Journal of Theoretical and Applied Information Technology
15st July 2026. Vol.104. No.13

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

AN ENHANCED BLOWFISH ALGORITHM WITH DYNAMIC S-BOX EXCHANGE AND MERGE PATTERNS

Dr.S. SWEETLIN SUSILABAI[1], Dr. J. JEBAMALAR TAMILSELVI[2], Dr. K. SUTHA[3], Dr SURYA SUSAN THOMAS[4], Dr. KAYATHRI K[5]

[1,2,3,] Department of Cyber Security, Faculty of Science and Humanities,
SRM Institute of Science and Technology, Ramapuram,Tamilnadu, India.
[4]Assistant Professor, Department of Computer Science with Data Science,
Patrician College of Arts and Science, Adyar, Chennai, Tamilnadu, India
[5]Assistant Professor, Department of Computer Science and Software Applications
Agurchand Manmull Jain College, Meenambakkam,Chennai, Tamil Nadu, India.

E-mail: [1]sweetlis1@srmist.edu.in ,suryasusan@patriciancollege.ac.in, kayathri.k@amjaincollege.edu.in

ABSTRACT

The rapid expansion of digital data transmission and the rise in cyber threats have made information security a crucial component of the modern communication systems. Sensitive data integrity, confidentiality, and authenticity are all greatly enhanced by encryption techniques. The Blowfish algorithm is a popular symmetric key cryptography algorithm due to its ease of use, adaptability, and computational effectiveness. However, the majority of current Blowfish-based methods rely on static S-box structures, which lessen algorithmic unpredictability and increase the algorithm's susceptibility to sophisticated cryptanalytic assaults like differential and linear cryptanalysis. Few studies have concentrated on dynamically altering S-box exchange techniques throughout both key allocation and encryption phases while maintaining computing efficiency, despite the fact that numerous improvements have been suggested in the literature.The primary objective of this research is to use the proposed - Exchange and Merge Patterns (EMPAT) method to create an improved Blowfish encryption model. To increase confusion, diffusion, randomness, and resistance to cryptanalytic attacks, the study incorporates dynamic S-box exchange operations and inter-bit shifting methods during encryption rounds and key scheduling.This paper proposes an improved Blowfish method based on the Exchange and Merge Patterns (EMPAT) approach to close this research gap. The proposed approach increases non-linearity, diffusion, and key dependency during encryption by introducing dynamic S-box exchange patterns and inter-bit shifting operations utilizing four different exchange combinations. The recommended EMPAT approach continuously rearranges S-box operations, which improves unpredictability and resilience against cryptanalytic assaults without significantly increasing computing overhead, in contrast to traditional Blowfish implementations.By offering a dynamic S-box exchange framework that improves cryptographic robustness while preserving the speed and efficiency features of the original Blowfish encryption, the study adds new information. The results verify that the EMPAT method provides a smaller and more secure encryption mechanism appropriate for modern secure communication applications.

Keywords - *Cryptography, Cipher Text, Blowfish Encryption algorithm and Data Security.*

1. INTRODUCTION

Data security is a crucial problem in today's world. We exchange a lot of personal information online in our daily lives, including messages, pictures, and personal information. This is why it is so important to keep that information secure. Keeping this information safe from unauthorized access, such as hackers or thieves, is known as data security. In our internet-driven world, it's crucial. Maintaining the security of our data requires knowing the history of security, identifying the risks we face, and creating new techniques to safeguard our data. Throughout the 21st decade and afterward, information security has several uses. One method to stop unwanted access to private information is encryption. Symmetric key encryption techniques are widely used because they are significantly faster than asymmetric key algorithms.

Symmetric key encryption techniques are widely used because they are significantly faster than asymmetric key algorithms. Privacy and security are now major concerns in business, industry, and administration due to the growing

prevalence of digital communication and electronic data transfer. These days, security is the top priority for any correspondence between a sender and a recipient. Any security breaches during communication could result in serious losses for both the sender and the recipient. Modern cryptography offers many essential techniques for data protection and information security.

Communication and information sharing have been altered by the internet. We can readily connect with others thanks to technology, but there are risks involved as well. There are numerous ways for malicious actors to attempt to steal or abuse our data via the internet. Because of this, security has grown to be a major worry. We must develop new strategies to safeguard our data since emerging forms of threats are making traditional security measures less effective. Creating new technology or enhancing current ones could be necessary to keep ahead of possible dangers.

Three key concepts can be used to summarize the primary objectives of data security: availability, integrity, and secrecy.

