VFAST Transactions on Software Engineering
https://vfast.org/journals/index.php/VTSE@ 2025, ISSN(e): 2309-3978, ISSN(p): 2411-6246
Volume 13, Number 1, January-March 2025 *pp:178-192*

Lightweight Cryptographic Algorithm Development Using Fundamental Cryptographic Techniques

Mubbashir Ahmed [1]*

[1]Department of Computer Science, Iqra University, Karachi, Pakistan

Keywords:
cryptography, encrypt, decrypt, data, security, algorithm

Journal Info:
Submitted:
February 4, 2025
Accepted:
March 21, 2025
Published:
March 31, 2025

Abstract

Cryptography is used to make data and information transmission, and computational systems secure over the networks by using mathematical and scientific techniques. The cryptographic algorithm should ensure authentication, confidentiality, integrity, and reliability. In today's era, where digital communication and data storage are increasing rapidly, while data leakage, breaches, and attacks are also on the rise. This emphasizes the increasing need for strong and secure cryptography algorithms to protect user information that ensures the integrity and confidentiality of data. The existing symmetric cryptographic algorithms like AES or DES can provide robust security, but they have very complex implementations and require high computational resources. The aim of this paper is to provide a review of existing research done in the field of cryptography, cryptographic techniques, and to propose a lightweight symmetric cryptographic algorithm using various fundamental secure techniques that satisfy the conditions of authentication, confidentiality, integrity, and reliability.

***Correspondence author email address:** mubbashirahmed12@gmail.com
DOI: 10.21015/vtse.v13i1.2050

1 Introduction

The word Cryptography consists of two Greek words. The first word is "Crypto" which means hidden or secret, and the second word is "Graphy" means to write or to draw.

Cryptography falls into the field of computer networks that deals with techniques that satisfy integrity, store secret keys and data confidentially, and share them securely by following authentication protocols [1].

In today's era, where almost all operations are performed on networks, cryptography is a method of keeping information secret and secure. When communications are through unreliable mediums, attackers can perform various attacks such as brute-force to acquire information. So, security and protection from attacks can be achieved by using cryptography.

In current digital era, the field of cryptography has been advanced with the integration of new techniques and enhancements that driven this towards an

This work is licensed under a Creative Commons Attribution 3.0 License.

emerging field known as Modern Cryptography. The field of modern cryptography deals with the study of advanced mathematical techniques and algorithms to secure data, information and computational system more confidentially against unwanted data leakage, theft and attacks [1].

Cryptography ensures data security and privacy by transforming the original data into an unreadable format for users who are not authorized to access the data and converts the data back to its original format for users who are authorized to access the data [2].

The field of cryptography is classified into two main categories, classical cryptography and modern cryptography. Where, classical cryptography holds substitution and transposition techniques that provide algorithms such as caesar ciphers, and vigenere ciphers. On the other hand, modern cryptography works on symmetric and asymmetric techniques. Further, symmetric cryptography is classified into block cipher and stream cipher techniques that provides algorithms such as AES, DES, Blowfish, Fish, and RC4. Asymmetric cryptography provided algorithms such as RSA, DSA, and Diffe-Hellman [3].

2 Literature Review

The authors presented a comprehensive overview of cryptographic principles, techniques, and applications. Their book provided readers with a solid foundation in cryptography by covering concepts such as symmetric and asymmetric encryption and decryption, hash functions, and digital signatures by formal definitions and proofs [1].

The authors addressed the importance of data security and the need to protect data by fulfilling all four pillars, that are authentication, confidentiality, integrity, and reliability. Further, the authors explained the detailed overview of both symmetric and asymmetric cryptographic algorithms based on block cipher approach. Furthermore, they also proposed an algorithm based on the block cipher approach with less execution time and impossible to break [2].

The authors explored a detailed overview of conventional and modern cryptographic algorithms by categorizing them into symmetric and asymmetric approaches. The paper highlights the vulnerabilities to brute-force and frequency analysis attacks for classical ciphers including caesar cipher and vigenere cipher and also highlights the weaknesses of modern symmetric algorithms such as Blowfish, Twofish, DES, and AES. Further, the study analyzed asymmetric algorithms and ECC achieved strong security with smaller keys. To conclude, the paper points out the use of symmetric and asymmetric approaches together to form better and secure communication and balance performance [4].

The authors provided a comprehensive analysis of different symmetric and asymmetric algorithms with respect to their security, speed, and efficiency. The findings also stated that symmetric algorithms can perform faster encryption and decryption processes and use low computational resources compared to asymmetric algorithms. They also pointed out as drawbacks that key management as key should be securely shared between authenticated users, as this enforces a major security challenge [5].

The authors highlighted the main role of cryptography in securing the data from unauthorized users. The authors also discussed the advantages of using the symmetric approach as less usage of computing resources, faster encryption and decryption operations, and they also discussed key management and security issue as a disadvantage. Furthermore, the authors made an advancement in the field by providing a new symmetric cryptographic algorithm with enhanced data security features that works on ASCII characters and binary numbers [6].

The authors inquired about usage, advantages and disadvantages of the existing cryptographic symmetric and asymmetric algorithms including AES, DES, 3DES, Blowfish, Twofish and RSA. Furthermore, the authors also gave a detailed table of comparisons of the mentioned algorithms with respect to key and sub-keys and their length, block size, their mathematical operations, speed, power usage, security, provided features, and the types of attacks that can be performed on the algorithm to acquire the data [7].

The authors explained in detail about the history

of cryptography, its fundamental definitions, ancient cryptography models, basic techniques, and theorems used to build these models and their applications. Further, this research also expanded the evolution of present or digital cryptography focusing on the importance of securing information. Furthermore, the author presented a new type of cryptographic model using a redesigned solvability proposition and series [8].

The authors conducted a detailed analysis of symmetric and asymmetric cryptographic algorithms by focusing on their structural foundations, integer factorization, and discrete logarithms. The study also explored more than 40 algorithms based on block size, key size, number of rounds, and weakness to attacks including brute-force, differential cryptanalysis, and key-related attacks. Furthermore, the paper also highlights challenges in key exchange, flexibility, and suitability for lightweight environments [9].

