Data Encryption Standard
The Chamber of Secrets: Guarding Digital Realms

Niloy Kumar Mondal
Student ID: 2105044

Monjur Hossain Khan Shovon
Student ID: 2105043

Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology

November 25, 2025

Contents

List of Tables

List of Figures

1 Abstract

Cryptography is a technique for secure data communication. Encryption is the process of encoding messages in such a way that only authorized parties can read it. Decryption is the process of transforming data that has been rendered unreadable through encryption back to its unencrypted form. DES algorithm is a 64 bit block cipher with key of 56 bits. In this report we will discuss the DES technique for secure data transmission while maintaining the authenticity and integrity of the message. In this, message is encrypted before the data transmission process starts. The encryption and decryption of data is done by using the data encryption standard algorithm.

2 Introduction

This report briefly describes the well known algorithm for Data Encryption process with simulation and example starting from the necessity of data encryption with a story. [1]

2.1 Problem Definition

The Data Encryption Standard (DES) is a symmetric-key block cipher that encrypts data in 64-bit blocks using a 56-bit key, aiming to provide data confidentiality by transforming plaintext into ciphertext. The encryption process involves 16 rounds of permutations and substitutions, producing an encrypted output that can only be decrypted with the correct key.

Input	Output
Plaintext: 0101010101...010101 Key: 56 bit Key	Ciphertext: 3A7BD3A...6A4C99C45

Table 1: Example of DES Encryption Input and Output

We want to convert human readable format into unreadable format using a blackbox that will offer reverese extraction of readable format later.

Figure 1: Comparison of Plaintext and Ciphertext

2.2 Key Constraints

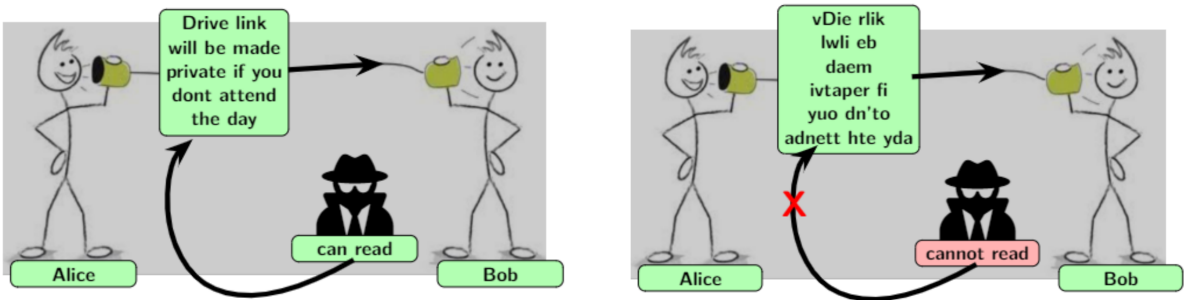
- Speed of Encryption and Reversibility
- Computationally hard backtracking

[1] The story is entirely fictional and used solely for illustrative purposes.

3 Motivation

Imagine a scenario,Alice wants to send bob some texts. If Alice sends it in a simple and understandable format in figure 2a , there will be a high chances of privace breach over the transmission line.

Can we do better?



(a) Sending Plaintext (b) Sending Ciphertext

Figure 2: Comparison between data transmission format

If we transmit any permutation of the data, it will be difficult for unauthorized parties or third parties to comprehend the format, even if they gain access to it. At the same time, we must ensure that the actual data can still be extracted. To defend against brute force or backtracking attacks, we should rely on NP-hard [2]problems.

4 Previous Work

The Data Encryption Standard (DES) has been widely studied since its inception. Rivest (1978) discusses the basic structure and security considerations of DES, highlighting its strengths and limitations in modern cryptography [3]. Despite being one of the most well-known encryption algorithms, DES has faced criticism over its key length and vulnerability to brute force attacks. In his book, Schneier (1996) provides an extensive overview of cryptographic algorithms, including DES, and discusses various methods to enhance its security [4].

Further cryptographic analysis has been conducted to evaluate the security of DES under different attack models. Bellare and Rogaway (1997) presented a comprehensive examination of the security of digital signature schemes, including those based on DES, emphasizing the importance of key management and proper implementation [2]. Moreover, Bell and Greenberg (1995) provided a cryptographic analysis of DES, focusing on its weaknesses and the potential for cryptanalysis techniques to break the algorithm [1].

