"ADVANCE ENCRYPTION STANDARD USING PYTHON"

KURALLA SRAVYA SAI SREE

A Report submitted

in partial fulfillment for the Degree of

B. Tech

in

DS&AI

By

KURALLA SRAVYA SAI SREE

Pursued in

Department Of DSAI

Faculty of Science and Technology
ICFAI Foundation for Higher Education
(A Deemed University under Section 3 of UGC Act, 1956)
Donthanapally, Shankarapally Road, Hyderabad-501203

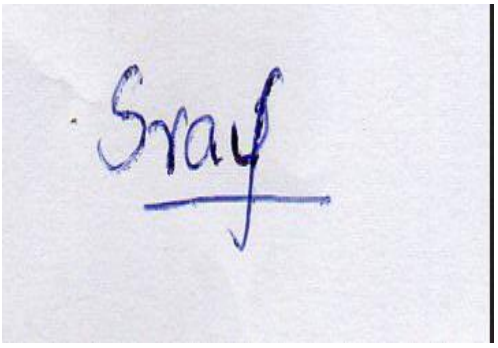
MAY, 2023

CERTIFICATE

This is to certify that the project entitled ADVANCE ENCRYPTION STANDARD USING PYTHON, submitted by **KURALLA SRAVYA SAI SREE**, to **ICFAI Foundation For Higher Education**, Hyderabad, in partial fulfilment for the award of degree of **B.tech** in **Data Science & Artificial Intelligence** is a bona fide record of project work carried out by kuralla sravya sai sree under the super vision of **Dr.B.Deevena Raju**. The contents of this report in full, or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

DECLARATION

I declare that this project report titled **ADVANCE ENCRYPTION STANDARD USING PYTHON**; submitted in partial fulfilment of the degree of **B. Tech in Data Science & Artificial Intelligence** is a record of original work carried out by me under the supervision of **Dr.B. Deevena Raju**, and has not formed the basis for the award of any other degree or diploma, in this or any other Institution or University. In keeping with the ethical practice in reporting scientific information, due acknowledgements have been made wherever the findings of others have been cited.



<Signature>

KURALLA SRAVYA SAI SREE

20STUCHH010265

ACKNOWLEDGMENT

I take this opportunity to thank Dr.B. Deevena Raju, Director- IcfaiTech Dr. K.L.Narayana, and other faculty members who helped in preparing the guidelines.

I extend my sincere thanks to one and all of IcfaiTech family for the completion of this document on the project report format guidelines.

NAME:KURALLA SRAVYA SAI SREE

ABSTRACT

Advanced Encryption Standard (AES) algorithm is one on the most common and widely symmetric block cipher algorithm used in worldwide. This algorithm has an own particular structure to encrypt and decrypt sensitive data and is applied in hardware and software all over the world. It is extremely difficult to hackers to get the real data
when encrypting by AES algorithm. Till date is not any evidence to crake this algorithm. AES has the ability to deal with three different key sizes such as AES 128, 192 and 256 bit and each of this ciphers has 128 bit block size.
The approach introduced in this project makes use of both steganographic as well as cryptographic techniques. In Cryptography we are using RSA. In Steganography we are using Image Steganography for hiding the data. And we also use Mutual Authentication process to satisfy all services in Cryptography i.e., Access Control, Confidentiality, Integrity, Authentication. In this way we can maintain the data more securely. Since we use RSA algorithm for securing the data and again on this we perform Steganography to hide the data in an image. Such that any other person in the network cannot access the data present in the network. Only the sender and receiver can retrieve the message from the data.

keywords : **Rivest-Shamir-Adelman(RSA), Crptography, Steganography**

TABLE OF CONENTS:-

FIGURES

CHAPTER 1

INTRODUCTION

. In this project Encryption is the conversion of data into a form, called a cipher text that cannot be easily understood by unauthorized people. Decryption is the process of converting encrypted data back into its original form, so it can be understood. The use of encryption/decryption is as old as the art of communication. In wartime, a cipher, often incorrectly called a "code," can be employed to keep the enemy from obtaining the contents of transmissions. (Technically, a code is a means of representing a signal without the intent of keeping it secret; examples are Morse code and ASCII.) Simple ciphers include the substitution of letters for numbers, the rotation of letters in the alphabet, and the "scrambling" of voice signals by inverting the sideband frequencies. More complex ciphers work according to sophisticated computer a algorithm that rearranges the data bits in digital signals.

1.1 **MOTIVATION:** The existing system based on AES comprises two block ciphers: AES-128, AES-192 .There is only possibility hiding a secret information in and text ,audio ,vedio .It is not available to secure the data in image format. The proposed system based on AES comprises two block ciphers: AES-128, AES-192 ,AES-256The image is more secure than DES and triple DES because it is encrypted using AES.It is more secured and it used for image ,texrt,audio and vedio.

1.2 **OBJECTIVE OF THE PROJECT:** In Order to be able to define our system architecture, we must first dearly state what our objective that will deriver system behavior at the same one of our objective is to create an experience, which is not only unique to the (user) client, but also makes him feel that he has loyal attachment to the system and approaches us whenever he/she needs. To achieve better results and success by implement computerized process instead of manual process.

1.3 **PROBLEM STATEMENT**: The purpose of this project is to provide the correct data with security to the users. For some of the users the data might be lost during the transmission process in the network and for some, the data might be changed by the unauthorized person in the network and there are some other security problems in the network. My application will give you more Security to the data present in the network and there will be able to reduce the loss of data in the network which will be transmitted from the sender to the receiver using the latest technologies.

The proposed algorithm is to hide the audio data effectively in an image without any suspicion of the data being hidden in the image. This algorithm, though requires a distinct image which we can use as a carrier and hide the data which is well within the limits of the threshold that the image can hide, that will secure the data.

CHAPTER 2

LITERATURE REVIEW:

Aparna, V. S., et al.in [1], "Implementation of AES Algorithm on Text And Image using MATLAB", Proposed an encryption algorithm for the secure transmission of data. Advanced Encryption Standard (AES) a symmetric block cipher of 128-bits that uses the same key for encryption as well as for decryption is used. Here encryption and decryption are done on character message, string-text message, and image message. Plain text is inputted to encryption algorithm and output is an encrypted message i.e. ciphertext, then this ciphertext is given to decryption algorithm to get the decrypted message where plain text is reconstructed. This algorithm is highly efficient as decrypted output is the same as the input and there is no distortion in the output.

Yashpal Lather et al. in [2]," Review Paper on Steganography Techniques" discuss different steganography techniques their uses and limitation. Also, provide the difference between steganography and cryptography. Text, as well as different image steganography techniques, is mention in this paper. Lastly, it concluded that steganography that uses a key has better security than non-key steganography

Arnold Gabriel Benedict et al. in [3], "Improved File Security System Using Multiple Image Steganography", proposed a slicing method where the secret data is sliced and stored on multiple cover images. The Least significant bit of all the selected cover image pixel values is used to hide the data this technique is called LSB based steganography technique. Payload which is a set of files that is to be hidden inside the cover file, are compressed using the ZIP compression algorithm. Image hashing algorithm ensures a random distribution of bits from compressed payload file; it has high latency in analyzing slicing pattern which makes it more difficult for the intruder to decrypt the pattern. Camouflage capacity or the capacity for hiding secret data in the cover image can be identified. Decoding follows equivalent steps as in encoding.

Karolin., et al.in [4], "Encryption and decryption of color images using visual cryptography" proposed a visual cryptography technique that allows digital images to be divided into multiple numbers of printable shares called transparent shares and transmitted physically by printing these shares on transparency sheets to the authorized users. Visual cryptography works on many forms of images such as grayscale images, black and white images as well as color images. Visual cryptography consists of three phases for color images. The first phase is to realize the color and print the color in the secret image on the shares directly. The second

phase converts a color image into a black and white image; the third phases utilize the binary representation of the color of a pixel and encode the secret image at the bit-level. Computational complexity of traditional cryptography is overcome here. Blowfish algorithm is a 64-bit block cipher with key values in the range 32 to 448 is used for securing the image.

Al-Haj., et al.in [5], "Digital image security based on data hiding and cryptography", proposed a hybrid algorithm that applied cryptography and watermarking to provide security to the medical images being transfer. This algorithm makes use of bit planes where it combines two images each consisting of 8-bit planes in a single image consisting of 16-bit planes. The first image of the 8-bit plane is watermarked using the RDH histogram shift method a copy of this is save and on the other hand encryption is performed on it and further, it is watermarked using the RDH histogram shift method. The combining process of two images takes place that is watermarked images and encrypted watermarked image each having 8-bit planes. This algorithm can be applied effectively to medical images of different modalities like CT, MRI, Ultrasound, and X-RAY. Using this hybrid approach its embedding capacity is increased.

CHAPTER 3

3.1) MODULES AND DESCRIPTION

Advanced Encryption Standard
Encryption
Decryption

AES (Advanced Encryption Standard)

The Advanced Encryption Standard (AES) was published by the National Institute of Standards and Technology (NIST) in 2001. AES is a symmetric block cipher that is intended to replace DES as the approved standard for a wide range of applications. Compared to public-key ciphers such as RSA, the structure of AES and most symmetric ciphers is quite complex and cannot be explained as easily as many other cryptographic algorithms. Accordingly, the reader may wish to begin with a simplified version of AES, which is described in Appendix I. This version allows the reader to perform encryption and decryption by hand and gain a good understanding of the working of the algorithm details. Classroom experience indicates that a study of this simplified version enhances understanding of AES.1 One possible approach is to read the chapter first, then carefully read Appendix I, and then re-read the main body

of the chapter. Appendix H looks at the evaluation criteria used by NIST to select from among the candidates for AES, plus the rationale for picking Rijndael, which was the winning candidate. This material is useful in understanding not just the AES design but also the criteria by which to judge any symmetric encryption algorithm.

In AES, all operations are performed on 8-bit bytes. In particular, the arithmetic operations of addition, multiplication, and division are performed over the finite field GF(28). Discusses such operations in some detail. For the reader who has not studied and as a quick review for those who have, this section summarizes the important concepts.

Encryption

Sender A does the following:-
Obtains the recipient B's public key (n, e).
Represents the plaintext message as a positive integer m
Computes the ciphertext c = me mod n.
Sends the ciphertext c to B.

3.2)Purpose:

The purpose of this project is to provide the correct data with security to the users.
We can use this program in any where like hospitals, schools, colleges, data center to keep the data secured and files to be safe.The image can only be viewed by the receiver as the image is encrypted using AES and the key is only known to the sender and receiver.
Since the image is encrypted using AES, it is more secure than the DES and triple DES

3.3)Existing system

The existing system based on AES comprises two block ciphers: AES-128, AES-192 .There is only possibility hiding a secret information in and text ,audio ,video .
It is not available to secure the data in image format.
The existing system using DES algorithm but I have improved and used AES algorithm which more features are there when compared to AES algorithm.

3.4)Proposed System:

Proposed method which combines two different hiding techniques, which are Cryptography and Steganography. In this proposed method first, the message is encrypted by use RSA algorithm. After that, we use the modified LSB technique to embed the encrypted information in image. So, this

technique combines the features of both cryptography and steganography and provides a high level of security. It is better than either of the technique used separately. There will be an agreement between the sender and the receiver about the key for the concealment algorithm as well as the key for the encryption algorithm or these keys may be exchanged by a secure communication method. Our method starts by encryption first then hide encrypted data.

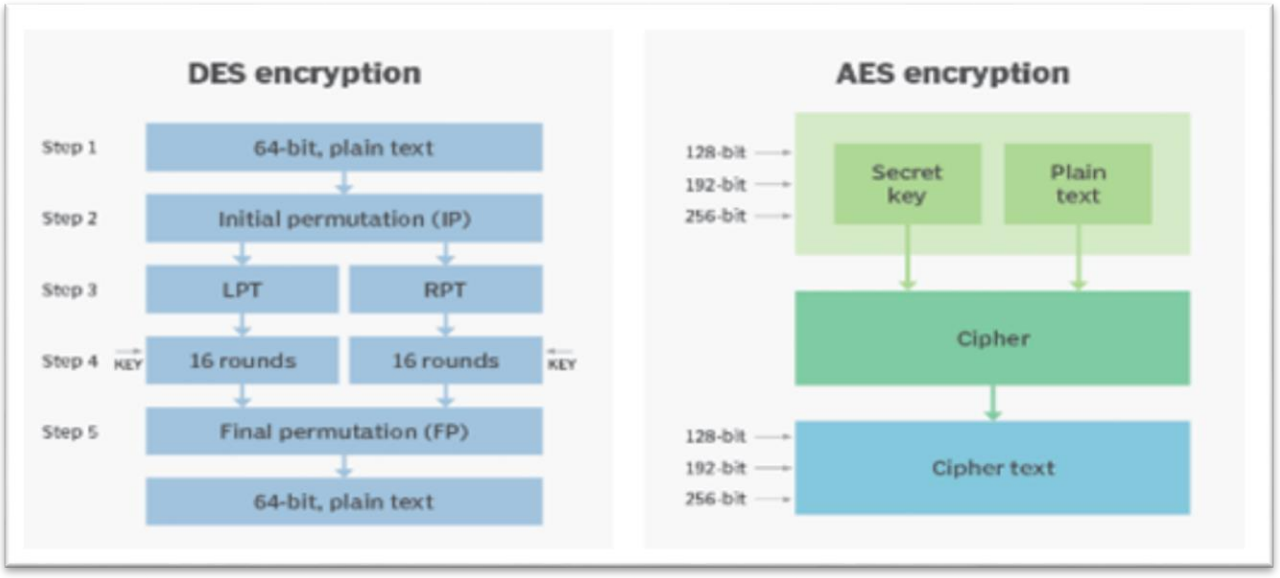
3.5)Scope of Project:

- The scope of the project is to maintain the secure the data.
- we can encrypt the data image, video, PDF, and some other files.
- By encrypting the data the data is secured and the normal person can not understand the files data. We encrypt the data by the password and we again decrypt the data by same password. We use the AES algorithm so they is very chaces to decode the file expect if the password is week.

This project has a broad scope because it includes the following features that make it simple to use, understand, and modify

- The files will be secure and safe.
- Improving the security in the system by providing AES 5algorithm
- It used in hospitals, schools, colleges, offices etc.. for security purpose.
- It will reduce the threat or riskiness

EXISTING AND PROPOSED SYSTEM:



3.6)PROJECT DESCRIPTION:

In order to easily recover the contents of an encrypted signal, the correct decryption key is required. The key is an algorithm that "undoes" the work of the encryption
algorithm. Alternatively, a computer can be used in an attempt to "break" the cipher.
The more complex the encryption algorithm, the more difficult it becomes to eavesdrop on the communications without access to the key. Encryption/decryption is especially important in wireless communications. This is because wireless circuits are easier to "tap" than their hard-wired counterparts. Nevertheless, encryption/decryption is a good idea when carrying out any kind of sensitive transaction, such as a credit-card purchase online, or the discussion of a company secret between different departments in the organization. The stronger the cipher – that is, the harder it is for unauthorized people to break it – the better, in general. However, as the strength of encryption/decryption increases, so does the cost.
In recent years, a controversy has arisen over so-called strong encryption. This refers to ciphers that are essentially unbreakable without the decryption keys. While most companies and their customers view it as a means of keeping secrets and minimizing fraud, some government's view strong encryption as a potential vehicle by which terrorists might evade authorities.

Decryption keys would be stored in a supposedly secure place, used only by authorities, and used only if backed up by a court order. Opponents of this scheme argue that criminals could hack into the key-escrow database and illegally obtain, steal, or alter the keys..

