



## **LAB ASSIGNMENT: 02**

**SUBMITTED BY:**

**Aliza Faisal**

**SP24-BSE-048**

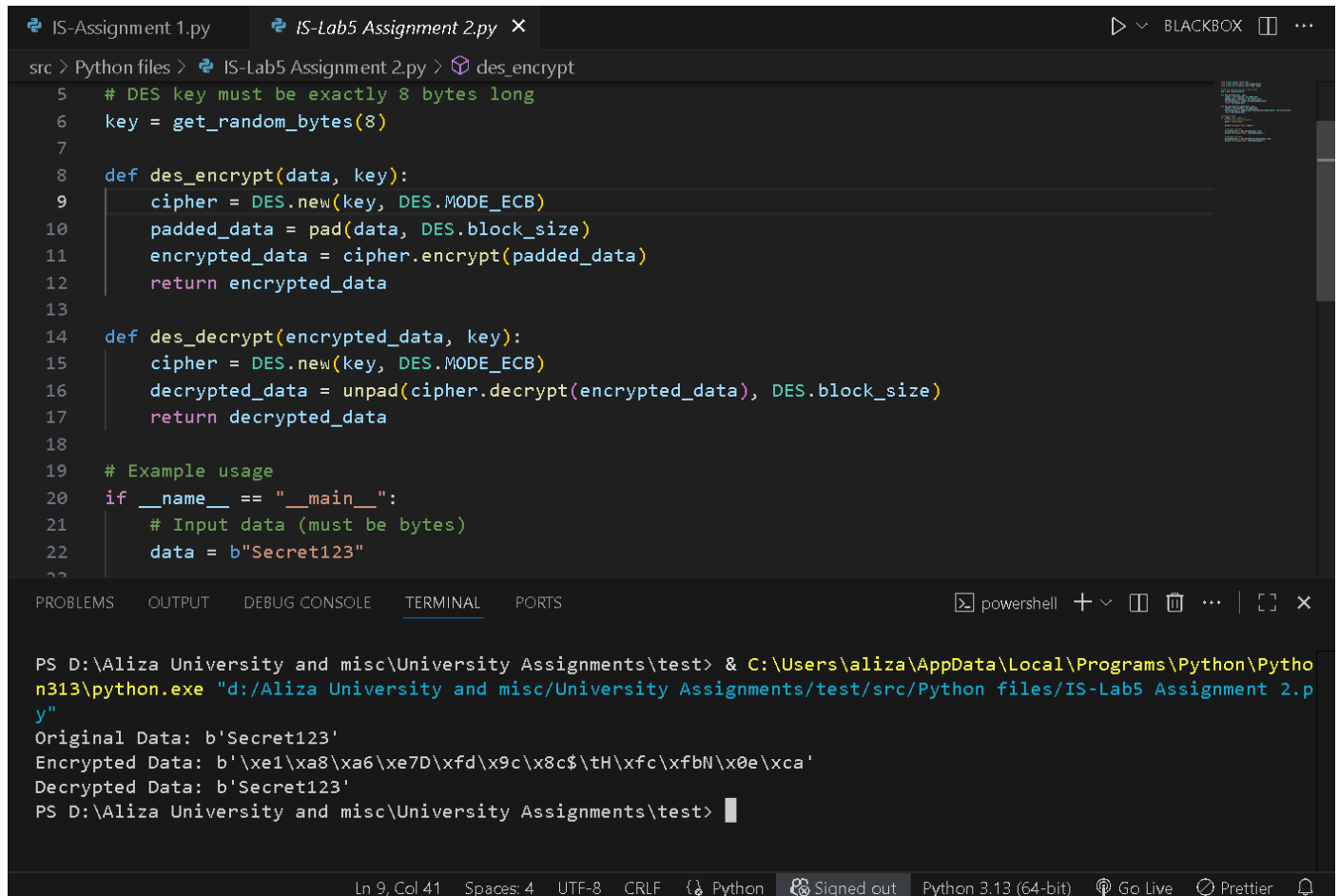
**SUBMITTED TO: MA'AM AMBREEN  
GUL**

**DATE: 12<sup>TH</sup> OCTOBER, 2025**

**COURSE: Information Security**

# DES Block Cipher Algorithm

*Graded Lab Task 1:* You have implemented DES there is built in implemented DES in python in crypto cipher module use it for encryption/decryption and provide output sample example



```
IS-Assignment 1.py  IS-Lab5 Assignment 2.py X
src > Python files > IS-Lab5 Assignment 2.py > des_encrypt
5  # DES key must be exactly 8 bytes long
6  key = get_random_bytes(8)
7
8  def des_encrypt(data, key):
9      cipher = DES.new(key, DES.MODE_ECB)
10     padded_data = pad(data, DES.block_size)
11     encrypted_data = cipher.encrypt(padded_data)
12     return encrypted_data
13
14 def des_decrypt(encrypted_data, key):
15     cipher = DES.new(key, DES.MODE_ECB)
16     decrypted_data = unpad(cipher.decrypt(encrypted_data), DES.block_size)
17     return decrypted_data
18
19 # Example usage
20 if __name__ == "__main__":
21     # Input data (must be bytes)
22     data = b"Secret123"
23
24
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
PS D:\Aliza University and misc\University Assignments\test> & C:\Users\aliza\AppData\Local\Programs\Python\Python313\python.exe "d:/Aliza University and misc/University Assignments/test/src/Python files/IS-Lab5 Assignment 2.py"
Original Data: b'Secret123'
Encrypted Data: b'\xe1\xa8\xa6\xe7D\xfd\x9c\x8c$\tH\xfc\xfbN\x0e\xca'
Decrypted Data: b'Secret123'
PS D:\Aliza University and misc\University Assignments\test>
Ln 9, Col 41  Spaces: 4  UTF-8  CRLF  Python  Signed out  Python 3.13 (64-bit)  Go Live  Prettier
```

## What This Code Does:

- This code is a **secret message machine** that uses DES encryption. It takes a normal message, turns it into secret code using a password, and then turns it back to the original message.

## How It Works:

### Step 1 - Make a Password:

- Creates a random 8-character password (this is the DES key)

### Step 2 - Hide the Message (Encryption):

- Takes your message and the password
- Adds some extra characters to make it the right size (called padding)

- Scrambles it into secret code that looks like random numbers

### **Step 3 - Get Back Message (Decryption):**

- Takes the secret code and the same password
- Unscrambles the secret code
- Removes the extra characters
