****

Hybrid AES DES Encryption Algorithm

*A project report submitted*

*to*

**MANIPAL ACADEMY OF HIGHER EDUCATION**

*For Partial Fulfilment of the Requirement for the*

*Award of the Degree*

*of*

**Bachelor of Technology**

*in*

**Information Technology**

*by*

**Adrian D Silva**

**Reg. No. 225811024**

*Under the guidance of*

Dr. Abhijit Das

Assistant Professor – Senior Scale

Department of I &T

Manipal Institute of Technology

Bengaluru, India

****

**Nov 2024**

**DECLARATION**

We hereby declare that this project work entitled Hybrid AES DES Encryption Algorithm is original and has been carried out by us in the Department of Information Technology of Manipal Institute of Technology, Bengaluru, under the guidance of **Dr. Abhijit Das**, **Assistant Professor- Senior Scale**, Department of Information Technology, MIT, Bengaluru. No part of this work has been submitted for the award of a degree or diploma either to this University or to any other Universities.

Place: Bengaluru

Date :07-11-24

ADRIAN D SILVA

****

**CERTIFICATE**

This is to certify that this project entitled **Hybrid AES DES Encryption Algorithm**

is a bonafide project work done by **Adrian D Silva (Reg.No.: 225811024)** at Manipal Institute of Technology, Bengaluru, independently under our guidance and supervision for the award of the Degree of Bachelor of Technology in Information Technology.

Dr. Abhijit Das Dr. Dayananda P

Assistant Professor – Senior Scale Professor & Head

Department of I &T Department of I & T

Manipal Institute of Technology Manipal Institute of Technology

Bengaluru, India Bengaluru, India

**Table of Contents**

Introduction

Objectives

Tools and Technology used

System Workflow

Implementation Details

Results

Challenges Faced

Future Enhancements

Conclusion

Codes

**1. Introduction**

Encryption is a crucial aspect of data security, widely used in protecting sensitive information across various fields, including finance, healthcare, and communications. This project aims to develop a Python-based graphical user interface (GUI) application for encrypting and decrypting data using a two-stage encryption process. The system utilizes both DES-like and AES-like algorithms for encryption, demonstrating a simple but effective approach to layered encryption. The project includes a user-friendly interface built with Tkinter, enabling seamless interaction with the encryption and decryption processes.

**2. Objectives**

* To develop a user-friendly application that allows users to input plaintext, encrypt it, and decrypt it back to the original text.
* To demonstrate a two-stage encryption process using DES-like and AES-like algorithms.
* To ensure a smooth user experience by implementing an intuitive interface with Tkinter.

**3. Tools and Technologies Used**

* Programming Language: Python
* Libraries:
  + Tkinter: For creating the graphical user interface.
  + os: For generating random keys used in the encryption process.
* Hardware Requirements: No specific hardware requirements other than a computer capable of running Python.

**4. System Workflow**

**4.1 User Interface**

The system provides an easy-to-use interface where users can:

* Input plaintext for encryption.
* Encrypt the input text using the two-stage encryption process.
* Decrypt the encrypted data back to the original plaintext.

**4.2 Encryption Process**

* DES-like Encryption: The system first encrypts the plaintext using a basic XOR-based DES-like algorithm with a randomly generated 8-byte key.
* AES-like Encryption: The data is then encrypted using another XOR-based AES-like algorithm with a 16-byte key.

**4.3 Decryption Process**

* The decryption is performed in reverse order, first decrypting using AES-like and then DES-like decryption to retrieve the original plaintext.

**4.4 Result Display**

* After encryption, the encrypted data is shown in hexadecimal format.
* After decryption, the system displays whether the decrypted data matches the original plaintext.

**5. Implementation Details**

**5.1 Key Functions**

* encryption\_logic.py:
  + xor\_encryption(): A simple XOR encryption function for both DES-like and AES-like encryption.
  + encrypt(): Manages the two-stage encryption process.
  + decrypt(): Manages the decryption process by reversing the encryption steps.
* tkinter\_interface.py:
  + create\_ui(): Builds the graphical interface for user interaction.
  + encrypt\_text(): Handles the encryption process when the user inputs plaintext.
  + decrypt\_text(): Handles the decryption process for the encrypted text.

**5.2 Key Design Choices**

* The use of XOR-based encryption is a simplified demonstration of how DES and AES encryption techniques work.
* Random key generation using os.urandom ensures that each encryption is unique, even with identical input data.

**6. Results**

After comparing two signatures, the system provides a percentage similarity score. If the score exceeds the threshold, it confirms that the signatures match. Otherwise, it highlights that they do not match.

Example Result:

* Plaintext: "Hello, World!"
* Encrypted Data: 90a1bd9c8f... (Hexadecimal format)
* Decrypted Data: "Hello, World!"
* Result: Success, Data Matches!

**7. Challenges Faced**

* Security of Algorithms: The XOR-based algorithms used in this project are not secure for real-world applications, limiting the system’s potential for practical use.
* GUI Layout: Ensuring the Tkinter interface was simple and intuitive while providing all necessary features.
* Key Management: Random key generation adds complexity to ensuring that encryption and decryption are correctly matched.