These works collectively contribute to understanding the vulnerabilities and strengths of DES, laying the groundwork for the development of more secure encryption standards in the cryptographic community.

[2]NP-hard problems are those for which no known polynomial-time algorithm exists to solve them, and they are at least as hard as the hardest problems in NP.

5 Approach

5.1 Algorithm

5.1.1 Overview

- Symmetric key encryption algorithm.
- Operates on 64-bit plaintext blocks.
- Uses a 56-bit secret key.
- Employs 16 rounds of Feistel network structure.
- Ensures data confidentiality through substitution and permutation.
- faster runtime

5.1.2 Visualization of the flow

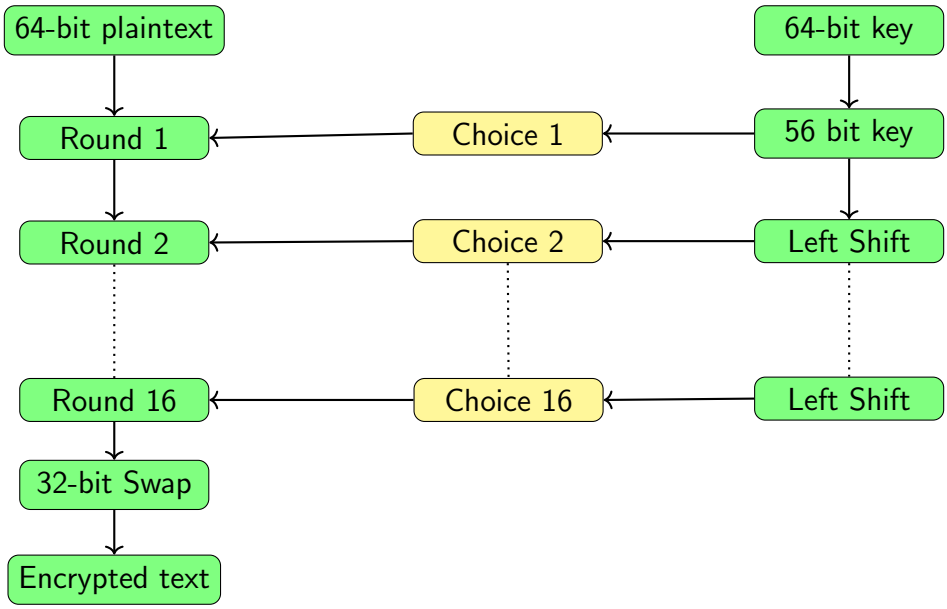


Figure 3: DES Algorithm Simulation

5.1.3 Single round simulation

In a single round of the Data Encryption Standard (DES) algorithm, the 64-bit input block is divided into two halves: the left half (L_i) and the right half (R_i). The right half undergoes a transformation through a function f using a round-specific subkey K_i, which is derived from the original key. The left half (L_i) is then XORed with the output of the function f and the right half, producing the new right half for the next round (R_{i+1}). Simultaneously, the right half becomes the new left half for the next round, while the left half is replaced by the newly computed right half. This process is repeated for 16 rounds, with each round using a new subkey. Finally, after the 16th round, the left and right halves are swapped and combined, and an inverse permutation is applied to generate the ciphertext.