CHAPTER 4

REQUIREMENTS SPECIFICATION:

To be used efficiently, all computer software needs certain hardware components or other software resources to be present on a computer. These prerequisites are known as computer system requirements and are often used as a guideline as opposed to an absolute rule. Most software defines two sets of system requirements: Minimum and Recommended. With increasing demand for higher processing power and resources in newer versions of software, system requirements tend to increase

over time. Industry analysts suggest that this trend plays a bigger part in driving upgrades to existing computer systems than technological advancements

Software Requirements Specification (SRS) is the starting point of the software development activity. Little importance was given to this phases in the early days of software development. The emphasis was first on coding and then shifted to design.As
systems grew more complex, it become evident that the goal of the entire system cannot
be easily comprehended. Hence need for the requirements analysis phase arose. Now, for large software systems, requirements analysis is perhaps the most difficult activity and also the most error prone.

Some of the difficulty is due to the scope of this phase. The software project is imitated
by the client needs. In the beginning these needs are in the minds of various people in the client organization. The requirement analyst has to identify the requirements by tacking to these people and understanding their needs. In situations where the software
is to automated a currently manuals process, most of the needs can be understood by observing the current practice.

4.1) Hardware Requirements

- PROCESSOR : PENTIUM III 866 MHz
- RAM : 128 MD SD RAM
- MONITOR : 15” COLOR
- HARD DISK : 20 GB
- FLOPPY DRIVE : 1.44 MB
- CD DRIVE : LG 52X
- KEYBOARD : STANDARD 102 KEYS
- MOUSE : 3 BUTTONS

3.3 4.2)Software Requirements

- OPERATING SYSTEM : Windows XP Professional
- TEXT EDITOR : Visual Studio (or) IDLE (python)
- LANGUAGE : Python 3.0 +

4.3)Performance Requirements

The project must the end user requirements. Accuracy and fast must be imposed in the Project. The project is development as easy as possible for the sake of end user. The project has to be developed with view of satisfying the future requirements and future enhancemen. The tool has been finally implemented satisfying the needs specified by the company. As per the performance is concerned this system said is performing This processing as well as tine taken to generate well reports where also.

4.4) Interface requirements

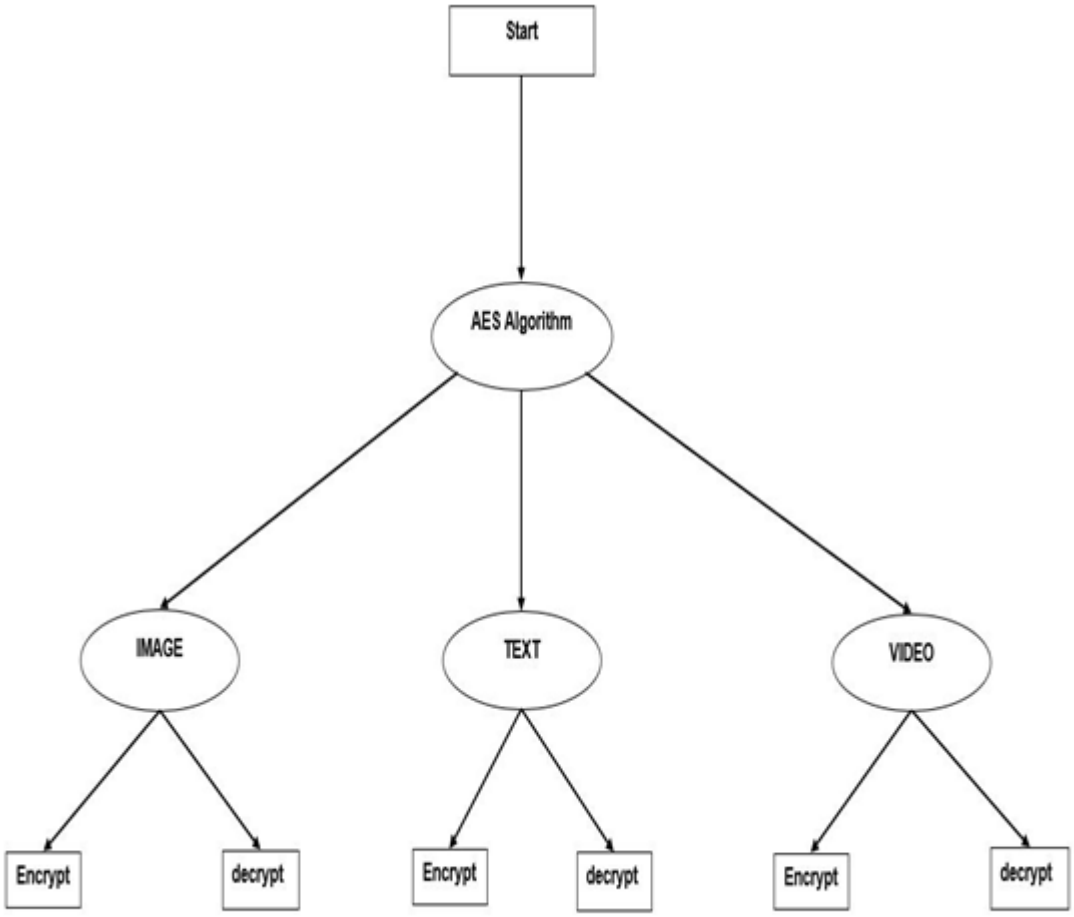
4.4.1 Hardware Interface

The stranded input device like keyboard and mouse are to get input. The output will be generated and display in the monitor. The reports can also be exported to a SQL-server document are text file. The stranded printer in used to take outputs.

4.4.2 Software Interface

In the software requirement we us the python and the pycypto modules. we extract the modules for the pip and we use the modules to execute the program.we use the command line interface for the executing the program. We use the python program to encrypt and decrypt the files

4.5)Functional Requirements



CHAPTER 5

AES ALGORITHM

ALGORITHM:

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

The National Institute of Standards and Technology (NIST) started development of AES in 1997 when it announced the need for a successor algorithm for the Data Encryption Standard (DES) which was starting to become vulnerable to brute-force attacks. His new, advanced encryption algorithm would be unclassified and had to be capable of protecting sensitive government information well into the next century, according to the NIST announcement of the process for development of an advanced encryption standard algorithm. It was intended to be easy to implement in hardware and software, as well as in restricted environments (for example, in a smart card) and offer good defenses against various attack techniques.

5.1 GENERAL STRUCTURE

AES comprises three block ciphers: AES-128, AES-192 and AES-256. Each cipher encrypts and decrypts data in blocks of 128 bits using cryptographic keys of 128-, 192-and 256-bits, respectively. The Rijndael cipher was designed to accept additional block sizes and key lengths, but for AES, those functions were not adopted.

Symmetric (also known as secret-key) ciphers use the same key for encrypting and decrypting, so the sender and the receiver must both know the same secret key. All key lengths are deemed sufficient to protect classified information up to the Secret level with Top Secret information requiring either 192- or 256-bit key lengths. There are 10 rounds for 128-bit keys, 12 rounds for 192-bit keys and 14 rounds for 256-bit key a round consists of several processing steps that include substitution, transposition and mixing of the input plaintext and transform it into the final output of ciphertext.

The AES encryption algorithm defines a number of transformations that are to be performed on data stored in an array. The first step of the cipher is to put the data into an array; after which the cipher transformations are repeated over a number of encryption rounds. The number of rounds is determined by the key length, with 10 rounds for 128-bit keys, 12 rounds for 192-bit keys and 14 rounds for 256-bit keys.

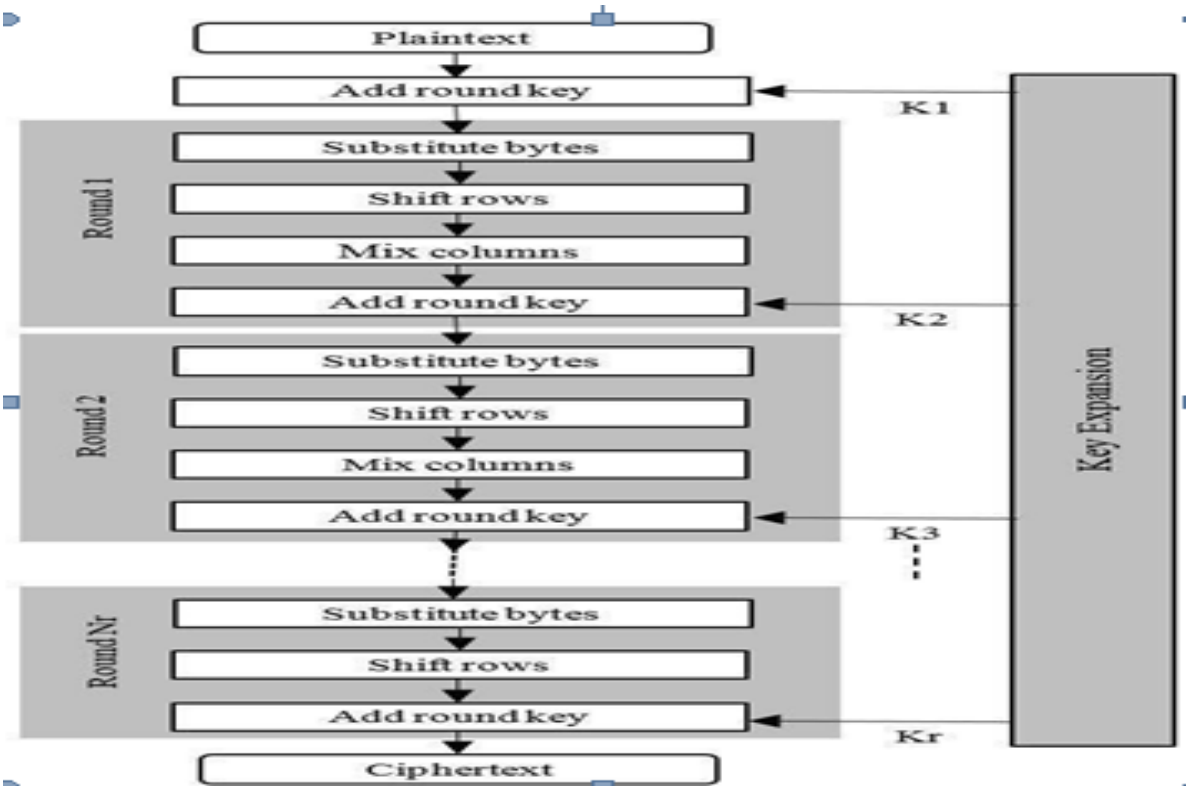
The first transformation in the AES encryption cipher is substitution of data using a substitution table; the second transformation shifts data rows, the third mixes columns. The last transformation is a simple exclusive or (XOR) operation performed on each column using a different part of the encryption key -- longer keys need more rounds to complete

Each word is four bytes, and the total key schedule is 44 words for the 128-bit key. Note that the ordering of bytes within a matrix is by column. So, for example, the first four bytes of a 128-bit plaintext input to the encryption cipher occupy the first column of the in matrix, the second four bytes occupy the second column, and so on.

The cipher consists of N rounds, where the number of rounds depends on the key length: 10 rounds for a 16-byte key, 12 rounds for a 24-byte key, and 14 rounds for a 32-byte key (Table 6.1). The first N - 1 rounds consist of four distinct transformation functions: SubBytes, ShiftRows, MixColumns, and AddRoundKey, which are described subsequently. The final round contains only three transformations, and there is a initial single transformation (AddRoundKey) before the first round, which can be considered Round 0. Each transformation takes one or more

4 * 4 matrices as input and produces a 4 * 4 matrix as output. Figure 6.1 shows that the output of each round is a 4 * 4 matrix, with the output of the final round being the ciphertext. Also, the key expansion function generates N + 1 round keys, each of which is a distinct 4 * 4 matrix. Each round key serves as one of the inputs to the AddRoundKey transformation in each round.

5.2 AES ENCRYPTION



One noteworthy feature of this structure is that it is not a Feistel structure. Recall that, in the classic Feistily structure, half of the data block is used to modify the other half of the data block and then the halves are swapped. AES instead processes the entire data block as a single matrix during each round using substitutions and permutation. The key that is provided as input is expanded into an array of forty-four 32-bit words, w[i]. Four distinct words (128 bits) serve as a round key for each round; these are indicated.

For encryption, each round consists of the following four steps up to n-1 rounder:

- **Substitute bytes:** Uses an S-box to perform a byte-by-byte substitution of the block.
- **Shift rows:** A simple permutation
- **Mix columns:** A substitution that makes use of arithmetic over GF(28).
- **Add round key:** A simple bitwise XOR of the current block with a portion of the expanded key

The last step consists of XOR the output of the previous three steps with four words from the key schedule.

- The last round for encryption does not involve the "Mix columns" step. The last round for decryption does not involve the "Inverse mix columns" step

Only three steps are in the last round

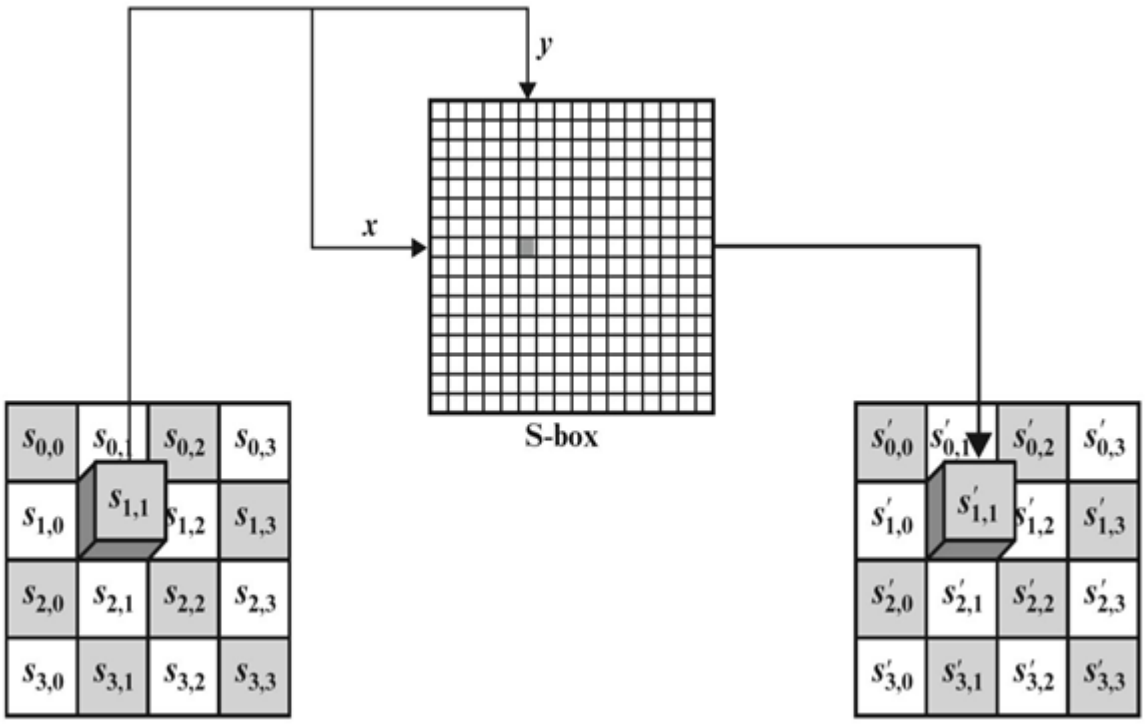
- Substitute bytes
- Shift rows
- **Add round key**

The structure is quite simple. For both encryption and decryption, the cipher begins with an AddRoundKey stage, followed by nine rounds that each includes all four stages, followed by a tenth round of three stages. Figure 6.4 depicts the structure of a full encryption round.Only the AddRoundKey stage makes use of the key. For this reason, the cipher begins and ends with an AddRoundKey stage. Any other stage, applied at the beginning or end, is reversible without knowledge of the key and so would add no security.The AddRoundKey stage is, in effect, a form of Vernam cipher and by itself would not be formidable. The other three stages together provide confusion, diffusion, and nonlinearity, but by themselves would provide no security because they do not use the key. We can view the cipher as alternating

operations of XOR encryption (AddRoundKey) of a block, followed by scrambling of the block (the other three stages), followed by XOR encryption, and so on. This scheme is both efficient and highly secure.