1.1.Cryptography

Our information is protected by the fascinating and significant field of cryptography [1]. It's the study of data security. Consider it a hidden code that protects your messages from unauthorized access. In terms of organization security, it is an essential component. Like a shield, it safeguards private data, fosters confidence, controls resources, and ensures the safety of online financial operations. This is crucial for protecting our data, particularly when we share it online. There are several ways to safeguard our data, such as video encryption, and it employs sophisticated mathematics to generate secure codes. One of the numerous tools used to guarantee that our information is protected and kept private is the Blowfish algorithm. It aids in resolving significant issues associated with the following:

Confidentiality: Confidentiality protects sensitive information from unauthorized access and ensures that only authorized individuals or systems can access it. The goal is to keep private information from being viewed, accessed, or used by unauthorized people.

Integrity: Integrity ensures that data is not altered during transmission or storage. The integrity of the data is compromised by any changes made to it. Data integrity is jeopardized when it is corrupted, which could lead to errors or malicious changes.

Availability: Users may access and utilize the Internet, its services, and data whenever they need to thanks to availability. When networks are unavailable, they can disrupt regular operations, which can lead to serious issues for consumers and organizations who rely on them.

Transmitting sensitive data securely over potentially insecure networks, such as the internet, is the primary objective of cryptography. It's similar to mailing a letter in a locked box that can only be opened by the key holder. The sender only gives the intended receiver the key to unlock the communication in order to protect it. This prevents others from reading it. Math and computer science play a major role in modern cryptography. Anyone attempting to crack the code will find it challenging because it employs intricate mathematical issues that are challenging to solve. Breaking into these frameworks is quite challenging in practice, even though it is feasible in theory.

1.1.1 Types of Cryptography

Two primary categories of cryptography methods exist [2][3][4]:

i. ***Cryptography using symmetric keys:*** Data is encrypted and decrypt-ed using the same keys in symmetric-key methods. This type of cryptography is crucial for data protection because the same key is used for both encryption and decryption.

ii. ***Cryptography Asymmetric Key Cryptography****:* Two keys are used in asymmetric key cryptography: one for encoding and another for decoding. Symmetric key algorithms use less processing resources than asymmetric key algorithms. But in reality, symmetric key algorithms are far faster than asymmetric key algorithms. Asymmetric algorithms, often known as public-key algorithms, require a key that is at least 3,000 bits in size in order to offer the same degree of security as a 128-bit symmetric method. RSA, ElGamal, and Elliptic Curve Cryptography (ECC) are well-known instances of asymmetric key cryptography. Public key cryptography is another name for asymmetric key cryptography. A secure first key exchange between the sender and the recipient is not necessary for this method to work. To make it easy for the recipient to produce the private and public keys and use the private key to decrypt the message, the encryption and decryption procedure was created. Additionally, the sender can easily encrypt the data using the public key, and

anyone who knows the public key will have a very hard time deciphering the private key.

iii.

1.2 Need of the proposed method Dynamic S-Box Exchange and Merge Pattern (EMPAT)

To prevent more sophisticated cyberattacks and cryptanalytic methods, current encryption algorithms must constantly change. The standard implementation of the Blowfish algorithm uses static S-box structures, which may show predictable patterns under sophisticated attack models, despite the technique's processing efficiency and widespread use in secure communication systems. In order to enhance confusion, diffusion, and unpredictability in symmetric encryption systems, recent cryptography research highlights the necessity of dynamic key-dependent transformations. Adaptive S-box exchange systems that can simultaneously improve encryption security and maintain processing efficiency, however, have received little attention in the literature to yet.The development of an improved Blowfish model that can boost protection against differential analyzing and linear cryptanalysis while avoiding undue processing complexity is therefore imperative. By using dynamic exchange and merging patterns during key allocation and encryption rounds, the suggested EMPAT method overcomes this restriction. By enhancing unpredictability and key reliance, this change increases the encrypted data's resistance to contemporary cryptographic attacks. Therefore, the proposed work is important for developing secure yet lightweight encryption methods appropriate for modern data protection needs.

1.3 Novelty of EMPAT

This study is novel because it introduces a dynamic EMPAT architecture that uses four alternative exchange patterns to continuously rearrange S-box operations. The proposed method uses adaptive S-box modifications to increase encryption unpredictability and key reliance, in contrast to current Blowfish enhancement solutions that primarily rely on static substitution structures. This method adds a novel lightweight cryptographic enhancement technique that maintains computational clarity while simultaneously enhancing security.

1.4 Scope of the study

The aim of this study is to develop an enhanced Blowfish encryption model using the suggested Exchange and Merge Patterns (EMPAT) approach. The work uses dynamic S-box exchange operations and inter-bit shifting techniques during encryption rounds and key scheduling to increase confusion, diffusion, unpredictability, and resistance against cryptanalytic attacks.