The authors provided a comprehensive analysis of both symmetric and asymmetric cryptographic approaches by reviewing their computational efficiency, structure, and weaknesses. The study reviews execution speed, key size, power consumption, and resistance to attacks such as brute-force, known plain-text attacks for algorithms including AES, DES, RSA, and ECC. Furthermore, the study shows AES has the strongest symmetric functionalities and ECC is the most efficient alternative to RSA for secure communications in resource-limited environments [10].

The authors provided an in-depth analysis of symmetric cryptographic algorithms including AES, DES and 3DES and asymmetric cryptographic algorithms including RSA and Diffie-Hellam by focusing on the role in the data security. Further, the authors also explained the operations of both symmetric and asymmetric cryptographic algorithms by using mathematical models. Furthermore, the authors also provided a summarized table of comparisons of the mentioned algorithms with respect to key management and length, power usage, speed [11].

The authors highlighted the analysis and comparison of symmetric, asymmetric, and hash cryptographic algorithms, where in symmetric algorithms, the authors included AES, DES, 3DES, Blowfish, and RC4 as they perform fast operations using the same key for encryption and decryption. In asymmetric algorithms, the authors included RSA, DSA, Diffie-Hellman, ElGamal, and Paillier because they provide strong security using a pair of keys for encryption and decryption. In the hash algorithms, the authors included MD5, MD6, SHA, and SHA256 as they perform strong data reliability and security [12].

The authors provided a context for the importance of password-based key derivation functions in enhancing data security and privacy against unauthenticated users (usually attackers or hackers) and also against brute-force attacks. Further, the authors explained that the use of low iteration counts for password-based key derivation can be vulnerable [13].

The authors provided a context for the challenges confronted by the users in creating and managing traditional text-based passwords, as they are the most commonly used authentication method. Further, the authors proposed a technique to generate strong passwords with minimum user input, also referred by authors as Safe Vault [14].

The author highlighted the techniques for generating strong passwords or keys to enhance data security and privacy. Further, the author proposed a system that operates in two steps to generate a secure derived key. According to the author, the first step of the system encrypts the username using the SHA-256 algorithm to protect users from attackers or hackers, and the second step of the system generates a strong secure derived key using the password-based key derivation function and using salt to extend the derived key to obtain a more secure derived key [15].

The authors presented a new approach to offer a strong password security and privacy using a salt-based password stretching technique. By using this technique, the generated key is more complex and it is difficult to perform attacks to acquire the original key. Furthermore, the authors improved security by increasing the complexity of in the process of key generation by applying factor of 108 [16].

The authors highlighted that the cryptographic algorithms should satisfy the conditions of performance, architecture, scalability, flexibility, limitation, and security. Furthermore, the authors proposed a faster algorithm based on binary digits and compared execution time and key size with symmetric algorithms [17].

The authors proposed recommendations to use password-based key derivation functions for the derivation of cryptographic keys from user-provided password or paraphrase to securely protect the data. Some recommendations include the usage of the master key only once, the iteration count should be as big as possible, the usage of long-length salts [18].

The authors highlighted the weaknesses of the password-based key derivation function PBKDF2 SHA-1. Due to its performance issue, attackers can crack the 50% key by using its precomputed values. Furthermore, the attackers can also perform speed brute-force attacks to crack the key. Furthermore, the authors pointed out another issue of unnecessary operations, which includes applying XOR operation on the same value multiple times. These issues affect the performance of many cryptography libraries [19].

The authors provided a detailed analysis of secure passwords. The authors explained the importance of strong and secure passwords and how it is difficult to break them. Furthermore, the authors performed an analysis of brute-force attacks on different lengths of passwords to figure out the best possible strong and secure password and to minimize the risk of password breaking to acquire the data [20].

The author conducted an analysis of strong passwords and common but small errors that users make when creating a password like using the same password for multiple accounts. It is easy for hackers to perform attacks on them. Furthermore, the author proposed an analysis of using and creating strong and secure passwords with recommended lengths and recommended characters that will help the user to be secure from hacker attacks [21].

The authors proposed a cryptographic scheme designed to increase the security of text message against brute-force and cryptanalytic attacks performed by hackers. Furthermore, their scheme generates a unique key for each message for strong security and provides good efficiency and speed with less computational resource usage [22].

The authors conducted the analysis of the performance of symmetric key cryptographic algorithms that includes DES, 3DES, AES and Blowfish. Furthermore, the author found that all algorithms are secure and met basic security conditions, but the blowfish performed efficiently. The study shows that the blowfish holds a good position in the cryptographic algorithms with respect to security and performance [23].

The authors conducted a review of various cryptographic algorithms, including symmetric and asymmetric cryptographic algorithms and their techniques. Furthermore, the authors conducted an analysis on the basis of data security, key size, block size, and other characteristics [24].

The authors performed a detailed analysis of lightweight block ciphers for resource-limited environments, including IoT devices. The research highlights the limitations including power consumption, memory usage, and computational overhead of cryptographic algorithms such as AES, and DES. Further, the study also performed comparative analysis of lightweight cipher designs focusing on performance including overhead, latency, and efficiency. Furthermore, the authors also highlight their security against standard cryptanalysis attacks in real-world applications on IoT systems [25].

The author explored the application of letter frequency analysis that offers a foundation for classical cryptography by providing decryption of different types of mono-alphabetic substitution ciphers. The study analyzed the types of mono-alphabetic ciphers including shift cipher, random substitution cipher, and affine cipher by reviewing their resistance to frequency-based cryptanalysis, and poly-alphabetic ciphers including hill cipher and playfair cipher by reviewing their ability to be safe from frequency-based cryptanalysis due to their distorted frequency distributions. Furthermore, the paper also shows that the shift cipher and affine cipher can be efficiently decrypted using frequency-based cryptanalysis due to their

predictable and limited character mappings, while random substitution cipher shows resistance due to its randomness and unpredictability in character mappings. [26].

The authors proposed a major enhancement in the conventional substitution caesar cipher, which is exposed to brute-force attacks due to the usage of limited character mappings. To overcome this problem, the authors presented a method which follows arithmetic and divisibility tests by decoupling encryption and decryption logic. Further, their proposed logic resists frequency analysis and shows strong security against unwanted access. [27].

The author applied cryptanalysis techniques to three classical ciphers including caesar cipher, transposition cipher, and hill cipher. The applied cryptanalysis techniques include frequency analysis, crib dragging, and matrix-based decryption to identify the weaknesses of these ciphers when encountered different attacks including known plain-text attacks. Caesar cipher is decrypted using statistical frequency, where transposition cipher is decrypted using permutation testing and plain-text recognition. On the other hand, the study highlights linear algebra to reverse-engineer encryption for hill cipher. [3].