5.2.1 Substitute Bytes

The forward substitute byte t ransformation, called SubBytes, is a simple table lookup . AES defines a 16 * 16 matrix of byte values, called an S-box , that contains a permutation of all possible 256 8-bit values. Each individual byte of State is mapped into a new byte in the following way: The leftmost 4 bits of the byte are used as a row value and the rightmost 4 bits are used as a column value. These row and column values serve as indexes into the S-box to select a unique 8-bit output value. For example, the hexadecimal value {95} references row 9, column 5 of the S-box, which contains the value {2A}. Accordingly, the value {95} is mapped into the value {2A}.



Substitute Bytes

.

The SubBytes phase of AES involves splitting the input into bytes and passing each through a Substitution Box or S-Box. Unlike DES, AES uses the same S-Box for all bytes. The AES S-Box implements inverse multiplication in Galois Field 28. The AES S-Box is shown in the Table below.

		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

(a) S-box

S-Box for the substitute bytes

To read this Table, the byte input is broken into two 4-bit halves. The first half determines the row and the second half determines the column. For example, the S-Box transformation of 35 or 0x23 can be found in the cell at the intersection of the row labeled 20 and the column labeled 03. Therefore decimal 35 becomes 0x26 or decimal 38.

Initialize the S-box with the byte values in ascending sequence row by row. The

first row contains {00}, {01}, {02}, c , {0F}; the second row contains {10}, {11}, etc.; and so on. Thus, the value of the byte at row y, column x is {yx}. 2. Map each byte in the S-box to its multiplicative inverse in the finite field GF(28); the value {00} is mapped to itself. 3. Consider that each byte in the S-box consists of 8 bits labeled (b7, b6, b5, b4, b3, b2, b1, b0). Apply the following transformation to each bit of each byte in the S-box: bi = = bi ⊕ b(i+4) mod 8 ⊕ b(i+5) mod 8 ⊕ b(i+6) mod 8 ⊕ b(i+7) mod 8 ⊕ ci

where ci is the ith bit of byte c with the value {63}; that is, (c7c6c5c4c3c2c1c0) = (01100011). The prime (′) indicates that the variable is to be updated by the value on the right. The AES standard depicts this transformation in matrix form

$$\begin{bmatrix} b'_0 \\ b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \\ b'_5 \\ b'_6 \\ b'_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

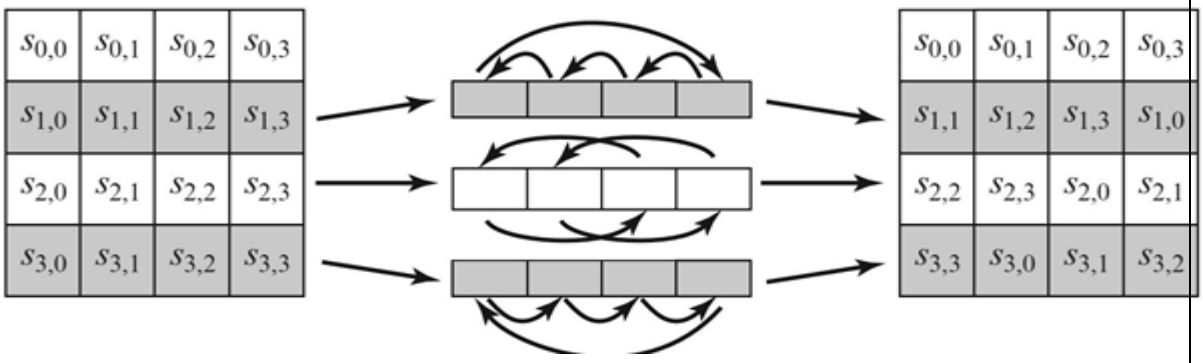
Substitute Bytes

The S-box is designed to be resistant to known cryptanalytic attacks. Specifically, the Rijndael developers sought a design that has a low correlation b etween input bits and output bits and the property that the output is not a linear mathematical function of the input [DAEM01]. The nonlinearity is due to the use of the multiplicative inverse. In addition, the constant in Equation (6.1) was chosen so that the S-box has no fixed points [S@box(a) = a] and no "opposite fixed points" [S@box(a) = a], where a is the bitwise complement of a.

In the SubBytes step, each byte in the matrix is updated using an 8-bit substitution box , the Rijndael S-box. This operation provides the non-linearity in the cipher. The S-box used is derived from the multiplicative inverse over **GF** (2^8), known to have good non-linearity properties.To avoid attacks based on simple algebraic properties, the S-box is constructed by combining the inverse function with an invertible affine

5.2.2 Shift Rows

The forward shift row transformation, called ShiftRows, is depicted. The Rijndael S-box. This operation provides the non-linearity in the cipher. The S-box used is derived from the multiplicative inverse over **GF** (2^8), known to have good non-linearity properties.To avoid attacks based on simple algebraic properties, the S-box is constructed by combining the inverse function with an invertible affine .The first row of State is not altered. For the second row, a 1-byte circular left shift is performed. For the third row, a 2-byte circular left shift is performed. For the fourth row, a 3-byte circular left shift is performed. The following is an example of ShiftRows. The shift row transformation is more substantial than it may first appear. This is because the State, as well as the cipher input and output, is treated as an array of four 4-byte columns. Thus, on encryption, the first 4 bytes of the plaintext are copied to the first column of State, and so on. Furthermore, as will be seen, the round key is applied to State column by column. Thus, a row shift moves an individual byte from one column to another, which is a linear.

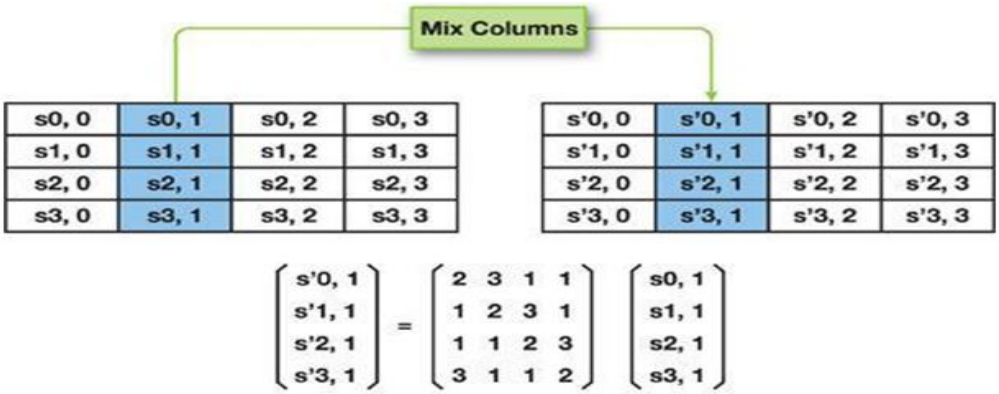


Shift Rows

In the Figure, the first number in each cell refers to the row number and the second refers to the column. The topmost row (row 0) does not shift at all, row 1 shifts left by one, and so on. Distance of a multiple of 4 bytes. Also note that the transformation ensures that the 4 bytes of one column are spread out to four different columns.

5.2.3 Mix Columns

The forward mix column transformation, called MixColumns, operates on each column individually. Each byte of a column is mapped into a new value that is a function of all four bytes in that column. The transformation can be defined by the following matrix multiplication on State. Like the Shift Rows phase of AES, the Mix Columns phase provides diffusion by mixing the input around. Unlike Shift Rows, Mix Columns performs operations splitting the matrix by columns instead of rows. A visual representation of the MixColumns operation is shown above. Unlike standard matrix multiplication, MixColumns performs matrix multiplication as per Galois Field 28. Although we won't describe this step in detail, it is important to note that this multiplication has the property of operating independently over each of the columns of the initial matrix, i.e. the first column when multiplied by the matrix, produces the first column of the resultant matrix.



Mix Columns

Each element in the product matrix is the sum of products of elements of one row and one column. In this case, the individual additions and multiplications are performed in GF(2^8). The MixColumns transformation on a single column of State can be expressed as

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} = \begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix}$$

Mix Columns of martix

Each element in the product matrix is the sum of products of elements of one row and one column. In this case, the individual additions and multiplications are performed in GF(2^8). The MixColumns transformation on a single column of State can be expressed as

1)For the bytes in the first row of the state array, this operation can be stated as

- **$s'_{0,j} = (0x02 \times s0,j) \otimes (0x03 \times s1,j) \otimes s2,j \otimes s3,j$**

2)For the bytes in the second row of the state array, this operation can be stated as

- **$s'_{1,j} = s0,j \otimes (0x02 \times s1,j) \otimes (0x03 \times s2,j) \otimes s3,j$**

3)For the bytes in the third row of the state array, this operation can be stated as

- **$s'_{2,j} = s0,j \otimes s1,j \otimes (0x02 \times s2,j) \otimes (0x03 \times s3,j)$**

4)And, for the bytes in the fourth row of the state array, this operation can be stated as

- **$s'_{3,j} = (0x03 \times s_{0,j}) \otimes s_{1,j} \otimes s_{2,j} \otimes (0x02 \times s_{3,j})$**

5.2.4 Add Round Key

In the forward add round key transformation, called AddRoundKey, the 128 bits of State are bitwise XORed with the 128 bits of the round key. As shown in Figure 6.5b, the operation is viewed as a columnwise operation between the 4 bytes of a State column and one word of the round key; it can also be viewed as a byte-level operation.

The add round key transformation is as simple as possible and affects every bit of State. The complexity of the round key expansion, plus the complexity of the other stages of AES, ensure security .The Add Round Key operation is the only phase of AES encryption that directly operates on the AES round key. In this operation, the input to the round is exclusive ored with the round key.

47	40	A3	4C
37	D4	70	9F
94	E4	3A	42
ED	A5	A6	BC

⊕

AC	19	28	57
77	FA	D1	5C
66	DC	29	00
F3	21	41	6A

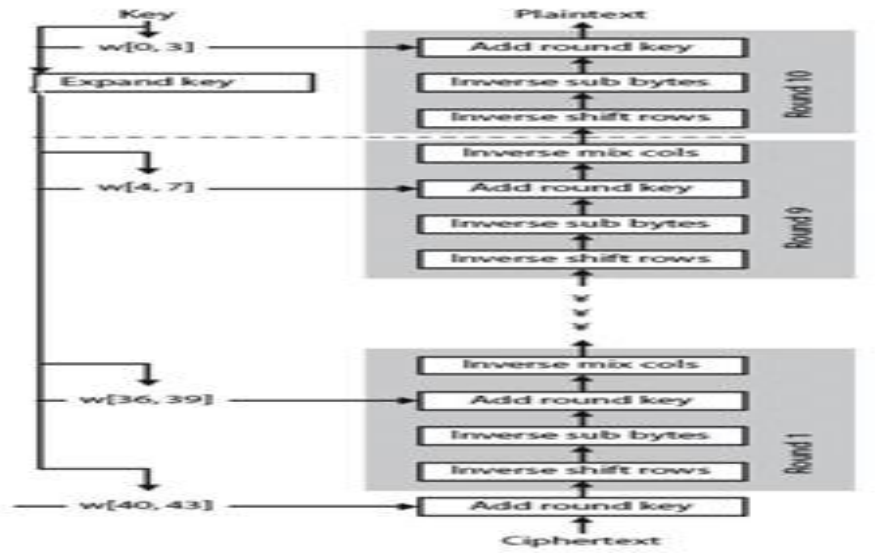
=

EB	59	8B	1B
40	2E	A1	C3
F2	38	13	42
1E	84	E7	D6

Add Round Key

CHAPTER6

6.1)AES DECRYPTION



AES Decryption

Once it is established that all four stages are reversible, it is easy to verify that decryption does recover the plaintext.Lays out encryption and decryption going in opposite vertical directions. At each horizontal point State is the same for both encryption and decryption. The final round of both encryption and decryption consists of only three stages.

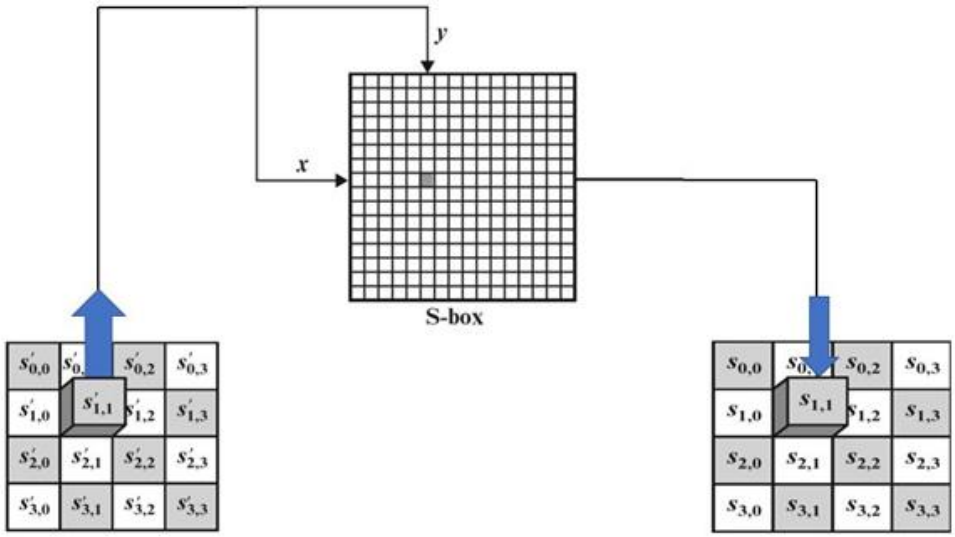
6.1.1 Inverse Sub bytes Rows

The inverse substitute byte transformation, called InvSubBytes, makes use of the inverse S-box shown in Table 6.2b. Note, for example, that the input {2A} produces the output {95}, and the input {95} to the S-box produces {2A}. The inverse S-box is constructed (Figure 6.6b) by applying the inverse of the transformation in Equation (6.1) followed by taking the multiplicative inverse in GF(28). The inverse transformation is

$$bi = b(i+2) \bmod 8 \oplus b(i+5) \bmod 8 \oplus b(i+7) \bmod 8 \oplus di$$

To see that InvSubBytes is the inverse of SubBytes, label the matrices in SubBytes and InvSubBytes as X and Y, respectively, and the vector versions of constants c and d as C and D, respectively. For some 8-bit vector B, Equation (6.2) becomes B= = XB
C. We need to show that Y(XB ⊕ C) ⊕ D = B. To multiply out, we must show YXB

YC ⊕ D = B. We have demonstrated that YX equals the identity matrix, and the YC = D, so that YC ⊕ D equals the null vector. the S-box must be invertible, that is, IS-box[S-box(a)] = a. However, the S-box does not self-inverse in the sense that it is not true that S-box(a) = IS-box(a). For example, S-box({95}) = {2A}, but IS-box({95}) = {AD}.