1.5 Delimitation of the Study

The modification of Blowfish S-box operations is the exclusive focus of this study; it does not address:

- Algorithms for asymmetric cryptography
- Techniques for quantum cryptography
- Optimization of encryption at the hardware level
- IoT-specific frameworks for lightweight encryption
- Attack prevention techniques at the network layer
- Deployment in real time within dispersed cloud infrastructures

Only software-level cryptographic enhancement utilizing dynamic S-box exchange patterns is the focus of this study.

2. LITERATURE SURVEY

Pradeep Mishra and Monika Agarwal [1] suggested a novel method for each time a fresh random number is generated and this leads to variations in how the F function is applied during each round. They have proposed and implemented a novel technique to further improve the current algorithm and achieve better outcomes in terms of measures like encryption time, decryption time, and throughput. The blowfish encryption technique, which generates encrypted text for the same input, was updated and used by them. This is shorter and quicker than the original method and greatly improves the technique's security. They concluded that the statistics clearly demonstrate that the encryption and decryption speeds of the modified blowfish algorithm are almost half that of the original technique. They suggested that case 4 is the most safest.

Saikumar Manku and K. Vasanth [2] presented a single blowfish round and a proposed blowfish algorithm that reduces algorithm rounds. Every round in this suggested system is launched with a fresh modification. The approach was developed to have two s-boxes linked to XOR, similar to

some other two s-boxes linked to XOR. The key plain text is then obtained from the two XORs combined. They found that the updated method provides excellent security, preventing data hacking by any attacks between the sender and the recipient. Poonia et al. [3] used a number of parameters, such as output file size, correlation coefficients, encryption quality, and key sensitivity test, to assess the efficiency of the modified Blowfish algorithm in four distinct cases. In each of those cases, they demonstrated that they had improved the Original Blowfish methodology in some way. The results of all scenarios and the tests stated above show that the original Blowfish technique is more compact and reliable than the previous one due to the security of the modified approach under different conditions. They used a number of parameters, including output file size, correlation coefficients, encryption quality, and key sensitivity test, to evaluate the performance of the modified Blowfish algorithm in four different scenarios. In each of those instances, they have improved the Original Blowfish technique to some extent. They demonstrated that the results of every test listed above generally indicate that the primary Blowfish algorithm is more secure and efficient than the original since the modified method is secure in a variety of situations.

Quilala et al [7], they examined and improved the encryption using the modified Blowfish algorithm (MBA). The modification kept the MBA's initial structure, technique, and use of two S-boxes, despite the introduction of two derivation processes in the f-function, which was originally positioned to avoid symmetry. Time efficiency and the avalanche effect were used to examine the performance of the derivation scenario. After comparing the MBA's first and second derivation processes, it was found that the second one increased security by 5.47% by enhancing the avalanche effect. Furthermore, the results showed that the second update took 38.34% less time to decrypt and 39.48% less time to encrypt. Because the shift rotation was positioned differently in the modified Blowfish, they proved that the first derivation example took longer to complete.

Harinath, Depavath, et al. [18], they have introduced the Blowfish algorithm combines cryptography and steganography to provide a safe and private communication system. They make it feasible to use the modified blowfish method to provide another level of protection to the data being sent. This work demonstrates how they provided an efficient data encryption technique to ensure data confidentiality.

3. METHODOLOGY

3.1 Blowfish Algorithm

Brute Schneier developed a symmetrical block cipher method called Blowfish [5][6][7][8] in December 1993. The replacement to DES or IDEA is Blowfish. Blowfish is among the most widely used feistel network ciphers. The Blowfish algorithm's variable key length ranges from 42 to 448 bits, while its block size is 64 bits. This kind of encryption is symmetric, which means that messages are locked and unlocked using the same key. Blowfish is a well-liked option for data security because of its reputation for speed and effectiveness. It is a symmetric key encryption method that is simple to use and effective. The Blowfish Algorithm [9][10][11] produces a number of subkey arrays totaling 4168 bytes, or 1042 32-bit values.

This sixteen-round feistel method provides exceptionally vast key-dependent S-boxes and supports 16 iterations. Data, a key-dependent substitution, and a key-associated permutation are used to construct each round. 32-bit words and XOR operations both employ additions. A P-array and four 32-bit S-boxes are present. Each S-box contains 256 entries, and the P-array contains 18 32-bit subkeys. The input is a 64-bit data bit. Each cycle consists of a key-and-data-dependent substitution and a key-independent randomization. Each step combines to 32-bit words and uses XOR. An example of the alternate key generating process may be found below.

i. Start by initializing the four S boxes and the P array.

ii. Each of the four S units in the P array correspond to the hexadecimal representation of Pi.

iii. Use the K array to perform bitwise XOR operations on the P-array, and re-use the necessary words from the K array with its key bits (P1 XOR utilizes the first 32-bit value of the key, while P2 XOR uses the second 32 bits).

iv. Encrypt the block of 64 bits. Use the existing P and S arrays to secure the all-zeros using the previously described encryption procedure.

v. Use the encryption output in place of p1 and p2. The new outcomes are now referred to as p1 and p2.

vi. Use the P and S boxes that are already there to encrypt the outcome of step 3, then swap out p3 and p4 for the encrypted text that results. The modified output is now referred to as P3 and P4.