3 Basic Terminologies

There are terminologies and concepts that are commonly used in the field of cryptography and in cryptographic algorithms.

3.1 Plain Data

The data that is in its original and understandable format before any encryption algorithm is performed on it is called plain data. It is given as an input to the encryption algorithm.

3.2 Cipher Data

The data that has been converted from its original format to mixed-up and unreadable format is called cipher data. It is produced using a key from the encryption algorithm.

3.3 Key

It is a valuable data that acts like a password or pass code for the encryption or decryption processes, which is referred as key. The security of the data depends upon the security of key. So, it is necessary to keep the key secure and confidential.

3.4 Encryption

The process to convert plain data into cipher data using a key and encryption algorithm is called as encryption also referred as encipher.

3.5 Decryption

The process to convert cipher data into plain data using a key and reverse process of encryption or decryption algorithm is called as decryption also known as decipher.

3.6 Encoder

The component or a process (in some cases, also referred to user who wants to encrypt the data), who runs an encryption algorithm by using a key.

3.7 Decoder

The component or a process (in some cases, also referred to user who wants to decrypt the encrypted data), who runs a reverse encryption algorithm also known as decryption algorithm by using a key.

3.8 Private Key

A key that is kept secure and confidential and only authenticated user have access to it to perform encryption and decryption process is called private key. Private keys are generally used in symmetric cryptographic algorithms.

3.9 Public Key

A key that is shared on the network and can be used by anyone to perform encryption is called public key. For decryption process, it is necessary to use the correct private key to decrypt the data. Public keys are generally used in asymmetric cryptographic algorithms

4 Process of Cryptography

The process of cryptography includes set of steps that flows as described.

4.1 Encryption Process

The information (plain data) is converted into an unreadable format (cipher data) by using a specific key. This part of flow is also known as encryption.

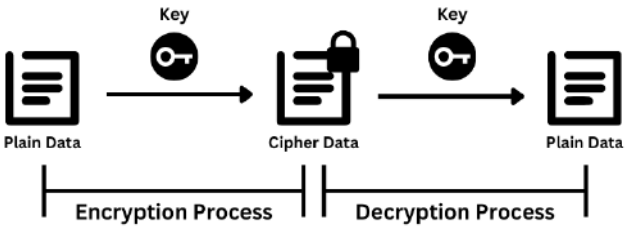

Figure 1. Process of Cryptography

Performing modification on plain data using encryption techniques which results in secure and encrypted data also known as cipher data [5].

4.2 Decryption Process

The unreadable format (cipher data) can only be converted into information (plain data) with the help of a secret key, used to encrypt it. This part of the flow is also known as decryption.

Performing reverse modification on cipher data using decryption techniques (reverse encryption techniques) which results in original data that was encrypted in the encryption process also known as plain text [5].

5 Pillars of Cryptography

Cryptography itself stands on the four pillars, and they are necessary for every cryptographic algorithm and should be followed.

Cryptography is ensured by security, privacy, stability and accessibility of the system by set of pillars [7].

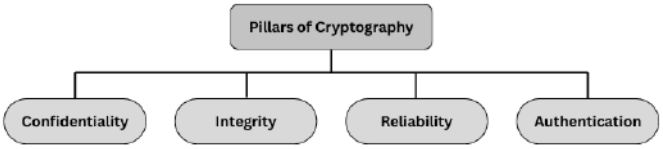

Figure 2. Pillars of Cryptography

5.1 Confidentiality

The first pillar of cryptography is Confidentiality, which concentrates on the privacy of data by transferring the data to the correct and authenticated receiver.

5.2 Integrity

Integrity is the second pillar of cryptography, which focuses on the security of data from any third party or unknown users (usually attackers or hackers) trying to access or perform modification on the data.

5.3 Reliability

The third pillar of cryptography is Reliability, which focuses on the successful conversion of plain data to encrypted data and encrypted data to plain data. It ensures that the system must be operating properly and performs all required operations successfully.

5.4 Authentication

Authentication is the fourth pillar of cryptography, which focuses on the verification and authorization of identities of users who are trying to gain access to the data. It helps the system in maintaining security and protect data from the access of unwanted and unauthorized users (usually attackers or hackers).

6 Types of Cryptography

In the field of cryptography, the techniques for cryptographic algorithms to perform encryption and decryption are mainly categorized into two different categories highlighted as Symmetric Key Cryptography and Asymmetric Key Cryptography.

6.1 Symmetric Key/Secret Key Cryptography

Symmetric Key Cryptography (SKC) also referred as Secret Key Cryptography, where one shared key is used for both encryption and decryption process.

In Symmetric Key Cryptography, the encryption and decryption process must have the access to secret key to obtain the data or perform modifications on the data [4, 6]. There are different algorithms available that were designed on the model of Symmetric Key Cryptography such as AES, DES, and 3DES.

6.2 Asymmetric Key/Public Key Cryptography

Asymmetric Key Cryptography (AKC) also known as Public Key Cryptography, where the pairs of keys consisting of a public key and a private key are used in the process of encryption and decryption.

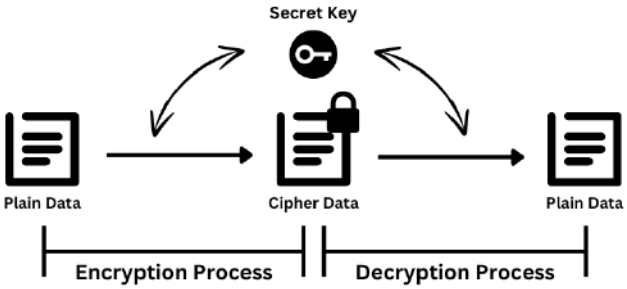

Figure 3. Symmetric Key/Secret Key Cryptography

In Asymmetric Key Cryptography, public key is responsible for the encryption of data and it is available at both encryption and decryption processes. After the data is received, the private key is responsible for the decryption of the data and it is only available at decryption process. Private key also helps in the authentication process of sender and receiver to maintain privacy and security of data [8].

There are various algorithms available that were designed on the model of Asymmetric Key Cryptography such as Diffie-Hellman (DH), Elliptic Curve (EC), and Rivest-Shamir-Adleman (RSA).

