**Final Year B.Tech. (CSE) – VII [ 2024-25]**

**6CS451: Cryptography and Network Security Lab (C&NS Lab)**

**Date: 26/08/2024**

**Assignment 5**

**PRN:** 21510017  **Name:** Onkar Anand Yemul

1. **Apply DES algorithm for practical applications**

**Ans:**The Data Encryption Standard (DES) is a symmetric-key algorithm for the encryption of digital data. Although DES is now considered insecure for many applications due to its small key size, it is still an important algorithm for understanding the basics of cryptography.

**Practical Application of DES Algorithm**

To apply the DES algorithm in a practical application, we can use the **pycryptodome** library in Python, which provides an implementation of DES. Below is an example that demonstrates how to use DES to encrypt and decrypt a message.

**Python Code:**

from Crypto.Cipher import DES

from Crypto.Util.Padding import pad, unpad

from Crypto.Random import get\_random\_bytes

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

    """

    Encrypt the plain text using DES algorithm.

    Parameters:

    plain\_text (str): The text to be encrypted.

    key (bytes): The encryption key (must be 8 bytes long).

    Returns:

    bytes: The encrypted cipher text.

    """

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

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

    encrypted\_text = cipher.encrypt(padded\_text)

    return encrypted\_text

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

    """

    Decrypt the cipher text using DES algorithm.

    Parameters:

    cipher\_text (bytes): The encrypted text to be decrypted.

    key (bytes): The decryption key (must be 8 bytes long).

    Returns:

    str: The decrypted plain text.

    """

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

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

    return decrypted\_text.decode()

def main():

    """

    The main function to run the program.

    """

    print("\nDES Encryption and Decryption")

    # Generate a random 8-byte key for DES

    key = get\_random\_bytes(8)

    print(f"\nGenerated Key (in hexadecimal): {key.hex()}")

    # Input plaintext

    plain\_text = input("Enter the plain text to encrypt: ")

    # Encrypt the plaintext

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

    print(f"\nEncrypted Text (in hexadecimal): {encrypted\_text.hex()}")

    # Decrypt the ciphertext

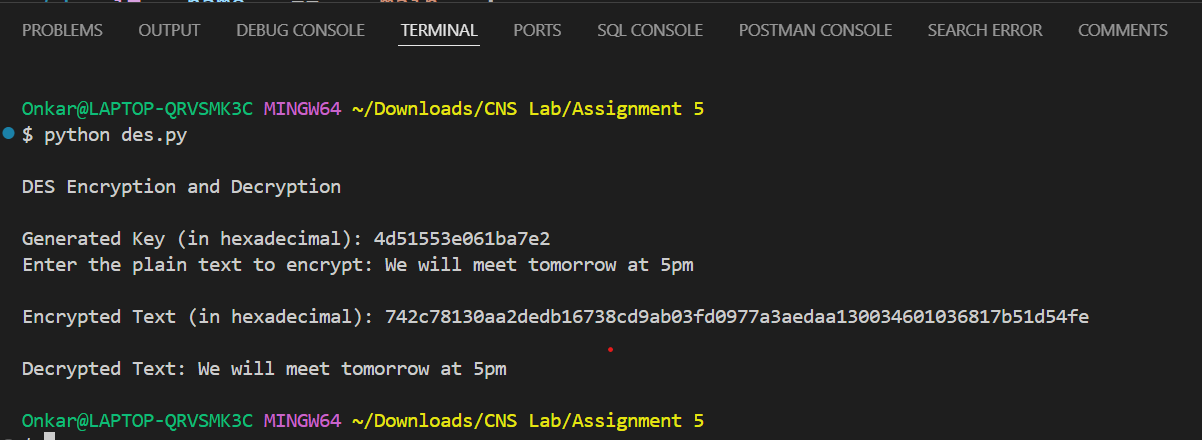
    decrypted\_text = des\_decrypt(encrypted\_text, key)

    print(f"\nDecrypted Text: {decrypted\_text}")

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

    main()

**Output:**

****

**Practical Applications:**

* **File Encryption:** DES can be used to encrypt sensitive files before storing them in insecure locations.
* **Secure Communication:** DES ensures that messages sent over a network are unreadable to unauthorized parties.
* **Password Storage:** Encrypting passwords before storing them in databases (though modern standards recommend stronger algorithms like AES).

While DES itself is outdated and not recommended for secure applications, understanding how it works is crucial for grasping more advanced encryption algorithms like AES.