AES Implementation

Project Report Submitted in Partial Fulfilment of the Requirements for the Degree of

Bachelor of Engineering

in

Computer Science and Engineering

Submitted by

Fatema Bohra: (Roll No.19UCSE4023)

Pragya Sankhla: (Roll No.19UCSE4033)

Under the Supervision of

Mr. Anil Gupta
Professor

Department of Computer Science and Engineering
Faculty of Engineering & Architecture
M.B.M University, Jodhpur

June, 2022

Department of Computer Science &Engineering

M.B.M. University

Ratanada, Jodhpur, Rajasthan, India –342011

CERTIFICATE

This is to certify that the work contained in this report entitled **"AES Implementation**" is submitted by the group members Ms. Fatema Bohra(Roll. No:19UCSE4023), Ms. Pragya Sankhla (RollNo:19UCSE4033) to the Department of M.B.M University, Jodhpur, for the partialfulfilment of the requirements for the degree of**Bachelor of Engineering** in **Computer Science and Engineering**.

They have carried out their work under my supervision. This work has not been submitted else-where for the award of any other degree or diploma.

The project work in our opinion, has reached the standard fulfilling of the requirements for the degree of Bachelor of Engineering in Computer Science and Engineering in accordance with the regulations of the Institute.

Anil Gupta
Prof.
(Guide)
Dept. of Computer Science &Engg.
MBM University, Jodhpur

Prof. N.C. Barwar
(Head)
Dept. of Computer Science &Engg.
MBM University, Jodhpur

Acknowledgment

We would like to thanks " Dr. N.C. Barwar", Head of Department, Department of Computer Science and Engineering, M.B.M. University, Jodhpur, who gave us such a wonderful opportunity to work on this Final year project and expand my knowledge. We would also like to express our special thanks of gratitude to "Anil Gupta " for his valuable guidance and support in completion of our final year project. We would also like to thanks "Simran Chaudhery". her suggestions and instructions served as a major contributor towards the completion of the project. We got to learn a lot more about Cryptography and AES which will be very helpful for me in future.

Apart from this We would also like to thank our parents and friends who encouraged and helped us in various phases of the completion of the final year project.

Abstract

Cryptography is the study of mathematical techniques related to aspects of information security such as confidentiality, data integrity, entity authentication and data origin authentication. In data and telecommunications, cryptography is necessary when communicating over any unreliable medium, which includes any network particularly the internet. In this project, a GUI(Graphical user interface) of 128 bit AES encryption and Decryption by using Rijndael algorithm (Advanced Encryption Standard algorithm) is been made into a synthesizable using Python and tkinter library code which can be easily implemented on any python IDE. The algorithm is composed of three main parts: cipher, inverse cipher and Key Expansion. Cipher converts data to an unintelligible form called plaintext. Key Expansion generates a Key schedule that is used in cipher and inverse cipher procedure. Cipher and inverse cipher are composed of special number of rounds. For the AES algorithm, the number of rounds to be performed during the execution of the algorithm uses a round function that is composed of four different byte-oriented transformations: Sub Bytes, Shift Rows, Mix columns and Add Round Key.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Cryptography

Cryptography is the science of information security. The word is derived from the Greek kryptos, meaning hidden. Cryptography includes techniques such as microdots, merging words with images, and other ways to hide information in storage or transit. Modern cryptography intersects the disciplines of mathematics, computer science, and electrical engineering. Applications of cryptography include ATM cards, computer passwords, and electronic commerce. Cryptology prior to the modern age was almost synonymous with encryption, the conversion of information from a readable state to apparent nonsense. The sender retained the ability to decrypt the information and therefore avoid unwanted persons being able to read it. Since WWI and the advent of the computer, the methods used to carry out cryptology have become increasingly complex and its application more widespread .Modern cryptography follows a strongly scientific approach, and designs cryptographic algorithms around computational hardness assumptions, making such algorithms hard to break by an adversary. Such systems are not unbreakable in theory but it is infeasible to do so by any practical means. These schemes are therefore computationally secure. There exist secure schemes that provably cannot be broken--an example is the one-time pad--but these schemes are more difficult to implement than the theoretically breakable but computationally secure mechanisms.

In other words we can say that cryptography is defined as:- Cryptography is a method of protecting information and communications through the use of codes, so that only those for whom the information is intended can read and process it.

In computer science, cryptography refers to secure information and communication techniques derived from mathematical concepts and a set of rule-based calculations called algorithms, to transform messages in ways that are hard to decipher. These deterministic algorithms are used for cryptographic key generation, digital signing, verification to protect data privacy, web browsing on the internet and confidential communications such as credit card transactions and email.

1.1.1 Cryptography techniques

Cryptography is closely related to the disciplines of cryptology and cryptanalysis. It includes techniques such as microdots, merging words with images and other ways to hide information in storage or transit. However, in today's computer-centric world, cryptography is most often associated with scrambling plaintext (ordinary text, sometimes referred to as clear text) into ciphertext (a process called encryption), then back again (known as decryption). Individuals who practice this field are known as cryptographers.

Modern cryptography concerns itself with the following four objectives:

1. **Confidentiality.** The information cannot be understood by anyone for whom it was unintended.
2. **Integrity.** The information cannot be altered in storage or transit between sender and intended receiver without the alteration being detected.
3. **Non-repudiation.** The creator/sender of the information cannot deny at a later stage their intentions in the creation or transmission of the information.
4. **Authentication.** The sender and receiver can confirm each other's identity and the origin/destination of the information.

   Procedures and protocols that meet some or all of the above criteria are known as cryptosystems. Cryptosystems are often thought to refer only to mathematical procedures and computer programs; however, they also include the regulation of human behavior, such as choosing hard-to-guess passwords, logging off unused systems and not discussing sensitive procedures with outsiders.

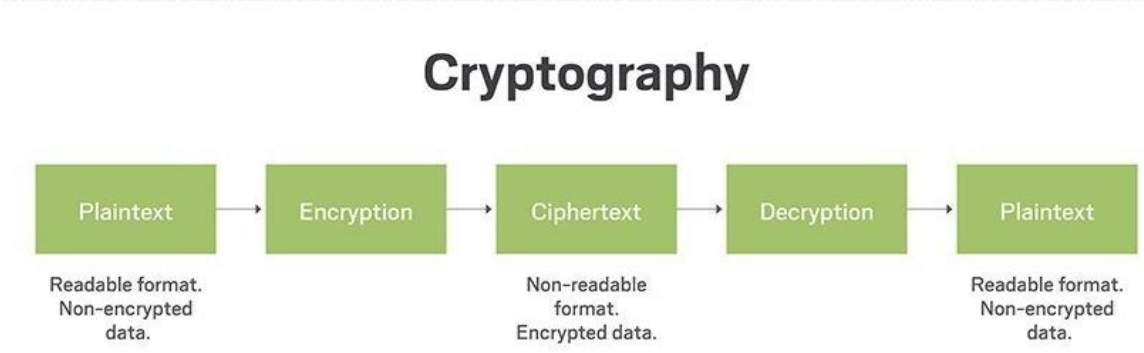


Fig 1.1 Cryptography General steps

1.1.2Cryptogrphyalgorit

Cryptosystems use a set of procedures known as cryptographic algorithms, or ciphers, to encrypt and decrypt messages to secure communications among computer systems, devices and applications.

A cipher suite uses one algorithm for encryption, another algorithm for message authentication and another for key exchange. This process, embedded in protocols and written in software that runs on operating systems (OSes) and networked computer systems, involves:

- public and private key generation for data encryption/decryption
- digital signing and verification for message authentication
- key exchange

1.1.3 Types of cryptography

Single-key or symmetric-key encryption algorithms create a fixed length of bits known as a block cipher with a secret key that the creator/sender uses to encipher data (encryption) and the receiver uses to decipher it. One example of symmetric-key cryptography is the Advanced Encryption Standard (AES). AES is a specification established in November 2001 by the National Institute of Standards and Technology (NIST) as a Federal Information Processing Standard (FIPS 197) to protect sensitive information. The standard is mandated by the U.S. government and widely used in the private sector.

In June 2003, AES was approved by the U.S. government for classified information. It is a royalty-free specification implemented in software and hardware worldwide. AES is the successor to the Data Encryption Standard (DES) and DES3. It uses longer key lengths -- 128-bit, 192-bit, 256-bit -- to prevent brute force and other attacks.

Public key or symmetric key encryption algorithms use a pair of keys, a public key associated with the creator/sender for encrypting messages and a private key that only the originator knows (unless it is exposed or they decide to share it) for decrypting that information.

Examples of public-key cryptography include

Elliptic Curve Digital Signature Algorithm (ECDSA) used by Bitcoin

- Digital Signature Algorithm (DSA) adopted as a Federal Information Processing Standard for digital signatures by NIST in FIPS 186-4
- Diffie-Hellman key exchange

To maintain data integrity in cryptography, **hash functions**, which return a deterministic output from an input value, are used to map data to a fixed data size. Types of cryptographic hash functions include SHA-1 (Secure Hash Algorithm 1), SHA-2 and SHA-3.

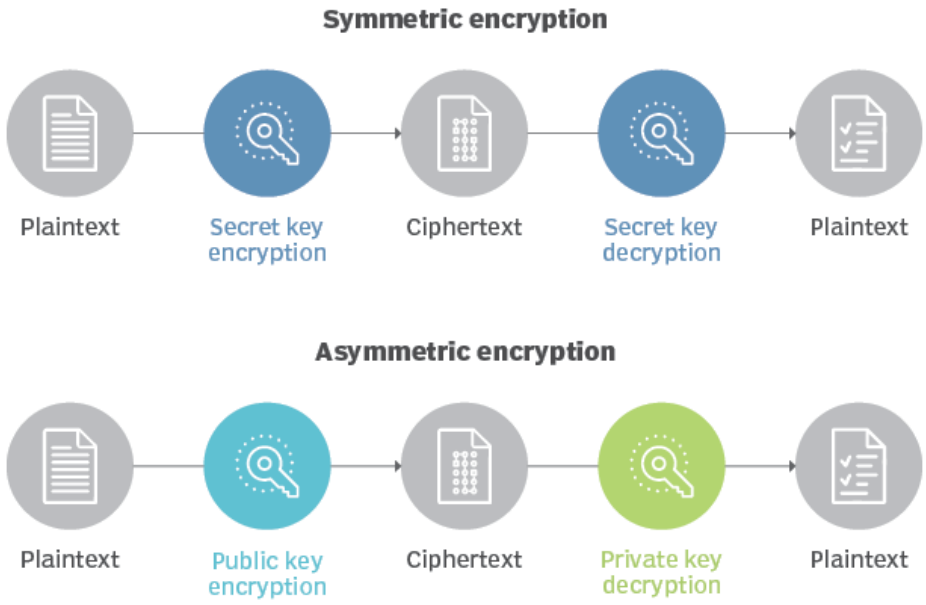


Fig1.2 Symmetric and Asymmetric Encryption

1.1.4 Cryptography concerns

Attackers can bypass cryptography, hack into computers that are responsible for data encryption and decryption, and exploit weak implementations, such as the use of default keys. However, cryptography makes it harder for attackers to access messages and data protected by encryption algorithms.

Growing concerns about the processing power of quantum computing to break current cryptography encryption standards led NIST to put out a call for papers among the mathematical and science community in 2016 for new public key cryptography standards.

Unlike today's computer systems, quantum computing uses quantum bits (qubits) that can represent both 0s and 1s, and therefore perform two calculations at once. While a large-scale quantum computer may not be built in the next decade, the existing infrastructure requires standardization of publicly known and understood algorithms that offer a secure approach, according to NIST. The deadline for submissions was in November 2017, analysis of the proposals is expected to take three to five years.