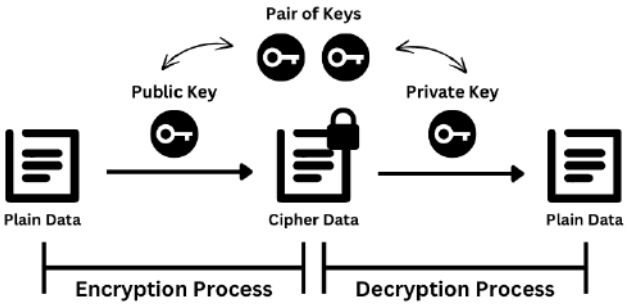

Figure 4. Asymmetric Key/Public Key Cryptography

7 Methodology

Common cryptographic algorithms like AES and DES are highly secure and offer strong security but require more computational resources. To address this challenge, this research presents a lightweight symmetric cryptographic algorithm using fundamental techniques and a symmetric key approach to maintain balance between security and performance. Furthermore, the presented algorithm is designed and optimized specially for resource-limited environments such as IoT devices and embedded systems.

The methodology takes a systematic approach, starting with key derivation using PBKDF2-SHA256, a secure algorithm designed to resist brute force and dictionary attacks. Furthermore, the encryption and decryption process integrate multiple cryptographic techniques such as substitution cipher, rail fence transposition cipher, block cipher and block flipping. This combination enhances security by introducing non-linearity, randomness, and structural complexity, making the system more resilient against cryptographic attacks.

7.1 Using Symmetric Key Approach

When developing algorithms, the algorithm should run cryptographic processes in less execution time while consuming less computational resources.

In symmetric key cryptographic approach, it holds some pros like, it is fast and it runs in minimum execution time and consumes less computational resources. Also, it is easy to apply on real-world applications.

But a disadvantage in this approach is the security and privacy of the key as we use the same key for both encryption and decryption process.

7.2 Cryptographic Techniques Used in the Development of Algorithm

7.2.1 Password-Based Key Derivation Function (PBKDF2)

It is a key derivation function that derivates a password-based key of fixed defined length using a password or keyword and a salt. Salt is used to add extra complexity in the key.

Password-based key derivation function provides a strong resistance against brute force attacks. The key generation is based on iteration counts which maps provided password or keyword values using random function to a derived key [13].

$$K = PBKDF2\_SHA256(P, S, iterations, key\_length)$$

where,

$P \rightarrow$ User provided key

$S \rightarrow$ random value for security

$iterations \rightarrow$ number of hash iterations

$key\_length \rightarrow$ size of key

7.2.2 Block Cipher

Block cipher falls in the symmetric cryptographic algorithm type that uses the same key to perform encryption and decryption. The encryption or decryption on the data is performed on chunks of data (e.g. 64-bits or 128-bits) by breaking data into n-length size of block and each block is encrypted or decrypted using the key.

$$C_i = E(P_i, K), \quad \text{for encryption}$$

$$P_i = E^{-1}(C_i, K), \quad \text{for decryption}$$

where,

$P_i \rightarrow$ Block of plain text

$C_i \rightarrow$ Block of cipher text

$K \rightarrow$ Secret key

$E \rightarrow$ Encryption and decryption function

7.2.3 Substitution Cipher

Substitution cipher is a type of symmetric cryptographic algorithm that works on the same key to perform encryption and decryption. It makes a shifted characters list by shifting the characters (including alphabets, numbers or symbols) by n-positions. The shifted characters list is used to encrypt or decrypt the data by mapping characters.

$$C_i = (P_i + k) \mod 26, \quad \text{for encryption}$$

$$P_i = (C_i + k) \mod 26, \quad \text{for decryption}$$

where,

$P_i \rightarrow$ Character of plain text

$C_i \rightarrow$ Character of cipher text

$k \rightarrow$ Shifted value of secret key

$26 \rightarrow$ number of alphabets in english

7.2.4 Transposition Cipher

Another type of symmetric cryptographic algorithm is transposition cipher that works on the same key to perform encryption and decryption. It works by making n-depth columns and mapping data in rows in a zig-zag pattern.

$$C = P_{\pi}, \quad \text{for encryption}$$

$$P = C_{\pi^{-1}}, \quad \text{for decryption}$$

where,

$\pi \rightarrow$ permutation function for reordering of plain text P

8 Proposed Algorithm

8.1 Algorithm for Encryption Process

Input:

$K_{\text{user}} \rightarrow$ User provided key

$S_{\text{user}} \rightarrow$ User provided salt

$D_{\text{plain}} \rightarrow$ User provided plain data

Output:

$D_{\text{enc}} \rightarrow$ Encrypted data

Step # 1: Generate a Secure Password-Based Derived Key

Generate a secure password-based derived key by user provided key K_{user} and salt S_{user} using PBKDF2 SHA-256 algorithm.

$$K_{\text{der}} = PBKDF2\_SHA256(K_{\text{user}}, S_{\text{user}}, 1000, 256) + S_{\text{user}}$$

where,

$1000 \rightarrow$ iterations

$256 \rightarrow$ key length

Step # 2: Generate a Space Character Placeholder

Generate a space character placeholder using the derived key K_{der} and by taking the mid-point of all the numbers in the derived key K_{der}.

$$N = \text{extract\_digits}(K_{\text{der}})$$

$$m = \begin{cases} 2, & \text{if } \lfloor |N|/2 \rfloor = 0 \\ N_{\lfloor |N|/2 \rfloor}, & \text{otherwise} \end{cases}$$

$$SP = K_{\text{der}} \left[: \frac{|K_{\text{der}}|}{m} \right]$$

Step # 3: Replace Spaces with the Generated Space Character Placeholder

Replace all the spaces with the generated space character placeholder *SP* from the provided data D_{plain}.

$$D = \text{Replace}(D_{\text{plain}}, \text{spaces}, SP)$$

Step # 4: Remove Newlines and Tabs

Clean newlines and tabs from the provided data D_{plain}.

$$D' = \text{Remove}(D, [\text{newlines}, \text{tabs}])$$

Step # 5: Apply Block Cipher Approach

By using block cipher approach, make blocks of n length where n will be calculated on the basis of mid-point of all the numbers *M* in the derived key K_{der}.

