|  |  |
| --- | --- |
| **Academic Year: 2024-25** | **Programme: BTECH-Cyber (CSE)** |
| **Year: 2nd** | **Semester: IV** |
| **Student Name: Jal Bafana** | **Batch : K1** |
| **Roll No: K005** | **Date of experiment: 20.02.2025** |
| **Faculty: Rejo Mathew** | **Signature with Date:** |

**Experiment 7: Data Encryption Standard (DES)**

**Aim:** Write a program to implement DES algorithm.

**Learning Outcomes:**

After completion of this experiment, student should be able to

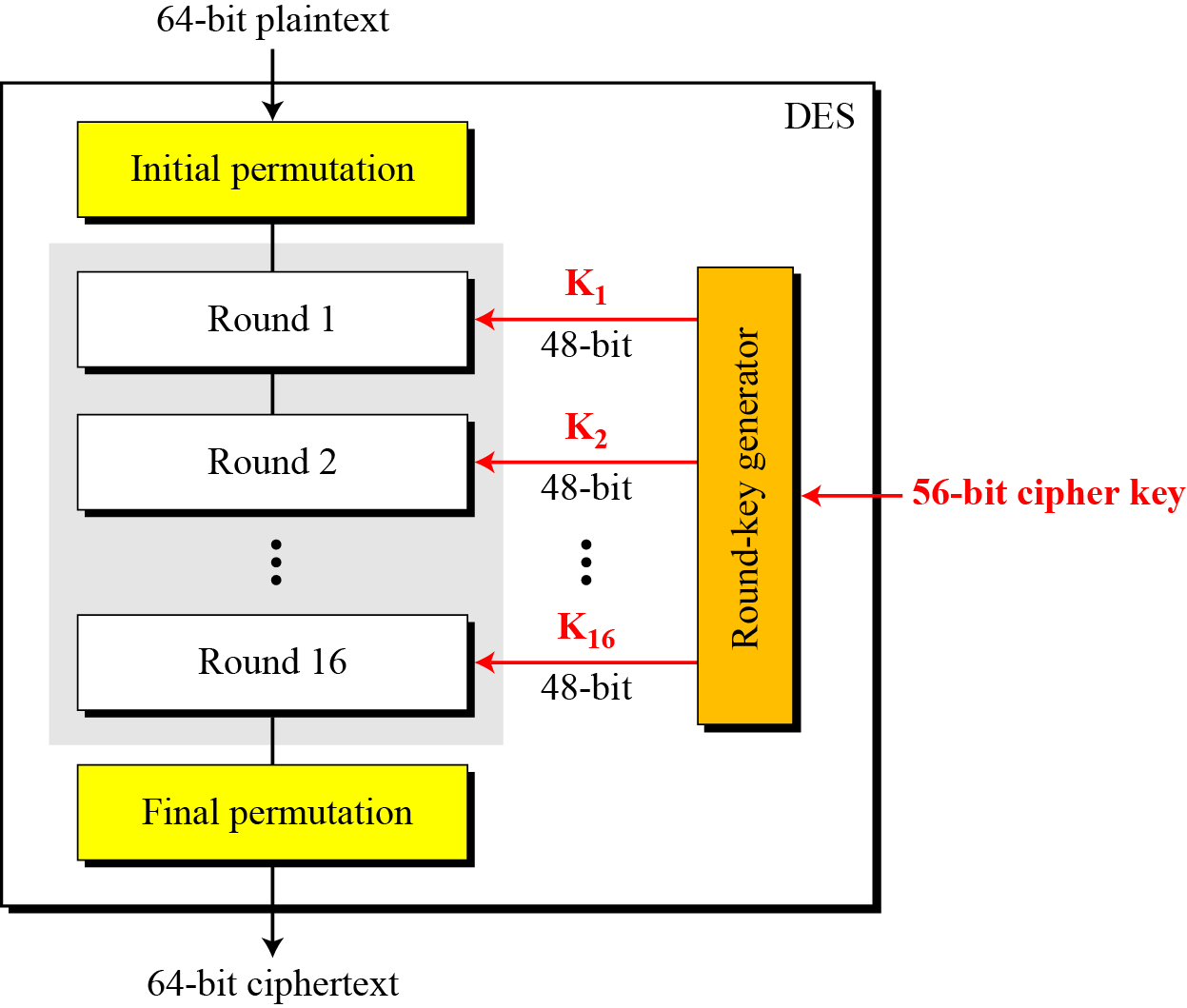
1. Describe working of DES algorithm.
2. Understand application of DES along with its advantage and limitations.