1.2. DATA ENCRYPTION STANDARDS (DES)

The Data Encryption Standard (DES) provides a single, standard, cryptographic algorithm for protecting computer and communications information. This standard was first approved by the NBS (National Bureau of Standards now NIST, National Institute of Standards and Technology) in 1977 as FIPS Pub 46 . DES is a Feistel-type block cipher that operates on 64 bit blocks of data using a 56 bit key.

Feistel Ciphers operate on the left and right halves of a block of bits in multiple rounds. An interesting property of Feistel Ciphers is that the function f, used to operate on the half-blocks of data bits, does not need to be invertible for the Feistel Cipher to be invertible. In the Data Encryption Algorithm, the function f is a product cipher, because the function performs both substitutions and permutations.

1.2.1 DES Algorithm Overview

The DES algorithm is a symmetric block cipher. For each plaintext block of 64-bits, a 56-bit key is used to produce a 64-bit block of ciphertext. The same 56-bit key is used to recover plaintext when cipher text is input. Encryption and decryption use the same key and algorithm, the only difference being the generation of subkeys. DES processes input blocks through permutations, initial and inverse initial, and 16 equivalent rounds that consist of permutations, summing, and bit-wise manipulations. The initial permutation simply routes input bits to different locations for processing. For example, the first bit is routed to location 40, and bit 58 is routed to location one. FIPS Pub-46 fully explains this rerouting. The inverse initial permutation merely reverses the initial permutation. Between the initial and inverse initial permutations, the block is processed by 16 rounds as shown in. The further details of each round are provided by FIPS Pub-46 . After the initial permutation, the 64-bit block is broken into two 32-bit halves.

The right half passes through an expansion permutation. This replicates several bits to expand from 32-bit to 48-bits. FIPS Pub-46 has full details for this expansion and Schneier explains its significance. The right half also exits the round as the left half output. The output of the expansion permutation is combined with the appropriate subkey in a 48-bit exclusive-or. This output becomes the input for eight s-boxes. Each sbox produces 4-bits of output for 6-bits of input. Specific s-box definitions are given in FIPS Pub-46 . The s-boxes generate 32-bits of outputs that pass to another permutation called the p-box. The p-box simply routes each input to a different location of output as defined in FIPS Pub-46 . A 32-bit exclusive-or combines the p-box output with the left half of the block input. The exclusive-or output then becomes the 32-Bit, right half output for the round.

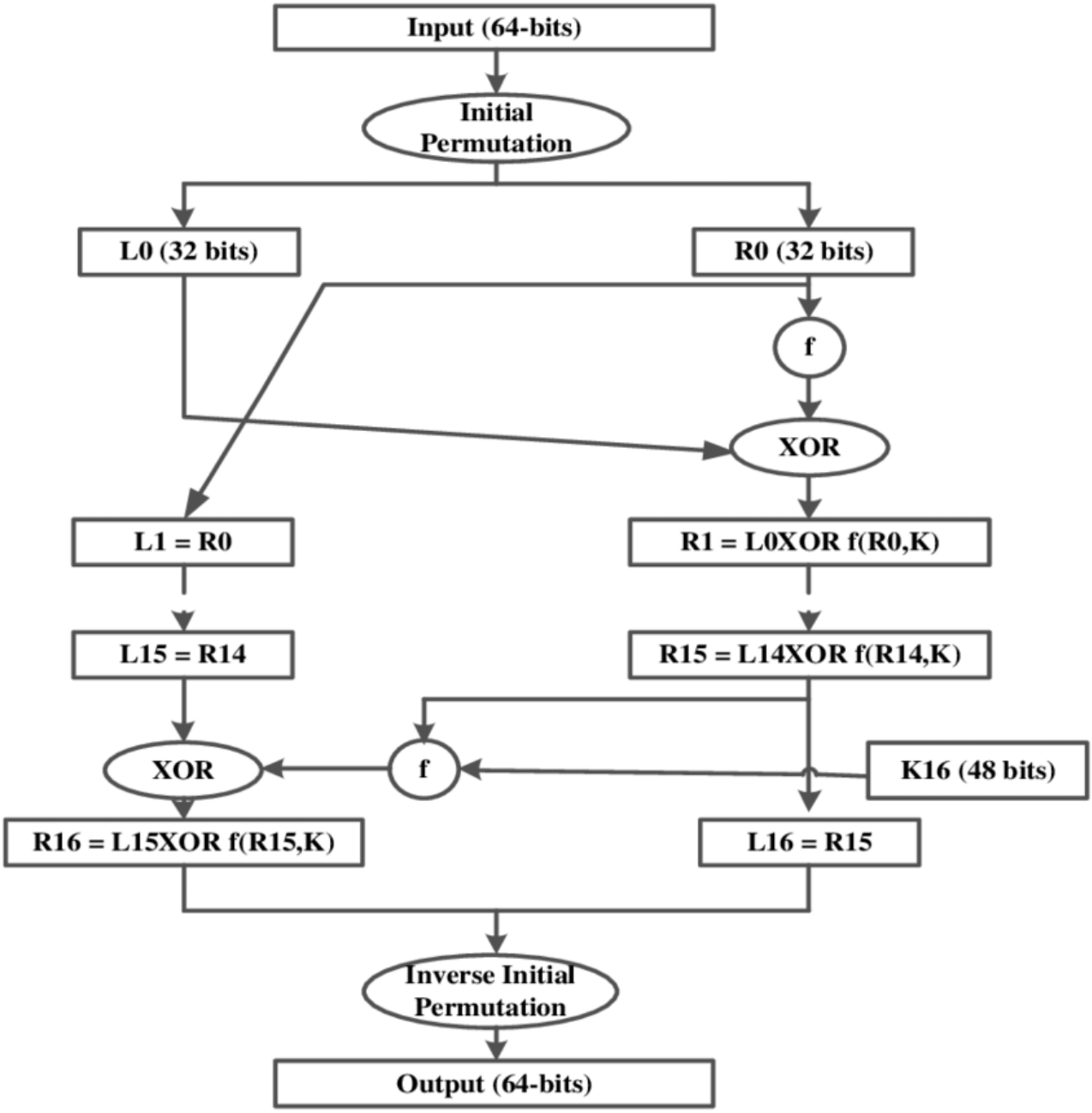


Fig 1.3 Flowchart of DES algorithm

1.2.2 Triple DES

Triple DES employs the Data Encryption Standard algorithm in a way sometimes referred to as encrypt-decrypt-encrypt (EDE) mode. EDE mode using two keys, was proposed by W. Tuchman and summarized by Schneier in . The incoming plaintext is encrypted with the first key, decrypted with the second key, and then encrypted again with the first key. On the other end, the received ciphertext is decrypted with the first key, encrypted with the second key, and again decrypted with the first key to produce plaintext. If the two keys are set alike, it has the effect of single encryption with one key, thereby preserving backward compatibility. Two key, triple DES schemes (with 56 bit keys) can be cryptanalyzed using a chosen plaintext attack with about 256 operations and 256 words of memory . Although in theory this is a weakness, Merkle and Hellman state that in practice it is very difficult to mount a chosen plaintext attack against a DES cryptosystem. This makes two key, triple DES significantly stronger than two key, double DES, because an attack would now require 2112 operations (and no memory). (Two key, double DES is susceptible to a known plaintext attack with 256 operations and 256 words of memory .)

Triple DES can also be performed with three independent keys, using one key for each of the encryption and decryption operations. Triple DES with three independent keys gives a higher level of protection, requiring about 2112 operations and 256 words of memory . Again, with regards to compatibility, keys one and three could be set to the same value, to interoperate with Tuchman's two key, triple DES, or all three keys could be set alike to interoperate with single DES. Because keys and control information march in lock step with the data, multiple SNL DES ASICs can be cascaded to provide the encrypt-decrypt-encrypt mode. Alternatively, data, key, and control information can be looped back, wrapping around the SNL DES ASIC to perform E-D-E triple DES using only one ASIC .

1.2.3 DRAWBACKS OF DES

1. The 56-bit key size is the biggest defect of DES. Chips to perform one million of DES encrypt or decrypt operations a second are available (in 1993). A $1 million DES cracking machine can search the entire key space in about 7 hours.

2. Hardware implementations of DES are very fast; DES was not designed for software and hence runs relatively slowly.

3. As the technology is improving lot more day by day so there is a possibility to break the encrypted code, so AES is preferred than DES.

4. As we know in DES only one private key is used for encryption as well as for decryption because it is symmetric encryption technique so if we lost that key to decrypt the data then we cannot get the readable data at the receiving end.

1.3 ADVANCE ENCRYTPTION SYSTEM (AES)

AES Algorithm AES is short for Advanced Encryption Standard and is a United States encryption standard defined in Federal Information Processing Standard (FIPS) 192. AES is the most recent of the four current algorithms approved for federal us in the United States. AES is a symmetric encryption algorithm processing data in block of 128 bits. AES is symmetric since the same key is used for encryption and the reverse transformation, decryption . The only secret necessary to keep for security is the key. AES may configured to use different key-lengths, the standard defines 3 lengths and the resulting algorithms are named AES-128, AES-192 and AES-256 respectively to indicate the length in bits of the key. The older standard, DES or Data Encryption Standard. DES is upto 56bits only . To overcome the disadvantages of des algorithm, the new standard is AES algorithm. this standard explicitly defines the allowed values for the key length (Nk), block size (Nb), and number of rounds (Nr).

1.3.1 AES Algorithm Specification

For the AES algorithm, the length of the input block, the output block and the State is 128 bits. This is represented by Nb = 4, which reflects the number of 32-bit words (number of columns) in the State. AES Algorithm Specification For the AES algorithm, the length of the input block, the output block and the State is 128 bits. This is represented by Nb = 4, which reflects the number of 32-bit words (number of columns) in the State.