The following is how the only additional phases for each round are carried out:

1. Slice each block in half.

2. Swapping the right segment for the left.

3. Apply "f" on the right segment and XOR the outcome with the second part to build the right half.

4. To access the previous rounds, the parameter "f" does not need to be inverted.

The Blowfish encryption method:

1. The original text (X)'s 32-bit segments are separated into LEnc0 and REnct0.Let's take the variables LEnci and REnci from round I and submit them to the data's left as well as right halves.

2. For values of j between 1 and 16,

RightEncj = LeftEncj-1 XOR Pj

F(RightEncj) XOR RightEncj-1 = LeftEncj

Trade LeftEnci and RightEnci.

3. Reverse the previous trade by exchanging RightEncj and LeftEncj.

4. RightEncj XOR P17 = RightEncj

5. LeftEncj = P18 XOR LeftEncj

LeftEnc17 and RightEnc17 appear in the final encrypted text that is produced.

6. Integrate RightEncj and LeftEncj

The Blowfish algorithm[12][13][14] operates as follows.

First, divide the 32-bit LeftEnc into the four 8-bit segments (a1, b1, c1, and d1). Next, use the formula that follows.

F (LeftEnc) is calculated using the formula

{(Firstbox1 [a1] + Secondbox2 [b1]) Thirdbox3 [c1]} + Fourthbox4 [d1])}, where the Firstbox1, Secondbox2, Thirdbox3, and Fourthbox4 are the four substitution boxes, and + indicates addition modulo 2^32 and absolute OR.The blowfish algorithm's S-Box operations are illustrated in Figure 3.1 below.

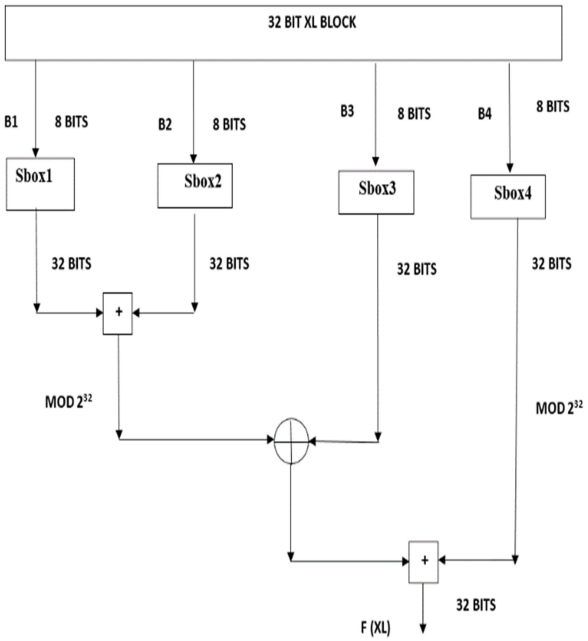

Figure. 3.1 Blowfish Algorithm's S-Box Operations.

3.4 Proposed Dynamic S-Box Exchange and Merge Pattern (EMPAT)

The Blowfish method, which we have named the blowfish algorithm [15][16], has been enhanced by using the prior version's implementation. Exchange and key-bit merging were used to improve the Blowfish approach and change the structure of SBoxes. The 64-bits of information is obtained as key in by the right and left sides of the function, and then they are divided equally into 32 bits. In addition to being EX-ORs with the designated key, the 32 bits on the left side are further divided into 8-bit inputs for each of the four types of substitution function boxes, or S-Box. The Exchange and Merge Pattern of Sbox Algorithm (EMPAT) is driven by Key Bit Exchanging and uses the function block output as input.

EMPATModes employ the Exchanging & Merge Key Bit method. Using this technique, the primary message can be split up into 64-bit pieces. Each block's 64 bits of data are then split into two 32-bit parts. XL and XR are the names of the segments on the left and right, respectively.