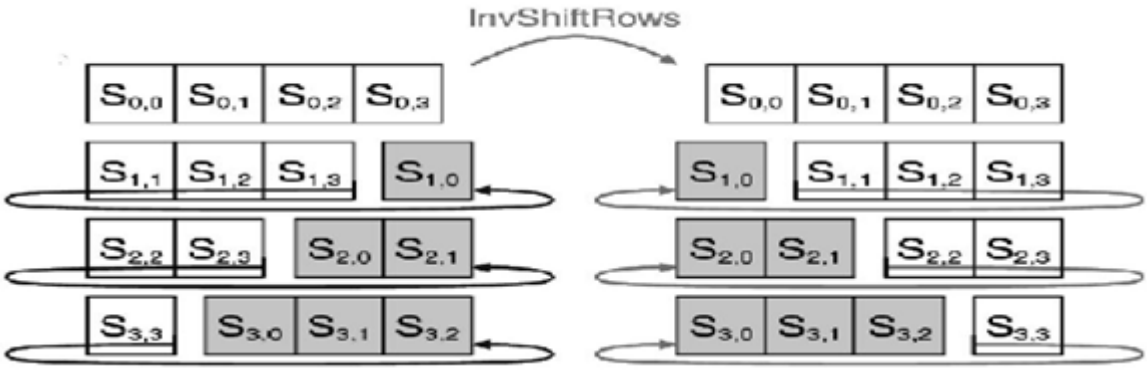


Inverse Sub bytes Rows

		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

S-Box for Inverse Sub bytes Rows

The inverse shift row transformation, called InvShiftRows, performs the circular shifts in the opposite direction for each of the last three rows, with a 1-byte circular right shift



Inverse Shift Row

6.2) Inverse Mix Column

The inverse mix column transformation, called InvMixColumns, is defined by the following matrix multiplication

$$\begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} = \begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix}$$

Inverse Mix Column

Each element in the product matrix is the sum of products of elements of one row and one column. In this case, the individual additions and multiplications are performed in GF(2^8)

6.2.1) Add Round Key

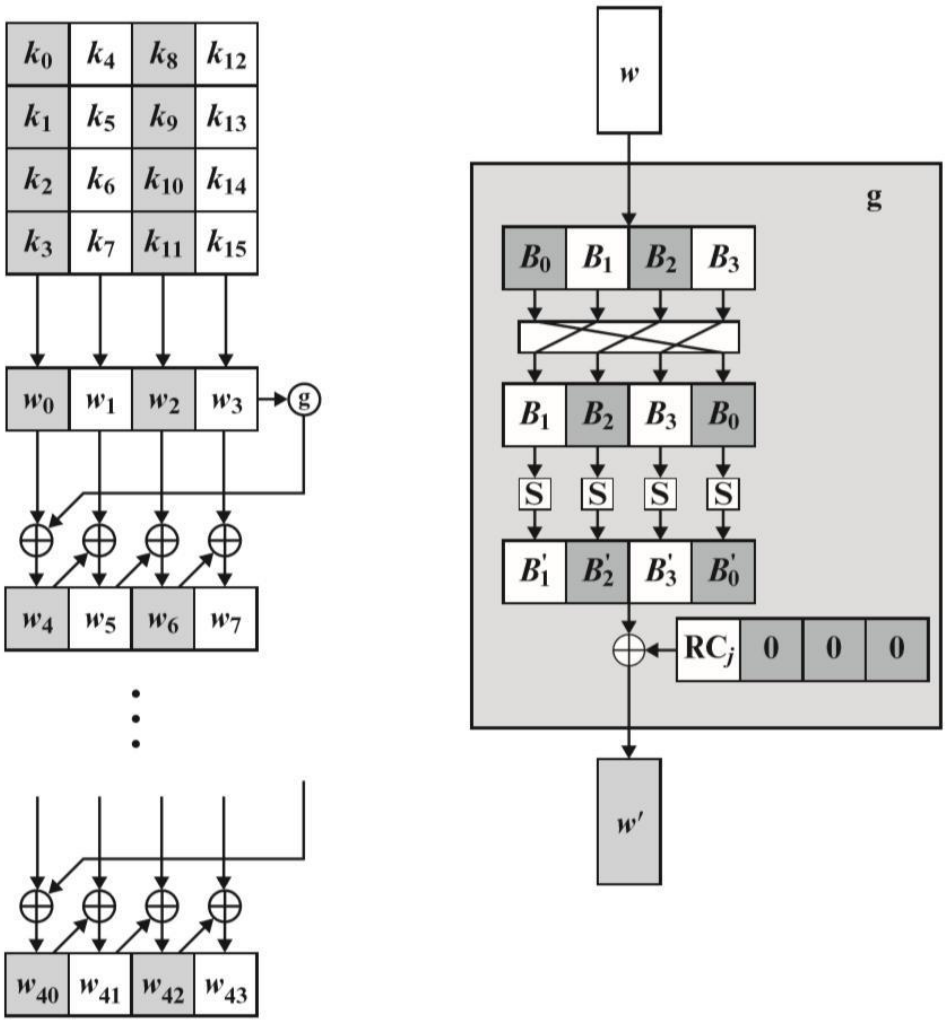
The inverse add round key transformation is identical to the forward add round key transformation, because the XOR operation is its own inverse. The add round key transformation is as simple as possible and affects every bit of State. The complexity of the round key expansion, plus the complexity of the other stages of AES, ensure security.

6.3)THE KEY EXPANSION ALGORITHM

Each round has its own round key that is derived from the original 128-bit encryption key in the manner described in this section. One of the four steps of each round, for both encryption and decryption, involves XORing of the round key with the state array. The AES Key Expansion algorithm is used to derive the 128bit round key for each round from the original 128-bit encryption key.

As you'll see, the logic of the key expansion algorithm is designed to ensure that if you change one bit of the encryption key, it should affect the round keys for several rounds.In the same manner as the 128-bit input block is arranged in the form of a state array, the algorithm first arranges the 16 bytes of the encryption key in the form of a 4×4 array of bytes.

The AES key expansion algorithm takes as input a four-word (16-byte) key and produces a linear array of 44 words (176 bytes). This is sufficient to provide a fourword round key for the initial AddRoundKey stage and each of the 10 rounds of the cipher. The pseudocode on the next page describes the expansion. The key is copied into the first four words of the expanded key. The remainder of the expanded key is filled in four words at a time. Each added word w[i] depends on the immediately preceding word, w[i - 1], and the word four positions back, w[i - 4]. In three out of four cases, a simple XOR is used. For a word whose position in the w array is a multiple of 4, a more complex function is used. Figure 6.9 illustrates the generation of the expanded key, using the symbol g to represent that complex function. The function g consists of the following subfunctions.



The Key Expansion Algorithm

Root Word performs a one-byte circular left shift on a word. This means that an input word [B0, B1, B2, B3] is transformed into [B1, B2, B3, B0]. 2. SubWord performs a byte substitution on each byte of its input word, using the S-box (Table 6.2a). The result of steps 1 and 2 is XORed with a round constant, Rcon[j].

The round constant is a word in which the three rightmost bytes are always 0. Thus, the effect of an XOR of a word with Rcon is to only perform an XOR on the leftmost byte of the word. The round constant is different for each round and is defined as Rcon[j] = (RC[j], 0, 0, 0), with RC[1] = 1, RC[j] = 2#RC[j - 1] and with multiplication defined over the field GF(28). The values of RC[j] in hexadecimal are

J	1	2	3	4	5	6	7	8	9	10
Rc[j]	01	02	04	08	10	20	40	80	1B	36

The Rijndael developers designed the expansion key algorithm to be resistant to known cryptanalytic attacks. The inclusion of a round-dependent round constant eliminates the symmetry, or similarity, between the ways in which round keys are generated in different rounds. The specific criteria that were used are [DAEM99]

- Knowledge of a part of the cipher key or round key does not enable calculation of many other round-key bits.
- An invertible transformation [i.e., knowledge of any Nk consecutive words of the expanded key enables regeneration of the entire expanded key Speed on a wide range of processors.
- Usage of round constants to eliminate symmetries.
- Diffusion of cipher key differences into the round keys; that is, each key bit affects many round key bits.
- Enough nonlinearity to prohibit the full determination of round key differences from cipher key differences only.
- Simplicity of description.

CHAPTER 7

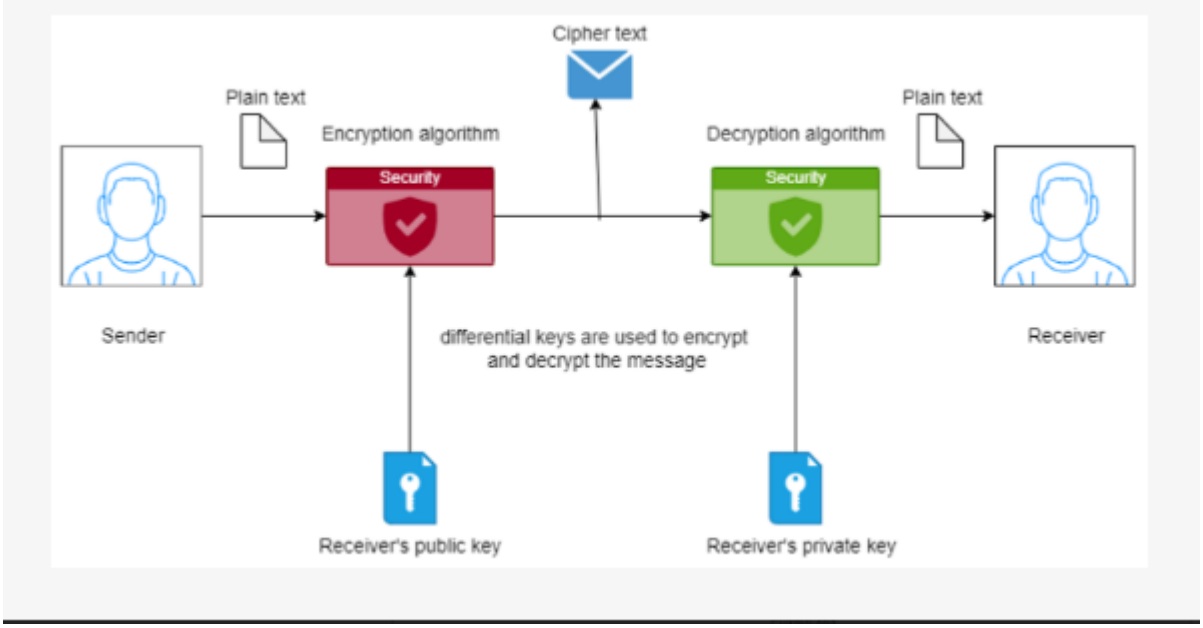
7)DESIGN

Project design is a major step towards a successful project. A project design is a strategic organization of ideas, materials and processes for the purpose of achieving a goal. Project managers rely on a good design to avoid pitfalls and provide parameters to maintain crucial aspects of the project. Project design is an early phase of the project where a project's key features, structure, criteria for success, and major deliverables are all planned out. The point is to develop one or more designs which can be used to achieve the desired project goals. Stakeholders can then choose the best design to use for the actual execution of the project. The project design phase might generate a variety of different outputs, including sketches, flowcharts, and more. So, the design can be implemented using Unified Modeling Language. diagrams such as class diagram, use case diagram, sequence diagram, activity diagrams. UML offers a way to visualize a system’s architectural blueprints in a diagram, including elements such as :

- Any activites
- Individual components of the system
- How the system will run
- How entities interact with others
- External user interface

UML is a common language for business analysts, software architects and developers used to describe, specify, design, and document existing or new business processes, structure and behaviour of artifacts of software systems. The key to making a UML diagram is connecting shapes that represent an object or class with other shapes to illustrate relationships and the flow of information and data.

7.1)System architecture:



7.2)UML DIAGRAMS:

A modelling language with many uses is called Unified Modeling Language (UML). The primary goal of UML is to establish a uniformmethod for visualising a system's design process. It resembles blueprints used in other engineering disciplines quite a bit.

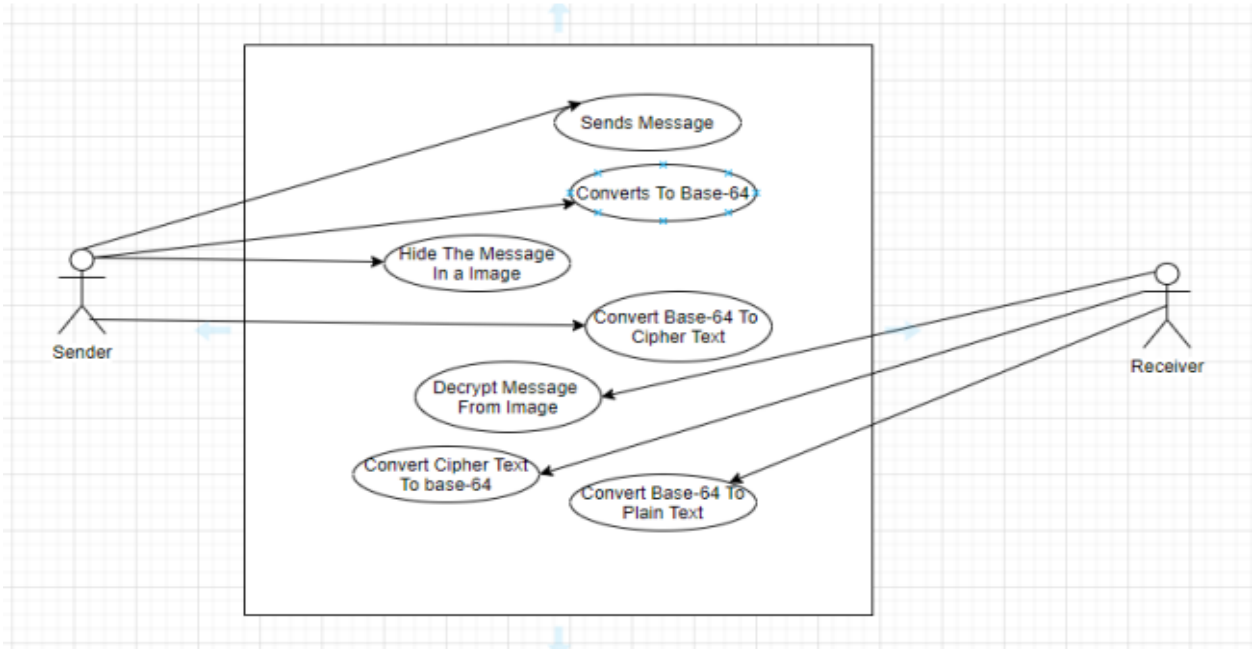
7.3)CLASS DIAGRAMS:

A class diagram in the Unified Modelling Language is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.

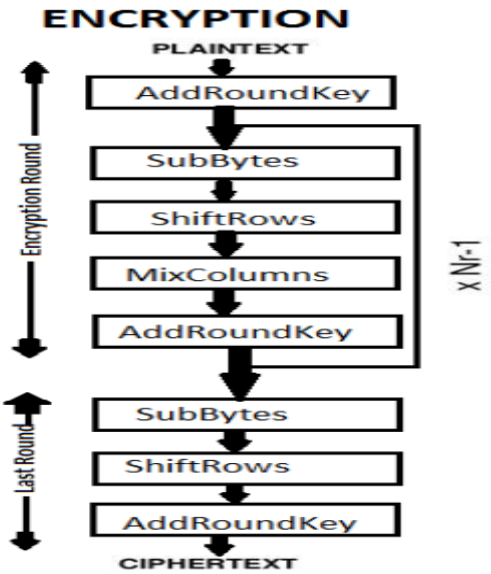
Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application. Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modelling of object-oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.

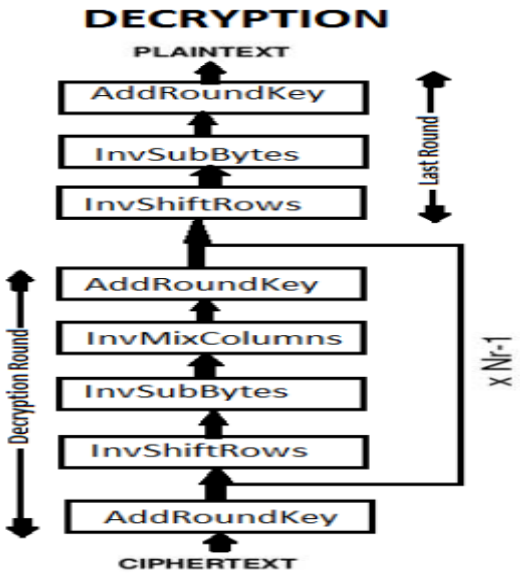
7.4)USE CASE DIAGRAMS:

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well. The use cases are represented by either circles or ellipses.