**8. Future Enhancements**

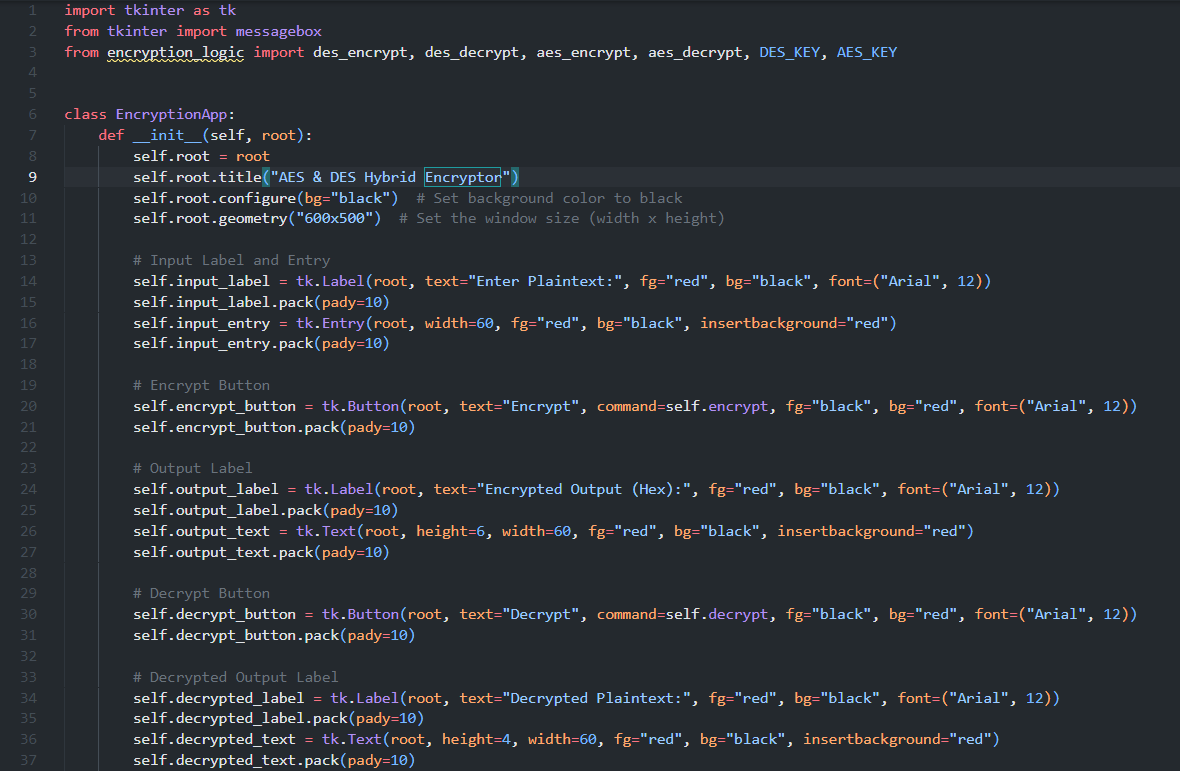
* Advanced Encryption Algorithms: Replacing the XOR-based methods with standard cryptographic libraries such as pycryptodome for implementing actual DES and AES encryption.
* File Encryption/Decryption: Adding the ability to encrypt and decrypt files, not just text data.
* User Authentication: Implementing an authentication layer to allow encrypted data to be accessed only by authorized users.

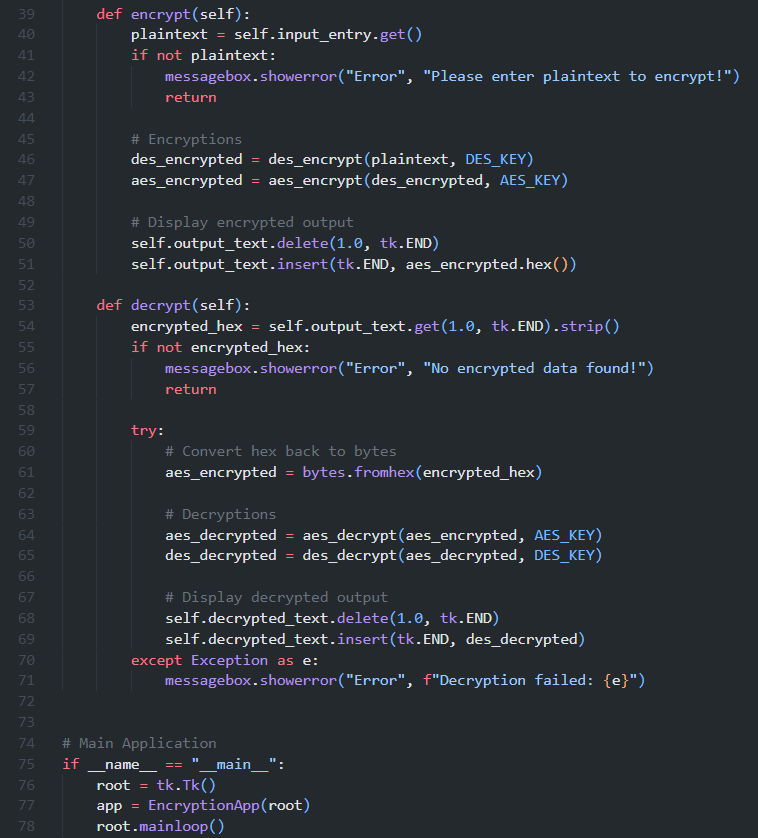
**9. Conclusion**

This project successfully demonstrates the concept of layered encryption using simple, yet effective algorithms for encryption and decryption. The application’s user interface, built with Tkinter, provides a seamless experience for users to interact with the encryption system. While the current encryption method is educational and does not provide secure encryption, it lays the foundation for implementing real-world encryption algorithms and offers an easy-to-understand introduction to cryptography concepts.

**CODES:**

**main\_app.py**

****

****

**encryption\_logic.py**



**Output:**