These subblocks are then divided into eight-bit subpatterns, EMPAT1, EMPAT2, EMPAT3, and

Journal of Theoretical and Applied Information Technology
15st July 2026. Vol.104. No.13

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

EMPAT4. EMPATMode uses the Exchange & Merge Pattern of Sbox, and these subpatterns are Sboxes. The primary message can be divided into 64-bit blocks using this method. The 64 bits of data in each block are then divided into two 32-bit halves. The subblocks on the left and right are called SLeft and SRight, respectively.

These subblocks are then divided into eight-bit subpatterns called Sbox1, Sbox2, Sbox3, and Sbox4. SBeven and SBodd are these subpatterns. The following is how EMPATOe is determined in the Exchange & Merge Key Bit Algorithm:

1) Get the even segment from SBeven and the odd segment from SBodd.

2) To create EMPATOe, swap these pieces in the same order.

3) SBox operations use this EMPATOe instead of the original 8 bit.

The Blowfish algorithm's Exchanging and Merge pattern (EMPAT) can be divided into four distinct scenarios. The following is a description of it:

3.2.1 EMPAT$_1$:

The dynamic bit representation of proposed *EMPAT*$_1$ are shown as following **Table 3.1:**

Table 3.1. EMPAT$_1$

EMPAT$_1$				
#Bits	**1**	**2**	**3**	**4**
1	0	0	S_{BOX1}	0
2	0	0	0	S_{BOX2}
3	0	S_{BOX3}	0	0
4	S_{BOX4}	0	0	0

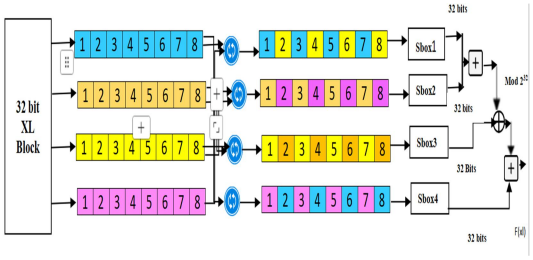

Figure 3.2 *Exchanging and Merge Pattern Algorithm EMPAT$_1$*

The following **Figure 3.2** illustrates the S-Box operations of the proposed Exchanging and Merge pattern algorithm (EMPAT$_1$).

3.2.2 EMPAT$_2$:

The dynamic bit representation of *EMPAT*$_2$ are shown as following **Table 3.2:**

Table 3.2. EMPAT$_2$

EMPAT$_2$				
#Bits	**1**	**2**	**3**	**4**
1	0	S_{BOX3}	0	0
2	0	0	0	S_{BOX2}
3	0	0	S_{BOX1}	0
4	S_{BOX4}	0	0	0

The following **Figure 3.3 i**llustrates the S-Box operations of the proposed Exchanging and Merge pattern algorithm EMPAT$_2$.

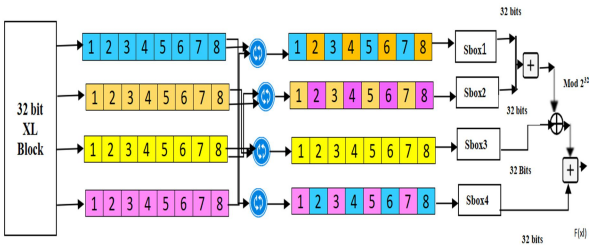

Figure 3.3 *Exchanging and Merge pattern Algorithm EMPAT$_2$*

3.2.3 EMPAT $_3$:

The dynamic bit representation of *EMPAT*$_2$ are shown as following **Table 3.3**:

Table 3.3. EMPAT$_3$

EMPAT$_3$				
#Bits	**1**	**2**	**3**	**4**
1	0	0	0	S_{BOX2}
2	0	0	S_{BOX1}	0
3	0	S_{BOX3}	0	0
4	S_{BOX4}	0	0	0

The following **Figure 3.4** illustrates the S-Box operations of the proposed Exchanging and Merge pattern algorithm EMPAT$_3$

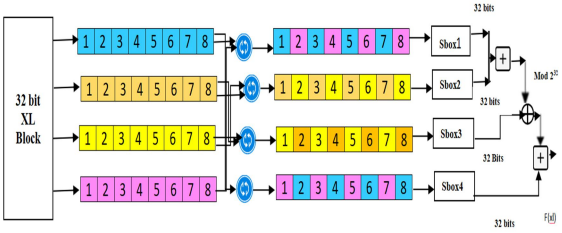

Figure 3.4 *Exchanging And Merge Pattern Algorithm* $EMPAT_3$

3.2.4 $EMKB_4$:

The dynamic bit representation of $EMPAT_2$ are shown as following **Table 3.4**:

Table 3.4. $EMPAT_4$

$EMPAT_4$				
#Bits	**1**	**2**	**3**	**4**
1	0	S_{BOX4}	0	0
2	S_{BOX3}	0	0	0
3	0	0	0	S_{BOX2}
4	0	0	S_{BOX1}	0

The following Figure 3.5 illustrates the S-Box operations of the proposed Exchanging and Merge pattern algorithm $EMPAT_4$

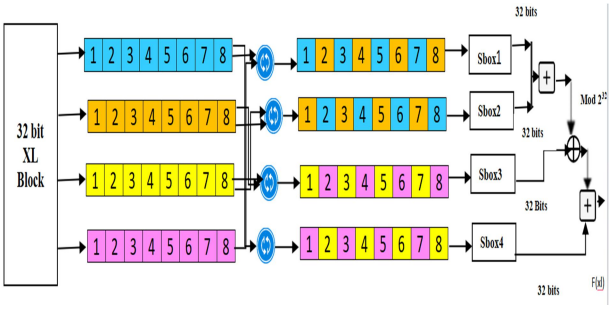

Figure 3.5 *Exchanging and Merge pattern Algorithm* $EMPAT_4$

EMPAT1 will be recognized as good when compared to the four situations mentioned above. EMKB4 has following parameters:

- Input Block: A 32-bit XL block is divided into four 8-bit segments: EMPAT1, EMPAT2, EMPAT 3, and EMPAT4.
- Bit Mixing: Each 8-bit segment undergoes some form of transformation (likely a permutation or key-mixing step).
- S-Box Substitution: Each segment is passed through an S-Box (SBox1, SBox2, SBox3, SBox4), expanding it to 32-bit outputs.
- Modulo Addition: The 32-bit outputs undergo addition operations modulo 232
- Final Combination: The outputs are combined to generate F(XL), which is likely used in a Feistel round.

The combinations of above algorithm is to develop dependable encryption techniques with a proposed EMPAT blowfish algorithm.

4. EXPERIMENTAL RESULTS AND PERFORMANCE ANALYSIS

The results Execution time throughput and encrypted file size are used to compare the characteristics of various test cases. Various input file sizes (kb) have been examined in relation to the performance requirements of the findings.

The Exchange and Merge Pattern of Sboxes (EMPAT) was implemented using the integrated environment of Eclipse 4.7.0 and Java 1.8. An Intel processor Core i5 (7th gen) running Windows 10 with a CPU speed of 2.7 GHz and 8 GB of RAM was used for all development, testing, and design work. In order to assess the security, effectiveness, and general performance of both cryptographic algorithms, this study analyzes the Blowfish algorithm with various combinations of the EMPAT algorithm. The proposed methods' throughput, execution speed, and avalanche effect have been compared with the original Blowfish algorithm's.

4.1 Performance comparison on the basis Execution Time

Only the contents of the message or text is used for the comparison. One of the performance metrics is encryption time, which is determined by measuring how long it takes to transform the original text into ciphertext during the encryption process. One of the most important metrics, decryption time, is determined by calculating how long it takes to transform ciphertext back into original text during encryption. The software is run a number of times in order to increase the timing measurement's accuracy.

Milliseconds are used to measure both the encryption and decryption times. The execution time is determined by adding the times for encryption and decryption.

Execution time=SUM (EncTime,DecTime)

Table 4.1 displays the execution time findings for the suggested EMPAT and Blowfish algorithms for a range of input data files (KB) and different conditions.

Table 4.1 Execution Time

File Size(kb)	Execution Time				
	Blowfish	***Proposed EMPAT$_1$***	***Proposed EMPAT$_2$***	***Proposed EMPAT$_3$***	***Proposed EMPAT$_4$***
49.16	15	26	25	29	20
99.814	47	61	59	60	50
149.975	51	60	58	62	53

The execution time results for the proposed EMPAT are shown in **Figure 4.1** for a variety of input file sizes (kb) and conditions.

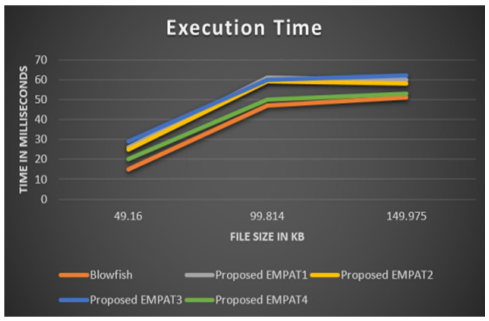

Figure 4.1 *Comparison of performance based on execution timez*

4.2 Comparison of performance based on Speed of Throughput

The total encoded message size in bytes divided by the total amount of execution time yields the execution strategy's throughput. The speed of encryption is gauged by throughput. The throughput for the proposed EMPAT are shown in **Table 4.2** for several scenarios with varied input file sizes (KB).