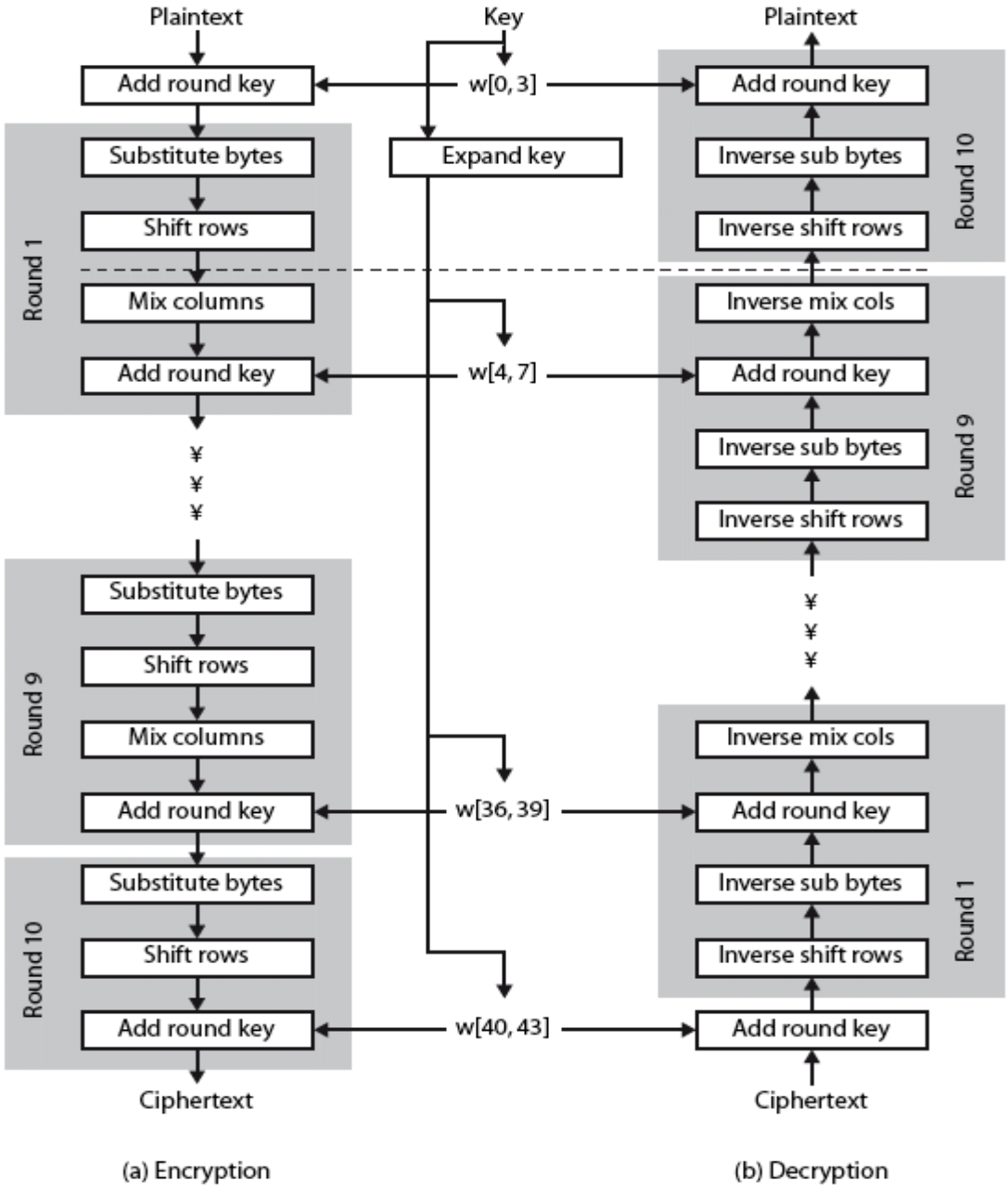


Fig1.4 General Structure Of AES Algorithm

An implementation of the AES algorithm shall support at least one of the three key lengths: 128, 192, or 256 bits (i.e., Nk = 4, 6, or 8, respectively). Implementations may optionally support two or three key lengths, which may promote the interoperability of algorithm implementations. For the AES algorithm, the length of the Cipher Key, K, is 128, 192 or 256 bits. The key length is represented by Nk = 4, 6, or 8which reflects the number of 32-bit words (number of columns) in the Cipher Key. For the AES algorithm, the number of rounds to be performed during the execution of the algorithm is dependent on the key size. The number of rounds is represented by Nr, where Nr = 10 when Nk = 4, Nr = 12 when Nk = 6, and Nr = 14 when Nk = 8. The only Key-Block-Round combinations that conform to this standard are given in Table.

Algorithm	Block Size (Nb words)	Key Length (Nk words)	Number of Rounds (Nr)
AES-128	4	4	10
AES-192	4	6	12
AES-256	4	8	14

Table 1.1 Key Block Round Combinations

For both its Cipher and Inverse Cipher, the AES algorithm uses a round function that is composed of four different byteoriented transformations:

1) Byte substitution using a substitution table (S-box),

2) Shifting rows of the State array by different offsets,

3) Mixing the data within each column of the State array, and

4) Adding a Round Key to the State.

1.3.2 ENCRYPTION

In encryption mode, the initial key is added to the input value at the very beginning, which is called an initial round. This is followed by 9 iterations of a normal round and ends with a slightly

modified final round, as one can see in Figure. During one normal round the following operations are performed in the following order: Sub Bytes, Shift Rows, Mix Columns, and Add Round key. The final round is a normal round without the Mix Columns stage.

A. Steps in AES Encryption

- Sub Bytes—a non-linear substitution step where each byte is replaced with another according to a lookup table.
- Shift Rows—a transposition step where each row of the state is shifted cyclically a certain number of steps.
- Mix Columns—a mixing operation which operates on the columns of the state, combining the four bytes in each column.
- Add Round Key—each byte of the state is combined with the round key; each round key is derived from the cipher key using a key schedule.

1. Sub bytes Transformation

The Sub Bytes transformation is a non-linear byte substitution that operates independently on each byte of the State using a substitution table (S-box). This S-box which is invertible is constructed by composing two transformations:

1. Take the multiplicative inverse in the finite field GF (28), the element {00} is mapped to itself.

2. Apply the following affine transformation (over GF (2)): For 0<i. the affine transformation element of the S-box can be expressed as:

$$\begin{bmatrix} b_0' \\ b_1' \\ b_2' \\ b_3' \\ b_4' \\ b_5' \\ b_6' \\ b_7' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}.$$

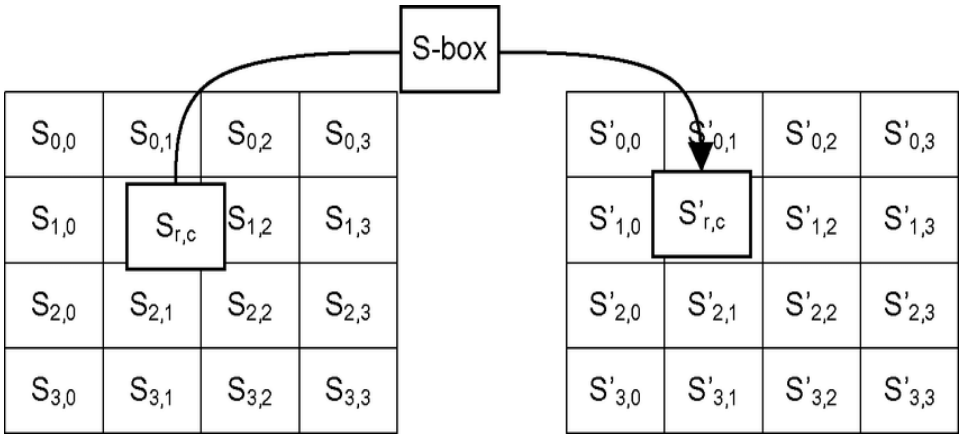


Fig 1.5 Sub Byte Transformation

2. shift rows transformation

In the Shift Rows transformation, the bytes in the last three rows of the State are cyclically shifted over different numbers of bytes (offsets). The first row is not shifted at all, the second row is shifted by one the third row by two, and the fourth row by three bytes to the left. Specifically, the Shift Rows transformation proceeds as follows: The shift value shift(r,Nb) depends on the row number, r, as follows (recall that Nb = 4): shift(1,4) 1; shift(2,4) 2 ; shift(3,4) 3 .This has the effect of moving bytes to ―lower‖ positions in the row (i.e., lower values of c in a given row), while the ―lowest‖ bytes wrap around into the ―top‖ of the row (i.e., higher values of c in a given row)

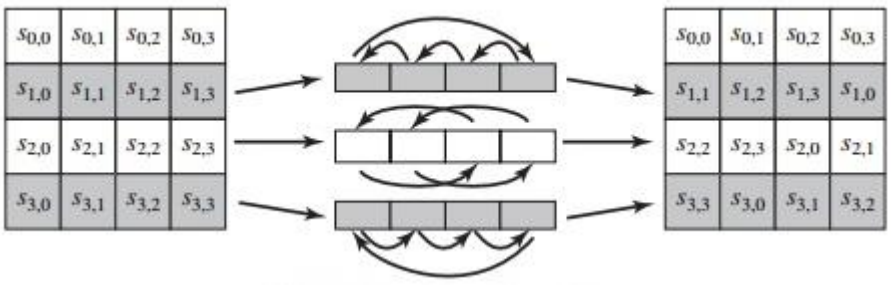


Fig 1.6 Shift Rows

3. MixColumns Transformation

The Mix Columns transformation operates on the State column-by-column, treating each column as a four-term polynomial. As a result of this multiplication, the four bytes in a column are replaced.

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 02\ 03\ 01\ 01 \\ 01\ 02\ 03\ 01 \\ 01\ 01\ 02\ 03 \\ 03\ 01\ 01\ 02 \end{bmatrix} \times \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

Fig 1.7 Mix Column Transformation

4. Add round Key Transformation

In the Add Round Key transformation, a Round Key is added to the State by a simple bitwise XOR operation. Each Round Key consists of Nb words from the key schedule.Those Nb words are each added into the columns of the State, such that [wi] are the key schedule words, and round is a value in the range 0 round Nr. In the Cipher, the initial Round Key addition occurs when round = 0, prior to the first application of the round function. The application of the Add Round Key transformation to the Nr rounds of the Cipher occurs when 1<round <**Nr**.The action of this transformation is illustrated
in Fig., where l =round *__Nb__.

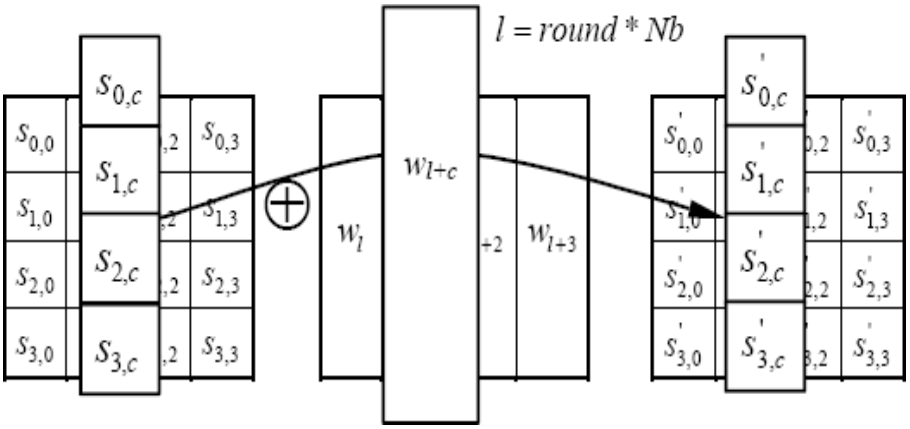


Fig 1.8 Add Round Key

1.3.3 Key Expansion

The AES algorithm takes the Cipher Key, *K*, and perform Key Expansion routine together at key schedule. The Key Expansion generates a total of *Nb*(*Nr*+1) words: the algorithm requires an initial set of *Nb* words, and each of the *Nr* rounds requires *Nb* words of key data.Theresultingkeyscheduleconsistsofalinear4-bytewords,denoted[*wi*], with *i* in the range 0<*i* <*Nb*(*Nr* + 1). The expansion of the input key into the key schedule

proceeds according to the pseudo code. Sub Word is a function that takes a four-byte input word and applies the S-box to each of the four bytes to produce an output word. The function Rot Word takes a word[*a*0,*a*1,*a*2,*a*3] perform cyclic permutation, and returns the word [*a*1,*a2,a3,a0*]. The round constant word array,Rcon[i],containsthevaluesgivenby[*xi-1*,{00},{00},{00}],with*xi*-1beingpowersof*x*(*x*isdenotedas{02})inthefieldGF(28). The first *Nk* words of the expanded key are filled with the Cipher Key .Every following word, w[i], is equal to the XOR of the previous word, w[i-1], and the word *Nk* positions earlier ,w[i-*Nk*]. Forwords in positions that are a multiple of*Nk*,atransformationisappliedtow[i-1]priortotheXOR,followedbyanXORwitharoundconstant,Rcon[i].Thistransformation consists of a cyclic shift of the bytes in a word(RotWord), followed by the application of a table lookup to all four bytes of the word(SubWord). It is important to note thatthe Key Expansion routine for 256-bit Cipher Keys (*Nk* = 8) isslightlydifferentthanfor128-and192-bitCipherKeys.If*Nk* =8andi-4isamultipleof*Nk*,thenSubWord() is applied to w [i-1] prior to the XOR.