**Theory:**

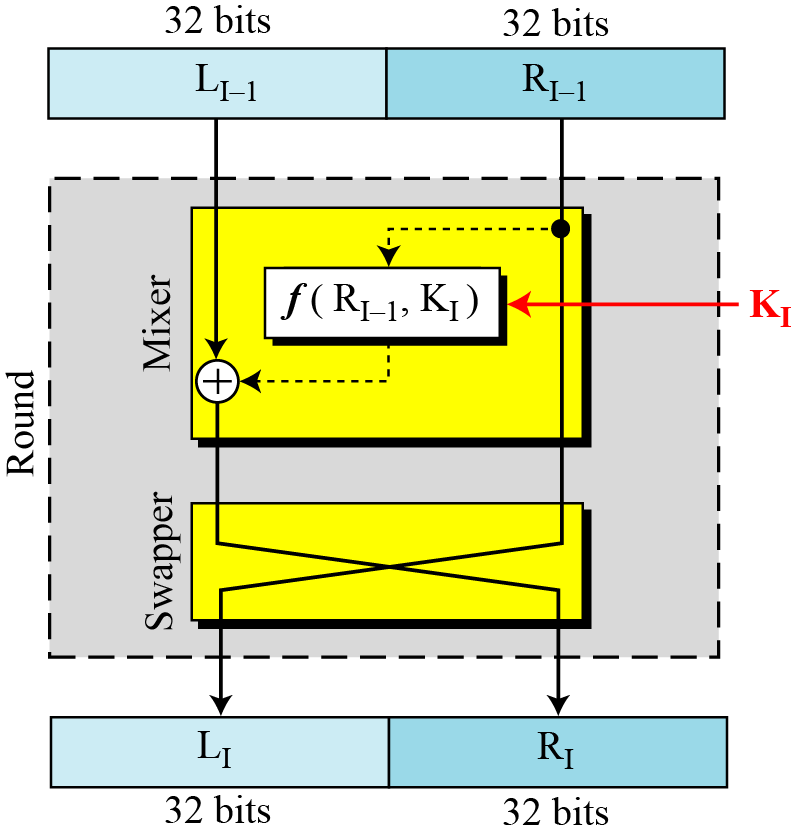
DES (Data Encryption Standard) algorithm was the most widely used encryption algorithm.

It is a block cipher that encrypts blocks of 64 bits using a 64-bit key (8 bits are discarded so we have 56 bit key) and outputs 64 bits of cipher text. It is a product cipher. It performs both substitution (also called confusion) and transposition (also called as diffusion) on the bits. Cipher consists of 16 rounds (iterations) each with a round key generated from the user-supplied key.

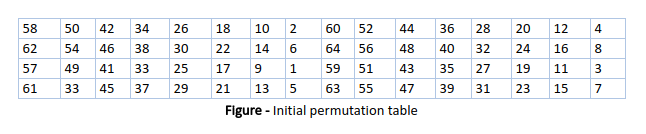
In the first step, the 64-bit plain text block is handed over to an initial Permutation (IP) function.The initial permutation is performed on plain text.



Next, the initial permutation (IP) produces two halves of the permuted block; saying Left Block (LB) and Right Block (RB).Now each LB and RB go through 16 rounds of the encryption process.In the end, LB and RB are rejoined and a Final Permutation (FP) is performed on the combined block. The result of this process produces 64-bit ciphertext.



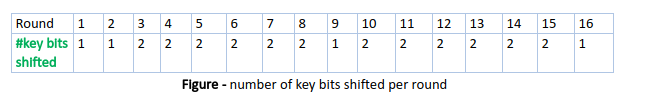
As we have noted, the initial permutation (IP) happens only once and it happens before the first round. It suggests how the transposition in IP should proceed, as shown in the figure. For example, it says that the IP replaces the first bit of the original plain text block with the 58th bit of the original plain text, the second bit with the 50th bit of the original plain text block, and so on. This is nothing but jugglery of bit positions of the original plain text block. the same rule applies to all the other bit positions shown in the figure.



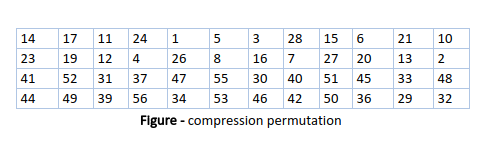
**Step 1: Key transformation**

We have noted initial 64-bit key is transformed into a 56-bit key by discarding every 8th bit of the initial key. Thus, for each a 56-bit key is available. From this 56-bit key, a different 48-bit Sub Key is generated during each round using a process called key transformation. For this, the 56-bit key is divided into two halves, each of 28 bits. These halves are circularly shifted left by one or two positions, depending on the round.

For example: if the round numbers 1, 2, 9, or 16 the shift is done by only one position for other rounds, the circular shift is done by two positions. The number of key bits shifted per round is shown in the figure.



