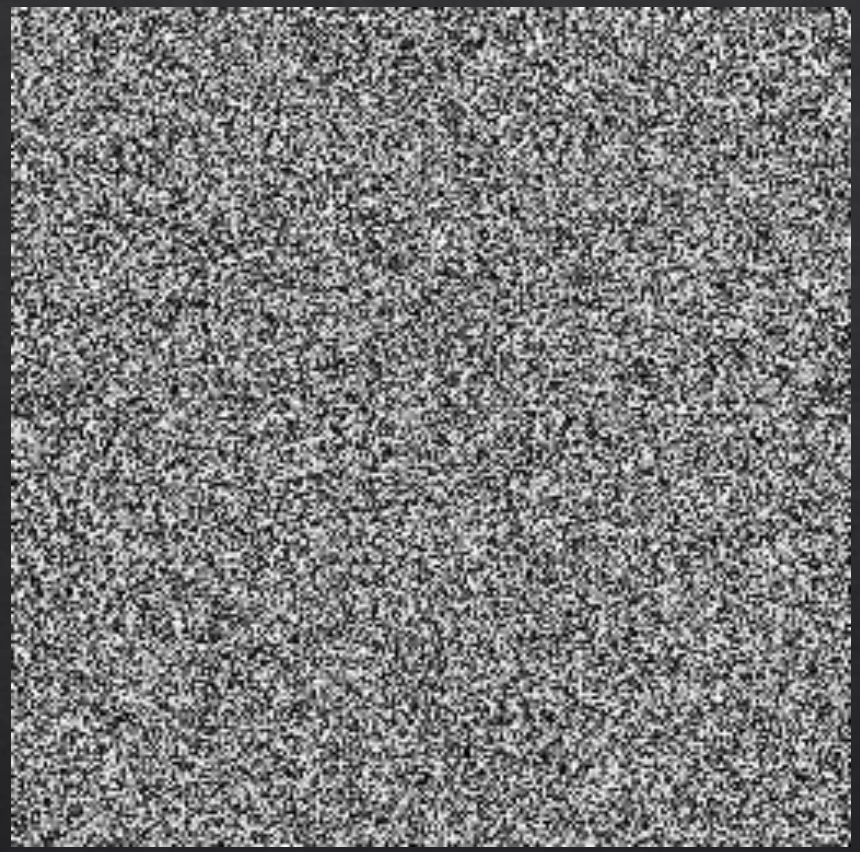
[redacted]



Encryption

Add Round Key Transformation

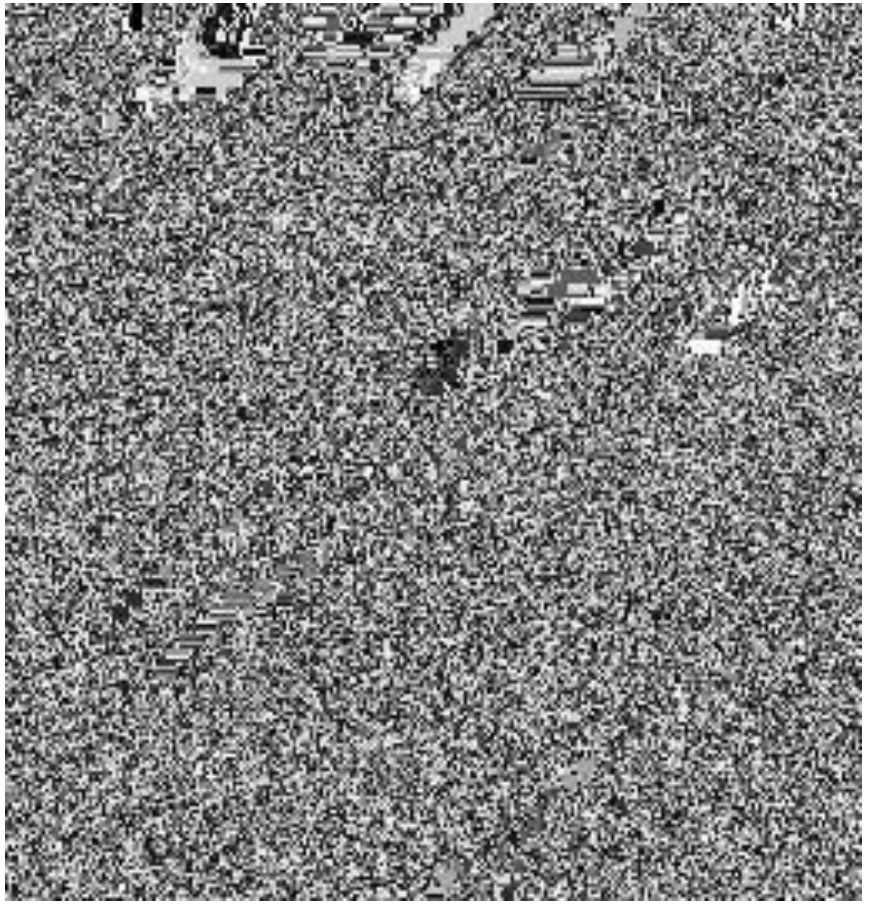
- A Round Key is added to the State by a simple bitwise XOR operation.
- The Round Key is derived from the Cipher key by means of key schedule process.
- The State and Round Key are of the same size and to obtain the next State an XOR operation is done per element: $b(i, j) = a(i, j) \oplus k(i, j)$