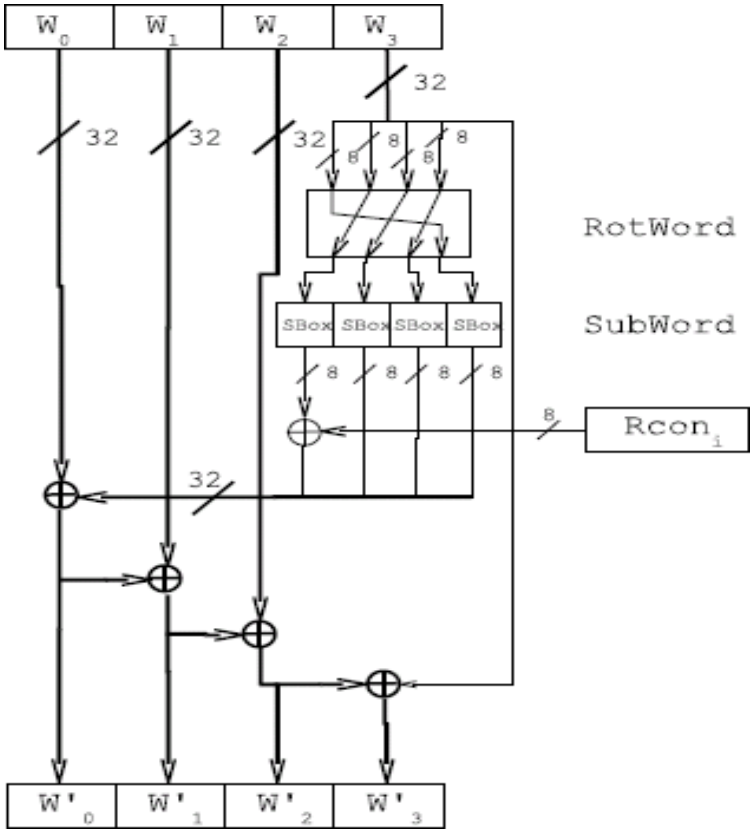


Fig 1.9KeyExpansion

1.3.4 DECRYPTION

In decryption mode, the operations are in reverse order compared to their order in encryption mode. Thus it starts with an initial round, followed by 9 iterations of an inverse normal round and ends with an Add Round Key. An in verse normal round consists of the following operations in this order: AddRoundKey, Inv MixColumns, Inv ShiftRows, and Inv SubBytes. An initial round is an inverse normal round without the InvMixColumns.

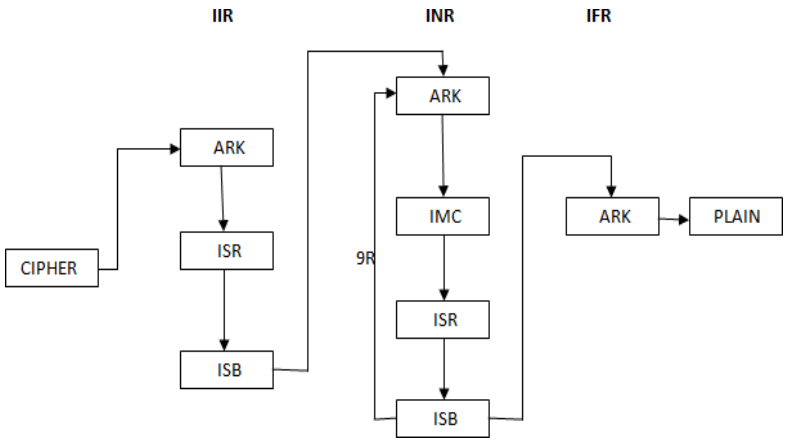


Fig 1.10Decryption

A. **Inv Shift rows Transformation**

InvShiftRows is the inverse of the ShiftRows transformation. The bytes in the last three rows of the State are cyclically shifted over different numbers of bytes (offsets). 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.

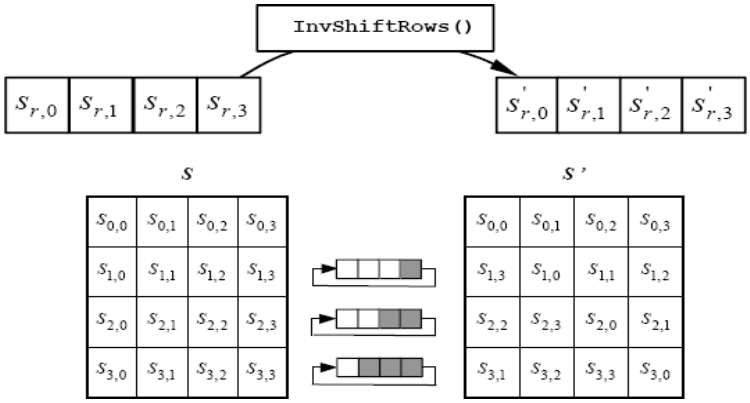


Fig 1.11 Inv Shift Rows transformation

B.**Inv Sub bytes Transformation**

Inv Sub Bytes is the inverse of the bytes ,in which the inverse Sbox is applied to each byte of the State. This is obtained by applying the inverse of the affine transformation followed by taking the multiplicative inverse in GF. The inverse S-box used in the Inv Sub Bytes() transformation is presented in Fig

52	09	6A	D5
7C	E3	39	82
54	7B	94	32
08	2E	A1	66

Table 1.2 Inverse S-BOX Table

A. Inv Mix Column Transformation

InvMixColumnsistheinverseoftheMixColumnstransformation.InvMixColumnsoperatesont heStatecolumn-by-column,treating each column as a four term polynomial. The columns are considered as polynomials over GF (28) and multiplied modulo $x4 + 1$ with a fixed polynomial$a$-1(x), given by a-1(x) = {0b} $x3$ + {0d} $x2$ + {09} x + {0e}, this can be written as a matrix multiplication.

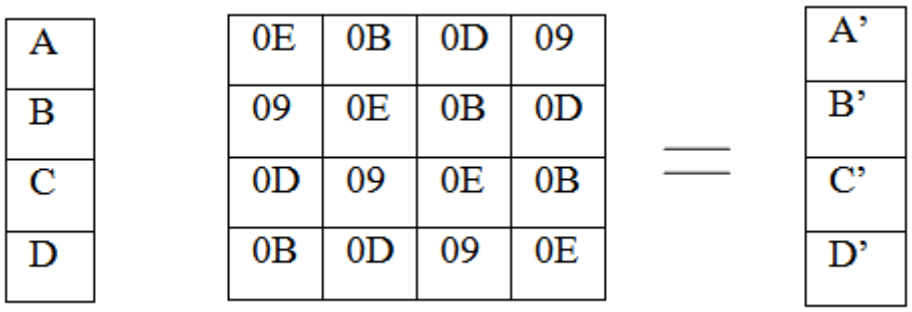


Fig 1.12 Inverse mix column transformation

B. Inverse of the Add round key Transformation

AddRoundKey is its own inverse, since it only involves an application of the XOR operation. Equivalent Inverse Cipher transformations differ from that of the Cipher, while the form of the key schedules for encryption and decryption remains the same. However, several properties of the AES algorithm allow for an Equivalent Inverse Cipher that has the same sequence of transformations as the Cipher (with the transformations replaced by their inverses).This is accomplished with a change in the key schedule. The two properties that allow for this Equivalent Inverse Cipher are as follows: The Sub Bytes and Shift Rows transformations commute; that is, a Sub Bytes transformation immediately followed by a Shift Rows transformation is equivalent to a Shift Rows transformation immediately followed by a Sub Bytes transformation.

The same is true for their inverses, InvSubBytes and InvShiftRows. The column mixing

operations - MixColumnsand InvMixColumns – are linear with respect to the column input, which means InvMixColumns(state XOR RoundKey) =InvMixColumns(state) XOR InvMixColumns (RoundKey).These properties allow the order of InvSubBytes and InvShiftRows transformation obereversed. The order of the AddRoundKey and InvMixColumns transformation oberevered ,provided that the columns(words) of the decryption key schedule are modified using the InvMixColumns transformation .The equivalent inverse cipher is defined by reversing the order of the InvSubBytes and InvShiftRows transformations and by reversing the order of the AddRoundKey and InvMixColumns transformations used in the―round loop ‖after first modifying the decryption key schedule for *round* =1to*Nr*- 1 using the InvMixColumns transformation.The first and last *Nb* words of the decryption Key schedule shall *not* be modified in this manner.

1.3.5 AES Applications

AES Encryption and Decryption has many applications .It is used in cases where data is too sensitive that only the authorized people are supposed to know and not to the rest. The following are the various applications

Secure Communication

- -Smart Cards
- - RFID.
- - ATM networks.
- - Image encryption

Secure Storage

- -Confidential Cooperate Documents
- - Government Documents
- - FBI Files
- - Personal Storage Devices
- - Person Information Protection

1.4 Mode of Operation in Cryptography

Modes of operation of a block cipher are procedural rules for a generic block cipher. The different modes of operation result in different properties being achieved which add to the security of the underlying block cipher in the cryptography.

The block cipher processes the data blocks of fixed size in the character. Eventually, the size of a message is larger than the block size in the modes. Hence, the long message is divided into a series of sequential message blocks which divides it into blocks, and the cipher operates on these blocks one at a time in the cryptography.

→ Types of mode of operations

There are 5 types of mode of operation,

1. Electronic Code Book (ECB).
2. Cipher Block Chaining (CBC).
3. Cipher feedback (CFB).
4. Output Feedback (OFB).
5. Counter Mode (CTR).

→Electronic Code Book (ECB) –

Electronic code book is the easiest block cipher mode of functioning. It is easier because of direct encryption of each block of input plaintext and output is in form of blocks of encrypted ciphertext. Generally, if a message is larger than *b* bits in size, it can be broken down into a bunch of blocks and the procedure is repeated.
Procedure of ECB is illustrated below:

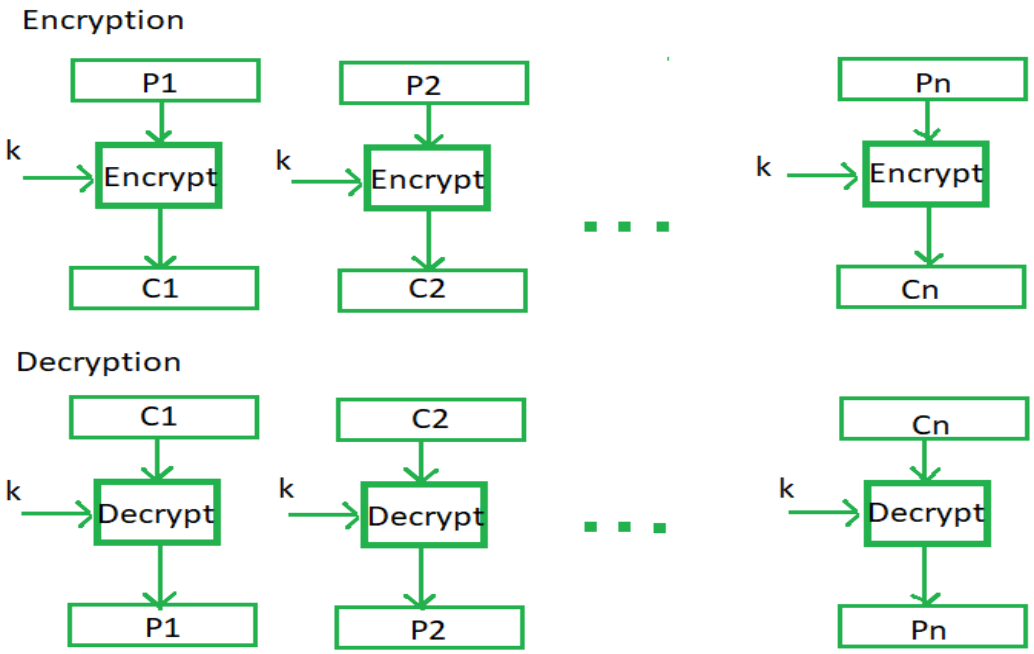


Fig 1.13Encryption and Decryption Process for ECB

Advantages of using ECB –

- Parallel encryption of blocks of bits is possible, thus it is a faster way of encryption.
- Simple way of the block cipher.