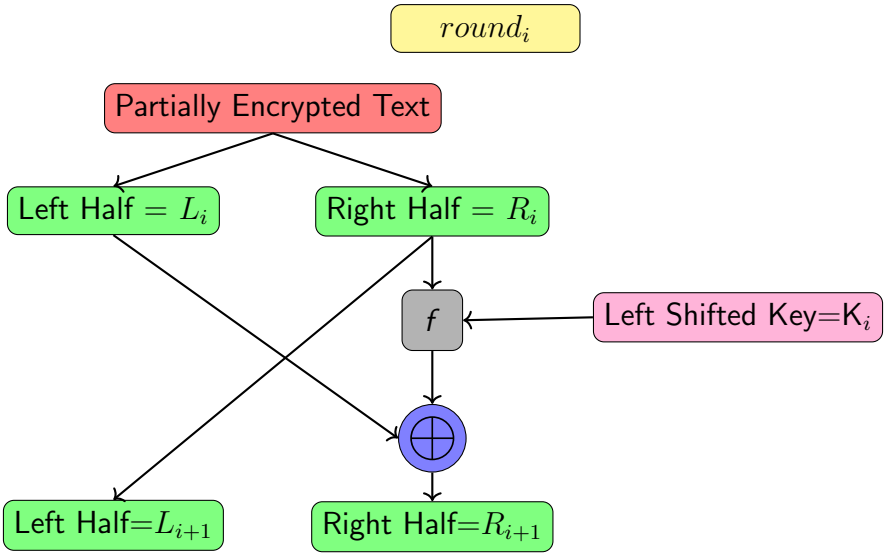


Figure 4: Visaulization of single round

5.2 Pseudocode

1. **Input:** 64-bit plaintext block P, 64-bit main key K.
2. Generate 16 round sub-keys $K_1, K_2, \ldots, K_{16}$ from K.
3. Apply initial permutation (IP) on P to get L_0 and R_0.
4. **for** $i \leftarrow 1$ to 16 **do**
   - Compute $L_i \leftarrow R_{i-1}$.
   - Compute $R_i \leftarrow L_{i-1} \oplus f(R_{i-1}, K_i)$.
5. **end for**
6. Combine R_{16} and L_{16}.
7. Apply inverse initial permutation (IP^{-1}).
8. **Output:** 64-bit ciphertext block C.

5.3 Runtime

- **Key Generation:** The key generation step involves creating 16 sub-keys from the original 64-bit key. This involves shifting the key, performing permutations, and splitting it into two halves. These operations are done for 16 rounds.
  - **Complexity:** Each operation (shifting, permutation) is constant time, i.e., $O(1)$. Since this process is done 16 times, the total complexity for key generation is $O(16) = O(1)$.
- **Initial Permutation (IP):** The initial permutation is a fixed permutation applied to the 64-bit plaintext.

- **Complexity:** This is a constant-time operation because the permutation is predefined and applied to a fixed-size block. So, the complexity is $O(1)$.

- **Rounds (16 rounds):** For each of the 16 rounds, the algorithm applies the following operations:
  - Splitting the input into two 32-bit blocks.
  - The f-function, which involves expansion, substitution, and permutation, all of which take constant time $O(1)$.
  - XOR operation with the round key.
  - A swapping operation (swapping the two halves of the block).
  - **Complexity:** Each round involves a constant number of operations (expansion, substitution, XOR, permutation). Since there are 16 rounds, the total complexity for all rounds is $O(16) = O(1)$.

- **Final Permutation ($\mathbf{IP}^{-1}$):** The final permutation is similar to the initial permutation and is applied to the result of the 16th round.
  - **Complexity:** This is also a constant-time operation, i.e., $O(1)$.

Total Complexity:

- Key Generation: $O(1)$
- Initial Permutation: $O(1)$
- Rounds: $O(1)$ for each round, repeated 16 times: $O(16) = O(1)$
- Final Permutation: $O(1)$

Thus, the overall time complexity of the DES algorithm is:

$O(1)$ (constant time for key generation, permutation, and rounds).

6 Mathmetical Simulation With an Example

- **64-bit Input Plaintext (Binary):**

```
00101001 00110011 11110000 10101010
11001100 11001100 11110000 00000000
```

- **56-bit Key (Binary):**

```
00010011 00110100 01010111 01111001
10011011 10111100 11011111 11100001
```

Step 2: Initial Permutation (IP)

The 64-bit input undergoes the initial permutation based on the IP table. Each bit is rearranged as specified.

Input (Plaintext):

```
00101001 00110011 11110000 10101010
11001100 11001100 11110000 00000000
```

Output After Initial Permutation:

```
11001100 10101010 11110000 11001100
11110000 10101010 00110011 00101001
```

Step 3: Splitting the Block

The 64-bit permuted block is divided into two 32-bit halves:

$L_0 =$ `11001100 10101010 11110000 11001100`, $R_0 =$ `11110000 10101010 00110011 00101001`

Step 4: Key Scheduling

The 56-bit key is used to generate 16 round keys $K_1, K_2, \ldots, K_{16}$ through permutation and left shifts.

Key Input:

```
00010011 00110100 01010111 01111001
10011011 10111100 11011111 11100001
```

Example Round Key K_1:

```
11110000 11001100 10101010 11110000
```

Step 5: 16 Rounds of Feistel Function

Each round applies the Feistel function F, which includes expansion, substitution (using S-boxes), and permutation.

