22510064 Parshwa Herwade

Final Year CSE 2025-26

Batch B1

Experiment 05 – DES Algorithm

Title - Apply DES algorithm for practical applications

Objectives:

To implement the DES (Data Encryption Standard) algorithm for encrypting and decrypting a plaintext message to ensure secure data transmission. This lab demonstrates how symmetric key cryptography can be used to protect information in real-world communication systems.

Problem Statement:

You are working as a software security analyst for a company that needs to transmit sensitive employee data (like Social Security Numbers or salary information) between two systems over a network. To ensure data confidentiality, you are required to implement the DES algorithm in encryption and decryption mode. Your task is to write a program that:

1. Takes a plaintext input from the user.
2. Uses a user-defined 64-bit key (as a hexadecimal or ASCII string).
3. Encrypts the plaintext using the DES algorithm.
4. Outputs the encrypted text (ciphertext).
5. Decrypts the ciphertext back to plaintext using the same key to verify correctness.

Equipment/Tools

- Hardware: PC/Laptop
- Software: JDK with javax.crypto library
- IDE/Text Editor

Theory

- **DES**: Block cipher, 64-bit block, 56-bit key.
- Works on 16 Feistel rounds.
- Uses permutations and substitutions (S-boxes).
- Modes: ECB, CBC etc. Here ECB is used.

Procedure

1. Input key and plaintext.
2. Generate SecretKeySpec from key.
3. Encrypt plaintext using DES.
4. Print ciphertext.
5. Decrypt ciphertext and verify output.