Disadvantages of using ECB –

- Prone to cryptanalysis since there is a direct relationship between plaintext and ciphertext.

→Cipher Block Chaining–

Cipher block chaining or CBC is an advancement made on ECB since ECB compromises some security requirements. In CBC, the previous cipher block is given as input to the next encryption algorithm after XOR with the original plaintext block. In a nutshell here, a cipher block is produced by encrypting an XOR output of the previous cipher block and present plaintext block.

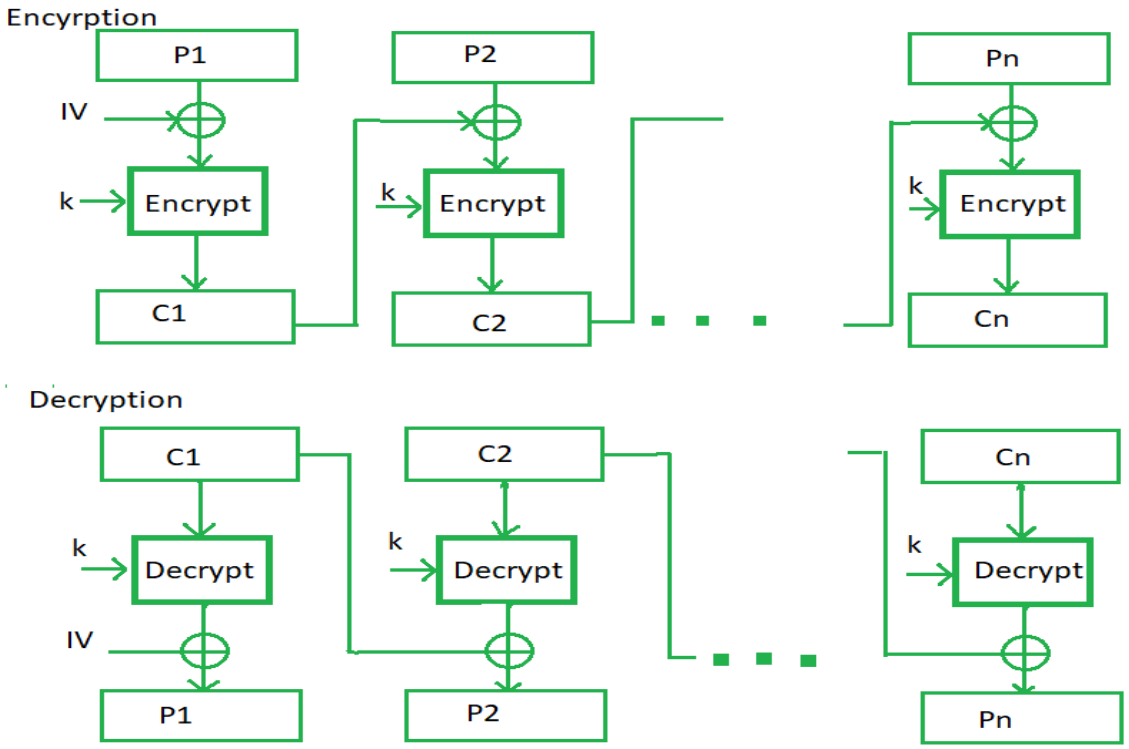


Fig 1.14 Encryption and Decryption Process for CBC

Advantages of CBC –

- CBC works well for input greater than *b* bits.
- CBC is a good authentication mechanism.
- Better resistive nature towards cryptanalysis than ECB.

Disadvantages of CBC –

- Parallel encryption is not possible since every encryption requires a previous cipher.

→Cipher Feedback Mode (CFB) –

In this mode the cipher is given as feedback to the next block of encryption with some new specifications: first, an initial vector IV is used for first encryption and output bits are divided as a set of *s* and *b*-*s* bits.The left-hand side *s* bits are selected along with plaintext bits to which an XOR operation is applied. The result is given as input to a shift register having b-s bits to lhs,

bits to rhs and the process continues. The encryption and decryption process for the same is shown below, both of them use encryption algorithms.

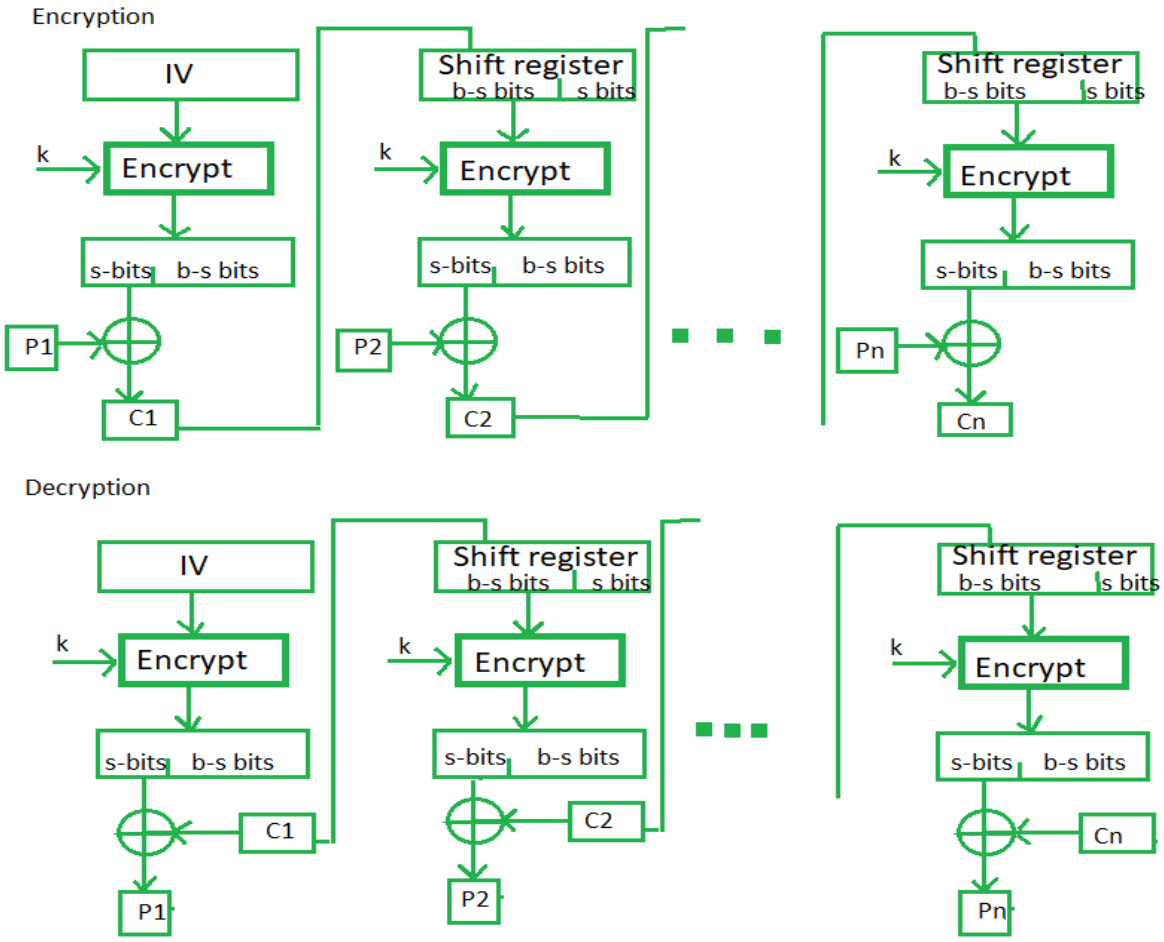


Fig 1.15 Encryption and Decryption Process for CFB

Advantages of CFB –

- Since, there is some data loss due to the use of shift register, thus it is difficult for applying cryptanalysis.

→Output Feedback Mode –

The output feedback mode follows nearly the same process as the Cipher Feedback mode except that it sends the encrypted output as feedback instead of the actual cipher which is XOR output. In this output feedback mode, all bits of the block are sent instead of sending selected *s* bits. The Output Feedback mode of block cipher holds great resistance towards bit transmission errors. It also decreases the dependency or relationship of the cipher on the plaintext.

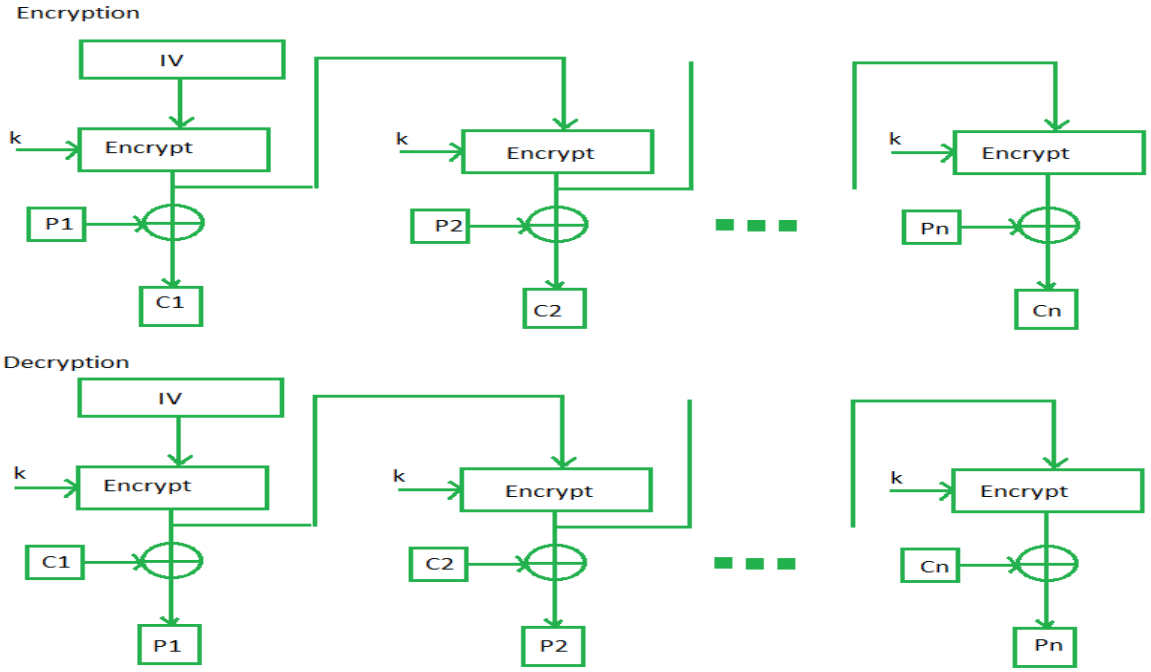


Fig 1.16 Encryption and Decryption Process for OFB

Advantages of OFB –

- In the case of CFB, a single bit error in a block is propagated to all subsequent blocks. This problem is solved by OFB as it is free from bit errors in the plaintext block.