Table 4.2. Throughput

File Size	Throughput				
	Blowfish	***Proposed EMPAT$_1$***	***Proposed EMPAT$_2$***	***Proposed EMPAT$_3$***	***Proposed EMPAT$_4$***
49.16	13426	11378	11387	11398	11795
99.814	8699	6931	6172	6242	6542
149.975	12045	9517	9653	9799	9954

The throughput results for the proposed EMPAT are shown in **Figure 4.2** for a variety of input file sizes (KB) and conditions.

Figure 4.2 *Comparing the performance of the proposed EMPAT various cases with input file sizes (Kb)*

4.3 Performance comparison on the basis Avalanche effect

The strong diffusion approach is used to measure the avalanche effect, which indicates the strength of the cryptography. Any encryption algorithm's resistance to crypt analysis is gauged by the avalanche effect, which states that a little alteration to a single bit of secret or plain text should produce cipher messages that differ significantly by a large number of bits. The length of the original message or the key space to be searched may be reduced if the changes are small, which would greatly simplify crypt analysis.The cipher text should change significantly in response to a little alteration in the key or the plain text. Data encryption uses the Avalanche effect, which is an accurate execution of mathematical operations. The amount of bits is displayed in the **Table 4.3** below.

Table 4.3 Avalanche Effect

Input File Size (in Kb)	(% of bits changed)				
	Blowfish Algorithm	***Proposed EMPAT$_1$***	***Proposed EMPAT$_2$***	***Proposed EMPAT$_3$***	***Proposed EMPAT$_4$***
1000	45	47	49	57	60
2000	43	49	53	55	58
3000	44	47	53	54	57
4000	46	48	49	52	55
5000	37	39	41	48	55

Input File Size (in Kb)	(% of bits changed)				
	Blowfish Algorithm	*Proposed EMPAT$_1$*	*Proposed EMPAT$_2$*	*Proposed EMPAT$_3$*	*Proposed EMPAT$_4$*
6000	39	43	48	51	59
7000	38	45	52	54	57
8000	49	46	53	56	59
9000	37	34	45	47	54
10000	48	50	52	53	55
11000	54	58	58	61	65
12000	42	43	46	48	59
13000	45	44	47	52	58
14000	37	41	49	51	55
15000	46	48	50	55	62

The Figure 4.4 illustrates how many bits were altered in the input file based on the file size with avalanche effect.

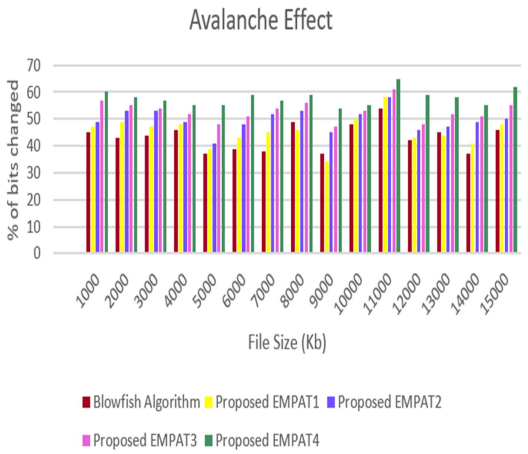

Figure 4.4 Comparison of Avalanche Effect between Blowfish, Proposed EMPAT1, EMPAT2, EMPAT3 and EMPAT4 methods.

When more than fifty percent of the output bits have changed, the algorithm produces a good avalanche effect. The total average measure of 58% indicates that the proposed EMPAT4 algorithms has a good avalanche effect. However, the total average measures for techniques Blowfish, proposed EMKB1, EMKB2, EMKB3 and EMKB4 43.33%, 45.46%, 49.66%,52.93% and 57.86% respectively.

5. CONCLUSION AND FUTURE ENHANCEMENT

This study examined and improved the encryption using the modified version of blowfish with Exchange and Merge Key Bit Algorithm (EMPAT). Although exchange and merge bit pattern were introduced before fed data into S-Boxes, which was initially positioned to avoid symmetry, the update kept the EMPAT's original structure, procedure, and usage of four swapping S-boxes. Time efficiency and the avalanche effect were used to examine the level of security derivation scenario. After comparing the original blowfish algorithm with the combinations of EMPAT's derivation processes, it was found that the EMPAT4 increased security enhancing the avalanche effect. Additionally, the results demonstrated that the EMPAT4 change allows highest security level. EMPAT4 addresses the limitations of the original Blowfish algorithm design by incorporating modern cryptographic techniques and increasing the security. With its enhanced security features and adaptability, it remains a robust choice for secure data encryption in the digital age. Further research and implementation will help refine the algorithm and expand its applications across diverse domains. Therefore, this proposed work offers the possibility of combining EMPAT with algorithms to improve data security so that hackers are unable to easily get past it. The proposed EMPAT will be harder to find and is safer compared with the original blowfish.In order to improve symmetric key encryption for safe communication systems and contemporary cybersecurity applications, this study offers a novel dynamic S-box transformation technique.

