**NATIONAL INSTITUTE OF TECHNOLOGY PUDUCHERRY**



(An Institution of National Importance under MoE, Govt. of India)

**KARAIKAL – 609 609**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**Roll Number:** CS22B1058 **Name: UDHAYAPRASAD S**

**Semester:** 6nd semester **Class:** CSE

**Subject Code:** CS1105  **Subject Name:** NETWORK SECURITY

**Assignment - 1 Date:** 21-04-2025

**DOUBLE ENCRYPTION STANDARD**

**Introduction**  
DES (Data Encryption Standard) is one of the earliest and most well-known symmetric key encryption algorithms. DES is a block cipher, which means that it works on fixed-size blocks of data in this case, 64-bit blocks. It uses a symmetric key for both encryption and decryption and an effective key length of 56 bits the full key is 64 bits, but 8 bits are used for parity checking

**Encryption Process**There are many steps in the overall process of encryption in DES. Initially, the 64-bit plaintext is passed through an initial permutation (IP),  after which 16 rounds of multiple operations—substitution, permutation, expansion, and the XOR of intermediate results with round-specific keys derived from the original key are performed. Each of these rounds modifies the data by splitting it into two halves on the left and right sides, and processing it through a Feistel function. This process is repeated for all 16 rounds, and a last permutation (inverse of IP) is applied, resulting in the 64-bit ciphertext.

**Decryption Process**  
The process of DES decryption is very similar to encryption, with the same 56-bit key used however, the 16 round keys are used in reverse order. The ciphertext is then first permuted, divided into two halves and passed successively through 16 Feistel rounds in which the inverse of the encryption steps is performed. Then, at the end of the rounds, a final permutation is performed in order to return to the original plaintext

**Code Implementation**For implementing Data Encryption Standard **Pycryptodome** Python library is used

**Encryption Function**  
  
def encrypt\_msg(plain\_text,key):

cipher = DES.new(key,DES.MODE\_ECB)

  pad\_msg = pad(plain\_text.encode(),DES.block\_size)

  cipher\_text = cipher.encrypt(pad\_msg)

  return cipher\_text

This **encrypt\_msg** function encrypts the plaintext using DES Algorithm in Electronic Codebook (ECB) Mode.  
  
Initially, this function takes two parameters, such as plaintext and key and a new DES cipher object is created using the key and ECB mode, which encrypts each block of the message independently.

**Padding**  
  
Padding is added to the message block to ensure it’s a multiple of the DES block size. **pad()** function is used for this purpose. Where the plaintext is provided in terms of bytes using the **encode()** function.  
  
**Encryption**   
  
Further, the padded message is encrypted using **encrypt()** function. And cipher text is returned.

**Decryption Function**  
def decrypt\_msg(cipher\_text,key):

  cipher = DES.new(key,DES.MODE\_ECB)

  decrypted\_msg = unpad(cipher.decrypt(cipher\_text),DES.block\_size)

  return decrypted\_msg.decode()

This decrypt\_msg() function decrypts the ciphertext which was encrypted using encrypt\_msg() function.  
  
**Initialization**  
  
A new DES cipher object is created using the same key used for encryption process

**Decryption**   
  
The ciphertext is decrypted using the **decrypt()** function and removes padding using **unpad()** function added during the encryption process.  
  
**Decoding**  
Further the decrypted message is decoded into string from bytes using **decode()** function and then returned

**User Input Message**

plain\_text = input("Enter the Message:")

print(plain\_text)

key = get\_random\_bytes(8)

cipher\_text = encrypt\_msg(plain\_text,key)

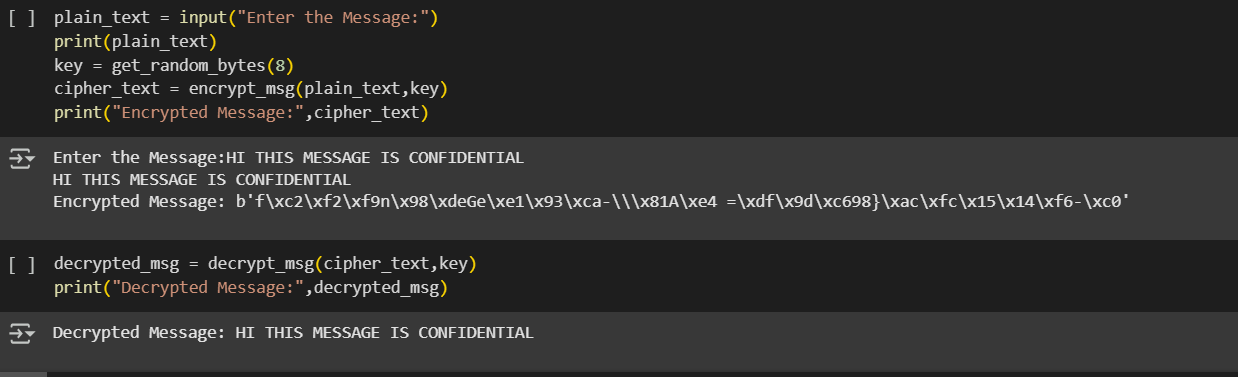
print("Encrypted Message:",cipher\_text)

Initially the Sender message is gathered and stored in plain\_text variable. And display the message provided by the user. a random 8-byte key is generated using get\_random\_bytes() function from Crypto.Random module. This key is used for both encryption and decryption process. Ciphertext is generated by calling the encrypt\_msg() function by providing plaintext and key as a argument.

**Decrypted Message**  
  
decrypted\_msg = decrypt\_msg(cipher\_text,key)

print("Decrypted Message:",decrypted\_msg)

The ciphertext is decrypted using decrypt\_msg() function which takes ciphertext and key as a argument and this decrypted message is stored in decrypted\_msg variable. And displayed the user.

**Output   
  
**

**Conclusion**This example shows a basic implementation of the Data Encryption Standard (DES) algorithm for the encryption and decryption of text messages. The Electronic Codebook (ECB) mode is used for both, using a randomly generated 8-byte key. This example includes encrypting a user-defined message and then decrypt it to get the original message.