→ Counter Mode –

The Counter Mode or CTR is a simple counter-based block cipher implementation. Every time a counter-initiated value is encrypted and given as input to XOR with plaintext which results in ciphertext block. The CTR mode is independent of feedback use and thus can be implemented in parallel.

Advantages of Counter –

- Since there is a different counter value for each block, the direct plaintext and cipher text relationship is avoided. This means that the same plain text can map to different cipher text

- Parallel execution of encryption is possible as outputs from previous stages are not chained as in the case of CBC.

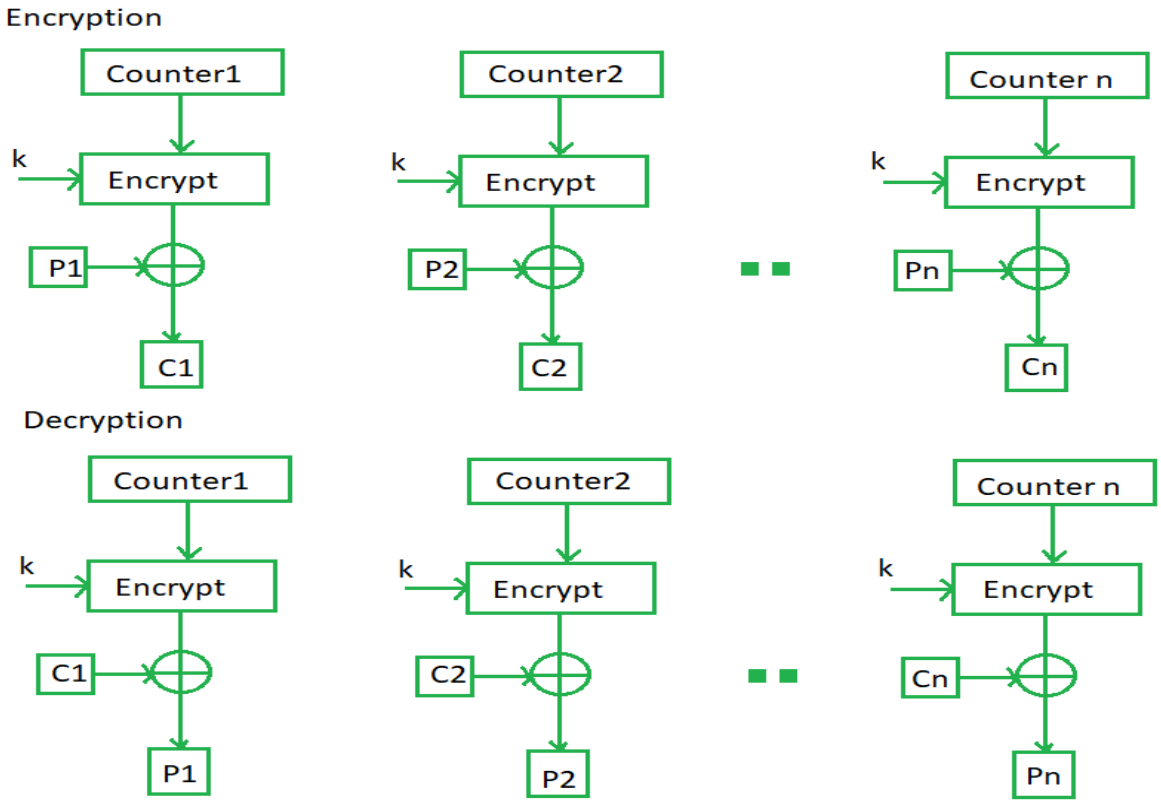


Fig 1.17 Encryption and Decryption Process for CTR

Chapter 2

Technical details

2.1 Dependencies -

- Numpy - Python's Array Processing Library
- Tinkter - Python's GUI Library
- Pillow - Python's Image Processing Library

Installing Dependencies -

- Install PIP (Python Package Installer) -
- sudo apt update
- sudo apt install python3-pip
- pip3 --version

Numpy -

- pip3 install numpy

Pillow -

- pip3 install Pillow

Tkinter -

- sudo apt-get install python-tk

2.2 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 in 2005 by Travis Oliphant. It is an open source project and you can use it freely.
- NumPy stands for Numerical Python.
- In Python we have lists that serve the purpose of arrays, but they are slow to process.
- NumPy aims to provide an array object that is up to 50x faster than traditional Python lists.
- The array object in NumPy is called ndarray, it provides a lot of supporting functions that make working with ndarray very easy.
- Arrays are very frequently used in data science, where speed and resources are very important.

2.3 Tkinter

Python provides various options for developing graphical user interfaces (GUIs).

Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit.

Creating a GUI application using Tkinter is an easy task. All you need to do is perform the following steps −

Import the Tkinter module.

- Create the GUI application main window.
- Add one or more of the above-mentioned widgets to the GUI application.
- Enter the main event loop to take action against each event triggered by the user.

Tkinter Widgets

Tkinter provides various controls, such as buttons, labels and text boxes used in a GUI application. These controls are commonly called widgets.

There are currently 15 types of widgets in Tkinter. We present these widgets as well as a brief description in the following table –

Sr.No.	Operator & Description
1	**Button** The Button widget is used to display buttons in your application.
2	**Canvas** The Canvas widget is used to draw shapes, such as lines, ovals, polygons and rectangles, in your application.

3	**Checkbutton** The Checkbutton widget is used to display a number of options as checkboxes. The user can select multiple options at a time.
4	**Entry** The Entry widget is used to display a single-line text field for accepting values from a user.
5	**Frame** The Frame widget is used as a container widget to organize other widgets.
6	**Label** The Label widget is used to provide a single-line caption for other widgets. It can also contain images.
7	**Listbox** The Listbox widget is used to provide a list of options to a user.
8	**Menubutton** The Menubutton widget is used to display menus in your application.
9	**Menu** The Menu widget is used to provide various commands to a user. These commands are contained inside Menubutton.
10	**Message** The Message widget is used to display multiline text fields for accepting values from a user.
11	**Radiobutton** The Radiobutton widget is used to display a number of options as radio buttons. The user can select only one option at a time.

12	**Scale** The Scale widget is used to provide a slider widget.
13	**Scrollbar** The Scrollbar widget is used to add scrolling capability to various widgets, such as list boxes.
14	**Text** The Text widget is used to display text in multiple lines.
15	**Toplevel** The Toplevel widget is used to provide a separate window container.
16	**Spinbox** The Spinbox widget is a variant of the standard Tkinter Entry widget, which can be used to select from a fixed number of values.
17	**PanedWindow** A PanedWindow is a container widget that may contain any number of panes, arranged horizontally or vertically.
18	**LabelFrame** A labelframe is a simple container widget. Its primary purpose is to act as a spacer or container for complex window layouts.
19	**tkMessageBox** This module is used to display message boxes in your applications

Table 2.1 Tkinter widgets

Standard attributes

Let us take a look at how some of their common attributes.such as sizes, colors and fonts are specified.

- Dimensions

- Colors
- Fonts
- Anchors
- Relief styles
- Bitmaps
- Cursors

Let us study them briefly −

Geometry Management

All Tkinter widgets have access to specific geometry management methods, which have the purpose of organizing widgets throughout the parent widget area. Tkinter exposes the following geometry manager classes: pack, grid, and place.

The pack() Method − This geometry manager organizes widgets in blocks before placing them in the parent widget.

The grid() Method − This geometry manager organizes widgets in a table-like structure in the parent widget.

The place() Method − This geometry manager organizes widgets by placing them in a specific position in the parent widget.

2.4 Pillow

Python Pillow module is built on top of PIL (Python Image Library). It is the essential modules for image processing in Python. But it is not supported by Python 3. But, we can use this module with the Python 3.x versions as PIL. It supports the variability of images such as jpeg, png, bmp, gif, ppm, and tiff.

We can do anything on the digital images using the pillow module. In the upcoming section, we will learn various operations on the images such as filtering images, Creating thumbnail, merging images, cropping images, blur an image, resizing an image, creating a water mark and many other operations.

Chapter 3

Project Details

Encryption Algorithm -

A typical AES encryption algorithm runs for 10 rounds -each round comprising of 4 processes.The 1st round is shown below -

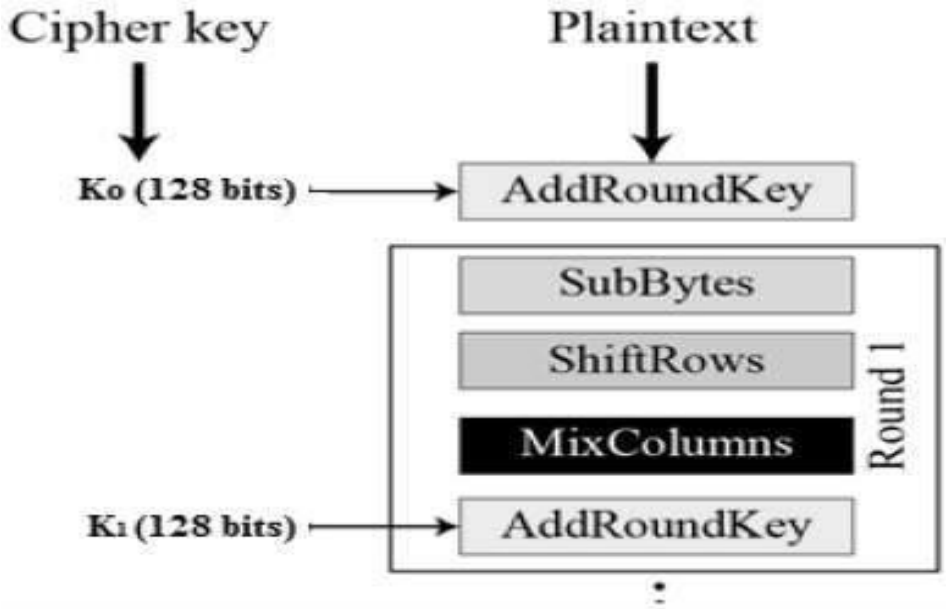


Fig 3.1 AES implementation

Byte Substitution (SubBytes)

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

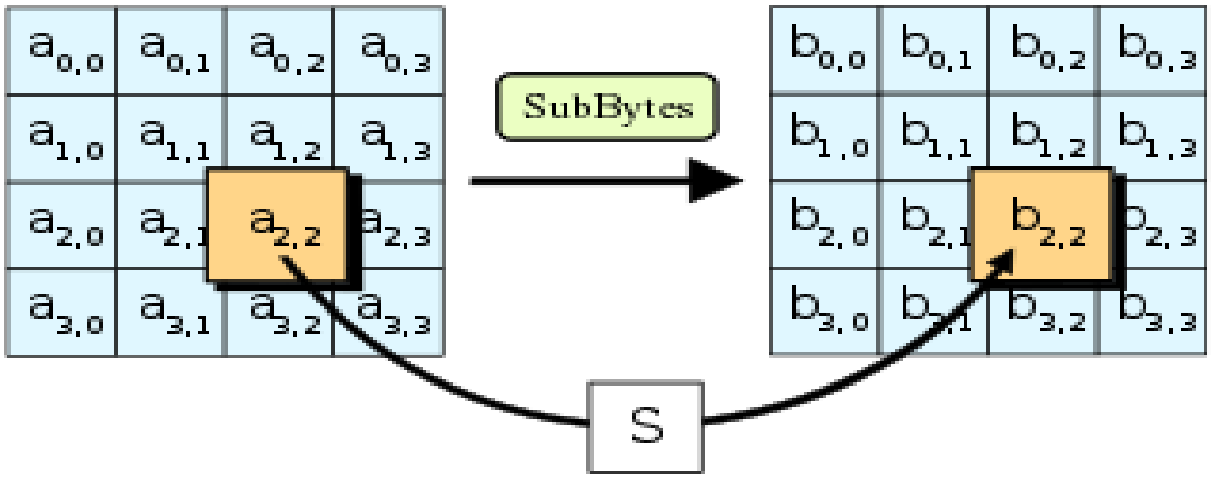


Fig 3.2 Byte Substitution

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 −

- First row is not shifted.
- Second row is shifted one (byte) position to the left.
- Third row is shifted two positions to the left.
- 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.