Furthermore, the study assesses resistance primarily against linear and differential cryptanalysis; other sophisticated attack models, like side-channel or quantum attacks, are not examined. Even if the suggested EMPAT framework increases encryption security, large-scale real-time systems with stringent resource restrictions might need more optimization.

REFERENCES

[1] Agrawal, Monika, and Pradeep Mishra. "A modified approach for symmetric key cryptography based on blowfish algorithm." International Journal of Engineering and Advanced Technology (IJEAT) 1.6 (2012): 79-83.

[2] Manku, Saikumar, and K. Vasanth. "Blowfish encryption algorithm for information security." ARPN journal of engineering and applied sciences 10.10 (2015): 4717-4719.

[3] Poonia, Vaibhav, and Narendra Singh Yadav. "Analysis of modified Blowfish Algorithm in different cases with various parameters." 2015 International Conference on Advanced Computing and Communication Systems. IEEE, 2015.

[4] Corpuz, Reynaldo R., Bobby D. Gerardo, and Ruji P. Medina. "A modified approach of Blowfish algorithm based on S-box permutation using shuffle algorithm." Proceedings of the 2018 VII International Conference on Network, Communication and Computing. 2018.

[5] Muin, Muhammad Abdul, et al. "Performance Comparison Between AES256-Blowfish and Blowfish-AES256 Combinations". 2018 5th International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE), 2018. https://doi.org/10.1109/icitacee.2018.8576929

[6] Manikandan, G., et al. "A Hybrid Technique for Enhancing Data Security". 2018. http://eprints.rclis.org/33253/

[7] Quilala, Theda Flare Ginoy, and Rogel L. Quilala. "Modified Blowfish algorithm analysis using derivation cases." Bulletin of Electrical Engineering and Informatics 10.4 (2021): 2192-2200.

[8] S. Sharma, K. N. Patel and A. Siddhath Jha, "Cryptography Using Blowfish Algorithm," 2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICAC3N), Greater Noida, India, 2021, pp. 1375-1377, http://doi: 10.1109/ICAC3N53548.2021.9725661.

[9] Hashim, Ashwaq T., Ammar H. Jassem, and Suhad A. Ali. "A novel design of Blowfish algorithm for image security." Journal of Physics: Conference Series. Vol. 1818. No. 1. IOP Publishing, 2021. Jurnal Komputer, Informasi dan Teknologi 2.2 (2022): 605-612.

[10] S. E. Alaojan and A. H. Alwattar, "A Modified Blowfish Algorithm to Secure Data in Cloud," 2022 International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT), Ankara, Turkey, 2022, pp. 218-222, https://doi: 10.1109/ISMSIT56059.2022.9932817

[11] Qader, Raghad Abdul Hadi Abdul, Auday H. Al-Wattar "A Review of the Blowfish Algorithm Modifications in Terms of Execution Time and Security". Technium: Romanian Journal of Applied Sciences and Technology, 2022. https://doi.org/10.47577/technium.v4i9.7452

[12] Saputra, Ronaldo Praja, Jusuf Wahyudi, and Juju Jumadi. "Comparative analysis of the blowfish algorithm and the des algorithm in the document file encryption and decryption process." Jurnal Komputer, Informasi dan Teknologi 2.2 (2022): 605-612.

[13] Nagamunthala, Mohan, and Ramakrishnan Manjula. "Implementation of a Hybrid Triple-Data Encryption Standard and Blowfish Algorithms for Enhancing Image Security in Cloud Environment." Journal of Computer and Communications 11.10 (2023): 135-149. https://doi.org/10.4236/jcc.2023.1110009

[14] Kumar, Sunil, Dilip Kumar "Performance evaluation and design of B-128 modified Blowfish algorithm". Security and Privacy, vol. 6, no. 5, 2023. https://doi.org/10.1002/spy2.307

[15] Zied, Hamed Shawky, et al. "An Optimized Implementation of the Blowfish Encryption Algorithm". 2024 International Telecommunications Conference (ITC-Egypt), 2024. https://doi.org/10.1109/itc-egypt61547.2024.10620572

[16] Harinath, Depavath, et al. "Blowfish Algorithm-An Efficient Data Encryption Technique to Ensure Data Confidentiality.", Journal of Engineering and Technology Management.pp. 72 (2024).