HOW ENCRYPTION AND DECRYPTION WORKS?

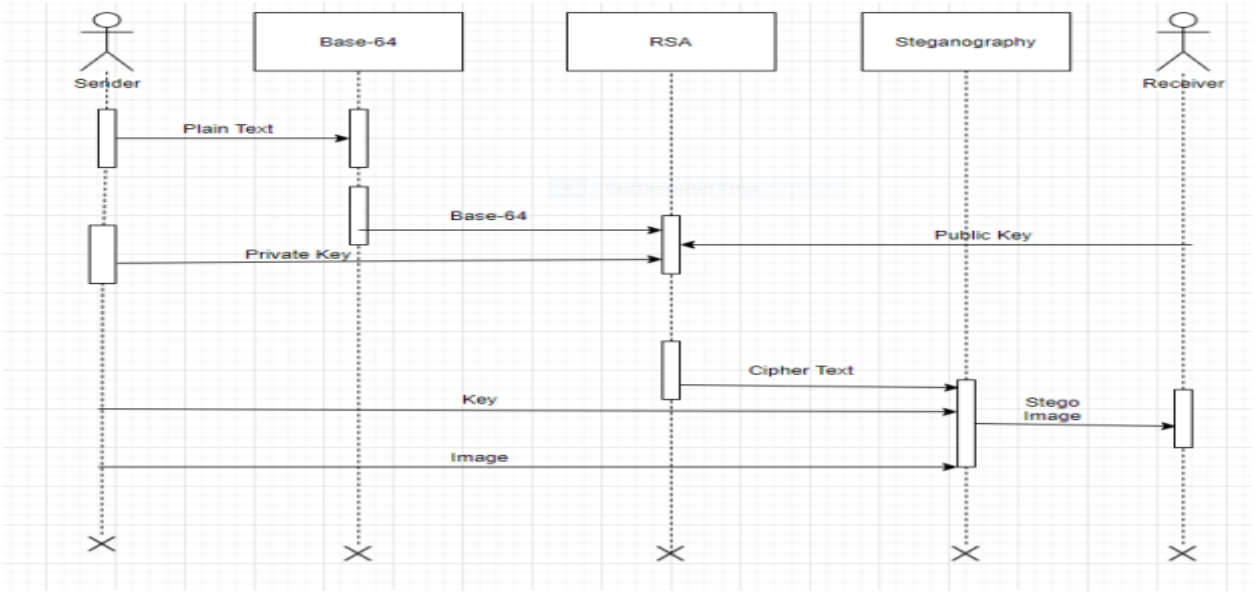




7.5)SEQUENCE DIAGRAM:

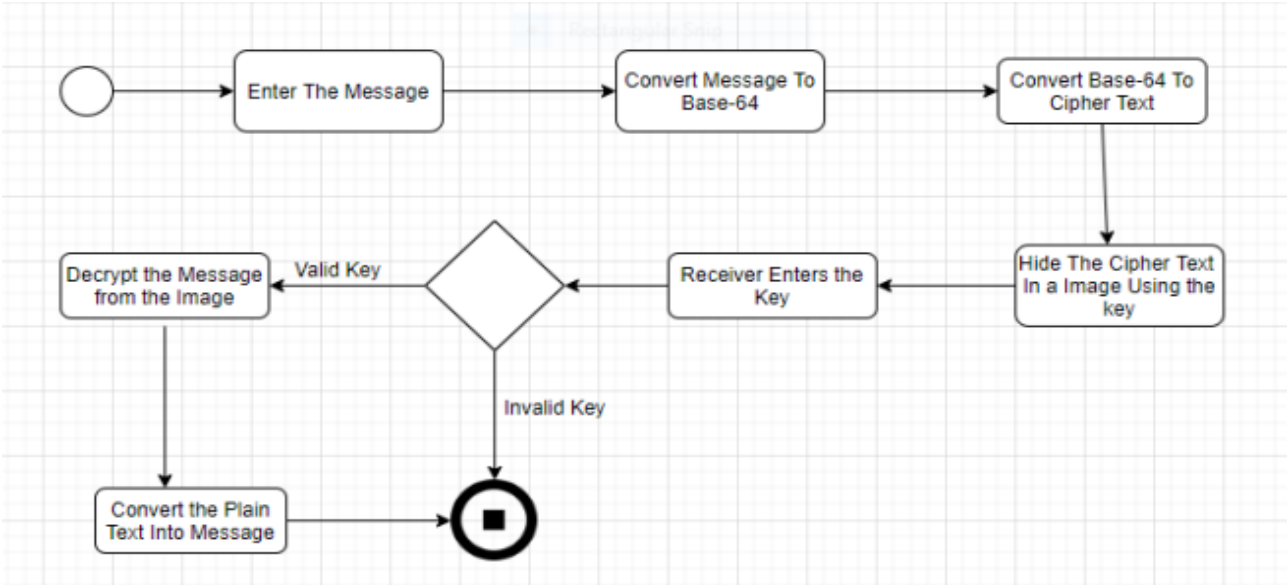
A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.

A sequence diagram shows, as parallel vertical lines, different processes or objects that live simultaneously and as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.



7.6) Activity Diagram

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams are intended to model both computational and organizational processes (i.e., workflows) as well as the data flows intersecting with the related activities. Although activity diagrams primarily show the overall flow of control, they can also include elements showing the flow of data between activities through one or more data stores.



CHAPTER 8

SOURCE CODE

Program:

```
import os
from Crypto.Cipher import AES
from Crypto.Hash import SHA256
from Crypto import Random

def encrypt(key, filename):
chunksize = 64*1024
outputFile = "encrpt_"+filename
filesize = str(os.path.getsize(filename)).zfill(16)
IV = Random.new().read(16)

encryptor = AES.new(key, AES.MODE_CBC, IV)

with open(filename, 'rb') as infile:
with open(outputFile, 'wb') as outfile:
outfile.write(filesize.encode('utf-8'))
outfile.write(IV)
```

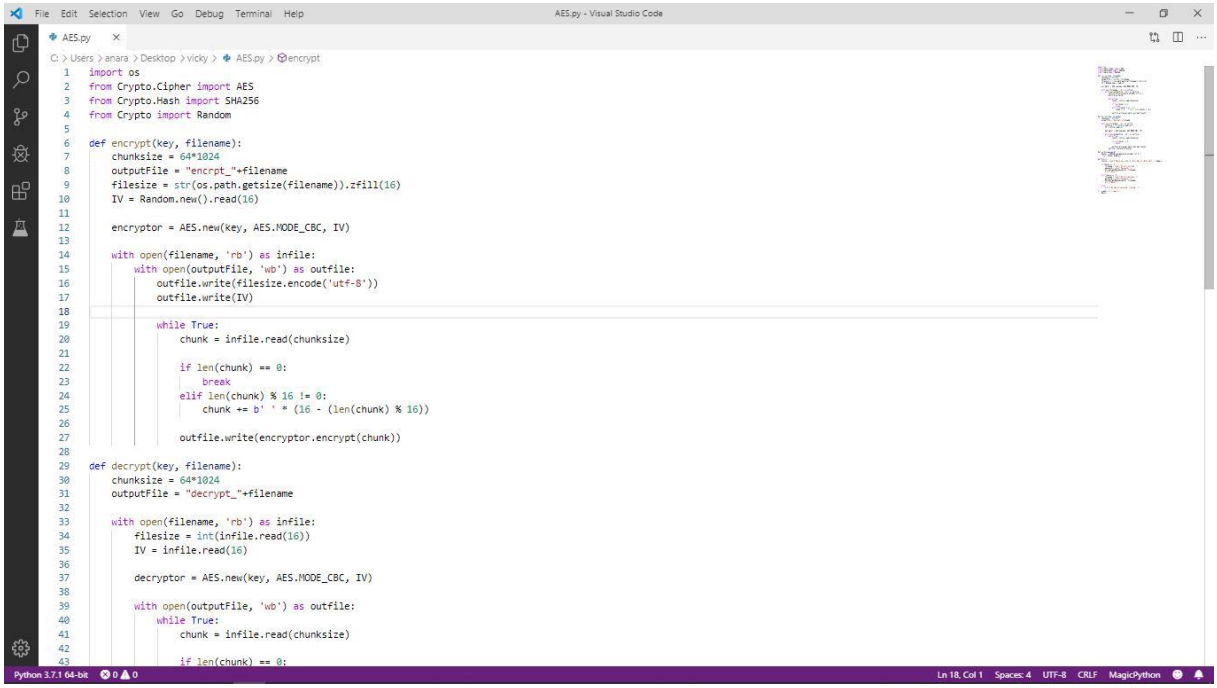
```
while True:
chunk = infile.read(chunksize)

if len(chunk) == 0:
break
elif len(chunk) % 16 != 0:
chunk += b' ' * (16 - (len(chunk) % 16))

outfile.write(encryptor.encrypt(chunk))



def decrypt(key, filename):
chunksize = 64*1024
outputFile = "decrypt_"+filename

with open(filename, 'rb') as infile:
filesize = int(infile.read(16))
IV = infile.read(16)

decryptor = AES.new(key, AES.MODE_CBC, IV)

with open(outputFile, 'wb') as outfile:
while True:
chunk = infile.read(chunksize)

if len(chunk) == 0:
break

outfile.write(decryptor.decrypt(chunk))
outfile.truncate(filesize)
```

```
def getKey(password):
hasher = SHA256.new(password.encode('utf-8'))
return hasher.digest()

def Main():
choice = input("Would you like to (E)ncrypt of (D)ecrypt?: ").upper()

if choice == 'E':
filename = input("File to encrypt: ")
password = input("Password: ")
encrypt(getKey(password), filename)
print("Done.")
elif choice == 'D':
filename = input("File to decrypt: ")
password = input("Password: ")
decrypt(getKey(password), filename)
print("Done.")

else:
print("No option selected, closing...")

if __name__ == "__main__":
Main()
```

CHAPTER 9

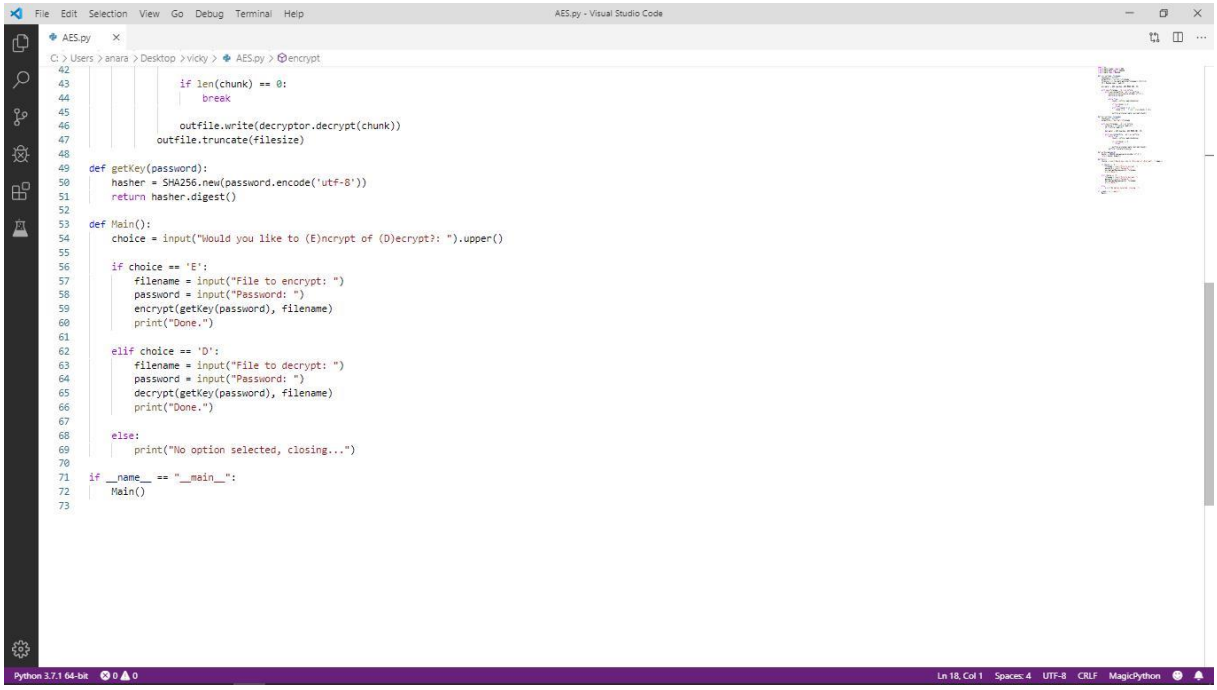
OUTPUT

9.1 SOURCE CODE



```
import os
from Crypto.Cipher import AES
from Crypto.Hash import SHA256
from Crypto import Random

def encrypt(key, filename):
    chunksize = 64*1024
    outputFile = "encrpt_"+filename
    filesize = str(os.path.getsize(filename)).zfill(16)
    IV = Random.new().read(16)

    encryptor = AES.new(key, AES.MODE_CBC, IV)

    with open(filename, 'rb') as infile:
        with open(outputFile, 'wb') as outfile:
            outfile.write(filesize.encode('utf-8'))
            outfile.write(IV)

            while True:
                chunk = infile.read(chunksize)

                if len(chunk) == 0:
                    break
                elif len(chunk) % 16 != 0:
                    chunk += b' ' * (16 - (len(chunk) % 16))

                outfile.write(encryptor.encrypt(chunk))

def decrypt(key, filename):
    chunksize = 64*1024
    outputFile = "decrypt_"+filename

    with open(filename, 'rb') as infile:
        filesize = int(infile.read(16))
        IV = infile.read(16)

        decryptor = AES.new(key, AES.MODE_CBC, IV)

        with open(outputFile, 'wb') as outfile:
            while True:
                chunk = infile.read(chunksize)

                if len(chunk) == 0:
```

Fig 9.1 source code page1



```
                if len(chunk) == 0:
                    break

                outfile.write(decryptor.decrypt(chunk))
            outfile.truncate(filesize)

def getKey(password):
    hasher = SHA256.new(password.encode('utf-8'))
    return hasher.digest()

def Main():
    choice = input("Would you like to (E)ncrypt of (D)ecrypt?: ").upper()

    if choice == 'E':
        filename = input("File to encrypt: ")
        password = input("Password: ")
        encrypt(getKey(password), filename)
        print("Done.")

    elif choice == 'D':
        filename = input("File to decrypt: ")
        password = input("Password: ")
        decrypt(getKey(password), filename)
        print("Done.")

    else:
        print("No option selected, closing...")

if __name__ == "__main__":
    Main()
```