- Returns your original message

### *Graded Lab Task 2:*

#### **1. Simple Step-by-Step Explanation**

##### **Step 1: The Attacker's Starting Point**

- **Attacker has:** A known message + its encrypted version
- **Example:**
  - Plaintext: "HELLO"
  - Ciphertext: "XJ9#L"

##### **Step 2: Split the Problem**

- DES uses 56-bit key (HUGE number: 72,057,594,037,927,936)
- Instead of trying all keys, split into two parts:
  - **K1** = First 28 bits
  - **K2** = Last 28 bits

##### **Step 3: Forward Attack (From Start)**

For each possible K1:

$P \rightarrow [\text{Encrypt with } K1] \rightarrow I1$

Store (I1, K1) in a table

**What happens:** Try all first-half keys and see where they lead

##### **Step 4: Backward Attack (From End)**

For each possible K2:

$C \rightarrow [\text{Decrypt with } K2] \rightarrow I2$

Check if I2 exists in table

**What happens:** Try all second-half keys and see where they come from

### Step 5: The "Meet" Moment

When  $I1 == I2$ :

SUCCESS! Full Key =  $K1 + K2$

**The magic:** Both paths meet at the same middle point!

## 2. Real Example

Let's say DES encryption works like this:

"HELLO" → [FIRST HALF] → "MID123" → [SECOND HALF] → "XJ9#L"

### Attack Process:

ATTACKER TRIES:

$K1 = \text{"ABC"} \rightarrow \text{"HELLO"} \rightarrow \text{Encrypt} \rightarrow \text{Gets "MID111"}$

$K1 = \text{"SEC"} \rightarrow \text{"HELLO"} \rightarrow \text{Encrypt} \rightarrow \text{Gets "MID123"}$

$K2 = \text{"XYZ"} \rightarrow \text{"XJ9\#L"} \rightarrow \text{Decrypt} \rightarrow \text{Gets "MID999"}$

$K2 = \text{"RET"} \rightarrow \text{"XJ9\#L"} \rightarrow \text{Decrypt} \rightarrow \text{Gets "MID123"}$

MATCH FOUND! Both reached "MID123"

FULL KEY = "SEC" + "RET" = "SECRET"

## 3. Why This is So Powerful

### Normal Brute Force:

Try all 56-bit keys one by one:

$2^{56} = 72,057,594,037,927,936$  attempts

Time: YEARS!

## MITM Attack:

Try all 28-bit K1 keys:  $2^{28} = 268,435,456$  attempts

Try all 28-bit K2 keys:  $2^{28} = 268,435,456$  attempts

Total: 536,870,912 attempts

Time: HOURS/DAYS!

**Speed Improvement: About 134 million times faster!**

### MITM ATTACK FLOW

PLAINTEXT (P)



| GUESS |

| ALL K1 |

| KEYS |



| PARTIAL |

| ENCRYPT |

| WITH K1 |



INTERMEDIATE I1

CIPHERTEXT (C)



| GUESS |

| ALL K2 |

| KEYS |



| PARTIAL |

| DECRYPT |

| WITH K2 |



INTERMEDIATE I2



COMPARE I1 == I2?



MATCH?



FOUND KEY!

$K = K1 + K2$