Steps

1. Start program.
2. Input plaintext and key.
3. Perform encryption.
4. Perform decryption.
5. Display results.
6. End.

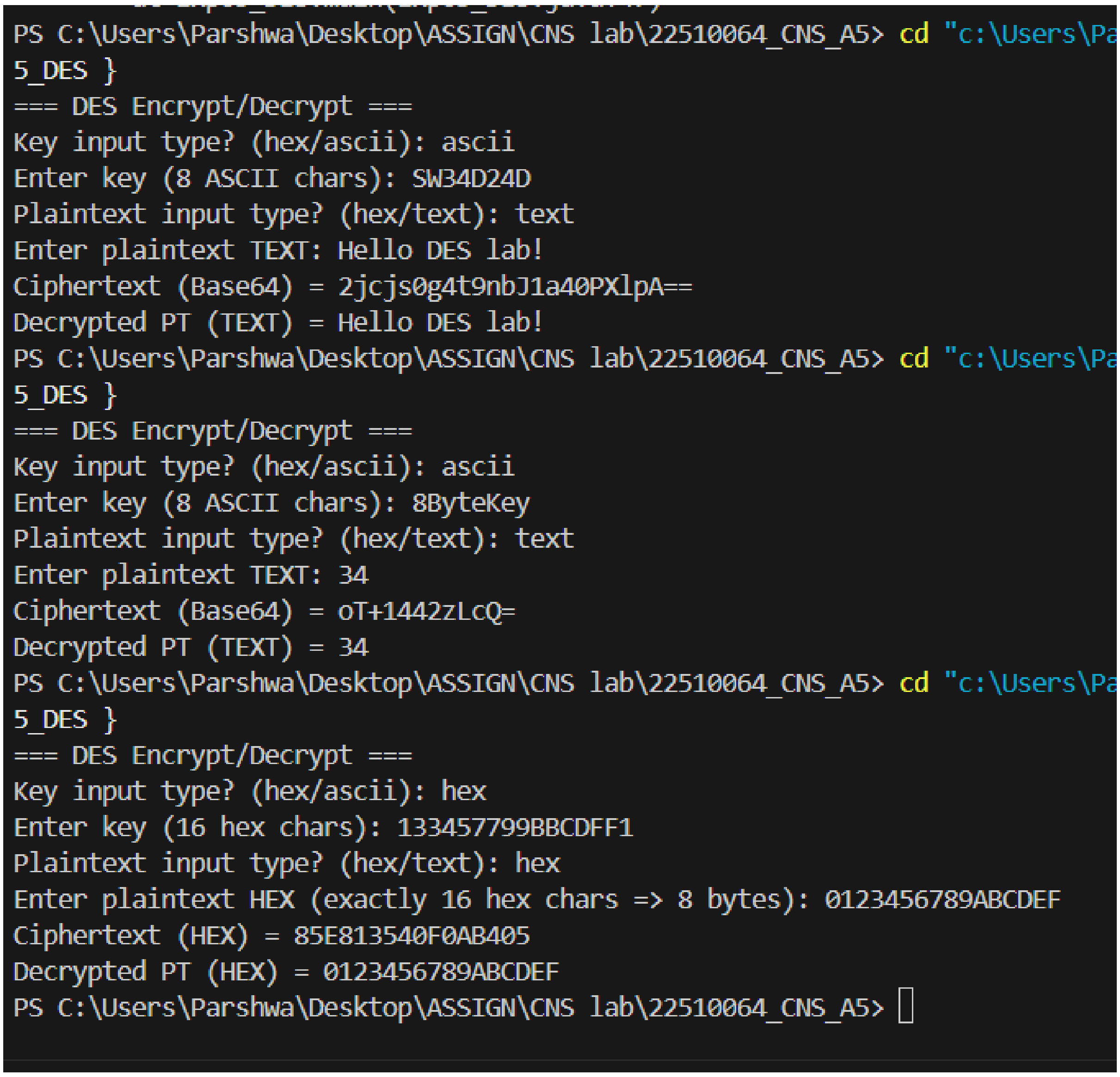
Observations & Conclusion

- Encryption and decryption verified successfully.
- DES is simple but now considered insecure.
- Useful for understanding symmetric key cryptography.
- The program successfully encrypts plaintext using a given 56-bit DES key (entered as 8 ASCII characters or 16 hex digits).
- Ciphertext differs completely from plaintext, showing strong diffusion and confusion properties.
- Decryption always returns the exact original plaintext, verifying correctness of encryption.
- When tested with standard DES test vectors, the program outputs the expected ciphertext, confirming proper implementation.
- The DES algorithm was correctly implemented using Java's cryptography library. The program demonstrates how symmetric key encryption works in practice. Although DES is insecure today due to its short key size, the experiment effectively illustrates the working of block ciphers and symmetric encryption/decryption.