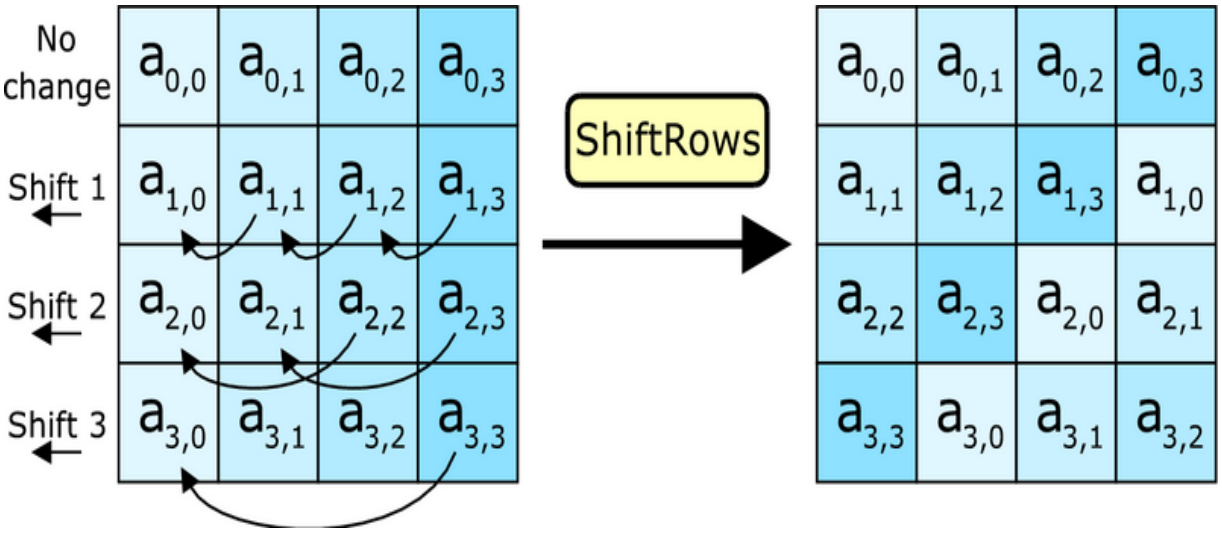


Fig 3.3 Shift Rows

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. We have used the Galois Field Lookup tables for the sake of simplicity.

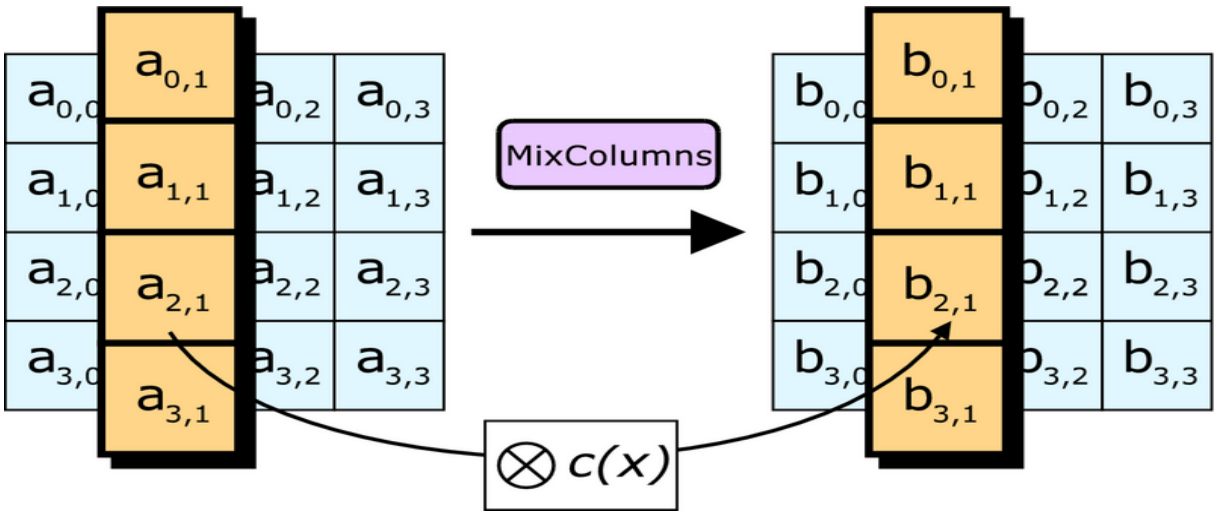


Fig 3.4 Mix Columns

Addroundkey

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.

Decryption Algorithm -

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 −

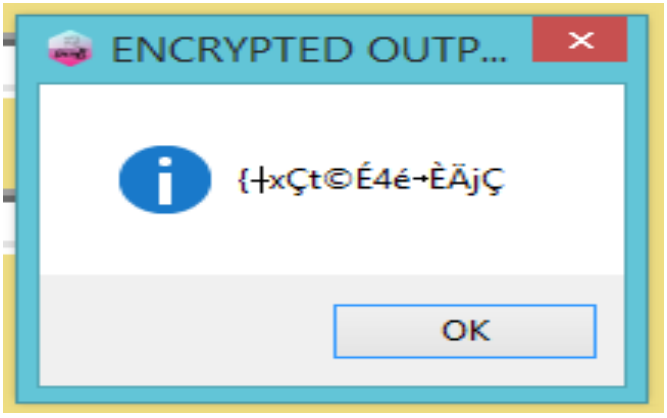
1. Add round key
2. Mix columns
3. Shift rows
4. Byte substitution

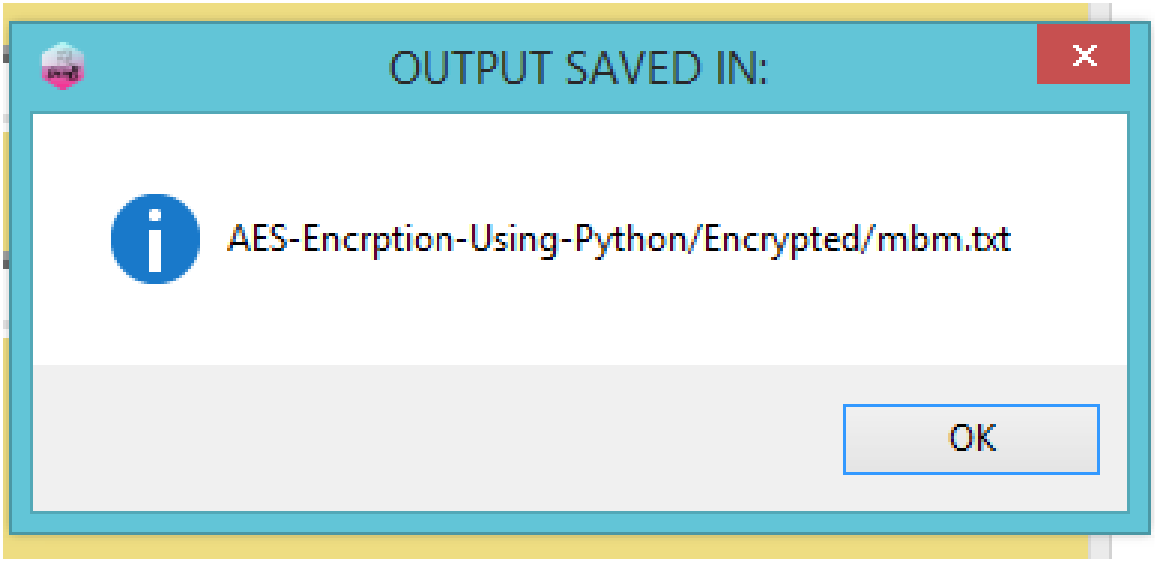
How to use :

Encryption -

1. Run the gui.py script to start the application
2. Select the encrypt tab
3. Enter the text you want to encrpyt. You can also select a txt file that contains the text to be encrypted
4. Enter a 16 character key (make sure to remember this otherwise you wont be able to decrypt later on)
5. Enter the name of the file you wish to be saved without any extensions.
6. Click encrypt. it will be saved as txt fille in the Encrypted folder.


ENCRYPTION PROGRAM
ENCRYPT
DECRYPT
ABOUT
TEXT ENCRYPTION
ADVANCED ENCRYPTION STANDARD
KEY(16 char)
****
TEXT
MBM UNIVERSITY
FILENAME
mbm
Encrypt
OR
KEY(16 char)