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: $b(i, j) = a(i, j) \oplus k(i, j)$.



Decryption

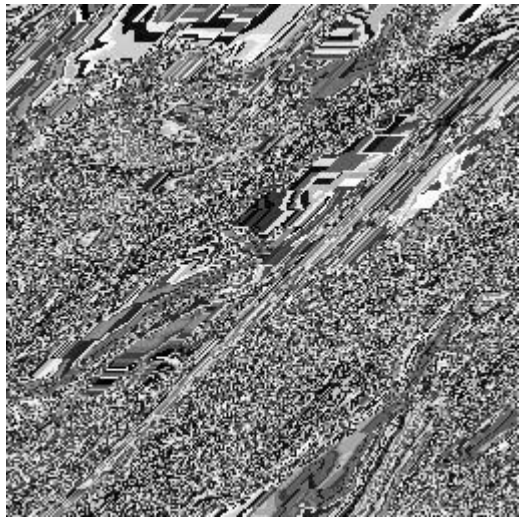
Inverse Mix Columns Transformation

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

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

[redacted]

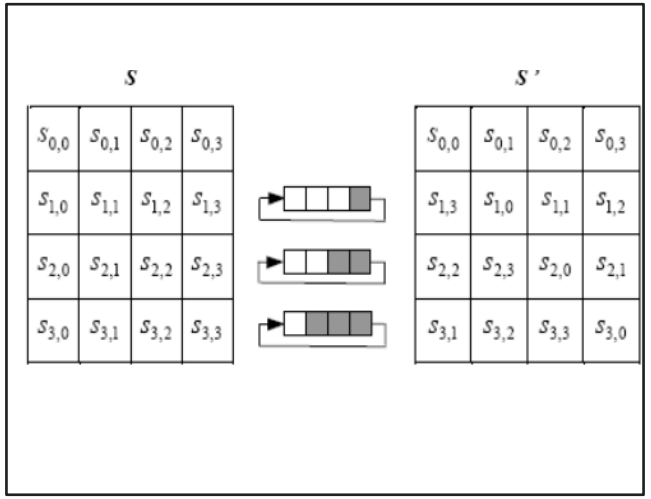
[redacted]



Decryption

Inverse Shift Rows Transformation

- It 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.