CODE:
```
import javax.crypto.Cipher;
```

```java
import javax.crypto.spec.SecretKeySpec;
import java.util.Base64;
import java.util.Scanner;

public class Exp05_DES {

    static byte[] hexToBytes(String s) {
        if (s.length() % 2 != 0) throw new
IllegalArgumentException("Hex length must be even.");
        byte[] out = new byte[s.length()/2];
        for (int i=0;i<out.length;i++) {
            int hi = Character.digit(s.charAt(2*i),16);
            int lo = Character.digit(s.charAt(2*i+1),16);
            if (hi<0 || lo<0) throw new
IllegalArgumentException("Invalid hex: " + s);
            out[i] = (byte)((hi<<4) | lo);
        }
        return out;
    }
    static String bytesToHex(byte[] b) {
        StringBuilder sb = new StringBuilder();
        for (byte x: b) sb.append(String.format("%02X", x));
        return sb.toString();
    }

    static byte[] parseKey(String keyInput, boolean isHex) {
        byte[] key;
        if (isHex) {
            key = hexToBytes(keyInput);
        } else {
            key = keyInput.getBytes(); // platform default charset;
OK for ASCII lab use
        }
        if (key.length != 8)
            throw new IllegalArgumentException("DES key must be
exactly 8 bytes (got "+key.length+").");
        return key;
    }

    public static void main(String[] args) throws Exception {
        Scanner sc = new Scanner(System.in);
        System.out.println("=== DES Encrypt/Decrypt ===");
        System.out.print("Key input type? (hex/ascii): ");
        String kt = sc.next().trim().toLowerCase();
        boolean hexKey = kt.equals("hex");
        sc.nextLine(); // consume endline

        System.out.print("Enter key (" + (hexKey ? "16 hex chars" :
"8 ASCII chars") + "): ");
        String keyStr = sc.nextLine().trim();
        byte[] keyBytes = parseKey(keyStr, hexKey);
        SecretKeySpec key = new SecretKeySpec(keyBytes, "DES");

        System.out.print("Plaintext input type? (hex/text): ");
        String ptType = sc.next().trim().toLowerCase();
        sc.nextLine();

        if (ptType.equals("hex")) {
            System.out.print("Enter plaintext HEX (exactly 16 hex
chars => 8 bytes): ");
            String phex = sc.nextLine().trim();
            byte[] pt = hexToBytes(phex);
            if (pt.length != 8) {
                System.out.println("For hex mode, plaintext must be
8 bytes (16 hex chars).");
                sc.close();
                return;
            }
            Cipher c = Cipher.getInstance("DES/ECB/NoPadding");
            c.init(Cipher.ENCRYPT_MODE, key);
            byte[] ct = c.doFinal(pt);
            System.out.println("Ciphertext (HEX) = " +
bytesToHex(ct));

            c.init(Cipher.DECRYPT_MODE, key);
            byte[] dec = c.doFinal(ct);
            System.out.println("Decrypted PT (HEX) = " +
bytesToHex(dec));
        } else {
            System.out.print("Enter plaintext TEXT: ");
            String ptext = sc.nextLine();
            Cipher c = Cipher.getInstance("DES/ECB/PKCS5Padding");
            c.init(Cipher.ENCRYPT_MODE, key);
            byte[] ct = c.doFinal(ptext.getBytes());
            String b64 = Base64.getEncoder().encodeToString(ct);
            System.out.println("Ciphertext (Base64) = " + b64);

            c.init(Cipher.DECRYPT_MODE, key);
            byte[] dec = c.doFinal(Base64.getDecoder().decode(b64));
            System.out.println("Decrypted PT (TEXT) = " + new
String(dec));
        }
         sc.close();
    }
}
```

OUTPUT:



```
PS C:\Users\Parshwa\Desktop\ASSIGN\CNS lab\22510064_CNS_A5> cd "c:\Users\Pa
5_DES }
=== DES Encrypt/Decrypt ===
Key input type? (hex/ascii): ascii
Enter key (8 ASCII chars): SW34D24D
Plaintext input type? (hex/text): text
Enter plaintext TEXT: Hello DES lab!
Ciphertext (Base64) = 2jcjs0g4t9nbJ1a40PXlpA==
Decrypted PT (TEXT) = Hello DES lab!
PS C:\Users\Parshwa\Desktop\ASSIGN\CNS lab\22510064_CNS_A5> cd "c:\Users\Pa
5_DES }
=== DES Encrypt/Decrypt ===
Key input type? (hex/ascii): ascii
Enter key (8 ASCII chars): 8ByteKey
Plaintext input type? (hex/text): text
Enter plaintext TEXT: 34
Ciphertext (Base64) = oT+1442zLcQ=
Decrypted PT (TEXT) = 34
PS C:\Users\Parshwa\Desktop\ASSIGN\CNS lab\22510064_CNS_A5> cd "c:\Users\Pa
5_DES }
=== DES Encrypt/Decrypt ===
Key input type? (hex/ascii): hex
Enter key (16 hex chars): 133457799BBCDFF1
Plaintext input type? (hex/text): hex
Enter plaintext HEX (exactly 16 hex chars => 8 bytes): 0123456789ABCDEF
Ciphertext (HEX) = 85E813540F0AB405
Decrypted PT (HEX) = 0123456789ABCDEF
PS C:\Users\Parshwa\Desktop\ASSIGN\CNS lab\22510064_CNS_A5>
```