Fig 9.2 source code page2

9.2 SCREENSHOTS OF TEXT FILE ENCRYPTION AND DECRYPTION

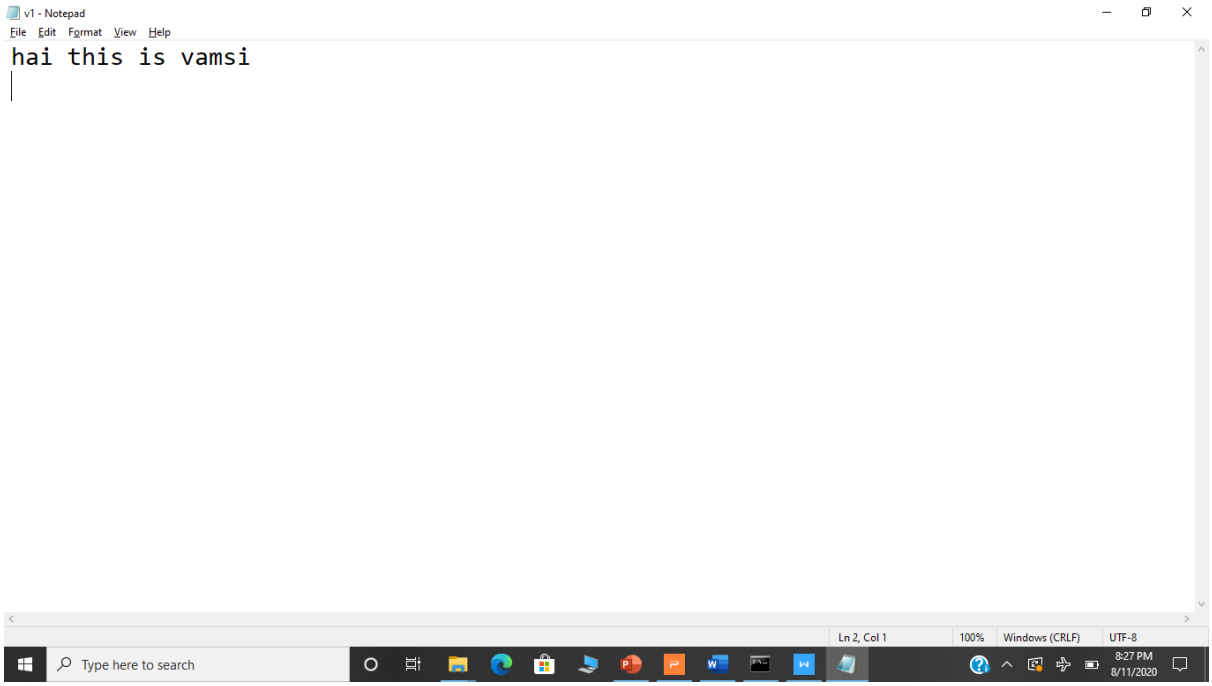


Fig 9.2.1 Before encryption of text file

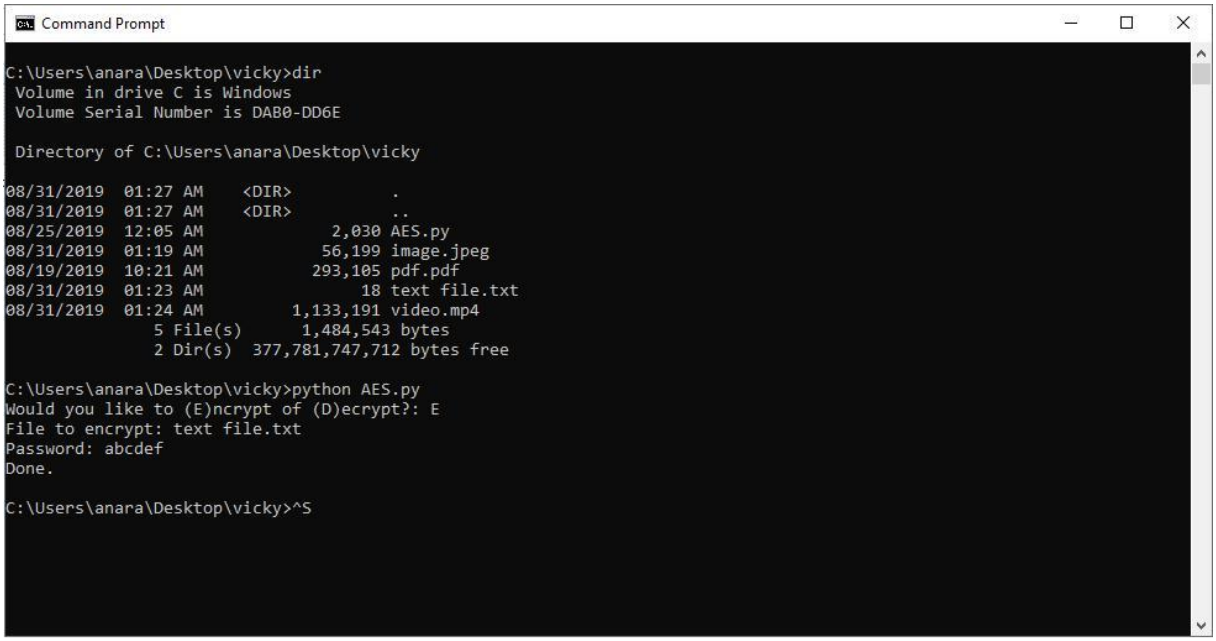


fig 9.2.2 successfully encrypted of text file

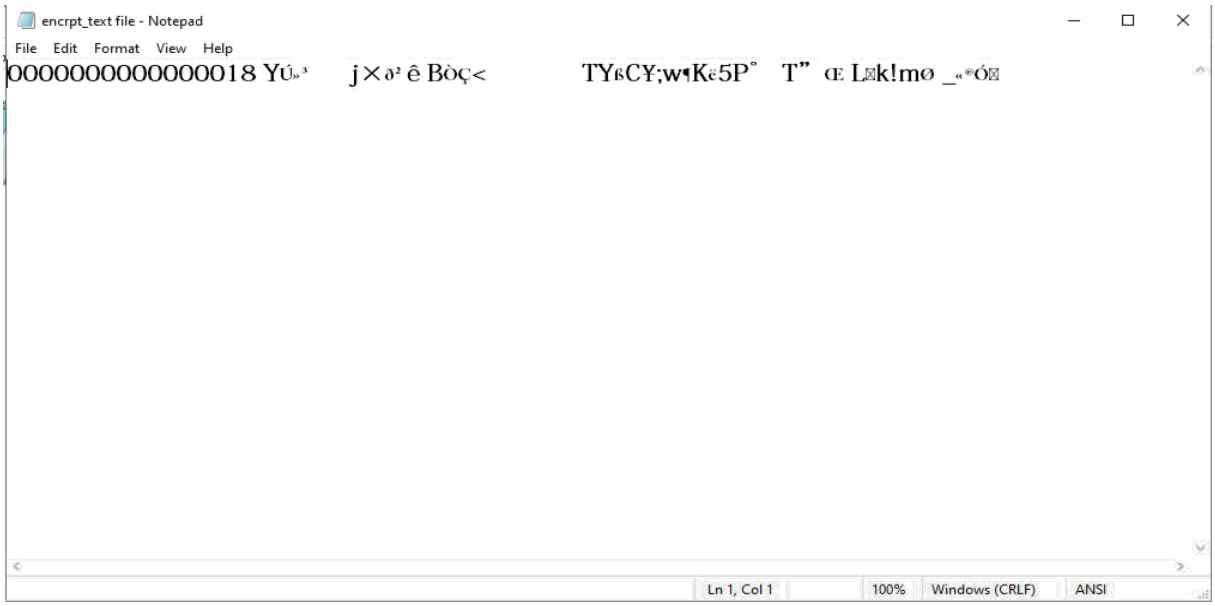


Fig 9.2.3 After encryption of text file

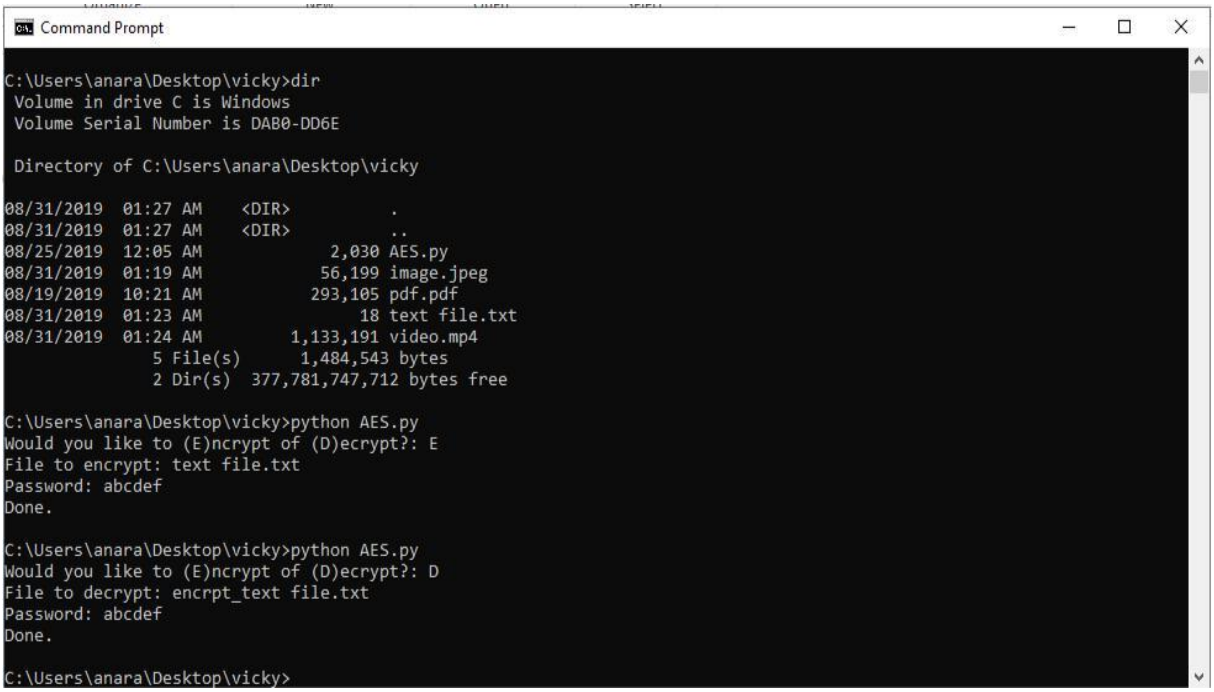


Fig 9.2.4 successfully decrypted of text file

Fig 9.2.5After decryption of text file

9.3 SCREENSHOTS OF IMAGE ENCRYPTION AND DECRYPTION

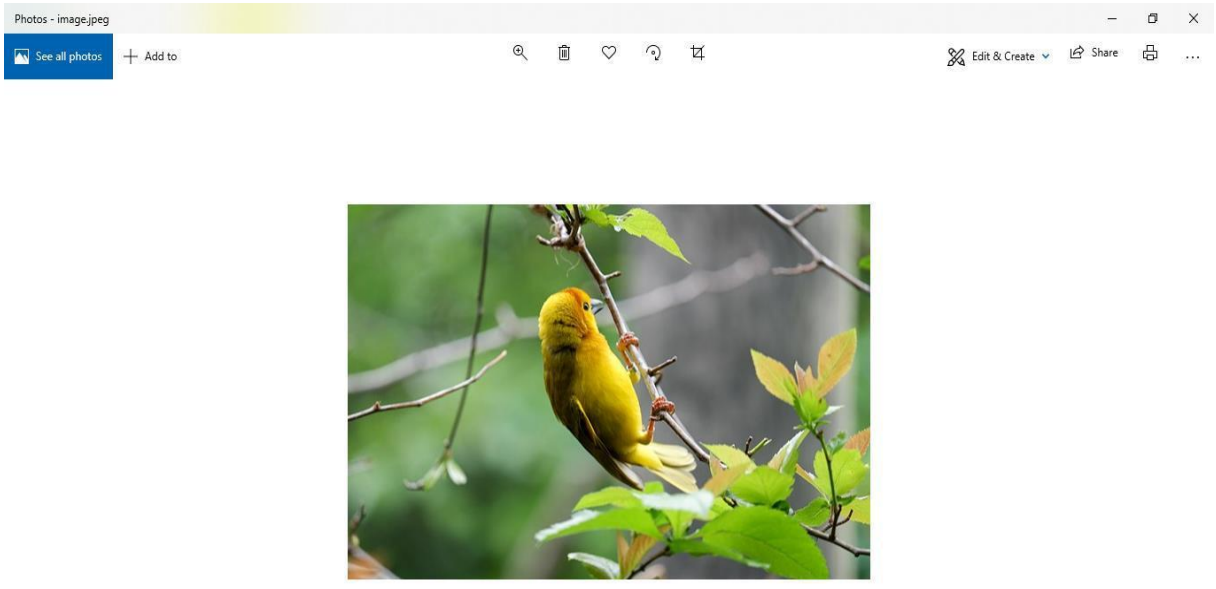


Fig 9.3.1 Before encryption of image

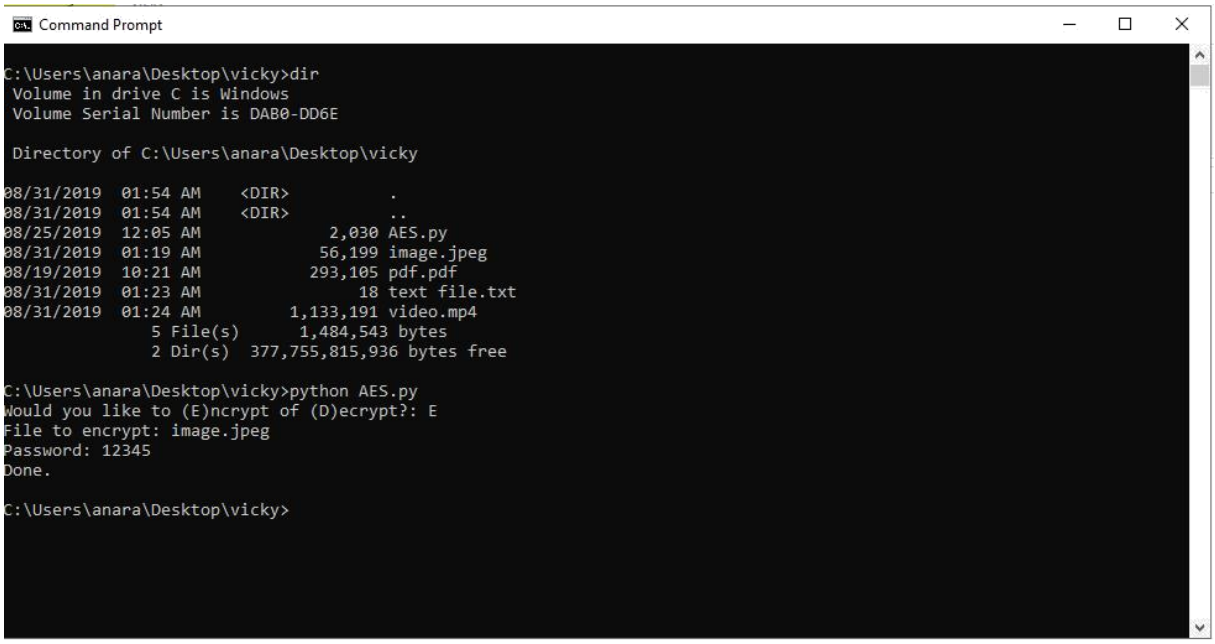


Fig 9.3.2 successfully encrypted of image

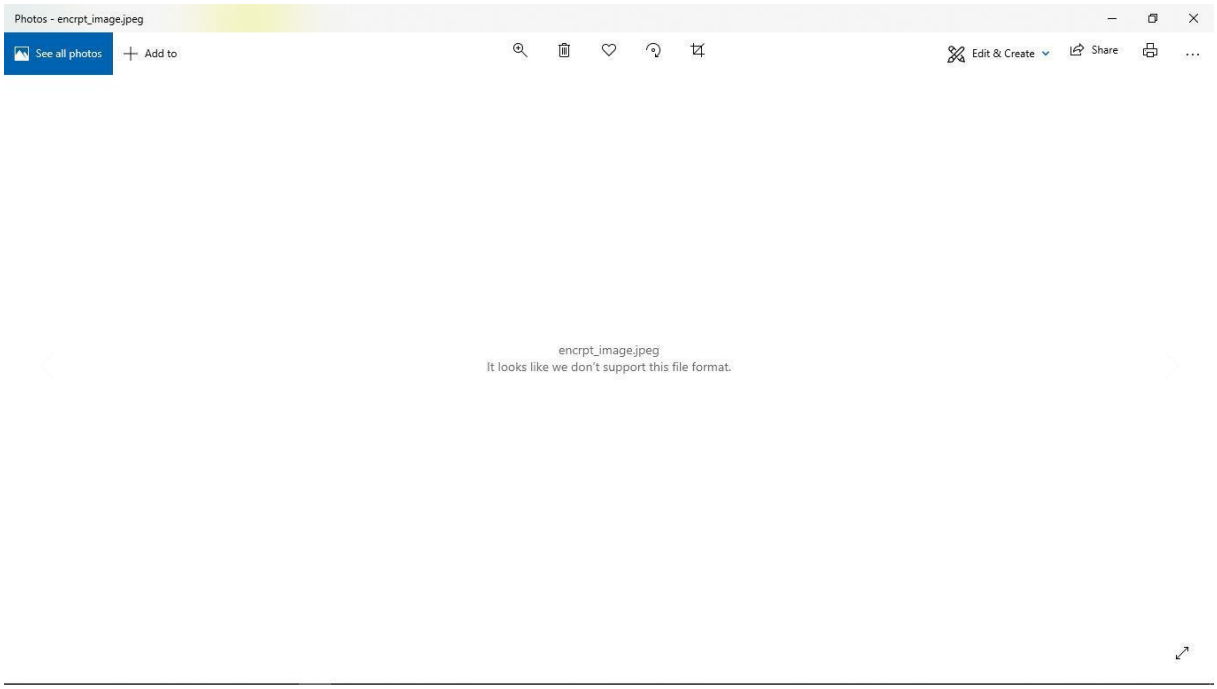


Fig 9.3.3 After encryption of image

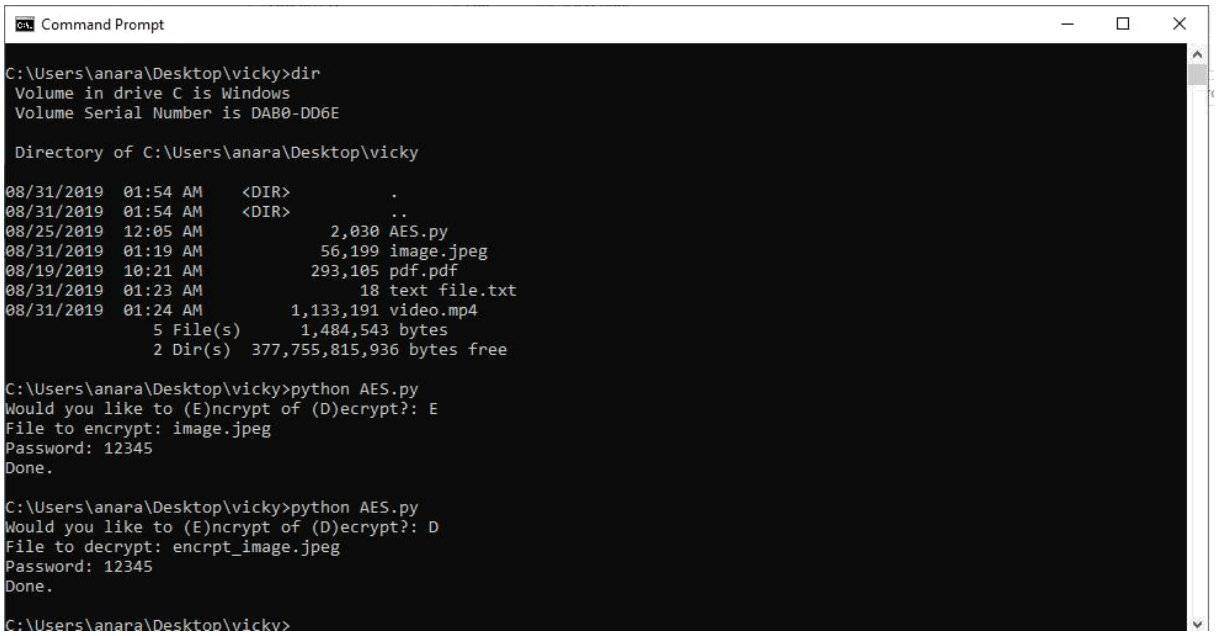


Fig 9.3.4 successfully decrypted of image

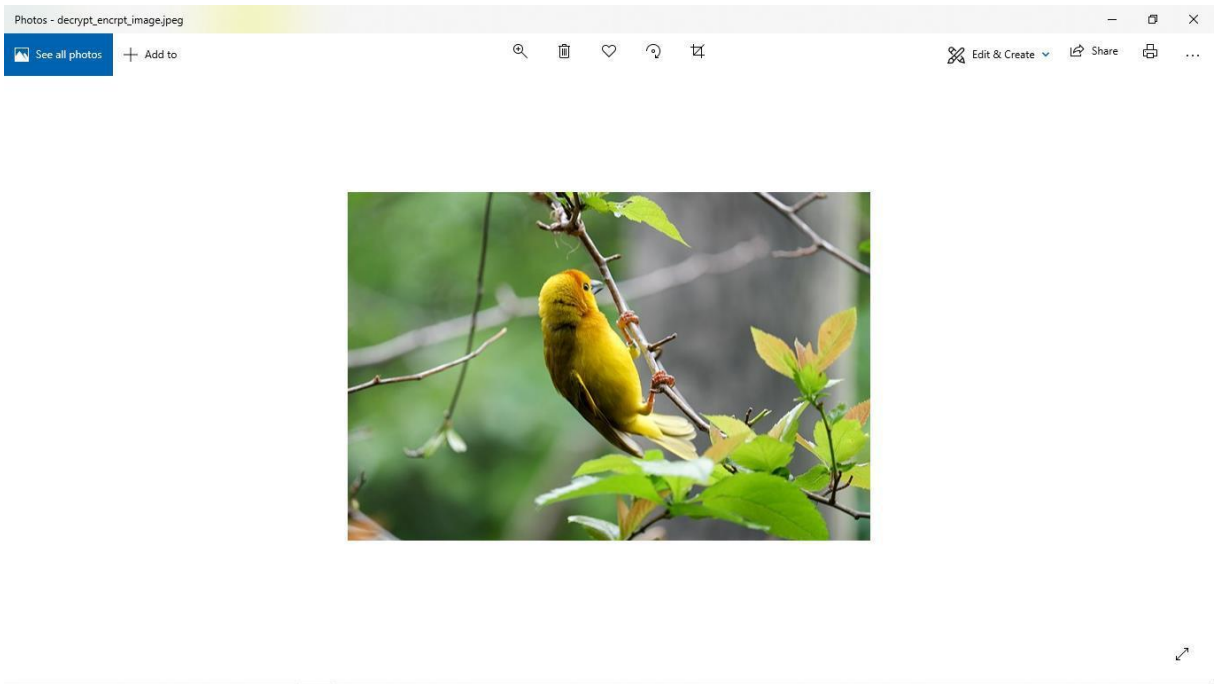


Fig 9.3.5 After decryption of image

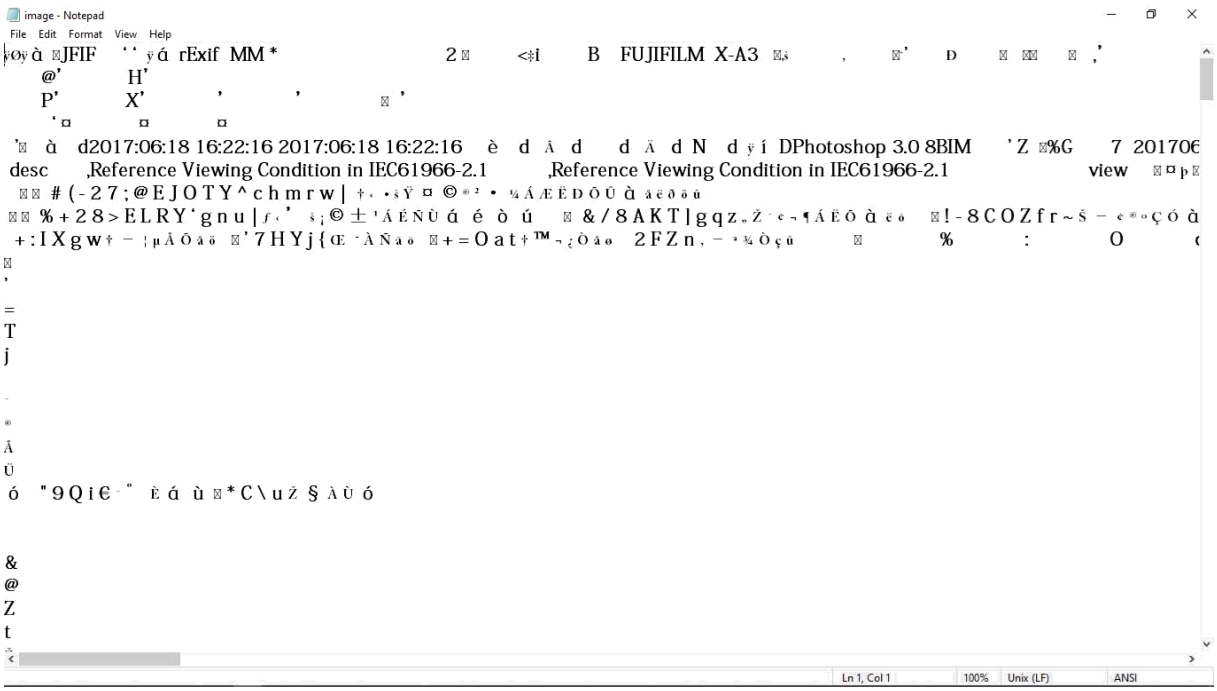


Fig 9.3.6 Before encryption open with notepad of image

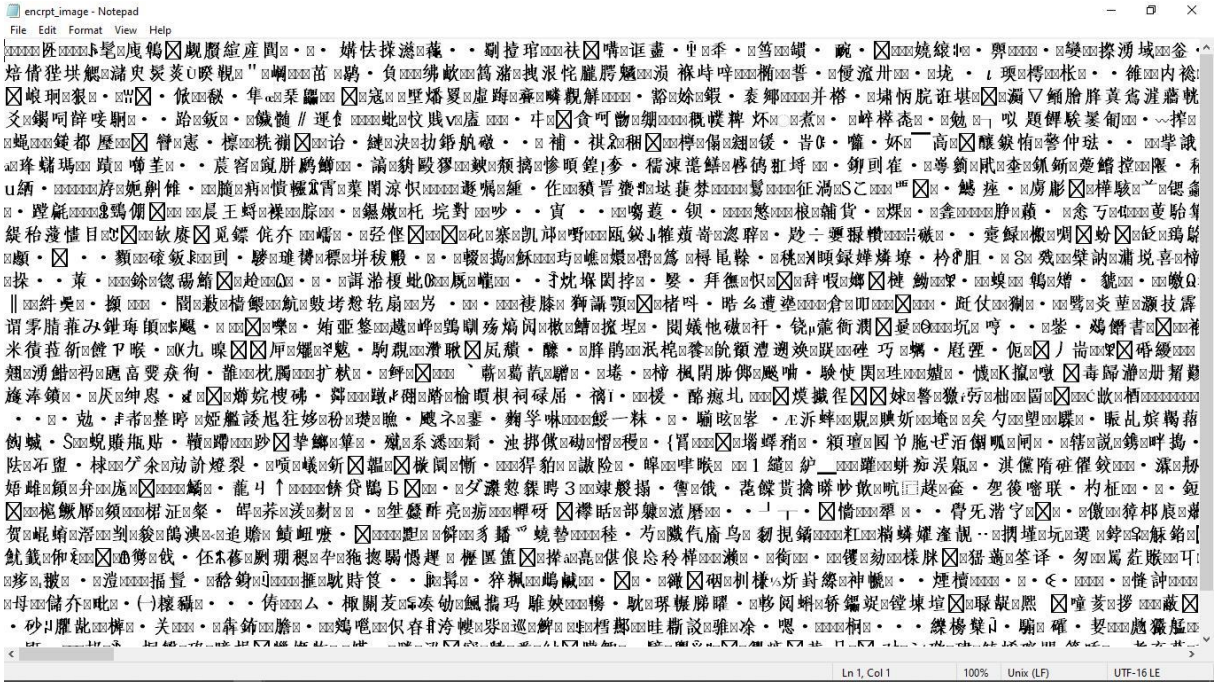


Fig 9.3.7 After encryption open with notepad of image

9.4 SCREENSHOTS OF VIDEO ENCRYPTION AND DECRYPTION

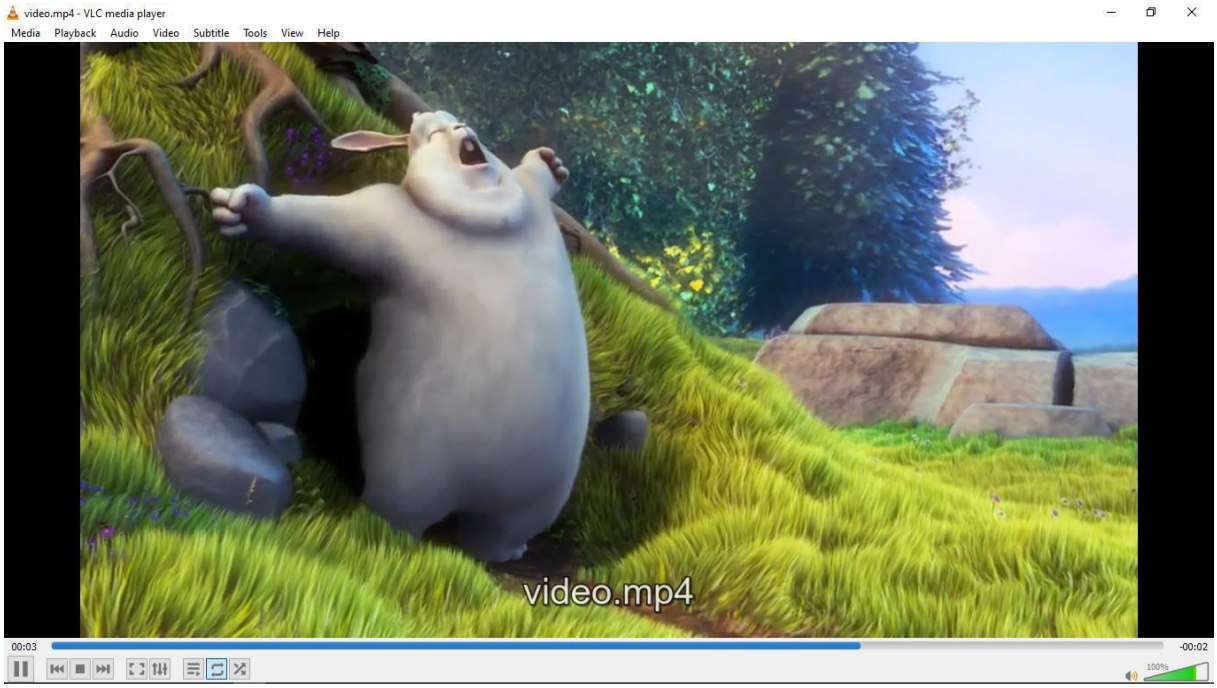


Fig 9.4.1 Before encryption of video

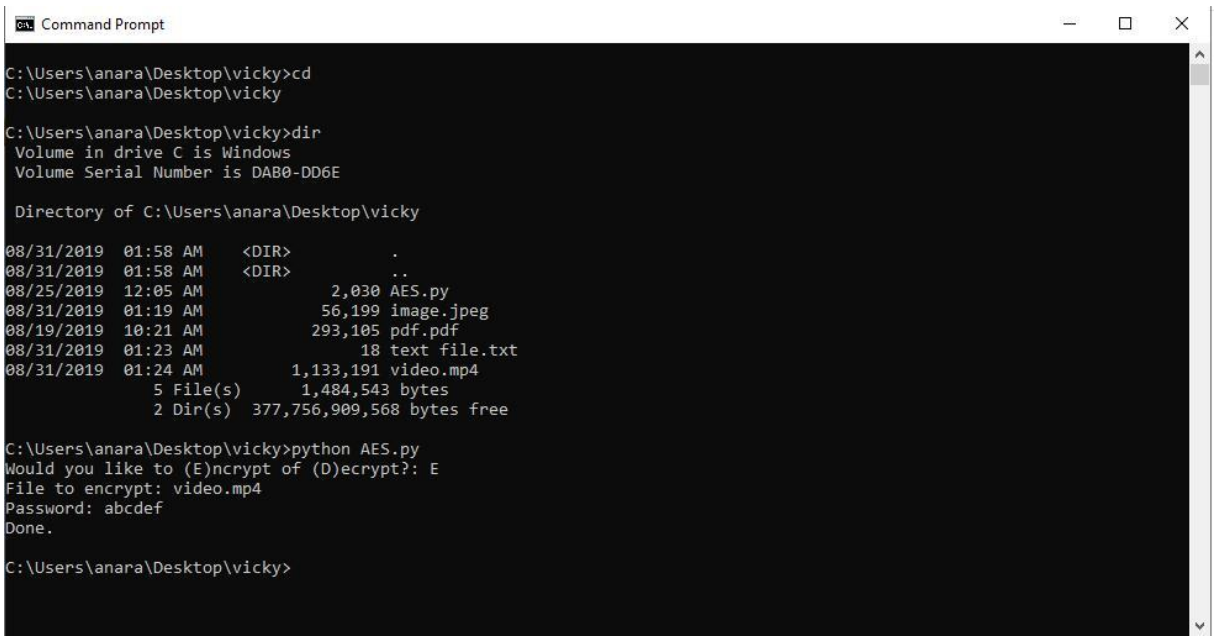


Fig 9.4.2 successfully encrypted of video

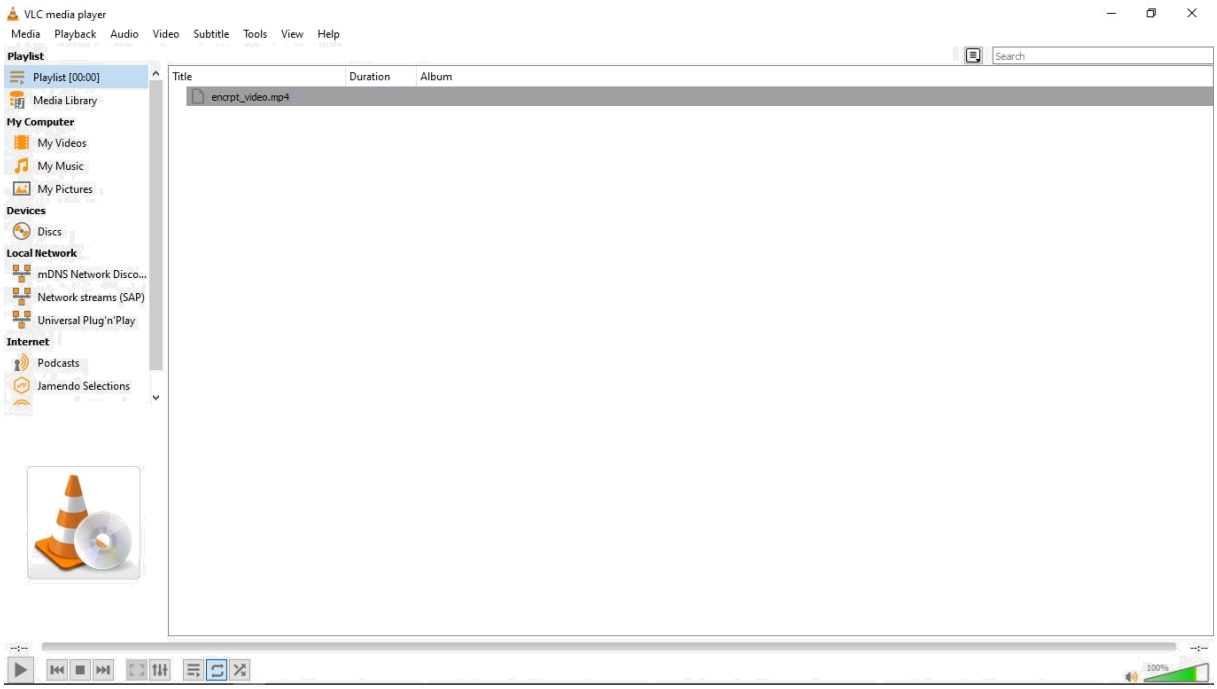


Fig 9.4.3 After encryption of video

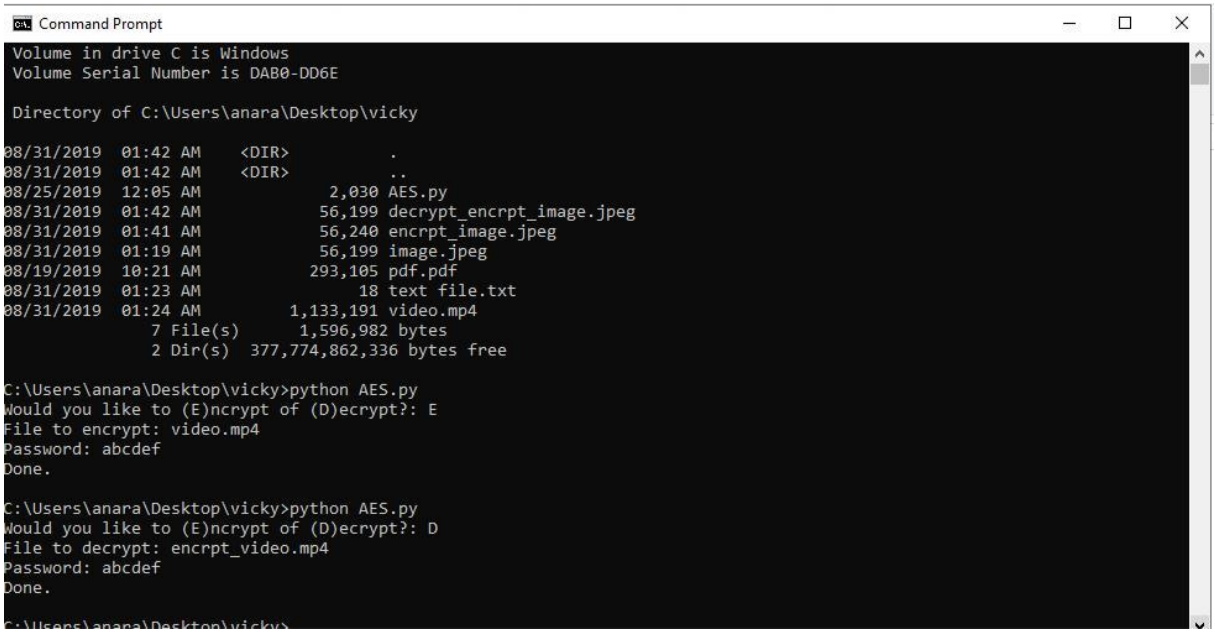


Fig 9.4.4 successfully decrypted of video