$$N = \text{extract\_digits}(K_{\text{der}})$$

$$m = \begin{cases} 2, & \text{if } \lfloor |N|/2 \rfloor = 0 \\ N_{\lfloor |N|/2 \rfloor}, & \text{otherwise} \end{cases}$$

$$b = \lfloor |D'|/m \rfloor$$

Step # 6: Flip the Blocks Horizontally

Flip the blocks horizontally and make a new data sequence by concatenating the flipped blocks together.

$$B_i = \text{flip}(D'[(i-1) \cdot b : i \cdot b]), \quad \text{for } i = 1, 2, \ldots, m$$

$$D' = \bigcup_{i=1}^{m} B_i$$

Step # 7: Apply Rail Fence Transposition Cipher

Using rail fence transposition cipher, make two columns of depth of data length and mapping data in rows in a zig-zag pattern.

$$M_{r,c} = D'[i], \quad \text{where}$$

$$r = \begin{cases} 0, & \text{if } i = 0 \\ r+1, & \text{if dir\_down} = 1 \text{ and } r < k-1 \\ r-1, & \text{if dir\_down} = 0 \text{ and } r > 0 \end{cases}$$

$$c = i$$

$$k = \lfloor |D'|/2 \rfloor$$

$$D' = \bigcup_{r=0}^{k-1} \bigcup_{c=0}^{|D'|-1} M_{r,c}$$

Step # 8: Reapply Block Cipher Approach

Again, by using block cipher approach, make blocks of n length where n will be calculated on the basis of mid-point of all the numbers *M* in the derived key K_{der}.

$$N = \text{extract\_digits}(K_{\text{der}})$$

$$m = \begin{cases} 2, & \text{if } \lfloor |N|/2 \rfloor = 0 \\ N_{\lfloor |N|/2 \rfloor}, & \text{otherwise} \end{cases}$$

$$b = \lfloor |D'|/m \rfloor$$

Step # 9: Flip the Blocks Horizontally Again

Again, flip the blocks horizontally and make a new data sequence by concatenating the flipped blocks together.

$$B_i = \text{flip}(D'[(i-1) \cdot b : i \cdot b]), \quad \text{for } i = 1, 2, \ldots, m$$

$$D' = \bigcup_{i=1}^{m} B_i$$

Step # 10: Apply Caesar Substitution Cipher

Apply Caesar substitution cipher by making a shifted characters mapping of n number character shift calculated by half of the length of derived key K_{der}.

$$A = \text{punctuation} \cup \text{lowercase} \cup \text{digits} \cup \text{uppercase}$$

$$s = \lfloor |K_{\text{der}}|/2 \rfloor$$

$$A' = A[s \mod |A| :] \cup A[: s \mod |A|]$$

$$D_{\text{enc}} = \bigcup_{i=1}^{|D'|} \text{map}(D'[i], A, A')$$

Returns:

$$D_{\text{enc}} \rightarrow \text{Encrypted Data}$$

8.2 Algorithm for Decryption Process

Input:

$$K_{\text{user}} \rightarrow \text{User provided key}$$

$$S_{\text{user}} \rightarrow \text{User provided salt}$$

$$D_{\text{enc}} \rightarrow \text{User provided encrypted data}$$

Output:

$$D_{\text{dec}} \rightarrow \text{Decrypted data}$$

Step # 1: Generate a Secure Password-Based Derived Key

Generate a secure password-based derived key by user provided key K_{user} and salt S_{user} using PBKDF2 SHA-256 algorithm.

$$K_{\text{der}} = PBKDF2\_SHA256(K_{\text{user}}, S_{\text{user}}, 1000, 256) + S_{\text{user}}$$

where,

$$1000 \rightarrow \text{iterations}$$

$$256 \rightarrow \text{key length}$$

Step # 2: Generate a Space Character Placeholder

Generate a space character placeholder using the derived key K_{der} and by taking the mid-point of all the numbers in the derived key K_{der}.

$$N = \text{extract\_digits}(K_{\text{der}})$$

$$m = \begin{cases} 2, & \text{if } \lfloor |N|/2 \rfloor = 0 \\ N_{\lfloor |N|/2 \rfloor}, & \text{otherwise} \end{cases}$$

$$SP = K_{\text{der}}\left[: \frac{|K_{\text{der}}|}{m} \right]$$

Step # 3: Remove Newlines and Tabs

Clean newlines and tabs from the provided data D_{enc}.

$$D = \text{Remove}(D_{\text{enc}}, [\text{newlines}, \text{tabs}])$$

Step # 4: Apply Caesar Substitution Cipher

Apply Caesar substitution cipher by making a shifted characters mapping of n number character shift calculated by half of the length of derived key K_{der}.

$$A = \text{punctuation} \cup \text{lowercase} \cup \text{digits} \cup \text{uppercase}$$

$$s = \lfloor |K_{\text{der}}|/2 \rfloor$$

$$A' = A[s \mod |A| :] \cup A[: s \mod |A|]$$

$$D' = \bigcup_{i=1}^{|D|} \text{map}(D[i], A, A')$$

Step # 5: Apply Block Cipher Approach

By using block cipher approach, make blocks of n length where n will be calculated on the basis of mid-point of all the numbers M in the derived key K_{der}.

$$N = \text{extract\_digits}(K_{\text{der}})$$

$$m = \begin{cases} 2, & \text{if } \lfloor |N|/2 \rfloor = 0 \\ N_{\lfloor |N|/2 \rfloor}, & \text{otherwise} \end{cases}$$

$$b = \lfloor |D'|/m \rfloor$$

Step # 6: Flip the Blocks Horizontally

Flip the blocks horizontally and make a new data sequence by concatenating the flipped blocks together.

$$B_i = \text{flip}(D'[(i-1) \cdot b : i \cdot b]), \quad \text{for } i = 1, 2, \ldots, m$$

$$D' = \bigcup_{i=1}^{m} B_i$$

Step # 7: Apply Rail Fence Transposition Cipher

Using rail fence transposition cipher, make two columns of depth of data length and mapping data in rows in a zig-zag pattern.

$$M_{r,c} = D'[i], \quad \text{where}$$

$$r = \begin{cases} 0, & \text{if } i = 0 \\ r+1, & \text{if dir\_down} = 1 \text{ and } r < k-1 \\ r-1, & \text{if dir\_down} = 0 \text{ and } r > 0 \end{cases}$$

$$c = i$$

$$k = \lfloor |D'|/2 \rfloor$$

$$D' = \bigcup_{r=0}^{k-1} \bigcup_{c=0}^{|D'|-1} M_{r,c}$$

Step # 8: Reapply Block Cipher Approach

Again, by using block cipher approach, make blocks of n length where n will be calculated on the basis of mid-point of all the numbers M in the derived key K_{der}.