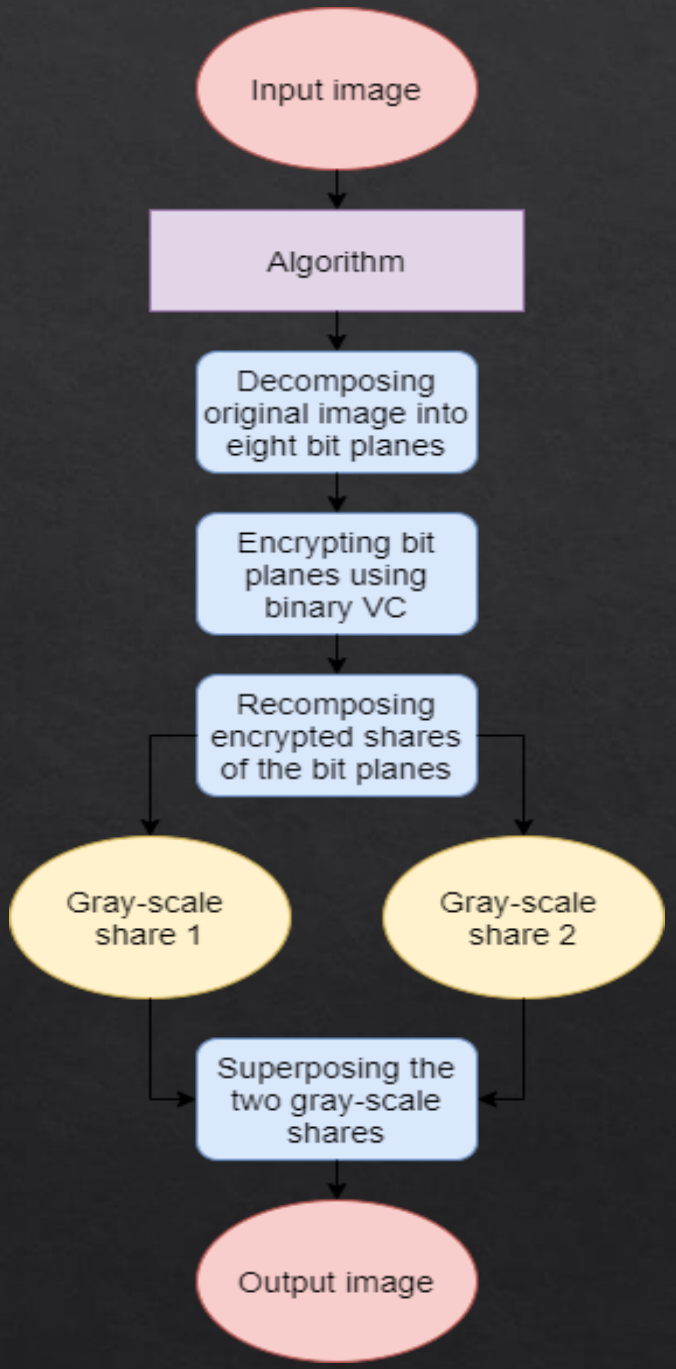


Decryption

Inverse Substitute Byte Transformation

- It is the inverse of the byte substitution transformation, in which the inverse S-box is applied to each byte of the State.

	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



Visual Cryptography for Gray-scale Images Using Bit-level

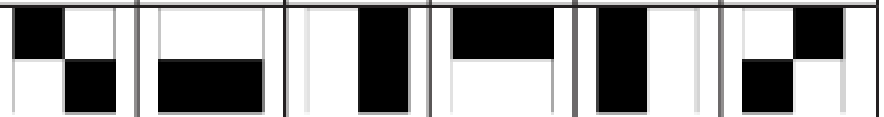
- It is an image cryptographic scheme in which a secret image is encrypted into two separate share images.
- Each share individually reveals no information about the secret, but when shares are superposed the secret is revealed.
- We use bit-level decomposition to extract binary bit planes from a gray-scale 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.

Pre-processing

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

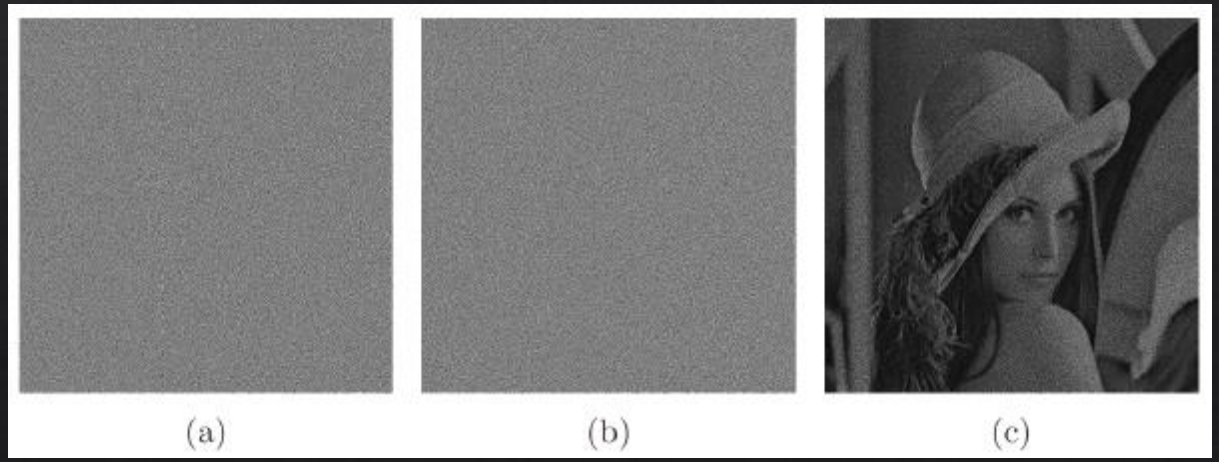
Blocks used in Shares and Stacking Results for Binary Images (Binary VC)

- Encrypted images are generated block by block corresponding to each pixel in the original image.
- 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.

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;

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 x 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.

Thank You