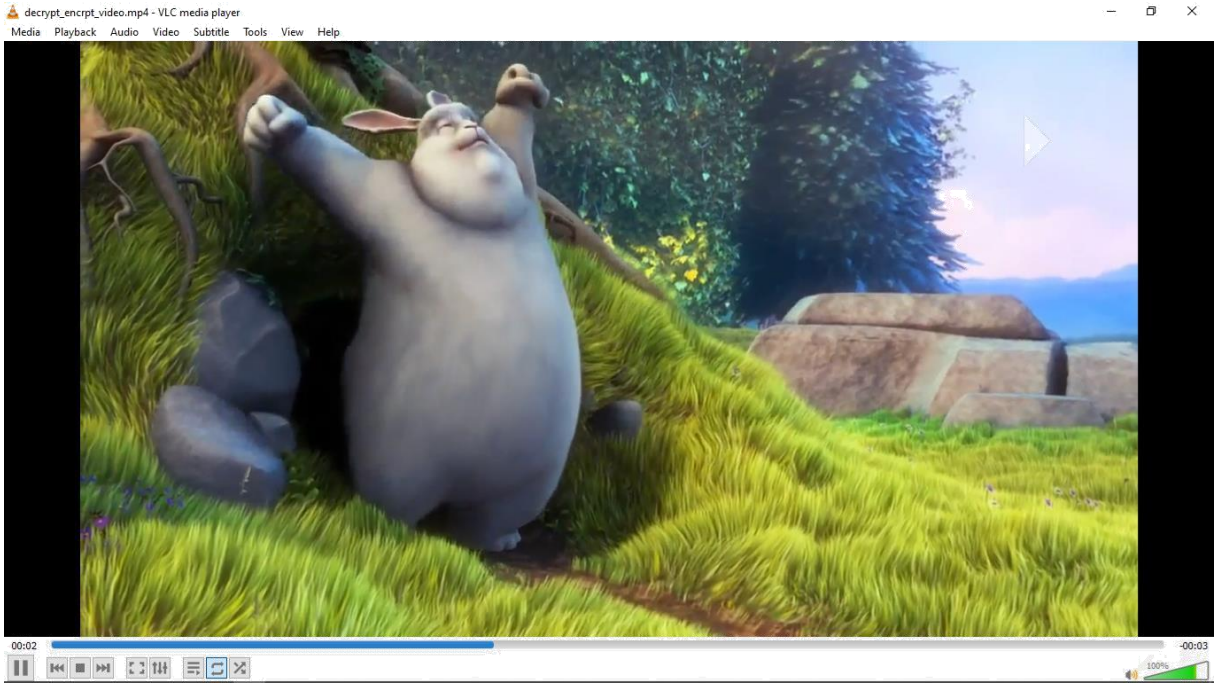


Fig 9.4.5 After decryption of video

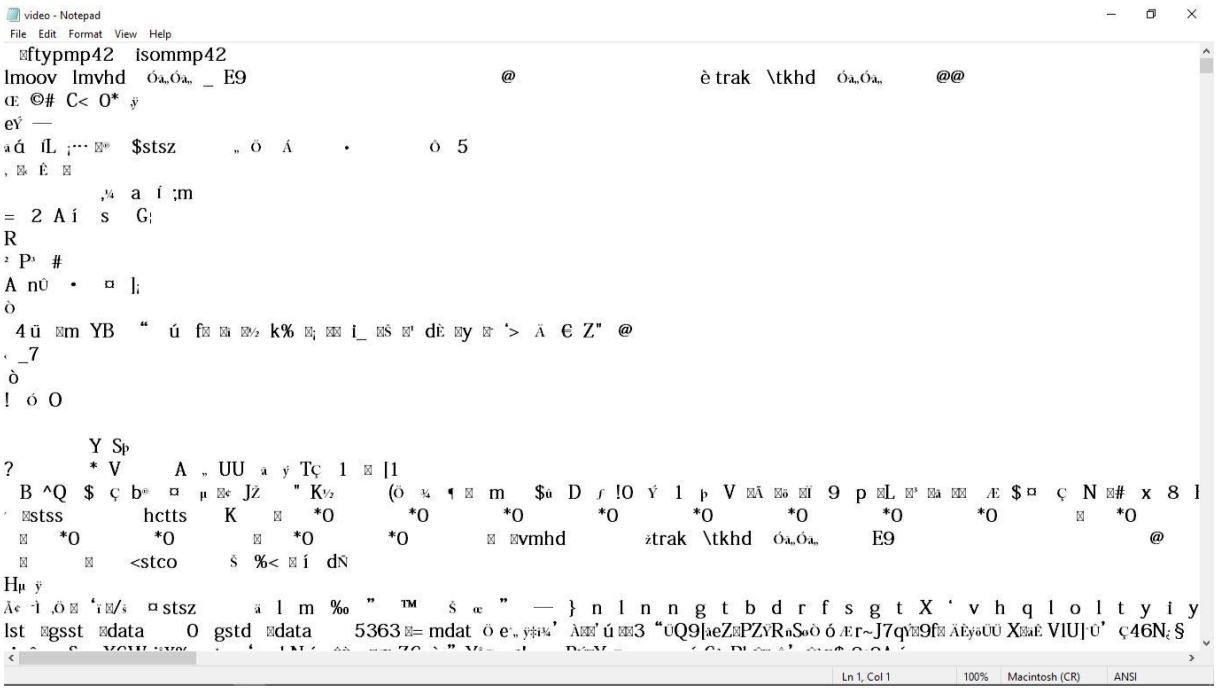


Fig 9.4.6 Before encryption open with notepad of video

Fig 9.4.7 After encryption open with notepad of video

CHAPTER 10

CONCLUSION AND FUTURE SCOPE:

In this project, we deal with the concepts of security of digital data communication across the network. This project is designed for combining the steganography and cryptography features factors for better performance. We performed a new steganography method and combined it with RSA algorithm. The data is hidden in the image so there will be no chances for the attacker to know that data is being hidden in the image. We performed our method on image by implementing a program written in Python language. The method proposed has proved successful in hiding various types of text, images, audio and videos in color images.

The Embedding of data is done such as Audio, Video, Image is done in the image, by choosing a distinct and new image, we can prevent the chance for the attacker to detect the data being hidden. Results achieved indicate that our proposed method is encouraging in terms of security, and robustness.

The project AES algorithm with python is for securing the data. It helps to secure the files, images, videos that no other person cannot understand the file after encrypted. After we can decrypt the data with the password. AES is the most secured and fast and they take many years to decrypt the encrypted data with password, it is very hard to decrypt the data without password