$$N = \text{extract\_digits}(K_{\text{der}})$$

$$m = \begin{cases} 2, & \text{if } \lfloor |N|/2 \rfloor = 0 \\ N_{\lfloor |N|/2 \rfloor}, & \text{otherwise} \end{cases}$$

$$b = \lfloor |D'|/m \rfloor$$

Step # 9: Flip the Blocks Horizontally Again

Again, flip the blocks horizontally and make a new data sequence by concatenating the flipped blocks together.

$$B_i = \text{flip}(D'[(i-1) \cdot b : i \cdot b]), \quad \text{for } i = 1, 2, \ldots, m$$

$$D' = \bigcup_{i=1}^{m} B_i$$

Step # 10: Replace Spaces with the Generated Space Character Placeholder

Replace all the spaces with the generated space character placeholder *SP* from the provided data D_{enc}.

$$D_{\text{dec}} = \text{Replace}(D_{\text{enc}}, \text{spaces}, SP)$$

Returns:

$$D_{\text{dec}} \rightarrow \text{Decrypted Data}$$

9 Features of Proposed Algorithm

The designed cryptographic algorithm is based on fundamental techniques to perform encryption and decryption operations by implementing various cryptographic techniques including substitution, block-based, and transposition techniques. The algorithm provides features including efficiency, accuracy, and easy-application especially for low resource environments.

9.1 Modular Architecture

The design follows a multi-layered pipeline having multiple cryptographic approaches including dynamic key generation mechanism using PBKDF2 with SHA-256 and salt, custom block-based shift-reversing mechanism, transposition by rail fence cipher, and a customized substitution using the caesar cipher approach. Where other algorithms that follow strict transformations, the proposed algorithm provides better performance and security for different applications.

9.2 Computational Performance

The proposed algorithm follows an optimized approach that allows the execution of encryption and decryption processes in milliseconds, low memory and resource usage, and minimizes computational overhead, which allows avoiding heavy processing and executions. The conducted experiments highlight that algorithm results in lower CPU and memory usage, and faster encryption and decryption operations.

9.3 Security Features

To make cryptographic algorithms strong and robust in terms of security, it is necessary to incorporate fundamental cryptographic confusion and diffusion properties that ensure secure cryptographic design. By introducing randomized substitution and dynamic transposition, the algorithm follows a complex transformation process, making it resistant to attacks including brute-force, frequency analysis, and known plain-text attacks.

9.4 Integration with Resource Limited Environments

The designed algorithm operates accurately on resource-limited devices where memory and processing power are limited. It ignores large memory allocations, high processing calls, and dependencies on heavy libraries. In scenarios where other heavyweight cryptographic algorithms are impractical due to their resource limitations, the proposed algorithm is highly effective on IoT devices and embedded systems.

10 Analysis of Experimental Results

To find the experimental results of the proposed symmetric key cryptographic algorithm, it is developed in Python version 3.11 and ran the analysis on an Intel Core i5-8350U 1.70GHz with 16.0 GB of RAM (DDR4) system.

Further, a set of input values was used during the experimental analysis. The table below represents the selected data including key, salt, and plain text values. It is important to note that the length of key and salt in the proposed algorithm can be configured according to the requirements.

Table 1. Experimental Test Data

Type	Size	Test Data
Key	16 bytes	automata12345678
Salt	6 bytes	pepper
Plain-text	30 bytes	Hello! I am Mubbashir Ahmed.

10.1 Encryption Process

The encryption process took 0.001992 seconds (execution time depends on the system specifications and key, salt, and data length) to completely encrypt the data **Hello! I am Mubbashir Ahmed.** to its encrypted form `(!#"!"EBB)E;DDLKGK*N&dR<9!A"A"qBC)"B!!Q#H(G&)&E9BL"!B""T@E&)(B#!!#B(`

Figure 5. Code and Output of Encryption

10.2 Decryption Process

The decryption process took 0.001454 seconds (execution time depends on the system specifications and key, salt, and data length) to completely decrypt the data `(!#"!"EBB)E;DDLKGK*N&dR<9!A"A"qBC)"B!!Q#H(G&)&E9BL"!B""T@E&)(B#!!#B(` to its original form **Hello! I am Mubbashir Ahmed.**

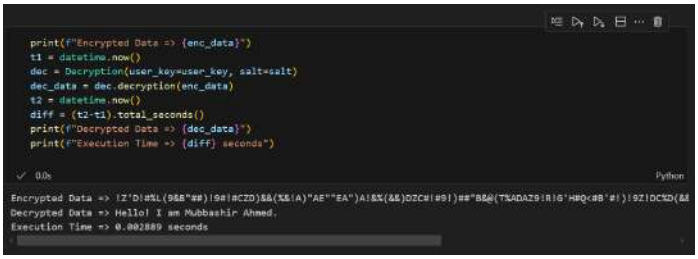

Figure 6. Code and Output of Decryption

10.3 Performance Comparison of Proposed Algorithm with Existing Algorithms

The table below represents a comparative analysis of resources including CPU and RAM usage of the proposed algorithm, and existing algorithms including AES, DES, 3DES, Blowfish. The aim of the comparison is to analyze the efficiency of computational resources usage for each algorithm. The experimental results highlight that every algorithm maintained the least RAM usage and a small difference in CPU usage for encryption and decryption processes. To conclude, the proposed algorithm shows the lowest average CPU usage focusing on its lightweight applicability for resource-limited environments.

Table 2. Resource-Based Comparison

Algorithm	CPU Usage (%)	RAM Usage (MB)
Proposed Algorithm	0.60 %	263 MB
AES	0.64 %	263 MB
DES	0.62 %	263 MB
3DES	0.76 %	263 MB
Blowfish	0.72 %	263 MB

The table below provides a comparative analysis of the resistance of the cryptographic algorithm, including proposed algorithm and AES, DES, 3DES, and Blowfish against brute-force attacks, frequency analysis, and known plain-text attacks. The proposed algorithm and AES both demonstrate strong resistance against analyzed attacks. Where, the proposed algorithm shows great resistance against analyzed attacks compared to DES which is vulnerable due to its limited key size, and 3DES which provides partial resistance. This comparative analysis shows that the proposed algorithm is effective and provides relative strength in terms of security.