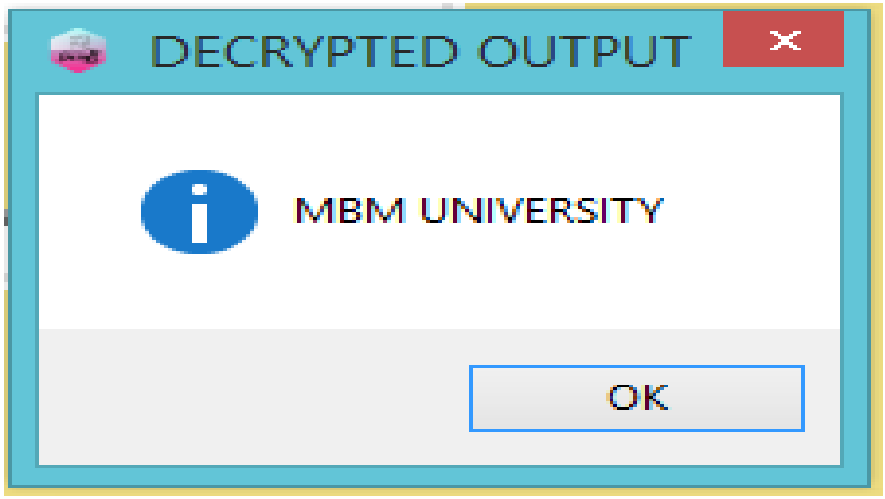

ENCRYPTED OUTP...
{┼xÇt©É4é→ÈÄjÇ
OK

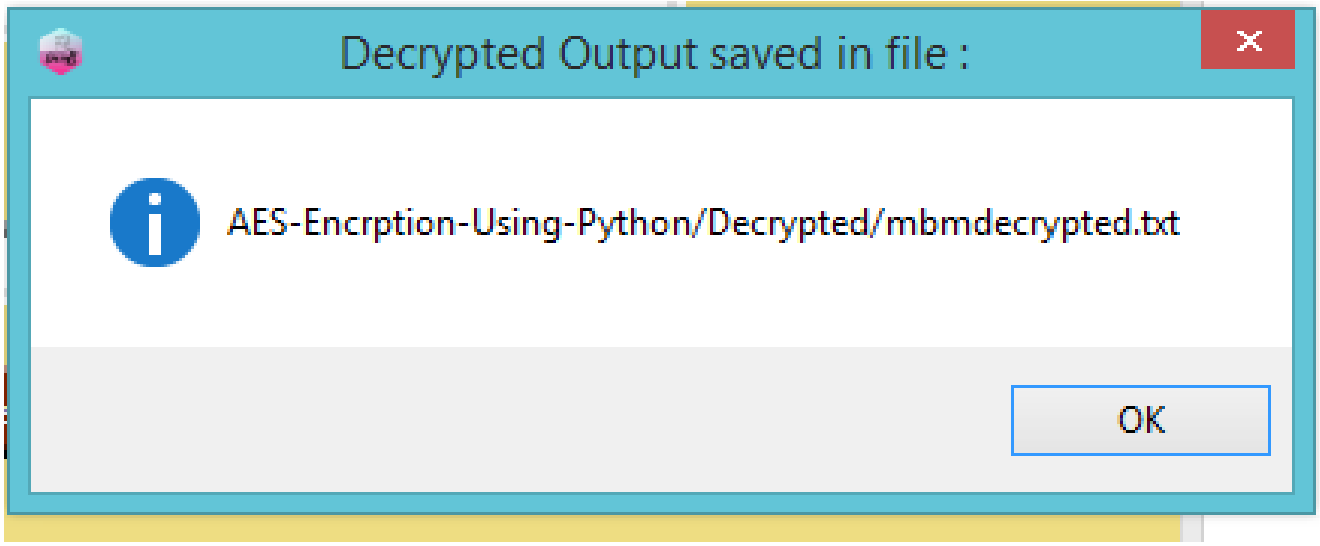


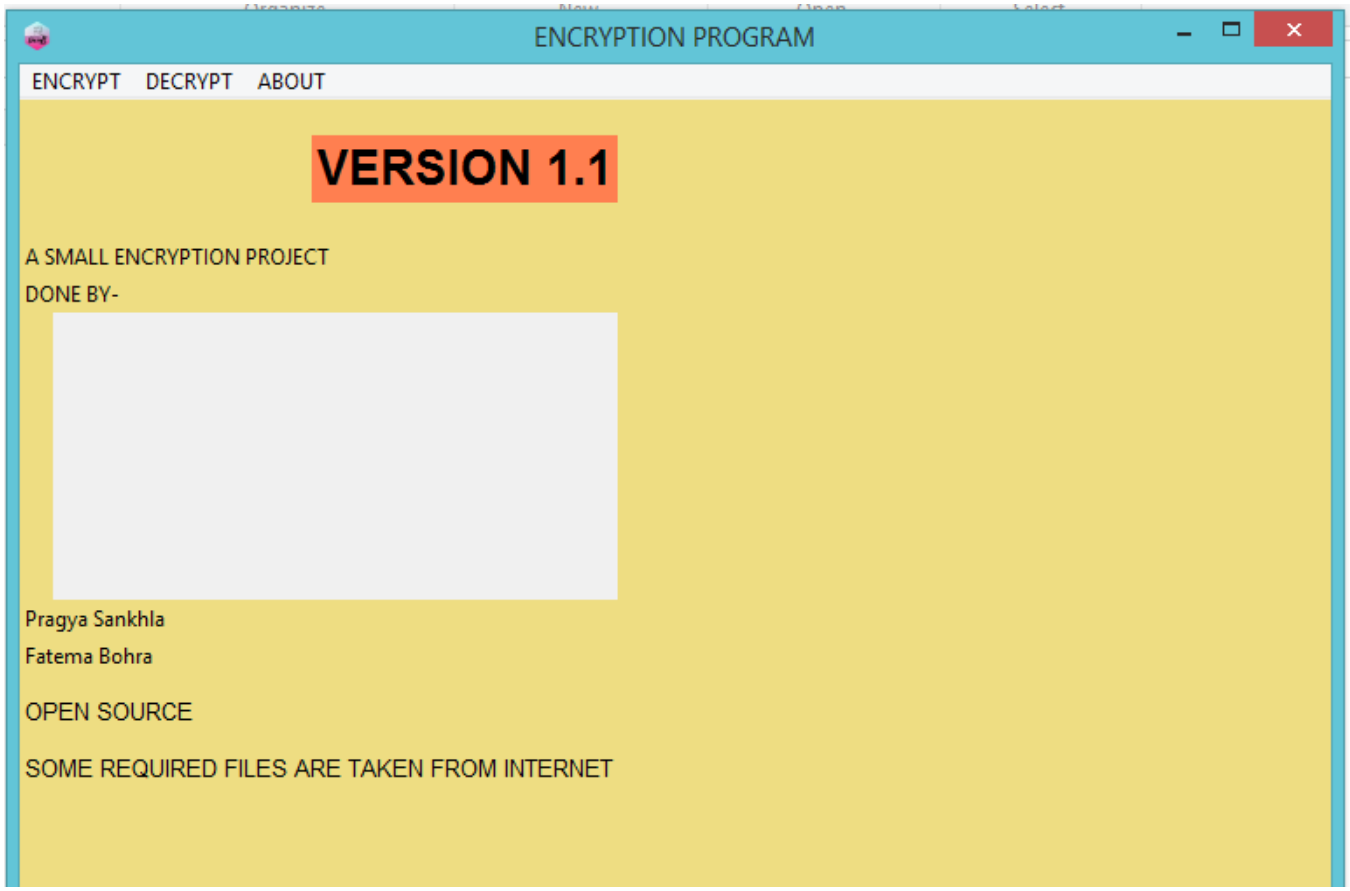
Decryption -

1. Select the decrypt tab
2. Enter the name of the encypted file (no need to worry about the relative path,just enter file name without extension)
3. Enter the 16 character key used for encryption
4. Click decrypt. it will be saved in the Decrypted folder as a txt file.


ENCRYPTION PROGRAM
ENCRYPT
DECRYPT
ABOUT
TEXT DECRYPTION
ADVANCED ENCRYPTION STANDARD
Enter the filename situated in '~/AES-Encrption-Using-Python/Encrypted'
FILENAME
mbm
KEY(16 BIT)
****
Decrypt
EXIT


DECRYPTED OUTPUT
MBM UNIVERSITY
OK


Decrypted Output saved in file :
AES-Encrption-Using-Python/Decrypted/mbmdecrypted.txt
OK


ENCRYPTION PROGRAM
ENCRYPT
DECRYPT
ABOUT
VERSION 1.1
A SMALL ENCRYPTION PROJECT
DONE BY-
Pragya Sankhla
Fatema Bohra
OPEN SOURCE
SOME REQUIRED FILES ARE TAKEN FROM INTERNET

→Security Analysis Of Existing AES

Avalanche Effect

Avalanche effect is a property that is very crucial for block cipher and cryptographic hashfunction. A cryptosystem has avalanche property if for flipping or change just a single bit in plaintext or in secret key the output change significantly (about half of the output).

If a cipher does not show desirable degree of avalanche effect then the cryptanalysts can guess the plaintext by analyzing the Cipertext. So they can be able to break the cipher.

Humming Distance:The Hamming Distance is a digit used to indicate the variation between two binary strings. It is a tiny section of a broader set of formulas used in information analysis.

Avalenche effect = (Hamming distance /Block size)∗ 100%

To compute the avalanche effect of proposed AES-128, we consider two case

Case 1: Here, the plaintext changes by 1 bit in every experiment, the 128 bit key is always constant Key (16 byte): 0123456789ABCDEF

S. No.	No Plaintext (Alphabet)	Ciphertext (Hex.)	Bit variance	Avalanche (%)
1.	ABCDEFGHIJKLMAAB	08E17182BB92E92F3D0786E315F17500	69	53.90
2.	ABCDEFGHIJKLMAAC (changed last character)	082EDF0EB271A6C8821FB019234d8C87		
3.	ABCDEFGHIJKLMAAP	393A42E7681CA8659B48DA2022771CD7	72	56.25

4.	ABCDEFGHIJKLMAAQ (changed last character)	FF59F19913AD598154775653 87D36494		
5.	ABCDEFGHIJKLMABT (changed last character)	8E61F8C978621C3397120176 FEEB1B65	73	57.03
6.	ABCDEFGHIJKLMABU	332B18222296ABEABC2F90A581C7AA59		

Table 3.1 Avalanche Calculate (a)

Case 2: In this case, the plaintext is always constant; the key changes by 1 bit or 2 bit and the Input plaintext (16 bytes): 0123456789ABCDEF

S. No.	Key	Cipher text	Bit variance	Avalanche %
1.	ABCDEFGHIJKLMAAB	5FBB8993C8DB4FAF6FA1BE917638E33A	13	10.15
2.	ABCDEFGHIJKLMAAC (changed only last character)	5FBB8993C8DB4FAF6FA1BE91A20623B2		
3.	ABCDEFGHIJKLMAAR	5FBB8993C8DB4FAF6FA1BE917838E33A	13	10.15
4.	BBCDEFGHIJKLMAAS (changed first and last character)	1FBB8993C8DB4FAF6FA1BE91490623B2		

5.	ABCDEFGHIJKLMAEJ	5FBB8993C8DB4FAF6FA1BE918D44EFC3	13	10.15
6.	ABCDEFKLIJKLMAEJ (changed middle 2 character)	5FBB899378C3C16A6FA1BE918D44EFC3		

Table 3.2 Avalanche Calculate (b)

Above Table shows that, single bit variance in input plaintext shows a small number of bit variance in output. Again previous table also shows that, single bit variance in 128 bit key shows a large number of bit variance in output. After all from both table we can say that, our proposed AES-128 maintains a good degree of Claude Shannon's Confusion and Diffusion property.

Chapter 4

Result/Outcome

This project was successfully completed with the implementation of Encryption and decryption for AES algorithm. We implemented different sub modules for AES algorithm by using Python code. This implementation will be useful in wireless security like military communication and mobile telephony where there is a gayer emphasis on the speed of communication.

Encryption simulation was successfully completed by the use of key expansion and transformations of shift Rows, sub bytes, mix columns, add round keys.

Decryption simulation was successfully completed by the use of key expansion and transformations of inverse shift Rows, inverse sub bytes, inverse mix columns, inverse add round keys.

Chapter 5

Conclusion and Future Work

This project was successfully completed with the implementation of AES algorithm on 128 bit message. The encrypted cipher text and the decrypted text are analyzed and proved to be correct. The encryption efficiency of the proposed AES algorithm was studied and met with satisfactory results. The following can be considered for the future works of this project:

- An extra modification to be used for 192 bit and 256 bit key AES which is an extension of the present paper.
- Power reduction and area minimization for the proposed AES algorithm is to be device.
- LCD can be used for display.

References

[1]International Journal of Scientific & Engineering Research Volume 3, Issue 3 ISSN 2229-5518

[2] A. Lee, NIST Special Publication 800-21, Guideline for Implementing Cryptography in the Federal Government, National Institute of Standards and Technology, November 1999.

[3] J. Daemen and V. Rijmen, The block cipher Rijndael, Smart Card research and Applications, LNCS 1820, SpringerVerlag, pp. 288-296.

[4] J. Nechvatal, et. al., Report on the Development of the Advanced Encryption Standard (AES), National Institute of Standards and Technology, October 2, 2000.

[5] —Specification for the Advanced Encryption Standard (AES),‖ Federal Information Processing Standards Publication 197, Nov. 2001 [

[6]https://www.includehelp.com/cryptography/mode-of-operation.aspx

[7] https://www.geeksforgeeks.org/block-cipher-modes-of-operation/

[8]https://www.techtarget.com/searchsecurity/definition/cryptography