CHAPTER 11

REFRENCES:

[1] Aparna, V. S., et al. "Implementation of AES Algorithm on Text And Image using MATLAB." *2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI)*. IEEE, 2019.

[2] Lather, Yashpal, Megha Goyal, and Vivek Lather. "Review Paper on Steganography Techniques." *IJCSMC, Signal Processing* 4.1 (2015): 571-576.

[3] Benedict, Arnold Gabriel. "Improved file security system using multiple image steganography." *2019 International Conference on Data Science and Communication (IconDSC)*. IEEE, 2020.

[4] Karolin, M., T. Meyyappan, and S. M. Thamarai. "Encryption and decryption of color images using visual cryptography." *Int. J. Pure Appl. Math* 118 (2018): 277-281.

[5] Al-Haj, Ali, and Hiba Abdel-Nabi. "Digital image security based on data hiding and cryptography." *2017 3rd International conference on information management (ICIM)*. IEEE, 2017.

Books:

1. "Cryptography and Network Security: Principles and Practice" by William Stallings
2. Automate the boring stuff with python (2015). A practical programming approach.
3. "A Course in Number Theory and Cryptography" by Neal Koblitz
4. "Introduction to Cryptography" by Johannes A Buchmann
5. "Cryptography Theory and Practice" by Doug Stinson

Websites:

1. https://onlinecourses.nptel.ac.in/noc18_cs21/course
2. https://www.w3schools.com/python/
3. https://www.udemy.com/python
4. https://www.javatpoint.com/python-tutorial

5. https://stackoverflow.com/questions/8822300/online-python-tutorials