Table 3. Resistance to Cryptographic Attacks

Algorithm	Brute-Force	Frequency Analysis	Known Plain-text
Proposed Algorithm	Yes	Yes	Yes
AES	Yes	Yes	Yes
DES	No	No	No
3DES	—	—	—
Blowfish	Yes	Yes	Yes

11 Cryptanalysis Testing of Proposed Algorithm

To analyze the robustness and reliability of the proposed algorithm we follow standard analysis approaches for encryption systems, various cryptanalysis tests were experimented including brute-force,

frequency analysis, and differential cryptanalysis. The aim of this testing phase is to evaluate the proposed algorithm's ability to defend itself from different attacks, and secure data transmission, specifically in lightweight and resource-limited environments.

For cryptanalysis testing, a set of input values was used. The table below represents the selected data including key, salt, and plain text values.

Table 4. Cryptanalysis Testing Data

Type	Size	Test Data
Key	16 bytes	automata12345678
Salt	6 bytes	pepper
Plain-text	30 bytes	Hello! I am Mubbashir Ahmed.

The evaluations of the cryptanalysis testing including brute-force attack, frequency analysis, differential cryptanalysis, and known plain-text attack authenticate that the proposed algorithm performs accurately against conventional and modern attack techniques.

11.1 Brute-Force Attack

To analyze the security of the proposed algorithm against brute-force attacks, a test was performed that targeted the algorithm's key derivation. The purpose of the test was to determine the original key without any knowledge. Since the algorithm uses PBKDF2 with SHA-256 for key derivation, brute-force attempts require more computational effort per key guess, making the process more time consuming.

To perform an attack, the brute-force application was programmed to try all possible combinations, including lowercase and uppercase characters, and digits. Since PBKDF2 is used to create a SHA-256 based key that includes lowercase and uppercase characters, and digits, the brute-force attack will perform approximately $94^{256} \approx 2.01 \times 10^{505}$ possible combinations, which is not possible due to the high computational cost and time.

The evaluation highlights that brute-force only a small part of the key becomes computationally high. This highlights the security of the algorithm against brute-force attacks, especially when using robust keys and salts.

11.2 Frequency Analysis

To evaluate the robustness of the proposed algorithm, a frequency analysis randomness test was performed. By these tests, we can analyze the algorithm's ability to follow natural patterns when performing encryption.

To perform analysis, the frequency distribution of characters of plain-text and encrypted text was analyzed. Afterwards, the analysis shows that the plain-text displays frequency peaks at certain characters including spaces and vowels.

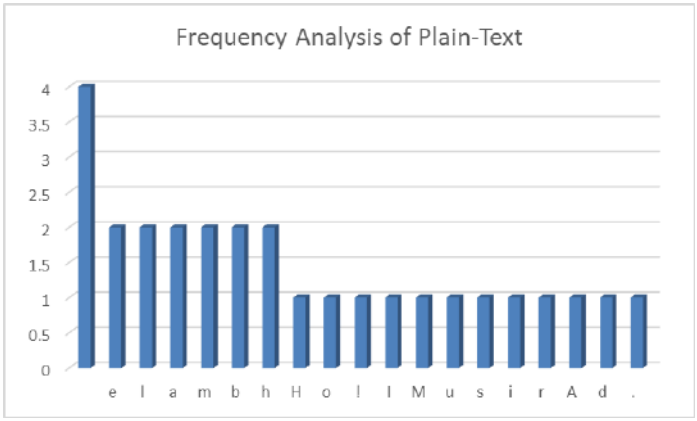

Figure 7. Frequency Analysis of Plain-Text

Further, the encrypted text shows irregular distributions of characters, and frequency peaks for frequently occurring characters including characters) and ', due to the dynamic space placeholder (generated by derived key) used in the proposed algorithm.

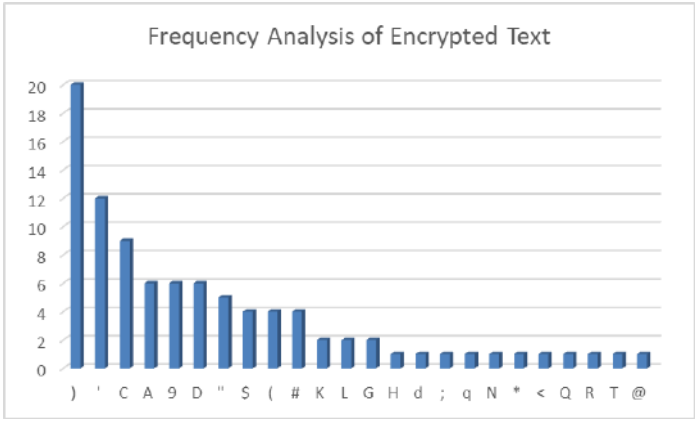

Figure 8. Frequency Analysis of Encrypted Text

11.3 Known Plain-text Attack

A type of cryptanalysis is referred as known plain-text attack, where the attacker has obtained plain-text and encrypted text. The aim of this attack is to recognize the patterns and find out the key used for data encryption or decryption processes.

A known plain-text attacks simulation has been performed to analyze the resistance of the proposed algorithm. The test involves encrypting the same plain-text but using different salts, including pepper, sodium, and vinegar.

The evaluated results show major differences between the encrypted texts. The results analyzed have hamming distances from 196 bits to 229 bits and percentage difference of 26.63% to 42.10%. These results highlight that the encrypted texts were very much different.

Table 5. Effects of Different Salts on Encrypted Text

Salts Comparison	Hamming Distance	% Difference
pepper vs sodium	196 bits	26.63%
pepper vs vinegar	212 bits	38.97%
sodium vs vinegar	229 bits	42.10%

12 Conclusion

Cryptography is the combination of encryption and decryption. Encryption is the process of converting information (plain data) into an unreadable format (cipher data) using a key. Decryption is the reverse of encryption, and is the process of converting unreadable data (cipher data) into information (plain data).

Cryptography is used for several purposes including data security which includes the data accessibility for authorized users and inaccessibility for unauthorized users (usually attackers or hackers) and data privacy which is connected with the concepts of encryption and decryption using a key.

