**CS1702 - Network Security  
Assignment -1**

Name: Jeff Emerson Mathew Roll Number: CS22B1018 Date: 29/04/2025



**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 has an effective key length of 56 bits; the full key is 64 bits, but 8 bits are used for parity checking.

Encryption:

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:

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:

Installing Libraries:

pip install pycryptodome

Encryption/Decryption Functions and Input/Output:

from Crypto.Cipher import DES

from Crypto.Util.Padding import pad,unpad

from Crypto.Random import get\_random\_bytes

def encrypt\_msg(plain\_text,key):

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

# Padding to ensure plain text is a multiple of block size.

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

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

return cipher\_text

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()

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

print(plain\_text)

key = get\_random\_bytes(8)

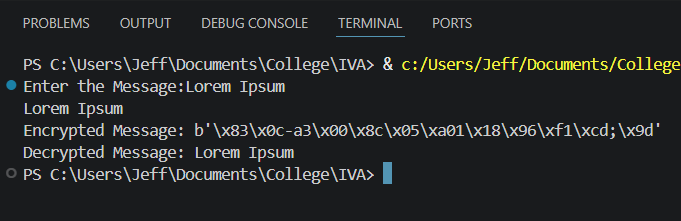
cipher\_text = encrypt\_msg(plain\_text,key)

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

decrypted\_msg = decrypt\_msg(cipher\_text,key)

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

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 decrypting it to get the original message.