After an appropriate shift, 48 of the 56 bits are selected. From the 48 we might obtain 64 or 56 bits based on requirement which helps us to recognize that this model is very versatile and can handle any range of requirements needed or provided. for selecting 48 of the 56 bits the table is shown in the figure given below. For instance, after the shift, bit number 14 moves to the first position, bit number 17 moves to the second position, and so on. If we observe the table, we will realize that it contains only 48-bit positions. Bit number 18 is discarded (we will not find it in the table), like 7 others, to reduce a 56-bit key to a 48-bit key. Since the key transformation process involves permutation as well as a selection of a 48-bit subset of the original 56-bit key it is called Compression Permutation.Because of this compression permutation technique, a different subset of key bits is used in each round. That makes DES not easy to crack.



**Step 2: Expansion Permutation**

During the expansion permutation, the Right block is expanded from 32 bits to 48 bits. Bits are permuted as well hence called expansion permutation. This happens as the 32-bit Right block is divided into 8 blocks, with each block consisting of 4 bits. Then, each 4-bit block of the previous step is then expanded to a corresponding 6-bit block, i.e., per 4-bit block, 2 more bits are added.

This process results in expansion as well as a permutation of the input bit while creating output. The key transformation process compresses the 56-bit key to 48 bits. Then the expansion permutation process expands the 32-bit Right block to 48-bits. Now the 48-bit key is XOR with 48-bit Right block and the resulting output is given to the next step, which is the S-Box substitution.

**Code: *type or copy your completed working code here***

# Python program to demonstrate the working of DES algorithm

# Import necessary libraries

from Crypto.Cipher import DES

from Crypto.Util.Padding import pad, unpad

from Crypto.Random import get\_random\_bytes

# DES encryption and decryption

def des\_encrypt(plain\_text, key):

cipher = DES.new(key, DES.MODE\_CBC) # CBC mode used

encrypted\_text = cipher.encrypt(pad(plain\_text.encode(), DES.block\_size))

return cipher.iv + encrypted\_text # Return IV with encrypted text

def des\_decrypt(encrypted\_text, key):

iv = encrypted\_text[:DES.block\_size] # Extract the IV

cipher\_text = encrypted\_text[DES.block\_size:] # Remaining is the cipher text

cipher = DES.new(key, DES.MODE\_CBC, iv)

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

return decrypted\_text.decode()

# Example usage

def main():

key = get\_random\_bytes(8) # 64-bit key (8 bytes)

plain\_text = "This is a secret message"

print("Original Message:", plain\_text)

# Encrypt the message

encrypted = des\_encrypt(plain\_text, key)

print("Encrypted Message (in hex):", encrypted.hex())