This paper has given the essence of cryptography, basic terminologies used in the field of cryptography, process of cryptography which includes encryption and decryption, pillars of cryptography, and types of cryptography.

Moreover, this paper also proposed a new efficient and lightweight cryptographic algorithm using a symmetric key approach that runs on less computational resources compared to modern cryptographic algorithms such as AES, DES, 3DES, or Blowfish. Furthermore, the algorithm is simple and easy to implement in real-world projects for cryptographic processes.

Author Contribution

Mubbashir Ahmed: Conceptualization, Data curation, Methodology, Writing—Original draft preparation, Writing—Original draft preparation, reviewing, editing, Methodology, software, investigation, Methodology, software, investigation, Supervision and reviewing.

Compliance with Ethical Standards

It is declared that all authors don't have any conflict of interest. It is also declared that this article does not contain any studies with human participants or animals performed by any of the authors. Furthermore, informed consent was obtained from all individual participants included in the study.

References

[1] J. Katz and Y. Lindell, *Introduction to modern cryptography: principles and protocols*. Chapman and hall/CRC, 2007.

[2] N. Sharma *et al.*, "A review of information security using cryptography technique.," *International Journal of Advanced Research in Computer Science*, vol. 8, no. 4, 2017.

[3] A. Al-Sabaawi, "Cryptanalysis of classic ciphers: Methods implementation survey," in *2021 International Conference on Intelligent Technologies (CONIT)*, pp. 1–6, IEEE, 2021.

[4] B. Sankhyan, A. Baliyan, and A. Kumar, "Review on symmetric and asymmetric cryptography," *International Journal for Research in Applied Science and Engineering Technology*, vol. 12, pp. 2934–2940, 2024.

[5] R. Kumar and D. Yadav, "Cryptographic algorithms: Current status and future directions," *European Chemical Bulletin*, 09 2023.

[6] A. S. Tushar and A. Mishra, "Cryptographic algorithm for enhancing data security: A theoretical approach," *International Journal of Engineering Research & Technology*, vol. 10, no. 03, pp. 274–277, 2021.

[7] K. Sasikala, "Comparative study of cryptographic algorithms," *International Journal of Engineering Research Technology*, vol. 9, 11 2020.

[8] S. Naser, "Cryptography: from the ancient history to now, it's applications and a new complete numerical model," *International journal of mathematics and statistics studies*, vol. 9, no. 3, pp. 11–30, 2021.

[9] Y. Salami, V. Khajevand, and E. Zeinali, "Cryptographic algorithms: A review of the literature, weaknesses and open challenges," *J. Comput. Robot*, vol. 16, no. 2, pp. 46–56, 2023.

[10] S. Banerjee, "Exploring cryptographic algorithms: Techniques, applications, and innovations," *International Journal of Advanced Research in Science, Communication and Technology*, pp. 607–620, 2024.

[11] M. A. Panhwar, S. A. Khuhro, G. Panhwar, and K. A. Memon, "Saca: A study of symmetric and asymmetric cryptographic algorithms," *International journal of computer science and network security*, vol. 19, no. 1, pp. 48–55, 2019.

[12] U. Thirupalu and E. K. Reddy, "Performance analysis of cryptographic algorithms in the information security," *Int. J. Eng. Res. Technol*, vol. 8, no. 2, pp. 1–6, 2020.

[13] A. F. Iuorio and A. Visconti, "Understanding optimizations and measuring performances of pbkdf2," in *2nd International Conference on Wireless Intelligent and Distributed Environment for Communication: WIDECOM 2019*, pp. 101–114, Springer, 2019.

[14] B. Singh and D. Jesi, "Safe vault: A password manager," *International Journal of Engineering Applied Sciences and Technology*, vol. 6, pp. 93–97, 04 2022.

[15] N. A. A. Mustafa, "Analysis attackers' methods with hashing secure password using csprng and pbkdf2," *Wasit Journal of Engineering Sciences*, vol. 12, no. 2, pp. 60–70, 2024.

[16] C. Lee and H. Lee, "A password stretching method using user specific salts," in *Proceedings of the 16th international conference on World Wide Web*, pp. 1215–1216, 2007.

[17] E. Agrawal and P. R. Pal, "A secure and fast approach for encryption and decryption of message communication," *Int. J. Eng. Sci*, vol. 11481, 2017.

[18] M. S. Turan, E. Barker, W. Burr, and L. Chen, "Recommendation for password-based key derivation," *NIST special publication*, vol. 800, p. 132, 2010.

[19] A. Visconti, S. Bossi, H. Ragab, and A. Calò, "On the weaknesses of pbkdf2," in *Cryptology and Network Security: 14th International Conference, CANS 2015, Marrakesh, Morocco, December 10-12, 2015, Proceedings 14*, pp. 119–126, Springer, 2015.

[20] M. Dell'Amico, P. Michiardi, and Y. Roudier, "Password strength: An empirical analysis," in *2010 Proceedings IEEE INFOCOM*, pp. 1–9, IEEE, 2010.

[21] K. Chanda, "Password security: an analysis of password strengths and vulnerabilities," *International Journal of Computer Network and Information Security*, vol. 8, no. 7, p. 23, 2016.

[22] A. Joshi, M. Wazid, and R. Goudar, "An efficient cryptographic scheme for text message protection against brute force and cryptanalytic attacks," *Procedia Computer Science*, vol. 48, pp. 360–366, 2015.

[23] A. Nadeem and M. Y. Javed, "A performance comparison of data encryption algorithms," in *2005 international Conference on information and communication technologies*, pp. 84–89, IEEE, 2005.

[24] A. Gupta and N. K. Walia, "Cryptography algorithms: a review," *International Journal of Engineering Development and Research*, vol. 2, no. 2, pp. 1667–1672, 2014.

[25] Y. Zhong and J. Gu, "Lightweight block ciphers for resource-constrained environments: A comprehensive survey," *Future Generation Computer Systems*, 2024.

[26] Y. Pan, "The scope of application of letter frequency analysis in substitution cipher," in *Journal of Physics: Conference Series*, vol. 2386, p. 012015, IOP Publishing, 2022.

[27] R. Verma, A. Kumari, A. Anand, and V. S. S. Yadavalli, "Revisiting shift cipher technique for amplified data security," *Journal of Computational and Cognitive Engineering*, vol. 3, no. 1, pp. 8–14, 2024.