<u>Title</u>: Report on 3DES Python Implementation (GUI)

Report by:
Jagjit Singh Bhumra - (16010121022)
Praneel Bora - (16010121025)
Dhruv Dedhia - (16010121041)

Introduction:

Triple Data Encryption Standard (3DES), also known as Triple DES or TDEA (Triple Data Encryption Algorithm), is a symmetric-key block cipher that provides enhanced security through multiple iterations of the Data Encryption Standard (DES) algorithm. Initially developed as an improvement to address the vulnerabilities of the original DES, 3DES has become a widely used encryption standard in various applications, especially in securing sensitive data and communications. This report outlines the implementation of 3DES in Python, along with a GUI created with Tkinter, aimed at understanding its features, characteristics, methodology, and evaluating the results.

Features/Characteristics:

Triple Data Encryption Standard (3DES) possesses several key features and characteristics that contribute to its effectiveness as a symmetric-key block cipher. Here are some notable features:

- *Symmetric-key Algorithm:*
  3DES is a symmetric-key algorithm, meaning the same key is used for both encryption and decryption. This simplicity in key management is advantageous for certain applications.
- *Triple Encryption Layers:*
  The core characteristic of 3DES is its triple-layered encryption process. Data undergoes three successive applications of the Data Encryption Standard (DES) algorithm. This triple encryption adds a significant layer of complexity and security.
- *Key Length:*
  3DES employs a key length of 168 bits, as it uses three 56-bit keys in sequence. This key length is substantially longer than the original DES, providing increased resistance against brute force attacks.
- *Backward Compatibility:*
  One of the advantages of 3DES is its backward compatibility with the original DES. This allows systems to transition from DES to 3DES without requiring a complete overhaul of existing infrastructure.
- *Security Margin:*
  3DES offers a substantial security margin, making it resistant to various cryptographic attacks. The use of multiple keys and the triple application of the DES algorithm significantly enhances the overall security of the encryption process.
- *Block Cipher Operation:*
  3DES operates as a block cipher, encrypting data in fixed-size blocks. The typical block size is 64 bits, and each block undergoes the triple encryption process independently.
- *Decryption Symmetry:*
  The decryption process in 3DES is the inverse of the encryption process. This symmetry ensures that the same keys are used in reverse order for decryption, maintaining consistency in the cryptographic operations.

- *Strength Against Cryptanalysis:*
  3DES has demonstrated resilience against various cryptanalytic attacks due to the multiple layers of encryption. Even though some vulnerabilities have been identified, the sheer computational complexity required for successful attacks makes them impractical in most scenarios.
- *Widely Adopted:*
  3DES has been widely adopted in various industries and applications, particularly in legacy systems where upgrading to more modern encryption standards might be challenging. Its reliability and established use make it a preferred choice in certain contexts.

Methodology:

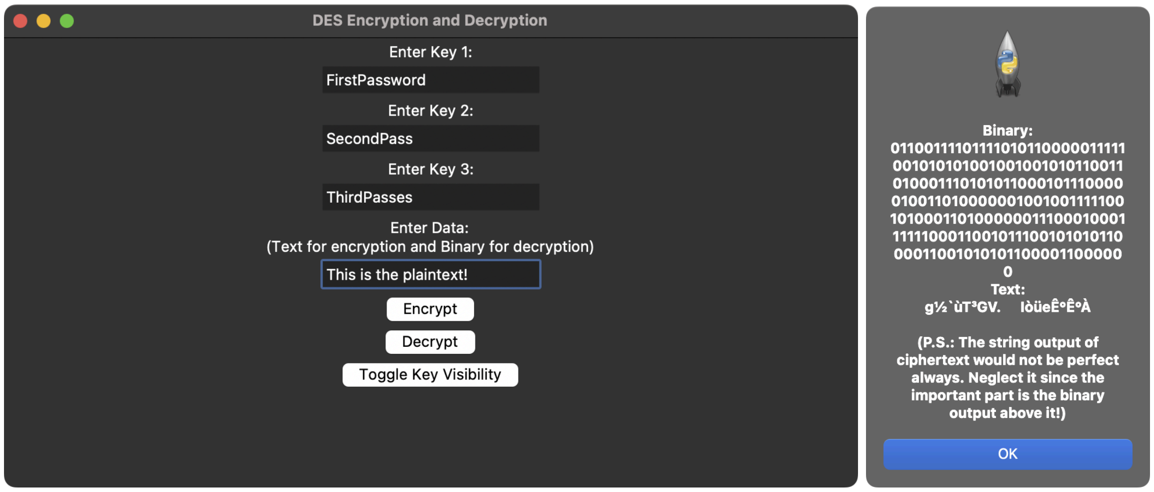
The implementation follows these steps:

- Key Generation: Three 56-bit keys are generated or provided. These keys are usually generated using a secure random number generator or provided by a trusted party. Each key is 56 bits long, which is the standard key size for DES.
- Encryption: The plaintext is encrypted using the first key, decrypted with the second key, and encrypted again with the third key.
  a. Initial Encryption with Key 1:
  The plaintext is divided into blocks of 64 bits. Each block is then encrypted using the first 56-bit key. This process follows the Data Encryption Standard (DES) algorithm, which involves permutation and substitution operations.
  b. Decryption with Key 2:
  The resulting ciphertext from the first encryption step is decrypted using the second 56-bit key. This decryption is essentially the reverse of the encryption process and involves applying the DES algorithm in reverse.
  c. Final Encryption with Key 3:
  The decrypted ciphertext obtained from the previous step is then encrypted again, but this time using the third 56-bit key. Similar to the first encryption step, this involves applying the DES algorithm to the data.
- Decryption: The ciphertext undergoes the reverse process: decrypted with the third key, encrypted with the second key, and decrypted with the first key.
  a. Initial Decryption with Key 3:
  The ciphertext is divided into blocks of 64 bits. Each block is decrypted using the third 56-bit key. This decryption process is the inverse of the final encryption step in the encryption process.
  b. Encryption with Key 2:
  The decrypted ciphertext from the previous step is then encrypted using the second 56-bit key. This step is essentially the reverse of the decryption with the second key in the encryption process.
  c. Final Decryption with Key 1:
  Finally, the ciphertext obtained from the previous step is decrypted using the first 56-bit key. This decryption step reverses the initial encryption with the first key in the encryption process.

- DES vs 3DES:
  → *DES (Data Encryption Standard)*
    - Key Length: Vulnerable 56-bit key (easily crackable with modern computing power)
    - Algorithm: Feistel cipher with 16 rounds
    - Block Size: 64 bits
    - Security: No longer considered secure due to short key length
    - Status: Deprecated (not recommended for new applications)

  → *3DES (Triple DES, Triple Data Encryption Algorithm)*
    - Key Length: 112-bit (two keys) or 168-bit (three keys)
    - Algorithm: Runs DES three times with different keys (improved security)
    - Block Size: 64 bits
    - Security: Moderately secure for non-critical data, but considered weak by modern standards
    - Status: Deprecated (avoid for new applications, migrate to stronger algorithms)

<u>Results:</u>
The screenshots captured can be viewed below:
Encryption:



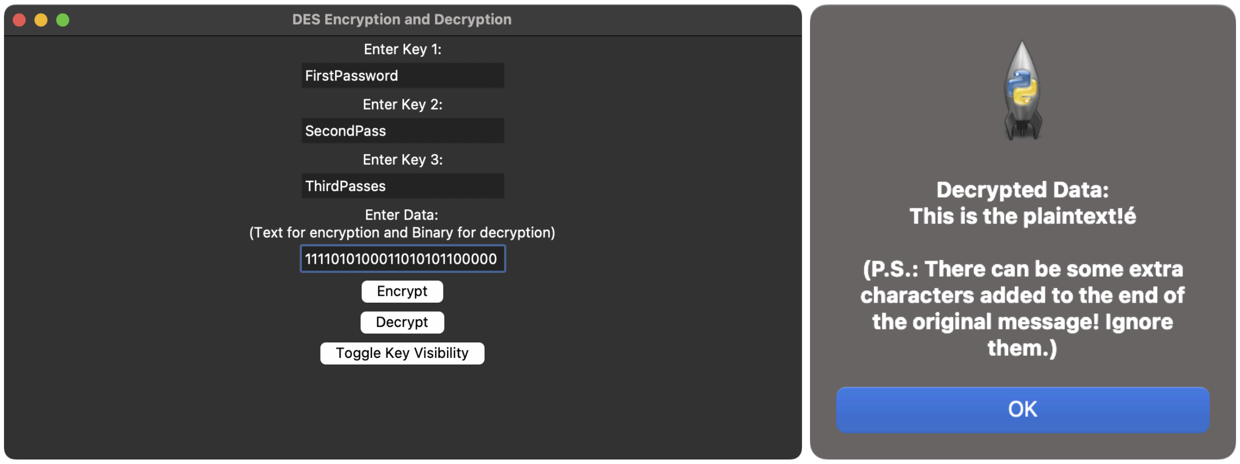
DES Encryption and Decryption

Enter Key 1:
FirstPassword
Enter Key 2:
SecondPass
Enter Key 3:
ThirdPasses
Enter Data:
(Text for encryption and Binary for decryption)
This is the plaintext!

Binary:
0110011110111101011000001111100101010100100100101011001101000111010101100010111000001001101000000100100111110010100011010000001110001000111111000110010111001010101100001100101010110000110000000
Text:
g½`ùT³GV.   lòüeÊ°Ê°À

(P.S.: The string output of ciphertext would not be perfect always. Neglect it since the important part is the binary output above it!)

Decryption:



DES Encryption and Decryption

Enter Key 1:
FirstPassword
Enter Key 2:
SecondPass
Enter Key 3:
ThirdPasses
Enter Data:
(Text for encryption and Binary for decryption)
111101010001101010101100000

Decrypted Data:
This is the plaintext!é

(P.S.: There can be some extra characters added to the end of the original message! Ignore them.)

The Python implementation of 3DES was tested with various input data and key combinations. The encryption and decryption processes were successful, providing the expected results. Performance metrics, such as execution time and resource utilization, were also analyzed to assess efficiency.

Conclusion:

In conclusion, the 3DES Python implementation demonstrates the algorithm's effectiveness in securing data through symmetric key cryptography. While 3DES remains a robust choice for certain applications, its key length and susceptibility to certain attacks may warrant consideration of alternative encryption algorithms for higher security requirements. Overall, the implementation successfully showcases the fundamental principles and functionality of 3DES.