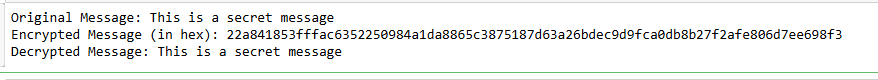
# Decrypt the message

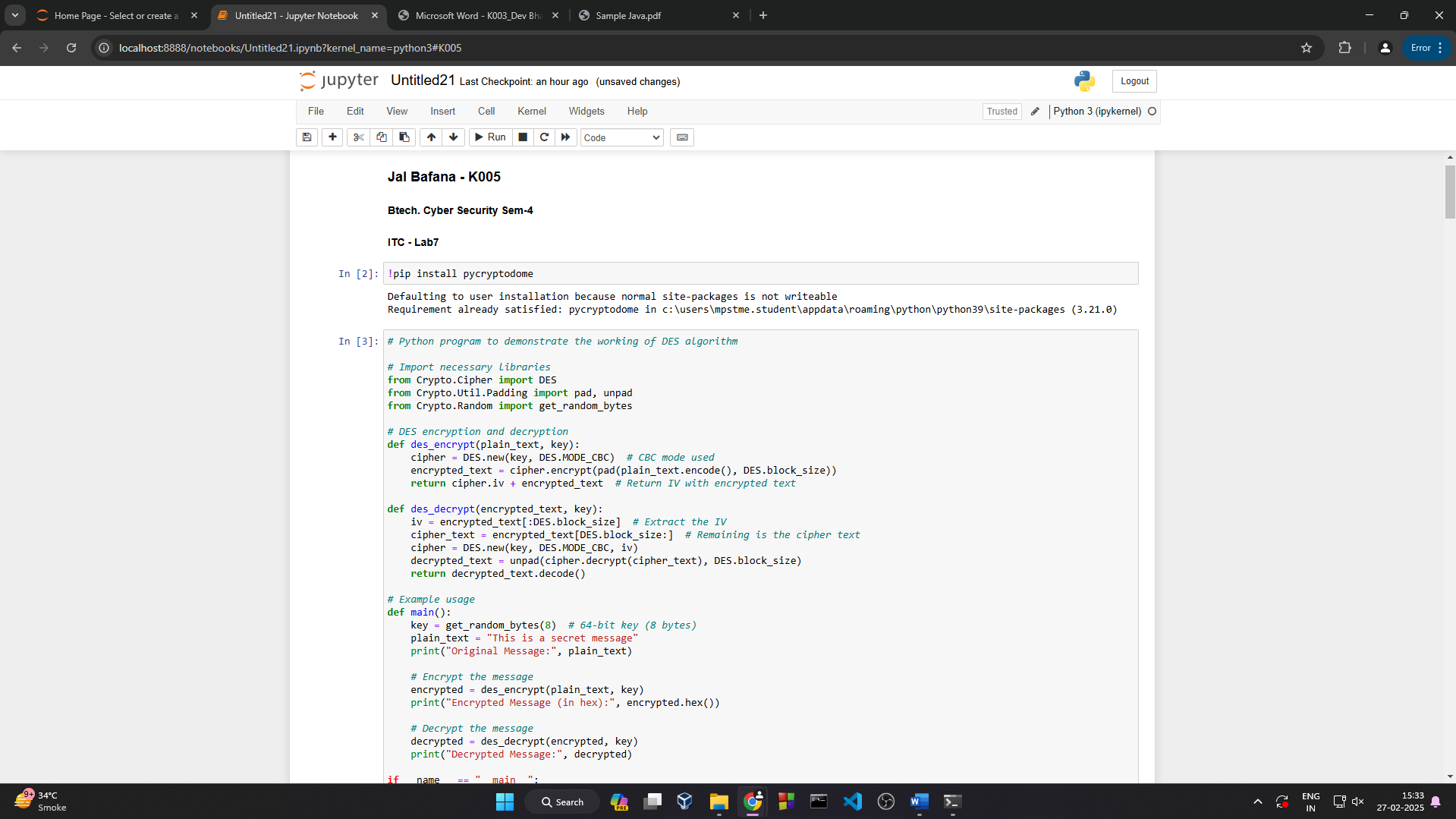
decrypted = des\_decrypt(encrypted, key)

print("Decrypted Message:", decrypted)

if \_\_name\_\_ == "\_\_main\_\_":

main()





**Questions:**

1. What is the significance of the **Initial Permutation (IP) and Final Permutation (FP)** in DES?

* The **Initial Permutation (IP)** in the Data Encryption Standard (DES) is a fixed bit-level permutation applied to the plaintext before the Feistel rounds begin. It does not add cryptographic strength but helps in hardware implementations.
* The **Final Permutation (FP)** is the inverse of IP and is applied at the end of encryption before producing the ciphertext. It essentially undoes the IP, ensuring the encryption process remains reversible.
* Both IP and FP do not contribute to security but were likely introduced to simplify hardware implementations at the time of DES's development.

2. How does **key whitening** enhance the security of DES-based algorithms?

* Key whitening is a technique used in modern encryption schemes (e.g., in Triple DES or DESX) to improve security.
* It involves XORing additional subkeys with the plaintext before the first DES round and/or with the ciphertext after the last round.
* This reduces vulnerabilities to brute force attacks and makes cryptanalysis more difficult by increasing the effective key size, making the encryption stronger against known attacks.

3. How does DES handle encryption of messages longer than **64 bits**?

DES operates on fixed-size blocks of 64 bits. To encrypt longer messages, a **block cipher mode of operation** is used, such as:

* **Electronic Codebook (ECB):** Encrypts each 64-bit block independently, making it vulnerable to pattern repetition.
* **Cipher Block Chaining (CBC):** Uses an Initialization Vector (IV) and XORs each plaintext block with the previous ciphertext block before encryption, improving security.
* **Counter (CTR) or Output Feedback (OFB) modes:** Convert DES into a stream cipher, allowing flexible encryption.

4. What are the advantages of using **CBC (Cipher Block Chaining) mode** over ECB in DES?

* **Prevents Pattern Leakage:** In ECB mode, identical plaintext blocks produce identical ciphertext blocks, making it insecure for structured data. CBC ensures that even identical plaintext blocks yield different ciphertexts.
* **Enhanced Security with Chaining:** Each plaintext block is XORed with the previous ciphertext before encryption, making it resistant to replay attacks and reducing predictability.
* **Randomization with IV:** CBC uses an Initialization Vector (IV) for the first block, adding randomness and further improving security.
* **Better for Large Messages:** Unlike ECB, CBC avoids direct pattern repetition, making it more suitable for encrypting long messages securely.

**Conclusion:** *The DES algorithm was successfully implemented, demonstrating its encryption and decryption process. This experiment provided insights into DES’s working, its strengths, and its limitations in modern cryptography.*