Round 1 Example:

- **Input:** $L_0 =$ `11001100 10101010 11110000 11001100`, $R_0 =$ `11110000 10101010 00110011 0`
- **Key:** $K_1 =$ `11110000 11001100 10101010 11110000`
- **Expansion (E):** R_0 expanded to 48 bits.
- **Feistel Output:** $F(R_0, K_1) =$ `01101010 10111011 11010010 11101010`
- **New Values:**

$$L_1 = R_0, \quad R_1 = L_0 \oplus F(R_0, K_1)$$

Output after Round 1:

```
L_1 = 11110000 10101010 00110011 00101001
R_1 = 10100110 00010001 00100010 00100110
```

Step 6: Final Permutation (FP)

After 16 rounds, the left and right halves are concatenated and subjected to the Final Permutation (FP).

Output Ciphertext:

```
10000101 11101000 00010010 10101101
00000001 00101011 10101101 01100000
```

Summary

The DES algorithm transforms the 64-bit plaintext into a secure 64-bit ciphertext after 16 rounds of processing.

Results:

- **Input Plaintext (Binary):**

  ```
  00101001 00110011 11110000 10101010
  11001100 11001100 11110000 00000000
  ```

- **Input Key (Binary):**

  ```
  00010011 00110100 01010111 01111001
  10011011 10111100 11011111 11100001
  ```

- **Output Ciphertext (Binary):**

  ```
  10000101 11101000 00010010 10101101
  00000001 00101011 10101101 01100000
  ```

7 Application

- **Secure Communications**: DES was widely used in securing communications for government organizations, military agencies, and financial institutions.
- **Financial Transactions**: DES was employed in securing electronic payment systems, including ATMs and point-of-sale terminals, as well as protecting financial data during transmission.
- **Data Integrity and Authentication**: DES was used in conjunction with other cryptographic mechanisms like DES-MAC to ensure data integrity and authentication during transmission.
- **Legacy Systems**: Many older systems still use DES for compatibility reasons, often combined with other encryption techniques such as Triple DES (3DES) for enhanced security.
- **Government and Military Applications**: DES was adopted by the U.S. government and military for encrypting sensitive communications and protecting classified information.

8 Challanges and Limitations

- **Short Key Length:** The 56-bit key length is considered too short for modern security requirements.
- **Vulnerability to Brute-Force Attacks:** DES can be cracked in less than 24 hours with modern computational techniques.
- **Limited Security Against Cryptanalysis:** DES is susceptible to advanced cryptanalytic techniques, such as differential cryptanalysis.
- **Small Block Size:** The 64-bit block size may lead to repeated ciphertext patterns, especially in large datasets.
- **Not Suitable for Large-Scale Systems:** The limitations of DES make it unsuitable for use in large-scale or high-security systems.
- **Outdated for Modern Applications:** DES has been replaced by more secure algorithms, such as AES, due to its insufficient security for contemporary applications.

9 Conclusions

9.1 Relative comparison and summary

Feature	DEA	DES
Key Length	56-bit	56-bit
Block Size	64-bit	64-bit
Brute-Force Resistance	Low	Low
Cryptanalysis Resistance	Weak	Moderate
Algorithm Design	Simple	Complex

Table 2: DEA vs DES Comparison

9.2 Future of DES

DES was a pioneering encryption standard, but it is now considered outdated due to its vulnerabilities. Its limitations led to the development of stronger algorithms, such as AES, which provide better security. As we move forward, the focus is shifting toward post-quantum cryptography, with continuous advancements needed to secure data in an increasingly digital world.

References

[1] David A. Bell and Leonard M. Greenberg. "A Cryptographic Analysis of the Data Encryption Standard (DES)". In: *Journal of Cryptographic Engineering* 2.1 (1995), pp. 25–38. DOI: 10.1007/BF02901068. URL: https://link.springer.com/article/10.1007/BF02901068.

[2] Mihir Bellare and Phillip Rogaway. “The Exact Security of Digital Signature Schemes”. In: *Journal of Cryptology* 10.2 (1997), pp. 101–137. DOI: 10.1007/s001459900039. URL: https://link.springer.com/article/10.1007/s001459900039.

[3] Ronald L. Rivest. “The Data Encryption Standard (DES) and its Strengths”. In: *Proceedings of the IEEE* 66.3 (1978), pp. 301–306. DOI: 10.1109/PROC.1978.10968. URL: https://ieeexplore.ieee.org/document/1465473.

[4] Bruce Schneier. *Applied Cryptography: Protocols, Algorithms, and Source Code in C*. 2nd. John Wiley & Sons, 1996. ISBN: 978-0471117094. URL: https://www.amazon.com/Applied-Cryptography-Protocols-Algorithms-Source/dp/0471117099.